



사용 설명서

AWS 데이터베이스 마이그레이션 서비스



AWS 데이터베이스 마이그레이션 서비스: 사용 설명서

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon의 상표 및 트레이드 드레스는 Amazon 외 제품 또는 서비스와 함께, Amazon 브랜드 이미지를 떨어뜨리거나 고객에게 혼동을 일으킬 수 있는 방식으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계와 관계없이 해당 소유자의 자산입니다.

Table of Contents

AWS Database Migration Service이란 무엇인가요?	1
AWS DMS에서 수행하는 마이그레이션 작업	2
용어 및 개념	4
의 상위 수준 보기 AWS DMS	4
구성 요소	6
소스	12
데이터 마이그레이션에 사용할 수 있는 소스	12
DMS Fleet Advisor가 지원하는 소스	16
DMS Schema Conversion이 지원하는 소스	16
DMS 동종 데이터 마이그레이션이 지원하는 소스	17
대상	18
대상 마이그레이션에 사용할 수 있는 대상	18
DMS Fleet Advisor가 지원하는 대상	21
DMS Schema Conversion이 지원하는 대상	21
DMS 동종 데이터 마이그레이션이 지원하는 대상	22
Amazon 리소스 이름	22
다른 AWS 서비스 사용	24
에 대한 지원 CloudFormation	25
시작하기	26
추가 리소스	26
설정	27
에 가입AWS 계정	27
관리자 액세스 권한이 있는 사용자 생성	27
사전 조건	29
VPC 생성	30
Amazon RDS 파라미터 그룹 생성	31
소스 Amazon RDS 데이터베이스 생성	32
대상 Amazon RDS 데이터베이스 생성	34
Amazon EC2 클라이언트 생성	34
소스 데이터베이스 채우기	36
스키마 마이그레이션	37
복제	39
1단계: 복제 인스턴스 생성	39
2단계: 소스 및 대상 엔드포인트 지정	41

3단계: 태스크 생성 및 데이터 마이그레이션	43
4단계: 복제 테스트	45
5단계: AWS DMS 리소스 정리	47
AWS DMS Fleet Advisor 지원 종료	49
AWS DMS Fleet Advisor 지원 종료 세부 정보	49
지원 종료 계획	49
FAQ	49
마이그레이션을 위한 데이터베이스 검색	51
지원되는 AWS 리전	52
DMS Fleet Advisor 이해	54
필요한 리소스 생성	55
CloudFormation을 사용하여 리소스 구성	55
에서 Amazon S3 및 IAM 구성 AWS Management Console	58
데이터베이스 사용자 생성	64
데이터베이스 사용자 사용	65
MySQL을 사용하여 데이터베이스 사용자 생성	65
Oracle을 사용하여 데이터베이스 사용자 생성	66
PostgreSQL을 사용하여 데이터베이스 사용자 생성	67
SQL Server를 사용하여 데이터베이스 사용자 생성	67
데이터베이스 사용자 삭제	69
데이터 수집기	71
권한	72
데이터 수집기 생성	73
데이터 수집기 설치	75
OS 및 데이터베이스 서버 검색	78
모니터링되는 객체 관리	82
SSL 사용	84
데이터 수집	85
문제 해결	91
인벤토리	94
데이터베이스 인벤토리 사용	95
스키마 인벤토리 사용	97
대상 권장 사항	98
대상 인스턴스	99
DMS Fleet Advisor는 대상 사양을 어떻게 결정합니까?	100
대상 권장 사항 생성	101

권장 사항 세부 정보	103
대상 권장 사항 내보내기	104
마이그레이션 제한 사항	106
문제 해결	122
제한 사항	124
데이터베이스 스키마 변환	125
지원되는 AWS 리전	126
기능	128
제한 사항	129
시작하기	130
사전 조건	131
인스턴스 프로파일 생성	139
데이터 공급자 구성	139
마이그레이션 프로젝트 생성	141
평가 보고서 생성	141
소스 코드 변환	142
변환된 코드 적용	143
정리 및 문제 해결	143
네트워크 설정	144
단일 VPC 구성	145
여러 VPC 구성	145
Direct Connect 또는 VPN 사용	145
인터넷 연결 사용	146
소스 데이터 공급자 생성	146
SQL Server를 소스로 사용	147
Oracle을 소스로 사용	148
Oracle Data Warehouse를 소스로 사용	149
PostgreSQL을 소스로 사용	152
MySQL을 소스로 사용	152
Linux, UNIX 및 Windows용 Db2를 소스로 사용	153
Db2 for z/OS를 소스로 사용	155
SAP ASE(Sybase ASE) 데이터베이스 사용	156
대상 데이터 공급자 생성 및 설정	157
MySQL을 대상으로 사용	158
PostgreSQL을 대상으로 사용	160
Amazon Redshift 대상으로 사용	160

대상으로 Amazon RDS for Db2 데이터베이스 사용	162
가상 데이터 공급자	162
가상 데이터 공급자 생성	162
가상 데이터 공급자 사용량	162
가상 데이터 공급자에서 실제 데이터 공급자로 전환	163
마이그레이션 프로젝트 관리	164
마이그레이션 프로젝트 설정 지정	165
AWS DMS Schema Conversion에 대한 액세스 로그	166
데이터베이스 마이그레이션 평가 보고서	167
평가 보고서 생성	167
평가 보고서 보기	168
평가 보고서 저장	169
스키마 변환	171
변환 규칙 설정	172
데이터베이스 스키마 변환: step-by-step 가이드	174
생성형 AI를 사용하여 데이터베이스 객체 변환	178
스키마 변환 설정 지정	205
데이터베이스 스키마 새로 고침	214
스키마 저장 및 적용	215
Java 애플리케이션의 임베디드 SQL 변환	216
확장 팩 사용	217
IAM에서 API로 매핑	218
동종 데이터베이스 마이그레이션	220
지원됨 AWS 리전	221
기능	223
제한 사항	224
개요	224
설정	225
IAM 리소스 생성	226
네트워크 설정	228
VPC 피어링 네트워크 구성	231
소스 데이터 공급자 생성	235
소스로 MySQL 또는 MariaDB 사용	235
PostgreSQL을 소스로 사용	239
MongoDB 또는 Amazon DocumentDB를 소스로 사용	242
대상 데이터베이스 생성 및 설정	245

대상으로 MySQL 또는 MariaDB 사용	246
PostgreSQL을 대상으로 사용	247
Amazon DocumentDB를 대상으로 사용	249
데이터 마이그레이션	249
데이터 마이그레이션 생성	250
선택 규칙	252
데이터 마이그레이션 관리	255
데이터 마이그레이션을 모니터링	256
마이그레이션 상태	258
MySQL에서 데이터 마이그레이션	259
PostgreSQL에서 데이터 마이그레이션	261
MongoDB에서 데이터 마이그레이션	264
대상 테이블 준비 모드	265
문제 해결	266
데이터 마이그레이션 생성	266
데이터 마이그레이션 시작	266
연결 문제	267
PostgreSQL에서 뷰를 테이블로 마이그레이션합니다.	267
마이그레이션 프로젝트 작업	268
서브넷 그룹 생성	269
인스턴스 프로파일 생성	269
데이터 공급자 생성	271
마이그레이션 프로젝트 생성	273
마이그레이션 프로젝트 관리	274
AWS 자체 관리형 데이터베이스 소스를 위한 제로 ETL 통합	276
자체 관리형 데이터베이스 소스에서 제로 ETL 통합이 작동하는 방식	276
통합을 생성하기 전에 소스 데이터베이스 구성	277
제로 ETL 통합을 위한 IAM 권한 및 암호화 설정	278
사전 조건	278
KMS 키 생성	279
KMS 키 정책 구성	279
IAM 사용자 생성 및 AWS CLI 구성	281
IAM 정책 생성 및 연결	281
AWS CLI 프로파일 구성	284
Redshift 리소스 정책 생성	285
예: 암호화와 제로 ETL 통합 생성	286

보안 모범 사례	287
모범 사례	289
에 대한 마이그레이션 계획 AWS Database Migration Service	289
스키마 변환	291
AWS DMS 설명서 검토	291
개념 증명 실행	291
성능 개선	292
자체 온프레미스 이름 서버 사용	296
에서 Amazon Route 53 Resolver 사용 AWS DMS	297
대용량 이진 객체(LOB) 마이그레이션	298
제한적 LOB 모드 사용	298
개선된 LOB 성능	299
행 필터링을 사용하여 큰 테이블 마이그레이션 시 성능 향상	302
지속적 복제	303
소스 데이터베이스의 로드 감소	304
대상 데이터베이스에서 병목 현상 감소	304
데이터 유효성 검사 사용	304
지표 모니터링	305
이벤트	305
작업 로그 사용	306
Time Travel을 통한 복제 문제 해결	306
Oracle 대상에서 사용자 및 스키마 변경	307
Oracle 대상에 대한 테이블 및 인덱스 테이블스페이스 변경	307
복제 인스턴스 업그레이드	308
마이그레이션 비용 이해	309
AWS DMS 서버리스 작업	310
DMS 서버리스 구성 요소	311
지원 엔드포인트	314
서버리스 애플리케이션 생성	315
AWS DMS 서버리스 복제 수정	317
컴퓨팅 구성	321
AWS DMS 서버리스의 오토 스케일링 이해	322
AWS DMS 서버리스 복제 모니터링	323
확장 전체 로드 처리량 Amazon Redshift	327
AWS DMS 서버리스의 스토리지 Autoscaling 이해	328
서버리스의 제한 사항	329

서버리스 마이그레이션 전 평가	330
KMS 키를 사용하여 파일 암호화	331
제한 사항	331
복제 인스턴스 작업	333
복제 인스턴스 유형 선택	338
사용할 인스턴스 클래스 결정	342
무제한 모드 버스트 가능 인스턴스	343
복제 인스턴스 크기 조정	344
고려해야 할 요소	345
일반적인 문제	346
모범 사례	346
복제 엔진 버전	347
콘솔을 사용하여 엔진 버전 업그레이드	347
를 사용하여 엔진 버전 업그레이드 AWS CLI	348
퍼블릭 및 프라이빗 복제 인스턴스	349
IP 주소 지정 및 네트워크 유형	349
복제 인스턴스용으로 네트워크 설정	351
데이터베이스 마이그레이션을 위한 네트워크 구성	351
복제 서브넷 그룹 생성	361
DNS를 사용하여 도메인 엔드포인트 확인	362
암호화 키 설정	362
복제 인스턴스 생성	363
복제 인스턴스 수정	368
복제 인스턴스 수정 모범 사례	372
복제 인스턴스 재부팅	372
복제 인스턴스 삭제	375
DMS 유지 관리 기간	377
유지 관리가 기존 마이그레이션 작업에 미치는 영향	377
유지 관리 기간 설정 변경	377
엔드포인트	380
소스 및 대상 엔드포인트 생성	380
에서 Amazon RDS 엔드포인트에 대한 IAM 인증 사용 AWS DMS	385
데이터 마이그레이션용 소스	389
Oracle을 소스로 사용	390
SQL Server를 소스로 사용	454
Azure SQL 데이터베이스를 소스로 사용	506

Microsoft Azure SQL 관리형 인스턴스를 소스로 사용	507
Azure Database for PostgreSQL을 소스로 사용	507
Azure Database for MySQL을 소스로 사용	509
OCI MySQL Heatwave를 소스로 사용	509
Google Cloud for MySQL을 소스로 사용	510
Google Cloud for PostgreSQL을 소스로 사용	510
PostgreSQL을 소스로 사용	512
MySQL을 소스로 사용	551
SAP ASE를 소스로 사용	566
MongoDB를 소스로 사용	575
Amazon DocumentDB를 소스로 사용	593
Amazon S3를 소스로 사용	609
IBM DB2 LUW를 소스로 사용	623
IBM DB2 for z/OS를 소스로 사용	631
마이그레이션에 적합한 대상	671
Oracle을 대상으로 사용	673
SQL Server를 대상으로 사용	683
PostgreSQL을 대상으로 사용	688
MySQL을 대상으로 사용	701
Amazon Redshift 대상으로 사용	708
SAP ASE를 대상으로 사용	733
Amazon S3를 대상으로 사용	735
대상으로 Amazon DynamoDB 사용	784
Amazon Kinesis Data Streams를 대상으로 사용	805
Apache Kafka를 대상으로 사용	824
OpenSearch 대상으로 사용	851
Amazon DocumentDB를 대상으로 사용	857
Amazon Neptune을 대상으로 사용	870
Redis OSS를 대상으로 사용	886
Babelfish를 대상으로 사용	893
Amazon Timestream을 대상으로 사용	901
Db2를 대상으로 사용	912
데이터 마이그레이션에 사용할 수 있는 VPC 엔드포인트	913
일반적인 AWS DMS 사전 조건	914
AWS Secrets Manager를 사용하여 Amazon VPC 엔드포인트 설정	914
Amazon S3를 사용하여 Amazon VPC 엔드포인트 설정	917

Amazon DynamoDB용 Amazon VPC 엔드포인트 설정	918
Amazon Kinesis용 Amazon VPC 엔드포인트 설정	919
Amazon Redshift용 Amazon VPC 엔드포인트 설정	919
Amazon OpenSearch Service용 Amazon VPC 엔드포인트 설정	919
DMS 복제 인스턴스(프로비저닝 및 서버리스), 서브넷 그룹 및 엔드포인트	919
AWS DMS 버전 3.4.7 이상으로 마이그레이션할 때 영향을 받는 사람은 누구입니까?	923
AWS DMS 버전 3.4.7 이상으로 마이그레이션할 때 영향을 받지 않는 사람은 누구입니까? ...	923
AWS DMS 버전 3.4.7 이상으로 마이그레이션 준비	923
지원되는 DDL 문	924
제한 사항 및 고려 사항	925
고급 엔드포인트 구성	925
에 대한 VPC 피어링 구성입니다 AWS DMS.	925
에 대한 보안 그룹 구성 AWS DMS	928
에 대한 네트워크 액세스 제어 목록(NACL) 구성 AWS DMS	930
AWS DMS 보안 암호 관리자 VPC 엔드포인트 구성	931
추가 고려 사항	933
작업	935
작업 생성	939
작업 설정	946
LOB 지원 설정	999
여러 작업 생성	1003
지속적 복제 작업	1003
CDC 시작 지점에서 복제 시작	1005
양방향 복제 수행	1011
작업 수정	1015
태스크 이동	1015
작업 중 테이블 다시 로드	1016
AWS Management Console	1017
테이블 매핑	1018
콘솔에서 테이블 선택 및 변환 규칙 지정	1019
JSON을 사용하여 테이블 선택 및 변환 지정	1023
선택 규칙 및 작업	1025
테이블 매핑의 와일드카드	1031
변환 규칙 및 작업	1032
변환 규칙 표현식을 사용하여 열 내용 정의	1055
테이블 및 컬렉션 설정 규칙과 작업	1069

데이터 마스킹을 사용하여 민감한 정보 숨기기	1098
소스 필터 사용	1102
필터 적용	1103
시간과 날짜를 기준으로 필터링	1109
마이그레이션 전 평가 활성화 및 활용	1110
사전 조건	1111
평가 실행 지정, 시작, 보기	1114
개별 평가	1119
데이터 형식 평가 시작 및 보기	1190
평가 실행의 문제 해결	1194
보충 데이터 지정	1196
작업 모니터링	1197
작업 상태	1199
작업 중 테이블 상태	1201
Amazon CloudWatch를 사용한 복제 태스크 모니터링	1202
AWS Database Migration Service 지표	1204
복제 인스턴스 지표	1207
복제 작업 지표	1209
AWS DMS 로그 보기 및 관리	1212
를 사용하여 AWS DMS API 호출 로깅 AWS CloudTrail	1214
AWS DMS CloudTrail의 정보	1214
AWS DMS 로그 파일 항목 이해	1215
컨텍스트 로깅	1219
객체 유형	1219
로깅 예제	1220
제한 사항	1222
향상된 모니터링 대시보드	1222
개요	1223
향상된 모니터링 대시보드에서 지표 보기	1223
향상된 모니터링 대시보드 보기	1223
향상된 모니터링 지표 보존	1227
AWS DMS 이벤트 보기	1227
EventBridge 이벤트 사용	1231
에 Amazon EventBridge 이벤트 규칙 사용 AWS DMS	1232
AWS DMS 이벤트 범주 및 이벤트 메시지	1233
복제 인스턴스 이벤트 메시지	1233

복제 태스크 이벤트 메시지	1237
복제 이벤트 메시지	1239
Amazon SNS 이벤트 작업	1241
이벤트 구독을 Amazon EventBridge로 이전	1241
Amazon SNS 이벤트 및 알림 작업	1242
AWS SNS 알림을 위한 DMS 이벤트 범주 및 이벤트 메시지	1243
SNS를 사용하여 AWS DMS 이벤트 알림 구독	1247
사용 AWS Management Console	1248
SNS 주제의 액세스 정책 검증	1250
데이터 유효성 검사	1252
복제 작업 통계	1253
Amazon CloudWatch를 사용한 복제 태스크 통계	1255
작업 중 테이블 다시 검증	1256
AWS Management Console	1256
JSON 편집기를 사용하여 검증 규칙 수정	1257
검증 전용 태스크	1258
전체 로드 검증 전용	1258
CDC 검증 전용	1259
검증 전용 사용 사례	1259
문제 해결	1260
awsdms_validation_failures_v2 제어 테이블	1261
Redshift 검증 성능	1263
에 대한 향상된 데이터 검증 AWS Database Migration Service	1263
사전 조건	1264
향상된 데이터 검증 제한 사항	1264
제한 사항	1265
S3 유효성 검사	1266
사전 조건	1267
권한	1267
제한 사항	1269
검증 전용 태스크	1270
AWS DMS 데이터 재동기화	1270
제한 사항	1271
예약 및 타이밍	1272
사용 사례	1272
모범 사례	1273

데이터 재동기화 구성 및 예제	1275
검증 및 문제 해결	1276
Cron 표현식 규칙	1278
리소스에 태그 지정	1280
API	1281
보안	1284
데이터 보호	1286
데이터 암호화	1286
인터넷워크 트래픽 개인 정보	1287
DMS Fleet Advisor의 데이터 보호	1287
데이터 저장 옵트아웃	1288
교차 리전 추론	1289
Identity and Access Management(IAM)	1293
대상	1293
ID를 통한 인증	1294
정책을 사용하여 액세스 관리	1295
AWS Database Migration Service 에서 IAM을 사용하는 방법	1296
자격 증명 기반 정책 예	1303
리소스 기반 정책 예제	1311
보안 암호를 사용하여 리소스에 액세스하기	1316
서비스 연결 역할 사용	1325
문제 해결	1330
필요한 IAM 권한	1333
AWS DMS 콘솔, CLI 및 API에 대한 IAM 역할	1338
교차 서비스 혼동된 대리인 방지	1344
AWS 관리형 정책	1347
규정 준수 확인	1360
복원성	1362
인프라 보안	1363
세분화된 액세스 제어	1366
리소스 이름을 사용하여 액세스 제어	1366
태그를 사용하여 액세스 제어	1369
마이그레이션을 위한 암호화	1376
암호화 키 설정	1378
네트워크 보안	1381
SSL 사용	1383

에서 SSL 사용 시 제한 사항 AWS DMS	1385
인증서 관리	1385
MySQL 호환, PostgreSQL 또는 SQL Server 엔드포인트에서 SSL 활성화	1386
데이터베이스 암호 변경	1389
Kerberos 인증 사용	1389
AWS DMS Kerberos 인증 아키텍처 개요	1389
에서 Kerberos 인증 사용에 대한 제한 사항 AWS DMS	1390
사전 조건	1391
AWS DMS 복제 인스턴스에서 Kerberos 지원 활성화	1392
소스 엔드포인트에서 Kerberos 지원 활성화	1393
소스 엔드포인트 테스트	1394
한도	1395
에 대한 리소스 할당량AWS Database Migration Service	1395
API 요청 제한에 대한 이해	1396
문제 해결 및 진단 지원	1398
마이그레이션 태스크가 느리게 실행됨	1399
태스크 상태 표시줄이 움직이지 않음	1400
태스크가 완료되었지만 아무것도 마이그레이션되지 않음	1400
외부 키와 보조 인덱스 누락	1400
AWS DMS 는 CloudWatch 로그를 생성하지 않습니다.	1401
Amazon RDS에 연결할 때 문제가 발생함	1401
오류 메시지: Incorrect thread connection string: incorrect thread value 0	1402
네트워킹 문제 발생	1402
전체 로드 후 CDC 중단	1403
태스크 재시작 시 프라이머리 키 위반 오류 발생	1403
스키마 초기 로드 실패	1403
알 수 없는 오류로 인해 태스크 실패	1403
시작부터 작업 재시작 시 테이블 로드	1404
태스크당 테이블 수로 인해 문제 발생	1404
LOB 열에 프라이머리 키가 생성되면 태스크가 실패함	1404
프라이머리 키가 없는 대상 테이블에서 중복 레코드 발생	1404
소스 엔드포인트가 예약된 IP 범위에 속함	1404
Amazon Athena 쿼리에서 타임스탬프가 깨져 보임	1405
Oracle 관련 문제 해결	1405
보기에서 데이터 가져오기	1406
12c에서 LOB 마이그레이션	1406

Oracle LogMiner와 Binary Reader 간 전환	1406
오류: Oracle CDC stopped 122301 Oracle CDC maximum retry counter exceeded.	1407
보충 로깅을 Oracle 소스 엔드포인트에 자동으로 추가합니다.	1407
LOB 변경 사항이 캡처되지 않음	1408
오류: ORA-12899: 값이 <i>column-name</i> 열에 대해 너무 큼	1408
NUMBER 데이터 형식 오해	1408
전체 로드 중 레코드 누락	1408
테이블 오류	1409
오류: Cannot retrieve Oracle archived Redo log destination ids	1409
Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능 평가	1409
LOB 데이터를 가져오지 못했습니다	1411
MySQL 관련 문제 해결	1412
이진 로깅 비활성화로 인해 Amazon RDS DB 인스턴스 엔드포인트에서 CDC 작업 실패	1413
작업 중에 대상 MySQL 인스턴스 연결이 끊김	1413
MySQL 호환 엔드포인트에 자동 커밋 추가	1414
대상 MySQL 호환 엔드포인트에서 외래 키 비활성화	1414
물음표로 대체된 문자	1415
'Bad event' 로그 항목	1415
MySQL 5.5로 변경 데이터 캡처	1415
Amazon RDS DB 인스턴스를 위한 이진 로그 보존 기간 증가	1415
로그 메시지: 소스 데이터베이스의 일부 변경 사항은 대상 데이터베이스에 적용될 때 영향이 전혀 없었습니다.	1416
오류: Identifier too long	1416
오류: 지원되지 않는 문자 집합으로 인해 필드 데이터 변환 실패	1416
오류: Codepage 1252 to UTF8 [120112] A field data conversion failed	1417
인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음	1417
PostgreSQL 관련 문제 해결	1419
잘리는 JSON 데이터 형식	1420
사용자 정의 데이터 형식 열이 올바르게 마이그레이션되지 않음	1420
오류: 생성하도록 선택된 스키마가 없음	1420
테이블 삭제와 업데이트가 CDC를 사용하여 복제되지 않음	1421
자르기 문이 전파되지 않고 있음	1421
PostgreSQL에서 DDL이 캡처되지 않음	1421
DDL 캡처를 위해 데이터베이스 객체가 생성되는 스키마 선택	1421
PostgreSQL로 마이그레이션한 후 Oracle 테이블 누락	1421

ETL 워크로드와 같은 긴 트랜잭션 중에 ReplicationSlotDiskUsage가 증가하고 restart_lsn의 진행이 중지됨	1422
소스로 보기를 사용한 작업에서 복사된 행이 없음	1422
"UTF8" 인코딩을 위한 잘못된 바이트 시퀀스	1422
Microsoft SQL Server 관련 문제 해결	1423
RDS for SQL Server가 보조로 장애 조치된 후 지속적인 복제 실패	1423
SQL Server 데이터베이스에서 변경 사항을 캡처하는 동안 발생하는 오류	1424
자격 증명 열 누락	1424
오류: SQL Server doesn't support publications	1424
대상에 변경 사항이 표시되지 않음	1424
파티션 간에 매핑된 비균일 테이블	1425
오류: 트랜잭션을 처리하는 동안 잘못된 봉투, 잘못된 데이터 컨텍스트/LCX 코드로 인해 CDC 작업이 실패했습니다	1426
Amazon Redshift 관련 문제 해결	1426
다른 AWS 리전의 Amazon Redshift 클러스터에 로드	1426
오류: 관계 "awsdms_apply_exceptions"가 이미 존재합니다	1426
awsdms_changes"로 시작하는 테이블 이름이 있는 오류	1427
dms.awsdms_changes0000000000XXXX와 같은 이름과 함께 클러스터에 테이블 표시	1427
Amazon Redshift 사용에 필요한 권한	1427
Amazon Aurora MySQL 관련 문제 해결	1427
오류: CHARACTER SET UTF8 fields terminated by ',' enclosed by '"' lines terminated by '\n'	1428
SAP ASE 관련 문제 해결	1428
오류: LOB columns have NULL values when source has a composite unique index with NULL values	1428
IBM DB2 관련 문제 해결	1429
오류: Resume from timestamp is not supported Task	1429
테이블이 "Failed to build 'where' 문" 오류와 함께 테이블을 일시 중지했습니다.	1429
지연 시간 문제 해결	1430
CDC 지연 시간 유형	1430
CDC 지연 시간이 발생하는 일반적인 원인	1431
지연 시간 문제 해결	1434
진단 지원 스크립트 작업	1448
Oracle 지원 스크립트	1450
SQL Server 지원 스크립트	1453
MySQL과 호환되는 지원 스크립트	1456

PostgreSQL 지원 스크립트	1458
진단 지원 AMI 작업	1461
새 AWS DMS 진단 Amazon EC2 인스턴스 시작	1462
IAM 역할 생성	1462
진단 테스트 실행	1463
다음 단계	1467
리전별 AMI ID	1467
AWS 시스템 패치 관리자	1468
레퍼런스	1469
AWS DMS 데이터 형식	1469
릴리스 노트	1472
AWS DMS 3.6.1 릴리스 정보	1473
AWS DMS 3.6.0 릴리스 정보	1477
AWS DMS 3.5.4 릴리스 정보	1479
AWS DMS 3.5.3 릴리스 정보	1487
AWS DMS 3.5.2 릴리스 정보	1497
AWS DMS 3.5.1 릴리스 정보	1504
스트리밍 대상에 대한 큰 숫자 값 처리 변경	1504
AWS DMS 3.5.0 베타 릴리스 정보	1519
AWS DMS 3.4.7 릴리스 정보	1524
AWS DMS 3.4.6 릴리스 정보	1533
AWS DMS 3.4.5 릴리스 정보	1539
AWS DMS 3.4.4 릴리스 정보	1542
AWS DMS 3.4.3 릴리스 정보	1544
AWS DMS 3.4.2 릴리스 정보	1546
AWS DMS 3.4.1 릴리스 정보	1548
AWS DMS 3.4.0 릴리스 정보	1549
AWS DMS 3.3.4 릴리스 정보	1550
AWS DMS 3.3.3 릴리스 정보	1551
문서 기록	1553
AWS 용어집	1559
.....	mdlx

AWS Database Migration Service이란 무엇인가요?

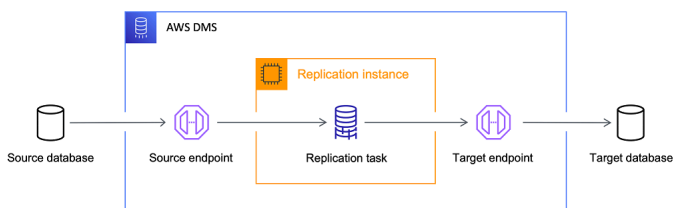
AWS Database Migration Service(AWS DMS)는 관계형 데이터베이스, 데이터 웨어하우스, NoSQL 데이터베이스 및 기타 유형의 데이터 스토어를 마이그레이션할 수 있는 클라우드 서비스입니다. AWS DMS를 사용하여 데이터를 AWS 클라우드로 마이그레이션하거나, 클라우드와 온프레미스 설정 조합 간에 마이그레이션할 수 있습니다.

AWS DMS를 사용하면 소스 데이터 스토어를 검색하고, 소스 스키마를 변환하고, 데이터를 마이그레이션할 수 있습니다.

- DMS Fleet Advisor를 사용하여 소스 데이터 인프라를 검색할 수 있습니다. 이 서비스는 온프레미스 데이터베이스 및 분석 서버에서 데이터를 수집하고 AWS 클라우드로 마이그레이션할 수 있는 서버, 데이터베이스, 스키마의 인벤토리를 구축합니다.
- 다른 데이터베이스 엔진으로 마이그레이션하려면 DMS Schema Conversion을 사용하면 됩니다. 이 서비스는 소스 스키마를 자동으로 평가한 후 새로운 대상 엔진으로 변환합니다. 또는 AWS Schema Conversion Tool(AWS SCT)을 로컬 PC에 다운로드하여 소스 스키마를 변환할 수도 있습니다.
- 소스 스키마를 변환하고 변환된 코드를 대상 데이터베이스에 적용한 후, AWS DMS를 사용하여 데이터를 마이그레이션할 수 있습니다. 마이그레이션을 한 번 수행하거나, 지속적인 변경 사항을 복제하여 소스와 대상을 동기화 상태로 유지할 수 있습니다. AWS DMS는 AWS 클라우드의 일부이기 때문에 AWS 서비스에서 제공하는 비용 효율성, 빠른 시장 출시 속도, 보안, 유연성을 누릴 수 있습니다.

기본 수준에서 보자면 AWS DMS는 복제 소프트웨어를 실행하는 AWS 클라우드의 서버입니다. 소스 및 대상 연결을 생성하여 데이터를 추출할 위치와 데이터를 로드할 위치를 AWS DMS에 지시합니다. 그 다음, 이 서버에서 실행되는 태스크를 예약하여 데이터를 이동합니다. AWS DMS는 테이블 및 관련 프라이머리 키가 대상에 없는 경우 이러한 항목을 생성합니다. 원하는 경우 대상 테이블을 직접 생성할 수 있습니다. 또는 AWS Schema Conversion Tool(AWS SCT)을 사용하여 대상 테이블, 인덱스, 보기, 트리거 등을 일부 또는 모두 생성할 수 있습니다.

아래 다이어그램에 AWS DMS 프로세스가 나와 있습니다.



참조

- AWS DMS를 지원하는 AWS 리전 – AWS DMS를 지원하는 AWS 리전 지원에 대한 자세한 내용은 [AWS DMS 복제 인스턴스 작업](#) 섹션을 참조하세요.
- 데이터베이스 마이그레이션 비용 – 데이터베이스 마이그레이션 비용에 대한 자세한 내용은 [AWS Database Migration Service 요금 페이지](#) 섹션을 참조하세요.
- AWS DMS 기능 및 이점 – AWS DMS 기능 및 이점에 대한 내용은 [AWS Database Migration Service 기능](#) 섹션을 참조하세요.
- 사용 가능한 데이터베이스 옵션 – Amazon Web Services Services에서 사용할 수 있는 다양한 데이터베이스 옵션에 대해 자세히 알아보려면 [조직에 적합한 데이터베이스 선택](#)을 참조하세요.

AWS DMS에서 수행하는 마이그레이션 작업

AWS DMS는 마이그레이션 프로젝트와 관련된 까다롭거나 지루한 수많은 태스크를 대신 처리합니다.

- 기존 솔루션에서는 용량 분석을 수행하고, 하드웨어와 소프트웨어를 구매하고, 시스템을 설치 및 관리하고, 설치를 테스트 및 디버깅해야 합니다. AWS DMS는 마이그레이션에 필요한 모든 하드웨어 및 소프트웨어의 배포, 관리, 모니터링을 자동으로 관리합니다. AWS DMS 구성 프로세스를 시작한 지 몇 분 이내에 마이그레이션을 실행할 수 있습니다.
- AWS DMS를 사용하면 실제 워크로드에 맞춰 필요에 따라 마이그레이션 리소스를 스케일 업(또는 스케일 다운)할 수 있습니다. 예를 들어, 추가 스토리지가 필요하다고 판단되면 할당된 스토리지를 쉽게 늘리고, 대개 몇 분 내로 마이그레이션을 다시 시작할 수 있습니다.
- AWS DMS는 사용한 만큼만 지불하는 모델을 사용합니다. 선결제 구매 비용과 지속적인 유지 관리 비용이 부과되는 기존 라이선스 모델과 달리, AWS DMS 리소스는 사용하는 동안에만 비용을 지불하면 됩니다.
- AWS DMS는 하드웨어 및 소프트웨어, 소프트웨어 패치 적용, 오류 보고 등을 비롯하여 마이그레이션 서버를 지원하는 모든 인프라를 자동으로 관리합니다.
- AWS DMS는 자동 장애 조치를 제공합니다. 어떤 이유로든 기본 복제 서버에 장애가 발생할 경우, 서비스 중단이 거의 발생하지 않은 채로 백업 복제 서버가 작업을 이어받을 수 있습니다.
- AWS DMS Fleet Advisor는 데이터 인프라의 인벤토리를 자동으로 생성합니다. 마이그레이션 대상을 식별하고, 마이그레이션을 계획하는 데 도움이 되는 보고서를 생성합니다.
- AWS DMS Schema Conversion은 소스 데이터 공급자의 마이그레이션 복잡성을 자동으로 평가합니다. 또한 데이터베이스 스키마와 코드 객체를 대상 데이터베이스와 호환되는 형식으로 변환한 다음, 변환된 코드를 적용합니다.

- AWS DMS는 현재 실행 중인 엔진보다 더 비용 효율적인 최신 데이터베이스 엔진으로 전환하는 데 도움이 될 수 있습니다. 예를 들어, AWS DMS는 Amazon Relational Database Service(RDS) 또는 Amazon Aurora에서 제공하는 관리형 데이터베이스 서비스를 활용하는 데 도움이 될 수 있습니다. 또는 Amazon Redshift에서 제공하는 관리형 데이터 웨어하우스 서비스, Amazon DynamoDB와 같은 NoSQL 플랫폼 또는 Amazon Simple Storage Service(S3)와 같은 저비용 스토리지 플랫폼으로 전환할 수 있습니다. 반대로 기존 인프라에서 벗어나 동일한 데이터베이스 엔진을 계속 사용하려는 경우에도, AWS DMS는 이러한 프로세스를 지원합니다.
- AWS DMS는 오늘날 가장 널리 사용되는 거의 모든 DBMS 엔진을 소스 엔드포인트로 지원합니다. 자세한 내용은 [데이터 마이그레이션용 소스](#) 섹션을 참조하세요.
- AWS DMS는 사용 가능한 대상 엔진을 광범위하게 제공합니다. 자세한 내용은 [마이그레이션에 적합한 대상](#) 섹션을 참조하세요.
- 지원되는 데이터 소스에서 지원되는 데이터 대상으로 마이그레이션할 수 있습니다. AWS DMS는 지원되는 엔진 간의 완전한 이기종 데이터 마이그레이션을 지원합니다.
- AWS DMS는 데이터 마이그레이션의 보안을 보장합니다. 저장 데이터는 AWS Key Management Service(AWS KMS) 암호화로 암호화됩니다. 마이그레이션하는 동안 Secure Socket Layer(SSL)를 사용하여 소스에서 대상으로 데이터를 이동할 때 처리 중인 데이터를 암호화할 수 있습니다.

AWS DMS 용어 및 개념

AWS Database Migration Service (AWS DMS)는 소스 데이터 스토어에서 대상 데이터 스토어로 데이터를 마이그레이션하는 데 사용할 수 있는 웹 서비스입니다. 이러한 두 데이터 저장소는 엔드포인트라고 합니다. Oracle 데이터베이스에서 Oracle 데이터베이스로와 같이 동일한 데이터베이스 엔진을 사용하는 원본 및 대상 엔드포인트를 마이그레이션할 수 있습니다. Oracle 데이터베이스에서 PostgreSQL 데이터베이스로와 같이 다른 데이터베이스 엔진을 사용하는 원본 및 대상 엔드포인트를 마이그레이션할 수 있습니다. 유일한 사용 요구 AWS DMS 사항은 엔드포인트 중 하나가 AWS 서비스에 있어야 한다는 것입니다. AWS DMS 를 사용하여 온프레미스 데이터베이스에서 다른 온프레미스 데이터베이스로 마이그레이션할 수 없습니다.

데이터베이스 마이그레이션 비용에 대한 자세한 내용은 [AWS Database Migration Service 요금 페이지](#)를 참조하세요.

다음 주제를 사용하여 더 잘 이해할 수 있습니다 AWS DMS.

주제

- [의 상위 수준 보기 AWS DMS](#)
- [의 구성 요소 AWS DMS](#)
- [의 소스 AWS DMS](#)
- [의 대상 AWS DMS](#)
- [에 대한 Amazon 리소스 이름\(ARN\) 구성 AWS DMS](#)
- [다른 AWS 서비스와 AWS DMS 함께 사용](#)

의 상위 수준 보기 AWS DMS

데이터베이스 마이그레이션을 수행하기 위해서는 소스 데이터 스토어에 AWS DMS 연결하고, 소스 데이터를 읽고, 대상 데이터 스토어에서 사용할 데이터의 형식을 지정합니다. 그런 다음 데이터를 대상 데이터 저장소에 로드합니다. 이러한 처리 대다수는 메모리에서 나타나지만, 큰 트랜잭션에는 디스크로의 일부 버퍼링이 필요할 수 있습니다. 캐시된 트랜잭션과 로그 파일도 디스크에 기록됩니다.

상위 수준에서 사용할 때 다음을 AWS DMS 수행합니다.

- 네트워크 환경에서 마이그레이션에 적합한 데이터베이스를 검색합니다.
- 소스 데이터베이스 스키마와 대부분의 데이터베이스 코드 객체를 대상 데이터베이스와 호환되는 형식으로 자동 변환합니다.

- 복제 서버를 생성합니다.
- 데이터 저장소에 대한 연결 정보가 있는 소스 및 대상 엔드포인트를 생성합니다.
- 하나 이상의 마이그레이션 작업을 생성하여 원본과 대상 데이터 저장소 간 데이터를 마이그레이션합니다.

작업은 세 가지 주요 단계로 구성되어 있습니다.

- 기존 데이터 마이그레이션(전체 로드)
- 캐시된 변경 사항 적용
- 지속적 복제(변경 데이터 캡처)

소스의 기존 데이터가 대상으로 이동하는 전체 로드 마이그레이션 중에는 소스 데이터 스토어의 테이블에서 대상 데이터 스토어의 테이블로 데이터를 AWS DMS 로드합니다. 전체 로드가 진행되는 동안 로드 중인 테이블에 적용된 변경 사항은 복제 서버에서 캐시되고, 이것은 캐시된 변경 사항입니다. 해당 테이블의 전체 로드가 시작될 때까지 해당 테이블의 변경 사항을 캡처하지 AWS DMS 애플리케이션은 이에 유의해야 합니다. 즉, 변경 캡처가 시작되는 시점은 각 테이블별로 다릅니다.

지정된 테이블의 전체 로드가 완료되면 AWS DMS 즉시 해당 테이블에 캐시된 변경 사항을 적용하기 시작합니다. 테이블이 로드되고 캐시된 변경 사항이 적용되면 진행 중인 복제 단계의 트랜잭션으로 변경 사항을 수집하기 AWS DMS 시작합니다. 트랜잭션에 테이블이 아직 완전히 로드되지 않은 경우 변경 사항은 복제 인스턴스에 로컬로 저장됩니다. 가 캐시된 모든 변경 사항을 모든 테이블에 AWS DMS 적용한 후 테이블은 트랜잭션상 일관성을 유지합니다. 이 시점에서는 진행 중인 복제 단계로 AWS DMS 이동하여 변경 사항을 트랜잭션으로 적용합니다.

지속적 복제 단계가 시작되면, 트랜잭션의 백로그는 일반적으로 원본과 대상 데이터베이스 사이에서 일부 지연 시간을 유발합니다. 결국 마이그레이션은 이 트랜잭션 백로그를 통해 진행된 후 일정한 상태에 도달합니다. 이 시점에서 애플리케이션을 종료하고, 남은 트랜잭션을 대상에 적용하도록 허용하고, 애플리케이션을 불러오면 이제 대상 데이터베이스를 가리킵니다.

AWS DMS 는 데이터 마이그레이션을 수행하는 데 필요한 대상 스키마 객체를 생성합니다. AWS DMS 를 사용하여 미니멀한 접근 방식을 취하고 데이터를 효율적으로 마이그레이션하는 데 필요한 객체만 생성할 수 있습니다. 이 접근 방식을 사용하면 테이블, 기본 키 및 경우에 따라 고유한 인덱스를 AWS DMS 생성하지만 소스에서 데이터를 효율적으로 마이그레이션하는 데 필요하지 않은 다른 객체는 생성하지 않습니다.

또는 내에서 DMS Schema Conversion AWS DMS 을 사용하여 소스 데이터베이스 스키마와 대부분의 데이터베이스 코드 객체를 대상 데이터베이스와 호환되는 형식으로 자동으로 변환할 수 있습니다.

이 변환에는 테이블, 뷰, 저장 프로시저, 함수, 데이터 형식, 동의어 등이 포함됩니다. DMS Schema Conversion에서 자동으로 변환할 수 없는 모든 객체는 명확하게 표시됩니다. 마이그레이션을 완료하기 위해 이러한 객체를 수동으로 변환할 수 있습니다.

의 구성 요소 AWS DMS

이 섹션에서는의 내부 구성 요소와 이러한 구성 요소가 함께 작동하여 데이터 마이그레이션을 수행하는 AWS DMS 방법을 설명합니다. AWS DMS 의 기본 구성 요소를 이해하면 데이터를 더 효율적으로 마이그레이션할 수 있고 문제를 해결하거나 원인을 파악할 때 문제를 더 심도 있게 파악할 수 있습니다.

AWS DMS 마이그레이션은 마이그레이션할 데이터베이스 검색, 자동 스키마 변환, 복제 인스턴스, 소스 및 대상 엔드포인트, 복제 작업의 5가지 구성 요소로 구성됩니다. 에서 필요한 복제 인스턴스, 엔드포인트 및 작업을 생성하여 AWS DMS 마이그레이션을 생성합니다 AWS 리전.

데이터베이스 검색

DMS Fleet Advisor는 여러 데이터베이스 환경에서 데이터를 수집하여 데이터 인프라에 대한 인사이트를 제공합니다. DMS Fleet Advisor는 모든 컴퓨터에 설치할 필요 없이 하나 이상의 중앙 위치에서 온프레미스 데이터베이스 및 분석 서버의 데이터를 수집합니다. 현재 DMS Fleet Advisor는 Microsoft SQL Server, MySQL, Oracle 및 PostgreSQL 데이터베이스 서버를 지원합니다.

DMS Fleet Advisor는 네트워크에서 검색된 데이터를 기반으로 인벤토리를 작성합니다. 이 인벤토리를 검토하여 모니터링할 데이터베이스 서버 및 객체를 결정할 수 있습니다. 이러한 서버, 데이터베이스 및 스키마에 대한 세부 정보가 수집되면 의도한 데이터베이스 마이그레이션의 실행 가능성을 분석할 수 있습니다.

스키마 및 코드 마이그레이션

의 DMS Schema Conversion AWS DMS 을 사용하면 다양한 유형의 데이터베이스 간에 데이터베이스를 더 쉽게 마이그레이션할 수 있습니다. DMS Schema Conversion을 사용하여 소스 데이터 공급자의 마이그레이션 복잡성을 평가한 다음 이를 사용하여 데이터베이스 스키마 및 코드 객체를 변환할 수 있습니다. 그런 다음 변환된 코드를 대상 데이터베이스에 적용할 수 있습니다.

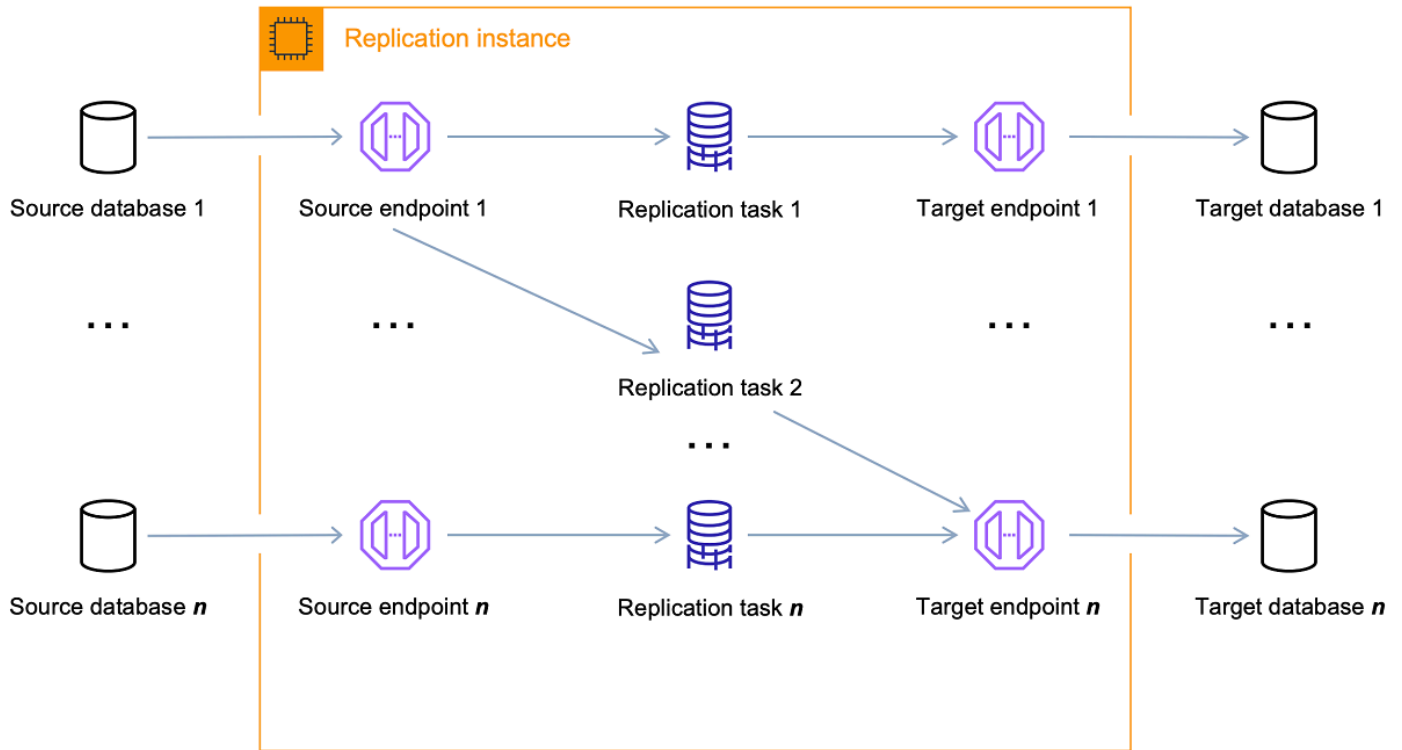
상위 수준에서 DMS Schema Conversion은 인스턴스 프로파일, 데이터 공급자, 마이그레이션 프로젝트 등 3가지 구성 요소를 사용하여 작동합니다. 인스턴스 프로파일은 네트워크 및 보안 설정을 지정합니다. 데이터 공급자는 데이터베이스 연결 자격 증명 정보를 저장합니다. 마이그레이션 프로젝트에는 데이터 공급자, 인스턴스 프로파일 및 마이그레이션 규칙이 포함됩니다.는 데이터 공급자와

인스턴스 프로파일을 AWS DMS 사용하여 데이터베이스 스키마 및 코드 객체를 변환하는 프로세스를 설계합니다.

복제 인스턴스

상위 수준에서 AWS DMS 복제 인스턴스는 단순히 하나 이상의 복제 작업을 호스팅하는 관리형 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스입니다.

다음 그림은 몇 가지 연결된 복제 작업을 실행하는 복제 인스턴스의 예시입니다.



단일 복제 인스턴스는 마이그레이션의 특성과 복제 서버의 용량에 따라 하나 이상의 복제 작업을 호스팅할 수 있습니다. 다양한 복제 인스턴스를 AWS DMS 제공하므로 사용 사례에 맞는 최적의 구성을 선택할 수 있습니다. 복제 인스턴스의 다양한 클래스에 대한 자세한 내용은 [마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택](#) 섹션을 참조하세요.

AWS DMS 는 Amazon EC2 인스턴스에 복제 인스턴스를 생성합니다. 소형 인스턴스 클래스 중에는 서비스를 테스트하거나 소규모 마이그레이션을 하기에 충분한 것이 있습니다. 마이그레이션에 대량의 테이블이 포함되어 있는 경우 또는 여러 동시 복제 작업을 실행해야 하는 경우, 대형 인스턴스 중 하나를 사용하는 것이 좋습니다. AWS DMS 는 상당량의 메모리와 CPU를 소모할 수 있으므로 이 접근 방식을 권장합니다.

선택하는 Amazon EC2 인스턴스 클래스에 따라 복제 인스턴스에는 50GB 또는 100GB의 데이터 스토리지가 포함되어 있습니다. 대개 이 정도 크기면 대부분 고객에게 충분합니다. 그러나 마이그

레이션에 대규모 트랜잭션 또는 대량 데이터 변경 사항이 포함되어 있다면 기본 스토리지 할당을 늘려야 할 수 있습니다. 변경 데이터 캡처(CDC)로 인해 데이터가 디스크에 쓰일 수 있는데, 이는 대상이 변경 사항을 쓰는 속도에 달려 있습니다. 로그 파일도 디스크에 기록되므로 로깅의 심각도가 높아지면 스토리지 사용량도 증가합니다.

AWS DMS 는 다중 AZ 배포를 사용하여 고가용성 및 장애 조치 지원을 제공할 수 있습니다. 다중 AZ 배포에서는 다른 가용 영역에 복제 인스턴스의 대기 복제본을 AWS DMS 자동으로 프로비저닝하고 유지 관리합니다. 기본 복제 인스턴스는 대기 복제본에 동기식으로 복제됩니다. 기본 복제 인스턴스에 장애가 발생하거나 무응답 상태가 되면 대기 복제본은 중단을 최소화하면서 실행 중이던 작업을 다시 시작합니다. 기본 복제본은 자신의 상태를 끊임없이 대기 복제본으로 복제하기 때문에 다중 AZ 배포는 약간의 성능 오버헤드를 발생시킵니다.

AWS DMS 복제 인스턴스에 대한 자세한 내용은 섹션을 참조하세요 [AWS DMS 복제 인스턴스 작업](#).

복제 인스턴스를 생성하고 관리하는 대신 AWS DMS Serverless를 사용하여 복제를 자동으로 AWS DMS 프로비저닝할 수 있습니다. 자세한 내용은 [AWS DMS 서버리스 작업](#) 단원을 참조하십시오.

엔드포인트

AWS DMS 는 엔드포인트를 사용하여 소스 또는 대상 데이터 스토어에 액세스합니다. 구체적인 연결 정보는 데이터 저장소에 따라 다르지만, 일반적으로 사용자는 엔드포인트 생성 시 다음과 같은 정보를 제공합니다.

- 엔드포인트 유형 소스 또는 대상.
- 엔진 유형 - Oracle 또는 PostgreSQL 같은 데이터베이스 엔진의 유형.
- 서버 이름 -가 연결할 AWS DMS 수 있는 서버 이름 또는 IP 주소입니다.
- 포트 - 데이터베이스 서버 연결에 사용되는 포트 번호.
- 암호화 - 연결을 암호화하는 데 SSL(보안 소켓 계층)이 사용되는 경우, SSL 모드.
- 자격 증명 - 필수 액세스 권한이 있는 계정의 사용자 이름 및 암호.

AWS DMS 콘솔을 사용하여 엔드포인트를 생성할 때 콘솔에서 엔드포인트 연결을 테스트해야 합니다. AWS DMS 태스크에서 엔드포인트를 사용하기 전에 테스트가 성공해야 합니다. 연결 정보와 마찬가지로 구체적인 테스트 기준은 엔진 유형에 따라 다릅니다. 일반적으로 AWS DMS에서는 주어진 서버 이름 및 포트에 데이터베이스가 있는지, 제공된 자격 증명을 마이그레이션을 수행하는 데 필요한 권한이 있는 데이터베이스에 연결하는 데 사용할 수 있는지 확인합니다. 연결 테스트가 성공하면는 나중에 작업 구성 중에 사용할 스키마 정보를 AWS DMS 다운로드하고 저장합니다. 스키마 정보에는 테이블 정의, 프라이머리 키 정의, 고유 키 정의 등이 포함될 수 있습니다.

복수의 복제 작업에서 단일 엔드포인트를 사용할 수 있습니다. 예를 들어, 동일한 소스 데이터베이스에서 호스팅되는 논리적으로 구분된 애플리케이션이 두 개 있는데, 사용자가 이를 서로 구분하여 마이그레이션하고자 하는 경우가 있을 수 있습니다. 이 경우 각 애플리케이션 테이블 집합에 대해 하나씩 두 개의 복제 작업을 생성합니다. 두 작업 모두에서 동일한 AWS DMS 엔드포인트를 사용할 수 있습니다.

엔드포인트 설정을 사용하여 엔드포인트의 동작을 사용자 지정할 수 있습니다. 엔드포인트 설정은 로깅 세부 정보, 파일 크기, 기타 파라미터 등 다양한 동작을 제어할 수 있습니다. 데이터 저장소 엔진 유형마다 사용 가능한 엔드포인트 설정이 다릅니다. 데이터 저장소의 소스 혹은 대상 섹션에서 각 데이터 저장소의 특정 엔드포인트 설정을 찾을 수 있습니다. 지원되는 원본 및 대상 데이터 저장소 목록은 [의 소스 AWS DMS](#) 및 [의 대상 AWS DMS](#) 섹션을 참조하세요.

AWS DMS 엔드포인트에 대한 자세한 내용은 섹션을 참조하세요 [AWS DMS 엔드포인트 작업](#).

복제 작업

AWS DMS 복제 작업을 사용하여 소스 엔드포인트에서 대상 엔드포인트로 데이터 세트를 이동합니다. 복제 작업 생성은 마이그레이션을 시작하기 전에 수행해야 할 최종 단계입니다.

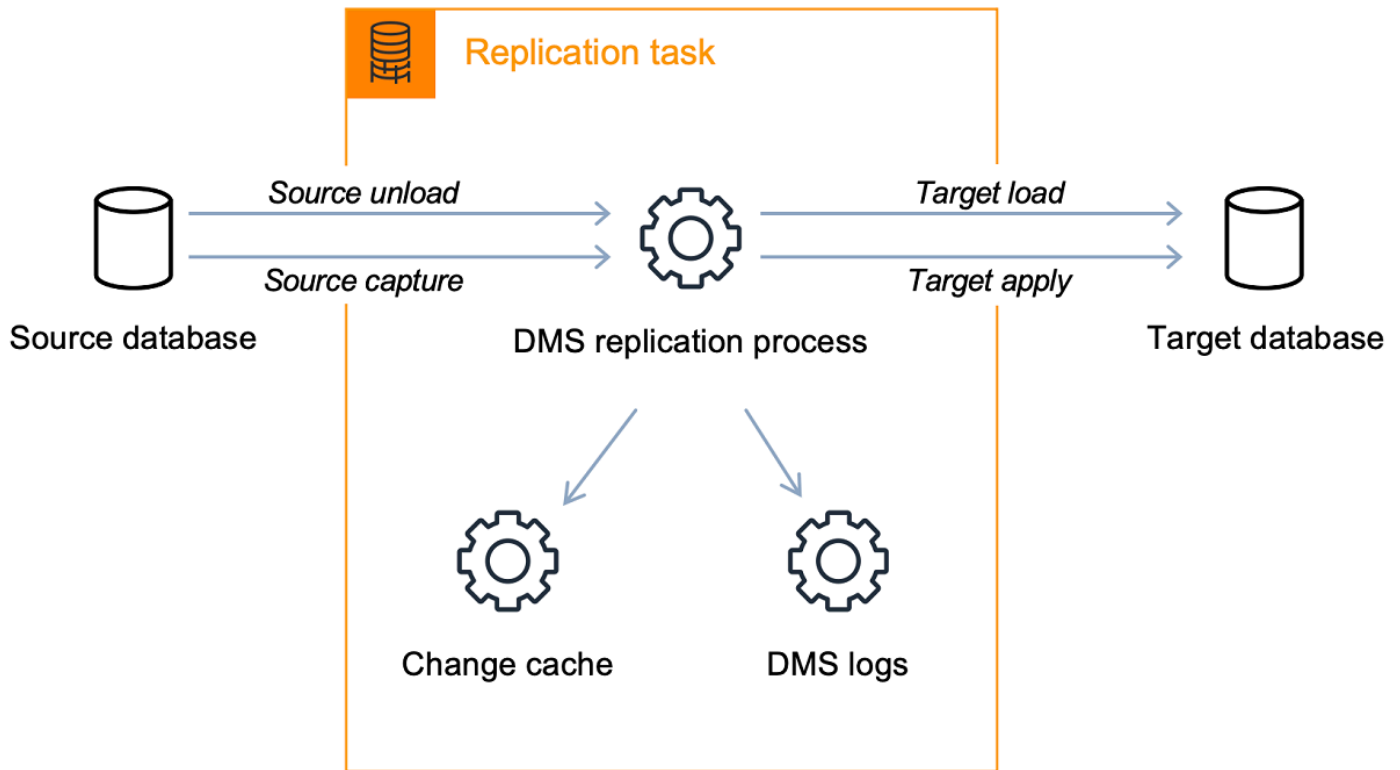
복제 작업을 생성할 때 다음 작업 설정을 지정합니다.

- 복제 인스턴스 - 태스크를 호스팅하고 실행할 인스턴스
- 소스 엔드포인트
- 타겟 엔드포인트
- 마이그레이션 유형 옵션은 다음과 같습니다. 마이그레이션 유형 옵션에 대한 자세한 설명은 [작업 생성](#)을 참조하세요.
 - 전체 로드(기존 데이터 마이그레이션) - 기존 데이터를 복사할 수 있을 정도로 긴 중단을 감당할 수 있는 경우 이 옵션이 적합합니다. 이 옵션은 간단히 원본 데이터베이스에서 대상 데이터베이스로 데이터를 마이그레이션하여 필요할 경우 테이블을 생성합니다.
 - 전체 로드 및 CDC(기존 데이터를 마이그레이션하고 진행 중 변경 사항 복제) - 이 옵션을 선택하면 소스에서 변경 사항을 캡처하는 동안 전체 데이터 로드를 수행합니다. 전체 로드가 완료되고 나면, 캡처된 변경 사항을 대상에 적용합니다. 결과적으로 변경 사항 적용은 안정적인 상태에 도달합니다. 이 시점에서 애플리케이션을 종료할 수 있고, 남은 변경 사항이 대상으로 흐르도록 한 후 대상을 나타내는 애플리케이션을 다시 시작할 수 있습니다.
 - CDC 전용(데이터 변경 사항만 복제) - 경우에 따라서는 AWS DMS 이외의 방법을 사용하여 기존 데이터를 복사하는 것이 더 효율적일 수 있습니다. 예를 들어, 같은 유형의 마이그레이션에서 기본 내보내기 및 가져오기 도구를 사용하면 대량 데이터를 로드하기에 더 효과적일 수 있습니다. 이 경우 AWS DMS 를 사용하여 대량 로드를 시작할 때부터 변경 사항을 복제하여 소스 데이터베이스와 대상 데이터베이스를 동기화된 상태로 유지할 수 있습니다.

- 대상 테이블 준비 모드 옵션은 다음과 같습니다. 대상 테이블 모드에 관한 완전한 설명을 보려면 [작업 생성](#) 섹션을 참조하세요.
- 아무 작업도 하지 않습니다. 대상 테이블에 대상 테이블이 미리 생성되어 있다고 AWS DMS 가 정합니다.
- 대상에 테이블 삭제 - 대상 테이블을 AWS DMS 삭제하고 다시 생성합니다.
- 자르기 - 대상에 테이블을 생성했다면 AWS DMS는 마이그레이션을 시작하기 전에 이 테이블을 자릅니다. 테이블이 없고 이 옵션을 선택하면 AWS DMS 가 누락된 테이블을 생성합니다.
- LOB 모드 옵션은 다음과 같습니다. LOB 모드에 관한 완전한 설명을 보려면 [AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정](#) 섹션을 참조하세요.
- LOB 열 포함 안 함 - LOB 열이 마이그레이션에서 제외됩니다.
- 전체 LOB 모드 - size. AWS DMS migrates에 LOBs 관계없이 전체 LOBs 크기 파라미터로 제어되는 청크로 구분하여 마이그레이션합니다. 이 모드는 제한적 LOB 모드를 사용하는 것보다 더 느립니다.
- 제한적 LOB 모드 - LOB를 최대 LOB 크기 파라미터에 지정된 값으로 자릅니다. 이 모드는 전체 LOB 모드를 사용하는 것보다 더 빠릅니다.
- 테이블 매핑 - 마이그레이션할 테이블과 마이그레이션 방법을 나타냅니다. 자세한 내용은 [작업 설정을 지정하기 위한 테이블 매핑 사용](#) 섹션을 참조하세요.
- 데이터 변환은 다음과 같습니다. 데이터 변환에 대한 자세한 내용은 [JSON을 사용하여 테이블 선택 및 변환 지정](#) 섹션을 참조하세요.
- 스키마, 테이블 및 열 이름 변경.
- 테이블스페이스 이름 변경(Oracle 대상 엔드포인트의 경우).
- 대상에서 프라이머리 키와 고유한 인덱스 정의.
- 데이터 유효성 검사
- Amazon CloudWatch 로깅

해당 작업을 사용하여 소스 엔드포인트에서 대상 엔드포인트로 데이터를 마이그레이션하고 나면 복제 인스턴스에서 작업 처리가 완료됩니다. 마이그레이션할 테이블 및 스키마와 모든 특수 처리(로깅 요구 사항, 제어 테이블 데이터 및 오류 처리 등)를 지정합니다.

개념적으로 AWS DMS 복제 작업은 다음 다이어그램과 같이 두 가지 개별 함수를 수행합니다.



전체 로드 프로세스는 이해하기 쉽습니다. 데이터는 대량 추출 방식으로 소스에서 추출되어 대상으로 직접 로드됩니다. 고급 설정에서 AWS DMS 콘솔에서 병렬로 추출하고 로드할 테이블 수를 지정할 수 있습니다.

AWS DMS 작업에 대한 자세한 내용은 단원을 참조하십시오 [AWS DMS 작업 작업](#).

지속적인 복제 또는 변경 데이터 캡처(CDC)

또한 데이터를 대상으로 마이그레이션하는 동안 AWS DMS 작업을 사용하여 소스 데이터 스토어의 지속적인 변경 사항을 캡처할 수 있습니다. 소스 엔드포인트에서 지속적인 변경 사항을 복제할 때 AWS DMS가 사용하는 변경 캡처 프로세스는 데이터베이스 엔진의 기본 API를 사용하여 데이터베이스 로그에 대한 변경 사항을 수집합니다.

CDC 프로세스에서 복제 작업은 인 메모리 버퍼를 사용해 전송 중인 데이터를 유지하는 방식으로 원본에서 대상으로 변경 사항을 스트림하도록 설계되어 있습니다. 어떤 이유로든 인 메모리 버퍼가 남지 않게 되면 복제 작업은 보류 중인 변경 사항을 디스크의 변경 사항 캐시로 유출합니다. 예를 들어 AWS DMS가 대상에 적용할 수 있는 것보다 더 빠르게 소스에서 변경 사항을 캡처하는 경우 이러한 문제가 발생할 수 있습니다. 이 경우 사용자는 작업의 대상 지연 시간이 작업의 소스 지연 시간을 초과하는지 확인해야 합니다.

AWS DMS 콘솔에서 작업으로 이동하여 작업 모니터링 탭을 열어 이를 확인할 수 있습니다.

CDCLatencyTarget 및 CDCLatencySource 그래프는 페이지 하단에 표시됩니다. 대상 지연 시간을

표시하는 작업이 있다면 애플리케이션 속도를 높이는 데 필요한 대상 엔드포인트 튜닝이 있을 가능성이 있습니다.

복제 태스크에서도 앞서 설명한 태스크 로그용 스토리지를 사용합니다. 복제 인스턴스로 사전 구성되어 제공되는 디스크 공간은 대개 로깅 및 유출된 변경 사항에 충분한 크기입니다. 추가 디스크 공간이 필요하다면(예: 마이그레이션 문제를 조사하기 위해 세부 디버깅을 사용하는 경우) 복제 인스턴스를 수정하여 더 많은 공간을 할당할 수 있습니다.

의 소스 AWS DMS

다양한 AWS DMS 기능의 다양한 소스 데이터 스토어를 사용할 수 있습니다. 다음 섹션에는 각 AWS DMS 기능에 대해 지원되는 소스 데이터 스토어 목록이 포함되어 있습니다.

주제

- [데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트](#)
- [DMS Fleet Advisor가 지원하는 소스 데이터베이스](#)
- [DMS Schema Conversion이 지원하는 소스 데이터 공급자](#)
- [DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자](#)

데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트

AWS DMS를 사용하여 다음 데이터 저장소를 데이터 마이그레이션용 소스 엔드포인트로 사용할 수 있습니다.

온프레미스 및 EC2 인스턴스 데이터베이스

- Enterprise, Standard, Standard One 및 Standard Two 에디션용 Oracle 버전 10.2 이상(버전 10.x의 경우), 11g 및 12.2 이하, 18c 및 19c
- Microsoft SQL Server 버전 2008(DMS v3.5.4에서 지원), 2008R2(DMS v3.5.4에서 지원), 2012, 2014, 2016, 2017, 2019 및 2022.
 - Enterprise, Standard, Workgroup, Developer 및 Web 에디션은 전체 로드 복제를 지원합니다.
 - Enterprise, Standard(버전 2016 이상) 및 Developer 에디션은 전체 로드 외에도 CDC(지속적인) 복제를 지원합니다.
 - Express 에디션은 지원되지 않습니다.
- MySQL 버전 5.5, 5.6, 5.7, 8.0 및 8.4

Note

MySQL 및 DMS 호환성

MySQL 버전	DMS 버전	압축 트랜잭션 페이로드 지원
8	3.4.0 이상	아니요
8.0(Google Cloud)	3.4.6 이상	아니요
8.4	3.5.4 이상	아니요

- MariaDB(MySQL 호환 데이터 소스로 지원됨) 버전 10.0(버전 10.0.24 이상만 해당), 10.2, 10.3, 10.4, 10.5, 10.6, 10.11 및 11.4.3~11.4.7.

Note

소스로서 MariaDB에 대한 지원은 MySQL이 지원되는 모든 AWS DMS 버전에서 사용할 수 있습니다.

- PostgreSQL 버전 9.4 이상(버전 9.x), 10.x, 11.x, 12.x, 13.x 14.x, 15.x, 16.x, 17.x 및 18.x.

Note

- AWS DMS 는 버전 3.5.3 이상에서 PostgreSQL 버전 9.x, 10.x, 11.x, 12.x, 13.x, 14.x, 15.x 및 16.x만 지원합니다.
- AWS DMS 는 버전 3.5.4 이상([제한](#) 사항 참조)에서 PostgreSQL 버전 17.x 및 18.x만 지원합니다.

- MongoDB 버전 3.x, 4.0, 4.2, 4.4, 5.0, 6.0, 7.0 및 8.0.
- SAP Adaptive Server Enterprise(ASE) 버전 12.5, 15, 15.5, 15.7, 16 이상
- Linux, UNIX 및 Windows용 IBM DB2(Db2 LUW) 버전:
 - 버전 9.7, 모든 수정 팩
 - 버전 10.1, 모든 수정 팩
 - 버전 10.5, 수정 팩 5를 제외한 모든 수정 팩

- 버전 11.1, 모든 수정 팩
- 버전 11.5, 수정 팩 제로만 포함된 Mod(0~8)
- IBM DB2 for z/OS 버전 12

서드 파티 관리형 데이터베이스 서비스:

- Microsoft Azure SQL Database
- Microsoft Azure PostgreSQL Flexible Server 버전 11.2, 12.15, 13.11, 14.8 및 15.3.
- Microsoft Azure MySQL Flexible Server 버전 5.7 및 8.
- Google Cloud for MySQL 버전 5.6, 5.7 및 8.0.
- Google Cloud for PostgreSQL 버전 9.6, 10, 11, 12, 13, 14 및 15.
- OCI MySQL Heatwave 버전 8.0.34.

Amazon RDS 인스턴스 데이터베이스 및 Amazon Simple Storage Service(Amazon S3)

- Enterprise, Standard, Standard One 및 Standard Two 에디션용 Oracle 버전 11g(버전 11.2.0.4 이상) 및 12.2 이하, 18c 및 19c.
- Enterprise, Standard, Workgroup 및 Developer 에디션용 Microsoft SQL Server 버전 2016, 2017, 2019 및 2022. 자세한 내용은 Amazon RDS 사용 설명서의 [Amazon RDS for Microsoft SQL Server](#)를 참조하세요.

 Note

AWS DMS 는 SQL Server Express를 지원하지 않습니다. Web 에디션은 전체 로드 전용 복제에만 지원됩니다.

- MySQL 버전 5.5, 5.6, 5.7, 8.0 및 8.4.

Note

MySQL 및 DMS 호환성

MySQL 버전	DMS 버전	압축 트랜잭션 페이로드 지원
8	3.4.0 이상	아니요
8.0(Google Cloud)	3.4.6 이상	아니요
8.4	3.5.4 이상	아니요

- MariaDB(MySQL 호환 데이터 소스로 지원됨) 버전 10.0.24~10.0.28, 10.2, 10.3, 10.4, 10.5, 10.6, 10.11 및 11.4.3~11.4.7.

Note

소스로서 MariaDB에 대한 지원은 MySQL이 지원되는 모든 AWS DMS 버전에서 사용할 수 있습니다.

- PostgreSQL 버전 10.x, 11.x, 12.x, 13.x, 14.x, 15.x, 16.x, 17.x 및 18.x.

Note

- AWS DMS 는 버전 3.5.3 이상에서 PostgreSQL 버전 10.x, 11.x, 12.x, 13.x, 14.x, 15.x 및 16.x만 지원합니다.
- AWS DMS 는 버전 3.5.4 이상([제한](#) 사항 참조)에서 PostgreSQL 버전 17.x 및 18.x만 지원합니다.

- MySQL 호환 Amazon Aurora(MySQL 호환 데이터 소스로 지원됨)
- PostgreSQL 호환 Amazon Aurora(PostgreSQL 호환 데이터 소스로 지원됨)
- Amazon S3
- Amazon DocumentDB(MongoDB 호환) 버전 3.6, 4.0 및 5.0.
- IBM DB2 LUW용 Amazon RDS.

특정 소스 작업에 대한 자세한 내용은 [AWS DMS 엔드포인트 작업을](#) 참조하세요.

지원되는 대상 엔드포인트에 대한 자세한 내용은 [데이터 마이그레이션에 사용할 수 있는 대상 엔드포인트](#) 섹션을 참조하세요.

DMS Fleet Advisor가 지원하는 소스 데이터베이스

DMS Fleet Advisor는 다음 소스 데이터베이스를 지원합니다.

- Microsoft SQL Server 버전 2012 및 2019 이하
- MySQL 버전 5.6 및 8 이하
- Oracle 버전 11g 릴리스 2 및 12c 이하, 19c 및 21c
- PostgreSQL 버전 9.6 및 13 이하

특정 소스를 사용한 작업에 대한 자세한 내용은 [AWS DMS Fleet Advisor용 데이터베이스 사용자 생성](#) 섹션을 참조하세요.

DMS Fleet Advisor가 대상 권장 사항을 생성하는 데 사용하는 데이터베이스의 목록은 [DMS Fleet Advisor가 지원하는 대상](#) 섹션을 참조하세요.

DMS Schema Conversion이 지원하는 소스 데이터 공급자

DMS Schema Conversion은 다음 데이터 공급자를 마이그레이션 프로젝트의 소스로 지원합니다.

- Microsoft SQL Server(버전 2008 R2, 2012, 2014, 2016, 2017, 2019, 2022)
- Oracle 버전 10.2 이상, 11g 및 12.2 이하, 18c 및 19c, Oracle Data Warehouse
- PostgreSQL 버전 9.2 이상
- MySQL 버전 5.5, 5.6, 5.7 및 8.0.
- Linux, UNIX 및 Windows용 IBM Db2 버전 9.1, 9.5, 9.7, 10.1, 10.5, 11.1 및 11.5
- IBM DB2 for z/OS 버전 12
- SAP ASE(Sybase ASE) 버전 16

Note

DMS Schema Conversion은이 주제에 나열된 모든 Amazon RDS 버전 소스를 지원합니다.

소스 데이터베이스 공급자는 온프레미스 또는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 실행되는 자체 관리형 엔진일 수 있습니다.

특정 소스를 사용한 작업에 대한 자세한 내용은 [DMS Schema Conversion에서 소스 데이터 공급자 생성](#) 섹션을 참조하세요.

지원되는 대상 데이터베이스에 관한 자세한 내용은 [DMS Schema Conversion이 지원하는 대상 데이터 공급자](#) 섹션을 참조하세요.

AWS Schema Conversion Tool (AWS SCT)는 DMS Schema Conversion보다 더 많은 소스 및 대상 데이터베이스를 지원합니다. 가 AWS SCT 지원하는 데이터베이스에 대한 자세한 내용은 [란 무엇입니까?를 AWS Schema Conversion Tool](#) 참조하십시오.

DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자

다음 데이터 공급자를 온프레미스 및 EC2 인스턴스 데이터베이스로의 동종 데이터 마이그레이션의 소스로 사용할 수 있습니다.

- MySQL 버전 5.7 및 8.0
- MariaDB 버전 10.2x
- PostgreSQL 버전 10.4~16.x.
- MongoDB 버전 4.x, 5.x, 6.0, 7.0, 8.0
- Amazon DocumentDB 버전 3.6, 4.0, 5.0

다음 데이터 공급자를 Amazon Relational Database Service 인스턴스 데이터베이스로 동종 데이터 마이그레이션을 위한 소스로 사용할 수 있습니다.

- Aurora MySQL 버전 5.7 및 8.0
- RDS for MySQL 버전 5.7 및 8.0
- RDS for MariaDB 버전 10.2.x
- Aurora PostgreSQL 버전 11.x~17.x
- RDS for PostgreSQL 버전 11.x~17.x

소스 데이터베이스 공급자는 온프레미스 또는 Amazon EC2 인스턴스에서 실행되는 자체 관리형 엔진일 수 있습니다. 또한 Amazon RDS DB 인스턴스를 소스 데이터 공급자로 사용할 수 있습니다.

특정 소스를 사용한 작업에 대한 자세한 내용은 [에서 동종 데이터 마이그레이션을 위한 소스 데이터 공급자 생성 AWS DMS](#) 섹션을 참조하세요.

지원되는 대상 데이터베이스에 관한 자세한 내용은 [DMS 동종 데이터 마이그레이션이 지원하는 대상 데이터 공급자](#) 섹션을 참조하세요.

의 대상 AWS DMS

다양한 AWS DMS 기능의 다양한 대상 데이터 스토어를 사용할 수 있습니다. 다음 섹션에는 각 AWS DMS 기능에 대해 지원되는 대상 데이터 스토어 목록이 포함되어 있습니다.

주제

- [데이터 마이그레이션에 사용할 수 있는 대상 엔드포인트](#)
- [DMS Fleet Advisor가 지원하는 대상 데이터베이스](#)
- [DMS Schema Conversion이 지원하는 대상 데이터 공급자](#)
- [DMS 동종 데이터 마이그레이션이 지원하는 대상 데이터 공급자](#)

데이터 마이그레이션에 사용할 수 있는 대상 엔드포인트

AWS DMS를 사용하여 다음 데이터 저장소를 데이터 마이그레이션용 대상 엔드포인트로 사용할 수 있습니다.

온프레미스 및 Amazon EC2 인스턴스 데이터베이스

- Enterprise, Standard, Standard One 및 Standard Two 에디션용 Oracle 버전 10g, 11g, 12c, 18c 및 19c
- Enterprise, Standard, Workgroup 및 Developer 에디션용 Microsoft SQL Server 버전 2005, 2008, 2008R2, 2012, 2014, 2016, 2017, 2019, 2022.

Note

AWS DMS 는 SQL Server Web 및 Express 에디션을 지원하지 않습니다.

- MySQL 버전 5.5, 5.6, 5.7, 8.0 및 8.4
- MariaDB(MySQL 호환 데이터 대상으로 지원됨) 버전 10.0.24~10.0.28, 10.2, 10.3, 10.4, 10.5, 10.6, 10.11 및 11.4.3~11.4.7.

Note

대상으로서 MariaDB에 대한 지원은 MySQL이 지원되는 모든 AWS DMS 버전에서 사용할 수 있습니다.

- PostgreSQL 버전 9.4 이상(버전 9.x), 10.x, 11.x, 12.x, 13.x, 14.x, 15.x, 16.x, 17.x 및 18.x.

Note

- AWS DMS 는 버전 3.5.3 이상에서 PostgreSQL 버전 9.x, 10.x, 11.x, 12.x, 13.x, 14.x, 15.x 및 16.x만 지원합니다.
- AWS DMS 는 버전 3.5.4 이상에서 PostgreSQL 버전 17.x 및 18.x만 지원합니다.

- SAP Adaptive Server Enterprise(ASE) 버전 15, 15.5, 15.7, 16 이상
- Redis OSS 버전 6.x

Amazon RDS 인스턴스 데이터베이스, Amazon Redshift, Amazon Redshift Serverless, Amazon DynamoDB, Amazon S3, Amazon OpenSearch Service, Amazon ElastiCache(Redis OSS), Amazon Kinesis Data Streams, Amazon DocumentDB, Amazon Neptune 및 Apache Kafka

- Enterprise, Standard, Standard One 및 Standard Two 에디션용 Oracle 버전 11g(버전 11.2.0.3.v1 이상), 12c, 18c 및 19c
- Enterprise, Standard, Workgroup 및 Developer 에디션용 Microsoft SQL Server 버전 2012, 2014, 2016, 2017, 2019, 2022

Note

AWS DMS 는 SQL Server Web 및 Express 에디션을 지원하지 않습니다.

- MySQL 버전 5.5, 5.6, 5.7, 8.0 및 8.4.
- MariaDB(MySQL 호환 데이터 대상으로 지원됨) 버전 10.0.24~10.0.28, 10.2, 10.3, 10.4, 10.5, 10.6, 10.11 및 11.4.3~11.4.7.

Note

대상으로서 MariaDB에 대한 지원은 MySQL이 지원되는 모든 AWS DMS 버전에서 사용할 수 있습니다.

- PostgreSQL 버전 10.x, 11.x, 12.x, 13.x, 14.x, 15.x, 16.x, 17.x 및 18.x.

Note

- AWS DMS 는 버전 3.5.3 이상에서 PostgreSQL 버전 10.x, 11.x, 12.x, 13.x, 14.x, 15.x 및 16.x만 지원합니다.
- AWS DMS 는 버전 3.5.4 이상에서 PostgreSQL 버전 17.x 및 18.x만 지원합니다.

- IBM DB2 LUW 버전 11.1 및 11.5
- Amazon Aurora MySQL 호환 버전
- Amazon Aurora PostgreSQL 호환 에디션
- Amazon Aurora PostgreSQL Limitless
- Amazon Aurora Serverless v2
- Amazon Redshift
- Amazon Redshift Serverless
- Amazon S3
- Amazon DynamoDB
- Amazon OpenSearch Service
- Amazon ElastiCache (Redis OSS)
- Amazon Kinesis Data Streams
- Amazon DocumentDB(MongoDB 호환)
- Amazon Neptune
- Apache Kafka – [Amazon Managed Streaming for Apache Kafka\(Amazon MSK\)](#) 및 [자체 관리형 Apache Kafka](#)
- Aurora PostgreSQL(버전 15.3/14.8 이상)용 Babelfish(버전 3.2.0 이상)

특정 대상 작업에 대한 자세한 내용은 [AWS DMS 엔드포인트 작업을](#) 참조하세요.

지원되는 소스 엔드포인트에 대한 자세한 내용은 [데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트](#) 섹션을 참조하세요.

DMS Fleet Advisor가 지원하는 대상 데이터베이스

DMS Fleet Advisor는 다음 대상 데이터베이스의 최신 버전을 사용하여 대상 권장 사항을 생성합니다.

- Amazon Aurora MySQL
- Amazon Aurora PostgreSQL
- Amazon RDS for MySQL
- Amazon RDS for Oracle
- Amazon RDS for PostgreSQL
- Amazon RDS for SQL Server

DMS Fleet Advisor의 대상 권장 사항에 대한 자세한 내용은 [AWS DMS Fleet Advisor 대상 권장 사항 기능 사용](#) 섹션을 참조하세요.

지원되는 소스 데이터베이스에 대한 자세한 내용은 [DMS Fleet Advisor가 지원하는 소스 데이터베이스](#) 섹션을 참조하세요.

DMS Schema Conversion이 지원하는 대상 데이터 공급자

DMS Schema Conversion은 다음 데이터 공급자를 마이그레이션 프로젝트의 대상으로 지원합니다.

- Amazon Aurora MySQL 8.0.32
- Amazon Aurora PostgreSQL 14.x, 15.x, 16.x, 17.x
- Amazon RDS for MySQL 8.0.23
- Amazon RDS for PostgreSQL 14.x, 15.x, 16.x, 17.x, 18.x
- Amazon Redshift
- Amazon RDS for Db2 버전 11.5.

특정 대상을 사용한 작업에 대한 자세한 내용은 [DMS Schema Conversion에서 대상 데이터 공급자 생성 및 설정](#) 섹션을 참조하세요.

지원되는 소스 데이터베이스에 대한 자세한 내용은 [DMS Schema Conversion이 지원하는 소스 데이터 공급자](#) 섹션을 참조하세요.

DMS 동종 데이터 마이그레이션이 지원하는 대상 데이터 공급자

다음 데이터 공급자를 동종 데이터 마이그레이션의 대상으로 사용할 수 있습니다.

- Amazon Aurora MySQL 버전 5.7 및 8.0
- Amazon Aurora PostgreSQL 버전 10.4~16.x
- Amazon Aurora Serverless v2
- Amazon RDS for MySQL 버전 5.7 및 8.0
- Amazon RDS for MariaDB 버전 10.2x
- Amazon RDS for PostgreSQL 버전 10.4~16.x
- Amazon DocumentDB 버전 4.0, 5.0 및 DocumentDB Elastic 클러스터

특정 대상을 사용한 작업에 대한 자세한 내용은 [AWS DMS 스키마 변환에 사용할 대상 데이터베이스 생성 및 설정](#) 섹션을 참조하세요.

지원되는 소스 데이터베이스에 대한 자세한 내용은 [DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자](#) 섹션을 참조하세요.

에 대한 Amazon 리소스 이름(ARN) 구성 AWS DMS

AWS CLI 또는 AWS DMS API를 사용하여 데이터베이스 마이그레이션을 자동화하는 경우 Amazon 리소스 이름(ARNs. Amazon Web Services에서 생성되는 각 리소스는 고유한 식별자인 ARN으로 식별됩니다. AWS CLI 또는 AWS DMS API를 사용하여 데이터베이스 마이그레이션을 설정하는 경우 작업할 리소스의 ARN을 제공합니다.

AWS DMS 리소스의 ARN은 다음 구문을 사용합니다.

`arn:aws:dms:region:account number:resourcetype:resourcename`

이 구문에는 다음이 적용됩니다.

- *region*은와 같이 AWS DMS 리소스가 AWS 리전 생성된 ID입니다us-west-2.

다음 표에는 ARN을 생성할 때 사용해야 하는 AWS 리전 이름과 값이 나와 있습니다.

리전	이름
아시아 태평양(도쿄) 리전	ap-northeast-1

리전	이름
아시아 태평양(서울) 리전	ap-northeast-2
아시아 태평양(뭄바이) 리전	ap-south-1
아시아 태평양(싱가포르) 리전	ap-southeast-1
아시아 태평양(시드니) 리전	ap-southeast-2
캐나다(중부) 리전	ca-central-1
중국(베이징) 리전	cn-north-1
중국(닝샤) 리전	cn-northwest-1
유럽(스톡홀름) 리전	eu-north-1
유럽(밀라노) 리전	eu-south-1
EU(프랑크푸르트) 리전	eu-central-1
유럽(아일랜드) 리전	eu-west-1
EU(런던) 리전	eu-west-2
EU(파리) 리전	eu-west-3
남아메리카(상파울루) 리전	sa-east-1
미국 동부(버지니아 북부) 리전	us-east-1
미국 동부(오하이오) 리전	us-east-2
미국 서부(캘리포니아 북부) 리전	us-west-1
미국 서부(오리곤) 리전	us-west-2

- **account number**는 대시가 생략된 계정 번호입니다. 계정 번호를 찾으려면 <http://aws.amazon.com> AWS 계정에 로그인하고 내 계정/콘솔을 선택한 다음 내 계정을 선택합니다.
- **resourcetype**는 AWS DMS 리소스 유형입니다.

다음 표에는 특정 리소스에 대한 ARN을 구성할 때 사용할 AWS DMS 리소스 유형이 나와 있습니다.

AWS DMS 리소스 유형	ARN 형식
복제 인스턴스	arn:aws:dms: <i>region</i> : <i>account</i> :rep: <i>resourcename</i>
엔드포인트	arn:aws:dms: <i>region</i> : <i>account</i> :endpoint: <i>resourcename</i>
복제 작업	arn:aws:dms: <i>region</i> : <i>account</i> :task: <i>resourcename</i>
서브넷 그룹	arn:aws:dms: <i>region</i> : <i>account</i> :subgrp: <i>resourcename</i>

- *resourcename*는 리소스에 할당된 AWS DMS 리소스 이름입니다. 이것은 생성된 임의의 문자입니다.

다음 표에는 AWS DMS 리소스에 대한 ARNs의 예가 나와 있습니다. 여기서는 123456789012라는 AWS 계정을 가정합니다. 이 계정은 미국 동부(버지니아 북부) 리전에서 생성되었고 리소스 이름이 있습니다.

리소스 유형	샘플 ARN
복제 인스턴스	arn:aws:dms:us-east-1:123456789012:rep:QLXQZ64MH7CXF4QCQMGRVYVXAI
엔드포인트	arn:aws:dms:us-east-1:123456789012:endpoint:D3HMZ2IGUCGFF3NTAXUXGF6S5A
복제 작업	arn:aws:dms:us-east-1:123456789012:task:2PVREMNPGYJCVU2IBPTOYTIV4
서브넷 그룹	arn:aws:dms:us-east-1:123456789012:subgrp:test-tag-grp

다른 AWS 서비스와 AWS DMS 함께 사용

를 다른 여러 AWS 서비스와 AWS DMS 함께 사용할 수 있습니다.

- Amazon EC2 인스턴스 또는 Amazon RDS DB 인스턴스를 데이터 마이그레이션 대상으로 사용할 수 있습니다.
- AWS Schema Conversion Tool (AWS SCT)를 사용하여 소스 스키마와 SQL 코드를 동등한 대상 스키마와 SQL 코드로 변환할 수 있습니다.
- Amazon S3를 데이터 스토리지 사이트로 사용하거나, 많은 데이터를 마이그레이션할 때 중간 단계로 사용할 수 있습니다.
- CloudFormation 를 사용하여 인프라 관리 또는 배포를 위한 AWS 리소스를 설정할 수 있습니다. 예를 들어 복제 인스턴스, 작업, 인증서 및 엔드포인트와 같은 AWS DMS 리소스를 프로비저닝할 수 있습니다. 원하는 모든 AWS 리소스를 설명하는 템플릿을 생성하면 해당 리소스를 CloudFormation 프로비저닝하고 구성합니다.

AWS DMS 에 대한 지원 AWS CloudFormation

를 사용하여 AWS DMS 리소스를 프로비저닝할 수 있습니다 AWS CloudFormation. AWS CloudFormation 는 인프라 관리 또는 배포를 위한 AWS 리소스를 모델링하고 설정하는 데 도움이 되는 서비스입니다. 예를 들어 복제 인스턴스, 작업, 인증서 및 엔드포인트와 같은 AWS DMS 리소스를 프로비저닝할 수 있습니다. 원하는 모든 AWS 리소스를 설명하는 템플릿을 생성하고 해당 리소스를 AWS CloudFormation 프로비저닝하고 구성합니다.

개발자나 시스템 관리자는 반복적인 마이그레이션 작업이나 조직에 리소스 배포 작업에 사용할 수 있는 이러한 리소스의 모음을 생성 및 관리할 수 있습니다. 에 대한 자세한 내용은 사용 설명서의 [AWS CloudFormation 개념](#)을 AWS CloudFormation참조하세요. AWS CloudFormation

AWS DMS 는 AWS CloudFormation다음을 사용하여 다음 AWS DMS 리소스를 생성할 수 있도록 지원합니다.

- [AWS::DMS::Certificate](#)
- [AWS::DMS::Endpoint](#)
- [AWS::DMS::EventSubscription](#)
- [AWS::DMS::ReplicationInstance](#)
- [AWS::DMS::ReplicationSubnetGroup](#)
- [AWS::DMS::ReplicationTask](#)

시작하기 AWS Database Migration Service

다음 자습서에서는 AWS Database Migration Service ()를 사용하여 데이터베이스 마이그레이션을 수행하는 방법을 알아볼 수 있습니다AWS DMS.

데이터베이스 마이그레이션을 수행하려면 다음 단계를 따릅니다.

1. 의 단계에 따라 AWS 계정을 설정합니다 [에 대한 설정AWS Database Migration Service](#).
2. 샘플 데이터베이스와 Amazon EC2 클라이언트를 생성하여 소스 데이터베이스를 채우고 복제를 테스트합니다. 또한 Amazon Virtual Private Cloud(Amazon VPC) 서비스를 기반으로 Virtual Private Cloud(VPC)를 생성하여 자습서 리소스를 포함시킵니다. 이러한 리소스를 생성하려면 [설정을 위한 사전 조건 완료 AWS Database Migration Service](#)의 단계를 따릅니다.
3. [샘플 데이터베이스 생성 스크립트](#)를 사용하여 소스 데이터베이스를 채웁니다.
4. DMS Schema Conversion을 사용하여 소스 데이터베이스에서 대상 데이터베이스로 스키마를 변환합니다. [DMS Schema Conversion 시작하기](#) 섹션의 단계를 따르세요. 또는 legacy AWS Schema Conversion Tool (AWS SCT)를 사용하는 경우의 단계를 따릅니다 [스키마 마이그레이션](#).
5. 마이그레이션을 위한 모든 프로세스를 수행하는 복제 인스턴스를 생성합니다. 이 태스크 및 후속 태스크를 수행하려면 [복제](#)의 단계를 수행합니다.
6. 소스 및 대상 엔드포인트를 지정합니다. 엔드포인트를 생성하는 방법에 대한 자세한 내용은 [소스 및 대상 엔드포인트 생성](#) 섹션을 참조하세요.
7. 사용할 테이블 및 복제 프로세스를 정의하는 태스크를 생성하고 복제를 시작합니다. 데이터베이스 마이그레이션 태스크를 생성하는 방법에 대한 자세한 내용은 [작업 생성](#) 섹션을 참조하세요.
8. 대상 데이터베이스에서 쿼리를 실행하여 복제가 제대로 작동하는지 확인합니다.

작업 방법에 대해 자세히 알아보기 AWS Database Migration Service

이 가이드의 뒷부분에서를 사용하여 가장 널리 사용되는 상용 및 오픈 소스 데이터베이스로 데이터를 AWS DMS 마이그레이션하는 방법을 배울 수 있습니다.

또한 데이터베이스 마이그레이션 프로젝트를 준비하고 수행할 때 다음 리소스를 확인하는 것이 좋습니다.

- [AWS DMS Step-by-Step 마이그레이션 가이드](#) -이 가이드에서는 데이터를 로 마이그레이션하는 프로세스를 단계별로 step-by-step 안내합니다 AWS.

- [AWS DMS API 참조](#) -이 참조는에 대한 모든 API 작업을 AWS Database Migration Service 자세히 설명합니다.
- [AWS CLI for AWS DMS](#) -이 참조는 AWS Command Line Interface (AWS CLI)를와 함께 사용하는 방법에 대한 정보를 제공합니다 AWS DMS.

에 대한 설정AWS Database Migration Service

에 가입AWS 계정

이 없는 경우 다음 단계를 AWS 계정완료하여 생성합니다.

에 가입하려면AWS 계정

1. <https://portal.aws.amazon.com/billing/signup>을 엽니다.
2. 온라인 지침을 따릅니다.

등록 절차 중 전화 또는 텍스트 메시지를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

에 가입하면AWS 계정 AWS 계정 루트 사용자의 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스에 액세스할 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 [루트 사용자 액세스 권한이 필요한 작업](#)을 수행하는 것입니다.

AWS는 가입 프로세스가 완료된 후 확인 이메일을 보냅니다. 언제든지 <https://aws.amazon.com/>으로 이동하고 내 계정을 선택하여 현재 계정 활동을 확인하고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

에 가입한 후 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 AWS 계정보호AWS IAM Identity Center, AWS 계정 루트 사용자활성화 및 생성합니다.

보안AWS 계정 루트 사용자

1. 루트 사용자를 선택하고 AWS 계정이메일 주소를 입력하여 계정 소유자[AWS Management Console](#)로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

루트 사용자를 사용하여 로그인하는 데 도움이 필요하다면 AWS 로그인 사용 설명서의 [루트 사용자 로 로그인](#)을 참조하세요.

2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

지침은 IAM 사용 설명서의 [AWS 계정루트 사용자\(콘솔\)에 대한 가상 MFA 디바이스 활성화를 참조하세요.](#)

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

지침은 AWS IAM Identity Center 사용자 안내서의 [AWS IAM Identity Center 설정](#)을 참조하세요.

2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

를 자격 증명 소스IAM Identity Center 디렉터리로 사용하는 방법에 대한 자습서는 사용 AWS IAM Identity Center 설명서의 [기본값으로 사용자 액세스 구성을IAM Identity Center 디렉터리](#) 참조하세요.

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM Identity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

IAM Identity Center 사용자를 사용하여 로그인하는 데 도움이 필요하다면 AWS 로그인사용 설명서의 [AWS 액세스 포털에 로그인](#)을 참조하세요.

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

지침은 [Create a permission set](#)에 대한 지침은 AWS IAM Identity Center 사용 설명서에서 참조하세요.

2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

지침은 AWS IAM Identity Center 사용 설명서의 [Add groups](#)를 참조하세요.

설정을 위한 사전 조건 완료 AWS Database Migration Service

이 섹션에서는 소스 및 대상 데이터베이스 설정 AWS DMS와 같은의 사전 필수 작업을 배울 수 있습니다. 이러한 작업의 일부로 Amazon VPC 서비스 기반의 가상 프라이빗 클라우드(VPC)를 설정하여 리소스를 포함시킵니다. 또한 소스 데이터베이스를 채우고 대상 데이터베이스에서 복제를 확인하는 데 사용할 Amazon EC2 인스턴스를 설정합니다.

Note

소스 데이터베이스를 채우는 데 최대 45분이 걸립니다.

이 자습서에서는 MariaDB 데이터베이스를 소스로 생성하고 PostgreSQL 데이터베이스를 대상으로 생성합니다. 이 시나리오에서는 일반적으로 사용되는 저렴한 데이터베이스 엔진을 사용하여 복제를 시연합니다. 서로 다른 데이터베이스 엔진을 사용하면 이기종 플랫폼 간에 데이터를 마이그레이션하는 AWS DMS 기능을 보여줍니다.

이 자습서의 리소스에서는 미국 서부(오리곤) 리전을 사용합니다. 다른 AWS 리전을 사용하려면 미국 서부(오레곤)가 나타날 때마다 선택한 리전을 대신 지정합니다.

Note

단순성을 위해 이 자습서에서 생성하는 데이터베이스는 암호화 또는 기타 고급 보안 기능을 사용하지 않습니다. 프로덕션 데이터베이스를 안전하게 보호하려면 보안 기능을 사용해야 합니다. 자세한 내용은 [Amazon RDS의 보안](#)을 참조하세요.

사전 조건 단계는 다음 항목을 참조하세요.

주제

- [VPC 생성](#)
- [Amazon RDS 파라미터 그룹 생성](#)
- [소스 Amazon RDS 데이터베이스 생성](#)
- [대상 Amazon RDS 데이터베이스 생성](#)
- [Amazon EC2 클라이언트 생성](#)
- [소스 데이터베이스 채우기](#)

VPC 생성

이 섹션에서는 AWS 리소스를 포함할 VPC를 생성합니다. VPC 사용은 AWS 리소스를 사용할 때 데이터베이스, Amazon EC2 인스턴스, 보안 그룹 등이 논리적으로 구성되고 안전하도록 하는 모범 사례입니다.

VPC를 자습서 리소스로 사용하면 자습서를 완료했을 때 사용한 모든 리소스도 삭제됩니다. VPC를 삭제하려면 먼저 VPC에 포함된 모든 리소스를 삭제해야 합니다.

와 함께 사용할 VPC를 생성하려면 AWS DMS

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/vpc/> Amazon VPC 콘솔을 엽니다.
2. 탐색 창에서 VPC 대시보드, 대시보드 생성을 차례로 선택합니다.
3. VPC 생성 페이지에서 다음 옵션을 선택합니다.
 - 생성할 리소스: VPC 등
 - 이름 태그 자동 생성: 자동 생성을 선택하고 **DMSVPC**를 입력합니다.
 - IPv4 블록: **10.0.1.0/24**
 - IPv6 CIDR 블록: IPv6 CIDR 블록 없음
 - 테넌시: 기본값
 - 가용 영역 수: 2
 - 퍼블릭 서브넷 수: 2
 - 프라이빗 서브넷 수: 2
 - NAT 게이트웨이(\$): 없음
 - VPC 엔드포인트: 없음

VPC 생성을 선택합니다.

4. 탐색 창에서 VPC를 선택합니다. DMSVPC의 VPC ID를 적어둡니다.
5. 탐색 창에서 보안 그룹을 선택합니다.
6. DMSVPC에 대해 적어둔 ID와 일치하는 VPC ID를 가진 default라는 그룹을 선택합니다.
7. 인바운드 규칙 탭을 선택하고 인바운드 규칙 편집을 선택합니다.
8. 규칙 추가를 선택합니다. MySQL/Aurora 유형의 규칙을 추가하고 소스로 Anywhere-IPv4를 선택합니다.

9. 규칙 추가를 다시 선택합니다. PostgreSQL 유형의 규칙을 추가하고 소스로 Anywhere-IPv4를 선택합니다.
10. 규칙 저장을 선택합니다.

Amazon RDS 파라미터 그룹 생성

소스 및 대상 데이터베이스의 설정을 지정하려면 Amazon RDS 파라미터 그룹을 AWS DMS 사용합니다. 데이터베이스 간 초기 복제와 지속적 복제를 허용하려면 다음을 구성해야 합니다.

- 가 복제해야 하는 증분 업데이트를 결정할 AWS DMS 수 있도록 소스 데이터베이스의 바이너리 로그입니다.
- 대상 데이터베이스의 복제 역할로,는 초기 데이터 전송 중에 외래 키 제약 조건을 AWS DMS 무시합니다. 이 설정을 사용하면가 데이터를 순서에 맞지 않게 마이그레이션할 AWS DMS 수 있습니다.

와 함께 사용할 파라미터 그룹을 생성하려면 AWS DMS

1. <https://console.aws.amazon.com/rds/>에서 Amazon RDS 콘솔을 엽니다.
2. 탐색 창에서 파라미터 그룹을 선택합니다.
3. 파라미터 그룹 페이지에서 파라미터 그룹 생성을 선택합니다.
4. 파라미터 그룹 생성페이지에서 다음 설정을 입력합니다.
 - 파라미터 그룹 패밀리: mariadb10.6
 - 그룹 이름: **dms-mariadb-parameters**
 - 설명: **Group for specifying binary log settings for replication**

생성을 선택합니다.

5. 파라미터 그룹 페이지에서 dms-mariadb-parameters를 선택하고, dms-mariadb-parameters 페이지에서 편집을 선택합니다.
6. 다음 파라미터를 다음 값 중 하나로 설정합니다.
 - binlog_checksum: 없음
 - binlog_format: 행

변경 사항 저장을 선택합니다.

7. 파라미터 그룹 페이지에서 파라미터 그룹 생성을 다시 선택합니다.
8. 파라미터 그룹 생성페이지에서 다음 설정을 입력합니다.
 - 파라미터 그룹 패밀리: postgres16
 - 그룹 이름: **dms-postgresql-parameters**
 - 설명: **Group for specifying role setting for replication**

생성을 선택합니다.

9. 파라미터 그룹 페이지에서 dms-postgresql-parameters를 선택합니다.
10. dms-postgresql-parameters 페이지에서 편집을 선택하고 session_replication_role 파라미터를 복제본으로 설정합니다. 참고로, session_replication_role 파라미터는 파라미터의 첫 페이지에 없습니다. 페이지 매김 컨트롤 또는 검색 필드를 사용하여 이 파라미터를 찾으세요.
11. 변경 사항 저장을 선택합니다.

소스 Amazon RDS 데이터베이스 생성

다음 절차에 따라 소스 Amazon RDS 데이터베이스를 생성합니다.

소스 Amazon RDS for MariaDB 데이터베이스를 생성하려면

1. <https://console.aws.amazon.com/rds/>에서 Amazon RDS 콘솔을 엽니다.
2. 대시보드 페이지의 데이터베이스 섹션에서 데이터베이스 생성을 선택합니다. 페이지 상단의 새로운 MySQL 및 PostgreSQL용 Amazon RDS Multi-AZ 배포 옵션 사용해 보기 섹션에서 데이터베이스 생성을 선택하지 마세요.
3. 데이터베이스 생성 페이지에서 다음 옵션을 선택합니다.
 - 데이터베이스 생성 방법 선택: 표준 생성을 선택합니다.
 - 엔진 옵션: 엔진 유형에서 MariaDB를 선택합니다. 버전의 경우 MariaDB 10.6.14를 선택한 상태로 둡니다.
 - 템플릿: 개발 및 테스트를 선택합니다.
 - 설정:
 - DB 인스턴스 식별자: **dms-mariadb**를 입력합니다.
 - 자격 증명 정보 설정 섹션에서 다음을 입력합니다.
 - 마스터 사용자 이름: **admin**을 그대로 둡니다.

- AWS Secrets Manager에서 마스터 자격 증명 관리를 선택하지 않은 상태로 둡니다.
- 암호 자동 생성: 선택하지 않은 상태로 둡니다.
- 마스터 암호: **changeit**을 입력합니다.
- 암호 확인: **changeit**을 다시 입력합니다.
- 인스턴스 구성:
 - DB 인스턴스 클래스: 표준 클래스가 선택된 상태로 둡니다.
 - DB 인스턴스 클래스에서 db.m5.large를 선택합니다.
- 스토리지:
 - 스토리지 자동 조정 활성화 확인란의 선택을 취소합니다.
 - 나머지 설정은 그대로 둡니다.
- 가용성 및 내구성: 대기 인스턴스를 생성하지 않음을 선택한 상태로 둡니다.
- 연결:
 - 컴퓨팅 리소스 EC2 컴퓨팅 리소스에 연결하지 않음을 그대로 둡니다.
 - 네트워크 유형: IPv4를 선택한 상태로 둡니다.
 - 가상 프라이빗 클라우드: DMSVPC-vpc
 - 퍼블릭 액세스: 예. AWS Schema Conversion Tool를 사용하려면 퍼블릭 액세스를 활성화해야 합니다.
 - 가용 영역: us-west-2a
 - 나머지 설정은 그대로 둡니다.
- 데이터베이스 인증: 암호 인증을 선택한 상태로 둡니다.
- 모니터링에서 Performance Insights 켜기 확인란의 선택을 취소합니다. 추가 구성 섹션을 확장하고 향상된 모니터링 활성화 확인란의 선택을 취소합니다.
- 추가 구성을 확장합니다.
 - 데이터베이스 옵션에서 초기 데이터베이스 이름에 **dms_sample**을 입력합니다.
 - DB 파라미터 그룹에서 dms-mariadb-parameters를 선택합니다.
 - 옵션 그룹에서 default:mariadb-10-6을 선택한 상태로 둡니다.
 - 백업에서 다음을 수행합니다.
 - 자동 백업 활성화를 선택한 상태로 둡니다. 지속적 복제를 지원하려면 소스 데이터베이스에 자동 백업이 활성화되어 있어야 합니다.
 - 백업 보존 기간으로 1일을 선택합니다.
 - 백업 기간에서 기본 설정 없음을 선택된 상태로 둡니다.

- 스냅샷에 태그 복사 확인란의 선택을 취소합니다.
- 다른 AWS 리전에서 복제 활성화를 선택하지 않은 상태로 둡니다.
- 암호화에서 암호화 활성화 확인란의 선택을 취소합니다.
- 로그 내보내기 섹션은 그대로 둡니다.
- 유지 관리에서 자동 마이너 버전 업그레이드 활성화 확인란의 선택을 취소하고 유지 관리 기간 설정을 기본 설정 없음으로 유지합니다.
- 삭제 방지 활성화를 선택하지 않은 상태로 둡니다.

4. 데이터베이스 생성을 선택합니다.

대상 Amazon RDS 데이터베이스 생성

이전 절차를 반복하되 다음과 같이 변경하여 대상 Amazon RDS 데이터베이스를 생성합니다.

대상 RDS for PostgreSQL 데이터베이스를 생성하려면

1. 이전 절차의 1 및 2단계를 반복합니다.
2. 데이터베이스 생성 페이지에서 다음을 제외하고 동일한 옵션을 설정합니다.
 - a. 엔진 옵션에서 PostgreSQL을 선택합니다.
 - b. 버전의 경우 사용 가능한 PostgreSQL 16 버전을 선택합니다
 - c. DB 인스턴스 식별자에 **dms-postgresql**을 입력합니다.
 - d. 마스터 사용자 이름은 **postgres**을 선택된 상태로 둡니다.
 - e. DB 파라미터 그룹에서 dms-postgresql-parameters를 선택합니다.
 - f. 자동 백업 활성화 확인란의 선택을 취소합니다.
3. 데이터베이스 생성을 선택합니다.

Amazon EC2 클라이언트 생성

이 섹션에서는 Amazon EC2 클라이언트를 생성합니다. 이 클라이언트를 사용하여 복제할 데이터로 소스 데이터베이스를 채웁니다. 또한 이 클라이언트를 사용하여 대상 데이터베이스에서 쿼리를 실행하여 복제를 확인할 수 있습니다.

Amazon EC2 클라이언트를 사용하여 데이터베이스에 액세스하면 인터넷을 통해 데이터베이스에 액세스하는 것에 비해 다음과 같은 이점이 있습니다.

- 데이터베이스에 대한 액세스를 동일한 VPC에 있는 클라이언트로만 제한할 수 있습니다.
- 이 자습서에서 사용하는 도구는 Amazon Linux 2023에서 작동하고 설치하기도 쉽다는 것을 확인했으므로 이 자습서에서는 이 도구를 권장합니다.
- VPC의 구성 요소 간 데이터 작업은 일반적으로 인터넷을 통한 작업보다 성능이 좋습니다.

Amazon EC2 클라이언트를 생성 및 구성하여 소스 데이터베이스를 채우려면

1. <https://console.aws.amazon.com/ec2/>에서 Amazon EC2 콘솔을 엽니다.
 2. 대시보드에서 인스턴스 시작을 선택합니다.
 3. 인스턴스 시작 페이지에서 다음 값을 입력합니다.
 - a. 이름 및 태그 섹션에서 이름에 **DMSClient**를 입력합니다.
 - b. 애플리케이션 및 OS 이미지(Amazon Machine Image) 섹션에서 설정을 그대로 유지합니다.
 - c. 인스턴스 유형 섹션에서 t2.xlarge를 선택합니다.
 - d. 키 페어(로그인) 섹션에서 새 키 페어 생성을 선택합니다.
 - e. 키 페어 생성 페이지에서 다음을 입력합니다.
 - Key pair name: **DMSKeyPair**
 - 키 페어 유형: RSA를 그대로 둡니다.
 - 프라이빗 키 파일 형식: MacOS 또는 Linux의 OpenSSH에는 pem을 선택하고, Windows의 PuTTY에는 ppk를 선택합니다.
- 메시지가 표시되면 키 파일을 저장합니다.
- 
- Note
- 새 키 페어를 생성하는 대신 기존 Amazon EC2 키 페어를 사용할 수도 있습니다.
- f. 네트워크 설정 섹션에서 편집을 선택합니다. 다음 설정을 선택합니다.
 - VPC - 필수: DMSVPC-vpc VPC에 대해 적어둔 ID가 있는 VPC를 선택합니다.
 - 서브넷에서 첫 번째 퍼블릭 서브넷을 선택합니다.
 - 퍼블릭 IP 자동 할당: 활성화를 선택합니다.

나머지 설정은 그대로 두고 인스턴스 시작을 선택합니다.

소스 데이터베이스 채우기

이 섹션에서는 나중에 사용할 소스 및 대상 데이터베이스의 엔드포인트를 찾고 다음 도구를 사용하여 소스 데이터베이스를 채웁니다.

- Git - 소스 데이터베이스를 채우는 스크립트를 다운로드합니다.
- MariaDB 클라이언트 - 이 스크립트를 실행합니다.

엔드포인트 가져오기

나중에 사용할 수 있도록 RDS for MariaDB 및 RDS for PostgreSQL DB 인스턴스의 엔드포인트를 찾아 적어둡니다.

DB 인스턴스 엔드포인트를 찾으려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/rds/> Amazon RDS 콘솔을 엽니다.
2. 탐색 창에서 데이터베이스를 선택합니다.
3. dms-mariadb 데이터베이스를 선택하고 데이터베이스의 엔드포인트 값을 적어둡니다.
4. dms-postgresql 데이터베이스에 대해 이전 단계를 반복합니다.

소스 데이터베이스 채우기

그런 다음 클라이언트 인스턴스에 연결하고, 필요한 소프트웨어를 설치하고, Git에서 AWS 샘플 데이터베이스 스크립트를 다운로드하고, 스크립트를 실행하여 소스 데이터베이스를 채웁니다.

소스 데이터베이스를 채우려면

1. 이전 단계에서 저장한 호스트 이름 및 퍼블릭 키를 사용하여 클라이언트 인스턴스에 연결합니다.

Amazon EC2 인스턴스에 연결에 대한 자세한 내용은 Amazon EC2 사용 설명서에서 [인스턴스 액세스](#)를 참조하세요.

Note

PuTTY를 사용하는 경우 비활성 상태로 인해 연결이 시간 초과되지 않도록 연결 설정 페이지에서 TCP keepalive를 활성화합니다.

2. Git, MariaDB 및 PostgreSQL을 설치합니다. 필요에 따라 설치를 확인합니다.

```
$ sudo yum install git
$ sudo dnf install mariadb105
$ sudo dnf install postgresql15
```

3. 다음 명령을 실행하여 GitHub에서 데이터베이스 생성 스크립트를 다운로드합니다.

```
git clone https://github.com/aws-samples/aws-database-migration-samples.git
```

4. 디렉토리를 aws-database-migration-samples/mysql/sampledб/v1/로 변경합니다.
5. 다음 명령을 실행합니다. 이전에 적어둔 소스 RDS 인스턴스의 엔드포인트를 제공합니다(예: dms-mariadb.cdv5fbeyiy4e.us-east-1.rds.amazonaws.com).

```
mysql -h dms-mariadb.abcdefg hij01.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
dms_sample < ~/aws-database-migration-samples/mysql/sampledб/v1/install-rds.sql
```

6. 데이터베이스 생성 스크립트를 실행합니다. 이 스크립트가 스키마를 생성하고 데이터를 채우는 데 최대 45분이 걸립니다. 스크립트에 표시되는 오류 및 경고는 무시해도 됩니다.

를 사용하여 소스 스키마를 대상 데이터베이스로 마이그레이션 AWS SCT

소스 데이터베이스 스키마를 변환하려면 DMS Schema Conversion을 사용하는 것이 좋습니다. 자세한 내용은 [DMS Schema Conversion 시작하기](#) 단원을 참조하십시오. 레거시를 사용하려면 아래 단계를 AWS Schema Conversion Tool따르세요.

를 사용하여 소스 스키마를 대상 데이터베이스로 마이그레이션하려면 AWS SCT

1. 를 설치합니다 AWS Schema Conversion Tool. 자세한 내용은 AWS 사용 설명서의 [AWS SCT설치, 확인 또는 업데이트](#)를 참조하세요.

MySQL 및 PostgreSQL용 JDBC 드라이버를 다운로드할 때 도구에서 드라이버 위치를 묻는 메시지가 표시될 경우를 대비하여 드라이버를 저장한 위치를 적어두세요.

2. 를 엽니다 AWS Schema Conversion Tool. 파일을 선택하고 새 프로젝트를 선택합니다.
3. 새 프로젝트 창에서 다음 값을 설정합니다.

- 프로젝트 이름을 **DMSProject**로 설정합니다.

- AWS SCT 프로젝트를 기본 폴더에 저장하려면 위치를 그대로 유지합니다.

확인을 선택합니다.

4. 소스 추가를 선택하여 소스 MySQL 데이터베이스를 프로젝트에 추가한 후 MySQL을 선택하고 다음을 선택합니다.

5. 소스 추가 페이지에서 다음 값을 설정합니다.

- 연결 이름: **source**
- 서버 이름: 이전에 적어둔 MySQL 데이터베이스의 엔드포인트를 입력합니다.
- 서버 포트: **3306**
- 사용자 이름: **admin**
- 암호: **changeit**

6. 대상 추가를 선택하여 대상 Amazon RDS for PostgreSQL 데이터베이스를 프로젝트에 추가한 다음 Amazon RDS for PostgreSQL을 선택합니다. 다음을 선택합니다.

7. 대상 추가 페이지에서 다음 값을 설정합니다.

- 연결 이름: **target**
- 서버 이름: 이전에 적어둔 PostgreSQL 데이터베이스의 엔드포인트를 입력합니다.
- 서버 포트: **5432**
- 데이터베이스: PostgreSQL 데이터베이스의 이름을 입력합니다.
- 사용자 이름: **postgres**
- 암호: **changeit**

8. 왼쪽 창의 스키마에서 dms_sample을 선택합니다. 오른쪽 창에서 대상 Amazon RDS for PostgreSQL 데이터베이스를 선택합니다. 매핑 생성을 선택합니다. 단일 AWS SCT 프로젝트에 여러 매핑 규칙을 추가할 수 있습니다. 매핑 규칙에 대한 자세한 내용은 [매핑 규칙 생성](#)을 참조하세요.

9. 기본 보기를 선택합니다.

10. 왼쪽 창의 스키마에서 dms_sample을 선택합니다. 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) 스키마 변환을 선택합니다. 작업을 확인합니다.

도구가 스키마를 변환하면 오른쪽 창에 dms_sample 스키마가 나타납니다.

11. 오른쪽 창의 스키마에서 dms_sample의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) 데이터베이스에 적용을 선택합니다. 작업을 확인합니다.

스키마 마이그레이션이 완료되었는지 확인합니다. 다음 단계를 수행합니다.

스키마 마이그레이션을 확인하려면

1. Amazon EC2 클라이언트에 연결합니다.
2. 다음 명령을 사용하여 PSQL 클라이언트를 시작합니다. PostgreSQL 데이터베이스 엔드포인트를 지정하고 메시지가 표시되면 데이터베이스 암호를 입력합니다.

```
psql \
--host=dms-postgresql.abcdefg12345.us-west-2.rds.amazonaws.com \
--port=5432 \
--username=postgres \
--password \
--dbname=dms_sample
```

3. (비어 있는) 테이블 중 하나를 쿼리하여가 스키마를 올바르게 AWS SCT 적용했는지 확인합니다.

```
dms_sample=> SELECT * from dms_sample.player;
 id | sport_team_id | last_name | first_name | full_name
-----+-----+-----+-----+-----
(0 rows)
```

에 대한 복제 설정 AWS Database Migration Service

이 항목에서는 소스 데이터베이스와 대상 데이터베이스 간의 복제를 설정합니다.

1단계: AWS DMS 콘솔을 사용하여 복제 인스턴스 생성

작업을 시작하려면 복제 인스턴스를 AWS DMS 생성합니다.

복제 인스턴스는 소스 엔드포인트와 대상 엔드포인트 간에 실제 데이터 마이그레이션을 수행합니다. 소스 데이터베이스에서 대상 데이터베이스로 데이터를 마이그레이션하는 작업을 수행하려면 인스턴스에 충분한 스토리지 및 처리 능력이 필요합니다. 이 복제 인스턴스의 크기는 마이그레이션할 데이터의 양과 인스턴스가 수행해야 하는 태스크에 따라 달라집니다. 복제 인스턴스에 대한 자세한 내용은 [AWS DMS 복제 인스턴스 작업](#) 섹션을 참조하세요.

Create replication instance [Info](#)

① Configuration best practices

[View details](#)Click [View details](#) button to review configuration best practices.

Settings

Name

The name must be unique among all of your replication instances in the current AWS region.

Replication instance name must not start with a numeric value

Descriptive Amazon Resource Name (ARN) - optional

A friendly name to override the default DMS ARN. You cannot modify it after creation.

Description - optional

The description must only have unicode letters, digits, whitespace, or one of these symbols: _:/~+@. 1000 maximum character.

Instance configuration [Info](#)[Estimate instance class and storage](#)Instance class [Info](#)dms.t3.medium
2 vCPUs 4 GiB Memory☐ Include previous-generation instance classesAllocated storage (GiB) [Info](#)

Choose the amount of storage space you want for your replication instance. AWS DMS uses this storage for log files and cached transactions while replication tasks are in progress.

50

Engine version

Choose an AWS DMS version to run on your replication instance. For more details, see the [AWS DMS release notes](#). For information about DMS version support, see [AWS DMS support lifecycle policy](#).

3.5.4

☐ Include Beta DMS versionsHigh Availability [Info](#)

The Multi-AZ option deploys a primary replication instance in one Availability Zone (AZ) and a standby in another AZ. The Single-AZ option deploys a single replication instance in one AZ. Billing is based on DMS pricing.

Choose a failover setting



콘솔을 사용하여 복제 인스턴스를 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 복제 인스턴스를 선택하고, 복제 인스턴스 생성을 선택합니다.
3. 복제 인스턴스 생성 페이지에서 복제 인스턴스 구성을 지정합니다.
 - a. 이름에 **DMS-instance**를 입력합니다.
 - b. 설명에 복제 인스턴스에 대한 간단한 설명을 입력합니다(선택 사항).
 - c. 인스턴스 클래스에서 dms.t3.medium을 선택한 상태로 둡니다.

인스턴스에는 마이그레이션을 위한 충분한 스토리지, 네트워킹 및 처리 능력이 필요합니다. 인스턴스 클래스를 선택하는 방법에 대한 자세한 내용은 [마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택](#) 섹션을 참조하세요.

- d. 엔진 버전에서 기본값을 그대로 사용합니다.

- e. 다중 AZ에서 개발 또는 테스트 워크로드(단일 AZ)를 선택합니다.
- f. 할당된 스토리지(GiB)에서 기본값인 50GiB를 그대로 사용합니다.

에서 AWS DMS스토리지는 대부분 로그 파일과 캐시된 트랜잭션에 사용됩니다. 캐시 트랜잭션에서 스토리지는 캐시된 트랜잭션을 디스크에 기록해야 할 때에만 사용됩니다. 따라서 AWS DMS 는 상당한 양의 스토리지를 사용하지 않습니다.

- g. 네트워크 유형에서 IPv4를 선택합니다.
 - h. VPC에서 DMSVPC를 선택합니다.
 - i. 복제 서브넷 그룹에서 현재 선택한 복제 서브넷 그룹을 그대로 둡니다.
 - j. 퍼블릭 액세스 가능 확인란의 선택을 취소합니다.
4. 필요한 경우 고급 보안 및 네트워크 구성 탭을 선택하여 네트워크 및 암호화 설정에 대한 값을 설정합니다.
- a. 가용 영역에서 us-west-2a를 선택합니다.
 - b. VPC 보안 그룹에서 기본 보안 그룹을 선택합니다(이미 선택되지 않은 경우).
 - c. AWS KMS key에서 (기본값) aws/dms를 선택한 상태로 둡니다.
5. 유지 관리 탭의 설정을 그대로 둡니다. 기본값은 각 AWS 리전의 8시간 블록 시간 중에서 임의로 선택한 30분 기간으로, 요일에 무작위로 발생합니다.
6. 생성(Create)을 선택합니다.

AWS DMS 는 마이그레이션을 수행할 복제 인스턴스를 생성합니다.

2단계: 소스 및 대상 엔드포인트 지정

복제 인스턴스가 생성되는 동안 앞서 생성한 Amazon RDS 데이터베이스를 위한 소스 및 대상 데이터 스토어를 지정할 수 있습니다. 각 엔드포인트를 개별적으로 생성합니다.

☰ [DMS](#) > [Migrate or replicate](#) > [Endpoints](#) > Create endpoint

Create endpoint [Info](#)

Endpoint type [Info](#)



Source endpoint

A source endpoint allows AWS DMS to read data from a database (on-premises or in the cloud), or from other data source such as Amazon S3.



Target endpoint

A target endpoint allows AWS DMS to write data to a database, or to other data stores such as Amazon DynamoDB or Kinesis.



Select RDS DB instance

Choose this option if the endpoint is an Amazon RDS DB instance. It provides a list of available RDS Instances from the current region.

Endpoint configuration

Endpoint identifier [Info](#)

A label for the endpoint to help you identify it.

ProdEndpoint

Descriptive Amazon Resource Name (ARN) - optional

A friendly name to override the default DMS ARN. You cannot modify it after creation.

Friendly-ARN-name



Use this source endpoint for migrating data to AWS Glue Elastic Views.

With this option, you can use this endpoint only for migration to target endpoints for Amazon Glue Elastic Views. This migration requires a properly configured IAM role called "dms-elastic-views-role".

Source engine

The type of database engine this endpoint is connected to. [Learn more](#)

Choose an engine

▶ Endpoint settings

▶ KMS key

▶ Tags

Add tags to your DMS resources to organize and track your DMS costs.

▶ Test endpoint connection - optional

Choose the replication instance to test the network and database connectivity for migration.

AWS DMS 콘솔을 사용하여 소스 엔드포인트 및 데이터베이스 엔드포인트를 지정하려면

1. 콘솔의 탐색 창에서 엔드포인트를 선택하고 엔드포인트 생성을 선택합니다.
2. 엔드포인트 생성 페이지에서 소스 엔드포인트 유형을 선택합니다. RDS DB 인스턴스 선택 확인란을 선택하고 dms-mariadb 인스턴스를 선택합니다.
3. 엔드포인트 구성 섹션에서 엔드포인트 식별자로 **dms-mysql-source**를 입력합니다.
4. 소스 엔진에서 MySQL을 선택한 상태로 둡니다.

5. 엔드포인트 데이터베이스 액세스에서 수동으로 액세스 정보 제공을 선택합니다. 포트, 보안 소켓 계층(SSL) 모드, 사용자 이름 및 암호가 올바른지 확인합니다.
6. 엔드포인트 연결 테스트(선택 사항) 탭을 선택합니다. VPC에서 DMSVPC를 선택합니다.
7. 복제 인스턴스에서 dms-instance를 선택한 상태로 둡니다.
8. 테스트 실행을 선택합니다.

테스트 실행을 선택하면는 사용자가 제공한 세부 정보가 포함된 엔드포인트를 AWS DMS 생성하고 연결합니다. 연결이 실패할 경우 엔드포인트 정의를 편집하고 연결을 다시 테스트하세요. 엔드포인트를 수동으로 삭제할 수도 있습니다.

9. 테스트에 성공하면 엔드포인트 생성을 선택합니다.
10. AWS DMS 콘솔을 사용하여 대상 데이터베이스 엔드포인트를 지정합니다. 이렇게 하려면 다음 설정을 사용하여 이전 단계를 반복합니다.
 - 엔드포인트 유형: 대상 엔드포인트
 - RDS 인스턴스: dms-postgresql
 - 엔드포인트 식별자: **dms-postgresql-target**
 - 대상 엔진: **PostgreSQL**을 선택한 상태로 둡니다.

엔드포인트에 대한 모든 정보 제공을 마치면는 데이터베이스 마이그레이션 중에 사용할 소스 및 대상 엔드포인트를 AWS DMS 생성합니다.

3단계: 태스크 생성 및 데이터 마이그레이션

이 단계에서는 생성한 데이터베이스 간에 데이터를 마이그레이션하는 태스크를 생성합니다.

Create task

① Click View details to review configuration best practices.

View details

Task configuration

Task identifier

Provide a name for this task.

Type a unique identifier for the task

Should begin with a letter; should contain only ASCII letters, digits, and hyphens; and should not end with a hyphen or contain two consecutive hyphens.

Descriptive Amazon Resource Name (ARN) - optional

A friendly name to override the default DMS ARN. You cannot modify it after creation.

Friendly ARN name

Should begin with a letter; should contain only ASCII letters, digits, and hyphens; and shouldn't end with a hyphen or contain two consecutive hyphens.

Source database endpoint

Task supports different [source/target](#) pairs based on your task mode. To create a new endpoint, go to [Endpoints](#).

Choose a source database endpoint



Target database endpoint

Task supports different [source/target](#) pairs based on your task mode. To create a new endpoint, go to [Endpoints](#).

Choose a target database endpoint



Task mode

Provisioned

Select the amount of compute you want to dedicate to your migration or replication task

Serverless

DMS dynamically adjusts the level of compute needed for your migration or replication task

Provisioned instance

Choose a provisioned instance



Task type

Info

Migrate only

Migrate data from source to target once (Full load)

Migrate and replicate

Migrate data from source to target once and continue to replicate changes (Full load and CDC)

Replicate only

Replicate data from source to target now or at a specified milestone (CDC only)

Settings

Editing mode

Info

Wizard

You can enter only a subset of the available task settings.

JSON editor

You can enter all available task settings directly in JSON format.



마이그레이션 태스크를 생성하고 데이터베이스 마이그레이션을 시작하려면

- 콘솔 탐색 창에서 데이터베이스 마이그레이션 태스크를 선택한 다음 태스크 생성을 선택합니다. 데이터베이스 마이그레이션 태스크 생성 페이지가 열립니다.
- 태스크 구성 섹션에서 다음 태스크 옵션을 지정합니다.
 - 태스크 식별자: **dms-task**를 입력합니다.
 - 복제 인스턴스: 복제 인스턴스를 선택합니다(dms-instance-vpc-**vpc id**).
 - 소스 데이터베이스 엔드포인트: dms-mysql-source를 선택합니다.
 - 대상 데이터베이스 엔드포인트: dms-postgresql-target을 선택합니다.
 - 마이그레이션 유형: 기존 데이터 마이그레이션 및 진행 중 변경 사항 복제를 선택합니다.
- 태스크 설정 탭을 선택합니다. 다음 설정을 지정합니다.
 - 대상 테이블 준비 모드: 아무 작업 안 함
 - 전체 로드 완료 후 태스크 중지: 중지 안 함

- 테이블 매핑 탭을 선택하고 선택 규칙을 확장합니다. 선택 규칙 추가를 선택합니다. 다음 설정을 지정합니다.
 - 스키마: 스키마 입력
 - 스키마 이름: **dms_sample**
- 마이그레이션 작업 시작 구성 탭을 선택합니다. 작업을 시작하기 전에 평가를 실행하려면 마이그레이션 전 평가 시작(권장)을 선택합니다. 또는 생성 시 자동으로 시작을 선택하여 마이그레이션을 즉시 시작하거나 평가 실행만 선택하여 작업을 시작하지 않고 평가할 수 있습니다.

Post-creation actions - new

Actions to follow task creation

Choose an action to run automatically after creating the migration task. DMS recommends running premigration assessments to identify potential blockers and limitations that may cause migration failures or delays.

- ☒ **Run premigration assessments and then start task**
DMS will run required assessments, then start the task based on the task behavior you specified.
- ☐ **Run premigration assessments only**
DMS will run required assessments only. You can start your task from task actions menu later.
- ☐ **No automatic action**
You can create premigration assessments or start your task from task actions menu later.

Next steps after task creation

DMS will automatically create a service role and S3 bucket if they don't exist to store the assessment results. The assessments usually complete within 1-2 minutes but in some cases they may take up to 30 minutes. Task modifications are unavailable while assessments run. You can monitor assessment progress, review and address any identified issues before starting your task in task details. [Learn more](#)

Task start behavior

Choose task start conditions:

- ☐ **Don't start task if assessments identify issues**
When this setting is checked (recommended), DMS won't start your task if premigration assessments identify issues.

Available assessments

- 작업 생성을 선택합니다.

AWS DMS 그런 다음은 마이그레이션 작업을 생성하고 시작합니다. 초기 데이터베이스 복제는 약 10 분이 걸립니다. AWS DMS 가 데이터 마이그레이션을 완료하기 전에 자습서의 다음 단계를 수행해야 합니다.

4단계: 복제 테스트

이 단원에서는 초기 복제 도중 및 이후에 소스 데이터베이스에 데이터를 삽입하고 대상 데이터베이스에서 삽입된 데이터를 쿼리합니다.

복제를 테스트하려면

- 데이터베이스 마이그레이션 태스크의 상태가 실행 중으로 표시되지만 이전 단계에서 시작한 초기 데이터베이스 복제가 완료되지 않았는지 확인합니다.

2. Amazon EC2 클라이언트에 연결하고 다음 명령을 사용하여 MySQL 클라이언트를 시작합니다. MySQL 데이터베이스 엔드포인트를 제공합니다.

```
mysql -h dms-mysql.abcdefg12345.us-west-2.rds.amazonaws.com -P 3306 -u admin -pchangeit dms_sample
```

3. 다음 명령을 실행하여 소스 데이터베이스에 레코드를 삽입합니다.

```
MySQL [dms_sample]> insert person (full_name, last_name, first_name) VALUES ('Test User1', 'User1', 'Test');
Query OK, 1 row affected (0.00 sec)
```

4. MySQL 클라이언트를 종료합니다.

```
MySQL [dms_sample]> exit
Bye
```

5. 복제가 완료되기 전에 대상 데이터베이스에서 새 레코드를 쿼리합니다.

다음 명령으로 Amazon EC2 인스턴스에서 대상 데이터베이스 엔드포인트를 제공하여 대상 데이터베이스에 연결합니다.

```
psql \
--host=dms-postgresql.abcdefg12345.us-west-2.rds.amazonaws.com \
--port=5432 \
--username=postgres \
--password \
--dbname=dms_sample
```

메시지가 표시되면 암호(**changeit**)를 제공합니다.

6. 복제가 완료되기 전에 대상 데이터베이스에서 새 레코드를 쿼리합니다.

```
dms_sample=> select * from dms_sample.person where first_name = 'Test';
 id | full_name | last_name | first_name
-----+-----+-----+-----
(0 rows)
```

7. 마이그레이션 태스크가 실행되는 동안 데이터베이스 마이그레이션의 진행 상황을 모니터링할 수 있습니다.

- DMS 콘솔 탐색 창에서 데이터베이스 마이그레이션 태스크를 선택합니다.

- dms-task를 선택합니다.
- 테이블 통계를 선택합니다.

모니터링에 대한 자세한 내용은 [AWS DMS 작업 모니터링](#) 섹션을 참조하세요.

- 복제가 완료되면 대상 데이터베이스에서 새 레코드를 다시 쿼리합니다. 초기 복제가 완료된 후 새 레코드를 AWS DMS 마이그레이션합니다.

```
dms_sample=> select * from dms_sample.person where first_name = 'Test';
   id   | full_name | last_name | first_name
-----+-----+-----+-----
 7077784 | Test User1 | User1     | Test
(1 row)
```

- psql 클라이언트를 종료합니다.

```
dms_sample=> quit
```

- 1단계를 반복하여 소스 데이터베이스에 다시 연결합니다.

- person 테이블에 다른 레코드를 삽입합니다.

```
MySQL [dms_sample]> insert person (full_name, last_name, first_name) VALUES ('Test
User2', 'User2', 'Test');
Query OK, 1 row affected (0.00 sec)
```

- 3단계 및 4단계를 반복하여 소스 데이터베이스의 연결을 끊고 대상 데이터베이스에 연결합니다.

- 대상 데이터베이스에서 복제된 데이터를 다시 쿼리합니다.

```
dms_sample=> select * from dms_sample.person where first_name = 'Test';
   id   | full_name | last_name | first_name
-----+-----+-----+-----
 7077784 | Test User1 | User1     | Test
 7077785 | Test User2 | User2     | Test
(2 rows)
```

5단계: AWS DMS 리소스 정리

시작하기 자습서를 완료하면 생성한 리소스를 삭제할 수 있습니다. AWS 콘솔을 사용하여 제거할 수 있습니다. 복제 인스턴스 및 엔드포인트를 삭제하기 전에 마이그레이션 태스크를 삭제해야 합니다.

콘솔을 사용하여 마이그레이션 태스크를 삭제하려면

1. AWS DMS 콘솔 탐색 창에서 데이터베이스 마이그레이션 작업을 선택합니다.
2. dms-task를 선택합니다.
3. 작업, 삭제를 선택합니다.

Console을 사용하여 복제 인스턴스를 삭제하려면

1. AWS DMS 콘솔 탐색 창에서 복제 인스턴스를 선택합니다.
2. DMS-instance를 선택합니다.
3. 작업, 삭제를 선택합니다.

AWS DMS 는 복제 인스턴스를 삭제하고 복제 인스턴스 페이지에서 제거합니다.

콘솔을 사용하여 엔드포인트를 제거하려면

1. AWS DMS 콘솔 탐색 창에서 엔드포인트를 선택합니다.
2. dms-mysql-source를 선택합니다.
3. 작업, 삭제를 선택합니다.

AWS DMS 리소스를 삭제한 후에는 다음 리소스도 삭제해야 합니다. 다른 서비스의 리소스를 삭제하는 데 도움이 필요하면 각 서비스의 설명서를 참조하세요.

- RDS 데이터베이스.
- RDS 데이터베이스 파라미터 그룹.
- RDS 서브넷 그룹.
- 데이터베이스 및 복제 인스턴스와 함께 생성된 모든 Amazon CloudWatch 로그
- Amazon VPC 및 Amazon EC2 클라이언트용으로 생성된 보안 그룹. launch-wizard-1 보안 그룹에 대한 기본값에서 인바운드 규칙을 제거해야 합니다. 이는 보안 그룹을 삭제하는 데 필요합니다.
- Amazon EC2 클라이언트.
- Amazon VPC.
- Amazon EC2 클라이언트용 Amazon EC2 키 페어

AWS DMS Fleet Advisor 지원 종료

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

AWS DMS Fleet Advisor 지원 종료 세부 정보

지원 종료 계획

신중한 고려 후 2026년 5월 20일부터 AWS DMS Fleet Advisor에 대한 지원을 종료하기로 결정했습니다. AWS DMS Fleet Advisor는 2025년 5월 20일부터 더 이상 신규 고객을 받지 않습니다. 2025년 5월 20일 AWS DMS 이전에 가입한 계정이 있는 기존 고객은 AWS DMS Fleet Advisor 및 해당 기능을 계속 사용할 수 있습니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor를 사용할 수 없습니다.

Fleet Advisor를 계획하고 마이그레이션하는 데 도움이 되도록 12개월 동안 점진적으로 제공을 중단할 예정입니다. 알아두어야 할 중요한 두 날짜는 2025년 5월 20일과 2026년 5월 20일입니다.

2026년 5월 20일에 서비스를 중단할 예정입니다. AWS DMS Fleet Advisor 기능을 다운로드하거나 사용할 수 없습니다. 해당 시점부터 AWS DMS Fleet Advisor에 대한 지원을 더 이상 사용할 수 없습니다. 2026년 5월 20일 이전에 데이터베이스 평가 프로젝트를 AWS 마이그레이션 평가자로 마이그레이션하는 것이 좋습니다.

FAQ

“현재 고객”의 마감점은 무엇입니까?

감지기를 포함하여 지난 90일 동안 AWS DMS Fleet Advisor를 사용한 계정 IDs의 허용 목록은 서비스 내에서 유지됩니다. 기존 고객이 서비스 사용에 어려움을 겪고 있는 경우 [지원 사례를 생성](#)합니다.

서비스에 대한 “액세스”란 무엇을 AWS 의미하나요?

기존 AWS DMS Fleet Advisor 고객은 이전에 할 수 있었던 모든 작업을 수행할 수 있습니다. 유일한 변경 사항은 기존 고객이 AWS DMS Fleet Advisor 또는 [Data Collectors 도구](#), [OS 및 데이터베이스 서버 검색](#), [데이터 수집](#), [대상 권장](#) 사항 생성 등 기능을 사용할 수 없다는 것입니다.

기존 AWS DMS Fleet Advisor 고객은 [Data Collectors 도구](#), [OS 및 데이터베이스 서버 검색](#), [데이터 수집](#) 및 [대상 권장 사항](#) 생성을 다운로드할 수 있나요?

예. 문제가 있는 경우 [지원 사례를 생성](#)합니다.

AWS DMS Fleet Advisor를 더 이상 제공하지 않는 이유는 무엇입니까?

AWS DMS Fleet Advisor의 대안으로 고객이 데이터베이스 마이그레이션 평가 프로젝트에 마이그레이션 [평가자](#)를 선호하는 것으로 확인되었습니다. 마이그레이션 평가자는 비즈니스 이해관계자와 기술 이해관계자 모두에게 AWS 클라우드에서 온프레미스 워크로드를 실행하는 예상 비용에 대한 가시성을 제공합니다. 비즈니스 이해관계자는 인프라 및 소프트웨어 라이선스별로 비용이 분류된 사용 패턴을 AWS 기반으로에서 리호스팅할 예상 절감액을 강조하는 1페이지 요약을 다운로드할 수 있습니다. 더 많은 기술 사용자를 위해 자세한 서버별 및 per-SQL-server 데이터도 사용할 수 있습니다. 이 내보내기는 온프레미스 검색 데이터(서버 하드웨어 프로비저닝, SQL 구성 및 리소스 사용률)를 권장 사항과 결합합니다.

현재 내 옵션은 무엇입니까?

모든 신규 및 진행 중인 데이터베이스 마이그레이션 평가 프로젝트에 [AWS 마이그레이션 평가자](#)를 사용하는 것이 좋습니다. Fleet Advisor의 Data Collectors 도구를 이미 다운로드한 경우 계속 사용할 수 있습니다. 그러나 이러한 도구에 대한 지원 및 업데이트는 2025년 5월 20일 이후에는 중단됩니다. 기존 Fleet Advisor 사용자의 경우 Fleet Advisor로 진행 중인 평가를 완료한 다음 마이그레이션 평가기로 전환하여 중요한 패치, 보안 업데이트, 성능 개선 및 새로운 기능을 활용하는 것이 좋습니다.

AWS DMS Fleet Advisor를 사용한 마이그레이션을 위한 데이터베이스 검색 및 평가

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS Fleet Advisor를 사용하여 여러 데이터베이스 환경에서 메타데이터 및 성능 지표를 수집할 수 있습니다. 이렇게 수집된 지표는 데이터 인프라에 대한 인사이트를 제공합니다. [DMS Fleet Advisor](#)는 모든 컴퓨터에 설치할 필요 없이 하나 이상의 중앙 위치에서 온프레미스 데이터베이스 및 분석 서버의 메타데이터와 지표를 수집합니다. 현재 DMS Fleet Advisor는 Microsoft SQL Server, MySQL, Oracle 및 PostgreSQL 데이터베이스 서버에 대한 검색 및 지표 수집을 지원합니다.

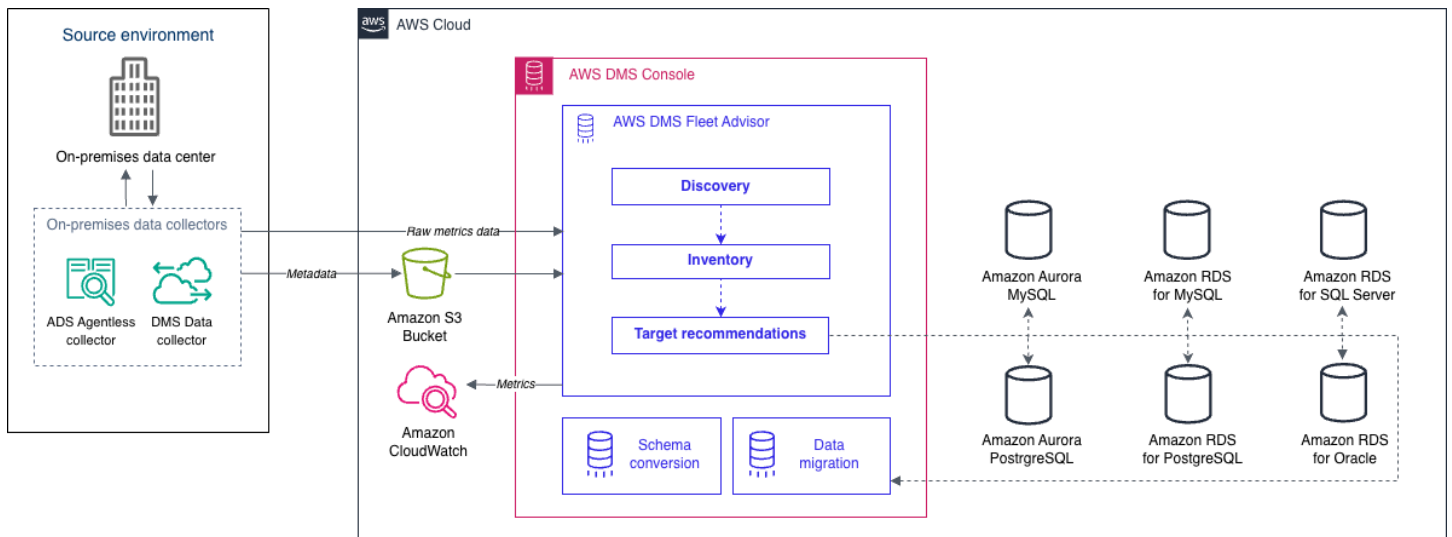
네트워크에서 검색된 데이터를 기반으로 인벤토리를 작성하여 추가 데이터 수집을 위한 데이터베이스 서버 목록을 정의할 수 있습니다. 각 서버, 데이터베이스 및 스키마에 대한 정보를 AWS DMS 수집한 후 의도한 데이터베이스 마이그레이션의 타당성을 분석할 수 있습니다.

로 마이그레이션하려는 인벤토리의 데이터베이스의 경우 AWS 클라우드 DMS Fleet Advisor는 적절한 크기의 대상 권장 사항을 생성합니다. 대상 권장 사항을 생성하기 위해 DMS Fleet Advisor는 데이터 수집기의 지표와 기본 설정을 고려합니다. DMS Fleet Advisor가 권장 사항을 생성한 후에는 각 대상 데이터베이스 구성에 대한 세부 정보를 볼 수 있습니다. 조직의 데이터베이스 엔지니어 및 관리자는 DMS Fleet Advisor 대상 권장 사항을 사용하여 온프레미스 데이터베이스를 AWS로 마이그레이션할 계획을 수립할 수 있습니다. 사용 가능한 다양한 마이그레이션 옵션을 탐색하고 이러한 권장 사항을 로 내보내 비용을 추가로 최적화 AWS Pricing Calculator 할 수 있습니다.

지원되는 소스 데이터베이스의 목록은 [DMS Fleet Advisor가 지원하는 소스](#) 섹션을 참조하세요.

DMS Fleet Advisor가 대상 권장 사항을 생성하는 데 사용하는 데이터베이스의 목록은 [DMS Fleet Advisor가 지원하는 대상](#) 섹션을 참조하세요. DMS Fleet Advisor는 예를 들어 소스 Oracle에서 대상 Oracle 데이터베이스까지 유사한 권장 사항을 생성합니다. DMS Fleet Advisor는 소스 Oracle 또는 Microsoft SQL Server에서 대상 RDS for PostgreSQL 또는 Aurora PostgreSQL 데이터베이스로 마이그레이션하는 등 이질적인 권장 사항도 생성합니다.

다음 다이어그램은 AWS DMS Fleet Advisor 대상 권장 사항 프로세스를 보여줍니다.



다음 주제를 사용하여 AWS DMS Fleet Advisor 사용 방법을 더 잘 이해할 수 있습니다.

지원됨 AWS 리전

다음에서 DMS Fleet Advisor를 사용할 수 있습니다 AWS 리전.

리전 이름	리전
미국 동부(버지니아 북부)	us-east-1
미국 동부(오하이오)	us-east-2
미국 서부(캘리포니아 북부)	us-west-1
미국 서부(오리건)	us-west-2
캐나다(중부)	ca-central-1
캐나다 서부(캘거리)	ca-west-1
Asia Pacific (Hong Kong)	ap-east-1
아시아 태평양(도쿄)	ap-northeast-1
아시아 태평양(서울)	ap-northeast-2

리전 이름	리전
아시아 태평양(오사카)	ap-northeast-3
아시아 태평양(뭄바이)	ap-south-1
아시아 태평양(싱가포르)	ap-southeast-1
아시아 태평양(시드니)	ap-southeast-2
아시아 태평양(자카르타)	ap-southeast-3
아시아 태평양(멜버른)	ap-southeast-4
아시아 태평양(하이데라바드)	ap-south-2
유럽(프랑크푸르트)	eu-central-1
유럽(취리히)	eu-central-2
유럽(스톡홀름)	eu-north-1
유럽(스페인)	eu-south-2
유럽(아일랜드)	eu-west-1
유럽(런던)	eu-west-2
유럽(파리)	eu-west-3
유럽(밀라노)	eu-south-1
이스라엘(텔아비브)	il-central-1
남아메리카(상파울루)	sa-east-1
중동(바레인)	me-south-1
중동(UAE)	me-central-1
Africa (Cape Town)	af-south-1

DMS Fleet Advisor를 사용하는 방법 이해

Important

지원 종료 공지: 2026년 5월 20일에 AWS는 AWS Database Migration Service Fleet Advisor에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS Fleet Advisor를 사용하여 AWS 클라우드로 마이그레이션할 소스 온프레미스 데이터베이스를 검색할 수 있습니다. 그러면 각 온프레미스 데이터베이스에 적합한 마이그레이션 대상을 AWS 클라우드에서 결정할 수 있습니다. 다음 워크플로를 사용하여 소스 데이터베이스의 인벤토리를 만들고 대상 권장 사항을 생성하세요.

1. Amazon S3 버킷, IAM 정책, 역할 및 사용자를 생성합니다. 자세한 내용은 [필요한 리소스 생성](#) 섹션을 참조하세요.
2. DMS 데이터 수집기에 필요한 최소 권한을 가진 데이터베이스 사용자를 생성하세요. 자세한 내용은 [데이터베이스 사용자 생성](#) 섹션을 참조하세요.
3. 데이터 수집기를 생성하고 다운로드합니다. 자세한 내용은 [데이터 수집기 생성](#) 섹션을 참조하세요.
4. 로컬 환경에 데이터 수집기를 설치하세요. 그런 다음, 수집된 데이터를 DMS Fleet Advisor로 전송할 수 있도록 데이터 수집기를 구성하세요. 자세한 내용은 [데이터 수집기 설치](#) 섹션을 참조하세요.
5. 데이터 환경의 OS 및 데이터베이스 서버를 확인해 보세요. 자세한 내용은 [OS 및 데이터베이스 서버 검색](#) 섹션을 참조하세요.
6. 데이터베이스 메타데이터와 리소스 사용률 지표를 수집하세요. 자세한 내용은 [데이터 수집](#) 섹션을 참조하세요.
7. 소스 데이터베이스와 스키마를 분석하세요. DMS Fleet Advisor는 데이터베이스의 대규모 평가를 실행하여 유사한 스키마를 식별합니다. 자세한 내용은 [AWS DMS Fleet Advisor에서 분석에 인벤토리 사용](#) 섹션을 참조하세요.
8. 소스 데이터베이스에 대한 대상 권장 사항의 로컬 사본을 생성, 확인 및 저장합니다. 자세한 내용은 [대상 권장 사항](#) 섹션을 참조하세요.

각 소스 데이터베이스의 마이그레이션 대상을 결정한 후에는 DMS Schema Conversion을 사용하여 데이터베이스 스키마를 새 플랫폼으로 변환할 수 있습니다. 그런 다음, 데이터를 마이그레이션하는 데

AWS DMS를 사용할 수 있습니다. 자세한 내용은 [DMS Schema Conversion을 사용하여 데이터베이스 스키마 변환](#) 및 [AWS Database Migration Service이란 무엇인가요?](#) 섹션을 참조하세요.

[이 동영상](#)은 DMS Schema Conversion 사용자 인터페이스를 소개하고 이 서비스의 핵심 구성 요소를 숙지하는 데 도움이 됩니다.

AWS DMS Fleet Advisor에 필요한 AWS 리소스 생성

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS Fleet Advisor는 인벤토리 정보를 전달 및 가져오고 DMS 데이터 수집기의 상태를 업데이트하기 위해 계정에 AWS 리소스 세트가 필요합니다.

처음으로 데이터를 수집하고 데이터베이스 및 스키마의 인벤토리를 생성하려면 먼저 다음과 같은 사전 조건을 완료해야 합니다.

Amazon S3 버킷과 IAM 리소스를 구성하려면 다음 중 하나를 수행합니다.

- [를 사용하여 Amazon S3 및 IAM 리소스 구성 CloudFormation](#)(권장).
- [에서 Amazon S3 및 IAM 리소스 구성 AWS Management Console](#)

를 사용하여 Amazon S3 및 IAM 리소스 구성 CloudFormation

CloudFormation 스택은 단일 단위로 관리할 수 있는 AWS 리소스 모음입니다. DMS Fleet Advisor에 필요한 리소스 생성을 간소화하려면 CloudFormation 템플릿 파일을 사용하여 CloudFormation 스택을 생성할 수 있습니다. 자세한 내용은 CloudFormation 사용 설명서 [의 CloudFormation 콘솔에서 스택 생성](#)을 참조하세요.

Note

이 섹션은 독립형 DMS Fleet Advisor 수집기 사용에만 적용됩니다. 단일 온프레미스 수집기를 사용하여 데이터베이스와 서버 모두에 대한 정보를 수집하는 방법에 대한 자세한 내용은

[AWS Application Discovery Service 사용 설명서](#)의 [Application Discovery Service Agentless Collector](#)를 참조하세요.

CloudFormation으로 생성된 Amazon S3 및 IAM 리소스

CloudFormation 템플릿을 사용하면 AWS 계정에 다음 리소스가 포함된 스택이 생성됩니다.

- dms-fleetadvisor-data-*accountId*-*region*이라는 Amazon S3 버킷
- FleetAdvisorCollectorUser-*region*이라는 IAM 사용자 이름
- FleetAdvisorS3Role-*region*이라는 IAM 서비스 역할
- FleetAdvisorS3Role-*region*-Policy라는 액세스 정책
- FleetAdvisorCollectorUser-*region*-Policy라는 액세스 정책
- AWSServiceRoleForDMSFleetAdvisor라는 IAM 서비스 연결 역할(SLR)

아래 나열된 단계에 따라 CloudFormation을 사용하여 리소스를 구성합니다.

- [1단계: CloudFormation 템플릿 파일 다운로드](#)
- [2단계: CloudFormation을 사용하여 Amazon S3 및 IAM 구성](#)

1단계: CloudFormation 템플릿 파일 다운로드

CloudFormation 템플릿이란 스택을 구성하는 AWS 리소스의 선언입니다. 템플릿은 JSON 파일로 저장됩니다.

CloudFormation 템플릿 파일을 다운로드하려면

1. 다음 링크 중 하나에 대한 컨텍스트(마우스 오른쪽 클릭) 메뉴를 열고 링크 저장을 선택합니다.
 - DMS Fleet Advisor를 사용할 계획이라면 [dms-fleetadvisor-iam-slr-s3.zip](#)을 선택합니다. DMS Fleet Advisor용 SLR을 이미 생성한 경우 [dms-fleetadvisor-iam-s3.zip](#)을 선택합니다.
 - AWS Application Discovery Service(ADS) Agentless Collector를 사용할 계획이고 SLR을 생성하지 않은 경우 [dms-fleetadvisor-ads-iam-slr-s3.zip](#)을 선택합니다. ADS와 함께 DMS Fleet Advisor용 SLR을 이미 생성한 경우 [dms-fleetadvisor-ads-iam-s3.zip](#)을 선택합니다.
2. 파일을 컴퓨터에 저장합니다.

2단계: CloudFormation을 사용하여 Amazon S3 및 IAM 구성

IAM에 CloudFormation 템플릿을 사용하는 경우, 이전에 나열한 Amazon S3 및 IAM 리소스가 생성됩니다.

CloudFormation을 사용하여 Amazon S3 및 IAM을 구성하려면

1. <https://console.aws.amazon.com/cloudformation> CloudFormation 콘솔을 엽니다.
2. 드롭다운 목록에서 스택 생성 및 새 리소스 사용을 선택하여 스택 생성 마법사를 시작합니다.
3. 스택 생성 페이지에서 다음을 수행합니다.
 - a. 템플릿 준비에서 템플릿 준비가 완료되었습니다를 선택합니다.
 - b. 템플릿 소스로 템플릿 파일 업로드를 선택합니다.
 - c. 파일 선택에서 `dms-fleetadvisor-iam-sl-r-S3.json`, `dms-fleetadvisor-iam-S3.json`, `dms-fleetadvisor-ads-iam-sl-r-s3.zip` 또는 `dms-fleetadvisor-ads-iam-s3.zip`을 찾아 선택합니다.
 - d. 다음을 선택합니다.
4. 스택 세부 정보 지정 페이지에서 다음 작업을 수행합니다.
 - a. 스택 이름에 **`dms-fleetadvisor-iam-sl-r-s3`**, **`dms-fleetadvisor-iam-s3`**, **`dms-fleetadvisor-ads-iam-sl-r-s3`** 또는 **`dms-fleetadvisor-ads-iam-s3`**를 입력합니다.
 - b. 다음을 선택합니다.
5. 스택 옵션 구성 페이지에서 다음을 선택합니다.
6. `dms-fleetadvisor-iam-sl-r-s3` 검토, `dms-fleetadvisor-iam-s3` 검토, `dms-fleetadvisor-ads-iam-sl-r-s3` 검토 또는 `dms-fleetadvisor-ads-iam-s3` 검토 페이지에서 다음을 수행합니다.
 - a. 사용자 지정 이름을 사용하여 CloudFormation 이 IAM 리소스를 생성할 수 있음에 동의합니다 확인란을 선택합니다.
 - b. 제출을 선택합니다.

CloudFormation이 DMS Fleet Advisor에 필요한 S3 버킷과 IAM 역할 및 사용자를 생성합니다. 왼쪽 패널에서 `dms-fleetadvisor-iam-sl-r-s3`, `dms-fleetadvisor-iam-s3`, `dms-fleetadvisor-ads-iam-sl-r-s3` 또는 `dms-fleetadvisor-ads-iam-s3`에 `CREATE_COMPLETE`가 표시되면 다음 단계로 진행합니다.

7. 왼쪽 패널에서 `dms-fleetadvisor-iam-sl-r-s3`, `dms-fleetadvisor-iam-s3`, `dms-fleetadvisor-ads-iam-sl-r-s3` 또는 `dms-fleetadvisor-ads-iam-s3`를 선택합니다. 오른쪽 패널에서 다음을 수행합니다.

- a. 스택 정보를 선택합니다. 스택에는 다음과 같은 형식의 ID가 있습니다.
`arn:aws:cloudformation:region:account-no:stack/dms-fleetadvisor-iam-slr-s3/identifier`, `arn:aws:cloudformation:region:account-no:stack/dms-fleetadvisor-iam-s3/identifier`, `arn:aws:cloudformation:region:account-no:stack/dms-fleetadvisor-ads-iam-slr-s3/identifier`, or `arn:aws:cloudformation:region:account-no:stack/dms-fleetadvisor-ads-iam-s3/identifier`.
- b. 리소스를 선택합니다. 다음과 같은 모양이어야 합니다.
 - `dms-fleetadvisor-data-accountId-region`이라는 Amazon S3 버킷
 - `FleetAdvisorS3Role-region`이라는 서비스 역할
 - `FleetAdvisorCollectorUser-region`이라는 IAM 사용자 이름
 - `AWSServiceRoleForDMSFleetAdvisor`라는 IAM SLR(`dms-fleet-advisor-iam-slr-s3.zip` 또는 `dms-fleet-advisor-ads-iam-slr-s3.zip`을 다운로드한 경우).
 - `FleetAdvisorS3Role-region-Policy`라는 액세스 정책
 - `FleetAdvisorCollectorUser-region-Policy`라는 액세스 정책

에서 Amazon S3 및 IAM 리소스 구성 AWS Management Console

Amazon S3 버킷 생성

인벤토리 메타데이터를 저장할 수 있는 Amazon S3 버킷을 생성합니다. DMS Fleet Advisor를 사용하기 전에 이 S3 버킷을 미리 구성하는 것이 좋습니다. AWS DMS 는 DMS Fleet Advisor 인벤토리 메타데이터를 이 S3 버킷에 저장합니다.

S3 버킷 만들기에 관한 자세한 내용은 Amazon S3 사용 설명서의 [첫 번째 S3 버킷 생성](#)을 참조하세요.

Note

DMS Fleet Advisor는 SSE-S3 암호화 버킷만 지원합니다.

Amazon S3 버킷을 생성하여 로컬 데이터 환경 정보를 저장하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/s3/> Amazon S3 콘솔을 엽니다.
2. 버킷 만들기를 선택합니다.

3. 버킷 만들기 페이지에서 버킷의 로그인 이름이 포함된 세계적으로 고유한 이름(예: fa-bucket-**yoursignin**)을 입력합니다.
4. DMS Fleet Advisor를 사용할 AWS 리전 을 선택합니다.
5. 나머지 설정을 유지하고 버킷 만들기를 선택합니다.

IAM 리소스 생성

이 섹션에서는 데이터 수집기, IAM 사용자, DMS Fleet Advisor를 위한 IAM 리소스를 생성합니다.

주제

- [데이터 수집기를 위한 IAM 리소스 생성](#)
- [DMS Fleet Advisor 서비스 연결 역할 생성](#)

데이터 수집기를 위한 IAM 리소스 생성

데이터 수집기가 올바르게 작동하는지 확인하고 수집된 메타데이터를 Amazon S3 버킷으로 업로드하려면 다음 정책을 생성하세요. 그런 다음, 다음과 같은 최소한의 권한을 가진 IAM 사용자를 생성합니다. DMS 수집기에 관한 자세한 내용은 [에서 데이터 수집기를 사용하여 마이그레이션할 데이터베이스 검색 AWS DMS](#) 섹션을 참조하세요.

DMS Fleet Advisor 및 데이터 수집기가 Amazon S3에 액세스할 수 있도록 IAM 정책을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON을 편집기에 붙여 넣으면서 예제 코드를 바꿉니다. **fa_bucket**을 이전 섹션에서 생성한 Amazon S3 버킷의 이름으로 바꿉니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```

    "Effect": "Allow",
    "Action": [
        "s3:GetObject*",
        "s3:GetBucket*",
        "s3:List*",
        "s3:DeleteObject*",
        "s3:PutObject*"
    ],
    "Resource": [
        "arn:aws:s3:::fa_bucket",
        "arn:aws:s3:::fa_bucket/*"
    ]
}
]
}

```

6. 다음: 태그와 다음: 검토를 선택합니다.
7. 이름*의 경우 **FleetAdvisorS3Policy**를 입력한 후 정책 생성을 선택합니다.

DMS 데이터 수집기가 DMS Fleet Advisor에 액세스할 수 있도록 IAM 정책을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON 코드를 편집기에 붙여 넣으면서 예제 코드를 바꿉니다.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dms:DescribeFleetAdvisorCollectors",
        "dms:ModifyFleetAdvisorCollectorStatuses",
        "dms:UploadFileMetadataList"
      ],

```

```

    "Resource": "*"
  }
}
}

```

6. 다음: 태그와 다음: 검토를 선택합니다.
7. 이름*의 경우 **DMSCollectorPolicy**를 입력한 후 정책 생성을 선택합니다.

DMS 데이터 수집기를 사용할 수 있는 최소한의 권한을 가진 IAM 사용자를 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 사용자를 선택합니다.
3. 사용자 추가를 선택합니다.
4. 사용자 추가 페이지의 사용자 이름*에 **FleetAdvisorCollectorUser**를 입력합니다. 액세스 키 - 선택 액세스 유형에 대한 프로그래밍 방식 AWS 액세스를 선택합니다. 다음: 권한을 선택합니다.
5. 권한 설정 섹션에서 기존 정책 직접 연결을 선택합니다.
6. 검색 컨트롤을 사용하여 이전에 생성한 DMSCollectorPolicy 및 FleetAdvisorS3Policy 정책을 찾아 선택합니다. 다음: 태그를 선택합니다.
7. 태그 페이지에서 다음: 검토를 선택합니다.
8. 검토 페이지에서 사용자 생성을 선택합니다. 다음 페이지에서 .csv 다운로드를 선택하여 새 사용자 자격 증명을 저장합니다. 필요한 최소 액세스 권한을 얻으려면 DMS Fleet Advisor와 함께 이 자격 증명을 사용하세요.

DMS Fleet Advisor 및 데이터 수집기가 Amazon S3에 액세스할 수 있도록 IAM 역할을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Roles를 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지의 신뢰할 수 있는 엔터티 유형 아래에서 AWS 서비스를 선택합니다. 다른 AWS 서비스의 사용 사례에서 DMS를 선택합니다.
5. DMS 확인란을 선택하고 다음을 선택합니다.
6. 권한 추가 페이지에서 FleetAdvisorS3Policy를 선택합니다. 다음을 선택합니다.

7. 이름 지정, 검토 및 생성 페이지에서 역할 이름에 **FleetAdvisorS3Role**을 입력하고 역할 생성을 선택합니다.
8. 역할 페이지에서 역할 이름에 **FleetAdvisorS3Role**을 입력합니다. FleetAdvisorS3Role을 선택합니다.
9. FleetAdvisorS3Role 페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.
10. 신뢰 정책 편집 페이지에서 기존 텍스트를 대체하여 다음 JSON을 편집기에 붙여넣습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "dms.amazonaws.com",
          "dms-fleet-advisor.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

위의 정책은가 Amazon S3 버킷에서 수집된 데이터를 가져오는 데 AWS DMS 사용하는 서비스에 sts:AssumeRole 권한을 부여합니다.

11. 정책 업데이트를 선택합니다.

DMS Fleet Advisor 서비스 연결 역할 생성

DMS Fleet Advisor는 서비스 연결 역할을 사용하여 AWS 계정에서 Amazon CloudWatch 지표를 관리합니다. DMS Fleet Advisor는 이 서비스 연결 역할을 사용하여 수집된 데이터베이스 성능 지표를 CloudWatch에 대신 게시합니다.

DMS Fleet Advisor에 대한 서비스 연결 역할을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다. 그런 다음 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형에서 AWS 서비스를 선택합니다.
4. 다른 AWS 서비스의 사용 사례에서 DMS – Fleet Advisor를 선택합니다.
5. DMS – Fleet Advisor 확인란을 선택하고 다음을 선택합니다.
6. 권한 추가 페이지에서 다음을 선택합니다.
7. 이름 지정, 검토 및 생성 페이지에서 역할 생성을 선택합니다.

또는 AWS API 또는 AWS CLI에서이 서비스 연결 역할을 생성할 수 있습니다. 자세한 내용은 [AWS DMS Fleet Advisor에 대한 서비스 연결 역할 생성](#) 단원을 참조하십시오.

DMS Fleet Advisor의 서비스 연결 역할을 생성한 후에는 대상 권장 사항에서 소스 데이터베이스의 성능 지표를 확인할 수 있습니다. 또한 CloudWatch 계정에서 이러한 지표를 볼 수 있습니다. 자세한 내용은 [대상 권장 사항](#) 섹션을 참조하세요.

DMS Fleet Advisor 서비스 연결 역할에 필요한 IAM 정책을 생성하려면

서비스 연결 역할을 생성하는 데 필요한 최소 권한은

DMSFleetAdvisorCreateServiceLinkedRolePolicy 정책에 지정되어 있습니다. 서비스 연결 역할을 생성할 수 없는 경우 계정에 이 IAM 정책을 생성합니다.

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON 코드를 편집기에 붙여 넣으면서 예제 코드를 바꿉니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
{
  "Effect": "Allow",
  "Action": "iam:CreateServiceLinkedRole",
  "Resource": "arn:aws:iam::*:role/aws-service-role/dms-fleet-
advisor.amazonaws.com/AWSServiceRoleForDMSFleetAdvisor*",
  "Condition": {"StringLike": {"iam:AWSServiceName": "dms-fleet-
advisor.amazonaws.com"}}
},
{
  "Effect": "Allow",
  "Action": [
    "iam:AttachRolePolicy",
    "iam:PutRolePolicy"
  ],
  "Resource": "arn:aws:iam::*:role/aws-service-role/dms-fleet-
advisor.amazonaws.com/AWSServiceRoleForDMSFleetAdvisor*"
}
]
```

6. 다음: 태그와 다음: 검토를 선택합니다.
7. 이름*의 경우 **DMSFleetAdvisorCreateServiceLinkedRolePolicy**를 입력한 후 정책 생성을 선택합니다.

이제 이 정책을 사용하여 DMS Fleet Advisor에 대한 서비스 연결 역할을 생성할 수 있습니다.

AWS DMS Fleet Advisor용 데이터베이스 사용자 생성

Important

지원 종료 공지: 2026년 5월 20일에 AWS는 AWS Database Migration Service Fleet Advisor에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

이 섹션에서는 DMS 데이터 수집기에 필요한 최소 권한을 가진 소스 데이터베이스 사용자를 생성하는 방법을 설명합니다.

이 섹션은 다음 주제를 포함합니다:

- [AWS DMS Fleet Advisor와 함께 데이터베이스 사용자 사용](#)
- [MySQL을 사용하여 데이터베이스 사용자 생성](#)
- [Oracle을 사용하여 데이터베이스 사용자 생성](#)
- [PostgreSQL을 사용하여 데이터베이스 사용자 생성](#)
- [Microsoft SQL Server를 사용하여 데이터베이스 사용자 생성](#)
- [데이터베이스 사용자 삭제](#)

AWS DMS Fleet Advisor와 함께 데이터베이스 사용자 사용

DMS 데이터 수집기에서는 root 이외의 데이터베이스 사용자를 사용할 수 있습니다. 인벤토리에 데이터베이스를 추가한 후 데이터 수집기를 실행하기 전에 사용자 이름과 암호를 지정하세요. 인벤토리에 데이터베이스를 추가하는 방법에 관한 자세한 내용은 [에서 모니터링되는 객체 관리 AWS DMS](#) 섹션을 참조하세요.

DMS 데이터 수집기 사용을 마친 후에는 생성한 데이터베이스 사용자를 삭제할 수 있습니다. 자세한 내용은 [데이터베이스 사용자 삭제](#) 섹션을 참조하세요.

Important

다음 예제에서는 `{your_user_name}`을 데이터베이스용으로 만든 데이터베이스 사용자 이름으로 바꿉니다. 그런 다음, `{your_password}`를 안전한 비밀번호로 바꾸세요.

MySQL을 사용하여 데이터베이스 사용자 생성

MySQL 소스 데이터베이스에서 데이터베이스 사용자를 생성하려면 다음 스크립트를 사용합니다. MySQL 데이터베이스 버전에 따라 한 가지 버전의 GRANT 문을 유지해야 합니다.

```
CREATE USER {your_user_name} identified BY '{your_password}';

GRANT PROCESS ON *.* TO {your_user_name};
GRANT REFERENCES ON *.* TO {your_user_name};
GRANT TRIGGER ON *.* TO {your_user_name};
GRANT EXECUTE ON *.* TO {your_user_name};

# For MySQL versions lower than 8.0, use the following statement.
GRANT SELECT, CREATE TEMPORARY TABLES ON `temp`.* TO {your_user_name};
```

```
# For MySQL versions 8.0 and higher, use the following statement.
GRANT SELECT, CREATE TEMPORARY TABLES ON `mysql`.* TO {your_user_name};

GRANT SELECT ON performance_schema.* TO {your_user_name};

SELECT
IF(round(Value1 + Value2 / 100 + Value3 / 10000, 4) > 5.0129, 'GRANT EVENT ON *.*
  TO {your_user_name};', 'SELECT ''Events are not applicable'';') sql_statement
INTO @stringStatement
FROM (
  SELECT
    substring_index(ver, '.', 1) value1,
    substring_index(substring_index(ver, '.', 2), '.', - 1) value2,
    substring_index(ver, '.', - 1) value3
  FROM (
    SELECT
      IF((@@version regexp '^[0-9\.]+' ) != 0, @@innodb_version, @@version) AS ver
    FROM dual
  ) vercase
) v;

PREPARE sqlStatement FROM @stringStatement;
SET @stringStatement := NULL;
EXECUTE sqlStatement;
DEALLOCATE PREPARE sqlStatement;
```

Oracle을 사용하여 데이터베이스 사용자 생성

Oracle 소스 데이터베이스에서 데이터베이스 사용자를 생성하려면 다음 스크립트를 사용합니다.

이 SQL 스크립트를 실행하려면 SYSDBA 권한을 사용하여 Oracle 데이터베이스에 연결하세요. 이 SQL 스크립트를 실행한 후 이 스크립트로 생성한 사용자의 자격 증명을 사용하여 데이터베이스에 연결합니다. 또한 이 사용자의 자격 증명을 사용하여 DMS 데이터 수집기를 실행할 수 있습니다.

다음 스크립트는 Oracle 멀티테넌트 컨테이너 데이터베이스(CDB)의 사용자 이름에 C## 접두사를 추가합니다.

```
CREATE USER {your_user_name} IDENTIFIED BY "{your_password}";
GRANT CREATE SESSION TO {your_user_name};
GRANT SELECT ANY DICTIONARY TO {your_user_name};
GRANT SELECT ON DBA_WM_SYS_PRIVS TO {your_user_name};
BEGIN
  DBMS_NETWORK_ACL_ADMIN.CREATE_ACL(
```



```

acl => UPPER('{your_user_name}') || '_Connect_Access.xml',
description => 'Connect Network',
principal => UPPER('{your_user_name}'),
is_grant => TRUE,
privilege => 'resolve',
start_date => NULL,
end_date => NULL);

DBMS_NETWORK_ACL_ADMIN.ASSIGN_ACL(
  acl => UPPER('{your_user_name}') || '_Connect_Access.xml',
  host => '*',
  lower_port => NULL,
  upper_port => NULL);
END;
```

PostgreSQL을 사용하여 데이터베이스 사용자 생성

PostgreSQL 소스 데이터베이스에서 데이터베이스 사용자를 생성하려면 다음 스크립트를 사용합니다.

```

CREATE USER "{your_user_name}" WITH LOGIN PASSWORD '{your_password}';
GRANT pg_read_all_settings TO "{your_user_name}";

-- For PostgreSQL versions 10 and higher, add the following statement.
GRANT EXECUTE ON FUNCTION pg_ls_waldir() TO "{your_user_name}";
```

Microsoft SQL Server를 사용하여 데이터베이스 사용자 생성

Microsoft SQL Server 소스 데이터베이스에서 데이터베이스 사용자를 생성하려면 다음 스크립트를 사용합니다.

```

USE master
GO

IF NOT EXISTS (SELECT * FROM sys.sql_logins WHERE name = N'{your_user_name}')

CREATE LOGIN [{your_user_name}] WITH PASSWORD=N'{your_password}',
  DEFAULT_DATABASE=[master], DEFAULT_LANGUAGE=[us_english], CHECK_EXPIRATION=OFF,
  CHECK_POLICY=OFF

GO

GRANT VIEW SERVER STATE TO [{your_user_name}]
```

```
GRANT VIEW ANY DEFINITION TO [{your_user_name}]

GRANT VIEW ANY DATABASE TO [{your_user_name}]

IF LEFT(CONVERT(SYSNAME,SERVERPROPERTY('ProductVersion')), CHARINDEX('.',
    CONVERT(SYSNAME,SERVERPROPERTY('ProductVersion')), 0)-1) >= 12
EXECUTE('GRANT CONNECT ANY DATABASE TO [{your_user_name}]')

DECLARE @dbname VARCHAR(100)
DECLARE @statement NVARCHAR(max)

DECLARE db_cursor CURSOR
LOCAL FAST_FORWARD
FOR
SELECT
name
FROM MASTER.sys.databases
WHERE state = 0
AND is_read_only = 0
    OPEN db_cursor
FETCH NEXT FROM db_cursor INTO @dbname
WHILE @@FETCH_STATUS = 0
BEGIN

SELECT @statement = 'USE ' + quotename(@dbname) + ';' + '
IF NOT EXISTS (SELECT * FROM sys.syslogins WHERE name = ''{your_user_name}'') OR NOT
EXISTS (SELECT * FROM sys.sysusers WHERE name = ''{your_user_name}'')
    CREATE USER [{your_user_name}] FOR LOGIN [{your_user_name}];

EXECUTE sp_addrolemember N'db_datareader'', [{your_user_name}]'

BEGIN TRY
EXECUTE sp_executesql @statement
END TRY
BEGIN CATCH
DECLARE @err NVARCHAR(255)

SET @err = error_message()

PRINT @dbname
PRINT @err
END CATCH
```

```

FETCH NEXT FROM db_cursor INTO @dbname
END
CLOSE db_cursor
DEALLOCATE db_cursor

USE msdb
GO

GRANT EXECUTE ON dbo.agent_datetime TO [{your_user_name}]

```

데이터베이스 사용자 삭제

모든 데이터 수집 작업을 완료한 후에는 DMS 데이터 수집기에 대해 생성한 데이터베이스 사용자를 삭제할 수 있습니다. 다음 스크립트를 사용하여 데이터베이스에서 최소 권한을 가진 사용자를 삭제할 수 있습니다.

MySQL 데이터베이스에서 사용자를 삭제하려면 다음 스크립트를 실행합니다.

```
DROP USER IF EXISTS "{your_user_name}";
```

Oracle 데이터베이스에서 사용자를 삭제하려면 다음 스크립트를 실행합니다.

```

DECLARE
-- Input parameters, please set correct value
cnst$user_name CONSTANT VARCHAR2(255) DEFAULT '{your_user_name}';

-- System variables, please, don't change
var$is_exists INTEGER DEFAULT 0;
BEGIN
SELECT COUNT(hal.acl) INTO var$is_exists
FROM dba_host_acls hal
WHERE hal.acl LIKE '%' || UPPER(cnst$user_name) || '_Connect_Access.xml';
IF var$is_exists > 0 THEN
DBMS_NETWORK_ACL_ADMIN.DROP_ACL(
    acl => UPPER(cnst$user_name) || '_Connect_Access.xml');
END IF;
SELECT COUNT(usr.username) INTO var$is_exists
FROM all_users usr
WHERE usr.username = UPPER(cnst$user_name);
IF var$is_exists > 0 THEN
EXECUTE IMMEDIATE 'DROP USER ' || cnst$user_name || ' CASCADE';
END IF;

```

```
END;
```

PostgreSQL 데이터베이스에서 사용자를 삭제하려면 다음 스크립트를 실행합니다.

```
DROP USER IF EXISTS "{your_user_name}";
```

SQL Server 데이터베이스에서 사용자를 삭제하려면 다음 스크립트를 실행합니다.

```
USE msdb
GO

REVOKE EXECUTE ON dbo.agent_datetime TO [{your_user_name}]

USE master
GO

DECLARE @dbname VARCHAR(100)
DECLARE @statement NVARCHAR(max)

DECLARE db_cursor CURSOR
LOCAL FAST_FORWARD
FOR
SELECT
name
FROM MASTER.sys.databases
WHERE state = 0
AND is_read_only = 0
OPEN db_cursor
FETCH NEXT FROM db_cursor INTO @dbname
WHILE @@FETCH_STATUS = 0
BEGIN

SELECT @statement = 'USE ' + quotename(@dbname) + ';' + '
EXECUTE sp_droprolemember N''db_datareader'', [{your_user_name}]

IF EXISTS (SELECT * FROM sys.syslogins WHERE name = ''{your_user_name}'')
OR EXISTS (SELECT * FROM sys.sysusers WHERE name = ''{your_user_name}'')
DROP USER [{your_user_name}];'

BEGIN TRY
EXECUTE sp_executesql @statement
END TRY
BEGIN CATCH
```

```
DECLARE @err NVARCHAR(255)

SET @err = error_message()

PRINT @dbname
PRINT @err
END CATCH

FETCH NEXT FROM db_cursor INTO @dbname
END
CLOSE db_cursor
DEALLOCATE db_cursor

GO

IF EXISTS (SELECT * FROM sys.sql_logins WHERE name = N'{your_user_name}')
DROP LOGIN [{your_user_name}] -- Use for SQL login

GO
```

에서 데이터 수집기를 사용하여 마이그레이션할 데이터베이스 검색 AWS DMS

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

소스 데이터 인프라를 검색하려면 [AWS Application Discovery Service Agentless Collector](#) 또는 AWS DMS 데이터 수집기를 사용할 수 있습니다. ADS Agentless Collector는 에이전트가 필요 없는 방법을 통해 서버 프로파일 정보(예: OS, CPU 수, RAM 용량), 데이터베이스 메타데이터, 사용률 지표 등 온프레미스 환경에 대한 정보를 수집하는 온프레미스 애플리케이션입니다. OVA(Open Virtualization Archive) 파일을 사용하여 Agentless Collector를 VMware vCenter Server 환경의 가상 머신(VM)으로 설치합니다. AWS DMS Andata Collector는 로컬 환경에 설치하는 Windows 애플리케이션입니다. 이 애플리케이션은 데이터 환경에 연결하고 온프레미스 데이터베이스 및 분석 서버에서 메타데이터 및 성능 지표를 수집합니다. ADS Agentless Collector 또는 DMS 데이터 수집기를 통해 데이터베이스 메

타데이터 및 성능 지표가 수집되면 DMS Fleet Advisor가 AWS 클라우드로 마이그레이션할 수 있는 서버, 데이터베이스 및 스키마의 인벤토리를 작성합니다.

DMS 데이터 수집기는 .NET 라이브러리, 커넥터 및 데이터 공급자를 사용하여 데이터베이스 검색 및 데이터 수집을 위해 소스 데이터베이스에 연결하는 Windows 애플리케이션입니다.

DMS 데이터 수집기는 Windows에서 실행됩니다. 하지만 DMS 데이터 수집기는 지원되는 데이터베이스 공급업체가 실행하는 OS 서버에 관계없이 지원되는 모든 데이터베이스 공급업체로부터 데이터를 수집할 수 있습니다.

DMS 데이터 수집기는 TLS 암호화를 적용한 보호된 RTPS 프로토콜을 사용하여 DMS Fleet Advisor와의 보안 연결을 설정합니다. 따라서 기본적으로 전송 중에 데이터가 암호화됩니다.

AWS DMS 에는에 대해 생성할 수 있는 최대 데이터 수집기 수가 있습니다 AWS 계정. AWS DMS 서비스 할당량에 대한 자세한 내용은 다음 섹션을 참조하세요 [에 대한 할당량AWS Database Migration Service](#).

주제

- [DMS 데이터 수집기에 대한 권한](#)
- [AWS DMS Fleet Advisor용 데이터 수집기 생성](#)
- [에서 데이터 수집기 설치 및 구성 AWS DMS](#)
- [에서 모니터링할 OS 및 데이터베이스 서버 검색 AWS DMS](#)
- [에서 모니터링되는 객체 관리 AWS DMS](#)
- [AWS DMS Fleet Advisor에서 SSL 사용](#)
- [AWS DMS Fleet Advisor에 대한 데이터 수집](#)
- [DMS 데이터 수집기 문제 해결](#)

DMS 데이터 수집기에 대한 권한

DMS 데이터 수집기를 위해 생성한 데이터베이스 사용자에게는 읽기 권한이 있어야 합니다. 그러나 경우에 따라서는 데이터베이스 사용자에게 EXECUTE 권한이 필요합니다. 자세한 내용은 [AWS DMS Fleet Advisor용 데이터베이스 사용자 생성](#) 단원을 참조하십시오.

검색 스크립트를 실행하려면 DMS 데이터 수집기에 추가 권한이 필요합니다.

- OS 검색의 경우, DMS 데이터 수집기에는 도메인 서버가 LDAP 프로토콜을 사용하여 요청을 실행하기 위한 자격 증명이 필요합니다.

- Linux에서 데이터베이스를 검색하려면 DMS 데이터 수집기에 sudo SSH 권한 부여가 포함된 자격 증명이 필요합니다. 또한 원격 SSH 스크립트를 실행할 수 있도록 Linux 서버를 구성해야 합니다.
- Windows에서 데이터베이스를 검색하려면 DMS 데이터 수집기가 WMI(Windows Management Instrumentation) 및 WQL(WMI Query Language) 쿼리를 실행하고 레지스트리를 읽을 수 있는 권한이 있는 자격 증명이 필요합니다. 또한 원격 WMI, WQL 및 PowerShell 스크립트를 실행할 수 있도록 Windows 서버를 구성해야 합니다.

AWS DMS Fleet Advisor용 데이터 수집기 생성

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS 데이터 수집기를 생성하고 다운로드하는 방법을 알아봅니다.

데이터 수집기를 생성하기 전에 IAM 콘솔을 사용해 DMS Fleet Advisor에 대한 서비스 연결 역할을 생성해야 합니다. 이 역할은 위탁자가 Amazon CloudWatch에 지표 데이터 포인트를 게시할 수 있도록 허용합니다. DMS Fleet Advisor는 이 역할을 사용하여 데이터베이스 지표와 함께 차트를 표시합니다. 자세한 내용은 [AWS DMS Fleet Advisor에 대한 서비스 연결 역할 생성](#) 단원을 참조하십시오.

데이터 수집기를 생성하고 다운로드하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

DMS Fleet Advisor를 사용할 리전을 선택합니다.

2. 탐색 창에서 검색 아래에 있는 데이터 수집기를 선택합니다. 데이터 수집기 페이지가 열립니다.
3. 데이터 수집기 생성을 선택합니다. 데이터 수집기 생성 페이지가 열립니다.

DMS > Discover: Data collectors > Create data collector

Create data collector [Info](#)

Create a data collector to identify servers, databases, and schemas on a network. After the data collector is created, you're prompted to register it by downloading and installing a local collector.

 You can create a maximum of 10 data collectors. [Learn more](#) 

General configuration

Name

Can have only Unicode letters, digits, white space, or one of the symbols in parentheses: (./=+-@{}). Maximum of 60 characters.

Description - *optional*

Provide a description of the data collector purpose, environment, or network to help you identify it in the future.

Can have only Unicode letters, digits, white space, or one of the symbols in parentheses: (./=+-@{}). Maximum of 255 characters.

Connectivity [Info](#)

Amazon S3 bucket

Choose or create an Amazon S3 bucket to store collected metadata. Ensure this bucket is the currently selected region.

[View](#) [Browse S3](#)

To create a bucket role, go to [S3](#) 

IAM role

Choose or create an IAM role that grants AWS DMS permissions to access the specified S3 bucket.

To create an IAM role, go to [IAM console](#) 

[Cancel](#)[Create data collector](#)

4. 일반 구성 섹션의 이름에 데이터 수집기의 이름을 입력합니다.

5. 연결 섹션에서 Browse S3을 선택합니다. 표시되는 목록에서 사전 구성한 Amazon S3 버킷을 선택합니다.

AWS DMS 는 DMS Fleet Advisor 인벤토리 메타데이터를이 S3 버킷에 저장합니다. Amazon S3 버킷이 AWS DMS Fleet Advisor가 현재 실행 중인 AWS 리전 동일한에 있는지 확인합니다.

 Note

DMS Fleet Advisor는 SSE-S3 암호화 버킷만 지원합니다.

6. IAM 역할 목록에서 화면에 표시되는 해당 목록 중 사전 구성한 IAM 역할을 선택합니다. 이 역할은 지정된 Amazon S3 버킷에 액세스할 수 있는 AWS DMS 권한을 부여합니다.
7. 데이터 수집기 생성을 선택합니다. 데이터 수집기 페이지가 열리고 생성된 데이터 수집기가 목록에 나타납니다.

첫 번째 데이터 수집기를 생성할 때는 데이터 형식을 지정하고 DMS Fleet Advisor와 함께 사용할 속성을 저장하는 Amazon S3 버킷의 환경을 AWS DMS 구성합니다.

8. 정보 배너에서 로컬 수집기 다운로드를 선택하여 새로 생성된 데이터 수집기를 다운로드합니다. 다운로드가 진행 중임을 알리는 메시지가 나타납니다. 다운로드가 완료된 후 `AWS_DMS_Collector_Installer_`*version_number*.msi 파일에 액세스할 수 있습니다.

이제 클라이언트에 DMS 데이터 수집기를 설치할 수 있습니다. 자세한 내용은 [에서 데이터 수집기 설치 및 구성 AWS DMS](#) 단원을 참조하십시오.

에서 데이터 수집기 설치 및 구성 AWS DMS

 Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS 데이터 수집기를 설치하는 방법, 데이터 전달 자격 증명을 지정하는 방법, 프로젝트에 LDAP 서버를 추가하는 방법을 알아봅니다.

다음 표에는 DMS 데이터 수집기를 설치하기 위한 하드웨어 및 소프트웨어 요구 사항이 설명되어 있습니다.

최소	권장
프로세서: CPU 벤치마크 점수가 8,000점 이상인 코어 2개	프로세서: CPU 벤치마크 점수가 11,000점 이상인 코어 4개
RAM: 8GB	RAM: 16GB
하드 드라이브 크기: 256GB	하드 드라이브 크기: 512GB
운영 체제: Microsoft Windows Server 2012 또는 이후 버전	운영 체제: Windows Server 2016 또는 이후 버전

네트워크의 클라이언트에 데이터 수집기를 설치하려면

1. .MSI 설치 관리자를 실행합니다. AWS DMS Fleet Advisor 수집기 설치 마법사 페이지가 나타납니다.
2. 다음을 선택합니다. 최종 사용자 라이선스 계약이 표시됩니다.
3. 최종 사용자 라이선스 계약을 읽고 동의하세요.
4. 다음을 선택합니다. 대상 폴더 페이지가 나타납니다.
5. 다음을 선택하여 기본 디렉터리에 데이터 수집기를 설치합니다.

또는 변경을 선택하여 다른 설치 디렉터리를 입력합니다. 그리고 다음을 선택합니다.

6. 바탕 화면 바로 가기 페이지에서 확인란을 선택하여 바탕 화면에 아이콘을 설치합니다.
7. 설치를 선택합니다. 데이터 수집기는 선택한 디렉터리에 설치됩니다.
8. 완료된 DMS 수집기 설정 마법사 페이지에서 AWS DMS 수집기 시작을 선택한 다음 완료를 선택합니다.

DMS 데이터 수집기는 .NET 라이브러리, 커넥터 및 데이터 공급자를 사용하여 소스 데이터베이스에 연결합니다. DMS 데이터 수집기 설치 프로그램은 지원되는 모든 데이터베이스에 필요한 이 소프트웨어를 서버에 자동으로 설치합니다.

데이터 수집기를 설치한 후 **http://localhost:11000/**을 주소로 입력하여 브라우저에서 실행할 수 있습니다. Microsoft Windows 시작 메뉴의 프로그램 목록에서 AWS DMS 수집기를 선택할 수도 있

습니다. DMS 데이터 수집기를 처음 실행하면 자격 증명을 구성하라는 메시지가 표시됩니다. 데이터 수집기에 로그인하기 위한 사용자 이름과 암호를 생성합니다.

DMS 데이터 수집기 홈페이지에서 다음 상태 조건을 포함하여 메타데이터 수집을 준비하고 실행하기 위한 정보를 찾을 수 있습니다.

- 데이터 수집의 상태.
- 데이터 수집기가 데이터를 전달할 수 AWS DMS 있도록 Amazon S3 버킷 및에 대한 액세스 가능성 AWS DMS.
- 설치된 데이터베이스 드라이버에 대한 연결.
- 초기 검색을 수행하기 위한 LDAP 서버의 자격 증명.

The screenshot shows the AWS DMS Collector interface. At the top, there's a navigation bar with the AWS logo and 'DMS Collector' text, and a 'Sign out' link. On the left is a sidebar with icons for Home, Data collection, Data forwarding, and Software check. The main content area is divided into three panels: 'Data collection' showing 'Status: Running' and 'Health: 100%', 'Data forwarding' showing 'Name: new-data-collector', 'Access to Amazon S3: Yes', 'Access to AWS DMS: Yes', and 'Last updated: 31-01-2023 11:43:30' with a 'Configure forwarding' button, and 'Software check (4/4)' showing four connectors (Microsoft SQL Server, MySQL, Oracle, PostgreSQL) all with 'Passed' status. Below these panels is the 'LDAP servers configuration' section with a '+ Server' button and a table listing LDAP servers.

LDAP server host name ↓	User name	Connection status
☐ dc01.dbm.local	shareduser	Passed

DMS 데이터 수집기는 LDAP 디렉터리를 사용하여 네트워크의 시스템 및 데이터베이스 서버에 관한 정보를 수집합니다. LDAP(Lightweight Directory Access Protocol)는 개방형 표준 애플리케이션 프로토콜입니다. LDAP는 IP 네트워크를 통해 분산 디렉터리 정보 서비스에 액세스하고 이 서비스를 유지 관리하는 데 사용됩니다. 시스템 인프라에 관한 정보를 검색하는 데 사용할 수 있는 데이터 수집기에 대한 기존 LDAP 서버를 프로젝트에 추가할 수 있습니다. 이렇게 하려면 +Server 옵션을 선택한 다음, 정규화된 도메인 이름(FQDN)과 도메인 컨트롤러의 자격 증명을 지정하세요. 서버를 추가한 후 연결 확인을 검증합니다. 검색 프로세스를 시작하려면 [에서 모니터링할 OS 및 데이터베이스 서버 검색 AWS DMS](#)을 참조하세요.

데이터 전달을 위한 자격 증명 구성

데이터 수집기를 설치한 후이 애플리케이션이 수집된 데이터를 AWS DMS Fleet Advisor로 전송할 수 있는지 확인합니다.

AWS DMS Fleet Advisor에서 데이터 전달을 위한 자격 증명을 구성하려면

1. DMS 데이터 수집기 홈페이지의 데이터 전달 섹션에서 전달 구성을 선택합니다. 데이터 전달을 위한 자격 증명 구성 대화 상자가 열립니다.
2. DMS Fleet Advisor를 사용할 AWS 리전 를 선택합니다.
3. 이전에 IAM 리소스를 생성할 때 얻은 AWS 액세스 키 ID와 AWS 시크릿 액세스 키를 입력합니다. 자세한 내용은 [IAM 리소스 생성](#) 단원을 참조하십시오.
4. 데이터 수집기 찾아보기를 선택합니다.

지정된 리전에서 데이터 수집기를 아직 생성하지 않았다면 계속 진행하기 전에 데이터 수집기를 생성하세요. 자세한 내용은 [데이터 수집기 생성](#) 단원을 참조하십시오.

5. 데이터 수집기 선택 창의 목록에서 데이터 수집기를 선택하고 선택을 선택합니다.
6. 데이터 전달을 위한 자격 증명 구성 대화 상자에서 저장을 선택합니다.

DMS 수집기 홈페이지의 데이터 전달 카드에서 Amazon S3에 대한 액세스 및 AWS DMS에 대한 액세스의 상태가 예로 설정되어 있는지 확인합니다.

Amazon S3에 대한 액세스 또는 AWS DMS에 대한 액세스의 상태가 아니요로 설정된 경우, Amazon S3 및 DMS Fleet Advisor에 액세스하기 위한 IAM 리소스를 생성했는지 확인하세요. 필요한 모든 권한을 사용하여 이러한 IAM 리소스를 생성한 후 데이터 전달을 다시 구성하세요. 자세한 내용은 [IAM 리소스 생성](#) 단원을 참조하십시오.

에서 모니터링할 OS 및 데이터베이스 서버 검색 AWS DMS

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS 데이터 수집기를 사용하여 네트워크에서 사용 가능한 모든 서버를 찾아 나열할 수 있습니다. 네트워크에서 사용 가능한 모든 데이터베이스 서버를 검색하는 것이 좋지만 필수 사항은 아닙니다. 선택적으로 추가 데이터 수집을 위해 서버 목록을 수동으로 추가하거나 업로드할 수 있습니다. 서버 목록을 수동으로 추가하는 방법에 관한 자세한 내용은 [모니터링되는 객체 관리](#) 섹션을 참조하세요.

운영 체제(OS) 서버에서 데이터베이스를 검색하기 전에 먼저 모든 OS 서버를 검색하는 것이 좋습니다. OS 서버를 검색하려면 원격 PowerShell, Secure Shell(SSH), WMI(Windows Management Instrumentation) 스크립트와 명령을 실행할 수 있는 권한과 Windows 레지스트리에 대한 액세스 권한이 필요합니다. 네트워크에서 데이터베이스 서버를 검색하고 해당 서버에서 메타데이터를 수집하려면 원격 데이터베이스 연결을 위한 읽기 전용 관리자 권한이 필요합니다. 검색을 진행하기 전에 LDAP 서버를 추가했는지 확인하세요. 자세한 내용은 [데이터 전달을 위한 자격 증명 구성](#) 단원을 참조하십시오.

DMS 데이터 수집기를 시작하려면 다음 작업을 완료하세요.

- 네트워크에 있는 모든 OS 서버를 검색하세요.
- 특정 OS 서버를 모니터링할 객체로 추가합니다.
- 모니터링되는 OS 서버의 연결을 확인합니다.
- OS 서버에서 실행되는 Microsoft SQL Server, MySQL, Oracle 및 PostgreSQL 데이터베이스를 검색합니다.
- 데이터 수집을 위해 데이터베이스 서버를 추가합니다.
- 모니터링되는 데이터베이스에 대한 연결을 확인합니다.

네트워크에서 모니터링할 수 있는 OS 서버를 검색하려면

1. DMS 데이터 수집기 탐색 창에서 검색을 선택합니다. 탐색 창을 표시하려면 DMS 데이터 수집기 홈페이지의 왼쪽 위 모서리에 있는 메뉴 아이콘을 선택합니다.

검색 페이지가 열립니다.
2. OS 서버 탭이 선택되어 있는지 확인한 다음, 검색 실행을 선택합니다. 검색 파라미터 대화 상자가 표시됩니다.
3. 네트워크를 스캔하는 데 사용할 LDAP 서버를 입력합니다.
4. 검색 실행을 선택합니다. 이 페이지에는 데이터베이스를 실행하는지 여부에 관계없이 네트워크 내에서 검색된 모든 OS 서버의 목록이 표시됩니다.

모든 OS 서버에서 데이터베이스 검색을 실행하기 전에 모든 OS 서버에 대해 검색을 실행하는 것이 좋습니다. 자격 증명을 사용하면 먼저 호스트 서버를 검색한 다음, 호스트 서버에 있는 데이터베이스를 검색할 수 있습니다. OS 서버에서 데이터베이스 검색을 실행하기 전에 필요하다면 먼

저 OS 서버를 검색합니다. LDAP 서버가 네트워크의 OS 서버를 찾는 데 사용하는 자격 증명은 특정 OS 서버의 데이터베이스를 검색하는 데 필요한 자격 증명과 다를 수 있다는 점에 유의하세요. 따라서 모니터링되는 객체에 OS 서버를 추가하고 자격 증명을 확인하여 필요하다면 수정한 다음, 계속 진행하기 전에 연결 상태를 확인하는 것이 좋습니다.

이제 네트워크에서 검색된 OS 서버의 목록에서 모니터링되는 객체에 추가할 서버를 선택할 수 있습니다.

OS 서버를 모니터링할 객체로 선택하려면

1. 검색 페이지에서 OS 서버 탭을 선택합니다.
2. 검색된 OS 서버 목록이 화면에 표시되면 이 목록에서 모니터링할 각 서버의 옆에 있는 확인란을 선택합니다.
3. 모니터링되는 객체에 추가를 선택합니다.

객체 모니터링 페이지에서 연결을 모니터링하고 확인할 OS 서버의 목록을 볼 수 있습니다.

모니터링할 서버로서 선택한 OS 서버의 연결을 확인하려면

1. DMS 데이터 수집기 탐색 창에서 모니터링되는 객체를 선택합니다.
2. 모니터링되는 객체 페이지에서 OS 서버 탭을 선택합니다. 모니터링할 서버로서 검색된 OS 서버의 목록이 나타납니다.
3. 열 상단의 확인란을 선택하여 목록에 나열된 OS 서버를 모두 선택합니다.
4. 작업을 선택한 다음, 연결 확인을 선택합니다. 각 서버 객체의 경우, 연결 상태 열에서 결과를 확인합니다.
5. 연결 상태가 성공 이외의 상태에 해당하는 서버를 선택합니다. 작업을 선택한 후, 편집을 선택합니다. 서버 편집 대화 상자가 열립니다.
6. 정보가 정확한지 확인하고 필요하다면 편집하세요. 확인 또는 편집이 완료되면 저장을 선택합니다. 자격 증명 재정의 대화 상자가 열립니다.
7. 덮어쓰기를 선택합니다. DMS 데이터 수집기는 각 연결의 상태를 성공으로 확인하고 업데이트합니다.

이제 모니터링하도록 선택한 서버에 있는 데이터베이스를 검색할 수 있습니다.

서버에서 실행되는 데이터베이스를 검색

1. DMS 데이터 수집기 탐색 창에서 검색을 선택합니다.
2. 데이터베이스 서버 탭을 선택하고 검색 실행을 선택합니다. 검색 파라미터 대화 상자가 열립니다.
3. 검색 파라미터 대화 상자의 검색 기준에서 모니터링되는 객체 선택합니다. 서버의 경우 데이터베이스 검색을 실행할 OS 서버를 선택합니다.
4. 검색 실행을 선택합니다. 이 페이지에는 모니터링하기로 선택한 OS 서버에 있는 모든 데이터베이스 목록이 표시됩니다.

데이터베이스 주소, 서버 이름, 데이터베이스 엔진 등의 정보를 보면 모니터링할 데이터베이스를 선택하는 데 도움이 됩니다.

모니터링할 데이터베이스를 선택하려면

1. 검색 페이지에서 데이터베이스 서버 탭을 선택합니다.
2. 검색된 데이터베이스 목록이 화면에 표시되면 이 목록에서 모니터링할 모든 데이터베이스의 옆에 있는 확인란을 선택합니다.
3. 모니터링되는 객체에 추가를 선택합니다.

이제 모니터링하기로 선택한 데이터베이스와의 연결을 확인할 수 있습니다.

모니터링되는 데이터베이스에 대한 연결을 확인하려면

1. DMS 데이터 수집기 탐색 창에서 모니터링되는 객체를 선택합니다.
2. 모니터링되는 객체 페이지에서 데이터베이스 서버 탭을 선택합니다. 모니터링할 서버로서 선택하는 검색된 모든 데이터베이스 서버의 목록이 나타납니다.
3. 열 상단의 확인란을 선택하여 목록에 나열된 데이터베이스 서버를 모두 선택합니다.
4. 작업을 선택한 다음, 연결 확인을 선택합니다. 각 데이터베이스의 경우, 연결 상태 열에서 결과를 확인합니다.
5. 상태가 정의되지 않았거나(비어 있음) 상태이거나 실패 상태가 있는 연결을 선택합니다. 작업을 선택한 후, 편집을 선택합니다. 모니터링되는 객체 편집 대화 상자가 열립니다.
6. 로그인 및 암호 자격 증명을 입력한 다음, 저장을 선택합니다. 자격 증명 변경 대화 상자가 열립니다.
7. 덮어쓰기를 선택합니다. DMS 데이터 수집기는 각 연결의 상태를 성공으로 확인하고 업데이트합니다.

모니터링할 OS 서버 및 데이터베이스를 검색한 후, 모니터링되는 객체를 관리하는 작업을 수행할 수도 있습니다.

에서 모니터링되는 객체 관리 AWS DMS

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

[OS 및 데이터베이스 서버 검색](#)에 설명된 대로 서버 검색 프로세스를 실행할 때 모니터링할 객체를 선택할 수 있습니다. 또한 운영 체제(OS) 서버 및 데이터베이스 서버와 같은 객체를 수동으로 관리할 수 있습니다. 다음 작업을 수행하여 모니터링되는 객체를 관리할 수 있습니다.

- 모니터링할 새 객체 추가
- 기존 객체 제거
- 기존 객체 편집
- 모니터링할 객체 목록 내보내기 및 가져오기
- 객체와의 연결 확인
- 데이터 수집 시작

예를 들어, 모니터링할 객체를 수동으로 추가할 수 있습니다.

모니터링할 객체를 수동으로 추가하려면

1. 모니터링되는 객체 페이지에서 +서버를 선택합니다. 모니터링되는 객체 추가 대화 상자가 열립니다.
2. 서버에 관한 정보를 추가한 다음 저장을 선택합니다.

.csv 파일을 사용하여 모니터링할 객체의 큰 목록을 가져올 수도 있습니다. 다음 .csv 파일 형식을 사용하여 객체 목록을 DMS 데이터 수집기로 가져올 수 있습니다.

```
Hostname - Hostname or IP address of Monitored Object
```



```

Port - TCP port of Monitored Object
Engine: (one of the following)
    • Microsoft SQL Server
    • Microsoft Windows
    • Oracle Database
    • Linux
    • MySQL Server
    • PostgreSQL
Connection type: (one of the following)
    • Login/Password Authentication
    • Windows Authentication
    • Key-Based Authentication
Domain name:(Windows authentication)
    • Use domain name for the account
User name
Password

```

모니터링할 객체 목록이 있는 .csv 파일을 가져오려면

1. 가져오기를 선택합니다. 모니터링되는 객체 가져오기 페이지가 열립니다.
2. 가져오려는 .csv 파일을 찾은 후 다음을 선택합니다.

모든 객체를 볼 수 있으며 필요하다면 메타데이터 수집을 시작할 객체를 선택할 수 있습니다.

수동으로 추가한 데이터베이스에 OS 서버를 연결

DMS Fleet Advisor는 MySQL 및 PostgreSQL 데이터베이스에서 성능 지표를 직접 수집할 수 없습니다. 대상 권장 사항에 필요한 지표를 수집하기 위해 DMS Fleet Advisor는 데이터베이스가 실행되는 OS 지표를 사용합니다.

MySQL 및 PostgreSQL 데이터베이스를 모니터링되는 객체 목록에 수동으로 추가하는 경우, DMS 데이터 수집기는 이러한 데이터베이스가 실행되는 OS 서버를 식별할 수 없습니다. 이러한 문제가 있기 때문에 MySQL 및 PostgreSQL 데이터베이스를 OS 서버에 연결해야 합니다.

DMS Fleet Advisor가 자동으로 검색한 데이터베이스와 OS 서버를 수동으로 연결할 필요는 없습니다.

OS 서버를 데이터베이스에 연결하려면

1. DMS 데이터 수집기 탐색 창에서 모니터링되는 객체를 선택합니다.
2. 모니터링되는 객체 페이지에서 데이터베이스 서버 탭을 선택합니다. 데이터베이스 서버 목록이 나타납니다.

3. 수동으로 추가한 MySQL 또는 PostgreSQL 데이터베이스 서버 옆에 있는 확인란을 선택합니다.
4. 작업을 선택한 후, 편집을 선택합니다. 데이터베이스 편집 대화 상자가 열립니다.
5. 이 데이터베이스가 실행되는 OS 서버를 DMS 데이터 수집기가 이미 발견했다면 자동 감지를 선택합니다. DMS 데이터 수집기는 SQL 스크립트를 실행하여 데이터베이스가 실행되는 OS 서버를 자동으로 식별합니다. 그러면 DMS 데이터 수집기가 이 OS 서버를 데이터베이스에 연결합니다. 다음 단계를 건너뛴 후, 편집한 데이터베이스 구성을 저장합니다.

DMS 데이터 수집기가 데이터베이스의 OS 서버를 자동으로 식별할 수 없는 경우, 올바른 자격 증명을 사용하고 데이터베이스 액세스 권한을 제공해야 합니다. 필요에 따라 OS 서버를 수동으로 추가할 수 있습니다.

6. OS 서버를 수동으로 추가하려면 +OS 서버 추가를 선택합니다. 호스트 OS 서버 추가 대화 상자가 열립니다.

OS 서버에 관한 정보를 추가한 다음, 저장을 선택합니다.

7. 데이터베이스 편집 대화 상자에서 연결 확인을 선택하여 DMS 데이터 수집기가 OS 서버에 연결할 수 있는지 확인합니다.
8. 연결을 확인한 후 저장을 선택합니다.

소스 데이터베이스에 대한 관련 OS 서버를 변경하는 경우, DMS Fleet Advisor는 업데이트된 지표를 사용하여 권장 사항을 생성합니다. 하지만 Amazon CloudWatch 차트에는 데이터베이스 서버의 이전 데이터가 표시됩니다. CloudWatch 차트에 관한 자세한 내용은 [권장 사항 세부 정보](#) 섹션을 참조하세요.

AWS DMS Fleet Advisor에서 SSL 사용

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

데이터를 보호하기 위해 AWS DMS Fleet Advisor는 SSL을 사용하여 데이터베이스에 액세스할 수 있습니다.

지원되는 데이터베이스

AWS DMS Fleet Advisor는 SSL을 사용하여 다음 데이터베이스에 액세스할 수 있도록 지원합니다.

- Microsoft SQL Server
- MySQL
- PostgreSQL

SSL 설정

SSL을 사용하여 데이터베이스에 액세스하려면 SSL을 지원하도록 데이터베이스 서버를 구성하세요. 자세한 내용은 다음의 데이터베이스 설명서를 참조하세요.

- SQL Server: [데이터베이스 엔진에 암호화된 연결을 활성화](#)
- MySQL: [암호화된 연결을 사용하도록 MySQL 구성](#)
- PostgreSQL: [SSL을 이용한 TCP/IP 연결의 보안](#)

SSL을 사용하여 데이터베이스에 연결하려면 서버를 수동으로 추가할 때 서버 인증서 신뢰 및 SSL 사용을 선택합니다. MySQL 데이터베이스의 경우, 사용자 지정 인증서를 사용할 수 있습니다. 사용자 지정 인증서를 사용하려면 CA 확인 확인란을 선택합니다. 서버 추가에 관한 자세한 내용은 [모니터링되는 객체 관리](#) 섹션을 참조하세요.

SQL Server용 서버 인증 기관(CA) 인증서 확인

SQL Server용 서버 인증 기관(CA) 인증서의 유효성을 검사하려면 서버를 추가할 때 서버 인증서 신뢰를 지우세요. 서버가 잘 알려진 CA를 사용하고 있으며 해당 CA가 OS에 기본적으로 설치되어 있다면 확인은 정상적으로 작동할 것입니다. DMS Fleet Advisor가 데이터베이스 서버에 연결할 수 없는 경우, 데이터베이스 서버에서 사용하는 CA 인증서를 설치하세요. 자세한 내용은 [클라이언트 구성](#) 섹션을 참조하세요.

AWS DMS Fleet Advisor에 대한 데이터 수집

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘

솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

데이터 수집을 시작하려면 모니터링되는 객체 페이지에서 해당 객체를 선택하고 데이터 수집 실행을 선택합니다. DMS 데이터 수집기는 한 번에 최대 100개의 데이터베이스를 수집할 수 있습니다. 또한 DMS 데이터 수집기는 최대 8개의 병렬 스레드를 사용하여 사용자 환경의 데이터베이스에 연결할 수 있습니다. DMS 데이터 수집기는 이 8개의 스레드에서 최대 5개의 병렬 스레드를 사용하여 단일 데이터베이스 인스턴스에 연결할 수 있습니다.

Important

데이터 수집을 시작하기 전에 DMS 데이터 수집기 홈페이지의 소프트웨어 검사 섹션을 확인하세요. 모니터링하려는 모든 데이터베이스 엔진이 통과 상태인지 확인하세요. 일부 데이터베이스 엔진이 실패 상태이고 모니터링되는 객체 목록에 해당 엔진이 포함된 데이터베이스 서버가 있는 경우, 계속 진행하기 전에 문제를 해결하세요. 소프트웨어 검사 섹션에 나열된 실패 상태 옆에 위치한 팁을 확인할 수 있습니다.

DMS 데이터 수집기는 두 가지 모드 즉, 단일 실행 모드 또는 지속적 모니터링 모드로 작동할 수 있습니다. 데이터 수집을 시작하면 데이터 수집 실행 대화 상자가 열립니다. 이어서 다음 옵션 중 하나를 선택합니다.

메타데이터 및 데이터베이스 용량

DMS 데이터 수집기는 데이터베이스 또는 OS 서버에서 정보를 수집합니다. 여기에는 스키마, 버전, 에디션, CPU, 메모리 및 디스크 용량이 포함됩니다. 또한 DMS 데이터 수집기는 IOPS, I/O 처리량 및 활성 데이터베이스 서버 연결과 같은 지표를 수집하고 제공합니다. 이 정보를 기반으로 DMS Fleet Advisor에서 대상 권장 사항을 고려할 수 있습니다. 소스 데이터베이스가 과다 프로비저닝되거나 과소 프로비저닝된 경우, 대상 권장 사항도 과다 프로비저닝되거나 과소 프로비저닝됩니다.

이것은 기본 옵션에 해당됩니다.

메타데이터, 데이터베이스 용량 및 리소스 사용률

DMS 데이터 수집기는 메타데이터 및 데이터베이스 용량 정보 외에도 데이터베이스 또는 OS 서버의 CPU, 메모리 및 디스크 용량에 대한 실제 사용률 지표를 수집합니다. 또한 DMS 데이터 수집기는 IOPS, I/O 처리량 및 활성 데이터베이스 서버 연결과 같은 지표를 수집하고 제공합니다. 제공된 대상 권장 사항은 실제 데이터베이스 워크로드를 기반으로 하기 때문에 더 정확할 것입니다.

이 옵션을 선택하는 경우, 데이터 수집 기간을 설정합니다. 다음 7일 동안 데이터를 수집하거나 사용자 지정 범위를 1~60일로 설정할 수 있습니다.

데이터 수집이 시작되면 데이터 수집 페이지로 리디렉션됩니다. 이 페이지에서 수집 쿼리의 실행 방식을 확인하고 실시간 진행 상황을 모니터링할 수 있습니다. 여기에서 전체 수집 상태를 보거나 DMS 데이터 수집기 홈페이지에서 확인할 수 있습니다. 전체 데이터 수집 상태가 100% 미만인 경우, 수집과 관련된 문제를 해결해야 할 수 있습니다.

메타데이터 및 데이터베이스 용량 모드에서 DMS 데이터 수집기를 실행하면 데이터 수집 페이지에서 완료된 쿼리 수를 확인할 수 있습니다.

메타데이터, 데이터베이스 용량 및 리소스 사용률 모드에서 DMS 데이터 수집기를 실행하면 DMS 데이터 수집기가 모니터링을 완료하기까지 남은 시간을 확인할 수 있습니다.

데이터 수집 페이지에서 각 객체의 수집 상태를 볼 수 있습니다. 무언가가 제대로 작동하지 않는 경우, 발생한 문제의 수를 보여주는 메시지가 나타납니다. 세부 정보를 확인하면 문제의 해결 방법을 결정하는 데 도움이 됩니다. 다음 탭에는 잠재적 문제들이 나열되어 있습니다.

- 쿼리별 요약 - Ping 테스트와 같은 테스트의 상태를 표시합니다. 상태 열에서 결과를 필터링할 수 있습니다. 상태 열은 데이터 수집 중에 발생한 오류 수를 나타내는 메시지를 제공합니다.
- 모니터링되는 객체별 요약 - 객체별 전체 상태를 보여줍니다.
- 쿼리 유형별 요약 - SQL, Secure Shell(SSH) 또는 WMI(Windows Management Instrumentation) 호출 등 수집기 쿼리 유형의 상태를 표시합니다.
- 문제별 요약 - 발생한 모든 고유한 문제를 문제 이름 및 각 문제의 발생 횟수와 함께 표시합니다.

🏠

📁

📄

📊

📊 Data collection

Export to CSV

Collection in progress...

Metadata, database capacity, and resource utilization data are being collected. Make sure you have proper connectivity to OS and database servers.

0 d 23 hr 9 min remains

⛔ Stop collection

Summary by query

Summary by monitored object

Summary by query type

Summary by issue

Monitored object add... ↓	Co...	Query name	User name	Engine	Time	Status
10.100.11.241:22	SSH	Linux CPU Stat	dbmuser	Linux	12-01-2023 03:48:30	✔ Complete
10.100.11.241:22	SSH	Linux MemInfo	dbmuser	Linux	12-01-2023 03:48:29	✔ Complete
10.100.11.241:22	SSH	Linux CPU Info	dbmuser	Linux	12-01-2023 02:57:30	✔ Complete
10.100.11.241:5432	Pgsql	AWS RDS Limitations (Database Level)	FA_Collect_User	PostgreSQL	12-01-2023 02:57:29	✔ Complete

Total items: 13

수집 결과를 내보내려면 CSV로 내보내기를 선택합니다.

문제를 식별하고 해결한 후 수집 시작을 선택하고 데이터 수집 프로세스를 다시 실행합니다. 데이터 수집을 수행한 후 데이터 수집기는 보안 연결을 사용하여 수집된 데이터를 DMS Fleet Advisor 인벤토리에 업로드합니다. DMS Fleet Advisor는 Amazon S3 버킷에 정보를 저장합니다. 데이터 전달을 위한 자격 증명 구성에 관한 자세한 내용은 [데이터 전달을 위한 자격 증명 구성](#)을 참조하세요.

AWS DMS Fleet Advisor를 사용하여 용량 및 리소스 사용률 지표 수집

메타데이터와 성능 지표를 두 가지 모드 즉, 단일 실행 모드 또는 지속적 모니터링 모드로 수집할 수 있습니다. 선택한 옵션에 따라 DMS 데이터 수집기는 데이터 환경에서 다양한 지표를 추적합니다. 단일 실행 모드 중 DMS 데이터 수집기는 데이터베이스 및 OS 서버의 메타데이터 지표만 추적합니다. 지속적 모니터링 모드 중 DMS 데이터 수집기는 리소스의 실제 사용률을 추적합니다.

AWS DMS 는 DMS 데이터 수집기를 한 번 실행하는 동안 다음과 같은 메타데이터와 지표를 수집합니다.

- OS 서버의 가용 메모리
- OS 서버의 가용 스토리지
- 데이터베이스 버전 및 에디션
- OS 서버의 CPU 수

- 스키마 수
- 저장된 프로시저의 수
- 테이블 수
- 트리거 수
- 조회수
- 스키마 구조

DMS Fleet Advisor는 이러한 지표를 사용하여 데이터베이스 및 OS 서버의 인벤토리를 구축합니다. 또한 DMS Fleet Advisor는 이러한 메타데이터 및 지표를 사용하여 소스 데이터베이스 스키마를 분석합니다.

DMS Fleet Advisor는 데이터 수집기의 단일 실행 중에 수집된 지표를 사용하여 대상 권장 사항을 생성할 수 있습니다. 그러나 이 경우, 과다 프로비저닝된 소스 데이터베이스에서 대상 권장 사항도 과다 프로비저닝됩니다. 따라서 AWS 클라우드의 리소스를 유지 관리하는 데 추가 비용이 발생합니다. 과소 프로비저닝된 소스 데이터베이스의 경우, 대상 권장 사항도 과소 프로비저닝되어 성능 문제가 발생할 수 있습니다. DMS 데이터 수집기의 메타데이터, 데이터베이스 용량 및 리소스 사용률 모드를 선택하여 지속적인 모니터링을 사용하여 데이터를 수집하는 것이 좋습니다.

AWS DMS 는 지속적인 모니터링 중에 다음 지표를 수집합니다. 1~60일 동안 DMS 데이터 수집기를 실행할 수 있습니다.

- 데이터베이스 서버의 I/O 처리량
- 데이터베이스 서버의 초당 입출력 작업량(IOPS)
- OS 서버에서 사용하는 CPU의 수
- OS 서버의 메모리 사용량
- 활성 데이터베이스 및 OS 서버 연결 수

DMS Fleet Advisor는 이러한 지표를 사용하여 정확한 대상 권장 사항을 생성하므로 대상 데이터베이스가 성능 요구 사항을 충족할 수 있습니다. 이렇게 하면 AWS 클라우드에서 리소스를 유지 관리하는 데 발생하는 추가 비용을 방지할 수 있습니다.

AWS DMS Fleet Advisor는 용량 및 리소스 사용률 지표를 어떻게 수집하나요?

DMS Fleet Advisor는 1분마다 성능 지표를 수집합니다.

Oracle 및 SQL Server의 경우, DMS Fleet Advisor는 SQL 쿼리를 실행하여 각 데이터베이스 지표의 값을 캡처합니다.

MySQL 및 PostgreSQL의 경우, DMS Fleet Advisor는 데이터베이스가 실행되는 OS 서버에서 성능 지표를 수집합니다. Windows에서 DMS Fleet Advisor는 WQL(WMI Query Language) 스크립트를 실행하고 WMI 데이터를 수신합니다. Linux에서 DMS Fleet Advisor는 OS 서버 지표를 캡처하는 명령을 실행합니다.

 Important

원격 SQL 스크립트를 실행하면 프로덕션 데이터베이스의 성능에 영향을 미칠 수 있습니다. 하지만 데이터 수집 쿼리에는 계산 로직이 포함되어 있지 않습니다. 따라서 데이터 수집 프로세스에서는 데이터베이스 리소스의 1% 이상을 사용할 가능성이 없습니다.

데이터 수집기가 지표를 수집하기 위해 실행하는 모든 쿼리를 볼 수 있습니다. 이렇게 하려면 `DMSCollector.Collections.json` 파일을 여세요. 이 파일은 데이터 수집기를 설치한 동일한 폴더에 있는 `etc` 폴더에서 찾을 수 있습니다. 기본 경로는 `C:\ProgramData\Amazon\AWS DMS Collector\etc\DMSCollector.Collections.json`입니다.

DMS 데이터 수집기는 수집된 모든 데이터의 임시 저장소로 로컬 파일 시스템을 사용합니다. DMS 데이터 수집기는 수집된 데이터를 JSON 형식으로 저장합니다. 오프라인 모드에서 로컬 수집기를 사용하고 데이터 전달을 구성하기 전에 수집된 파일을 수동으로 검사하거나 확인할 수 있습니다. 수집된 모든 파일은 DMS 데이터 수집기를 설치한 동일한 폴더에 있는 `out` 폴더에서 확인할 수 있습니다. 기본 경로는 `C:\ProgramData\Amazon\AWS DMS Collector\out`입니다.

 Important

DMS 데이터 수집기를 오프라인 모드에서 실행하고 수집된 데이터를 서버에 14일 이상 저장하는 경우, Amazon CloudWatch를 사용하여 이러한 지표를 표시할 수는 없습니다. 하지만 DMS Fleet Advisor는 여전히 이 데이터를 사용하여 권장 사항을 생성합니다. CloudWatch 차트에 관한 자세한 내용은 [권장 사항 세부 정보](#) 섹션을 참조하세요.

온라인 모드에서 수집된 데이터 파일을 검사하거나 확인할 수도 있습니다. DMS 데이터 수집기는 모든 데이터를 DMS 데이터 수집기 설정에서 지정한 Amazon S3 버킷으로 전달합니다.

DMS 데이터 수집기를 사용하여 온프레미스 데이터베이스에서 데이터를 수집할 수 있습니다. 또한 Amazon RDS 및 Aurora 데이터베이스에서 데이터를 수집할 수 있습니다. 하지만 Amazon RDS 또는 Aurora와 온프레미스 DB 인스턴스 간의 차이로 인해 클라우드에서 모든 DMS 데이터 수집기 쿼리를 성공적으로 실행할 수는 없습니다. DMS 데이터 수집기는 호스트 OS에서 MySQL 및 PostgreSQL 데

이터베이스의 사용률 지표를 수집하기 때문에 Amazon RDS 및 Aurora에서는 이 방법을 사용할 수 없습니다.

DMS 데이터 수집기 문제 해결

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

다음 목록에서는 데이터 수집기로 데이터를 수집하는 동안 특정 문제가 발생할 때 취할 수 있는 조치를 찾을 수 있습니다.

주제

- [의 네트워크 및 서버 연결과 관련된 데이터 수집 문제 해결 AWS DMS](#)
- [에서 Windows Management Instrumentation과 관련된 데이터 수집 문제 해결 AWS DMS](#)
- [Windows 웹 페이지 작성기와 관련된 데이터 수집 문제 해결](#)
- [에서 SSL과 관련된 데이터 수집 문제 해결 AWS DMS](#)

의 네트워크 및 서버 연결과 관련된 데이터 수집 문제 해결 AWS DMS

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 Fleet Advisor에 대한 AWS Database Migration Service 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS 데이터 수집기와의 네트워크 및 서버 연결과 관련된 문제가 발생하면 다음 작업을 시도하세요.

NET: ping 요청 중에 예외가 발생했습니다.

컴퓨터의 이름을 확인하여 IP 주소로 확인할 수 없는 상태인지 확인합니다.

예를 들어 컴퓨터가 꺼져 있는지, 네트워크 연결이 끊겼는지, 사용 중지되었는지 확인합니다.

NET: 시간 초과됨

인바운드 방화벽 규칙 '파일 및 프린터 공유(에코 요청 - ICMPv4-In)'를 활성화합니다. 예제:

* Inbound ICMPv4

NET: DestinationHostUnreachable

컴퓨터의 IP 주소를 확인합니다. 특히 DMS 데이터 수집기를 실행하는 컴퓨터와 동일한 서브넷에 있는지, 주소 확인 프로토콜(ARP) 요청에 응답하는지 확인하세요.

컴퓨터가 다른 서브넷에 있는 경우, 게이트웨이의 IP 주소를 미디어 액세스 제어(MAC) 주소로 확인할 수 없습니다.

또한 컴퓨터가 꺼져 있는지, 네트워크 연결이 끊겼는지, 사용 중지되었는지 확인합니다.

에서 Windows Management Instrumentation과 관련된 데이터 수집 문제 해결 AWS DMS

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 Fleet Advisor에 대한 AWS Database Migration Service 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS 데이터 수집기에서 WMI(Windows Management Instrumentation)과 관련된 문제가 발생하면 다음 작업을 시도하세요.

WMI: RPC 서버를 사용할 수 없습니다. (HRESULT 예외: 0x800706BA)

인바운드 방화벽 규칙 'Windows Management Instrumentation(DCOM-in)'을 활성화합니다. 예제:

* Inbound TCP/IP at local port 135.

또한 인바운드 방화벽 규칙 'Windows Management Instrumentation(WMI-In)'도 활성화합니다. 예제:

* Inbound TCP/IP at local port 49152 - 65535: Windows Server 2008 및 이후 버전용.

* Inbound TCP/IP at local port 1025 - 5000: Windows Server 2003 및 이전 버전용.

WMI: 액세스가 거부되었습니다. (HRESULT 예외: 0x80070005(E_ACCESSDENIED))

다음을 시도해 보세요.

- DMS 데이터 수집기 사용자를 Windows 그룹, 분산 COM 사용자 또는 관리자에 추가합니다.
- WMI(Windows Management Instrumentation) 서비스를 시작하고 시작 유형을 자동으로 설정합니다.
- DMS 데이터 수집기 사용자 이름이 \ 형식인지 확인하세요.

WMI: 액세스가 거부됨

루트 WMI 네임스페이스의 DMS 데이터 수집기 사용자에게 원격 활성화 권한을 추가합니다.

고급 설정을 사용하고 권한이 '이 네임스페이스 및 하위 네임스페이스'에 적용되는지 확인하세요.

WMI: 메시지 필터에 의해 호출이 취소되었습니다. (HRESULT 예외: 0x80010002...)

WMI(Windows Management Instrumentation) 서비스를 다시 시작합니다.

Windows 웹 페이지 작성기와 관련된 데이터 수집 문제 해결

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 Fleet Advisor에 대한 AWS Database Migration Service 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS 데이터 수집기에서 Windows 웹 페이지 작성기와 관련된 문제가 발생하면 다음 작업을 시도하세요.

WPC: 네트워크 경로를 찾을 수 없습니다

인바운드 방화벽 규칙 '파일 및 프린터 공유(SMB-in)'를 활성화합니다. 예제:

* Inbound TCP/IP at local port 445.

또한 원격 레지스트리 서비스를 시작하고 시작 유형을 자동으로 설정합니다.

WPC: 액세스 거부됨

DMS 데이터 수집기 사용자를 성능 모니터 사용자 또는 관리자 그룹에 추가합니다.

WPC: 카테고리가 존재하지 않음

loader /r을 실행하여 성능 카운터 캐시를 재구축한 다음, 컴퓨터를 다시 시작합니다.

Note

AWS Database Migration Service (AWS DMS)를 사용하여 데이터를 마이그레이션할 때 발생하는 문제 해결에 대한 자세한 내용은 [문제 해결 및 진단 지원을 참조하세요](#).

에서 SSL과 관련된 데이터 수집 문제 해결 AWS DMS

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 Fleet Advisor에 대한 AWS Database Migration Service 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS 데이터 수집기에서 SSL과 관련된 문제가 발생하면 다음 작업을 시도하세요.

SSL 오류

데이터베이스에 보안 SSL 연결이 필요한데 해당 연결에 대해 CA 확인 및 SSL 사용 옵션을 활성화하지 않았습니다. 이 두 옵션을 활성화한 후 데이터베이스에서 사용하는 인증 기관(CA)이 로컬 OS에 설치되어 있는지 확인하세요. 자세한 내용은 [SSL 설정](#) 단원을 참조하십시오.

AWS DMS Fleet Advisor에서 분석에 인벤토리 사용

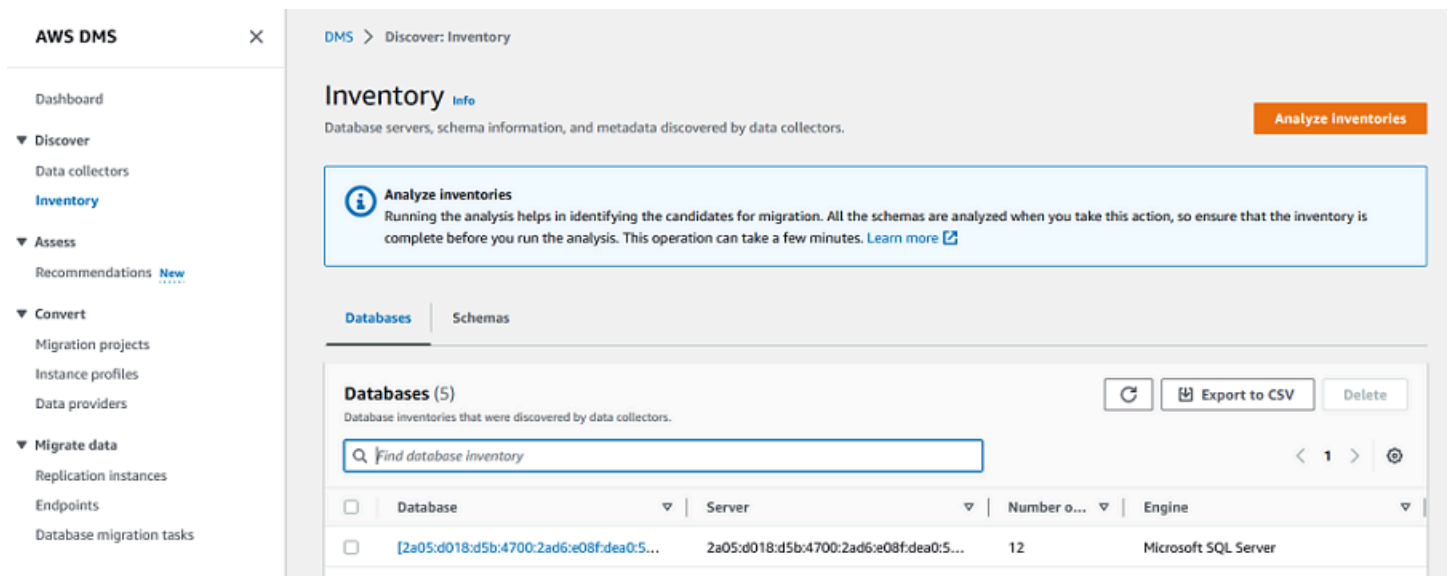
Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘

솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

검색된 데이터베이스 및 스키마의 인벤토리를 활용하면 잠재적 데이터베이스 마이그레이션의 타당성을 확인할 수 있습니다. 이러한 인벤토리의 정보를 사용하여 마이그레이션에 적합한 데이터베이스 및 스키마를 파악할 수 있습니다.

콘솔에서 데이터베이스 및 스키마 인벤토리에 액세스할 수 있습니다. 이렇게 하려면 콘솔에서 인벤토리를 선택합니다.



DMS Fleet Advisor는 데이터베이스 스키마를 분석하여 여러 스키마의 유사성을 확인합니다. 이 분석에서는 객체의 실제 코드를 비교하지 않습니다. DMS Fleet Advisor는 함수 및 프로시저와 같은 스키마 객체의 이름만 비교하여 서로 다른 데이터베이스 스키마에서 유사한 객체를 식별합니다.

주제

- [에서 분석을 위한 데이터베이스 인벤토리 사용 AWS DMS](#)
- [에서 분석에 스키마 인벤토리 사용 AWS DMS](#)

에서 분석을 위한 데이터베이스 인벤토리 사용 AWS DMS

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘

솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

데이터가 수집된 네트워크 내 검색된 모든 서버의 모든 데이터베이스 목록을 보려면 다음 절차를 적용하세요.

데이터를 수집한 출처에 해당하는 네트워크 서버의 데이터베이스 목록을 보려면

1. 콘솔에서 인벤토리를 선택합니다.

인벤토리 페이지가 열립니다.

2. 데이터베이스 탭을 선택합니다.

검색된 데이터베이스의 목록이 나타납니다.

The screenshot shows the 'Inventory' page in the AWS DMS Fleet Advisor console. It features a header with 'Inventory' and an 'Info' icon, a sub-header 'Database servers, schema information, and metadata discovered by data collectors.', and an 'Analyze inventories' button. Below this is a blue box with an information icon and text explaining the 'Analyze inventories' action. The main content area has two tabs: 'Databases' (selected) and 'Schemas'. Under the 'Databases' tab, there's a section titled 'Databases (7)' with a refresh button, 'Export to CSV', and 'Delete' buttons. A search bar is present with the placeholder 'Find database inventory'. Below the search bar is a table with the following columns: 'Database', 'Server', 'Number of s...', 'Engine', 'Engine version', and 'Engine ...'. The table contains three rows of data:

Database	Server	Number of s...	Engine	Engine version	Engine ...
WinServ2016.d...	-	No data	PostgreSQL	-	-
VM-MSSQL14-...	10.11.1.10	44	Microsoft SQL ...	2014 (Extended support)	Enterprise
MSSQL01.dbm...	-	No data	Microsoft SQL ...	2019 (Mainstream support)	Express

3. 인벤토리 분석을 선택하여 유사성 및 복잡성과 같은 스키마 속성을 확인합니다. 프로세스에 걸리는 시간은 분석할 객체의 수에 따라 차이가 있으며 다만 1시간 이상 걸리지는 않습니다. 분석 결과는 인벤토리 페이지에 있는 스키마 탭에서 확인할 수 있습니다.

DMS Fleet Advisor는 모든 검색된 데이터베이스에서 스키마를 분석하여 해당 객체의 교차율을 정의합니다. 분석 결과는 백분율로 표시됩니다. DMS Fleet Advisor는 객체 교차율이 50% 이상인 스

키마를 중복으로 간주합니다. 원본 스키마는 중복이 발견된 스키마로 식별됩니다. 이렇게 하면 먼저 변환하거나 마이그레이션할 원본 스키마를 식별하는 데 도움이 됩니다.

전체 인벤토리를 함께 분석하여 중복된 스키마를 식별합니다.

에서 분석에 스키마 인벤토리 사용 AWS DMS

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

데이터가 수집된 네트워크 내 서버에서 검색된 데이터베이스 스키마의 목록을 볼 수 있습니다. 다음 절차를 수행합니다.

데이터를 수집한 출처에 해당하는 네트워크 서버의 스키마 목록을 보려면

1. 콘솔에서 인벤토리를 선택합니다. 인벤토리 페이지가 열립니다.
2. 스키마 탭을 선택합니다. 스키마 목록이 나타납니다.
3. 목록에서 스키마를 선택하면 서버, 데이터베이스, 크기, 복잡성 등 스키마에 관한 정보를 볼 수 있습니다.

각 스키마에 대해 객체 유형, 객체 수, 객체 크기 및 코드 줄에 관한 정보를 제공하는 객체 요약을 볼 수 있습니다.

4. (선택 사항) 인벤토리 분석을 선택하여 중복된 스키마를 식별합니다. DMS Fleet Advisor는 데이터베이스 스키마를 분석하여 해당 객체의 교차율을 정의합니다.
5. 추가 검토를 위해 인벤토리 정보를 .csv 파일로 내보낼 수 있습니다.

Analysis complete
Schema inventory

DMS > Discover: Inventory

Inventory Info

Database servers, schema information, and metadata discovered by data collectors.

Analyze inventories
Running the analysis helps in identifying the candidates for migration. All the schemas are analyzed when you take this action, so ensure that the inventory is complete before you run the analysis. This operation can take a few minutes. [Learn more](#)

Analyze inventories

Databases | **Schemas**

Schemas (13)
Schema inventories that were discovered by data collectors.

Find schema inventory
< 1 >

Schema	Server	Database	Engine	Complexity	Similarity...	Original schema
lsa_tests_src_lsa_tests_src	linuxsql02.db.local	linuxsql02.db.local:3306	MySQL Server	Simple	100	lsa_tests_src_100.lsa_tests_s...
lsa_tests_src_90e_30a.lsa_t...	linuxsql02.db.local	linuxsql02.db.local:3306	MySQL Server	Simple	90	lsa_tests_src_49.lsa_tests_sr...
lsa_tests_src_50.lsa_tests_s...	linuxsql02.db.local	linuxsql02.db.local:3306	MySQL Server	Simple	50	lsa_tests_src_100.lsa_tests_s...
lsa_tests_src_49.lsa_tests_s...	linuxsql02.db.local	linuxsql02.db.local:3306	MySQL Server	Simple	-	None

AWS Schema Conversion Tool (AWS SCT) 또는 DMS Schema Conversion을 사용하면 마이그레이션할 스키마를 식별하고 마이그레이션 대상을 결정할 수 있습니다. 자세한 내용은 [에서 새 프로젝트 마법사 사용을 AWS SCT](#) 참조하세요.

마이그레이션할 스키마를 식별한 후 AWS SCT 또는 DMS Schema Conversion을 사용하여 스키마를 변환할 수 있습니다. DMS Schema Conversion에 관한 자세한 내용은 [DMS Schema Conversion을 사용하여 데이터베이스 스키마 변환](#) 섹션을 참조하세요.

AWS DMS Fleet Advisor 대상 권장 사항 기능 사용

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

최적의 마이그레이션 대상을 탐색하고 선택하기 위해 DMS Fleet Advisor에서 소스 온프레미스 데이터베이스에 대한 대상 권장 사항을 생성할 수 있습니다. 권장 사항에는 소스 온프레미스 데이터베이스

스 마이그레이션을 위해 선택할 수 있는 하나 이상의 가능한 AWS 대상 엔진이 포함됩니다. DMS Fleet Advisor는 이들 가능한 대상 엔진 중에서 단일 대상 엔진을 적정 규모의 마이그레이션 대상으로 제안하고 이 대상을 DMS에서 권장하는 대상으로 표시합니다. 적정 규모의 마이그레이션 대상을 결정하기 위해 DMS Fleet Advisor는 데이터 수집기에서 수집한 인벤토리 메타데이터와 지표를 사용합니다.

마이그레이션을 시작하기 전에 권장 사항을 사용하여 마이그레이션 옵션을 검색하고 비용을 절감하며 위험을 줄일 수 있습니다. 권장 사항을 CSV(쉼표로 구분된 값) 파일로 내보내고 이를 주요 이해 관계자와 공유하여 의사 결정을 촉진할 수 있습니다. 권장 사항을 로 내보내 유지 관리 비용을 추가로 최적화 AWS Pricing Calculator 할 수 있습니다. 자세한 내용은 <https://calculator.aws/#/>를 참조하세요.

DMS Fleet Advisor에서는 대상 권장 사항을 수정할 수 없습니다. 따라서 가정 분석에는 DMS Fleet Advisor를 사용할 수 없습니다. 가정 분석은 대상 파라미터를 변경하여 이러한 변경이 권장 사항의 예상 요금에 어떤 영향을 미치는지 확인하는 프로세스입니다. 권장 대상 파라미터들의 시작점으로 AWS Pricing Calculator 사용하여 what-if 분석을 실행할 수 있습니다 AWS Pricing Calculator. 자세한 내용은 <https://calculator.aws/#/>를 참조하세요.

DMS Fleet Advisor 권장 사항을 마이그레이션 계획의 출발점으로 삼는 것이 좋습니다. 그런 다음, 권장 인스턴스 파라미터를 변경하기로 결정하면 데이터베이스 워크로드의 비용 또는 성능을 최적화할 수 있습니다.

주제

- [권장 대상 인스턴스](#)
- [DMS Fleet Advisor는 권장 사항의 대상 인스턴스 사양을 어떻게 결정합니까?](#)
- [AWS DMS Fleet Advisor를 사용하여 대상 권장 사항 생성](#)
- [AWS DMS Fleet Advisor를 사용하여 대상 권장 사항의 세부 정보 탐색](#)
- [AWS DMS Fleet Advisor를 사용하여 대상 권장 사항 내보내기](#)
- [AWS DMS Fleet Advisor를 사용하여 마이그레이션 제한 사항 검색 및 분석](#)
- [AWS DMS Fleet Advisor를 사용한 대상 권장 사항 문제 해결](#)

권장 대상 인스턴스

대상 권장 사항의 경우, DMS Fleet Advisor는 메모리에 최적화되고 버스트 가능한 다음과 같은 범용 Amazon RDS DB 인스턴스를 고려합니다.

- db.m5
- db.m6i

- db.r5
- db.r6i
- db.t3
- db.x1
- db.x1e
- db.z1d

Amazon RDS DB 인스턴스 클래스에 관한 자세한 내용은 Amazon RDS 사용 설명서의 [DB 인스턴스 클래스](#)를 참조하세요.

DMS Fleet Advisor는 권장 사항의 대상 인스턴스 사양을 어떻게 결정합니까?

DMS Fleet Advisor는 데이터베이스 용량 또는 사용률을 기반으로 권장 사항을 생성할 수 있습니다.

- 데이터베이스 용량을 기반으로 권장 사항을 생성하도록 선택하면 DMS Fleet Advisor는 기존 데이터베이스 용량을 가장 가까운 인스턴스 클래스의 사양에 매핑합니다.
- 리소스 사용률을 기반으로 권장 사항을 생성하도록 선택하면 DMS Fleet Advisor가 CPU, 메모리, IO 처리량 및 IOPS와 같은 지표의 95번째 백분위수 값을 결정합니다. 95번째 백분위수는 수집된 데이터의 95%가 이 값보다 낮다는 것을 의미합니다. 그러면 DMS Fleet Advisor는 이 값을 가장 가까운 인스턴스 클래스의 사양에 매핑합니다.

대상 데이터베이스의 크기를 결정하기 위해 DMS Fleet Advisor는 소스 데이터베이스의 크기에 관한 정보를 수집합니다. 그러면 DMS Fleet Advisor는 대상 스토리지에 동일한 크기를 사용할 것을 권장합니다. 소스 데이터베이스 스토리지가 과다 프로비저닝되면 대상 스토리지의 권장 크기도 과다 프로비저닝됩니다.

를 사용하여 데이터를 마이그레이션하려면 대상 DB 인스턴스에 대한 IOPS 프로비저닝을 늘려야 할 AWS DMS 수 있습니다. DMS Fleet Advisor가 대상 권장 사항을 생성할 때 서비스는 소스 데이터베이스 지표만 고려합니다. DMS Fleet Advisor는 데이터 마이그레이션 작업을 실행하는 데 필요할 수 있는 추가 IOPS를 고려하지 않습니다. 자세한 내용은 [마이그레이션 태스크가 느리게 실행됨](#) 단원을 참조하십시오.

IOPS 비용을 추정하기 위해 DMS Fleet Advisor는 소스 IOPS 사용량의 일대일 매핑을 기준으로 사용합니다. DMS Fleet Advisor는 피크 로드를 기준선 값으로 간주하고 IOPS 요금 계산 시 100% 사용률을 고려합니다.

PostgreSQL 및 MySQL 소스 데이터베이스의 경우, DMS Fleet Advisor는 대상 권장 사항에 Aurora 및 Amazon RDS DB 인스턴스를 포함할 수 있습니다. Aurora 구성이 소스 요구 사항에 매핑되는 경우, DMS Fleet Advisor는 이 옵션을 권장 옵션으로 표시합니다.

AWS DMS Fleet Advisor를 사용하여 대상 권장 사항 생성

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

데이터베이스 및 분석 플릿의 데이터 수집 및 인벤토리를 완료한 후 DMS Fleet Advisor에서 대상 권장 사항을 생성할 수 있습니다. 이렇게 하려면 소스 데이터베이스를 선택하고 DMS Fleet Advisor 대상 권장 기능이 대상 인스턴스의 크기를 결정하는 데 사용하는 설정을 구성하세요. 또한 DMS Fleet Advisor 대상 권장 기능은 소스 데이터베이스에서 수집한 용량 및 사용률 지표를 사용합니다.

대상 권장 사항을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

DMS Fleet Advisor를 사용하는 AWS 리전 를 선택해야 합니다.

2. 탐색 창의 평가에서 권장 사항을 선택한 다음, 권장 사항 생성을 선택합니다.
3. 소스 데이터베이스 선택 패널에서 AWS 클라우드로 마이그레이션할 데이터베이스의 이름 옆에 있는 확인란을 선택합니다.

소스 데이터베이스 검색에 데이터베이스의 이름을 입력하여 인벤토리를 필터링합니다.

DMS Fleet Advisor는 한 번에 최대 100개의 데이터베이스에 대한 권장 사항을 생성할 수 있습니다.

4. 가용성 및 내구성을 위해 선호하는 배포 옵션을 선택하세요.

프로덕션 데이터베이스의 대상 권장 사항을 계산하려면 프로덕션(다중 AZ)을 선택합니다. DMS Fleet Advisor는 대상 권장 사항에 서로 다른 가용 영역(AZ)에 있는 두 개의 DB 인스턴스를 포함합니다. 이 다중 AZ 배포 옵션은 고가용성, 데이터 중복 및 장애 조치 지원을 제공합니다.

Aurora가 권장 대상 엔진이고 가용성 및 내구성이 다중 AZ 배포인 경우 대상 권장 사항에는 리더 및 라이터 DB 인스턴스가 포함됩니다.

개발 또는 테스트에 사용하는 데이터베이스의 대상 권장 사항을 계산하려면 개발 및 테스트(단일 AZ)를 선택합니다. DMS Fleet Advisor는 대상 권장 사항에 단일 DB 인스턴스를 포함합니다. 이 단일 AZ 배포 옵션은 유지 관리 비용을 줄여줍니다.

5. 대상 인스턴스 크기 조정에서는 DMS Fleet Advisor가 대상 권장 사항을 계산하는 데 사용하는 선호 옵션을 선택하세요.

소스 데이터베이스 또는 OS 서버 구성을 기반으로 대상 권장 사항을 계산하려면 총 용량을 선택합니다. DMS Fleet Advisor는 소스 데이터베이스 또는 OS 서버의 총 CPU, 메모리, 디스크 용량과 같은 지표를 사용하여 대상 권장 사항을 생성합니다. 그런 다음, DMS Fleet Advisor는 데이터베이스 용량 지표를 가장 가까운 Amazon RDS DB 인스턴스 클래스의 사양에 매핑합니다.

소스 데이터베이스 또는 OS 서버의 실제 사용량을 기반으로 대상 권장 사항을 계산하려면 리소스 사용량을 선택합니다. DMS Fleet Advisor는 소스 데이터베이스 또는 OS 서버의 CPU, 메모리 및 디스크 용량에 대한 사용률 지표를 사용하여 대상 권장 사항을 생성합니다. DMS Fleet Advisor는 사용률 지표에서 각 지표의 95번째 백분위수를 계산합니다. 95번째 백분위수는 기간 내 데이터의 95%가 이 값보다 낮음을 의미합니다. 그러면 DMS Fleet Advisor는 이 값을 가장 가까운 Amazon RDS DB 인스턴스 클래스에 매핑합니다.

더 정확한 권장 사항을 확인하려면 리소스 사용률 옵션을 사용하는 것이 좋습니다. 이를 위해 총 용량 및 리소스 사용률 지표를 수집했는지 확인하세요.

6. 생성을 선택합니다.

DMS Fleet Advisor는 선택한 데이터베이스에 대한 대상 권장 사항을 생성합니다. 권장 사항을 성공적으로 생성한 경우, DMS Fleet Advisor는 상태를 계산됨으로 설정합니다. 또한 DMS Fleet Advisor는 AWS Pricing Calculator 를 사용하여 권장 대상 DB 인스턴스의 월별 예상 비용을 결정합니다. 이제 생성된 권장 사항의 자세한 내용을 살펴볼 수 있습니다. 자세한 내용은 [권장 사항 세부 정보](#) 단원을 참조하십시오.

데이터 인벤토리의 월별 총 비용을 추정하려면 클라우드로 이전하려는 데이터베이스의 확인란을 선택합니다. DMS Fleet Advisor는 월별 총 예상 비용과 AWS 클라우드내 대상 데이터베이스의 요약 정보를 표시합니다. DMS Fleet Advisor는 AWS 가격표 쿼리 API를 사용하여 사용자 정보에 대한 요금 세부 정보만 제공합니다. 실제 요금은 AWS 서비스의 실제 사용량을 포함한 다양한 요인에 따라 달라집니다. AWS 서비스 요금에 대한 자세한 내용은 [클라우드 서비스 요금](#)을 참조하세요.

AWS DMS Fleet Advisor를 사용하여 대상 권장 사항의 세부 정보 탐색

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS Fleet Advisor가 대상 권장 사항을 생성한 후 권장 사항 테이블에서 권장 마이그레이션 대상의 주요 파라미터를 볼 수 있습니다. 이러한 주요 파라미터에는 대상 엔진, 인스턴스 클래스, 가상 CPU 수, 메모리, 스토리지 및 스토리지 유형이 포함됩니다. DMS Fleet Advisor는 이러한 파라미터 외에도 이 권장 마이그레이션 대상의 월별 예상 비용을 표시합니다.

각 권장 사항에는 하나 이상의 가능한 AWS 대상 엔진이 포함될 수 있습니다. 권장 사항에 여러 대상 엔진이 포함된 경우는이 중 하나를 권장 사항으로 AWS DMS 표시합니다. 또한 권장 사항 테이블에이 권장 옵션의 파라미터와 예상 월별 비용을 AWS DMS 표시합니다.

대상 권장 사항을 소스 데이터베이스의 사용률 및 용량과 비교하려면 권장 사항의 세부 내용을 살펴보세요. 또한 선택한 권장 사항에 대한 마이그레이션 제한 사항을 볼 수 있습니다. 이러한 제한 사항에는 지원되지 않는 데이터베이스 기능, 작업 항목 및 기타 마이그레이션 고려 사항이 포함됩니다.

권장 사항의 자세한 내용을 살펴보려면

1. DMS Fleet Advisor를 사용하여 대상 권장 사항을 생성하세요. 자세한 내용은 [대상 권장 사항 생성](#) 단원을 참조하십시오.
2. 권장 사항 테이블의 권장 사항 이름을 선택합니다. 권장 사항 페이지가 열립니다.
3. 권장 사항에 대상 옵션이 두 개 이상 포함된 경우, 대상 권장 사항의 경우 대상 옵션을 선택하세요.
4. 소스 사용률 및 용량 섹션을 확장하세요. DMS Fleet Advisor는 다음 지표에 대한 리소스 사용률 차트를 표시합니다.
 - CPU 수
 - Memory
 - I/O 처리량
 - 초당 입출력 작업량(IOPS)
 - 스토리지

- 활성 데이터베이스 서버 연결 수

이 차트를 사용하여 DMS 데이터 수집기의 소스 데이터베이스 지표를 선택한 대상 엔진의 지표와 비교할 수 있습니다.

소스 사용률 및 용량 섹션을 확장한 후 차트가 표시되지 않는 경우, IAM 사용자에게 Amazon CloudWatch 대시보드를 볼 수 있는 권한을 부여했는지 확인하세요. 자세한 내용은 Amazon CloudWatch 사용 설명서의 [Amazon CloudWatch 대시보드 사용](#)을 참조하세요.

5. 선택한 대상 엔진 이름이 있는 링크를 선택합니다. 대상 세부 정보 페이지가 열립니다.
6. 대상 권장 사항을 CSV로 내보내려면 작업 드롭다운에서 CSV로 내보내기 옵션을 선택합니다.
7. 대상 권장 사항을 내보내려면 작업 드롭다운에서 로 비용 최적화 AWS Pricing Calculator 옵션을 AWS Pricing Calculator 선택합니다.
8. 구성 섹션에서 소스 데이터베이스 파라미터의 값을 대상 엔진의 파라미터와 비교합니다. 대상 엔진의 경우, DMS Fleet Advisor는 클라우드 리소스의 월별 예상 비용을 표시합니다. DMS Fleet Advisor는 AWS 가격표 쿼리 API를 사용하여 사용자 정보에 대한 요금 세부 정보만 제공합니다. 실제 요금은 AWS 서비스의 실제 사용량을 포함한 다양한 요인에 따라 달라집니다. AWS 서비스 요금에 대한 자세한 내용은 <https://aws.amazon.com/pricing/> 클라우드 서비스 요금을 참조하세요.
9. 마이그레이션 제한 사항 섹션에서 마이그레이션 제한 사항을 확인하세요. 소스 데이터베이스를 AWS 클라우드로 마이그레이션할 때는 이러한 제한 사항을 고려하는 것이 좋습니다.

AWS DMS Fleet Advisor를 사용하여 대상 권장 사항 내보내기

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

대상 권장 사항을 생성한 후 권장 사항 목록의 사본을 CSV(쉼표로 구분된 값) 파일로 저장할 수 있습니다.

대상 권장 사항을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

DMS Fleet Advisor를 사용하는 AWS 리전 를 선택해야 합니다.

2. 탐색 창의 평가 에서 권장 사항을 선택한 다음, CSV 파일에 포함할 권장 사항을 선택합니다.
3. CSV로 내보내기를 선택하고 파일 이름을 입력한 다음, PC에서 이 파일을 저장할 폴더를 선택합니다.
4. CSV 파일을 엽니다.

권장 사항이 수록된 CSV 파일에는 다음 정보가 포함됩니다.

- CreatedDate – DMS Fleet Advisor가 대상 엔진 권장 사항을 생성한 날짜.
- Databaseld – DMS Fleet Advisor가 이 권장 사항을 생성한 소스 데이터베이스의 식별자.
- DeploymentOption - 권장되는 Amazon RDS DB 인스턴스를 위한 배포 옵션.
- EngineEdition – 권장되는 대상 Amazon RDS 엔진 에디션.
- EngineName – 대상 엔진의 이름.
- InstanceMemory - 권장되는 Amazon RDS DB 인스턴스의 메모리 크기.
- InstanceSizingType – 대상 인스턴스의 크기.
- InstanceType – 권장되는 대상 Amazon RDS 인스턴스 유형.
- InstanceVcpu – 권장되는 Amazon RDS DB 인스턴스의 가상 CPU 수.
- 선호됨 – 이 대상 옵션이 권장됨을 나타내는 부울 플래그.
- 상태 - 대상 엔진 권장 사항의 상태.
- Storagelops - 권장되는 Amazon RDS DB 인스턴스에서 완료된 초당 I/O 작업 횟수(IOPS).
- StorageSize – 권장되는 Amazon RDS DB 인스턴스의 스토리지 크기.
- StorageType – 권장되는 Amazon RDS DB 인스턴스의 스토리지 유형.
- WorkloadType – 다중 AZ 또는 단일 AZ 배포와 같은 대상 엔진의 배포 옵션.

AWS DMS Fleet Advisor를 사용하여 마이그레이션 제한 사항 검색 및 분석

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS 데이터 수집기를 사용하여 대상 엔진이 지원하지 않는 데이터베이스 기능을 검색할 수 있습니다. 올바른 마이그레이션 대상을 선택하려면 이러한 제한 사항을 고려해야 합니다.

DMS 데이터 수집기는 특정 소스 데이터베이스 기능을 검색합니다. 그런 다음 DMS Fleet Advisor는 마이그레이션 관점에서 지정된 대상으로 소스 기능을 분석하고 제한에 대한 추가 정보를 제공하며 이러한 제한을 해결하거나 피하기 위한 권장 조치를 포함합니다. 또한 DMS Fleet Advisor는 이러한 제한 사항에 따른 영향을 고려합니다.

제한 목록은 대상 엔진 세부 정보 페이지에서 확인할 수 있습니다. 왼쪽 탐색 메뉴의 권장 사항 페이지에서 이 페이지로 이동합니다. 대상 목록에서 검사할 대상 엔진을 선택합니다. 제한 목록은 페이지 하단에 있습니다.

다음 표에는 Amazon RDS for MySQL이 지원하지 않는 MySQL 데이터베이스 기능이 나와 있습니다.

제한 사항	설명	영향
인증 플러그인	Amazon RDS에서는 MySQL 인증 플러그인을 지원하지 않습니다.	낮음
시스템 로그에 오류 로깅	Amazon RDS는 시스템 로그에 오류 로그를 쓰는 것을 지원하지 않습니다.	낮음
글로벌 트랜잭션 식별자(GTID)	모든 RDS for MySQL 버전 5.7과 RDS for MySQL 버전 8.0.26 및 그 이후의 RDS for MySQL 8.0 버전에서 글로벌	낮음

제한 사항	설명	영향
	트랜잭션 식별자를 사용할 수 있습니다.	
그룹 복제	Amazon RDS는 MySQL 그룹 복제 플러그인을 지원하지 않습니다.	낮음
InnoDB 테이블스페이스 암호화	Amazon RDS에서는 InnoDB 테이블스페이스 암호화를 지원하지 않습니다.	낮음
InnoDB 예약어	InnoDB는 Amazon RDS for MySQL의 예약어입니다. MySQL 데이터베이스에는 이 이름을 사용할 수 없습니다.	낮음
키링 플러그인	Amazon RDS는 MySQL 키링 플러그인을 지원하지 않습니다.	낮음
암호 확인 플러그인	Amazon RDS는 MySQL <code>validate_password</code> 플러그인을 지원하지 않습니다.	낮음
지속 시스템 변수	Amazon RDS에서는 MySQL 지속 시스템 변수를 지원하지 않습니다.	낮음
제한된 액세스	Amazon RDS는 고급 권한이 필요한 특정 시스템 절차와 테이블에 대한 액세스를 제한합니다. 또한 Amazon RDS는 Telnet, Secure Shell(SSH) 또는 Windows 원격 데스크톱 연결을 사용하여 DB 인스턴스에 대한 직접적인 호스트 액세스를 허용하지 않습니다.	낮음

제한 사항	설명	영향
리라이터 쿼리 다시 쓰기 플러그인	Amazon RDS는 MySQL 리라이터 쿼리 다시 쓰기 플러그인을 지원하지 않습니다.	낮음
반동기식 복제	Amazon RDS는 MySQL 반동기식 복제를 지원하지 않습니다.	낮음
전송 가능 테이블스페이스	Amazon RDS는 MySQL 전송 가능 테이블스페이스 기능을 지원하지 않습니다.	낮음
X 플러그인	Amazon RDS는 MySQL X 플러그인을 지원하지 않습니다.	낮음

다음 표에는 Amazon RDS for Oracle이 지원하지 않는 Oracle 데이터베이스 기능이 나와 있습니다.

제한 사항	설명	영향
Active Data Guard	Active Data Guard는 Oracle 멀티테넌트 컨테이너 데이터베이스(CDB)와 함께 사용할 수 없습니다.	중간
Automatic Storage Management(ASM)	Amazon RDS는 Oracle Automatic Storage Management(Oracle ASM)를 지원하지 않습니다.	중간
데이터베이스 활동 스트림	Amazon RDS는 단일 테넌트 아키텍처에 대해 Oracle 데이터베이스 활동 스트림을 지원하지 않습니다.	높음

제한 사항	설명	영향
파일 크기 제한	RDS for Oracle DB 인스턴스에서 단일 파일의 최대 크기는 16TiB입니다.	중간
FTP 및 SFTP	Amazon RDS에서는 FTP와 SFTP를 지원하지 않습니다.	중간
파티셔닝된 하이브리드 테이블	Amazon RDS에서는 파티셔닝된 Oracle 하이브리드 테이블을 지원하지 않습니다.	높음
Oracle Data Guard	Amazon RDS는 단일 테넌트 아키텍처에 대한 Oracle Data Guard를 지원하지 않습니다.	높음
Oracle Database Vault	Amazon RDS에서는 Oracle Database Vault를 지원하지 않습니다.	높음
Oracle DBA 권한 저장소	Amazon RDS는 Oracle DBA 권한에 대한 제한 사항이 있습니다. 자세한 내용은 Oracle DBA 권한에 대한 제한 사항 을 참조하세요.	높음
Oracle Enterprise Manager	Amazon RDS는 단일 테넌트 아키텍처에 대한 Oracle Enterprise Manager를 지원하지 않습니다.	높음
Oracle Enterprise Manager Agent	Amazon RDS는 단일 테넌트 아키텍처에 대한 Oracle Enterprise Manager Agent를 지원하지 않습니다.	중간

제한 사항	설명	영향
Oracle Enterprise Manager Cloud Control Management Repository	Amazon RDS for Oracle DB 인스턴스는 Oracle Enterprise Manager Cloud Control Management Repository에 사용할 수 없습니다.	높음
Oracle Flashback Database	Amazon RDS는 Oracle Flashback Database 기능을 지원하지 않습니다.	높음
Oracle 레이블 보안	Amazon RDS는 단일 테넌트 아키텍처에 대한 Oracle Label Security를 지원하지 않습니다. Oracle Label Security는 멀티 테넌트 컨테이너 데이터베이스(Oracle CDB)에만 사용할 수 있습니다.	높음
Oracle Messaging Gateway	Amazon RDS에서는 Oracle Messaging Gateway를 지원하지 않습니다.	높음
Oracle Snapshot Standby 데이터베이스	Amazon RDS는 Oracle Snapshot Standby 데이터베이스를 지원하지 않습니다.	높음
공개 동의어	Amazon RDS는 Oracle에서 제한된 스키마에 대한 공개 동의를 지원하지 않습니다.	중간
지원되지 않는 기능에 대한 스키마	Amazon RDS는 시스템 권한이 필요한 Oracle 기능 및 구성 요소에 대한 스키마를 지원하지 않습니다.	높음

제한 사항	설명	영향
순수 통합 감사	Amazon RDS에서는 순수 통합 감사를 지원하지 않습니다. 통합 감사는 혼합 모드에서 사용할 수 있습니다.	중간
Workspace Manager	Amazon RDS는 Oracle Database Workspace Manager WMSYS 스키마를 지원하지 않습니다.	높음

다음 표에는 Amazon RDS for PostgreSQL이 지원하지 않는 PostgreSQL 데이터베이스 기능이 나와 있습니다.

제한 사항	설명	영향
동시 연결	RDS for PostgreSQL 인스턴스에 동시에 연결할 수 있는 최대 수는 <code>max_connections</code> 파라미터에 의해 제한됩니다.	낮음
최신 버전	Amazon RDS는 자동으로 메이저 버전 업그레이드를 적용하지 않습니다. 메이저 버전 업그레이드를 수행하려면 DB 인스턴스를 수동으로 수정합니다. 자세한 내용은 PostgreSQL에 대한 메이저 버전 업그레이드 선택 을 참조하세요.	낮음
예약된 연결	Amazon RDS는 시스템 유지 관리를 위해 최대 3개의 연결을 예약합니다. 사용자 연결 파라미터의 값을 지정할 경우, 사용	낮음

제한 사항	설명	영향
	할 것으로 예상하는 연결 개수에 3을 더합니다.	
지원되는 확장	RDS for PostgreSQL은 PostgreSQL 데이터베이스 엔진을 위한 한정된 수의 확장을 지원합니다. 해당 PostgreSQL 버전의 기본 DB 파라미터 그룹 내에서 지원되는 확장 기능의 목록을 확인할 수 있습니다. 또한 <code>rds.extensions</code> 파라미터를 표시하면 <code>psql</code> 을 사용하여 현재 확장 기능 목록을 확인할 수도 있습니다.	낮음
테이블스페이스 분할 또는 격리	I/O 분할 또는 격리를 위해 테이블스페이스를 사용할 수 없습니다. RDS for PostgreSQL에서 모든 스토리지는 단일 논리 볼륨에 있습니다.	낮음

다음 표에는 Amazon RDS for SQL Server가 지원하지 않는 SQL Server 데이터베이스 기능이 나와 있습니다.

제한 사항	설명	영향
Microsoft Azure Blob Storage로 백업	RDS for SQL Server는 Microsoft Azure Blob Storage에 백업하는 것을 지원하지 않습니다.	중간
버퍼 풀 확장	RDS for SQL Server는 버퍼 풀 확장을 지원하지 않습니다.	높음

제한 사항	설명	영향
사용자 지정 암호 정책	RDS for SQL Server는 사용자 지정 암호 정책을 지원하지 않습니다.	중간
데이터 품질 서비스	RDS for SQL Server는 SQL Server DQS(Data Quality Services)를 지원하지 않습니다.	높음
데이터베이스 로그 전달	RDS for SQL Server는 데이터베이스 로그 전달을 지원하지 않습니다.	높음
데이터베이스 이름	데이터베이스 이름에는 다음과 같은 제한이 있습니다. rdsadmin으로 시작할 수 없고, 공백이나 탭으로 시작하거나 끝날 수 없으며, 새 줄을 생성하는 문자를 포함할 수 없으며, 작은따옴표(")를 포함할 수 없습니다.	중간
데이터베이스 스냅샷	RDS for SQL Server는 데이터베이스 스냅샷을 지원하지 않습니다. Amazon RDS에서는 DB 인스턴스 스냅샷만 사용할 수 있습니다.	중간
확장된 저장 프로시저	RDS for SQL Server는 xp_cmdshell 을 포함하여 확장된 저장 프로시저를 지원하지 않습니다.	높음
파일 테이블	RDS for SQL Server는 파일 테이블을 지원하지 않습니다.	중간

제한 사항	설명	영향
FILESTREAM 지원	RDS for SQL Server는 FILESTREAM을 지원하지 않습니다.	중간
연결된 서버	RDS for SQL Server는 연결된 서버를 제한적으로 지원합니다.	높음
기계 학습 및 R 서비스	RDS for SQL Server는 기계 학습 및 R 서비스를 지원하지 않습니다. 이러한 서비스를 설치하려면 OS 액세스가 필요하기 때문입니다.	높음
유지 관리 계획	RDS for SQL Server는 유지 관리 계획을 지원하지 않습니다.	높음
성능 데이터 수집기	RDS for SQL Server는 성능 데이터 수집기를 지원하지 않습니다.	높음
정책 기반 관리	RDS for SQL Server는 정책 기반 관리를 지원하지 않습니다.	중간
PolyBase	RDS for SQL Server는 PolyBase를 지원하지 않습니다.	높음
복제	RDS for SQL Server는 복제를 지원하지 않습니다.	중간
리소스 거버너	RDS for SQL Server는 리소스 거버너를 지원하지 않습니다.	높음
서버 수준 트리거	RDS for SQL Server는 서버 수준 트리거를 지원하지 않습니다.	중간

제한 사항	설명	영향
서비스 브로커 엔드포인트	RDS for SQL Server는 서비스 브로커 엔드포인트를 지원하지 않습니다.	높음
SSAS	RDS for SQL Server에서 SQL Server Analysis Services(SSAS)를 실행하는 데 적용되는 제한 사항을 고려하세요. 자세한 내용은 제한 사항 을 참조하세요.	낮음
SSIS	RDS for SQL Server에서 SQL Server Integration Services(SSIS)를 실행하는 데 적용되는 제한 사항을 고려하세요. 자세한 내용은 제한 사항 을 참조하세요.	낮음
SSRS	RDS for SQL Server에서 SQL Server Reporting Services(SSRS)를 실행하는 데 적용되는 제한 사항을 고려하세요. 자세한 내용은 제한 사항 을 참조하세요.	낮음
SQL Server DB 인스턴스의 스토리지 크기	SQL Server 범용(SSD) 스토리지 및 프로비저닝된 IOPS 스토리지 인스턴스의 최대 스토리지 크기는 16TiB입니다. SQL Server 마그네틱 스토리지 인스턴스를 위한 최대 스토리지 크기는 1TiB입니다.	높음

제한 사항	설명	영향
Stretch 데이터베이스	RDS for SQL Server는 SQL Server Stretch 데이터베이스 기능을 지원하지 않습니다.	중간
T-SQL 엔드포인트	RDS for SQL Server는 CREATE ENDPOINT를 사용하는 모든 작업을 지원하지는 않습니다.	높음
TRUSTWORTHY 데이터베이스 속성	RDS for SQL Server는 sysadmin 역할이 필요하지 않으므로 TRUSTWORTHY 데이터베이스 속성을 지원하지 않습니다.	중간

다음 표에는 권장 사항 문제 목록이 포함되어 있습니다. DMS Fleet Advisor는 소스 및 대상 데이터베이스 기능을 분석하고 이러한 마이그레이션 제한을 제공합니다. 차단기 영향의 제한은 DMS Fleet Advisor가 소스 데이터베이스에 대한 대상 권장 사항을 생성할 수 없음을 의미합니다.

제한 사항	설명	영향
적절한 인스턴스를 찾을 수 없음	AWS DMS 는 소스 데이터베이스 지표의 조합에 적합한 크기의 마이그레이션 대상으로 작동할 수 있는 대상 인스턴스를 찾을 수 없습니다.	차단기
IOPS에서 적절한 인스턴스를 찾을 수 없음	소스 데이터베이스는 가능한 대상 DB 인스턴스의 최대 IOPS 수를 초과하는 IOPS 수를 사용합니다.	차단기
RAM에서 적절한 인스턴스를 찾을 수 없음	소스 데이터베이스는 가능한 대상 DB 인스턴스의 최대	차단기

제한 사항	설명	영향
	RAM 크기를 초과하는 GB 수의 RAM을 사용합니다.	
스토리지 크기별로 적절한 인스턴스를 찾을 수 없습니다.	소스 데이터베이스는 가능한 대상 DB 인스턴스의 최대 스토리지 크기를 초과하는 TB 수의 스토리지를 사용합니다.	차단기
에디션에서 적절한 인스턴스를 찾을 수 없음	소스 데이터베이스에는 Amazon RDS에서 지원하지 않는 에디션이 있습니다.	차단기
CPU 코어에서 적절한 인스턴스를 찾을 수 없음	소스 데이터베이스에는 가능한 대상 DB 인스턴스의 최대 CPU 코어 수를 초과하는 CPU 코어 수가 있습니다.	차단기
버전별로 적절한 인스턴스를 찾을 수 없습니다.	소스 데이터베이스에 인식 AWS DMS 할 수 없는 버전이 있습니다.	차단기
CPU 파라미터가 정의되지 않음	DMS 데이터 수집기는 소스 데이터베이스가 사용하는 CPU에 대한 정보를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 지표와 구성된 자격 증명 정보를 수집했는지 확인합니다. 데이터 전달을 위한 자격 증명 구성 을 참조하세요.	차단기

제한 사항	설명	영향
메모리 파라미터가 정의되지 않음	DMS 데이터 수집기는 소스 데이터베이스가 사용하는 메모리에 대한 정보를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 지표와 구성된 자격 증명 정보를 수집했는지 확인합니다. 데이터 전달을 위한 자격 증명 구성 를 참조하세요.	차단기
스토리지 크기 파라미터가 정의되지 않음	DMS 데이터 수집기는 소스 데이터베이스가 사용하는 스토리지 크기에 대한 정보를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 지표와 구성된 자격 증명 정보를 수집했는지 확인합니다. 데이터 전달을 위한 자격 증명 구성 를 참조하세요.	차단기
스토리지 IOPS 파라미터가 정의되지 않음	DMS 데이터 수집기가 소스 데이터베이스가 사용하는 스토리지 IOPS 지표를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 지표와 구성된 자격 증명 정보를 수집했는지 확인합니다.	차단기

제한 사항	설명	영향
데이터가 충분하지 않음	DMS 데이터 수집기가 대상 권장 사항을 생성하기에 충분한 데이터를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달을 위한 자격 증명을 구성했는지 확인합니다. 데이터 전달을 위한 자격 증명 구성 를 참조하세요.	차단기
데이터베이스 에디션이 정의되지 않았습니다.	DMS 데이터 수집기가 소스 데이터베이스 에디션에 대한 정보를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 지표와 구성된 자격 증명 정보를 수집했는지 확인합니다. 데이터 전달을 위한 자격 증명 구성 를 참조하세요.	차단기
알 수 없는 오류	DMS Fleet Advisor는 소스 데이터베이스에 대한 대상 권장 사항을 생성할 수 없습니다.	차단기
데이터베이스 버전이 정의되지 않음	DMS Fleet Advisor가 소스 데이터베이스 버전에 대한 정보를 수집하지 않았습니다. DMS Fleet Advisor는 소스 데이터베이스에 최신 데이터베이스 버전을 사용할 것을 권장합니다. 이 권장 사항을 선택하면 데이터베이스 버전을 업그레이드해야 합니다. 소스 데이터베이스에 대해 생성된 대상 권장 사항을 검토하고 이러한 권장 사항이 요구 사항을 충족하는지 확인합니다.	높음

제한 사항	설명	영향
RDS 설정에서 데이터베이스 연결 수 증가	소스 데이터베이스에는 특정 수의 연결이 필요합니다. 기본적으로 Amazon RDS 데이터베이스 인스턴스에 사용 가능한 연결 수는 다릅니다. RDS 데이터베이스 인스턴스를 생성할 때 이 기본값을 변경해야 합니다. 이렇게 하려면 파라미터 그룹의 <code>max_connections</code> 파라미터 값을 업데이트합니다.	중간
대상 에디션이 호환됨	소스 데이터베이스의 대상 권장 사항은 다른 데이터베이스 에디션을 사용합니다. 소스 데이터베이스 에디션은 권장 대상 에디션과 동일한 기능을 지원합니다. 그러나 이 새 데이터베이스 에디션을 선택하면 비용이 증가할 수 있습니다.	중간
스토리지 처리량 파라미터가 정의되지 않음	DMS 데이터 수집기가 소스 데이터베이스가 사용하는 스토리지 처리량 지표를 수집하지 않았습니다. 소스 데이터베이스에 대해 생성된 대상 권장 사항을 검토하고 이러한 권장 사항이 요구 사항을 충족하는지 확인합니다.	중간

제한 사항	설명	영향
데이터베이스 연결 번호 파라미터가 정의되지 않음	DMS 데이터 수집기는 소스 데이터베이스가 사용하는 연결 수에 대한 정보를 수집하지 않았습니다. 소스 데이터베이스에 대해 생성된 대상 권장 사항을 검토하고 이러한 권장 사항이 요구 사항을 충족하는지 확인합니다. 또는 할당량 증가를 요청합니다.	중간
데이터베이스 다운그레이드 버전	소스 데이터베이스는 Amazon RDS 데이터베이스보다 높은 버전에서 실행됩니다. 데이터베이스 버전을 다운그레이드하려면 하위 버전에서 구현되지 않은 기능을 사용하지 않아야 합니다. 또는 Amazon EC2를 마이그레이션 대상으로 사용합니다.	중간
대상 에디션이 다릅니다.	소스 데이터베이스의 대상 권장 사항은 다른 데이터베이스 에디션을 사용합니다. 소스 데이터베이스 에디션은 권장 대상 에디션과 호환됩니다. 그러나 권장 대상 데이터베이스 에디션은 소스 데이터베이스 에디션의 일부 기능을 지원하지 않습니다. 이 새 데이터베이스 에디션을 선택하면 비용이 증가할 수 있습니다.	중간

제한 사항	설명	영향
지원되지 않는 버전에서 업그레이드	소스 데이터베이스가 지원 단계 종료에 도달했습니다. 최신 DB 엔진 버전을 대상으로 사용하려면 마이그레이션 전에 데이터베이스를 업그레이드합니다. 또는 Amazon EC2를 마이그레이션 대상으로 사용합니다. 데이터베이스 엔진에 따라 자세히 알아보기에 다음 링크 중 하나를 사용합니다. MySQL 업그레이드 SQL Server 업그레이드 OracleDB 업그레이드 PostgreSQL 업그레이드	중간

AWS DMS Fleet Advisor를 사용한 대상 권장 사항 문제 해결

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

다음 목록에서 DMS Fleet Advisor 대상 권장 기능에 문제가 발생할 때 취해야 할 조치를 찾을 수 있습니다.

주제

- [대상 권장 사항의 예상 가격을 볼 수 없습니다.](#)

- [리소스 사용률 차트가 보이지 않습니다.](#)
- [지표 수집 상태를 볼 수 없습니다.](#)

대상 권장 사항의 예상 가격을 볼 수 없습니다.

성공 상태의 권장 사항에 대한 예상 월별 비용에 대한 데이터 없음이 표시되면 IAM 사용자에게 AWS 가격표 서비스 API에 액세스할 수 있는 권한을 부여했는지 확인합니다. 이렇게 하려면 `pricing:GetProducts` 권한이 포함된 정책을 생성하여 [IAM 리소스 생성](#)에 설명된 대로 IAM 사용자에게 추가해야 합니다.

DMS Fleet Advisor는 실패 상태인 권장 사항에 대한 월별 예상 비용을 계산하지 않습니다.

리소스 사용률 차트가 보이지 않습니다.

소스 사용률 및 용량 섹션을 확장한 후 지표를 로드하지 못함 메시지가 표시되는 경우, Amazon CloudWatch 대시보드를 볼 수 있는 권한을 IAM 사용자에게 부여했는지 확인하세요. 이렇게 하려면 [IAM 리소스 생성](#)에 설명된 대로 IAM 사용자에게 필수 정책을 추가해야 합니다.

또는 `cloudwatch:GetDashboard`, `cloudwatch:ListDashboards`, `cloudwatch:PutDashboard` 및 `cloudwatch:DeleteDashboards` 권한이 포함된 사용자 지정 정책을 생성할 수도 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 [Amazon CloudWatch 대시보드 사용](#)을 참조하세요.

지표 수집 상태를 볼 수 없습니다.

권장 사항 생성을 선택할 때 지표 수집에 사용 가능한 데이터 없음이라는 메시지가 표시되면 데이터를 수집했는지 확인하세요. 자세한 내용은 [AWS DMS Fleet Advisor에 대한 데이터 수집](#) 단원을 참조하십시오.

데이터를 수집한 후 이 문제가 발생하는 경우, Amazon CloudWatch에 액세스할 수 있는 `cloudwatch:Get*` 권한을 IAM 사용자에게 부여했는지 확인하세요. DMS Fleet Advisor는 서비스 연결 역할을 사용하여 수집된 데이터베이스 성능 지표를 CloudWatch에 대신 게시합니다. DMS Fleet Advisor와 함께 사용할 서비스 연결 역할을 생성해야 합니다. 자세한 내용은 [IAM 리소스 생성](#) 단원을 참조하십시오.

DMS Fleet Advisor 제한 사항 이해

Important

지원 종료 공지: 2026년 5월 20일에 AWS 는 AWS Database Migration Service Fleet Advisor 에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMS Fleet Advisor 콘솔 또는 AWS DMS Fleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 [AWS DMS Fleet Advisor 지원 종료](#)를 참조하세요.

DMS Fleet Advisor를 사용할 때의 제한 사항은 다음과 같습니다.

- DMS Fleet Advisor는 일대일 권장 사항을 생성합니다. 각 소스 데이터베이스에 대해 DMS Fleet Advisor는 단일 대상 엔진을 결정합니다. DMS Fleet Advisor는 멀티테넌트 서버를 처리하지 않으며 단일 대상 DB 인스턴스에서 여러 데이터베이스를 실행하기 위한 권장 사항도 제공하지 않습니다.
- DMS Fleet Advisor는 사용 가능한 데이터베이스 버전 업그레이드에 관한 권장 사항을 제공하지 않습니다.
- DMS Fleet Advisor는 한 번에 최대 100개의 데이터베이스에 대한 권장 사항을 생성합니다.
- Windows 애플리케이션인 DMS 데이터 수집기를 설치하는 경우 .NET Framework 4.8 및 PowerShell 6.0 이상도 설치해야 합니다. 하드웨어 요구 사항은 [데이터 수집기 설치](#)을 참조하세요.
- DMS 데이터 수집기에는 도메인 서버에서 LDAP 프로토콜을 사용하여 요청을 실행할 수 있는 권한이 필요합니다.
- DMS 데이터 수집기를 사용하려면 Linux에서 실행되는 sudo SSH 스크립트가 필요합니다.
- DMS 데이터 수집기를 사용하려면 Windows에서 원격 PowerShell, WMI(Windows Management Instrumentation), WQL(WMI Query Language) 및 레지스트리 스크립트를 실행할 수 있는 권한이 필요합니다.
- MySQL 및 PostgreSQL의 경우, DMS Fleet Advisor는 데이터베이스에서 성능 지표를 수집할 수 없습니다. 그 대신 DMS Fleet Advisor는 OS 서버 지표를 수집합니다. 따라서 Amazon RDS 및 Aurora에서 실행되는 MySQL 및 PostgreSQL 데이터베이스의 사용률 지표를 기반으로 권장 사항을 생성할 수 없습니다.

DMS Schema Conversion을 사용하여 데이터베이스 스키마 변환

Note

이제 생성형 AI 기능을 사용한 DMS 스키마 변환을 사용할 수 있습니다. 자세한 내용은 [DMS Schema Conversion용 데이터베이스 마이그레이션 평가 보고서 보기](#) 및 [DMS Schema Conversion에서 데이터베이스 스키마 변환: step-by-step 가이드](#) 섹션을 참조하세요.

DMS Schema Conversion in AWS Database Migration Service (AWS DMS)를 사용하면 다양한 유형의 데이터베이스 간에 데이터베이스를 더 쉽게 마이그레이션할 수 있습니다. DMS Schema Conversion을 사용하면 소스 데이터 공급자의 마이그레이션 복잡성을 평가하고 데이터베이스 스키마 및 코드 객체를 변환할 수 있습니다. 그런 다음 변환된 코드를 대상 데이터베이스에 적용할 수 있습니다.

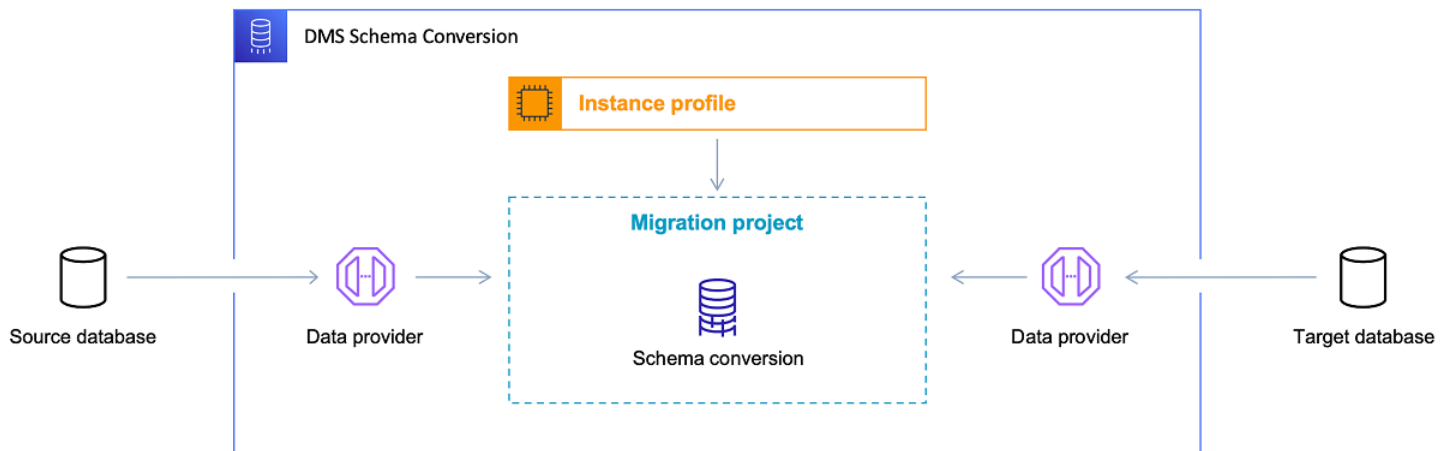
DMS Schema Conversion은 소스 데이터베이스 스키마와 대부분의 데이터베이스 코드 객체를 대상 데이터베이스와 호환되는 형식으로 자동 변환합니다. 이 변환에는 테이블, 뷰, 저장 프로시저, 함수, 데이터 형식, 동의어 등이 포함됩니다. DMS Schema Conversion에서 자동으로 변환할 수 없는 모든 객체는 명확하게 표시됩니다. 마이그레이션을 완료하기 위해 이러한 객체를 수동으로 변환할 수 있습니다.

상위 수준에서 [DMS Schema Conversion](#)은 인스턴스 프로파일, 데이터 공급자, 마이그레이션 프로젝트 등 3가지 구성 요소를 사용하여 작동합니다. 인스턴스 프로파일은 네트워크 및 보안 설정을 지정합니다. 데이터 공급자는 데이터베이스 연결 자격 증명 정보를 저장합니다. 마이그레이션 프로젝트에는 데이터 공급자, 인스턴스 프로파일 및 마이그레이션 규칙이 포함됩니다. 데이터 공급자와 인스턴스 프로파일을 AWS DMS 사용하여 데이터베이스 스키마 및 코드 객체를 변환하는 프로세스를 설계합니다.

지원되는 소스 데이터베이스의 목록은 [DMS Schema Conversion이 지원하는 소스](#) 섹션을 참조하세요.

지원되는 대상 데이터베이스의 목록은 [DMS Schema Conversion이 지원하는 대상](#) 섹션을 참조하세요.

다음 다이어그램은 DMS Schema Conversion 프로세스를 보여 줍니다.



DMS Schema Conversion을 사용하는 방법을 더 잘 이해하려면 다음 항목을 참조하세요.

주제

- [지원됨 AWS 리전](#)
- [스키마 변환 기능](#)
- [스키마 변환 제한 사항](#)
- [DMS Schema Conversion 시작하기](#)
- [DMS Schema Conversion을 위한 네트워크 설정](#)
- [DMS Schema Conversion에서 소스 데이터 공급자 생성](#)
- [DMS Schema Conversion에서 대상 데이터 공급자 생성 및 설정](#)
- [가상 데이터 공급자](#)
- [DMS Schema Conversion의 마이그레이션 프로젝트 관리](#)
- [DMS Schema Conversion을 사용하여 데이터베이스 마이그레이션 평가 보고서 생성](#)
- [DMS Schema Conversion 사용](#)
- [DMS Schema Conversion에서 확장 팩 사용](#)
- [AWS DMS Schema Conversion 및 Common Studio Framework\(CSF\)를 위한 API 매핑에 대한 IAM 작업](#)

지원됨 AWS 리전

다음에서 DMS Schema Conversion 마이그레이션 프로젝트를 생성할 수 있습니다 AWS 리전. 다른 리전에서는 AWS Schema Conversion Tool를 사용할 수 있습니다. 에 대한 자세한 내용은 [AWS Schema Conversion Tool 사용 설명서](#)를 AWS SCT참조하세요.

리전 이름	리전
Africa (Cape Town)	af-south-1
아시아 태평양(홍콩)	ap-east-1
아시아 태평양(뭄바이)	ap-south-1
아시아 태평양(하이데라바드)	ap-south-2
아시아 태평양(도쿄)	ap-northeast-1
아시아 태평양(서울)	ap-northeast-2
아시아 태평양(싱가포르)	ap-southeast-1
아시아 태평양(시드니)	ap-southeast-2
아시아 태평양(자카르타)	ap-southeast-3
아시아 태평양(멜버른)	ap-southeast-4
캐나다(중부)	ca-central-1
캐나다 서부(캘거리)	ca-west-1
유럽(프랑크푸르트)	eu-central-1
유럽(취리히)	eu-central-2
유럽(스톡홀름)	eu-north-1
Europe (Milan)	eu-south-1
유럽(스페인)	eu-south-2
유럽(아일랜드)	eu-west-1
유럽(파리)	eu-west-3
이스라엘(텔아비브)	il-central-1

리전 이름	리전
중동(UAE)	me-central-1
남아메리카(상파울루)	sa-east-1
미국 동부(버지니아 북부)	us-east-1
미국 동부(오하이오)	us-east-2
미국 서부(캘리포니아 북부)	us-west-1
미국 서부(오리건)	us-west-2

스키마 변환 기능

DMS Schema Conversion은 다음과 같은 기능을 제공합니다.

- DMS Schema Conversion은 데이터베이스 마이그레이션 프로젝트에 필요한 AWS 클라우드 리소스를 자동으로 관리합니다. 이러한 리소스에는 인스턴스 프로파일, 데이터 공급자 및 AWS Secrets Manager 보안 암호가 포함됩니다. 여기에는 AWS Identity and Access Management (IAM) 역할, Amazon S3 버킷 및 마이그레이션 프로젝트도 포함됩니다.
- DMS Schema Conversion을 사용하여 소스 데이터베이스에 연결하고, 메타데이터를 읽고, 데이터베이스 마이그레이션 평가 보고서를 생성할 수 있습니다. 그런 다음 Amazon S3 버킷에 보고서를 저장할 수 있습니다. 이 보고서를 통해 스키마 변환 작업의 요약 및 DMS Schema Conversion이 대상 데이터베이스로 자동 변환할 수 없는 항목에 대한 세부 정보를 얻을 수 있습니다. 데이터베이스 마이그레이션 평가 보고서는 DMS Schema Conversion이 마이그레이션 프로젝트를 얼마나 자동화할 수 있는지 평가하는 데 도움이 됩니다. 이러한 보고서는 변환을 완료하는 데 필요한 수작업량을 추정하는 데에도 도움이 됩니다. 자세한 내용은 [DMS Schema Conversion을 사용하여 데이터베이스 마이그레이션 평가 보고서 생성](#) 단원을 참조하십시오.
- 소스 및 대상 데이터 공급자에 연결하면 DMS Schema Conversion이 기존 소스 데이터베이스 스키마를 대상 데이터베이스 엔진으로 변환할 수 있습니다. 소스 데이터베이스에서 변환할 스키마 항목을 선택할 수 있습니다. DMS Schema Conversion에서 데이터베이스 코드를 변환한 후 소스 코드와 변환된 코드를 검토할 수 있습니다. 또한 변환된 SQL 코드를 Amazon S3 버킷에 저장할 수 있습니다.

- 소스 데이터베이스 스키마를 변환하기 전에 변환 규칙을 설정할 수 있습니다. 변환 규칙을 사용하여 열의 데이터 형식을 변경하고, 한 스키마에서 다른 스키마로 객체를 이동하고, 객체 이름을 변경할 수 있습니다. 변환 규칙을 데이터베이스, 스키마, 테이블 및 열에 적용할 수 있습니다. 자세한 내용은 [변환 규칙 설정](#) 단원을 참조하십시오.
- 변환 설정을 변경하여 변환된 코드의 성능을 향상시킬 수 있습니다. 이러한 설정은 각 변환 쌍에 따라 다르고 코드에서 사용하는 소스 데이터베이스의 기능에 따라 달라집니다. 자세한 내용은 [스키마 변환 설정 지정](#) 단원을 참조하십시오.
- 경우에 따라 DMS Schema Conversion이 소스 데이터베이스 기능을 동등한 Amazon RDS 기능으로 변환할 수 없습니다. 이러한 경우 DMS Schema Conversion은 대상 데이터베이스에 확장 팩을 생성하여 변환되지 않은 기능을 에뮬레이션합니다. 자세한 내용은 [확장 팩 사용](#) 단원을 참조하십시오.
- 변환된 코드와 확장 팩 스키마를 대상 데이터베이스에 적용할 수 있습니다. 자세한 내용은 [변환된 코드 적용](#) 단원을 참조하십시오.
- DMS Schema Conversion은 최신 AWS SCT 릴리스의 모든 기능을 지원합니다. 자세한 내용은 [AWS SCT에 대한 최신 릴리스 정보를](#) 참조하십시오.
- 변환된 SQL 코드는 DMS가 대상 데이터베이스로 마이그레이션하기 전에 편집할 수 있습니다. 자세한 내용은 [변환된 SQL 코드 편집 및 저장](#) 단원을 참조하십시오.

스키마 변환 제한 사항

DMS Schema Conversion은 AWS Schema Conversion Tool ()의 웹 버전입니다. AWS SCT. DMS 스키마 변환은 AWS SCT 데스크톱 애플리케이션에 비해 지원하는 데이터베이스 플랫폼 수가 적고 기능이 제한적입니다. 데이터 웨어하우스 스키마, 빅 데이터 프레임워크, 애플리케이션 SQL 코드 및 ETL 프로세스를 변환하려면 AWS SCT를 사용하십시오. 에 대한 자세한 내용은 [AWS Schema Conversion Tool 사용 설명서](#)를 AWS SCT참조하십시오.

데이터베이스 스키마 변환에 DMS Schema Conversion을 사용할 때는 다음과 같은 제한 사항이 적용됩니다.

- 마이그레이션 프로젝트를 저장한 후 오프라인 모드에서 사용할 수 없습니다.
- DMS Schema Conversion의 마이그레이션 프로젝트에서 소스의 SQL 코드를 편집할 수 없습니다. 소스 데이터베이스의 SQL 코드를 편집하려면 일반 SQL 편집기를 사용하십시오. 데이터베이스에서 새로 고침을 선택하여 마이그레이션 프로젝트에 업데이트된 코드를 추가합니다.
- DMS Schema Conversion의 마이그레이션 규칙은 열 데이터 정렬 변경을 지원하지 않습니다. 마이그레이션 규칙을 사용하여 객체를 새 스키마로 이동할 수 없습니다.

- 소스 및 대상 데이터베이스 트리에 필터를 적용하여 필터 절을 충족하는 데이터베이스 객체만 표시할 수는 없습니다.
- DMS Schema Conversion 확장 팩에는 변환된 코드의 이메일 전송, 작업 예약 및 기타 기능을 예물레이션하는 AWS Lambda 함수가 포함되지 않습니다.
- DMS Schema Conversion은 고객 AWS 리소스에 액세스하는 데 고객 관리형 KMS 키를 사용하지 않습니다. 예를 들어, DMS Schema Conversion은 고객 관리형 KMS 키를 사용하여 Amazon S3의 고객 데이터에 액세스하는 것을 지원하지 않습니다.

DMS Schema Conversion 시작하기

DMS Schema Conversion을 시작하려면 다음 자습서를 사용하세요. 자습서에서는 DMS Schema Conversion을 설정하고 마이그레이션 프로젝트를 생성하며 데이터 공급자와 연결하는 방법을 배울 수 있습니다. 그런 다음, 마이그레이션의 복잡성을 평가하고 소스 데이터베이스를 대상 데이터베이스와 호환되는 형식으로 변환하는 방법을 배울 수 있습니다. 또한 변환된 코드를 대상 데이터베이스에 적용하는 방법도 배울 수 있습니다.

다음 자습서에서는 필수 작업에 관한 내용을 다루며 Amazon RDS for SQL Server 데이터베이스를 Amazon RDS for MySQL로 변환하는 방법을 보여줍니다. 지원되는 소스 및 대상 데이터 공급자를 모두 사용할 수 있습니다. 자세한 내용은 [DMS Schema Conversion이 지원하는 소스 데이터 공급자](#) 단원을 참조하십시오.

DMS 스키마 변환에 관한 자세한 내용은 [Oracle에서 PostgreSQL로](#), [SQL Server에서 MySQL로](#)의 마이그레이션에 대한 단계별 마이그레이션 안내를 참조하세요.

[이 동영상](#)은 DMS Schema Conversion 사용자 인터페이스를 소개하고 이 서비스의 핵심 구성 요소를 숙지하는 데 도움이 됩니다.

주제

- [DMS Schema Conversion 작업을 위한 전체 사전 조건](#)
- [DMS Schema Conversion을 위한 인스턴스 프로파일 생성](#)
- [DMS Schema Conversion을 위한 데이터 공급자 구성](#)
- [DMS Schema Conversion에서 마이그레이션 프로젝트 생성](#)
- [DMS Schema Conversion에서 평가 보고서 생성](#)
- [DMS Schema Conversion에서 소스 코드 변환](#)
- [DMS Schema Conversion에 변환된 코드 적용](#)

- [DMS Schema Conversion 작업을 위한 정리 및 문제 해결](#)

DMS Schema Conversion 작업을 위한 전체 사전 조건

DMS Schema Conversion을 설정하려면 다음 작업을 완료하세요. 그런 다음, 인스턴스 프로파일을 설정하고 데이터 공급자를 추가하며 마이그레이션 프로젝트를 생성할 수 있습니다.

주제

- [Amazon VPC를 기반으로 VPC 생성](#)
- [Amazon S3 버킷 생성](#)
- [예 데이터베이스 자격 증명 저장 AWS Secrets Manager](#)
- [IAM 정책 생성](#)
- [IAM 역할 생성](#)

Amazon VPC를 기반으로 VPC 생성

이 단계에서는에서 Virtual Private Cloud(VPC)를 생성합니다 AWS 계정. 이 VPC는 Amazon Virtual Private Cloud(VPC) 서비스를 기반으로 하며 AWS 리소스를 포함합니다.

DMS Schema Conversion을 위한 VPC를 생성하려면

1. 예 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/vpc/> Amazon VPC 콘솔을 엽니다.
2. VPC 생성을 선택합니다.
3. VPC 생성 페이지에서 다음 설정을 입력합니다.
 - 생성할 리소스 – VPC 등
 - 이름 태그 자동 생성 – 자동 생성을 선택하고 글로벌 고유 이름을 입력합니다. 예를 들면 **sc-vpc**를 입력합니다.
 - IPv4 CIDR 블록 - **10.0.1.0/24**
 - NAT 게이트웨이 – In 1 AZ
 - VPC 엔드포인트 – 없음
4. 해당 설정의 나머지 부분은 그대로 유지하고 VPC 만들기를 선택합니다.
5. 서브넷을 선택하고 퍼블릭 및 프라이빗 서브넷 ID를 기록해 둡니다.

Amazon RDS 데이터베이스에 연결하려면 퍼블릭 서브넷이 포함된 서브넷 그룹을 생성하세요.

온프레미스 데이터베이스에 연결하려면 프라이빗 서브넷이 포함된 서브넷 그룹을 생성하세요. 자세한 내용은 [DMS Schema Conversion을 위한 인스턴스 프로파일 생성](#) 단원을 참조하십시오.

6. NAT 게이트웨이를 선택합니다. NAT 게이트웨이를 선택하고 탄력적 IP 주소를 기록해 둡니다.

가이 NAT 게이트웨이의 퍼블릭 IP 주소에서 소스 온프레미스 데이터베이스에 액세스할 AWS DMS 수 있도록 네트워크를 구성합니다. 자세한 내용은 [VPC에 인터넷 연결 사용](#) 단원을 참조하십시오.

Amazon RDS에서 인스턴스 프로파일과 대상 데이터베이스를 생성할 때 이 VPC를 사용하세요.

Amazon S3 버킷 생성

마이그레이션 프로젝트의 정보를 저장하려면 Amazon S3 버킷을 생성하세요. DMS Schema Conversion은 이 Amazon S3 버킷을 사용하여 평가 보고서, 변환된 SQL 코드, 데이터베이스 스키마 객체에 관한 정보 등의 항목을 저장합니다.

DMS Schema Conversion에 대한 Amazon S3 버킷을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/s3/> Amazon S3 콘솔을 엽니다.
2. 버킷 생성을 선택합니다.
3. 버킷 만들기 페이지에서 S3 버킷의 글로벌 고유 이름을 선택합니다. 예를 들면 **sc-s3-bucket**을 입력합니다.
4. AWS 리전의 경우, 리전을 선택합니다.
5. 버킷 버전 관리의 경우, 활성화를 선택합니다.
6. 해당 설정의 나머지 부분은 그대로 유지하고 버킷 만들기를 선택합니다.

Note

DMS 스키마 변환(SC)은 Amazon S3 관리 키로 서버측 암호화(SSE-S3)를 사용하는 S3 버킷에서만 동작합니다. AWS KMS를 사용한 서버 측 암호화(SSE-KMS), 고객 제공 키(SSE-C) 또는 클라이언트 측 암호화를 포함한 다른 암호화 구성은 현재 SC에서 지원되지 않으므로 S3 버

킷에 액세스할 수 없습니다. 다른 암호화 방법을 사용하는 경우 DMS 스키마 변환과 함께 사용할 수 있도록 SSE-S3 버킷을 생성해야 합니다.

에 데이터베이스 자격 증명 저장 AWS Secrets Manager

소스 및 대상 데이터베이스 자격 증명에 저장합니다 AWS Secrets Manager. 이러한 보안 암호를 복제해야 합니다 AWS 리전. DMS Schema Conversion은 이러한 보안 암호를 사용하여 마이그레이션 프로젝트의 데이터베이스에 연결합니다.

에 데이터베이스 자격 증명을 저장하려면 AWS Secrets Manager

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/secretsmanager/> AWS Secrets Manager 콘솔을 엽니다.
2. 새 보안 암호 저장을 선택합니다.
3. 보안 암호 유형 선택 페이지가 열립니다. 보안 암호 유형에서 저장할 데이터베이스 자격 증명 정보의 유형을 선택합니다.
 - Amazon RDS 데이터베이스의 자격 증명 정보 – Amazon RDS 데이터베이스의 자격 증명을 저장하려면 이 옵션을 선택합니다. 자격 증명 정보에서 데이터베이스에 대한 자격 증명 정보를 입력합니다. 데이터베이스에서 데이터베이스를 선택합니다.
 - 기타 데이터베이스의 자격 증명 정보 – Oracle 또는 SQL Server 소스 데이터베이스의 자격 증명 정보를 저장하려면 이 옵션을 선택합니다. 자격 증명 정보에서 데이터베이스에 대한 자격 증명 정보를 입력합니다.
 - 기타 유형의 보안 암호 - 데이터베이스에 연결하는 데 필요한 사용자 이름과 암호만 저장하려면 이 옵션을 선택합니다. 행 추가를 선택하여 두 개의 키-값 쌍을 추가합니다. 키 이름에는 반드시 **username**과 **password**를 사용해야 합니다. 이러한 키와 관련된 값에는 데이터베이스의 자격 증명 정보를 입력합니다.
4. 암호화 키에서 Secrets Manager가 보안 암호 값을 암호화하는 데 사용하는 AWS KMS 키를 선택합니다. 다음을 선택합니다.
5. 보안 암호 구성 페이지에서 설명이 포함된 보안 암호 이름을 입력합니다. 예를 들면, **sc-source-secret** 또는 **sc-target-secret**을 입력합니다.
6. 보안 암호 복제를 선택한 다음, AWS 리전에서 리전을 선택합니다. 다음을 선택합니다.
7. 로테이션 구성 페이지에서 다음을 선택합니다.
8. 검토 페이지에서 보안 암호 세부 정보를 검토한 후 저장을 선택합니다.

소스 및 대상 데이터베이스의 자격 증명 정보를 저장하려면 이 단계를 반복하세요.

IAM 정책 생성

DMS 스키마 변환이 Amazon S3에 접근할 수 있도록 IAM 정책을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택합니다.
3. 정책 생성을 선택합니다.
4. 서비스 선택 페이지의 목록에서 Amazon S3를 선택합니다.
5. 허용된 작업에서 PutObject, GetObject, GetObjectVersion, GetBucketVersioning, GetBucketLocation, ListBucket을 선택합니다.
6. 리소스에서 이전 섹션에서 생성한 버킷의 ARN을 지정합니다. 다음을 선택합니다.
7. 검토 및 생성 페이지에서 설명이 포함된 이름을 입력합니다. 예를 들어 sc-s3-policy입니다. 그런 다음 정책 생성을 선택합니다.

DMS Schema Conversionto 액세스를 위한 IAM 정책을 생성하려면 AWS Secrets Manager

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택합니다.
3. 정책 생성을 선택합니다.
4. 서비스 선택 페이지의 목록에서 Secrets Manager를 선택합니다.
5. 다음을 선택합니다. 권한 추가 페이지가 열립니다.
6. 허용된 작업에서 GetSecretValue 및 DescribeSecret을 선택합니다.
7. 검토 및 생성 페이지에서 설명이 포함된 이름을 입력합니다. 예를 들어 sc-secrets-manager-policy입니다. 그런 다음 정책 생성을 선택합니다.

IAM 역할 생성

마이그레이션 프로젝트에 사용할 AWS Identity and Access Management (IAM) 역할을 생성합니다. DMS Schema Conversion은 이러한 IAM 역할을 사용하여 AWS Secrets Manager에 저장된 Amazon S3 버킷 및 데이터베이스 자격 증명 정보에 액세스합니다.

Amazon S3 버킷에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Roles를 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지에서 AWS 서비스를 선택합니다. DMS를 선택합니다.
5. 다음을 선택합니다. 권한 추가 페이지가 열립니다.
6. 필터 정책의 경우 **S3**을 입력합니다. 이전 섹션에서 생성한 sc-s3-policy 정책을 선택합니다.
7. 다음을 선택합니다. 이름, 검토 및 생성 페이지가 열립니다.
8. 역할 이름에 설명이 포함된 이름을 입력합니다. 예를 들면 **sc-s3-role**을 입력합니다. 역할 생성을 선택합니다.
9. 역할 페이지에서 역할 이름에 **sc-s3-role**을 입력합니다. sc-s3-role을 선택합니다.
10. sc-s3-role 페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.
11. AWS DMS 리전 서비스 보안 주체의 형식은 다음과 같습니다.

```
dms.region-name.amazonaws.com
```

region-name을 us-east-1와 같은 리전 이름으로 바꿉니다. 다음 코드 예제에서는 us-east-1 리전의 위탁자를 보여줍니다.

```
dms.us-east-1.amazonaws.com
```

다음 코드 예제에서는 AWS DMS 스키마 변환에 액세스하기 위한 신뢰 정책을 보여줍니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.us-east-1.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

```
}
]
}
```

에 대한 액세스를 제공하는 IAM 역할을 생성하려면 AWS Secrets Manager

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Roles를 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지에서 AWS 서비스를 선택합니다. DMS를 선택합니다.
5. 다음을 선택합니다. 권한 추가 페이지가 열립니다.
6. 필터 정책의 경우 s3을 입력합니다. 이전 섹션에서 생성한 sc-secrets-manager-policy를 선택합니다.
7. 다음을 선택합니다. 이름, 검토 및 생성 페이지가 열립니다.
8. 역할 이름에 설명이 포함된 이름을 입력합니다. 예를 들면 **sc-secrets-manager-role**을 입력합니다. 역할 생성을 선택합니다.
9. 역할 페이지에서 역할 이름에 **sc-secrets-manager-role**을 입력합니다. sc-secrets-manager-role을 선택합니다.
10. sc-secrets-manager-role 페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.
11. 신뢰 정책 편집 페이지에서 및 AWS DMS 리전 서비스 보안 주체를 신뢰할 수 있는 엔터티로 사용할 역할의 신뢰 관계를 편집 `schema-conversion.dms.amazonaws.com`합니다. 이 AWS DMS 리전 서비스 보안 주체의 형식은 다음과 같습니다.

```
dms.region-name.amazonaws.com
```

region-name을 us-east-1와 같은 리전 이름으로 바꿉니다. 다음 코드 예제에서는 us-east-1 리전의 위탁자를 보여줍니다.

```
dms.us-east-1.amazonaws.com
```

다음 코드 예제에서는 AWS DMS 스키마 변환에 액세스하기 위한 신뢰 정책을 보여줍니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.us-east-1.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

AWS CLI 또는 AWS DMS API와 함께 사용할 **dms-vpc-role** IAM 역할을 생성하려면

1. 다음과 같은 IAM 정책이 있는 JSON 파일을 생성합니다. JSON 파일의 이름을 `dmsAssumeRolePolicyDocument.json`으로 지정합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

그런 다음 다음 명령을 사용하여 AWS CLI를 사용하여 역할을 생성합니다.

```
aws iam create-role --role-name dms-vpc-role --assume-role-policy-document file://dmsAssumeRolePolicyDocument.json
```

2. 다음 명령을 사용하여 AmazonDMSVPCManagementRole 정책을 dms-vpc-role에 연결합니다.

```
aws iam attach-role-policy --role-name dms-vpc-role --policy-arn arn:aws:iam::aws:policy/service-role/AmazonDMSVPCManagementRole
```

AWS CLI 또는 AWS DMS API와 함께 사용할 **dms-cloudwatch-logs-role** IAM 역할을 생성하려면

1. 다음과 같은 IAM 정책이 있는 JSON 파일을 생성합니다. JSON 파일의 이름을 dmsAssumeRolePolicyDocument2.json으로 지정합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

그런 다음 다음 명령을 사용하여 AWS CLI를 사용하여 역할을 생성합니다.

```
aws iam create-role --role-name dms-cloudwatch-logs-role --assume-role-policy-document file://dmsAssumeRolePolicyDocument2.json
```

2. 다음 명령을 사용하여 AmazonDMSCloudWatchLogsRole 정책을 dms-cloudwatch-logs-role에 연결합니다.


```
aws iam attach-role-policy --role-name dms-cloudwatch-logs-role --policy-arn
arn:aws:iam::aws:policy/service-role/AmazonDMSCloudWatchLogsRole
```

DMS Schema Conversion을 위한 인스턴스 프로파일 생성

인스턴스 프로파일을 생성하기 전에 인스턴스 프로파일의 서브넷 그룹을 구성하세요. AWS DMS 마이그레이션 프로젝트의 서브넷 그룹 생성에 대한 자세한 내용은 [섹션을 참조하세요](#) [서브넷 그룹 생성](#).

다음 절차의 설명에 따라 인스턴스 프로파일을 생성할 수 있습니다. 이 인스턴스 프로파일에서는 DMS Schema Conversion 프로젝트의 네트워크 및 보안 설정을 지정합니다.

인스턴스 프로파일 생성

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 인스턴스 프로파일을 선택한 다음, 인스턴스 프로파일 생성을 선택합니다.
3. 이름에 인스턴스 프로파일의 고유한 이름을 입력합니다. 예를 들면 **sc-instance**를 입력합니다.
4. 네트워크 유형에서 IPv4를 선택하여 IPv4 주소 지정만 지원하는 인스턴스 프로파일을 생성합니다. IPv4 및 IPv6 주소 지정을 지원하는 인스턴스 프로파일을 생성하려면 듀얼 스택 모드를 선택합니다.
5. 가상 프라이빗 클라우드(VPC)의 경우, 사전 조건 단계에서 생성한 VPC를 선택합니다.
6. 서브넷 그룹의 경우, 인스턴스 프로파일에 대한 서브넷 그룹을 선택합니다. Amazon RDS 데이터베이스에 연결하려면 퍼블릭 서브넷이 포함된 서브넷 그룹을 생성하세요. 온프레미스 데이터베이스에 연결하려면 프라이빗 서브넷이 포함된 서브넷 그룹을 생성하세요.
7. 인스턴스 프로파일 생성을 선택합니다.

마이그레이션 프로젝트를 생성하려면 이 인스턴스 프로파일을 사용하세요.

DMS Schema Conversion을 위한 데이터 공급자 구성

다음으로 소스 및 대상 데이터베이스를 설명하는 데이터 공급자를 생성합니다. 각 데이터 공급자에 대해 데이터 저장소 유형과 위치 정보를 지정합니다. 데이터베이스 자격 증명 정보는 데이터 공급자에 저장하지 않습니다.

온프레미스 데이터베이스의 데이터 공급자를 생성하려면

1. AWS Management Console에 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 데이터 공급자를 선택한 다음, 데이터 공급자 생성을 선택합니다.
3. 이름에 소스 데이터 공급자의 고유한 이름을 입력합니다. 예를 들면 sc-dp를 입력합니다.
4. 용도에서 스키마 변환을 선택합니다.
5. 엔진 유형에서 데이터 공급자의 데이터베이스 엔진 유형을 선택합니다.
6. 대상 데이터베이스에 연결하지 않고 스키마 변환을 사용하려면 가상 모드를 켭니다. 자세한 내용은 [가상 데이터 공급자](#) 단원을 참조하십시오.
7. 가상 모드가 켜져 있으면 기본값을 사용하여 연결 정보가 자동으로 사전 설정됩니다. 가상 데이터 공급자는 마이그레이션 프로젝트 또는 스키마 변환에서만 대상으로 사용할 수 있습니다. 가상화 모드에 대한 자세한 내용은 [가상 데이터 공급자](#) 섹션을 참조하세요.

가상 모드가 꺼져 있는 경우 소스 데이터베이스에 대한 연결 정보를 제공합니다. 연결 파라미터는 데이터베이스 엔진에 따라 달라집니다. 자세한 내용은 [데이터 공급자 생성](#) 단원을 참조하십시오.

8. 엔진 구성을 선택합니다. 그런 다음 수동으로 입력을 선택합니다. 소스 데이터베이스 연결 정보를 제공합니다. 연결 파라미터는 소스 데이터베이스 엔진에 따라 달라집니다. 자세한 내용은 [데이터 공급자 생성](#) 단원을 참조하십시오.
9. 보안 소켓 계층(SSL) 모드의 경우, SSL 적용 유형을 선택합니다.
10. 데이터 공급자 생성을 선택합니다.

Amazon RDS 데이터베이스를 위한 데이터 공급자를 생성하려면

1. AWS Management Console에 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 데이터 공급자를 선택한 다음, 데이터 공급자 생성을 선택합니다.
3. 이름에 소스 데이터 공급자의 고유한 이름을 입력합니다. 예를 들면 sc-dp를 입력합니다.
4. 용도에서 스키마 변환을 선택합니다.
5. 엔진 유형에서 데이터 공급자의 데이터베이스 엔진 유형을 선택합니다.
6. 대상 데이터베이스에 연결하지 않고 스키마 변환에 이 데이터 공급자를 사용하려는 경우 가상화 모드를 켭니다. 가상화 모드에 대한 자세한 내용은 [가상 데이터 공급자](#) 섹션을 참조하세요.
7. 엔진 구성을 선택합니다. 목록에서 RDS 데이터베이스 인스턴스 선택을 선택합니다.
8. RDS의 데이터베이스에서 찾아보기를 선택하고 데이터베이스를 선택합니다. DMS Schema Conversion은 엔진 유형, 서버 이름 및 포트에 관한 정보를 자동으로 검색합니다.
9. 데이터베이스 이름에 데이터베이스 이름을 입력합니다.

10. 보안 소켓 계층(SSL) 모드의 경우, SSL 적용 유형을 선택합니다.
11. 데이터 공급자 생성을 선택합니다.

DMS Schema Conversion에서 마이그레이션 프로젝트 생성

이제 마이그레이션 프로젝트를 생성할 수 있습니다. 마이그레이션 프로젝트에서 소스 및 대상 데이터 공급자와 인스턴스 프로파일을 지정합니다.

마이그레이션 프로젝트를 생성하려면

1. 마이그레이션 프로젝트를 선택한 다음, 마이그레이션 프로젝트 생성을 선택합니다.
2. 이름에 마이그레이션 프로젝트의 고유한 이름을 입력합니다. 예를 들면 **sc-project**를 입력합니다.
3. 인스턴스 프로파일에서 **sc-instance**를 선택합니다.
4. 소스에서 찾아보기를 선택한 다음, **sc-source**를 선택합니다.
5. 보안 암호 ID에서 **sc-source-secret**을 선택합니다.
6. IAM 역할에서 **sc-secrets-manager-role**을 선택합니다.
7. 대상에서 찾아보기를 선택한 다음, **sc-target**을 선택합니다.
8. 보안 암호 ID에서 **sc-target-secret**을 선택합니다.
9. IAM 역할에서 **sc-secrets-manager-role**을 선택합니다.
10. 마이그레이션 프로젝트 생성을 선택합니다.

DMS Schema Conversion에서 평가 보고서 생성

마이그레이션의 복잡성을 평가하려면 데이터베이스 마이그레이션 평가 보고서를 생성하세요. 이 보고서에는 DMS Schema Conversion이 자동으로 변환할 수 없는 모든 데이터베이스 객체의 목록이 포함됩니다.

평가 보고서를 생성하려면

1. 마이그레이션 프로젝트를 선택한 다음, **sc-project**를 선택하세요.
2. 스키마 변환을 선택한 다음, 스키마 변환 시작을 선택합니다.
3. 소스 데이터베이스 창에서 평가할 데이터베이스 스키마를 선택합니다. 또한 이 스키마의 이름에 대한 확인란을 선택합니다.

4. 소스 데이터베이스 창의 작업 메뉴에서 평가를 선택합니다. 평가 대화 상자가 나타납니다.
5. 대화 상자에서 평가를 선택하여 선택을 확인합니다.

요약 탭에는 DMS Schema Conversion이 데이터베이스 저장소 객체 및 데이터베이스 코드 객체에 대해 자동으로 변환할 수 있는 항목의 수가 표시됩니다.

6. 작업 항목을 선택하면 DMS Schema Conversion이 자동으로 변환할 수 없는 모든 데이터베이스 객체의 목록을 볼 수 있습니다. 각 항목에 대한 권장 조치를 검토하세요.
7. 평가 보고서 사본을 저장하려면 결과 내보내기를 선택합니다. 그런 다음, CSV 또는 PDF 형식 중 하나를 선택합니다. 내보내기 대화 상자가 나타납니다.
8. 내보내기를 선택하여 선택을 확인합니다.
9. S3 버킷을 선택합니다. Amazon S3 콘솔이 열립니다.
10. 다운로드를 선택하여 평가 보고서를 저장합니다.

DMS Schema Conversion에서 소스 코드 변환

다음 절차에 따라 소스 데이터베이스 스키마를 변환할 수 있습니다. 그런 다음, 변환된 코드를 텍스트 파일에 SQL 스크립트로 저장할 수 있습니다.

데이터베이스 스키마를 변환하려면

1. 소스 데이터베이스 창에서 변환할 데이터베이스 스키마를 선택합니다. 또한 이 스키마의 이름에 대한 확인란을 선택합니다.
2. 소스 데이터베이스 창의 작업 메뉴에서 변환을 선택합니다. 변환 대화 상자가 나타납니다.
3. 대화 상자에서 변환을 선택하여 선택을 확인합니다.
4. 소스 데이터베이스 창에서 데이터베이스 객체를 선택합니다. DMS Schema Conversion은 이 객체의 소스 코드와 변환된 코드를 표시합니다. SQL 편집 기능을 사용하여 데이터베이스 객체에 대해 변환된 SQL 코드를 편집할 수 있습니다. 자세한 내용은 [변환된 SQL 코드 편집 및 저장](#) 단원을 참조하십시오.
5. 대상 데이터베이스 창에서 변환된 데이터베이스 스키마를 선택합니다. 또한 이 스키마의 이름에 대한 확인란을 선택합니다.
6. 작업에서 SQL로 저장을 선택합니다. 저장 대화 상자가 나타납니다.
7. SQL로 저장을 선택해 선택을 확인합니다.
8. S3 버킷을 선택합니다. Amazon S3 콘솔이 열립니다.
9. 다운로드를 선택하여 SQL 스크립트를 저장합니다.

DMS Schema Conversion에 변환된 코드 적용

DMS Schema Conversion은 변환된 코드를 대상 데이터베이스에 즉시 적용하지 않습니다. 이전 단계에서 생성한 SQL 스크립트를 사용하면 대상 데이터베이스를 업데이트할 수 있습니다. 또는 다음 절차를 사용하여 DMS Schema Conversion에서 변환된 코드를 적용할 수 있습니다.

변환된 코드를 적용하려면

1. 대상 데이터베이스 창에서 변환된 데이터베이스 스키마를 선택합니다. 또한 이 스키마의 이름에 대한 확인란을 선택합니다.
2. 작업에서 변경 사항 적용을 선택합니다. 변경 사항 적용 대화 상자가 나타납니다.
3. 적용을 선택하여 선택을 확인합니다.

DMS Schema Conversion 작업을 위한 정리 및 문제 해결

Amazon CloudWatch를 사용하여 DMS Schema Conversion 로그를 검토하거나 공유할 수 있습니다.

DMS Schema Conversion 로그를 검토하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/cloudwatch/> CloudWatch 콘솔을 엽니다.
2. 로그, 로그 그룹을 선택합니다.

DMS Schema Conversion 로그 그룹의 이름은 `dms-tasks-sct`로 시작합니다. 생성 시간별로 로그 그룹을 정렬하여 DMS Schema Conversion 로그 그룹을 찾을 수 있습니다.

또한 로그 그룹의 이름에는 마이그레이션 프로젝트의 Amazon 리소스 이름(ARN)이 포함됩니다. 프로젝트의 ARN은 DMS Schema Conversion의 마이그레이션 프로젝트 페이지에서 확인할 수 있습니다. 기본 설정에서 ARN을 선택했는지 확인하세요.

3. 로그 그룹의 이름을 선택한 다음, 로그 스트림의 이름을 선택합니다.
4. 작업에서 결과 내보내기를 선택하여 DMS Schema Conversion 로그를 저장합니다.

DMS Schema Conversion에서 스키마 변환을 완료한 후에는 리소스를 정리합니다.

DMS Schema Conversion 리소스를 정리하려면

1. 에 로그인 AWS Management Console 하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 마이그레이션 프로젝트를 선택합니다.

- a. **sc-project**을 선택합니다.
 - b. 스키마 변환을 선택한 다음, 스키마 변환 종료를 선택합니다.
 - c. 삭제를 선택한 후 선택을 확인합니다.
3. 탐색 창에서 인스턴스 프로파일을 선택합니다.
 - a. **sc-instance**을 선택합니다.
 - b. 삭제를 선택한 후 선택을 확인합니다.
 4. 탐색 창에서 데이터 공급자를 선택합니다.
 - a. **sc-source**와 **sc-target**을 선택합니다.
 - b. 삭제를 선택한 후 선택을 확인합니다.

또한 Amazon S3 버킷, 데이터베이스 보안 암호, AWS Secrets Manager IAM 역할, Virtual Private Cloud(VPC) 등 생성한 다른 AWS 리소스를 정리해야 합니다.

DMS Schema Conversion을 위한 네트워크 설정

DMS Schema Conversion은 Amazon VPC 서비스를 기반으로 Virtual Private Cloud(VPC)에 스키마 변환 인스턴스를 생성합니다. 인스턴스 프로파일을 생성할 때 사용할 VPC를 지정합니다. 계정과 AWS 리전에 기본 VPC를 사용하거나 새 VPC를 생성할 수도 있습니다.

다양한 네트워크 구성을 사용하여 DMS Schema Conversion을 통해 소스 및 대상 데이터베이스의 상호 작용을 설정할 수 있습니다. 이러한 구성은 소스 데이터 공급자의 위치와 네트워크 설정에 따라 달라집니다. 다음 주제에서는 일반적인 네트워크 구성에 대해 설명합니다.

주제

- [소스 및 대상 데이터 공급자를 위한 단일 VPC 사용](#)
- [소스 및 대상 데이터 공급자에 대해 여러 VPC를 사용](#)
- [Direct Connect 또는 VPN을 사용하여 네트워크와 VPC 간 연결 구성](#)
- [VPC에 인터넷 연결 사용](#)

소스 및 대상 데이터 공급자를 위한 단일 VPC 사용

DMS Schema Conversion을 위한 가장 간단한 네트워크 구성은 단일 VPC 구성입니다. 여기서 소스 데이터 공급자, 인스턴스 프로파일, 대상 데이터 공급자는 모두 동일한 VPC에 있습니다. 이 구성을 사용하여 Amazon EC2 인스턴스의 소스 데이터베이스를 변환할 수 있습니다.

이 구성을 사용하려면 인스턴스 프로파일에서 사용하는 VPC 보안 그룹에 데이터 공급자에 대한 액세스 권한이 있어야 합니다. 예를 들어, VPC Classless Inter-Domain Routing(CIDR) 범위 또는 Network Address Translation(NAT) 게이트웨이의 탄력적 IP 주소를 허용할 수 있습니다.

소스 및 대상 데이터 공급자에 대해 여러 VPC를 사용

소스 및 대상 데이터 공급자가 서로 다른 VPC에 있는 경우, VPC 중 하나에 인스턴스 프로파일을 생성할 수 있습니다. 그런 다음, VPC 피어링을 사용하여 이 두 VPC를 연결할 수 있습니다. 이 구성을 사용하여 Amazon EC2 인스턴스의 소스 데이터베이스를 변환할 수 있습니다.

VPC 피어링 연결은 두 VPC가 동일한 네트워크에 있는 것처럼 각 VPC의 프라이빗 IP 주소를 사용하여 라우팅을 활성화하는 두 VPC 간 네트워크 연결입니다. 자체 VPC 간, 다른 AWS 계정에서 VPC를 사용하여 또는 다른 AWS 리전에서 VPC를 사용하여 VPC 피어링 연결을 생성할 수 있습니다. VPC 피어링에 관한 자세한 내용은 Amazon VPC 사용 설명서에서 [VPC 피어링](#)을 참조하세요.

VPC 피어링을 구현하려면 Amazon VPC 사용 설명서의 [VPC 피어링 연결 작업](#)에 나와 있는 지침을 따르십시오. 한 VPC의 라우팅 테이블에 다른 VPC의 CIDR 블록이 포함되어 있는지 확인하십시오. 예를 들어, VPC A가 대상 10.0.0.0/16을 사용하고 VPC B가 대상 172.31.0.0을 사용한다고 가정해 보겠습니다. 이 경우, VPC A의 라우팅 테이블에는 172.31.0.0이 포함되어야 하고 VPC B의 라우팅 테이블에는 10.0.0.0/16이 포함되어야 합니다. 자세한 내용은 Amazon VPC 피어링 가이드의 [VPC 피어링 연결에 대한 라우팅 테이블 업데이트](#)를 참조하십시오.

Direct Connect 또는 VPN을 사용하여 네트워크와 VPC 간 연결 구성

Direct Connect 또는 소프트웨어나 하드웨어 VPN 연결과 같은 여러 옵션을 사용하여 원격 네트워크를 VPC에 연결할 수 있습니다. 이러한 옵션을 사용하여 내부 네트워크를 AWS 클라우드로 확장하여 기존 현장 서비스를 통합할 수 있습니다. 모니터링, 인증, 보안, 데이터 또는 기타 시스템과 같은 현장 서비스를 통합할 수 있습니다. 이러한 유형의 네트워크 확장을 활용하여 AWS에서 호스팅하는 리소스(예: VPC)에 현장 서비스를 원활하게 연결할 수 있습니다. 이 구성을 사용하여 소스 온프레미스 데이터베이스를 변환할 수 있습니다.

이 구성에서 VPC 보안 그룹에는 VPC CIDR 범위 또는 특정 IP 주소로 지정된 트래픽을 호스트로 전송하는 라우팅 규칙이 포함되어야 합니다. 이 호스트는 트래픽을 VPC에서 온프레미스 VPN으로 연결할

수 있어야 합니다. 이 경우, NAT 호스트에는 자체 보안 그룹 설정이 포함됩니다. 이러한 설정은 VPC CIDR 범위 또는 보안 그룹에서 NAT 인스턴스로 들어오는 트래픽을 허용해야 합니다. 자세한 내용은 AWS Site-to-Site VPN 사용 설명서의 [Site-to-Site VPN 연결 생성](#)을 참조하세요.

VPC에 인터넷 연결 사용

AWS 리소스에 연결하기 위해 VPN 또는 Direct Connect를 사용하지 않는 경우, 인터넷 연결을 사용할 수 있습니다. 이 구성에는 인터넷 게이트웨이가 있는 VPC의 프라이빗 서브넷이 포함됩니다. 게이트웨이에는 대상 데이터 공급자와 인스턴스 프로파일도 포함됩니다. 이 구성을 사용하여 소스 온프레미스 데이터베이스를 변환할 수 있습니다.

VPC에 인터넷 게이트웨이를 추가하려면 Amazon VPC 사용 설명서의 [인터넷 게이트웨이 연결](#)을 참조하십시오.

VPC 라우팅 테이블에는 기본적으로 VPC로 향하지 않는 트래픽을 인터넷 게이트웨이로 보내는 라우팅 규칙이 포함되어야 합니다. 이 구성에서는 데이터 공급자에 대한 연결이 NAT 게이트웨이의 퍼블릭 IP 주소에서 시작되는 것으로 보입니다. 자세한 내용은 Amazon VPC 사용 설명서의 [VPC 라우팅 테이블](#)을 참조하세요.

DMS Schema Conversion에서 소스 데이터 공급자 생성

Microsoft SQL Server, Oracle, PostgreSQL, MySQL, IBM DB2 for LUW, IBM DB2 for z/OS 및 SAP ASE(Sybase ASE) 데이터베이스를 DMS Schema Conversion의 마이그레이션 프로젝트에서 소스 데이터 공급자로 사용할 수 있습니다.

소스 데이터 공급자와 DMS Schema Conversion 간의 상호 작용을 허용하도록 네트워크를 구성해야 합니다. 자세한 내용은 [DMS Schema Conversion을 위한 네트워크 설정](#) 단원을 참조하십시오.

주제

- [Microsoft SQL Server 데이터베이스를 DMS Schema Conversion의 소스로 사용](#)
- [DMS Schema Conversion에서 Oracle 데이터베이스를 소스로 사용](#)
- [DMS Schema Conversion에서 Oracle Data Warehouse 데이터베이스를 소스로 사용](#)
- [DMS Schema Conversion에서 PostgreSQL 데이터베이스를 소스로 사용](#)
- [DMS Schema Conversion에서 MySQL 데이터베이스를 소스로 사용](#)
- [Linux, UNIX 및 Windows용 IBM Db2 데이터베이스를 DMS Schema Conversion의 소스로 사용](#)
- [DMS Schema Conversion에서 IBM DB2 for z/OS 데이터베이스를 소스로 사용](#)

- [AWS DMS Schema Conversion에서 SAP ASE\(Sybase ASE\) 데이터베이스를 소스로 사용](#)

Microsoft SQL Server 데이터베이스를 DMS Schema Conversion의 소스로 사용

SQL Server 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

DMS Schema Conversion을 사용하여 SQL Server의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- Aurora MySQL
- Aurora PostgreSQL
- RDS for MySQL
- RDS for PostgreSQL

지원되는 SQL Server 데이터베이스 버전에 관한 자세한 내용은 [DMS Schema Conversion이 지원하는 소스 데이터 공급자](#)를 참조하세요.

소스 SQL Server 데이터베이스에서 DMS Schema Conversion을 사용하는 방법에 관한 자세한 내용은 [SQL Server에서 MySQL로 마이그레이션: 단계별 안내](#)를 참조하세요.

소스로서 Microsoft SQL Server에 대한 권한

Microsoft SQL Server를 소스로 사용하는 데 필요한 다음 권한 목록을 확인하세요.

- 정의 보기
- 데이터베이스 상태 보기

퍼블릭 액세스 권한이 있는 사용자는 이 VIEW DEFINITION 권한을 통해 객체 정의를 볼 수 있습니다. DMS Schema Conversion은 VIEW DATABASE STATE 권한을 사용하여 SQL Server Enterprise 에디션의 기능을 확인합니다.

변환하려는 스키마의 각 데이터베이스에 대해 권한 부여를 반복합니다.

또한 master 데이터베이스에 다음 권한을 부여합니다.

- VIEW SERVER STATE
- 정의 보기

DMS Schema Conversion은 VIEW SERVER STATE 권한을 사용하여 서버 설정 및 구성을 수집합니다. 데이터 공급자를 볼 수 있는 VIEW ANY DEFINITION 권한을 부여했는지 확인하세요.

Microsoft Analysis Services에 관한 정보를 읽으려면 master 데이터베이스에서 다음 명령을 실행합니다.

```
EXEC master..sp_addsrvrolemember @loginame = N'<user_name>', @rolename = N'sysadmin'
```

앞의 예제에서 <user_name> 자리 표시자를 이전에 필요한 권한을 부여한 사용자의 이름으로 바꿉니다.

SQL Server 에이전트에 관한 정보를 읽으려면 사용자를 SQLAgentUser 역할에 추가하세요. msdb 데이터베이스에서 다음 명령을 실행합니다.

```
EXEC sp_addrolemember <SQLAgentRole>, <user_name>;
```

앞의 예제에서 <SQLAgentRole> 자리 표시자를 SQL Server 에이전트 역할의 이름으로 바꿉니다. 그런 다음, <user_name> 자리 표시자를 이전에 필요한 권한을 부여한 사용자의 이름으로 바꿉니다. 자세한 내용은 Amazon RDS 사용 설명서의 [SQLAgentUser 역할에 사용자 추가](#)를 참조하세요.

로그 전달을 감지하려면 msdb 데이터베이스에 대한 SELECT on dbo.log_shipping_primary_databases 권한을 부여하세요.

데이터 정의 언어(DDL) 복제의 알림 접근 방식을 사용하려면 소스 데이터베이스에 대해 RECEIVE ON <schema_name>.<queue_name> 권한을 부여하세요. 이 예제에서는 <schema_name> 자리 표시자를 데이터베이스의 스키마 이름으로 바꿉니다. 그런 다음, <queue_name> 자리 표시자를 대기열 테이블 이름으로 바꾸세요.

DMS Schema Conversion에서 Oracle 데이터베이스를 소스로 사용

Oracle 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

Oracle 데이터베이스에 연결하려면 Oracle 시스템 ID(SID)를 사용합니다. Oracle SID를 확인하려면 Oracle 데이터베이스에 다음 쿼리를 제출합니다.

```
SELECT sys_context('userenv','instance_name') AS SID FROM dual;
```

DMS Schema Conversion을 사용하여 Oracle 데이터베이스의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- Aurora MySQL
- Aurora PostgreSQL
- RDS for MySQL
- RDS for PostgreSQL

지원되는 Oracle 데이터베이스 버전에 관한 자세한 내용은 [DMS Schema Conversion이 지원하는 소스 데이터 공급자](#)를 참조하세요.

소스 Oracle 데이터베이스에서 DMS Schema Conversion을 사용하는 방법에 관한 자세한 내용은 [Oracle에서 PostgreSQL로 마이그레이션: 단계별 안내](#)를 참조하세요.

소스로서 Oracle에 대한 권한

Oracle이 소스일 경우 필요한 권한은 다음과 같습니다.

- CONNECT
- SELECT_CATALOG_ROLE
- SELECT ANY DICTIONARY
- SELECT ON SYS.ARGUMENT\$

DMS Schema Conversion에서 Oracle Data Warehouse 데이터베이스를 소스로 사용

Oracle Data Warehouse 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용하여 데이터베이스 코드 객체와 애플리케이션 코드를 Amazon Redshift로 변환할 수 있습니다.

지원되는 Oracle 데이터베이스 버전에 대한 자세한 내용은 [DMS Schema Conversion이 지원하는 소스 데이터 공급자](#) 섹션을 참조하세요. 소스 Oracle 데이터베이스에서 DMS Schema Conversion을 사용하는 방법에 관한 자세한 내용은 [Oracle에서 PostgreSQL로 마이그레이션: 단계별 안내](#)를 참조하세요.

Oracle Data Warehouse 데이터베이스를 소스로 사용하기 위한 권한

Oracle Data Warehouse를 소스로 사용하기 위해 필요한 권한은 다음과 같습니다.

- CONNECT
- SELECT_CATALOG_ROLE
- SELECT ANY DICTIONARY

Oracle Data Warehouse에서 Amazon Redshift로의 변환 설정

DMS Schema Conversion 설정에 대한 자세한 내용은 [마이그레이션 프로젝트에 대한 스키마 변환 설정 지정](#) 섹션을 참조하세요.

Oracle Data Warehouse를 Amazon Redshift로 변환하는 설정에는 다음과 같은 옵션이 포함됩니다.

- 선택한 심각도 이상의 작업 항목에 대해 변환된 코드에 주석 추가: 이 설정은 변환된 코드에서 작업 항목에 대한 주석 수를 제한합니다. DMS는 선택한 수준 또는 그보다 높은 수준의 심각도에 해당하는 작업 항목에 대해 변환된 코드에 주석을 추가합니다.

예를 들어, 변환된 코드의 설명 수를 최소화하려면 오류만을 선택하세요. 변환된 코드의 모든 작업 항목에 대한 설명을 포함하려면 모든 메시지를 선택합니다.

- 대상 Amazon Redshift 클러스터의 최대 테이블 수: 이 설정은 DMS가 대상 Amazon Redshift 클러스터에 적용할 수 있는 최대 테이블 수를 설정합니다. Amazon Redshift에는 여러 클러스터 노드 유형에 사용하는 테이블을 제한하는 할당량이 있습니다. 이 설정은 다음과 같은 값을 지원합니다.
 - 자동: DMS가 노드 유형에 따라 대상 Amazon Redshift 클러스터에 적용되는 테이블 수를 결정합니다.
 - 값 설정: 테이블 수를 수동으로 설정합니다.

테이블 수가 Amazon Redshift 클러스터가 저장할 수 있는 양보다 많더라도 DMS는 모든 소스 테이블을 변환합니다. DMS는 변환된 코드를 프로젝트에 저장하며 대상 데이터베이스에는 적용하지 않습니다. 변환된 코드를 적용할 때 테이블의 Amazon Redshift 클러스터 할당량에 도달하면 DMS에서 경고 메시지가 표시됩니다. 또한 DMS는 테이블 수가 한도에 도달할 때까지 대상 Amazon Redshift 클러스터에 테이블을 적용합니다.

Amazon Redshift 테이블 할당량에 대한 자세한 내용은 [Amazon Redshift의 할당량 및 제한](#)을 참조하세요.

- UNION ALL 보기 사용: 이 설정을 사용하면 DMS가 단일 소스 테이블에 대해 생성할 수 있는 대상 테이블의 최대 수를 설정할 수 있습니다.

Amazon Redshift는 테이블 파티셔닝을 지원하지 않습니다. 테이블 파티셔닝을 에뮬레이션하고 쿼리를 더 빠르게 실행하기 위해 DMS는 소스 테이블의 각 파티션을 Amazon Redshift의 개별 테이블로 마이그레이션할 수 있습니다. 그런 다음 DMS는 생성하는 모든 대상 테이블의 데이터를 포함하는 보기를 생성합니다.

DMS는 소스 테이블의 파티션 수를 자동으로 결정합니다. 소스 테이블 파티셔닝 유형에 따라서는 이 숫자가 Amazon Redshift 클러스터에 적용할 수 있는 테이블의 할당량을 초과할 수 있습니다. 이 할

당량에 도달하지 않도록 하려면 DMS가 단일 소스 테이블의 파티션에 대해 생성할 수 있는 최대 대상 테이블 수를 입력합니다. 기본 옵션은 368개 테이블이며, 이는 1년 중 366일 동안의 파티션과 NO RANGE 및 UNKNOWN 파티션에 대한 테이블 두 개를 나타냅니다.

- Oracle 코드에서 사용하는 날짜 유형 형식 요소가 Amazon Redshift의 날짜/시간 형식 문자열과 유사: 이 설정을 사용하면 Amazon Redshift에서 지원하지 않는 날짜/시간 형식 요소를 포함하여 TO_CHAR, TO_DATE, TO_NUMBER 같은 데이터 유형 형식 지정 함수를 변환할 수 있습니다. 기본적으로 DMS는 확장 팩 함수를 사용하여 변환된 코드에서 지원되지 않는 이러한 형식 요소를 에뮬레이션합니다.

Oracle의 날짜/시간 형식 모델에는 Amazon Redshift의 날짜/시간 형식 문자열보다 더 많은 요소가 포함되어 있습니다. 소스 코드에 Amazon Redshift가 지원하는 날짜/시간 형식 요소만 포함된 경우 변환된 코드에 확장 팩 함수가 없도록 하려면 이 값을 설정합니다. 확장 함수를 사용하지 않으면 변환된 코드가 더 빠르게 실행됩니다.

- Oracle 코드에서 사용하는 숫자 형식 요소가 Amazon Redshift의 숫자 형식 문자열과 유사: 이 설정을 사용하면 Amazon Redshift에서 지원하지 않는 숫자 데이터 유형 형식 지정 함수를 변환할 수 있습니다. 기본적으로 DMS는 확장 팩 함수를 사용하여 변환된 코드에서 지원되지 않는 이러한 형식 요소를 에뮬레이션합니다.

Oracle의 숫자 형식 모델에는 Amazon Redshift의 숫자 형식 문자열에 비해 더 많은 요소가 포함되어 있습니다. 소스 코드에 Amazon Redshift가 지원하는 숫자 형식 요소만 포함된 경우 변환된 코드에 확장 팩 함수가 없도록 하려면 이 값을 설정합니다. 확장 함수를 사용하지 않으면 변환된 코드가 더 빠르게 실행됩니다.

- NVL 함수를 사용하여 Oracle LEAD 및 LAG 함수의 동작 에뮬레이션: 소스 코드에서 LEAD 및 LAG 함수의 오프셋 기본값을 사용하지 않는 경우 DMS는 NVL 함수를 사용하여 이러한 함수를 에뮬레이션할 수 있습니다. 기본적으로 DMS는 각 LEAD 및 LAG 함수 사용에 대해 작업 항목을 발생시킵니다. NVL을 사용하여 이러한 함수를 에뮬레이션하면 변환된 코드가 더 빠르게 실행됩니다.
- 프라이머리 키와 고유 키의 동작 에뮬레이션: 이 설정을 지정하면 DMS가 대상 Amazon Redshift 클러스터에서 프라이머리 키와 고유 키 제약 조건 동작을 에뮬레이션합니다. Amazon Redshift는 프라이머리 키와 고유 키를 적용하지 않으며 정보 제공 목적으로만 사용합니다. 소스 코드에서 프라이머리 키 또는 고유 키 제약 조건을 사용하는 경우 DMS가 해당 동작을 에뮬레이션하도록 하려면 이 설정을 지정합니다.
- 압축 인코딩 사용: Amazon Redshift 테이블 열에 압축 인코딩을 적용하려면 이 설정을 지정합니다. DMS는 기본 Redshift 알고리즘을 사용하여 압축 인코딩을 자동으로 할당합니다. 압축 인코딩에 대한 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서의 [압축 인코딩](#)을 참조하세요.

기본적으로 Amazon Redshift는 정렬 및 배포 키로 정의된 열에 압축을 적용하지 않습니다. 이러한 열에 압축을 적용하려면 KEY 열에 압축 인코딩 사용을 설정합니다. 압축 인코딩 사용을 설정한 경우에만 이 옵션을 선택할 수 있습니다.

DMS Schema Conversion에서 PostgreSQL 데이터베이스를 소스로 사용

PostgreSQL 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

DMS Schema Conversion을 사용하여 PostgreSQL 데이터베이스의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- MySQL
- Aurora MySQL

PostgreSQL을 소스로 사용하기 위해 필요한 권한은 다음과 같습니다.

- CONNECT ON DATABASE <database_name>
- USAGE ON SCHEMA <database_name>
- SELECT ON ALL TABLES IN SCHEMA <database_name>
- SELECT ON ALL SEQUENCES IN SCHEMA <database_name>

DMS Schema Conversion에서 MySQL 데이터베이스를 소스로 사용

MySQL 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

DMS Schema Conversion을 사용하여 MySQL 데이터베이스의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- PostgreSQL
- Aurora PostgreSQL

MySQL이 소스일 경우 필요한 권한은 다음과 같습니다.

- SELECT ON *.*
- SHOW VIEW ON *.*

MySQL에서 PostgreSQL로 변환 설정

DMS Schema Conversion 설정에 대한 자세한 내용은 [마이그레이션 프로젝트에 대한 스키마 변환 설정 지정](#) 섹션을 참조하세요.

MySQL에서 PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- 변환된 SQL 코드의 주석: 선택한 심각도 이상의 작업 항목에 대한 변환된 코드에 주석을 추가하려면 이 설정을 지정합니다.

유효한 값:

- 오류만
- 오류 및 경고
- 모든 메시지

Linux, UNIX 및 Windows용 IBM Db2 데이터베이스를 DMS Schema Conversion의 소스로 사용

Linux, UNIX 및 Windows용 IBM Db2 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

DMS Schema Conversion을 사용하여 Linux, UNIX 및 Windows 데이터베이스용 Db2의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- Aurora PostgreSQL
- Amazon RDS for PostgreSQL

지원되는 Linux, UNIX 및 Windows용 IBM Db2 데이터베이스 버전에 대한 자세한 내용은 섹션을 참조하세요. [DMS Schema Conversion이 지원하는 소스 데이터 공급자](#)

Linux, UNIX 및 Windows용 IBM Db2를 소스 데이터베이스로 사용하기 위한 사전 조건

다음 코드 예제를 사용하여 Linux, UNIX 및 Windows용 소스 IBM Db2 데이터베이스의 버전을 확인할 수 있습니다.

```
SELECT SERVICE_LEVEL FROM SYSIBMADM.ENV_INST_INFO;
```

Linux, UNIX 및 Windows용 IBM Db2를 소스 데이터베이스로 사용할 수 있는 권한

Linux, UNIX 및 Windows용 Db2 데이터베이스에 연결하고 시스템 카탈로그 및 테이블을 읽는 데 필요한 권한은 다음과 같습니다.

```
EXECUTE ON PACKAGE NULLID.SYSSH200
EXECUTE ON PACKAGE NULLID.SQLC2026 /* If Db2 LUW runs on a Linux operating system */
EXECUTE ON FUNCTION SYSPROC.ENV_GET_INST_INFO
EXECUTE ON FUNCTION SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID
EXECUTE ON FUNCTION SYSPROC.AUTH_LIST_GROUPS_FOR_AUTHID
EXECUTE ON FUNCTION SYSPROC.AUTH_LIST_ROLES_FOR_AUTHID
SELECT ON SYSIBMADM.ENV_INST_INFO
SELECT ON SYSIBMADM.ENV_SYS_INFO
SELECT ON SYSIBMADM.PRIVILEGES
SELECT ON SYSCAT.ATTRIBUTES
SELECT ON SYSCAT.CHECKS
SELECT ON SYSCAT.COLIDENTATTRIBUTES
SELECT ON SYSCAT.COLUMNS
SELECT ON SYSCAT.DATAPARTITIONEXPRESSION
SELECT ON SYSCAT.DATAPARTITIONS
SELECT ON SYSCAT.DATATYPEDEP
SELECT ON SYSCAT.DATATYPES
SELECT ON SYSCAT.HIERARCHIES
SELECT ON SYSCAT.INDEXCOLUSE
SELECT ON SYSCAT.INDEXES
SELECT ON SYSCAT.INDEXPARTITIONS
SELECT ON SYSCAT.KEYCOLUSE
SELECT ON SYSCAT.MODULEOBJECTS
SELECT ON SYSCAT.MODULES
SELECT ON SYSCAT.NICKNAMES
SELECT ON SYSCAT.PERIODS
SELECT ON SYSCAT.REFERENCES
SELECT ON SYSCAT.ROUTINEPARMS
SELECT ON SYSCAT.ROUTINES
SELECT ON SYSCAT.ROWFIELDS
SELECT ON SYSCAT.SCHEMATA
SELECT ON SYSCAT.SEQUENCES
SELECT ON SYSCAT.TABCONST
SELECT ON SYSCAT.TABLES
SELECT ON SYSCAT.TRIGGERS
SELECT ON SYSCAT.VARIABLEDEP
SELECT ON SYSCAT.VARIABLES
```



```
SELECT ON SYSCAT.VIEWS
SELECT ON SYSIBM.SYSDUMMY1
```

DMS Schema Conversion에서 IBM DB2 for z/OS 데이터베이스를 소스로 사용

IBM DB2 for z/OS 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

DMS Schema Conversion을 사용하여 IBM DB2 for z/OS 데이터베이스의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- Amazon RDS for Db2
- Amazon RDS for PostgreSQL
- Aurora PostgreSQL

지원되는 z/OS용 IBM DB2 데이터베이스 버전에 대한 자세한 내용은 [DMS Schema Conversion용 소스 데이터 공급자](#)를 참조하세요.

IBM DB2 for z/OS를 소스 데이터베이스로 사용하기 위한 사전 조건

IBM DB2 for z/OS 버전 12 함수 수준 100 데이터베이스 버전은 IBM DB2 for z/OS 버전 12의 새로운 기능을 대부분 지원하지 않습니다. 이 데이터베이스 버전은 Db2 버전 11에 대한 폴백 및 Db2 버전 11과의 데이터 공유를 지원합니다. Db2 버전 11의 지원되지 않는 기능이 변환되지 않도록 하려면 z/OS용 IBM Db2 데이터베이스 함수 수준 500 이상을 AWS DMS SC의 소스로 사용하는 것이 좋습니다.

다음 코드 예제를 사용하여 소스 IBM DB2 for z/OS 데이터베이스의 버전을 확인할 수 있습니다.

```
SELECT GETVARIABLE('SYSIBM.VERSION') as version FROM SYSIBM.SYSDUMMY1;
```

이 코드가 버전 DSN12015 이상을 반환해야 합니다.

다음 코드 예제를 사용하여 소스 IBM DB2 for z/OS 데이터베이스의 APPLICATION COMPATIBILITY 특수 레지스터 값을 확인할 수 있습니다.

```
SELECT CURRENT APPLICATION COMPATIBILITY as version FROM SYSIBM.SYSDUMMY1;
```

이 코드가 버전 V12R1M500 이상을 반환해야 합니다.

IBM DB2 for z/OS를 소스 데이터베이스로 사용하기 위한 권한

Db2 for z/OS 데이터베이스에 연결하고 시스템 카탈로그 및 테이블을 읽는 데 필요한 권한은 다음과 같습니다.

```
SELECT ON SYSIBM.LOCATIONS
SELECT ON SYSIBM.SYSCHECKS
SELECT ON SYSIBM.SYSCOLUMNS
SELECT ON SYSIBM.SYSDATABASE
SELECT ON SYSIBM.SYSDATATYPES
SELECT ON SYSIBM.SYSDUMMY1
SELECT ON SYSIBM.SYSFOREIGNKEYS
SELECT ON SYSIBM.SYSINDEXES
SELECT ON SYSIBM.SYSKEYCOLUSE
SELECT ON SYSIBM.SYSKEYS
SELECT ON SYSIBM.SYSKEYTARGETS
SELECT ON SYSIBM.SYSJAROBJECTS
SELECT ON SYSIBM.SYSPACKAGE
SELECT ON SYSIBM.SYSPARMS
SELECT ON SYSIBM.SYSRELS
SELECT ON SYSIBM.SYSROUTINES
SELECT ON SYSIBM.SYSSEQUENCES
SELECT ON SYSIBM.SYSSEQUENCESDEP
SELECT ON SYSIBM.SYSSYNONYMS
SELECT ON SYSIBM.SYSTABCONST
SELECT ON SYSIBM.SYSTABLES
SELECT ON SYSIBM.SYSTABLESPACE
SELECT ON SYSIBM.SYSTRIGGERS
SELECT ON SYSIBM.SYSVARIABLES
SELECT ON SYSIBM.SYSVIEWS
```

AWS DMS Schema Conversion에서 SAP ASE(Sybase ASE) 데이터베이스를 소스로 사용

SAP ASE(Sybase ASE) 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

DMS Schema Conversion을 사용하여 SAP ASE(Sybase ASE) 데이터베이스의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- Aurora PostgreSQL
- RDS for PostgreSQL

지원되는 SAP ASE(Sybase ASE) 데이터베이스 버전에 대한 자세한 내용은 섹션을 참조하세요. [DMS Schema Conversion](#)이 지원하는 소스 데이터 공급자

소스 데이터베이스로서 SAP ASE(Sybase ASE)에 대한 권한

SAP ASE(Sybase ASE)를 소스 데이터베이스로 사용할 때는 다음 권한이 필요합니다.

- 마스터 사용
- dbo.spt_values에서 선택
- asehostname에서 선택

마이그레이션할 각 데이터베이스에 대해 다음 권한을 부여합니다.

- USE db_name(db_name을 마이그레이션 중인 데이터베이스 이름으로 대체)
- dbo.sysusers에서 선택
- dbo.sysobjects에서 선택
- dbo.sysindexes에서 선택
- dbo.syscolumns에서 선택
- dbo.sysreferences에서 선택
- dbo.syscomments에서 선택
- dbo.syspartitions에서 선택
- dbo.syspartitionkeys에서 선택
- dbo.sysconstraints에서 선택
- dbo.systypes에서 선택
- dbo.sysqueryplans에서 선택

DMS Schema Conversion에서 대상 데이터 공급자 생성 및 설정

MySQL, PostgreSQL, Amazon Redshift 및 Amazon RDS for Db2 데이터베이스를 DMS Schema Conversion을 위한 마이그레이션 프로젝트에서 대상 데이터 공급자로 사용할 수 있습니다.

주제

- [DMS Schema Conversion에서 MySQL 데이터베이스를 대상으로 사용](#)
- [DMS Schema Conversion에서 PostgreSQL 데이터베이스를 대상으로 사용하기](#)
- [Amazon Redshift 클러스터를 DMS Schema Conversion의 대상으로 사용](#)
- [DMS Schema Conversion에서 Amazon RDS for Db2 데이터베이스를 대상으로 사용](#)

DMS Schema Conversion에서 MySQL 데이터베이스를 대상으로 사용

MySQL 데이터베이스를 DMS Schema Conversion의 마이그레이션 대상으로 사용할 수 있습니다.

지원되는 대상 데이터베이스에 관한 자세한 내용은 [DMS Schema Conversion이 지원하는 대상 데이터 공급자](#) 섹션을 참조하세요.

대상으로서 MySQL에 대한 권한

대상으로서 MySQL에 필요한 권한은 다음과 같습니다.

- CREATE ON *.*
- ALTER ON *.*
- DROP ON *.*
- INDEX ON *.*
- REFERENCES ON *.*
- SELECT ON *.*
- CREATE VIEW ON *.*
- SHOW VIEW ON *.*
- TRIGGER ON *.*
- CREATE ROUTINE ON *.*
- ALTER ROUTINE ON *.*
- EXECUTE ON *.*
- CREATE TEMPORARY TABLES ON *.*
- AWS_LAMBDA_ACCESS (Only valid for Aurora MySQL)
- INSERT, UPDATE ON AWS_ORACLE_EXT.*

- INSERT, UPDATE, DELETE ON AWS_ORACLE_EXT_DATA.*
- INSERT, UPDATE ON AWS_SQLSERVER_EXT.*
- INSERT, UPDATE, DELETE ON AWS_SQLSERVER_EXT_DATA.*
- CREATE TEMPORARY TABLES ON AWS_SQLSERVER_EXT_DATA.*

다음 코드 예제를 사용하여 데이터베이스 사용자를 생성하고 권한을 부여할 수 있습니다.

```
CREATE USER 'user_name' IDENTIFIED BY 'your_password';
GRANT CREATE ON *.* TO 'user_name';
GRANT ALTER ON *.* TO 'user_name';
GRANT DROP ON *.* TO 'user_name';
GRANT INDEX ON *.* TO 'user_name';
GRANT REFERENCES ON *.* TO 'user_name';
GRANT SELECT ON *.* TO 'user_name';
GRANT CREATE VIEW ON *.* TO 'user_name';
GRANT SHOW VIEW ON *.* TO 'user_name';
GRANT TRIGGER ON *.* TO 'user_name';
GRANT CREATE ROUTINE ON *.* TO 'user_name';
GRANT ALTER ROUTINE ON *.* TO 'user_name';
GRANT EXECUTE ON *.* TO 'user_name';
GRANT CREATE TEMPORARY TABLES ON *.* TO 'user_name';
GRANT AWS_LAMBDA_ACCESS TO 'user_name'; (Only valid for Aurora MySQL)
GRANT INSERT, UPDATE ON AWS_ORACLE_EXT.* TO 'user_name';
GRANT INSERT, UPDATE, DELETE ON AWS_ORACLE_EXT_DATA.* TO 'user_name';
GRANT INSERT, UPDATE ON AWS_SQLSERVER_EXT.* TO 'user_name';
GRANT INSERT, UPDATE, DELETE ON AWS_SQLSERVER_EXT_DATA.* TO 'user_name';
GRANT CREATE TEMPORARY TABLES ON AWS_SQLSERVER_EXT_DATA.* TO 'user_name';
```

이전 예제에서 *user_name*을 사용자 이름으로 바꿉니다. 그런 다음 *your_password*를 안전한 암호로 바꿉니다.

Amazon RDS for MySQL 또는 Amazon RDS for Aurora MySQL을 대상으로 사용하려면 `lower_case_table_names` 파라미터를 1로 설정합니다. 이 값은 MySQL 서버가 테이블, 인덱스, 트리거 및 데이터베이스와 같은 객체 이름의 식별자를 대소문자 구분 없이 처리한다는 것을 의미합니다. 대상 인스턴스에서 이진 로깅을 활성화했다면 `log_bin_trust_function_creators` 파라미터를 1로 설정합니다. 이 경우 저장된 함수를 생성하기 위해 DETERMINISTIC, READS SQL DATA 또는 NO SQL 특성을 사용할 필요가 없습니다. 이들 파라미터를 구성하려면 새 DB 파라미터 그룹을 생성하거나 기존 DB 파라미터 그룹을 수정해야 합니다.

DMS Schema Conversion에서 PostgreSQL 데이터베이스를 대상으로 사용하기

PostgreSQL 데이터베이스를 DMS Schema Conversion에서 마이그레이션 대상으로 사용할 수 있습니다.

지원되는 대상 데이터베이스에 관한 자세한 내용은 [DMS Schema Conversion이 지원하는 대상 데이터 공급자](#) 섹션을 참조하세요.

대상으로서 PostgreSQL에 대한 권한

PostgreSQL을 대상으로 사용하려면 DMS Schema Conversion에 다음과 같은 최소 권한이 필요합니다. 연결하기 전에 대상 데이터베이스가 있어야 합니다. 마이그레이션 프로젝트에서 사용하려는 각 데이터베이스에 대해 마이그레이션에 사용되는 사용자 계정에 다음 권한을 부여합니다.

```
GRANT CONNECT ON DATABASE <db_name> TO <user>;
GRANT CREATE ON DATABASE <db_name> TO <user>;
```

Amazon RDS for PostgreSQL을 대상으로 사용하려면 DMS Schema Conversion에 `rds_superuser` 역할이 필요합니다.

변환된 공개 동의어를 사용하려면 다음 명령을 사용하여 데이터베이스 기본 검색 경로를 변경합니다.

```
ALTER DATABASE <db_name> SET SEARCH_PATH = "$user", public_synonyms, public;
```

이 예제에서는 `<db_name>` 자리 표시자를 데이터베이스의 이름으로 바꿉니다.

PostgreSQL에서는 스키마 소유자 또는 `superuser`만 스키마를 삭제할 수 있습니다. 스키마 소유자가 일부 객체를 소유하지 않은 경우에도 스키마와 이 스키마에 포함된 모든 객체를 삭제할 수 있습니다.

여러 사용자를 사용하여 대상 데이터베이스를 변환하고 다른 스키마를 적용할 때 DMS Schema Conversion에서 스키마를 삭제할 수 없는 경우 오류 메시지가 표시될 수 있습니다. 이 오류 메시지가 표시되지 않도록 하려면 `superuser` 역할을 사용하세요.

Amazon Redshift 클러스터를 DMS Schema Conversion의 대상으로 사용

Amazon Redshift 데이터베이스를 DMS Schema Conversion의 마이그레이션 대상으로 사용할 수 있습니다. 지원되는 대상 데이터베이스에 관한 자세한 내용은 [DMS Schema Conversion이 지원하는 대상 데이터 공급자](#) 섹션을 참조하세요.

Amazon Redshift를 대상으로 사용하기 위한 권한

Amazon Redshift를 DMS Schema Conversion의 대상으로 사용하려면 다음과 같은 권한이 필요합니다.

- CREATE ON DATABASE: DMS가 데이터베이스에 새 스키마를 생성할 수 있습니다.
- CREATE ON SCHEMA: DMS가 데이터베이스 스키마에 객체를 생성할 수 있습니다.
- GRANT USAGE ON LANGUAGE: DMS가 데이터베이스에 새 함수와 프로시저를 생성할 수 있습니다.
- GRANT SELECT ON ALL TABLES IN SCHEMA pg_catalog: Amazon Redshift 클러스터에 대한 시스템 정보를 사용자에게 제공합니다.
- GRANT SELECT ON pg_class_info: 사용자에게 테이블 배포 스타일에 대한 정보를 제공합니다.

다음 코드 예제를 사용하여 데이터베이스 사용자를 생성하고 권한을 부여할 수 있습니다. 예제 값을 사용자의 값으로 바꿉니다.

```
CREATE USER user_name PASSWORD your_password;  
GRANT CREATE ON DATABASE db_name TO user_name;  
GRANT CREATE ON SCHEMA schema_name TO user_name;  
GRANT USAGE ON LANGUAGE plpythonu TO user_name;  
GRANT USAGE ON LANGUAGE plpgsql TO user_name;  
GRANT SELECT ON ALL TABLES IN SCHEMA pg_catalog TO user_name;  
GRANT SELECT ON pg_class_info TO user_name;  
GRANT SELECT ON sys_serverless_usage TO user_name;  
GRANT SELECT ON pg_database_info TO user_name;  
GRANT SELECT ON pg_statistic TO user_name;
```

변환된 코드를 적용하거나 데이터를 마이그레이션할 각 대상 스키마에 대해 GRANT CREATE ON SCHEMA 작업을 반복합니다.

대상 Amazon Redshift 데이터베이스에 확장 팩을 적용할 수 있습니다. 확장 팩은 객체를 Amazon Redshift로 변환할 때 필요한 소스 데이터베이스 함수를 에뮬레이션하는 추가 기능 모듈입니다. 자세한 내용은 [DMS Schema Conversion에서 확장 팩 사용](#) 단원을 참조하십시오.

DMS Schema Conversion에서 Amazon RDS for Db2 데이터베이스를 대상으로 사용

Amazon RDS for Db2 데이터베이스를 DMS Schema Conversion의 마이그레이션 대상으로 사용할 수 있습니다.

지원되는 대상 데이터베이스에 대한 자세한 내용은 [DMS Schema Conversion의 대상 데이터 공급자](#)를 참조하세요.

Amazon RDS for Db2를 대상으로 사용하기 위한 권한

Amazon RDS for Db2를 대상으로 사용하려면 DMS Schema Conversion에 master_user_role 역할이 필요합니다. 자세한 내용은 Amazon Relational Database 사용 설명서의 [Amazon RDS for Db2 기본값 역할](#)을 참조하세요.

가상 데이터 공급자

AWS Database Migration Service(DMS)는 스키마 변환 시 데이터 공급자를 위한 가상 모드를 제공합니다. 이 기능을 사용하면 대상 데이터베이스에 연결하지 않고도 스키마 변환을 수행하여 인프라 비용을 절감하고 마이그레이션 계획에 유연성을 제공할 수 있습니다. 가상 모드를 사용하면 리소스를 커밋하기 전에 변환 작업을 즉시 시작하여 최적의 마이그레이션 전략을 계획할 수 있습니다. 호환성을 평가하고, 스키마 코드를 변환 및 검토하고, 다양한 대상 옵션을 테스트할 수도 있습니다. 그런 다음 준비가 되면 데이터베이스에 연결할 수 있습니다. 가상 모드는 MySQL, PostgreSQL, Amazon Redshift 및 Amazon RDS for Db2를 포함하여 AWS DMS Schema Conversion과 호환되는 모든 대상 데이터베이스를 지원합니다.

가상 데이터 공급자 생성

가상 데이터 공급자를 생성하려면 데이터 공급자를 생성할 형식으로 가상 모드를 활성화하기만 하면 됩니다. 자세한 정보는 [DMS Schema Conversion을 위한 데이터 공급자 구성](#)을 참조하세요.

가상 데이터 공급자 사용량

가상 데이터 공급자를 사용하려면 새 마이그레이션 프로젝트를 생성하거나 기존 마이그레이션 프로젝트를 수정합니다. 그런 다음 프로젝트의 대상 데이터 공급자를 생성한 가상 데이터 공급자로 설정합니다.

보안 암호 및 IAM 역할에 가상 공급자를 사용하여 해당 보안 암호의 필드를 읽으려면 IAM 역할에 부여된 보안 암호를 사용하여 마이그레이션 프로젝트를 성공적으로 설정합니다.

대상 공급자를 설정한 후 스키마 변환 탭을 선택합니다. 그런 다음 스키마 변환 시작 버튼을 선택합니다. 프로젝트가 시작될 때까지 기다립니다.

이제 실제 대상 데이터 공급자와 마찬가지로 가상 대상 데이터 공급자에서 스키마 변환을 사용할 수 있습니다. 실제 대상 데이터베이스에 연결해야 하는 작업은 비활성화되지만 대상 트리에 SQL로 저장을 사용할 수 있습니다.

가상 데이터 공급자에서 실제 데이터 공급자로 전환

실제 마이그레이션을 진행할 준비가 되면 실제 데이터 공급자로 전환할 수 있습니다.

가상 데이터 공급자에서 실제 데이터 공급자로 전환하려면 다음 단계를 따르세요.

1. AWS Management Console에 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 데이터 공급자를 선택합니다. 그런 다음 수정하려는 데이터 공급자를 선택합니다.
3. 관련 마이그레이션 프로젝트 섹션으로 이동하여 이 가상 데이터 공급자를 사용하는 모든 마이그레이션 프로젝트를 확인합니다.
4. 첫 번째로 연결된 프로젝트를 선택합니다.
5. 스키마 변환 탭을 선택합니다.
6. 세부 정보 섹션 - 상태를 확인합니다.
7. 상태가 열기이고 스키마 변환 닫기 버튼이 회색으로 표시되지 않는 경우 해당 상태를 선택하고 프로젝트가 닫힐 때까지 기다립니다.
8. 관련된 나머지 모든 마이그레이션 프로젝트에 대해 이전 단계를 반복합니다.
9. 열려 있는 프로젝트가 남아 있지 않은지 확인합니다.
10. 데이터 공급자로 돌아가 수정을 선택합니다.
11. 가상화 모드를 끕니다.
12. 연결 설정을 올바르게 채웁니다. 연결 파라미터는 데이터베이스 엔진에 따라 달라집니다. 자세한 정보는 [데이터 공급자 생성](#)을 참조하세요.
13. 변경 사항 저장을 선택합니다.

Note

변경 사항을 저장한 후에는 가상 모드를 다시 켤 수 없습니다.

14. AWS DMS 콘솔로 돌아갑니다.

15. 탐색 창에서 마이그레이션 프로젝트를 선택합니다. 그런 다음 데이터 공급자를 변경할 마이그레이션 프로젝트를 선택합니다.
16. 수정을 선택합니다.
17. 대상 데이터 공급자에 연결하는 데 사용할 보안 암호를 채웁니다. 데이터베이스에 연결하는 데 필요한 자격 증명이 포함된 올바른 보안 암호를 사용합니다.
18. 대상 보안 암호를 읽는 데 사용할 IAM 역할을 채웁니다. 여기에 지정된 IAM 역할이 올바르고, 연결 자격 증명 보안 암호에 대한 읽기 권한이 부여되었으며, AWS DMS Schema Conversion 서비스에서 사용할 수 있는지 확인합니다.
19. 변경 사항 저장을 선택합니다.
20. 스키마 변환 탭을 선택합니다.
21. 스키마 변환 시작 버튼을 선택합니다.
22. 프로젝트가 시작될 때까지 기다립니다.

Important

가상 모드를 비활성화하고 이 변경 사항을 데이터 공급자에 저장하면 이 작업을 실행 취소할 수 없습니다. 데이터 공급자에 대한 실제 데이터베이스 연결 세부 정보와 관련 마이그레이션 프로젝트에서 보안 암호에 액세스하기 위한 작업 보안 암호 및 IAM 역할을 제공해야 합니다.

전송된 데이터 공급자가 포함된 프로젝트의 첫 번째 시작 시 변환 결과가 표시됩니다. 다음 작업만 사용할 수 있습니다.

- 노드 스키마 - 데이터베이스에서 새로 고침.
- 데이터베이스 객체 - 변경 사항을 적용한 다음 SQL로 저장합니다.

적용 작업은 변환된 객체를 실제 데이터베이스에 적용합니다.

데이터베이스에서 새로 고침 작업은 실제 데이터베이스에서 데이터베이스 객체를 로드합니다. 저장되지 않은 변환 객체는 모두 손실됩니다.

DMS Schema Conversion의 마이그레이션 프로젝트 관리

스키마 변환을 위한 인스턴스 프로파일과 호환 가능한 데이터 공급자를 생성한 후 마이그레이션 프로젝트를 생성하세요. 자세한 내용은 [마이그레이션 프로젝트 생성](#) 단원을 참조하십시오.

DMS Schema Conversion에서 이 새 프로젝트를 사용하려면 마이그레이션 프로젝트 페이지의 목록에서 프로젝트를 선택합니다. 그런 다음, 스키마 변환 탭에서 스키마 변환 시작을 선택합니다.

DMS Schema Conversion을 처음 시작하려면 몇 가지 설정이 필요합니다. AWS Database Migration Service (AWS DMS)는 최대 5분이 걸리는 스키마 변환 인스턴스를 시작합니다. 또한 이 프로세스는 소스 및 대상 데이터베이스에서 메타데이터를 읽습니다. 첫 번째 실행에 성공하면 DMS Schema Conversion에 더 빠르게 액세스할 수 있습니다.

Amazon은 마이그레이션 프로젝트에서 사용하는 스키마 변환 인스턴스를 프로젝트 완료 후 3일 이내에 종료합니다. DMS Schema Conversion에 사용하는 Amazon S3 버킷에서 변환된 스키마와 평가 보고서를 검색할 수 있습니다.

DMS Schema Conversion을 위한 마이그레이션 프로젝트 설정 지정

마이그레이션 프로젝트를 생성하고 스키마 변환을 시작한 후 마이그레이션 프로젝트 설정을 지정할 수 있습니다. 변환 설정을 변경하여 변환된 코드의 성능을 향상시킬 수 있습니다. 또한 스키마 변환 보기를 사용자 지정할 수 있습니다.

변환 설정은 소스 및 대상 데이터베이스 플랫폼에 따라 달라집니다. 자세한 내용은 [소스 데이터 공급자 생성](#) 및 [대상 데이터 공급자 생성 및 설정](#)을 참조하세요.

소스 및 대상 데이터베이스 창에 표시할 스키마와 데이터베이스를 지정하려면 트리 보기 설정을 사용합니다. 빈 스키마, 빈 데이터베이스, 시스템 데이터베이스, 사용자 정의 데이터베이스 또는 스키마를 숨길 수 있습니다.

트리 보기에서 데이터베이스와 스키마를 숨기려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택하고 스키마 변환 탭에서 스키마 변환 시작을 선택합니다.
4. 설정을 선택합니다. 설정 페이지가 열립니다.
5. 트리 보기 섹션에서 다음을 수행합니다.
 - 빈 스키마를 숨기려면 빈 스키마 숨기기를 선택합니다.
 - 빈 데이터베이스를 숨기려면 빈 데이터베이스 숨기기를 선택합니다.
 - 시스템 데이터베이스 또는 스키마의 경우, 시스템 데이터베이스 및 스키마를 이름별로 선택하여 숨깁니다.

- 사용자 정의 데이터베이스 또는 스키마의 경우, 숨기려는 사용자 정의 데이터베이스 및 스키마의 이름을 입력합니다. 추가를 선택합니다. 이름은 대/소문자를 구분하지 않습니다.

여러 데이터베이스 또는 스키마를 추가하려면 쉼표를 사용하여 이름을 구분합니다. 이름이 비슷한 객체를 여러 개 추가하려면 백분율(%)을 와일드카드로 사용합니다. 이 와일드카드는 데이터베이스 또는 스키마 이름에 있는 기호의 수를 대체합니다.

소스 및 대상 섹션에 대해 이 단계를 반복합니다.

6. 적용을 선택한 다음, 스키마 변환을 선택합니다.

AWS DMS Schema Conversion에 대한 액세스 로그

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택하고 개요 탭의 ARN 필드에서 마이그레이션 프로젝트 ID를 복사합니다.

Overview

Schema conversion

Summary

Migration project

Name
ora23ai-to-pg163

ARN
 arn:aws:dms:eu-central-1:498160209112:migration-project:K32W6ELWVZBRNG6TSAAP3DPGMQ

4. CloudWatch 서비스를 엽니다.
5. 로그 그룹을 선택하고 3단계에서 {migration_project_id}가 id인 dms-tasks-sct-{migration_project_id}를 입력합니다.
6. 로그 그룹 내에서 로그가 있는 로그 스트림을 찾을 수 있습니다.

DMS Schema Conversion을 사용하여 데이터베이스 마이그레이션 평가 보고서 생성

DMS Schema Conversion의 중요한 부분은 스키마를 변환하는 데 도움이 되도록 생성하는 보고서입니다. 이 데이터베이스 마이그레이션 평가 보고서는 모든 스키마 변환 태스크를 요약합니다. 또한 대상 DB 인스턴스의 DB 엔진으로 변환할 수 없는 스키마에 대한 작업 항목도 자세히 설명합니다. AWS DMS 콘솔에서 보고서를 보거나 이 보고서의 사본을 PDF 또는 CSV(쉼표로 구분된 값) 파일로 저장할 수 있습니다.

마이그레이션 평가 보고서에는 다음 내용이 포함됩니다.

- 핵심 요약
- 서버 객체 변환, 백업 제안, 연결된 서버 변경을 포함한 권장 사항

자동으로 변환할 수 없는 항목이 있을 경우, DMS Schema Conversion 보고서는 대상 DB 인스턴스에 맞는 동등한 코드를 작성하는 데 얼마나 많은 작업이 필요한지에 대한 추정치를 제공합니다.

주제

- [DMS Schema Conversion용 데이터베이스 마이그레이션 평가 보고서 생성](#)
- [DMS Schema Conversion용 데이터베이스 마이그레이션 평가 보고서 보기](#)
- [DMS Schema Conversion용 데이터베이스 마이그레이션 평가 보고서 저장](#)

DMS Schema Conversion용 데이터베이스 마이그레이션 평가 보고서 생성

마이그레이션 프로젝트를 생성한 후 다음 절차를 사용하여 데이터베이스 마이그레이션 평가 보고서를 작성합니다.

데이터베이스 마이그레이션 평가 보고서를 작성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음, 스키마 변환을 선택합니다.
4. 스키마 변환 시작을 선택합니다. 스키마 변환 페이지가 열립니다.
5. 소스 데이터베이스 창에서 평가할 데이터베이스 스키마 또는 스키마 항목을 선택합니다. 보고서에 여러 객체를 포함하려면 모든 항목을 선택해야 합니다.
6. 평가하려는 모든 스키마 객체의 확인란을 선택한 후에는 선택한 객체의 상위 노드를 선택해야 합니다. 이제 소스 데이터베이스 창의 작업 메뉴를 사용할 수 있습니다.
7. 작업 메뉴에서 평가를 선택합니다. 확인 대화 상자가 표시됩니다.
8. 대화 상자에서 평가를 선택하여 선택을 확인합니다.

DMS Schema Conversion용 데이터베이스 마이그레이션 평가 보고서 보기

평가 보고서를 작성하면 DMS Schema Conversion은 다음 탭에 정보를 추가합니다.

- 요약
- 작업 항목

요약 탭에는 DMS Schema Conversion이 자동으로 변환할 수 있는 항목 수가 표시됩니다.

작업 항목 탭에는 DMS Schema Conversion이 자동으로 변환할 수 없는 항목이 표시되며 이러한 항목을 관리하는 방법에 대한 권장 사항이 제공됩니다.

평가 보고서 요약

요약 탭에는 데이터베이스 마이그레이션 평가 보고서의 요약 정보가 표시됩니다. 여기에는 DMS Schema Conversion이 데이터베이스 스토리지 객체 및 데이터베이스 코드 객체에 대해 자동으로 변환할 수 있는 항목의 수가 표시됩니다.

대부분의 경우 DMS Schema Conversion은 모든 스키마 항목을 대상 데이터베이스 엔진으로 자동 변환할 수 없습니다. 요약 탭은 대상 DB 인스턴스에서 소스의 스키마 항목과 동일한 스키마 항목을 생성하는 데 필요한 예상 작업량을 제공합니다.

테이블, 시퀀스, 제약 조건, 데이터 형식 등과 같은 데이터베이스 스토리지 객체의 변환 요약을 보려면 데이터베이스 스토리지 객체를 선택합니다.

프로시저, 함수, 뷰, 트리거 등과 같은 데이터베이스 코드 객체의 변환 요약을 보려면 데이터베이스 코드 객체를 선택합니다.

평가 보고서의 범위를 변경하려면 소스 데이터베이스 트리에서 필요한 노드를 선택합니다. DMS Schema Conversion은 선택한 범위와 일치하도록 평가 보고서 요약을 업데이트합니다.

평가 보고서 작업 항목

작업 항목 탭에는 DMS Schema Conversion에서 대상 데이터베이스 엔진과 호환되는 형식으로 자동 변환할 수 없는 항목 목록이 포함되어 있습니다. 각 작업 항목에 대해 DMS Schema Conversion은 문제의 설명 및 권장 조치를 제공합니다. DMS Schema Conversion은 유사한 작업 항목을 그룹화하고 발생 횟수를 표시합니다.

관련 데이터베이스 객체의 코드를 보려면 목록에서 작업 항목을 하나 선택합니다.

생성형 AI를 사용한 평가 보고서

생성형 AI를 사용한 DMS Schema Conversion은 일반적으로 복잡한 수동 변환이 필요한 이전에 변환되지 않은 코드 객체를 변환하는 데 도움이 되는 권장 사항을 제공하여 데이터베이스 마이그레이션 프로세스를 간소화합니다. 현재 이 기능은 Oracle에서 PostgreSQL/Aurora PostgreSQL로, SQL Server에서 PostgreSQL/Aurora PostgreSQL로 변환하는 데 사용할 수 있습니다.

평가 보고서는 생성형 AI를 사용하여 변환할 수 있는 데이터베이스 코드 객체를 보고합니다. 이 보고서를 PDF로 내보내 작업 항목당 객체 5개의 짧은 목록을 볼 수 있습니다. 보고서를 CSV 파일로 내보내서 전체 후보 목록을 볼 수 있습니다.

생성형 AI를 사용하여 스키마 변환을 활성화하고 변환을 수행하면 업데이트된 평가 보고서는 AI를 사용하여 성공적으로 변환된 객체를 식별합니다. 이러한 객체는 해당 작업 항목으로 명확하게 표시되므로 AI 생성 변환 권장 사항을 쉽게 식별하고 추적할 수 있습니다.

생성형 AI로 변환되는 객체를 빠르게 찾으려면

1. 작업 항목 탭으로 이동합니다.
2. 오른쪽 상단 모서리에서 드롭다운 메뉴를 찾습니다.
3. 옵션에서 AI에 의해 생성됨을 선택합니다.

이 필터링 방법을 사용하면 이 기능을 사용하여 성공적으로 변환된 모든 데이터베이스 객체를 효율적으로 볼 수 있습니다.

DMS Schema Conversion용 데이터베이스 마이그레이션 평가 보고서 저장

데이터베이스 마이그레이션 평가 보고서를 작성한 후 이 보고서의 사본을 PDF 또는 CSV(선택으로 구분된 값) 파일로 저장할 수 있습니다.

데이터베이스 마이그레이션 평가 보고서를 PDF 파일로 저장하려면

1. 내보내기를 선택한 다음 PDF를 선택합니다. 대화 상자를 검토하고 PDF로 내보내기를 선택합니다.
2. DMS Schema Conversion은 PDF 파일로 아카이브를 생성하고 이 아카이브를 Amazon S3 버킷에 저장합니다. Amazon S3 버킷을 변경하려면 인스턴스 프로파일에서 스키마 변환 설정을 편집하세요.
3. Amazon S3 버킷에서 평가 보고서 파일을 엽니다.

데이터베이스 마이그레이션 평가 보고서를 CSV 파일로 저장하려면

1. 내보내기를 선택한 다음 CSV를 선택합니다. 대화 상자를 검토하고 CSV로 내보내기를 선택합니다.
2. DMS Schema Conversion은 CSV 파일로 아카이브를 생성하고 이 아카이브를 Amazon S3 버킷에 저장합니다. Amazon S3 버킷을 변경하려면 인스턴스 프로파일에서 스키마 변환 설정을 편집하세요.
3. Amazon S3 버킷에서 평가 보고서 파일을 엽니다.

PDF 파일에는 요약과 작업 항목 정보가 모두 들어 있습니다.

평가 보고서를 CSV로 내보내는 경우 DMS Schema Conversion이 세 개의 CSV 파일을 생성합니다.

첫 번째 CSV 파일은 다음 작업 항목에 대한 정보를 포함합니다.

- 카테고리
- 발생
- 작업 항목
- 제목
- Group
- 설명
- 문서 참조
- 권장 조치
- 행
- 위치
- 소스

- 대상
- 서버 IP 주소 및 포트
- Database
- 스키마

두 번째 CSV 파일은 이름에 Action_Items_Summary 접미사가 포함되며 다음 정보를 포함하고 있습니다.

- 스키마
- 작업 항목
- 발생 횟수
- 학습 곡선 노력(각 작업 항목을 변환하기 위한 접근 방식을 설계하는 데 필요한 노력의 양)
- 각 작업 항목을 변환하기 위한 작업량(설계된 접근 방식에 따라 각 작업 항목을 변환하는 데 필요한 노력)
- 작업 항목 설명
- 권장 조치

필요한 노력의 수준을 나타내는 값은 낮음(최소)에서 높음(가장)까지의 가중치 척도를 기반으로 합니다.

세 번째 CSV 파일은 이름에 Summary가 포함되며 다음 정보를 포함하고 있습니다.

- 카테고리
- 객체 수
- 자동으로 변환된 객체
- 단순 작업이 포함된 객체
- 중간 복잡도의 작업이 포함된 객체
- 복잡한 작업이 포함된 객체
- 총 코드 줄 수

DMS Schema Conversion 사용

DMS Schema Conversion은 기존 데이터베이스 스키마와 대부분의 데이터베이스 코드 객체를 대상 데이터베이스와 호환되는 형식으로 변환합니다.

DMS Schema Conversion은 온라인 트랜잭션 처리(OLTP) 데이터베이스 스키마를 Amazon RDS for MySQL 또는 RDS for PostgreSQL로 변환하는 대부분의 프로세스를 자동화합니다. 소스 및 대상 데이터베이스 엔진에는 다양한 특징과 기능이 포함되어 있으며 DMS Schema Conversion은 가능하다면 동일한 스키마를 생성하려고 합니다. 직접 변환이 불가능한 데이터베이스 객체의 경우, DMS Schema Conversion은 수행할 작업 목록을 제공합니다.

데이터베이스 스키마를 변환하려면 다음 프로세스를 사용하세요.

- 데이터베이스 스키마를 변환하기 전에 변환 중에 데이터베이스 객체의 이름을 변경하는 변환 규칙을 설정하세요.
- 마이그레이션의 복잡성을 추정하려면 데이터베이스 마이그레이션 평가 보고서를 생성하세요. 이 보고서는 DMS Schema Conversion이 자동으로 변환할 수 없는 스키마 요소에 관한 세부 정보를 제공합니다.
- 소스 데이터베이스 스토리지와 코드 객체를 변환하세요. DMS Schema Conversion은 변환된 데이터베이스 객체의 로컬 버전을 생성합니다. 마이그레이션 프로젝트에서 이러한 변환된 객체에 액세스할 수 있습니다.
- 변환된 코드를 SQL 파일에 저장하여 변환 작업 항목을 검토, 편집 또는 처리할 수 있습니다. 변환된 코드를 대상 데이터베이스에 직접 적용할 수도 있습니다.

주제

- [DMS Schema Conversion에서 변환 규칙 설정](#)
- [DMS Schema Conversion에서 데이터베이스 스키마 변환: step-by-step 가이드](#)
- [생성형 AI를 사용하여 데이터베이스 객체 변환](#)
- [마이그레이션 프로젝트에 대한 스키마 변환 설정 지정](#)
- [DMS Schema Conversion에서 데이터베이스 스키마 새로 고침](#)
- [변환된 코드를 DMS Schema Conversion에 저장 및 적용](#)
- [Java 애플리케이션의 임베디드 SQL 변환](#)

DMS Schema Conversion에서 변환 규칙 설정

DMS Schema Conversion으로 데이터베이스 스키마를 변환하기 전에 변환 규칙을 설정할 수 있습니다. 변환 규칙은 객체 이름을 소문자나 대문자로 변경하고 접두사나 접미사를 추가하거나 제거하며 객체 이름을 바꾸는 등의 작업을 수행할 수 있습니다. 예를 들면, 소스 스키마에 test_TABLE_NAME이라는 테이블 집합이 있다고 가정해 보세요. 대상 스키마의 접두사 test_를 접두사 demo_로 변경하는 규칙을 설정할 수 있습니다.

다음 작업을 수행하는 변환 규칙을 생성할 수 있습니다.

- 접두사 추가, 제거 또는 교체
- 접미사 추가, 제거 또는 교체
- 열의 데이터 형식 변경
- 객체 이름을 소문자나 대문자로 변경
- 객체 이름 변경

다음 객체에 대한 변환 규칙을 생성할 수 있습니다.

- 스키마
- 표
- 열

변환 규칙 생성

DMS Schema Conversion은 변환 규칙을 마이그레이션 프로젝트의 일부로 저장합니다. 마이그레이션 프로젝트를 생성할 때 변환 규칙을 설정하거나 나중에 편집할 수 있습니다.

프로젝트에 여러 변환 규칙을 추가할 수 있습니다. 변환 중에 DMS Schema Conversion은 변환 규칙을 추가한 순서와 동일한 순서로 적용합니다.

변환 규칙을 생성하려면

1. 마이그레이션 프로젝트 생성 페이지에서 변환 규칙 추가를 선택합니다. 자세한 내용은 [마이그레이션 프로젝트 생성](#) 단원을 참조하십시오.
2. 규칙 대상에서 이 규칙을 적용할 데이터베이스 객체 유형을 선택합니다.
3. 소스 스키마에서 스키마 입력을 선택합니다. 그런 다음, 이 규칙이 적용되는 소스 스키마, 테이블 및 열의 이름을 입력합니다. 정확한 이름을 입력하여 하나의 객체를 선택하거나 패턴을 입력하여 여러 객체를 선택할 수 있습니다. 백분율(%)을 와일드카드로 사용하여 데이터베이스 객체 이름에 있는 여러 기호를 바꿀 수 있습니다.
4. 작업에서 수행할 작업을 선택합니다.
5. 규칙 유형에 따라 하나 또는 두 개의 추가 값을 입력합니다. 예를 들어, 객체의 이름을 바꾸려면 객체의 새 이름을 입력합니다. 접두사를 바꾸려면 이전 접두사와 새 접두사를 입력합니다.
6. 변환 규칙 추가를 선택하여 다른 변환 규칙을 추가합니다.

규칙 추가를 완료한 후 마이그레이션 프로젝트 생성을 선택합니다.

기존 변환 규칙을 복제하려면 복제를 선택합니다. 기존 변환 규칙을 편집하려면 목록에서 규칙을 선택합니다. 기존 변환 규칙을 삭제하려면 제거를 선택합니다.

변환 규칙 편집

마이그레이션 프로젝트에서 기존 변환 규칙을 새로 추가, 제거 또는 편집할 수 있습니다. DMS Schema Conversion은 스키마 변환을 시작하는 동안 변환 규칙을 적용하므로 규칙을 편집한 후 스키마 변환을 종료하고 다시 실행해야 합니다.

변환 규칙을 편집하려면

1. 에 로그인 AWS Management Console하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택한 다음, 마이그레이션 프로젝트를 선택합니다.
3. 스키마 변환을 선택한 다음, 스키마 변환 종료를 선택합니다.
4. 가 스키마 변환을 AWS DMS 닫은 후 수정을 선택하여 마이그레이션 프로젝트 설정을 편집합니다.
5. 변환 규칙에서 다음 작업 중 하나를 선택합니다.
 - 기존 변환 규칙을 복제하여 목록 끝에 추가하려면 복제를 선택합니다.
 - 제거를 선택하여 기존 변환 규칙을 제거합니다.
 - 기존 변환 규칙을 선택하여 편집합니다.
6. 편집 규칙을 완료한 후 변경 내용 저장을 선택합니다.
7. 마이그레이션 프로젝트 페이지의 목록에서 프로젝트를 선택합니다. 스키마 변환을 선택한 다음, 스키마 변환 시작을 선택합니다.

DMS Schema Conversion에서 데이터베이스 스키마 변환: step-by-step 가이드

마이그레이션 프로젝트를 생성하고 소스 및 대상 데이터베이스에 연결한 후 소스 데이터베이스 객체를 대상 데이터베이스와 호환되는 형식으로 변환할 수 있습니다. DMS Schema Conversion은 소스 데이터베이스 스키마를 왼쪽 패널에 트리 보기 형식으로 표시합니다.

데이터베이스 트리의 각 노드가 지연 로드됩니다. 트리 보기에서 노드를 선택하면 DMS Schema Conversion이 이때 소스 데이터베이스의 스키마 정보를 요청합니다. 스키마 정보를 더 빨리 로드하려면 스키마를 선택한 다음, 작업 메뉴에서 메타데이터 로드를 선택합니다. 그런 다음, DMS Schema Conversion은 데이터베이스 메타데이터를 읽고 Amazon S3 버킷에 정보를 저장합니다. 이제 데이터베이스 객체를 더 빠르게 찾아볼 수 있습니다.

전체 데이터베이스 스키마를 변환하거나 소스 데이터베이스에서 변환할 스키마 항목을 선택할 수 있습니다. 선택한 스키마 항목이 상위 항목에 따라 달라지는 경우, DMS Schema Conversion은 상위 항목에 대한 스키마도 생성합니다. 예를 들어, 변환할 테이블을 선택하면 DMS Schema Conversion은 변환된 테이블과 이 테이블이 속한 데이터베이스 스키마를 생성합니다.

데이터베이스 객체 변환

DMS Schema Conversion을 사용하여 전체 데이터베이스 스키마 또는 개별 데이터베이스 스키마 객체를 변환할 수 있습니다.

전체 데이터베이스 스키마를 변환하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음, 스키마 변환을 선택합니다.
4. 스키마 변환 시작을 선택합니다. 스키마 변환 페이지가 열립니다.
5. 소스 데이터베이스 창에서 스키마 이름 확인란을 선택합니다.
6. 마이그레이션 프로젝트의 왼쪽 창에서 이 스키마를 선택합니다. DMS Schema Conversion은 스키마 이름을 파란색으로 강조 표시하고 작업 메뉴를 활성화합니다.
7. 작업에서 변환을 선택합니다. 변환 대화 상자가 나타납니다.
8. 대화 상자에서 변환을 선택하여 선택을 확인합니다.

소스 데이터베이스 객체를 변환하려면

1. 에 로그인 AWS Management Console하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음, 스키마 변환을 선택합니다.
4. 스키마 변환 시작을 선택합니다. 스키마 변환 페이지가 열립니다.

5. 소스 데이터베이스 창에서 소스 데이터베이스 객체를 선택합니다.
6. 변환하려는 객체의 확인란을 모두 선택한 후 왼쪽 패널에서 선택한 모든 객체의 상위 노드를 선택합니다.

DMS Schema Conversion은 상위 노드를 파란색으로 강조 표시하고 작업 메뉴를 활성화합니다.

7. 작업에서 변환을 선택합니다. 변환 대화 상자가 나타납니다.
8. 대화 상자에서 변환을 선택하여 선택을 확인합니다.

예를 들어, 10개 테이블 중 2개를 변환하려면 변환하려는 두 테이블의 확인란을 선택합니다. 작업 메뉴가 비활성 상태인 것을 알 수 있습니다. 테이블 노드를 선택하면 DMS Schema Conversion에서 해당 이름이 파란색으로 강조 표시되고 작업 메뉴가 활성화됩니다. 그런 다음, 이 메뉴에서 변환을 선택할 수 있습니다.

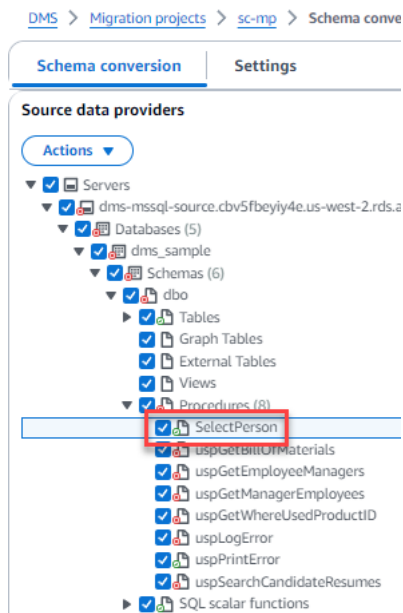
마찬가지로 2개의 테이블과 3개의 프로시저를 변환하려면 객체 이름의 확인란을 선택합니다. 그런 다음, 스키마 노드를 선택하여 작업 메뉴를 활성화하고 스키마 변환을 선택합니다.

변환된 SQL 코드 편집 및 저장

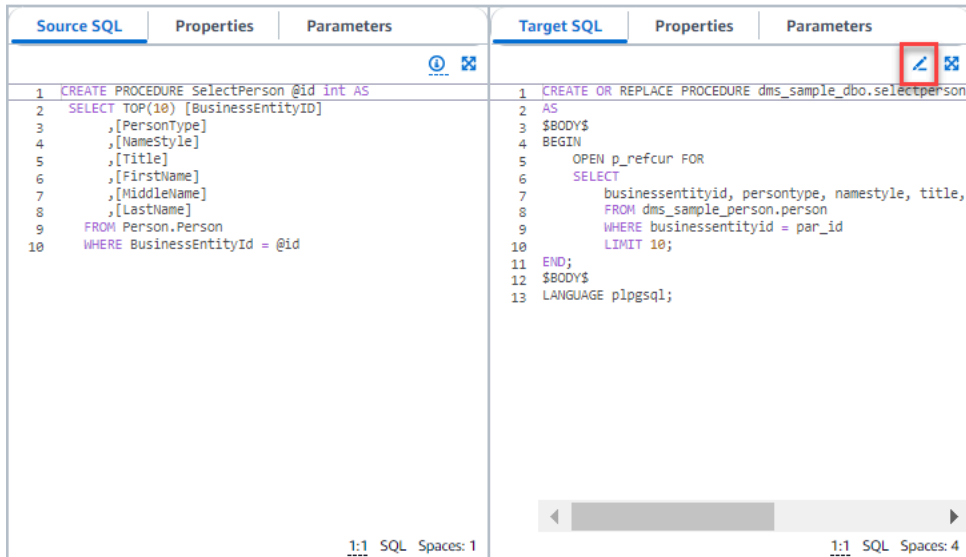
스키마 변환 페이지에서는 데이터베이스 객체에서 변환된 SQL 코드를 편집할 수 있습니다. 다음 절차를 사용하여 변환된 SQL 코드를 편집하고 변경 내용을 적용한 다음 저장합니다.

변환된 SQL 코드를 편집하고, 변경 내용을 적용하고 저장하려면

1. 스키마 변환 페이지에서 소스 데이터 공급자 창에서 트리 보기를 열어 코드 객체를 표시합니다.



2. 소스 데이터 공급자 창에서 작업, 변환을 선택합니다. 작업을 확인합니다.
3. 변환이 완료되면 변환된 SQL을 보려면 필요한 경우 중앙 창을 확장합니다. 변환된 SQL을 편집하려면 대상 SQL 창에서 편집 아이콘을 선택합니다.



4. 대상 SQL을 편집한 후 페이지 상단의 확인 아이콘을 선택하여 변경 사항을 확인합니다. 작업을 확인합니다.
5. 대상 데이터 공급자 창에서 작업, 변경 사항 적용을 선택합니다. 작업을 확인합니다.
6. DMS는 편집된 프로시저를 대상 데이터 저장소에 기록합니다.

변환된 데이터베이스 객체 검토

소스 데이터베이스 객체를 변환한 후 프로젝트의 왼쪽 창에서 객체를 선택할 수 있습니다. 그러면 해당 객체의 소스 및 변환된 코드를 볼 수 있습니다. DMS Schema Conversion은 왼쪽 창에서 선택한 객체의 변환된 코드를 자동으로 로드합니다. 선택한 객체의 속성이나 파라미터도 볼 수 있습니다.

DMS Schema Conversion은 변환된 코드를 마이그레이션 프로젝트의 일부로 자동 저장합니다. 대상 데이터베이스에는 이러한 코드 변경 내용이 적용되지 않습니다. 변환된 코드를 대상 데이터베이스에 적용하는 방법에 관한 자세한 내용은 [변환된 코드 적용](#)을 참조하세요. 변환된 코드를 마이그레이션 프로젝트에서 제거하려면 오른쪽 창에서 대상 스키마를 선택한 다음, 작업에서 데이터베이스에서 새로 고침을 선택합니다.

소스 데이터베이스 객체를 변환한 후에는 하단 중앙 창에서 변환 요약과 작업 항목을 볼 수 있습니다. 평가 보고서를 생성할 때 동일한 정보를 볼 수 있습니다. 평가 보고서는 DMS Schema Conversion이 변환할 수 없는 스키마 항목을 식별하고 해결하는 데 유용합니다. 평가 보고서 요약과 전환 작업 항목

목록을 CSV 파일에 저장할 수 있습니다. 자세한 내용은 [데이터베이스 마이그레이션 평가 보고서](#) 단원을 참조하십시오.

생성형 AI를 사용하여 데이터베이스 객체 변환

생성형 AI 기능을 사용한 DMS 스키마 변환 기능은 일반적으로 복잡한 수동 변환이 필요한 이전에 변환되지 않은 코드 객체를 변환하는 데 도움이 되는 권장 사항을 제공하여 데이터베이스 마이그레이션 프로세스를 간소화합니다. 이 기능은 Oracle에서 PostgreSQL/Aurora PostgreSQL로, SQL Server에서 PostgreSQL/Aurora PostgreSQL로, SAP ASE(Sybase ASE)에서 PostgreSQL/Aurora PostgreSQL로 변환하는 데 사용할 수 있습니다. 전체 데이터베이스 스키마 또는 개별 데이터베이스 스키마 객체를 변환할 수 있습니다.

Warning

DMS Schema Conversion의 생성형 AI 기능은 교차 리전 추론을 사용합니다. 자세한 내용은 [DMS Schema Conversion의 교차 리전 추론](#) 단원을 참조하십시오.

생성형 AI를 사용하여 소스 데이터베이스 객체를 변환하려면, [데이터베이스 객체 변환](#)의 1~6단계를 수행한 다음 아래 두 방법 중 하나를 계속합니다.

- 방법 1: 작업 메뉴에서 변환을 선택합니다. 표시되는 변환 대화 상자에서 생성형 AI로 스키마 변환 옵션을 활성화하고 변환을 클릭합니다.
- 방법 2: 오른쪽 상단 모서리에서 생성형 AI로 스키마 변환을 클릭합니다. 변환 대화 상자에서 옵션이 활성화되어 있는지 확인하고 변환을 클릭합니다.

DMS Schema Conversion 콘솔에서 언제든지 이 설정을 수동으로 조정하려면

- 설정 탭으로 이동합니다.
- 변환 설정 섹션에서 생성형 AI 옵션을 활성화하여 생성형 AI 사용을 승인합니다.

Note

지원되는 리전:

- 아시아 태평양(도쿄)(ap-northeast-1)
- 아시아 태평양(오사카) (ap-northeast-3)

- 아시아 태평양(시드니)(ap-southeast-2)
- 캐나다(중부)(ca-central-1)
- 유럽(프랑크푸르트)(eu-central-1)
- 유럽(스톡홀름)(eu-north-1)
- 유럽(아일랜드)(eu-west-1)
- 유럽(런던) (eu-west-2)
- 유럽(파리)(eu-west-3)
- 미국 동부(버지니아 북부)(us-east-1)
- 미국 동부(오하이오)(us-east-2)
- 미국 서부(오리건)(us-west-2)

Note

지원되는 변환 경로:

- Oracle에서 Amazon RDS for PostgreSQL로
- Oracle에서 Amazon Aurora PostgreSQL로
- Microsoft SQL Server에서 Amazon RDS for PostgreSQL로
- Microsoft SQL Server에서 Amazon Aurora PostgreSQL로
- SAP ASE(Sybase ASE)에서 Amazon RDS for PostgreSQL로
- SAP ASE(Sybase ASE)에서 Amazon Aurora PostgreSQL로
- Linux, UNIX 및 Windows용 IBM Db2(LUW)에서 Amazon RDS for PostgreSQL로
- Linux, UNIX 및 Windows용 IBM Db2(LUW)에서 Amazon Aurora PostgreSQL로
- z/OS용 IBM Db2에서 Amazon RDS for PostgreSQL로
- z/OS용 IBM Db2에서 Amazon Aurora PostgreSQL로

생성형 AI 변환 범위

생성형 AI 지원 스키마 변환은 지정된 작업 항목이 있는 특정 SQL 요소에 중점을 둡니다. 다른 모든 SQL 요소는 기본 규칙 기반 접근 방식을 사용하여 변환됩니다. 생성형 AI 변환의 확장 범위 내에 있는 SQL 요소는 다음과 같습니다.

변환 방향	작업 항목	메시지	구문 요소
Oracle에서 Amazon RDS for PostgreSQL로 및 Oracle에서 Amazon Aurora PostgreSQL로	5578	AWS DMS Schema Conversion은 SELECT 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	30415	MERGE 문에는 대상 테이블 열의 값을 기반으로 하는 WHERE 절의 필터링 조건이 포함됩니다	제한을 제외한 모든 발생
	5591	AWS DMS Schema Conversion은 시스템 객체를 변환할 수 없습니다.	제한을 제외한 모든 발생
	5029	AWS DMS Schema Conversion은 지원되지 않는 데이터 형식의 객체 사용을 변환할 수 없습니다.	함수 및 프로시저 인수에서 지원되지 않는 STANDARD.BFILE 데이터 형식의 객체 사용.
	5031	AWS DMS Schema Conversion은 CURSOR 표현식을 변환할 수 없습니다.	제한을 제외한 모든 발생
	5043	AWS DMS Schema Conversion은 SELECT 절에서 별표가 있는 계층적 쿼리를 변환할 수 없습니다.	제한을 제외한 모든 발생
	5073	PostgreSQL은 의사 열을 사용한 계층적 쿼	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
		리를 지원하지 않습니다	
	5102	PostgreSQL은 MERGE 문을 지원하지 않습니다	제한을 제외한 모든 발생
	5585	AWS DMS Schema Conversion은 외부 조인을 상관관계가 있는 하위 쿼리로 변환할 수 없습니다.	제한을 제외한 모든 발생
	5608	AWS DMS Schema Conversion은 SET 절에서 여러 열을 반환하는 하위 쿼리가 있는 UPDATE 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	5619	AWS DMS Schema Conversion은 시스템 객체를 변환할 수 없습니다.	제한을 제외한 모든 발생
	5852	PostgreSQL은 MERGE 문에서 테이블만 대상으로 지원합니다.	제한을 제외한 모든 발생
	5853	AWS DMS Schema Conversion은 MERGE 문의 USING 절에서 테이블, 뷰 또는 하위 쿼리만 소스로 지원합니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	5855	MERGE 문에는 대상 테이블 열의 값을 기반으로 하는 WHERE 절의 필터링 조건이 포함됩니다	제한을 제외한 모든 발생
	9996	내부 변환기 오류 발생	제한을 제외한 모든 발생
	9993	해결되지 않은 객체에 대한 참조로 인해 문을 변환할 수 없음	제한을 제외한 모든 발생
	5598	PostgreSQL은 ROWID를 지원하지 않습니다	제한을 제외한 모든 발생
	5340	AWS DMS Schema Conversion은 함수를 변환할 수 없습니다.	제한을 제외한 모든 발생
	5071	PostgreSQL은 하위 쿼리에 대한 INSERT 문을 지원하지 않습니다	제한을 제외한 모든 발생
	5068	PostgreSQL은 하위 쿼리에 대한 DELETE 문을 지원하지 않습니다	제한을 제외한 모든 발생
	5065	PostgreSQL은 하위 쿼리에 대한 UPDATE 문을 지원하지 않습니다	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	5586	AWS DMS Schema Conversion은 NOCYCLE 절을 사용하여 쿼리를 변환할 수 없습니다.	제한을 제외한 모든 발생
	5351	AWS DMS Schema Conversion은 객체를 변환할 수 없습니다.	제한을 제외한 모든 발생
	5077	PostgreSQL은 SELECT 문에 대한 PIVOT 절을 지원하지 않습니다	제한을 제외한 모든 발생
	5126	PostgreSQL은 MODEL 문을 지원하지 않습니다	제한을 제외한 모든 발생
	5121	PostgreSQL은 FORALL 문을 지원하지 않습니다	제한을 제외한 모든 발생
	5141	AWS DMS Schema Conversion은 이러한 유형의 변환을 지원하지 않습니다.	제한을 제외한 모든 발생
	5142	AWS DMS Schema Conversion은 동일한 메서드의 중첩된 호출을 변환할 수 없습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	5245	PostgreSQL은 중첩된 테이블 열이 있는 뷰를 지원하지 않습니다	제한을 제외한 모든 발생
	5500	AWS DMS Schema Conversion은 데이터베이스 메일 전송을 변환할 수 없습니다.	제한을 제외한 모든 발생
	5501	AWS DMS Schema Conversion은 예약된 작업을 변환할 수 없습니다.	제한을 제외한 모든 발생
	5645	PostgreSQL은 여러 객체 테이블 대상에 대해 BULK COLLECT INTO 절을 지원하지 않습니다.	제한을 제외한 모든 발생
	5665	PostgreSQL은 PRAGMA AUTONOMOUS_TRANSACTION으로 정의된 컬렉션 데이터 형식을 지원하지 않습니다	제한을 제외한 모든 발생
	5637	PostgreSQL은 VARRAY의 VARRAY로 대량 수집을 지원하지 않습니다	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	5594	AWS DMS Schema Conversion은 날짜 시간 표현식을 변환할 수 없습니다.	제한을 제외한 모든 발생
	5622	AWS DMS Schema Conversion은 파라미터를 true로 설정하여 dbms_transaction.local_transaction_id 함수를 변환합니다.	제한을 제외한 모든 발생
	5643	PostgreSQL은 SELECT 문에서 멀티레벨 컬렉션 유형에 대해 BULK COLLECT INTO 절을 지원하지 않습니다.	제한을 제외한 모든 발생
	5649	PostgreSQL은 멀티레벨 컬렉션 유형에 대한 멀티셋 연산자를 지원하지 않습니다.	제한을 제외한 모든 발생
	5651	AWS DMS Schema Conversion은 파이프라인 테이블 함수를 변환할 수 없습니다.	제한을 제외한 모든 발생
	5793	AWS DMS Schema Conversion은 GRANT ALL 옵션을 사용하여 대기열을 생성합니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
Microsoft SQL Server 에서 Amazon RDS for PostgreSQL 로, Microsoft SQL Server에서 Amazon Aurora PostgreSQL로	5794	PostgreSQL은 대기열 모드를 기본적으로 활 성화로 설정합니다	제한을 제외한 모든 발 생
	5795	Amazon Simple Queue Service는 비활 성화 모드의 대기열을 지원하지 않습니다	제한을 제외한 모든 발 생
	7610	AWS DMS Schema Conversion은 지원되 지 않는 DDL 문을 변 환할 수 없습니다.	제한을 제외한 모든 발 생
	7622	AWS DMS Schema Conversion은 복잡한 인라인 함수를 사용하 여 DELETE 문을 변환 할 수 없습니다.	제한을 제외한 모든 발 생
	7624	AWS DMS Schema Conversion은 기본 키 가 없는 테이블의 인라 인 함수에서 DELETE 문을 변환할 수 없습니 다.	제한을 제외한 모든 발 생
	7626	AWS DMS Schema Conversion은 복잡한 인라인 함수를 사용하 여 UPDATE 문을 변환 할 수 없습니다.	제한을 제외한 모든 발 생

변환 방향	작업 항목	메시지	구문 요소
	7627	AWS DMS Schema Conversion은이 구문 요소를 변환할 수 없습니다.	제한을 제외한 모든 발생
	7628	PostgreSQL은 GOTO 문을 지원하지 않습니다	제한을 제외한 모든 발생
	7637	PostgreSQL은 글로벌 커서를 지원하지 않습니다	제한을 제외한 모든 발생
	7639	PostgreSQL은 동적 커서를 지원하지 않습니다	제한을 제외한 모든 발생
	7644	PostgreSQL은 %s 절을 지원하지 않습니다	제한을 제외한 모든 발생
	7645	PostgreSQL은 관련된 서버에서 패스스루 명령 실행을 지원하지 않습니다	제한을 제외한 모든 발생
	7653	PostgreSQL은 GROUP BY ROLLUP 절을 지원하지 않습니다	제한을 제외한 모든 발생
	7654	PostgreSQL은 GROUP BY CUBE 절을 지원하지 않습니다	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	7655	PostgreSQL은 GROUP BY GROUPING SETS 절을 지원하지 않습니다	제한을 제외한 모든 발생
	7672	PostgreSQL은 문자열을 실행하는 EXECUTE 문을 지원하지 않습니다	제한을 제외한 모든 발생
	7683	대상이 뷰, 구체화된 뷰 또는 외부 테이블인 경우 MERGE가 지원되지 않습니다	제한을 제외한 모든 발생
	7687	PostgreSQL은 CONTAINS 조건자를 지원하지 않습니다	제한을 제외한 모든 발생
	7688	PostgreSQL은 FREETEXT 조건자를 지원하지 않습니다	제한을 제외한 모든 발생
	7691	PostgreSQL은 WAITFOR TIME 기능을 지원하지 않습니다	제한을 제외한 모든 발생
	7695	PostgreSQL은 프로시저 호출을 변수로 지원하지 않습니다	제한을 제외한 모든 발생
	7696	AWS DMS Schema Conversion은 객체가 생성되지 않아 %s 객체를 변환할 수 없습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	7708	AWS DMS Schema Conversion은 지원되지 않는 %s 데이터 유형의 사용을 변환할 수 없습니다.	제한을 제외한 모든 발생
	7709	AWS DMS Schema Conversion은 대칭 키의 사용을 변환할 수 없습니다.	제한을 제외한 모든 발생
	7773	AWS DMS Schema Conversion은 날짜로 산술 연산을 변환할 수 없습니다.	제한을 제외한 모든 발생
	7774	AWS DMS Schema Conversion은 혼합된 피연산자 유형으로 산술 연산을 변환할 수 없습니다.	제한을 제외한 모든 발생
	7794	PostgreSQL은 사용자 정의 데이터 형식을 지원하지 않습니다	제한을 제외한 모든 발생
	7796	PostgreSQL은 UPDATE 문에서 TOP 절을 지원하지 않습니다	제한을 제외한 모든 발생
	7797	PostgreSQL은 UPDATE 문에서 OUTPUT 절의 DELETED 열 접두사를 지원하지 않습니다	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	7798	PostgreSQL은 DELETE 문에서 TOP 절을 지원하지 않습니다	제한을 제외한 모든 발생
	7799	PostgreSQL은 INSERT 연산자의 TOP 절을 지원하지 않습니다	제한을 제외한 모든 발생
	7804	PostgreSQL은 비트 배타적 OR 연산자를 지원하지 않습니다	제한을 제외한 모든 발생
	7805	PostgreSQL은 !<를 지원하지 않습니다	제한을 제외한 모든 발생
	7806	PostgreSQL은 !>(다음보다 크지 않음) 연산자를 지원하지 않습니다	제한을 제외한 모든 발생
	7811	PostgreSQL은 %s 함수를 지원하지 않습니다. AWS DMS 스키마 변환은 변환된 코드에서 지원되지 않는 함수를 건너뜁니다.	DDL을 제외한 제한을 제외한 모든 발생
	7816	PostgreSQL은 XML 데이터 유형에 대한 메서드를 지원하지 않습니다	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	7817	PostgreSQL은 SQL 쿼리에서 FOR XML PATH 옵션을 지원하지 않습니다	제한을 제외한 모든 발생
	7818	PostgreSQL은 이진 데이터 형식의 산술 연산을 지원하지 않습니다	제한을 제외한 모든 발생
	7819	PostgreSQL은 INSERT...EXECUTE 문을 지원하지 않습니다	제한을 제외한 모든 발생
	7820	PostgreSQL은 VALUE() 메서드를 지원하지 않습니다	제한을 제외한 모든 발생
	7824	RECURSIVE CTE는 MERGE 문에서 지원되지 않습니다	제한을 제외한 모든 발생
	7829	AWS DMS Schema Conversion은 UPDATE 문을 사용하여 변수 할당을 변환할 수 없습니다.	제한을 제외한 모든 발생
	7830	AWS DMS Schema Conversion은 CASE 피연산자를 사용하여 산술 연산을 변환할 수 없습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	7832	AWS DMS Schema Conversion은 뷰에서 INSTEAD OF 트리거를 변환할 수 없습니다.	제한을 제외한 모든 발생
	7833	AWS DMS Schema Conversion은 현재 컨텍스트에서 @@rowcount 함수를 변환할 수 없습니다.	제한을 제외한 모든 발생
	7836	PostgreSQL은 바이너리 데이터에 대한 쓰기 작업을 지원하지 않습니다	제한을 제외한 모든 발생
	7840	AWS DMS Schema Conversion은 데이터베이스 콘솔 명령문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	7904	AWS DMS Schema Conversion은 %s 시스템 객체를 변환할 수 없습니다.	제한을 제외한 모든 발생
	7905	PostgreSQL은 SELECT 문에 대한 PIVOT 절을 지원하지 않습니다	제한을 제외한 모든 발생
	7906	PostgreSQL은 SELECT 문에 대한 UNPIVOT 절을 지원하지 않습니다	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	7909	AWS DMS Schema Conversion은 UPDATE(열) 또는 COLUMNS_UPDATED 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	7916	AWS DMS Schema Conversion은 INSERT ON CONFLICT 문을 사용하여 MERGE 문을 에뮬레이션할 수 없습니다.	제한을 제외한 모든 발생
	7917	PostgreSQL은 %s 함수를 지원하지 않습니다	제한을 제외한 모든 발생
	7918	PostgreSQL은 테이블 값 함수를 지원하지 않습니다	제한을 제외한 모든 발생
	7919	PostgreSQL은 %s 지시문이 있는 FOR XML을 지원하지 않습니다	제한을 제외한 모든 발생
	7920	PostgreSQL은 FOR XML에서 EXPLICIT 모드를 지원하지 않습니다	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	7925	PostgreSQL은 OPENXML 플래그에 대한 백분율 문자를 지원하지 않습니다	제한을 제외한 모든 발생
	7927	PostgreSQL은 프라이머리 키가 없는 자체 참조 테이블에 대한 OUTER 조인을 지원하지 않습니다	제한을 제외한 모든 발생
	7929	AWS DMS Schema Conversion은 EXEC 문에서 INSERT를 변환할 수 없습니다.	제한을 제외한 모든 발생
	7939	AWS DMS Schema Conversion은 %s JSON 시스템 함수를 변환할 수 없습니다.	제한을 제외한 모든 발생
	7940	AWS DMS Schema Conversion은 OPENJSON 시스템 테이블 값 함수를 변환할 수 없습니다.	제한을 제외한 모든 발생
	7941	AWS DMS 열린 데이터 세트가 여러 개 있기 때문에 Schema Conversion이 모든 열린 데이터 세트를 변환할 수 없음	제한을 제외한 모든 발생
	9996	내부 변환기 오류 발생	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
SAP ASE(Sybase ASE)에서 Amazon RDS for PostgreSQL로, SAP ASE(Sybase ASE)에서 Amazon Aurora PostgreSQL로	3014	함수를 변환할 수 없음	제한을 제외한 모든 발생
	3016	PostgreSQL은 DML 연산자에서 TOP 옵션을 지원하지 않습니다.	제한을 제외한 모든 발생
	3021	산술 연산의 자동 마이그레이션을 수행할 수 없음	제한을 제외한 모든 발생
	3023	PostgreSQL은 이진 데이터 형식의 산술 연산을 지원하지 않습니다	제한을 제외한 모든 발생
	3025	날짜/시간 형식은 일치시킬 수 없습니다.	제한을 제외한 모든 발생
	3026	변수를 사용한 연산자 WAITFOR의 자동 변환은 지원되지 않습니다.	제한을 제외한 모든 발생
	3027	PostgreSQL은 WAITFOR TIME 기능을 지원하지 않습니다.	제한을 제외한 모든 발생
	3028	PostgreSQL은 명령과 함께 WAITFOR를 지원하지 않습니다.	제한을 제외한 모든 발생
	3061	시스템 객체를 변환할 수 없음	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	3064	PostgreSQL에서는 UPDATE 문의 FROM 절에서 대상 테이블을 반복해서는 안 됩니다.	제한을 제외한 모든 발생
	3065	FROM 절 및 OUTER JOIN에 자체 참조 테이블이 있는 DELETE 문은 자동으로 변환할 수 없습니다.	제한을 제외한 모든 발생
	3069	문을 변환할 수 없음	제한을 제외한 모든 발생
	3081	DMS SC는 지원되지 않는 DDL 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	3088	PostgreSQL은 글로벌 커서를 지원하지 않습니다	제한을 제외한 모든 발생
	3089	PostgreSQL은 동적 커서를 지원하지 않습니다	제한을 제외한 모든 발생
	3121	DMS SC는 지원되지 않는 데이터 유형의 사용을 변환할 수 없습니다.	제한을 제외한 모든 발생
	3122	DMS SC는 날짜로 산술 연산을 변환할 수 없습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	3123	DMS SC는 혼합된 피연산자 유형으로 산술 연산을 변환할 수 없습니다.	제한을 제외한 모든 발생
	3146	PostgreSQL은 비트 배타적 OR 연산자를 지원하지 않습니다	제한을 제외한 모든 발생
	3147	PostgreSQL은 !<(이상) 연산자를 지원하지 않습니다.	제한을 제외한 모든 발생
	3148	PostgreSQL은 !>(다음보다 크지 않음) 연산자를 지원하지 않습니다	제한을 제외한 모든 발생
	3150	DMS SC가 함수를 변환할 수 없음	제한을 제외한 모든 발생
	3156	PostgreSQL은 이진 데이터 형식의 산술 연산을 지원하지 않습니다	제한을 제외한 모든 발생
	3162	DMS SC는 UPDATE 문으로 변수 할당을 변환할 수 없습니다.	제한을 제외한 모든 발생
	3163	DMS SC는 CASE 피연산자를 사용하여 산술 연산을 변환할 수 없습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	3168	PostgreSQL은 바이너리 데이터에 대한 쓰기 작업을 지원하지 않습니다	제한을 제외한 모든 발생
	3172	DMS SC가 데이터베이스 콘솔 명령문을 변환할 수 없음	제한을 제외한 모든 발생
	3177	DMS SC가 시스템 객체를 변환할 수 없음	제한을 제외한 모든 발생
	3182	DMS SC는 UPDATE(열) 또는 COLUMNS_UPDATED 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	3190	DMS SC가 함수를 변환할 수 없음	제한을 제외한 모든 발생
	3191	PostgreSQL은 테이블 값 함수를 지원하지 않습니다	제한을 제외한 모든 발생
	9996	내부 변환기 오류 발생	제한을 제외한 모든 발생
Linux, UNIX 및 Windows용 IBM Db2(LUW)에서 Amazon RDS for PostgreSQL로, Linux, UNIX 및 Windows용 IBM Db2(LUW)에서 Amazon Aurora PostgreSQL로	4506	PostgreSQL은 최상위 수준에서 설정된 작업이 포함된 보기 정의를 지원하지 않습니다.	제한을 제외한 모든 발생
	4523	PostgreSQL은 쿼리에 삽입을 지원하지 않습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	4524	PostgreSQL은 삽입 문에서 계산하기 전에 값 반환을 지원하지 않습니다.	제한을 제외한 모든 발생
	4526	PostgreSQL은 %s 함수를 지원하지 않습니다.	제한을 제외한 모든 발생
	4527	PostgreSQL은 쿼리를 사용한 데이터 업데이트를 지원하지 않습니다.	제한을 제외한 모든 발생
	4528	PostgreSQL은 업데이트 문에서 계산하기 전에 값 반환을 지원하지 않습니다.	제한을 제외한 모든 발생
	4529	PostgreSQL은 업데이트 문에서 이전 값 반환을 지원하지 않습니다.	제한을 제외한 모든 발생
	4531	PostgreSQL은 INCLUDE 열을 지원하지 않습니다.	제한을 제외한 모든 발생
	4532	PostgreSQL은 삽입 문에서 INCLUDE 열을 지원하지 않습니다.	제한을 제외한 모든 발생
	4533	PostgreSQL은 업데이트 문에서 행의 수량 제한을 지원하지 않습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	4534	PostgreSQL은 쿼리를 사용하여 데이터로 삭제를 지원하지 않습니다.	제한을 제외한 모든 발생
	4536	PostgreSQL은 삭제문에서 행의 수량 제한을 지원하지 않습니다.	제한을 제외한 모든 발생
	4537	PostgreSQL은 기간 지정 절을 지원하지 않습니다.	제한을 제외한 모든 발생
	4538	PostgreSQL은 outer-table-reference를 지원하지 않습니다.	제한을 제외한 모든 발생
	4539	PostgreSQL은 continue-handler 절을 지원하지 않습니다.	제한을 제외한 모든 발생
	4541	PostgreSQL은 모듈을 지원하지 않습니다.	제한을 제외한 모든 발생
	4549	PostgreSQL은 `GOTO` 문을 지원하지 않습니다.	제한을 제외한 모든 발생
	4556	PostgreSQL 오류 코드 유형은 숫자가 아니며 숫자 유형 변수와 호환되지 않습니다.	제한을 제외한 모든 발생
	4558	PostgreSQL은 유형화된 뷰를 지원하지 않습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	4559	PostgreSQL은 `MERGE` 문을 지원하지 않습니다.	제한을 제외한 모든 발생
	4572	PostgreSQL은 유형 메서드를 사용하여 로행을 반환하는 함수를 지원하지 않습니다.	제한을 제외한 모든 발생
	4573	PostgreSQL은 메서드 호출을 지원하지 않습니다.	제한을 제외한 모든 발생
	4575	PostgreSQL은 스키마 이름이 없거나 스키마 이름이 - SESSION 인 CREATE GLOBAL TEMPORARY TABLE 을 지원하지 않습니다.	제한을 제외한 모든 발생
	4583	시스템 객체를 변환할 수 없음	제한을 제외한 모든 발생
	4597	PostgreSQL에는 UTL_MAIL 모듈과 유사한 기능이 없습니다.	제한을 제외한 모든 발생
	4598	PostgreSQL에는 UTL_SMTP 모듈과 유사한 기능이 없습니다.	제한을 제외한 모든 발생
	4605	DMS SC는 MERGE 문에서 지원되지 않는 절을 변환할 수 없습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	4606	DMS SC는 지원되지 않는 절이 포함된 MERGE 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	4612	PostgreSQL은 특수 등록 기능을 지원하지 않습니다.	제한을 제외한 모든 발생
	9996	내부 변환기 오류 발생	제한을 제외한 모든 발생
z/OS용 IBM Db2에서 Amazon RDS for PostgreSQL로, z/OS용 IBM Db2에서 Amazon Aurora PostgreSQL로	8507	PostgreSQL은 %s 함수를 지원하지 않습니다	제한을 제외한 모든 발생
	8519	FETCH 절은 지원되지 않습니다.	제한을 제외한 모든 발생
	8521	위치 삭제는 지원되지 않습니다.	제한을 제외한 모든 발생
	8542	PostgreSQL은 기본 네임스페이스를 지원하지 않습니다.	제한을 제외한 모든 발생
	8559	DMS SC는 지원되지 않는 절이 포함된 MERGE 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	8560	PostgreSQL은 RAISE 문에서 변수 또는 파라미터 사용을 지원하지 않습니다.	제한을 제외한 모든 발생

변환 방향	작업 항목	메시지	구문 요소
	8563	PostgreSQL은 z/OS용 Db2와 비교하여 간격 데이터 유형에 다른 형식을 사용합니다.	제한을 제외한 모든 발생
	8566	DMS SC는 커서에서 가져온 대량 및 역방향 데이터를 변환할 수 없습니다.	제한을 제외한 모든 발생
	8570	DMS SC는 FINAL TABLE 테이블 참조에서 MERGE 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	8581	DMS SC는 지원되지 않는 절이 포함된 UPDATE 문을 변환할 수 없습니다.	제한을 제외한 모든 발생
	9996	내부 변환기 오류 발생	제한을 제외한 모든 발생

제한 사항

생성형 AI 기능을 사용하여 데이터베이스 객체 변환에는 다음과 같은 제한 사항이 있습니다.

- 생성형 AI 변환을 지원하는 데이터베이스 엔드포인트는 AWS 콘솔에 표시되지 않습니다. 평가 보고서를 PDF 또는 CSV 파일로 내보내야만 볼 수 있습니다.
- 확률적 시스템인 생성형 AI 지원 스키마 변환은 모든 변환에서 100% 정확도를 달성하지 못할 수 있습니다. 또한 일정 기간 동안 동일한 SQL 문에 대해 다른 결과를 생성할 수 있습니다. 모든 변환 출력을 검토하고 검증해야 합니다.
- 다음과 같은 경우에는 생성형 AI 변환이 지원되지 않습니다.
 - 테이블의 DEFAULT 제약 조건
 - 함수 또는 프로시저 파라미터의 DEFAULT 값

- 테이블의 컴퓨팅 열
- TRIGGER
- 열 데이터 유형
- Dynamic SQL
- INDEX
- CONSTRAINT
- 소스 문이 여러 작업 항목으로 변환되고 하나 이상의 작업 항목이 생성형 AI를 사용하여 처리되는 경우 모든 작업 항목은 Oracle의 대상에서 하나의 작업 항목 5444 및 Microsoft SQL Server의 7744로 대체됩니다. 작업 항목 9997은 생성형 AI를 사용하여 처리된 후 저장되는 예외입니다.

 Warning

생성형 AI를 사용한 변환은 기본 변환보다 오래 걸립니다.

모든 AWS 계정에는 생성형 AI를 사용하여 변환할 수 있는 SQL 문 수를 제한하는 분당 할당량이 있습니다. 이 제한을 초과하는 문은 이후 몇 분 내에 처리를 위해 대기열에 저장됩니다. 할당량은 다음과 같습니다.

리전	분당 AWS 계정당 SQL 문
아시아 태평양(도쿄)(ap-northeast-1)	최대 24개의 문
아시아 태평양(오사카) (ap-northeast-3)	
아시아 태평양(시드니)(ap-southeast-2)	
캐나다(중부)(ca-central-1)	
유럽(스톡홀름)(eu-north-1)	
유럽(아일랜드)(eu-west-1)	
유럽(런던) (eu-west-2)	
유럽(파리)(eu-west-3)	

리전	분당 AWS 계정당 SQL 문
미국 동부(오하이오)(us-east-2)	최대 40개의 문
유럽(프랑크푸르트)(eu-central-1)	
미국 동부(버지니아 북부)(us-east-1)	
미국 서부(오레곤)(us-west-2)	

마이그레이션 프로젝트에 대한 스키마 변환 설정 지정

마이그레이션 프로젝트를 생성한 후 DMS Schema Conversion에서 변환 설정을 지정할 수 있습니다. 스키마 변환 설정을 구성하면 변환된 코드의 성능이 향상됩니다.

변환 설정을 편집하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택합니다. 스키마 변환을 선택한 다음, 스키마 변환 시작을 선택합니다.
4. 설정을 선택합니다. 설정 페이지가 열립니다.
5. 변환 섹션에서 설정을 변경합니다.
6. 적용을 선택한 다음, 스키마 변환을 선택합니다.

모든 변환 쌍에 대해 변환된 코드에서 작업 항목이 있는 설명 수를 제한할 수 있습니다. 변환된 코드의 설명 수를 제한하려면 마이그레이션 프로젝트에서 변환 설정을 여세요.

변환된 SQL 코드의 설명에서 작업 항목의 심각도 수준을 선택합니다. DMS Schema Conversion은 선택한 수준 또는 그보다 높은 수준의 심각도에 해당하는 작업 항목에 대해 변환된 코드에 설명을 추가합니다. 예를 들어, 변환된 코드의 설명 수를 최소화하려면 오류만을 선택하세요.

변환된 코드의 모든 작업 항목에 대한 설명을 포함하려면 모든 메시지를 선택합니다.

기타 변환 설정은 소스 및 대상 데이터베이스 쌍마다 다릅니다.

주제

- [Oracle에서 MySQL로 변환 설정 이해](#)
- [Oracle에서 PostgreSQL로 변환 설정 이해](#)
- [SQL Server에서 MySQL로 변환 설정 이해](#)
- [SQL Server에서 PostgreSQL로 변환 설정 이해](#)
- [PostgreSQL에서 MySQL로 변환 설정 이해](#)
- [LUW용 IBM Db2에서 Amazon RDS for PostgreSQL로의 변환 설정 이해](#)
- [z/OS용 IBM Db2에서 Amazon RDS for Db2로의 변환 설정 이해](#)
- [z/OS용 IBM Db2에서 Amazon RDS for PostgreSQL로의 변환 설정 이해](#)
- [SAP ASE\(Sybase ASE\)에서 PostgreSQL로의 변환 설정 이해](#)

Oracle에서 MySQL로 변환 설정 이해

DMS Schema Conversion의 경우, Oracle에서 MySQL로의 변환 설정에는 다음이 포함됩니다.

- 소스 Oracle 데이터베이스는 ROWID 의사 열을 사용할 수 있습니다. MySQL에서는 유사한 기능을 지원하지 않습니다. DMS Schema Conversion은 변환된 코드의 ROWID 의사 열을 에뮬레이션할 수 있습니다. 이렇게 하려면 행 ID 생성 옵션을 활성화하세요.

소스 Oracle 코드에서 ROWID 의사 열을 사용하지 않는 경우, 행 ID 생성 옵션을 비활성화하세요. 이 경우, 변환된 코드는 더 빠르게 작동합니다.

- 소스 Oracle 코드에는 MySQL이 지원하지 않는 파라미터가 있는 TO_CHAR, TO_DATE, TO_NUMBER 함수가 포함될 수 있습니다. 기본적으로 DMS Schema Conversion은 변환된 코드에서 이러한 파라미터의 사용을 에뮬레이션합니다.

소스 Oracle 코드에 MySQL에서 지원되지 않는 파라미터가 없는 경우, 네이티브 MySQL TO_CHAR, TO_DATE 및 TO_NUMBER 함수를 사용할 수 있습니다. 이 경우, 변환된 코드는 더 빠르게 작동합니다. 이렇게 하려면 다음 값을 선택합니다.

- 기본 MySQL TO_CHAR 함수 사용
- 기본 MySQL TO_DATE 함수 사용
- 기본 MySQL TO_NUMBER 함수 사용
- 데이터베이스와 애플리케이션은 서로 다른 시간대에 실행할 수 있습니다. 기본적으로 DMS Schema Conversion은 변환된 코드로 시간대를 에뮬레이션합니다. 하지만 데이터베이스와 애플리케이션이 동일한 시간대를 사용하는 경우에는 이 에뮬레이션이 필요하지 않습니다. 이 경우, 데이터베이스와 애플리케이션이 동일한 시간대를 사용하는 경우 변환된 코드의 성능 개선을 선택합니다.

Oracle에서 PostgreSQL로 변환 설정 이해

DMS Schema Conversion의 경우, Oracle에서 PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- 생성형 AI를 사용하여 해당 데이터베이스 객체를 변환하려면 생성형 AI 설정을 활성화합니다. 생성형 AI를 사용하여 성공적으로 변환된 객체는 "이 변환은 데이터의 패턴을 기반으로 예측을 생성하는 기계 학습 모델을 사용합니다"라는 작업 항목 5444로 명확하게 식별됩니다. 자세한 내용은 [the section called “생성형 AI를 사용하여 데이터베이스 객체 변환”](#) 단원을 참조하십시오.
- 최신 버전뿐만 아니라 지원되는 대상 데이터베이스의 다양한 버전에 대한 스키마를 변환할 수 있습니다. 버전을 지정하려면 대상 엔진 버전 설정을 사용합니다. 사용 가능한 변환 옵션은 선택한 엔진 버전에 따라 달라집니다. 현재 PostgreSQL 버전 14 및 15가 지원됩니다(설정이 동일함). 스키마 변환 중에이 설정에 지정된 버전과 다른 대상 엔진 버전을 사용하면 대상에 적용 작업을 수행할 때 호환성 문제와 오류가 발생할 수 있습니다.
- AWS DMS는 Oracle 구체화된 뷰를 PostgreSQL의 테이블 또는 구체화된 뷰로 변환할 수 있습니다. 구체화된 뷰의 경우, 소스 구체화된 뷰를 변환하는 방법을 선택합니다.
- 소스 Oracle 데이터베이스는 ROWID 의사 열을 사용할 수 있습니다. PostgreSQL에서는 유사한 기능을 지원하지 않습니다. DMS Schema Conversion은 bigint 또는 character varying 데이터 유형을 사용하여 변환된 코드로 ROWID 의사 열을 에뮬레이션할 수 있습니다. 이렇게 하려면 bigint 데이터 유형 사용을 선택하여 ROWID 의사 열을 에뮬레이션을 선택하거나 행 ID에 대해 문자 가변 데이터 유형을 사용하여 ROWID 의사 열을 에뮬레이션을 선택하세요.
- 정밀도와 규모에 따라 NUMBER 데이터 형식을 가장 적합한 숫자 데이터 유형으로 변환하려면 NUMBER 데이터 유형의 열에 최적화된 데이터 유형 매핑 사용을 선택할 수 있습니다.
- 소스 Oracle 코드에서 ROWID 의사 열을 사용하지 않는 경우, 생성 안 함을 선택합니다. 이 경우, 변환된 코드는 더 빠르게 작동합니다.
- 소스 Oracle 코드에는 PostgreSQL에서 지원하지 않는 파라미터가 있는 TO_CHAR, TO_DATE, TO_NUMBER 함수가 포함될 수 있습니다. 기본적으로 DMS Schema Conversion은 변환된 코드에서 이러한 파라미터의 사용을 에뮬레이션합니다.

PostgreSQL에서 지원되지 않는 파라미터가 소스 Oracle 코드에 없는 경우, 기본 PostgreSQL TO_CHAR, TO_DATE 및 TO_NUMBER 함수를 사용할 수 있습니다. 이 경우, 변환된 코드는 더 빠르게 작동합니다. 이렇게 하려면 다음 값을 선택합니다.

- 기본 PostgreSQL TO_CHAR 함수 사용
- 기본 PostgreSQL TO_DATE 함수 사용
- 기본 PostgreSQL TO_NUMBER 함수 사용

- 데이터베이스와 애플리케이션은 서로 다른 시간대에 실행할 수 있습니다. 기본적으로 DMS Schema Conversion은 변환된 코드로 시간대를 에뮬레이션합니다. 하지만 데이터베이스와 애플리케이션이 동일한 시간대를 사용하는 경우에는 이 에뮬레이션이 필요하지 않습니다. 이 경우, 데이터베이스와 애플리케이션이 동일한 시간대를 사용하는 경우 변환된 코드의 성능 개선을 선택합니다.
- 변환된 코드의 시퀀스를 계속 사용하려면 변환된 시퀀스를 소스 측에서 생성된 마지막 값으로 채우기를 선택합니다.
- 경우에 따라 차이는 있겠지만 소스 Oracle 데이터베이스는 NUMBER 데이터 유형의 프라이머리 키 또는 외래 키 열에 정수 값만 저장할 수 있습니다. 이러한 경우는 이러한 열을 BIGINT 데이터 형식으로 변환할 수 AWS DMS 있습니다. 이 방식을 적용하면 변환된 코드의 성능이 향상됩니다. 이렇게 하려면 NUMBER 데이터 유형의 프라이머리 키 및 외래 키 열을 BIGINT 데이터 유형으로 변환을 선택합니다. 데이터 손실을 방지하려면 소스에서 이러한 열에 부동 소수점 값이 포함되지 않도록 해야 합니다.
- 소스 코드에서 비활성화된 트리거와 제약 조건을 건너뛰려면 활성 트리거 및 제약 조건만 변환을 선택합니다.
- DMS Schema Conversion을 사용하여 동적 SQL이라고 하는 문자열 변수를 변환할 수 있습니다. 데이터베이스 코드는 이러한 문자열 변수의 값을 변경할 수 있습니다. 가 AWS DMS 항상 이 문자열 변수의 최신 값을 변환하도록 하려면 호출된 루틴에서 생성된 동적 SQL 코드 변환을 선택합니다.
- PostgreSQL 버전 10 및 이전 버전은 프로시저를 지원하지 않습니다. PostgreSQL에서 프로시저를 사용하는 데 익숙하지 않은 경우는 Oracle 프로시저를 PostgreSQL 함수로 변환할 AWS DMS 수 있습니다. 이렇게 하려면 프로시저를 함수로 변환을 선택합니다.
- 확장 팩에 특정 함수를 추가하면 발생한 작업 항목에 관한 추가 정보를 볼 수 있습니다. 이렇게 하려면 사용자 정의 예외를 발생시키는 확장 팩 함수 추가를 선택합니다. 그런 다음, 사용자 정의 예외가 발생하도록 심각도 수준을 선택합니다. 소스 데이터베이스 객체를 변환한 후에는 확장 팩 스키마를 적용해야 합니다. 확장 팩에 관한 자세한 내용은 [확장 팩 사용](#)을 참조하세요.
- 소스 Oracle 데이터베이스에는 자동으로 생성된 이름과 함께 제약 조건이 포함될 수 있습니다. 소스 코드에서 이러한 이름을 사용하는 경우, 시스템 생성 제약 조건의 이름 유지를 선택해야 합니다. 소스 코드에서 이러한 제약 조건을 사용하지만 이름을 사용하지 않는 경우 이 옵션을 선택 해제하여 변환 속도를 높이세요.
- 소스 및 대상 데이터베이스가 서로 다른 시간대에서 실행되는 경우, SYSDATE 내장 Oracle 함수를 에뮬레이션하는 함수는 소스 함수와 비교해 다른 값을 반환합니다. 소스 함수와 대상 함수가 동일한 값을 반환하도록 하려면 소스 데이터베이스의 시간대 설정을 선택합니다.
- orafce 확장의 함수를 변환된 코드에 사용할 수 있습니다. 이렇게 하려면 Oracle 기본 제공 루틴에서 사용할 함수를 선택하세요. orafce에 관한 자세한 내용은 GitHub에서 [orafce](#)를 참조하세요.

SQL Server에서 MySQL로 변환 설정 이해

DMS Schema Conversion의 경우, SQL Server에서 MySQL로의 변환 설정에는 다음이 포함됩니다.

- 소스 SQL Server 데이터베이스는 EXEC의 출력을 테이블에 저장할 수 있습니다. DMS Schema Conversion은 이 기능을 에뮬레이션하기 위한 임시 테이블과 추가 프로시저를 생성합니다. 이 에뮬레이션을 사용하려면 열린 데이터세트를 처리할 추가 루틴 생성을 선택합니다.

SQL Server에서 PostgreSQL로 변환 설정 이해

DMS Schema Conversion의 경우, SQL Server에서 PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- 생성형 AI를 사용하여 해당 데이터베이스 객체를 변환하려면 생성형 AI 설정을 활성화합니다. 생성형 AI를 사용하여 성공적으로 변환된 객체는 "이 변환은 데이터의 패턴을 기반으로 예측을 생성하는 기계 학습 모델을 사용합니다"라는 작업 항목 7744로 명확하게 식별됩니다. 자세한 내용은 [the section called “생성형 AI를 사용하여 데이터베이스 객체 변환”](#) 단원을 참조하십시오.
- SQL Server에서는 서로 다른 테이블에서 같은 이름의 인덱스를 사용할 수 있습니다. 하지만 PostgreSQL에서는 스키마에서 사용하는 모든 인덱스 이름이 고유해야 합니다. DMS Schema Conversion이 모든 인덱스에 대해 고유한 이름을 생성하도록 하려면 인덱스에 고유한 이름 생성을 선택합니다.
- PostgreSQL 버전 10 및 이전 버전은 프로시저를 지원하지 않습니다. PostgreSQL에서 프로시저를 사용하는 데 익숙하지 않은 경우는 SQL Server 프로시저를 PostgreSQL 함수로 변환할 AWS DMS 수 있습니다. 이렇게 하려면 프로시저를 함수로 변환을 선택합니다.
- 소스 SQL Server 데이터베이스는 EXEC의 출력을 테이블에 저장할 수 있습니다. DMS Schema Conversion은 이 기능을 에뮬레이션하기 위한 임시 테이블과 추가 프로시저를 생성합니다. 이 에뮬레이션을 사용하려면 열린 데이터세트를 처리할 추가 루틴 생성을 선택합니다.
- 변환된 코드의 스키마 이름에 사용할 템플릿을 정의할 수 있습니다. 스키마 이름에서 다음 옵션 중 하나를 선택합니다.
 - DB – PostgreSQL에서 SQL Server 데이터베이스 이름을 스키마 이름으로 사용합니다.
 - SCHEMA – PostgreSQL에서 SQL Server 스키마 이름을 스키마 이름으로 사용합니다.
 - DB_SCHEMA – PostgreSQL에서 SQL Server 데이터베이스와 스키마 이름의 조합을 스키마 이름으로 사용합니다.
- 소스 피연산자의 이름에 문자 대소문자를 유지할 수 있습니다. 피연산자 이름을 소문자로 변환되는 것을 방지하려면 대소문자를 구분하는 연산에서는 소문자로 변환하지 마세요를 선택합니다. 이 옵션은 소스 데이터베이스에서 대소문자 구분 기능이 활성화된 경우에만 적용됩니다.

- 소스 데이터베이스의 파라미터 이름을 유지할 수 있습니다. DMS Schema Conversion은 변환된 코드의 파라미터 이름에 큰따옴표를 추가할 수 있습니다. 이렇게 하려면 원래 파라미터 이름 유지를 선택합니다.
- 소스 데이터베이스에서 루틴 파라미터 길이를 유지할 수 있습니다. DMS Schema Conversion은 도메인을 생성하고 이를 사용하여 루틴 파라미터 길이를 지정합니다. 이렇게 하려면 파라미터 길이 유지를 선택합니다.
- 지원되지 않는 기본 제공 객체를 스텝 객체로 변환하려면 지원되지 않는 기본 제공 객체를 스텝 객체로 변환 설정을 활성화합니다.
- 활성화되면 DMS SC는 지원되지 않는 기본 제공 객체를 대상 데이터베이스의 해당 스텝 객체로 바꿉니다. 이 기능은 일반적으로 마이그레이션 문제 7811 또는 7904로 묶이는 코드 섹션을 변환합니다. 프로시저의 경우 소스 기본 제공 객체 PROCEDURE를, 뷰나 테이블의 경우 VIEW를 기준으로 스텝 객체를 생성합니다.

지원되지 않는 객체를 직접 호출하여 소스 데이터베이스 객체를 변환하면 스텝 객체 및 마이그레이션 문제 7822가 호출됩니다.

별도의 스키마에서 스텝 객체 생성 옵션을 활성화하여 별도의 스키마에서 스텝 객체를 생성하도록 선택할 수 있습니다. 선택하면 스텝 객체가 대상 데이터베이스에 `aws_sqlserver_stub` 이름이 지정된 특수 스키마에 생성됩니다. 선택하지 않으면 직접 호출 객체와 동일한 스키마에 생성됩니다.

- 스텝 루틴은 원래 기본 제공의 정규화된 이름을 기반으로 이름이 지정됩니다. 스텝 보기의 경우 `system_schema_name$builtin_view_name` 이름 지정 규칙에 시스템 스키마 이름이 포함됩니다.

다시 변환하는 동안 DMS SC는 대상 데이터베이스의 기존 스텝 루틴을 확인합니다. 동일한 이름과 입력 파라미터를 가진 루틴이 이미 존재하는 경우 덮어쓰지 않습니다.

변환 후 필요에 따라 스텝 루틴에 대한 사용자 지정 코드를 검토하고 구현합니다.

- CITEXT DMS Schema Conversion의 모든 문자열 데이터 형식 설정에 대한 데이터 형식에는 다음이 포함됩니다.
- SQL Server에서 PostgreSQL로 변환할 때 대소문자를 구분하지 않는 문자열 작업에 CITEXT 데이터 유형을 사용하려면 모든 문자열 데이터 유형에 CITEXT 사용 설정을 활성화합니다. 이 옵션은 대소문자를 구분하지 않는 SQL Server에서 대소문자를 구분하는 PostgreSQL 환경으로 마이그레이션할 때 일관된 동작을 유지하는 데 도움이 됩니다.

- 활성화되면 DMS SC는 소스 SQL Server 데이터베이스의 모든 관련 문자열 데이터 유형을 PostgreSQL의 CITEXT로 변환합니다. 따라서 조건에서 명시적 LOWER() 함수 호출이 필요하지 않으며 조건부 작업에서 문자열 표현식을 CITEXT로 자동 캐스팅합니다.
- SQL Server 인스턴스가 대/소문자를 구분하는지 확인하려면 다음 쿼리를 실행합니다.

```
SELECT SERVERPROPERTY('COLLATION');
```

'CI'가 포함된 결과는 대소문자를 구분하지 않고 'CS'는 대소문자를 구분합니다.

- 서버, 데이터베이스 또는 열 수준에서 명시적으로 대/소문자를 구분하는 데이터 정렬 설정이 사용되는 시나리오에서는 CITEXT 변환이 적용되지 않을 수 있습니다.

이 기능을 사용하려면 CITEXT 모듈이 대상 PostgreSQL 데이터베이스에 설치되어 있고 사용 가능한지 확인합니다.

- CITEXT 데이터 유형 변환을 사용할 때는 다음 모범 사례를 고려하세요.
 - PostgreSQL에서 일관된 동작을 유지하기 위해 대/소문자를 구분하지 않는 SQL Server에서 마이그레이션할 때 이 기능을 활성화합니다.
 - 애플리케이션 코드를 검토하여 대소문자를 구분하는 문자열 작업에 의존하지 않는지 확인합니다.
 - 마이그레이션 후 애플리케이션을 철저히 테스트하여 대소문자를 구분하지 않는 동작이 예상대로 유지되는지 확인합니다.

PostgreSQL에서 MySQL로 변환 설정 이해

DMS Schema Conversion에서 PostgreSQL에서 MySQL로의 변환 설정에는 다음이 포함됩니다.

- 변환된 SQL 코드의 주석: 이 설정은 선택한 심각도 이상의 작업 항목에 대한 변환된 코드에 주석을 추가합니다. 이 설정은 다음과 같은 값을 지원합니다.
 - 오류만
 - 오류 및 경고
 - 모든 메시지

LUW용 IBM Db2에서 Amazon RDS for PostgreSQL로의 변환 설정 이해

Note

AWS Management Console은 Amazon RDS for PostgreSQL을 대상 데이터 공급자로 사용하는 소스 데이터 공급자로 IBM Db2 for LUW를 사용하는 DMS Schema Conversion에서 마이그레이션 프로젝트 생성을 지원하지 않습니다. 대신 AWS CLI 또는 DMS Schema Conversion API를 사용합니다.

DMS Schema Conversion의 Linux, UNIX 및 Windows용 IBM Db2에서 Amazon RDS for PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- 변환된 SQL 코드의 주석: 이 설정은 선택한 심각도 이상의 작업 항목에 대한 변환된 코드에 주석을 추가합니다. 이 설정은 다음과 같은 값을 지원합니다.
 - 오류만
 - 오류 및 경고
 - 모든 메시지

z/OS용 IBM Db2에서 Amazon RDS for Db2로의 변환 설정 이해

DMS Schema Conversion의 IBM for z/OS에서 Amazon RDS for DB2로의 변환 설정에는 다음이 포함됩니다.

- 변환된 SQL 코드의 주석: 이 설정은 선택한 심각도 이상의 작업 항목에 대한 변환된 코드에 주석을 추가합니다. 이 설정은 다음과 같은 값을 지원합니다.
 - 오류만
 - 오류 및 경고
 - 모든 메시지

z/OS용 IBM Db2에서 Amazon RDS for PostgreSQL로의 변환 설정 이해

Note

AWS Management Console은 z/OS용 IBM Db2를 소스 데이터 공급자로 사용하고 Amazon RDS for PostgreSQL을 대상 데이터 공급자로 사용하는 DMS Schema Conversion에서 마이

그레이션 프로젝트 생성을 지원하지 않습니다. 대신 AWS CLI 또는 DMS Schema Conversion API를 사용합니다.

DMS Schema Conversion의 z/OS용 IBM Db2에서 Amazon RDS for PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- 변환된 SQL 코드의 주석: 이 설정은 선택한 심각도 이상의 작업 항목에 대한 변환된 코드에 주석을 추가합니다. 이 설정은 다음과 같은 값을 지원합니다.
 - 오류만
 - 오류 및 경고
 - 모든 메시지

SAP ASE(Sybase ASE)에서 PostgreSQL로의 변환 설정 이해

DMS Schema Conversion의 SAP ASE(Sybase ASE)에서 PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- 생성형 AI를 사용하여 해당 데이터베이스 객체를 변환하려면 생성형 AI 설정을 활성화합니다. 생성형 AI를 사용하여 성공적으로 변환된 객체는 "이 변환은 데이터의 패턴을 기반으로 예측을 생성하는 기계 학습 모델을 사용합니다"라는 Action Item 3077로 명확하게 식별됩니다. 자세한 내용은 [생성형 AI를 사용하여 데이터베이스 객체 변환](#) 단원을 참조하십시오.
- 변환된 SQL 코드의 주석: 이 설정은 선택한 심각도 이상의 작업 항목에 대한 변환된 코드에 주석을 추가합니다. 이 설정은 다음과 같은 값을 지원합니다.
 - 오류만
 - 오류 및 경고
 - 모든 메시지
- 변환된 코드의 스키마 이름에 사용할 템플릿을 정의할 수 있습니다. 스키마 이름에서 다음 옵션 중 하나를 선택합니다.
 - DB - PostgreSQL에서 SAP ASE(Sybase ASE) 데이터베이스 이름을 스키마 이름으로 사용합니다.
 - SCHEMA - PostgreSQL에서 SAP ASE(Sybase ASE) 스키마 이름을 스키마 이름으로 사용합니다.
 - DB_SCHEMA - SAP ASE(Sybase ASE) 데이터베이스와 스키마 이름의 조합을 스키마 이름으로 사용합니다.

- 소스 데이터베이스에서 객체 이름의 정확한 대/소문자를 유지할 수 있습니다. 이렇게 하려면 소스 데이터베이스 객체 이름을 대/소문자를 구분하는 것으로 처리를 선택합니다. 이 옵션을 끄면 모든 데이터베이스 객체 이름이 소문자로 변환됩니다.
- 대소문자를 구분하지 않는 작업 중에 피연산자를 소문자로 캐스팅하지 않아도 됩니다. DMS Schema Conversion은 변환된 코드의 피연산자에 LOWER 함수를 적용하지 않습니다. 이렇게 하려면 대소문자를 구분하지 않는 작업을 위해 피연산자를 소문자로 캐스팅하지 않음을 선택합니다. 이 옵션을 선택하지 않으면(기본값) DMS Schema Conversion은 대소문자를 구분하지 않는 비교를 수행하기 전에 LOWER 함수를 자동으로 적용하여 피연산자를 소문자로 변환합니다.
- SAP ASE(Sybase ASE)에서는 여러 테이블에서 이름이 동일한 인덱스를 생성할 수 있습니다. PostgreSQL을 사용하려면 스키마 내의 모든 인덱스 이름이 고유해야 합니다. AWS Schema Conversion이 마이그레이션 중에 고유한 인덱스 이름을 생성하도록 하려면 인덱스의 고유 이름 생성을 선택합니다. 이 옵션은 인덱스 이름에 접두사를 자동으로 추가하여 대상 PostgreSQL 데이터베이스의 이름 충돌을 방지합니다.

DMS Schema Conversion에서 데이터베이스 스키마 새로 고침

마이그레이션 프로젝트를 생성한 후 DMS Schema Conversion은 소스 및 대상 스키마에 관한 정보가 이 프로젝트에 저장합니다. DMS Schema Conversion은 데이터베이스 트리에서 노드를 선택하는 경우와 같이 필요한 경우에만 지연 로딩을 사용하여 메타데이터를 로드합니다. 즉시 로딩을 사용하면 스키마 정보를 더 빨리 로드할 수 있습니다. 이렇게 하려면 스키마를 선택한 다음, 작업에서 메타데이터 로드를 선택합니다.

객체를 마이그레이션 프로젝트에 자동 또는 수동으로 로드한 후에는 DMS Schema Conversion에서 지연 로딩을 다시 사용하지 않습니다. 따라서 데이터베이스의 테이블 및 프로시저와 같은 객체를 변경할 때는 마이그레이션 프로젝트에서 해당 객체에 대해 새로 고침을 실행해야 합니다.

데이터베이스로부터 스키마를 새로 고치려면 새로 고침을 실행할 해당 객체를 선택한 후 작업에서 데이터베이스에서 새로 고침을 선택합니다. 소스 및 대상 데이터베이스 스키마의 데이터베이스 객체는 다음과 같이 새로 고칠 수 있습니다.

- 소스 – 소스 데이터베이스 스키마를 업데이트하는 경우, 데이터베이스에서 새로 고침을 선택하여 프로젝트의 스키마를 소스 데이터베이스의 최신 스키마로 바꾸세요.
- 대상 – 대상 데이터베이스의 스키마를 업데이트하면 DMS Schema Conversion이 프로젝트의 스키마를 대상 데이터베이스의 최신 스키마로 바꿉니다. DMS Schema Conversion은 변환된 코드를 대상 데이터베이스의 코드로 바꿉니다. 데이터베이스에서 새로 고침을 선택하기 전에 변환된 코드를 대상 데이터베이스에 적용했는지 확인하세요. 그렇지 않으면 소스 데이터베이스 스키마를 다시 변환하세요.

변환된 코드를 DMS Schema Conversion에 저장 및 적용

DMS Schema Conversion은 소스 데이터베이스 객체를 변환한 후 변환된 코드를 대상 데이터베이스에 즉시 적용하지 않습니다. 그 대신 DMS Schema Conversion은 대상 데이터베이스에 적용할 준비가 될 때까지 변환된 코드를 프로젝트에 저장합니다.

변환된 코드를 적용하기 전에 소스 데이터베이스 코드를 업데이트하고 업데이트된 객체를 다시 변환하여 기존 작업 항목을 처리할 수 있습니다. DMS Schema Conversion에서 자동으로 변환할 수 없는 항목에 관한 자세한 내용은 [DMS Schema Conversion을 사용하여 데이터베이스 마이그레이션 평가 보고서 생성](#)을 참조하세요. DMS Schema Conversion에 대한 마이그레이션 프로젝트의 소스 데이터베이스 객체를 새로 고치는 방법에 관한 자세한 내용은 [데이터베이스 스키마 새로 고침](#)을 참조하세요.

변환된 코드를 DMS Schema Conversion의 데이터베이스에 직접 적용하는 대신, 이 코드를 SQL 스크립트로 파일에 저장할 수 있습니다. 이러한 SQL 스크립트를 검토하고 필요하다면 편집한 후 대상 데이터베이스에 해당 SQL 스크립트를 수동으로 적용할 수 있습니다.

변환된 코드를 SQL 파일에 저장

변환된 스키마를 텍스트 파일에 SQL 스크립트로 저장할 수 있습니다. 변환된 코드를 수정하여 DMS Schema Conversion에서 자동으로 변환할 수 없는 작업 항목을 처리할 수 있습니다. 그런 다음, 대상 데이터베이스에서 업데이트된 SQL 스크립트를 실행하여 변환된 코드를 대상 데이터베이스에 적용할 수 있습니다.

변환된 스키마를 SQL 스크립트로 저장하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음, 스키마 변환을 선택합니다.
4. 스키마 변환 시작을 선택합니다. 스키마 변환 페이지가 열립니다.
5. 오른쪽 창에서 대상 데이터베이스 스키마를 선택하거나 저장하려는 변환된 객체를 선택합니다. DMS Schema Conversion이 상위 노드 이름을 파란색으로 강조 표시하고 대상 데이터베이스의 작업 메뉴를 활성화하는지 확인하세요.
6. 작업에서 SQL로 저장을 선택합니다. 저장 대화 상자가 나타납니다.
7. SQL로 저장을 선택해 선택을 확인합니다.

DMS Schema Conversion은 SQL 파일로 아카이브를 생성하고 이 아카이브를 Amazon S3 버킷에 저장합니다.

8. (선택 사항) 인스턴스 프로파일에서 스키마 변환 설정을 편집하여 아카이브의 S3 버킷을 변경합니다.
9. S3 버킷에서 SQL 스크립트를 엽니다.

변환된 코드 적용

변환된 코드를 대상 데이터베이스에 적용할 준비가 되면 프로젝트의 오른쪽 창에서 데이터베이스 객체를 선택합니다. 전체 데이터베이스 스키마 또는 선택한 데이터베이스 스키마 객체에 변경 내용을 적용할 수 있습니다.

데이터베이스 객체를 선택하면 DMS Schema Conversion에서는 선택한 노드 또는 상위 노드의 이름이 파란색으로 강조 표시됩니다. 그러면 작업 메뉴가 활성화됩니다. 작업에서 변경 사항 적용을 선택합니다. 화면에 표시되는 대화 상자에서 적용을 선택하여 선택을 확인하고 변환된 코드를 대상 데이터베이스에 적용합니다.

확장 팩 스키마 적용

변환된 스키마를 대상 데이터베이스에 맨 처음 적용하는 경우 DMS Schema Conversion에서 확장 팩 스키마도 적용할 수 있습니다. 확장 팩 스키마는 대상 데이터베이스의 변환된 코드를 실행하는 데 필요한 소스 데이터베이스의 시스템 함수를 에뮬레이션합니다. 변환된 코드에서 확장 팩의 함수를 사용하는 경우, 확장 팩 스키마를 적용해야 합니다.

확장 팩을 대상 데이터베이스에 수동으로 적용하려면 작업에 대해 변경 사항 적용을 선택합니다. 화면에 표시되는 대화 상자에서 확인을 선택하여 대상 데이터베이스에 확장 팩을 적용합니다.

변환된 코드에서 예상치 못한 결과가 발생하지 않도록 확장 팩 스키마를 수정하지 않는 것이 좋습니다.

자세한 내용은 [DMS Schema Conversion에서 확장 팩 사용](#) 단원을 참조하십시오.

Java 애플리케이션의 임베디드 SQL 변환

AWS DMS 및 DMS Schema Conversion을 사용하여 데이터베이스를 마이그레이션하는 경우 대상 데이터베이스와 호환되도록 애플리케이션의 임베디드 SQL을 변환해야 할 수 있습니다. 수동으로 변환하는 대신 IDE에서 Amazon Q를 사용하여 변환을 자동화할 수 있습니다. Amazon Q는 DMS 스키마 변환의 메타데이터를 사용하여 애플리케이션에 포함된 SQL을 대상 데이터베이스와 호환되는 버전으로 변환합니다. Amazon Q는 애플리케이션에서 Oracle SQL 코드를 감지하여 PostgreSQL 구문으로 변환합니다. 자세한 정보는, [Amazon Q Developer를 사용하여 Java 애플리케이션에 임베딩된 SQL 변환](#)을 참조하세요.

Java 애플리케이션에서 임베디드 SQL 변환을 위한 메타데이터 파일 다운로드

1. 변환을 완료하면 프로젝트를 닫고 프로젝트가 저장된 S3 버킷으로 이동합니다.
2. 폴더를 열고 프로젝트 폴더("sct-project")를 찾습니다.
3. 프로젝트 폴더 내에서 객체를 다운로드합니다. 이 파일은 zip 파일입니다.
4. Amazon Q Developer를 사용하여 Java [애플리케이션에서 임베디드 SQL 변환](#)을 통해 다운로드한 zip 파일을 변환에 대한 입력으로 사용합니다.

DMS Schema Conversion에서 확장 팩 사용

DMS Schema Conversion의 확장 팩은 대상 데이터베이스에서 지원되지 않는 소스 데이터베이스 함수를 에뮬레이션하는 추가 모듈입니다. 확장 팩을 사용하여 변환된 코드가 소스 코드와 동일한 결과를 생성하는지 확인하십시오. 확장 팩을 설치하기 전에 데이터베이스 스키마를 변환하세요.

확장 팩마다 데이터베이스 스키마가 포함되어 있습니다. 이 스키마에는 소스 데이터베이스를 출처로 한 특정 온라인 트랜잭션 처리(OLTP) 객체 또는 지원되지 않는 내장 함수를 에뮬레이션하기 위한 SQL 함수, 프로시저, 테이블 및 뷰가 포함됩니다.

소스 데이터베이스를 변환할 때 DMS Schema Conversion은 대상 데이터베이스에 추가 스키마를 추가합니다. 이 스키마는 대상 데이터베이스에서 변환된 코드를 실행하는 데 필요한 소스 데이터베이스의 SQL 시스템 함수를 구현합니다. 이 추가 스키마를 일컬어 확장 팩 스키마라고 합니다.

확장 팩 스키마는 소스 데이터베이스에 따라 다음과 같이 이름이 지정됩니다.

- Microsoft SQL Server - aws_sqlserver_ext
- Oracle – aws_oracle_ext

다음 두 가지 방법으로 확장 팩을 적용할 수 있습니다.

- DMS Schema Conversion은 변환된 코드를 적용할 때 확장 팩을 자동으로 적용할 수 있습니다. DMS Schema Conversion은 다른 모든 스키마 객체를 적용하기 전에 확장 팩을 적용합니다.
- 확장 팩을 수동으로 적용할 수 있습니다. 이렇게 하려면 대상 데이터베이스 트리에서 확장 팩 스키마를 선택한 후 적용, 확장 팩 적용을 차례로 선택합니다.

AWS DMS Schema Conversion 및 Common Studio Framework(CSF)를 위한 API 매핑에 대한 IAM 작업

DMS 스키마 변환 및 Common Studio Framework에 대한 액세스 제어를 설정하고 IAM 권한 정책을 작성할 때 API 작업이 IAM 권한에 매핑되는 방법을 이해하는 것이 중요합니다. 일부 작업은 두 인터페이스에서 동일한 이름을 공유하지만 다른 작업은 크게 다릅니다.

다음 표에는 API 작업과 IAM 작업 간의 올바른 매핑이 나열되어 있습니다.

API에서 IAM으로 매핑

서비스	API	IAM
Common Studio Framework (CSF)	CreateMigrationProject	dms:CreateMigrationProject
	DeleteMigrationProject	dms>DeleteMigrationProject
	ModifyMigrationProject	dms:UpdateMigrationProject
	DescribeMigrationProjects	dms:ListMigrationProjects
	CreateInstanceProfile	dms:CreateInstanceProfile
	DeleteInstanceProfile	dms>DeleteInstanceProfile
	ModifyInstanceProfile	dms:UpdateInstanceProfile
	DescribeInstanceProfiles	dms:ListInstanceProfiles
	CreateDataProvider	dms:CreateDataProvider
	DeleteDataProvider	dms>DeleteDataProvider
DMS 스키마 변환	ModifyDataProvider	dms:UpdateDataProvider
	DescribeDataProviders	dms:ListDataProviders
	ModifyConversionConfiguration	dms:dms.UpdateConversionConfiguration

서비스	API	IAM
	DescribeConversionConfiguration	dms:DescribeConversionConfiguration
	StartMetadataModelImport	dms:StartMetadataModelImport
	ExportMetadataModelAssessment	dms:ExportMetadataModelAssessment
	StartMetadataModelConversion	dms:StartMetadataModelConversion
	StartMetadataModelExportAsScript	dms:StartMetadataModelExportAsScripts
	StartMetadataModelExportToTarget	dms:StartMetadataModelExportToTarget
	StartExtensionPackAssociation	dms:AssociateExtensionPack
	DescribeMetadataModelConversions	dms:ListMetadataModelConversions
	DescribeMetadataModelExportsToTarget	dms:ListMetadataModelExports
	DescribeMetadataModelExportsAsScript	dms:ListMetadataModelExports
	DescribeMetadataModelImports	dms:DescribeMetadataModelImports
	DescribeMetadataModelAssessments	dms:ListMetadataModelAssessments
	DescribeExtensionPackAssociations	dms:ListExtensionPacks

를 사용하여 데이터베이스를 Amazon RDS에 상응하는 데이터베이스로 마이그레이션 AWS DMS

AWS Database Migration Service (AWS DMS)의 동종 데이터 마이그레이션은 자체 관리형 온프레미스 데이터베이스를 Amazon Relational Database Service(RDS)에 상응하는 데이터베이스로 마이그레이션하는 것을 간소화합니다. 예를 들어, 동종 데이터 마이그레이션을 사용하여 온프레미스 PostgreSQL 데이터베이스를 Amazon RDS for PostgreSQL 또는 Aurora PostgreSQL로 마이그레이션할 수 있습니다. 동종 데이터 마이그레이션의 경우는 기본 데이터베이스 도구를 AWS DMS 사용하여 쉽고 성능이 뛰어난 like-to-like 제공합니다.

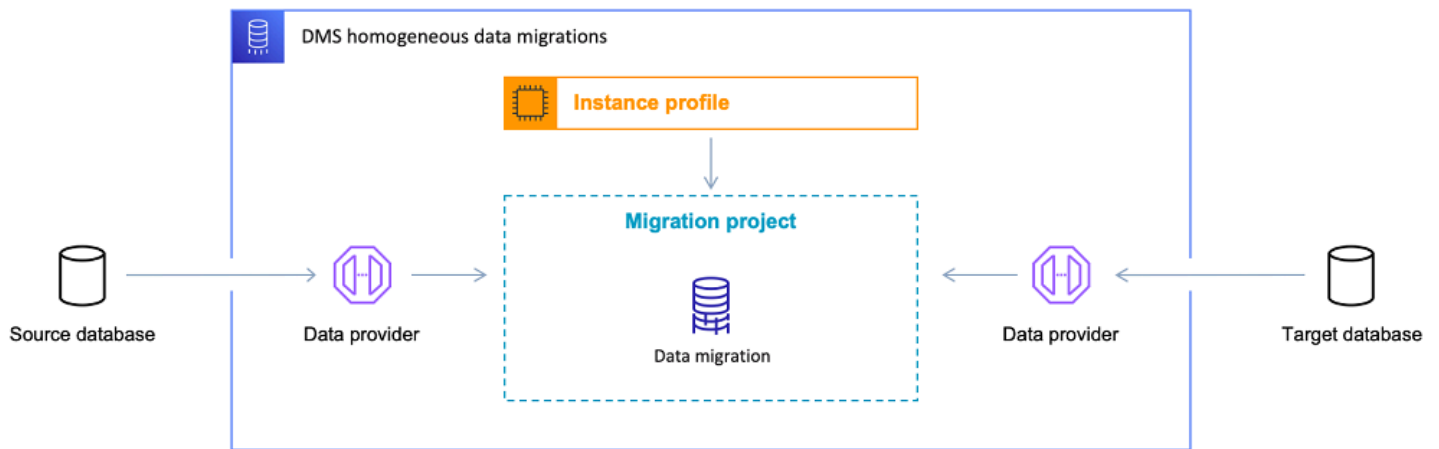
동종 데이터 마이그레이션은 서버리스이므로는 마이그레이션에 필요한 리소스를 AWS DMS 자동으로 확장합니다. 동종 데이터 마이그레이션을 사용하면 데이터, 테이블 파티션, 데이터 형식 및 보조 객체 (예: 함수, 저장 프로시저 등)를 마이그레이션할 수 있습니다.

상위 수준에서, 동종 데이터 마이그레이션은 인스턴스 프로파일, 데이터 공급자 및 마이그레이션 프로젝트를 사용하여 작동합니다. 동일한 유형의 호환되는 소스 및 대상 데이터 공급자를 사용하여 마이그레이션 프로젝트를 생성하면 데이터 마이그레이션이 실행되는 서버리스 환경을 AWS DMS 배포합니다. 그런 다음 소스 데이터 공급자에 AWS DMS 연결하고, 소스 데이터를 읽고, 디스크에 파일을 덤프하고, 기본 데이터베이스 도구를 사용하여 데이터를 복원합니다. 인스턴스 프로파일, 데이터 공급자, 마이그레이션 프로젝트에 대한 자세한 내용은 [에서 데이터 공급자, 인스턴스 프로파일 및 마이그레이션 프로젝트 작업 AWS DMS](#) 섹션을 참조하세요.

지원되는 소스 데이터베이스의 목록은 [DMS 동종 데이터 마이그레이션이 지원하는 소스](#) 섹션을 참조하세요.

지원되는 대상 데이터베이스의 목록은 [DMS 동종 데이터 마이그레이션이 지원하는 대상](#) 섹션을 참조하세요.

다음 다이어그램은 동종 데이터 마이그레이션의 작동 방식을 설명합니다.



다음 섹션에서는 동종 데이터 마이그레이션 사용에 대한 정보를 제공합니다.

주제

- [지원됨 AWS 리전](#)
- [기능](#)
- [동종 데이터 마이그레이션 제한 사항](#)
- [AWS DMS의 동종 데이터 마이그레이션 프로세스 개요](#)
- [에서 동종 데이터 마이그레이션 설정 AWS DMS](#)
- [에서 동종 데이터 마이그레이션을 위한 소스 데이터 공급자 생성 AWS DMS](#)
- [AWS DMS 스키마 변환에 사용할 대상 데이터베이스 생성 및 설정](#)
- [에서 동종 데이터 마이그레이션 실행 AWS DMS](#)
- [AWS DMS에서 동종 데이터 마이그레이션 문제 해결](#)

지원됨 AWS 리전

다음에서 동종 데이터 마이그레이션을 실행할 수 있습니다 AWS 리전.

리전 이름	리전
미국 동부(버지니아 북부)	us-east-1
미국 동부(오하이오)	us-east-2
미국 서부(캘리포니아 북부)	us-west-1

리전 이름	리전
미국 서부(오리건)	us-west-2
캐나다(중부)	ca-central-1
캐나다 서부(캘거리)	ca-west-1
남아메리카(상파울루)	sa-east-1
아시아 태평양(도쿄)	ap-northeast-1
아시아 태평양(서울)	ap-northeast-2
아시아 태평양(오사카)	ap-northeast-3
아시아 태평양(싱가포르)	ap-southeast-1
아시아 태평양(시드니)	ap-southeast-2
아시아 태평양(자카르타)	ap-southeast-3
아시아 태평양(멜버른)	ap-southeast-4
Asia Pacific (Hong Kong)	ap-east-1
아시아 태평양(뭄바이)	ap-south-1
아시아 태평양(하이데라바드)	ap-south-2
유럽(프랑크푸르트)	eu-central-1
유럽(취리히)	eu-central-2
유럽(스톡홀름)	eu-north-1
유럽(아일랜드)	eu-west-1
유럽(런던)	eu-west-2
유럽(파리)	eu-west-3

리전 이름	리전
유럽(밀라노)	eu-south-1
유럽(스페인)	eu-south-2
중동(UAE)	me-central-1
중동(바레인)	me-south-1
이스라엘(텔아비브)	il-central-1
Africa (Cape Town)	af-south-1

기능

동종 데이터 마이그레이션은 다음 기능을 제공합니다.

- AWS DMS 는 동종 데이터 마이그레이션에 필요한 AWS 클라우드 의 컴퓨팅 및 스토리지 리소스를 자동으로 관리합니다.는 데이터 마이그레이션을 시작할 때 서버리스 환경에서 이러한 리소스를 AWS DMS 배포합니다.
- AWS DMS 는 기본 데이터베이스 도구를 사용하여 동일한 유형의 데이터베이스 간에 완전 자동 마이그레이션을 시작합니다.
- 동종 데이터 마이그레이션을 사용하여 데이터는 물론 파티션, 함수, 저장 프로시저 등과 같은 보조 객체도 마이그레이션할 수 있습니다.
- 동종 데이터 마이그레이션은 전체 로드, 지속적 복제 및 지속적 복제 포함 전체 로드의 세 가지 마이그레이션 모드에서 실행할 수 있습니다.
- 동종 데이터 마이그레이션에는 온프레미스, Amazon EC2, Amazon RDS 데이터베이스를 소스로 사용할 수 있습니다. Amazon RDS 또는 Amazon Aurora를 동종 데이터 마이그레이션의 마이그레이션 대상으로 선택할 수 있습니다.
- 동종 데이터 마이그레이션은 PostgreSQL, MongoDB 및 Amazon DocumentDB 마이그레이션을 위한 대상 테이블 준비 모드만 지원합니다. 자세한 내용은 [the section called “대상 테이블 준비 모드”](#) 섹션을 참조하세요.
- 동종 데이터 마이그레이션을 사용하여 MySQL 기반 읽기 전용 복제본에서 Amazon RDS 또는 Aurora 인스턴스로 데이터를 마이그레이션할 수 있습니다

동종 데이터 마이그레이션 제한 사항

동종 데이터 마이그레이션을 사용할 경우 다음 제한 사항이 적용됩니다.

- AWS DMS 동종 데이터 마이그레이션의 선택 규칙 지원은 소스 데이터베이스 엔진 및 마이그레이션 유형에 따라 달라집니다. PostgreSQL 및 MongoDB 호환 소스는 모든 마이그레이션 유형에 대한 선택 규칙을 지원하는 반면, MySQL 소스는 전체 로드 마이그레이션 유형에 대한 선택 규칙만 지원합니다.
- 동종 데이터 마이그레이션에는 데이터 유효성 검사를 위한 기본 제공 도구가 없습니다.
- PostgreSQL과 함께 동종 데이터 마이그레이션을 사용하는 경우는 뷰를 대상 데이터베이스의 테이블로 AWS DMS 마이그레이션합니다.
- 동종 데이터 마이그레이션은 지속적 데이터 복제 중에 MySQL 엔진에 대해서만 스키마 수준의 변경 사항을 캡처합니다. 다른 엔진의 경우 소스 데이터베이스에 새 테이블을 생성하면가이 테이블을 마이그레이션할 AWS DMS 수 없습니다. 이 새 테이블을 마이그레이션하려면 데이터 마이그레이션을 다시 시작해야 합니다.
- 에서 동종 데이터 마이그레이션 AWS DMS 을 사용하여 상위 데이터베이스 버전에서 하위 데이터베이스 버전으로 데이터를 마이그레이션할 수 없습니다.
- 동종 데이터 마이그레이션은 VPC 보조 CIDR 범위의 데이터베이스 인스턴스에 대한 연결 설정을 지원하지 않습니다.
- 데이터 공급자의 동종 마이그레이션에는 8081 포트를 사용할 수 없습니다.
- 동종 데이터 마이그레이션은 암호화된 MySQL 데이터베이스 및 테이블을 대상 데이터베이스에서 암호화되지 않은 것으로 마이그레이션합니다. 이는 RDS for MySQL이 키링 플러그인을 사용한 암호화를 지원하지 않기 때문입니다. 자세한 내용은 Amazon RDS 사용 설명서의 [MySQL 키링 플러그인 지원되지 않음 설명서](#)를 참조하세요.

AWS DMS의 동종 데이터 마이그레이션 프로세스 개요

AWS DMS에서 동종 데이터 마이그레이션을 사용하여 동일한 유형의 두 데이터베이스 간에 데이터를 마이그레이션할 수 있습니다. 다음 워크플로를 사용하여 데이터 마이그레이션을 생성하고 실행할 수 있습니다.

1. 필요한 AWS Identity and Access Management(IAM) 정책 및 역할을 생성합니다. 자세한 내용은 [IAM 리소스 생성](#) 섹션을 참조하세요.

2. 소스 및 대상 데이터베이스를 구성하고 AWS DMS에서 동종 데이터 마이그레이션에 필요한 최소 권한을 가진 데이터베이스 사용자를 생성하십시오. 자세한 내용은 [소스 데이터 공급자 생성 및 대상 데이터베이스 생성 및 설정](#)(을)를 참조하세요.
3. 소스 및 대상 데이터베이스 자격 증명을 AWS Secrets Manager에 저장하십시오. 자세한 정보는 AWS Secrets Manager 사용 설명서의 [1단계: 보안 암호 생성](#)을 참조하세요.
4. AWS DMS 콘솔에서 서브넷 그룹, 인스턴스 프로파일, 데이터 공급자를 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#), [인스턴스 프로파일 생성](#), [데이터 공급자 생성](#) 섹션을 참조하세요.
5. 이전 단계에서 생성한 리소스를 사용하여 마이그레이션 프로젝트를 생성합니다. 자세한 내용은 [마이그레이션 프로젝트 생성](#) 섹션을 참조하세요.
6. 데이터 마이그레이션을 생성, 구성 및 시작합니다. 자세한 내용은 [데이터 마이그레이션 생성](#) 섹션을 참조하세요.
7. 전체 로드 또는 지속적인 복제를 완료한 후에는 전환하여 새 대상 데이터베이스를 사용할 수 있습니다.
8. 리소스를 정리합니다. Amazon은 마이그레이션 완료 후 3일 이내에 마이그레이션 프로젝트의 데이터 마이그레이션을 종료합니다. 하지만 인스턴스 프로파일, 데이터 공급자, IAM 정책 및 역할, AWS Secrets Manager의 보안 정보와 같은 리소스를 수동으로 삭제해야 합니다.

AWS DMS의 동종 데이터 마이그레이션에 관한 자세한 내용은 [PostgreSQL에서 Amazon RDS for PostgreSQL](#)로의 단계별 마이그레이션 안내를 참조하십시오.

[이 동영상](#)은 AWS DMS의 동종 데이터 마이그레이션을 소개하고 있으며 이 기능을 익히는 데 도움이 됩니다.

에서 동종 데이터 마이그레이션 설정 AWS DMS

에서 동종 데이터 마이그레이션을 설정하려면 다음 사전 조건 작업을 AWS DMS완료합니다.

주제

- [에서 동종 데이터 마이그레이션에 필요한 IAM 리소스 생성 AWS DMS](#)
- [에서 동종 데이터 마이그레이션을 위한 네트워크 설정 AWS DMS](#)
- [VPC 피어링 네트워크 구성](#)

에서 동종 데이터 마이그레이션에 필요한 IAM 리소스 생성 AWS DMS

동종 데이터 마이그레이션을 실행하려면 계정에 IAM 정책과 IAM 역할을 생성하여 다른 AWS 서비스와 상호 작용해야 합니다. 이 섹션에서는 이러한 필수 IAM 리소스를 생성합니다.

주제

- [에서 동종 데이터 마이그레이션을 위한 IAM 정책 생성 AWS DMS](#)
- [에서 동종 데이터 마이그레이션을 위한 IAM 역할 생성 AWS DMS](#)

에서 동종 데이터 마이그레이션을 위한 IAM 정책 생성 AWS DMS

를 사용하여 데이터베이스에 액세스하고 데이터를 마이그레이션하려면 동종 데이터 마이그레이션을 위한 서버리스 환경을 생성할 AWS DMS 수 있습니다. 또한 Amazon CloudWatch에서 각 데이터 마이그레이션에 대한 로그, 지표 및 진행 상황을 AWS DMS 저장합니다. 데이터 마이그레이션 프로젝트를 생성하려면 이러한 서비스에 대한 액세스 권한이 AWS DMS 필요합니다.

이 단계에서는 Amazon EC2 및 CloudWatch 리소스에 대한 AWS DMS 액세스를 제공하는 IAM 정책을 생성합니다. 그런 다음, IAM 역할을 생성하고 이 정책을 연결합니다.

에서 동종 데이터 마이그레이션을 위한 IAM 정책을 생성하려면 AWS DMS

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON을 편집기에 붙여넣습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeVpcs"
      ]
    }
  ]
}
```



```

        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogGroup"
        ],
        "Resource": "arn:aws:logs:*:*:log-group:dms-data-migration-*"
    },
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:PutLogEvents"
        ],
        "Resource": "arn:aws:logs:*:*:log-group:dms-data-migration-*:log-stream:dms-data-migration-*"
    },
    {
        "Effect": "Allow",
        "Action": "cloudwatch:PutMetricData",
        "Resource": "*"
    }
]
}

```

6. 다음을 선택합니다.
7. 정책 이름의 경우, **HomogeneousDataMigrationsPolicy**를 입력한 후 정책 생성을 선택합니다.

에서 동종 데이터 마이그레이션을 위한 IAM 역할 생성 AWS DMS

이 단계에서는 AWS Secrets Manager Amazon EC2 및 CloudWatch에 대한 AWS DMS 액세스 권한을 제공하는 IAM 역할을 생성합니다.

IAM 역할을 생성할 때 `dms-vpc-role`도 생성해야 합니다. 자세한 내용은 [Amazon Relational Database Service 사용 설명서의](#)에 대한 IAM 역할 생성을 참조 AWS DMS 하세요. Amazon Relational Database Service

에서 동종 데이터 마이그레이션을 위한 IAM 역할을 생성하려면 AWS DMS

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Roles를 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지의 신뢰할 수 있는 엔터티 유형 아래에서 AWS 서비스를 선택합니다. 기타 AWS 서비스 사용 사례에서 DMS를 선택합니다.
5. DMS 확인란을 선택하고 다음을 선택합니다.
6. 권한 추가 페이지에서 이전에 만든 HomogeneousDataMigrationsPolicy를 선택합니다.
7. 이름 지정, 검토 및 생성 페이지에서 역할 이름에 **HomogeneousDataMigrationsRole**을 입력하고 역할 생성을 선택합니다.
8. 정책 업데이트를 선택합니다.

에서 동종 데이터 마이그레이션을 위한 네트워크 설정 AWS DMS

를 사용하면 네트워크 인터페이스에 의존하는 네트워킹 연결 모델을 사용하는 동종 데이터 마이그레이션을 위한 서버리스 환경을 생성할 AWS DMS 수 있습니다. 각 데이터 마이그레이션에 대해 AWS DMS 는 인스턴스 프로파일 DMS 서브넷 그룹에 정의된 서브넷 중 하나 내에 프라이빗 IP를 할당합니다. 또한 인스턴스 프로파일이 퍼블릭 액세스를 위해 구성된 경우 비정적 퍼블릭 IP가 할당될 수 있습니다. 인스턴스 프로파일에 사용되는 서브넷은 데이터 공급자에 정의된 대로 소스 호스트와 대상 호스트 모두에 대한 액세스를 제공해야 합니다. 이 액세스는 로컬 VPC 내에 있거나 VPC 피어링, Direct Connect, VPN 등을 통해 설정될 수 있습니다.

또한 지속적인 데이터 복제를 위해서는 소스 데이터베이스와 대상 데이터베이스 간의 상호 작용을 설정해야 합니다. 이러한 구성은 소스 데이터 공급자의 위치와 네트워크 설정에 따라 달라집니다. 다음 섹션에서는 일반적인 네트워크 구성에 대해 설명합니다.

주제

- [단일 가상 프라이빗 클라우드\(VPC\)를 사용하여 네트워크 구성](#)
- [다른 가상 프라이빗 클라우드\(VPC\)를 사용하여 네트워크 구성](#)
- [Direct Connect 또는 VPN을 사용하여 VPC에 네트워크 구성](#)
- [DNS를 사용하여 도메인 엔드포인트 확인](#)

단일 가상 프라이빗 클라우드(VPC)를 사용하여 네트워크 구성

이 구성에서 AWS DMS 는 프라이빗 네트워크 내의 소스 및 대상 데이터 공급자에 연결합니다.

소스 및 대상 데이터 공급자가 동일한 VPC에 있을 때 네트워크를 구성하려면

1. 소스 및 대상 데이터 공급자가 사용하는 VPC 및 서브넷을 사용하여 AWS DMS 콘솔에서 서브넷 그룹을 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#) 단원을 참조하십시오.
2. 생성한 VPC 및 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 소스 및 대상 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하세요. 자세한 내용은 [인스턴스 프로파일 생성](#) 섹션을 참조하세요.
3. 소스 및 대상 데이터베이스에 사용되는 보안 그룹이 복제 서브넷 그룹에 지정된 데이터 마이그레이션 또는 서브넷 CIDR 블록에서 사용하는 인스턴스 프로파일에 연결된 보안 그룹의 연결을 허용하는지 확인합니다.

이 구성에서는 데이터 마이그레이션에 퍼블릭 IP 주소를 사용할 필요가 없습니다.

다른 가상 프라이빗 클라우드(VPC)를 사용하여 네트워크 구성

이 구성에서는 프라이빗 네트워크를 AWS DMS 사용하여 소스 또는 대상 데이터 공급자에 연결합니다. 다른 데이터 공급자의 경우 퍼블릭 네트워크를 AWS DMS 사용합니다. 인스턴스 프로파일과 동일한 VPC에 있는 데이터 공급자에 따라 다음 구성 중 하나를 선택하세요.

프라이빗 네트워크를 통해 연결하려면

1. 소스 데이터 공급자가 사용하는 VPC 및 서브넷을 사용하여 AWS DMS 콘솔에서 서브넷 그룹을 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#) 단원을 참조하십시오.
2. 생성한 VPC 및 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 소스 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하세요. 자세한 내용은 [인스턴스 프로파일 생성](#) 섹션을 참조하세요.
3. 소스 데이터베이스 VPC와 대상 데이터베이스 VPC 간에 VPC 피어링 연결을 구성합니다. 자세한 정보는 [VPC 피어링 연결로 작업하기](#) 섹션을 참조하세요.
4. 엔드포인트를 직접적인 프라이빗 IP 대신 사용하려는 경우 양방향 DNS 해석을 활성화해야 합니다. 자세한 정보는 [VPC 피어링 연결에 대한 DNS 확인 사용 설정](#)을 참조하세요.
5. 대상 데이터베이스 보안 그룹에 대해 소스 데이터베이스 VPC의 CIDR 블록에서 액세스를 허용합니다. 자세한 내용은 [보안 그룹을 통한 액세스 제어](#)를 참조하세요.

6. 대상 데이터베이스 보안 그룹에 대해 대상 데이터베이스 VPC의 CIDR 블록에서 액세스를 허용합니다. 자세한 내용은 [보안 그룹을 통한 액세스 제어](#)를 참조하세요.

퍼블릭 네트워크를 통해 연결하려면

데이터베이스가 어떤 주소에서든 연결을 수락하는 경우

1. VPC 및 퍼블릭 서브넷을 사용하여 AWS DMS 콘솔에서 서브넷 그룹을 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#) 단원을 참조하십시오.
2. 생성한 VPC 및 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 인스턴스 프로파일에 대해 공개적으로 사용 가능 옵션을 켜기로 설정합니다.

데이터 마이그레이션과 연결할 수 있는 영구 퍼블릭 IP 주소가 필요한 경우:

1. VPC 및 프라이빗 서브넷을 사용하여 AWS DMS 콘솔에서 서브넷 그룹을 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#) 단원을 참조하십시오.
2. 생성한 VPC 및 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 인스턴스 프로파일에 대해 공개적으로 사용 가능 옵션을 끄기로 설정합니다.
3. NAT 게이트웨이를 설정합니다. 자세한 내용은 [NAT 게이트웨이 작업](#)을 참조하세요.
4. NAT 게이트웨이에 대한 라우팅 테이블을 설정합니다. 자세한 내용은 [NAT 게이트웨이 사용 사례](#) 섹션을 참조하세요.
5. 데이터베이스 보안 그룹에서 NAT 게이트웨이의 퍼블릭 IP 주소를 통한 액세스를 허용하세요. 자세한 내용은 [보안 그룹을 통한 액세스 제어](#)를 참조하세요.

Direct Connect 또는 VPN을 사용하여 VPC에 네트워크 구성

Direct Connect 또는 VPN 연결(소프트웨어 또는 하드웨어)을 통해 원격 네트워크를 VPC에 연결할 수 있습니다. 이러한 옵션을 사용하면 내부 네트워크를 로 확장 AWS 클라우드 하고 모니터링, 인증, 보안 및 데이터 시스템과 같은 기존 온프레미스 서비스를 AWS 리소스와 통합할 수 있습니다. 이 구성의 경우 VPC 보안 그룹에 트래픽을 온프레미스 VPN으로 연결할 수 있는 호스트로 보내는 라우팅 규칙이 포함되어야 합니다. 이 트래픽은 VPC CIDR 범위 또는 특정 IP 주소를 사용하여 지정할 수 있습니다. NAT 호스트에는 VPC CIDR 범위 또는 보안 그룹에서 NAT 인스턴스로의 트래픽을 허용하도록 구성된 자체 보안 그룹이 있어야 VPC와 온프레미스 인프라 간의 원활한 통신을 보장할 수 있습니다. 자세한 내용은 [AWS Site-to-Site VPN 사용 설명서의 Get started with Site-to-Site VPN procedure](#) 5 단계를 참조하세요.

DNS를 사용하여 도메인 엔드포인트 확인

AWS DMS 동종 마이그레이션의 DNS 확인을 위해 서비스는 주로 Amazon ECS DNS 해석기를 사용하여 도메인 엔드포인트를 확인합니다. 추가 DNS 확인 기능이 필요한 경우 Amazon Route 53 Resolver를 대체 솔루션으로 사용할 수 있습니다. 자세한 내용은 Amazon Route 53 사용자 안내서의 [Route 53 Resolver 시작하기](#)를 참조하세요. Amazon Route 53 Resolver에서 온프레미스 이름 서버를 사용하여 엔드포인트 확인을 구성하는 방법에 대한 자세한 내용은 [the section called “자체 온프레미스 이름 서버 사용”](#) 섹션을 참조하세요.

Note

데이터 마이그레이션 로그에 "연결 시작 - 네트워킹 모델: VPC 피어링" 메시지가 표시되면 [the section called “VPC 피어링 네트워크 구성”](#) 주제를 참조하세요.

VPC 피어링 네트워크 구성

를 사용하면 Amazon VPC 서비스를 기반으로 Virtual Private Cloud(VPC)에서 동종 데이터 마이그레이션을 위한 서버리스 환경을 생성할 AWS DMS 수 있습니다. 인스턴스 프로파일을 생성할 때 사용할 VPC를 지정합니다. 계정 및에 기본 VPC를 사용하거나 새 VPC를 생성할 AWS 리전 수 있습니다.

각 데이터 마이그레이션에 대해서는 인스턴스 프로파일에 사용하는 VPC와 VPC 피어링 연결을 AWS DMS 설정합니다. 다음으로 인스턴스 프로파일과 연결된 보안 그룹에 CIDR 블록을 AWS DMS 추가합니다. 는 인스턴스 프로파일에 퍼블릭 IP 주소를 AWS DMS 연결하므로 동일한 인스턴스 프로파일을 사용하는 모든 데이터 마이그레이션은 동일한 퍼블릭 IP 주소를 갖습니다. 데이터 마이그레이션이 중지되거나 실패할 경우, AWS DMS 는 VPC 피어링 연결을 삭제합니다.

CIDR 블록이 인스턴스 프로파일 VPC의 VPC와 겹치지 않도록 하려면 , 10.0.0.0/8 172.16.0.0/12 및 CIDR 블록 중 하나의 /24 접두사를 AWS DMS 사용합니다 192.168.0.0/16. 예를 들어, 세 개의 데이터 마이그레이션을 동시에 실행하는 경우, AWS DMS 는 다음 CIDR 블록을 사용하여 VPC 피어링 연결을 설정합니다.

- 192.168.0.0/24 - 첫 번째 데이터 마이그레이션에 사용
- 192.168.1.0/24 - 두 번째 데이터 마이그레이션에 사용
- 192.168.2.0/24 - 세 번째 데이터 마이그레이션에 사용

다양한 네트워크 구성을 사용하여 소스 데이터베이스와 대상 데이터베이스 간의 상호 작용을 설정할 수 있습니다 AWS DMS. 또한 지속적인 데이터 복제를 위해서는 소스 데이터베이스와 대상 데이터베이스

이스 간의 상호 작용을 설정해야 합니다. 이러한 구성은 소스 데이터 공급자의 위치와 네트워크 설정에 따라 달라집니다. 다음 섹션에서는 일반적인 네트워크 구성에 대해 설명합니다.

주제

- [단일 가상 프라이빗 클라우드\(VPC\)를 사용하여 네트워크 구성](#)
- [다른 가상 프라이빗 클라우드\(VPC\)를 사용하여 네트워크 구성](#)
- [온프레미스 소스 데이터 공급자 사용](#)
- [지속적인 데이터 복제 구성](#)

단일 가상 프라이빗 클라우드(VPC)를 사용하여 네트워크 구성

이 구성에서 AWS DMS 는 프라이빗 네트워크 내의 소스 및 대상 데이터 공급자에 연결합니다.

소스 및 대상 데이터 공급자가 동일한 VPC에 있을 때 네트워크를 구성하려면

1. 소스 및 대상 데이터 공급자가 사용하는 VPC 및 서브넷을 사용하여 AWS DMS 콘솔에서 서브넷 그룹을 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#) 단원을 참조하십시오.
2. 생성한 VPC 및 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 소스 및 대상 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하세요. 자세한 내용은 [인스턴스 프로파일 생성](#) 섹션을 참조하세요.

이 구성에서는 데이터 마이그레이션에 퍼블릭 IP 주소를 사용할 필요가 없습니다.

다른 가상 프라이빗 클라우드(VPC)를 사용하여 네트워크 구성

이 구성에서는 프라이빗 네트워크를 AWS DMS 사용하여 소스 또는 대상 데이터 공급자에 연결합니다. 다른 데이터 공급자의 경우 퍼블릭 네트워크를 AWS DMS 사용합니다. 인스턴스 프로파일과 동일한 VPC에 있는 데이터 공급자에 따라 다음 구성 중 하나를 선택하세요.

소스 데이터 공급자를 위한 프라이빗 네트워크와 대상 데이터 공급자를 위한 퍼블릭 네트워크를 구성하려면

1. 소스 데이터 공급자가 사용하는 VPC 및 서브넷을 사용하여 AWS DMS 콘솔에서 서브넷 그룹을 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#) 단원을 참조하십시오.
2. 생성한 VPC 및 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 소스 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하세요. 자세한 내용은 [인스턴스 프로파일 생성](#) 섹션을 참조하세요.

3. 마이그레이션 프로젝트를 엽니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 연결 및 보안에서 퍼블릭 IP 주소를 기록해 둡니다.
4. 대상 데이터베이스 보안 그룹에서 데이터 마이그레이션의 퍼블릭 IP 주소를 통한 액세스를 허용하세요. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 [보안 그룹을 통한 액세스 제어](#)를 참조하세요.

소스 데이터 공급자를 위한 퍼블릭 네트워크와 대상 데이터 공급자를 위한 프라이빗 네트워크를 구성하려면

1. 대상 데이터 공급자가 사용하는 VPC 및 서브넷을 사용하여 AWS DMS 콘솔에서 서브넷 그룹을 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#) 단원을 참조하십시오.
2. 생성한 VPC 및 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 대상 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하세요. 자세한 내용은 [인스턴스 프로파일 생성](#) 섹션을 참조하세요.
3. 마이그레이션 프로젝트를 엽니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 연결 및 보안에서 퍼블릭 IP 주소를 기록해 둡니다.
4. 소스 데이터베이스 보안 그룹에서 데이터 마이그레이션의 퍼블릭 IP 주소를 통한 액세스를 허용하세요. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 [보안 그룹을 통한 액세스 제어](#)를 참조하세요.

온프레미스 소스 데이터 공급자 사용

이 구성에서 AWS DMS 는 퍼블릭 네트워크 내의 소스 데이터 공급자에 연결합니다.는 프라이빗 네트워크를 AWS DMS 사용하여 대상 데이터 공급자에 연결합니다.

Note

동종 데이터 마이그레이션의 경우 AWS DMS 는 퍼블릭 네트워크 내의 소스 데이터베이스에 연결합니다. 그러나 퍼블릭 네트워크 내의 소스 데이터베이스에 대한 연결이 항상 가능한 것은 아닙니다. 자세한 내용은 [AWS DMS 동종 데이터 마이그레이션 및 Network Load Balancer를 사용하여 프라이빗 네트워크를 통해 온프레미스 MySQL 데이터베이스를 Amazon Aurora MySQL로 마이그레이션](#)을 참조하세요.

소스 온프레미스 데이터 공급자를 위한 네트워크를 구성하려면

1. 대상 데이터 공급자가 사용하는 VPC 및 서브넷을 사용하여 AWS DMS 콘솔에서 서브넷 그룹을 생성합니다. 자세한 내용은 [서브넷 그룹 생성](#) 단원을 참조하십시오.
2. 생성한 VPC 및 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 대상 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하세요. 자세한 내용은 [인스턴스 프로파일 생성](#) 섹션을 참조하세요.
3. 마이그레이션 프로젝트를 엽니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 연결 및 보안에서 퍼블릭 IP 주소를 기록해 둡니다.
4. AWS DMS내 데이터 마이그레이션의 퍼블릭 IP 주소에서 소스 데이터베이스에 액세스할 수 있도록 허용하세요.

AWS DMS 는 VPC 보안 그룹의 인바운드 또는 아웃바운드 규칙을 생성합니다. 이 작업을 수행하면 데이터 마이그레이션이 실패할 수 있으므로 이러한 규칙을 삭제하지 마세요. VPC 보안 그룹에서 고유한 규칙을 구성할 수 있습니다. 규칙을 관리할 수 있도록 설명을 규칙에 추가하는 것이 좋습니다.

지속적인 데이터 복제 구성

전체 로드 및 변경 데이터 캡처(CDC) 또는 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션을 실행하려면 소스 데이터베이스와 대상 데이터베이스 간에 연결을 허용해야 합니다.

공개적으로 액세스할 수 있는 소스 데이터베이스와 대상 데이터베이스 간의 연결을 구성하려면

1. 소스 및 대상 데이터베이스의 퍼블릭 IP 주소를 기록해 두세요.
2. 대상 데이터베이스의 퍼블릭 IP 주소에서 소스 데이터베이스에 액세스할 수 있도록 허용하세요.
3. 소스 데이터베이스의 퍼블릭 IP 주소에서 대상 데이터베이스에 액세스할 수 있도록 허용하세요.

단일 VPC에서 비공개로 액세스할 수 있는 소스 데이터베이스와 대상 데이터베이스 간의 연결을 구성하려면

1. 소스 및 대상 데이터베이스의 프라이빗 IP 주소를 기록해 두세요.

Important

소스 및 대상 데이터베이스가 서로 다른 VPC나 네트워크에 있는 경우, 소스 및 대상 데이터베이스에 퍼블릭 IP 주소만 사용할 수 있습니다. 데이터 공급자에서는 퍼블릭 호스트 이름 또는 IP 주소만 사용할 수 있습니다.

2. 대상 데이터베이스의 보안 그룹에서 소스 데이터베이스에 액세스할 수 있도록 허용하세요.
3. 소스 데이터베이스의 보안 그룹에서 대상 데이터베이스에 액세스할 수 있도록 허용하세요.

에서 동종 데이터 마이그레이션을 위한 소스 데이터 공급자 생성 AWS DMS

AWS DMS Schema Conversion은 Microsoft SQL Server, Oracle, PostgreSQL, MySQL, IBM DB2 for z/OS 및 SAP ASE(Sybase ASE) 데이터베이스를 [동종 데이터베이스 마이그레이션](#) 프로젝트의 소스 데이터 공급자로 지원합니다.

지원되는 데이터베이스 버전은 [DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자](#) 섹션을 참조하세요.

소스 데이터 공급자는 온프레미스, Amazon EC2 또는 Amazon RDS 데이터베이스일 수 있습니다.

주제

- [에서 MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS](#)
- [에서 PostgreSQL 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS](#)
- [에서 MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS](#)

에서 MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

MySQL 호환 데이터베이스(MySQL 또는 MariaDB)를 AWS DMS의 [동종 데이터베이스 마이그레이션](#)에 대한 소스로 사용할 수 있습니다. 이 경우, 소스 데이터 공급자는 온프레미스, Amazon EC2, RDS for MySQL 또는 MariaDB 데이터베이스일 수 있습니다.

동종 데이터 마이그레이션을 실행하려면 모든 소스 테이블과 복제용 보조 객체에 대한 SELECT 권한이 있는 데이터베이스 사용자를 사용해야 합니다. 변경 데이터 캡처(CDC) 작업의 경우, 이 사용자는 REPLICATION CLIENT(MariaDB 10.5.2 이후 버전의 경우 BINLOG MONITOR) 및 REPLICATION SLAVE 권한도 있어야 합니다. 전체 로드 데이터 마이그레이션의 경우, 이 두 가지 권한이 필요하지 않습니다.

다음 스크립트를 사용하여 MySQL 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다. 마이그레이션하는 모든 데이터베이스에 대해 GRANT 쿼리를 실행합니다 AWS.

```
CREATE USER 'your_user'@'%' IDENTIFIED BY 'your_password';

GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'your_user'@'%;
GRANT SELECT, RELOAD, LOCK TABLES, SHOW VIEW, EVENT, TRIGGER ON *.* TO 'your_user'@'%;

GRANT BACKUP_ADMIN ON *.* TO 'your_user'@'%;
```

이전 예제에서는 자신의 정보로 각각의 `### ## ## ###`를 바꿉니다. 원본 MySQL 데이터베이스 버전이 8.0보다 낮으면 `GRANT BACKUP_ADMIN` 명령을 건너뛸 수 있습니다.

다음 스크립트를 사용하여 MariaDB 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다. 마이그레이션하는 모든 데이터베이스에 대해 `GRANT` 쿼리를 실행합니다 AWS.

```
CREATE USER 'your_user'@'%' IDENTIFIED BY 'your_password';
GRANT SELECT, RELOAD, LOCK TABLES, REPLICATION SLAVE, BINLOG MONITOR, SHOW VIEW ON *.*
TO 'your_user'@'%;
```

이전 예제에서는 자신의 정보로 각각의 `### ## ## ###`를 바꿉니다.

다음 섹션에서는 자체 관리형 및 AWS관리형 MySQL 데이터베이스의 특정 구성 사전 요구 사항을 설명합니다.

주제

- [자체 관리형 MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용](#)
- [에서 동종 데이터 마이그레이션의 소스로 AWS관리형 MySQL 호환 데이터베이스 사용 AWS DMS](#)
- [MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항](#)

자체 관리형 MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용

이 섹션에서는 온프레미스나 Amazon EC2 인스턴스에서 호스팅되는 MySQL 호환 데이터베이스를 구성하는 방법에 대해 설명합니다.

원본 MySQL 또는 MariaDB 데이터베이스의 버전을 선택합니다. 에서에 설명된 대로 소스 MySQL 또는 MariaDB 데이터베이스 버전을 AWS DMS 지원하는지 확인합니다 [DMS 동종 데이터 마이그레이션이 지원하는 소스](#).

CDC를 사용하려면 이진 로깅을 활성화해야 합니다. 바이너리 로깅을 활성화하려면 MySQL 또는 MariaDB의 `my.ini`(Windows) 또는 `my.cnf`(UNIX) 파일에서 다음 파라미터를 구성해야 합니다.

파라미터	값
server-id	이 파라미터 값을 1 이상의 값으로 설정합니다.
log-bin	log-bin=E:\MySQL_Logs\BinLog 과 같은 이진 로그 파일로 이 경로를 설정합니다. 파일 확장자를 포함해서는 안 됩니다.
binlog_format	이 파라미터를 ROW으로 설정합니다. binlog_format 을 STATEMENT 로 설정하면 대상에 데이터를 복제할 때 불일치가 발생할 수 있으므로 복제 중에 이 설정을 사용하는 것이 좋습니다. 또한 데이터베이스 엔진이 자동으로 STATEMENT 기반 로깅으로 전환되기 때문에 binlog_format 이 MIXED로 설정된 경우 데이터베이스 엔진은 일관되지 않은 유사 데이터를 대상에 기록합니다.
expire_logs_days	이 파라미터 값을 1 이상의 값으로 설정합니다. 디스크 공간의 과사용을 방지하려면 기본값 0을 사용하지 않는 것이 좋습니다.
binlog_checksum	이 파라미터를 NONE으로 설정합니다.
binlog_row_image	이 파라미터를 FULL로 설정합니다.
log_slave_updates	MySQL 또는 MariaDB 복제본을 소스로 사용하고 있는 경우 이 파라미터를 TRUE로 설정합니다.

에서 동종 데이터 마이그레이션의 소스로 AWS관리형 MySQL 호환 데이터베이스 사용 AWS DMS

이 섹션에서는 Amazon RDS for MySQL과 Amazon RDS for MariaDB 데이터베이스 인스턴스를 구성하는 방법에 대해 설명합니다.

AWS관리형 MySQL 또는 MariaDB 데이터베이스를에서 동종 데이터 마이그레이션의 소스로 사용하는 경우 CDC에 대해 다음과 같은 사전 조건이 있는지 AWS DMS확인합니다.

- RDS for MySQL 및 MariaDB에 대한 바이너리 로그를 활성화하려면 인스턴스 수준에서 자동 백업을 활성화하세요. Aurora MySQL 클러스터의 이진 로그를 활성화하려면 파라미터 그룹에서 binlog_format 변수를 변경합니다. Aurora MySQL 클러스터에 대해서는 자동 백업을 활성화할 필요는 없습니다.

그런 다음, `binlog_format` 파라미터를 ROW로 설정합니다.

자동 백업 설정에 관한 자세한 내용은 Amazon RDS 사용 설명서의 [자동 백업 활성화](#) 섹션을 참조하세요.

Amazon RDS for MySQL 또는 MariaDB 데이터베이스의 바이너리 로깅 설정에 관한 자세한 내용은 Amazon RDS 사용 설명서의 [바이너리 로깅 형식 설정](#)을 참조하세요.

Aurora MySQL 클러스터의 이진 로깅을 설정하는 방법에 관한 자세한 내용은 [Amazon Aurora MySQL 클러스터의 이진 로깅을 활성화하려면 어떻게 해야 하나요?](#)를 참조하세요.

- 바이너리 로그를 사용할 수 있는지 확인합니다 AWS DMS. AWS관리형 MySQL 및 MariaDB 데이터베이스는 이진 로그를 최대한 빨리 제거하므로 로그를 계속 사용할 수 있는 시간을 늘려야 합니다. 예를 들어, 로그 보존 시간을 24시간으로 늘리려면 다음 명령을 실행합니다.

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

- `binlog_row_image` 파라미터를 Full로 설정합니다.
- `binlog_checksum` 파라미터를 NONE으로 설정합니다.
- Amazon RDS MySQL 또는 MariaDB 복제본을 소스로 사용하고 있는 경우, 읽기 전용 복제본에서 백업을 활성화하고 `log_slave_updates` 파라미터를 TRUE로 설정해야 합니다.

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 시 적용되는 제한 사항은 다음과 같습니다.

- 시퀀스와 같은 MariaDB 객체는 동종 마이그레이션 작업에서 지원되지 않습니다.
- MariaDB에서 Amazon RDS MySQL/Aurora MySQL로의 마이그레이션은 호환되지 않는 객체 차이로 인해 실패할 수 있습니다.
- 데이터 소스에 연결하는 데 사용하는 사용자 이름에는 다음과 같은 제한이 있습니다.
 - 길이는 2~64자일 수 있습니다.
 - 스페이스를 포함할 수 없습니다.
 - a-z, A-Z, 0-9, 밑줄(_) 문자를 포함할 수 있습니다.
 - a-z 또는 A-Z로 시작해야 합니다.

- 데이터 소스에 연결하는 데 사용하는 암호에는 다음과 같은 제한이 있습니다.
 - 길이는 1~128자일 수 있습니다.
 - 작은따옴표('), 큰따옴표("), 세미콜론(;) 또는 스페이스는 포함할 수 없습니다.
- AWS DMS 동종 데이터 마이그레이션은 소스 객체가 암호화된 경우에도 대상 Amazon RDS 인스턴스에 암호화되지 않은 MySQL 및 MariaDB 객체를 생성합니다. RDS for MySQL은 암호화된 객체에 필요한 MySQL keyring_aws AWS 키링 플러그인을 지원하지 않습니다. Amazon RDS 사용자 설명서의 [MySQL 키링 플러그인 지원되지 않음 설명서](#)를 참조하세요.
- AWS DMS 는 소스 데이터에 GTIDs(Global Transaction Identifier)가 포함되어 있더라도 데이터 복제를 위해에 GTID를 사용하지 않습니다.

에서 PostgreSQL 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

PostgreSQL 데이터베이스를 AWS DMS의 [동종 데이터베이스 마이그레이션](#)에서 소스로 사용할 수 있습니다. 이 경우, 소스 데이터 공급자는 온프레미스, Amazon EC2, RDS for PostgreSQL 데이터베이스일 수 있습니다.

동종 데이터 마이그레이션을 실행하려면 AWS DMS PostgreSQL 소스 데이터베이스에 대해에서 지정된 데이터베이스 사용자에게 수퍼유저 권한을 부여합니다. 데이터베이스 사용자가 원본의 복제 관련 함수에 액세스하려면 수퍼유저 권한이 필요합니다. 전체 로드 데이터 마이그레이션의 경우, 테이블을 마이그레이션할 수 있는 SELECT 권한이 데이터베이스 사용자에게 있어야 합니다.

다음 스크립트를 사용하여 PostgreSQL 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다. 마이그레이션하는 모든 데이터베이스에 대해 GRANT 쿼리를 실행합니다 AWS.

```
CREATE USER your_user WITH LOGIN PASSWORD 'your_password';
ALTER USER your_user WITH SUPERUSER;
GRANT SELECT ON ALL TABLES IN SCHEMA schema_name TO your_user;
```

이전 예제에서는 자신의 정보로 각각의 **### ## ## ###**를 바꿉니다.

AWS DMS 는 프라이머리 키가 있는 PostgreSQL 테이블의 변경 데이터 캡처(CDC)를 지원합니다. 테이블에 프라이머리 키가 없는 경우, 미리 쓰기 로그(WAL)에 데이터베이스 행의 이전 이미지가 포함되지 않습니다. 여기서는 추가 구성 설정과 테이블 복제본 ID를 해결 방법으로 사용할 수 있습니다. 하지만 이 접근 방식을 사용하면 추가 로그가 생성될 수 있습니다. 신중하게 테스트한 후에만 테이블 복제본 ID를 해결 방법으로 사용하는 것이 좋습니다. 자세한 내용은 [PostgreSQL 데이터베이스를 DMS 소스로 사용 시 추가 구성 설정](#) 단원을 참조하십시오.

다음 섹션에서는 자체 관리형 및 AWS관리형 PostgreSQL 데이터베이스의 특정 구성 사전 요구 사항을 설명합니다.

주제

- [자체 관리형 PostgreSQL 데이터베이스를에서 동종 데이터 마이그레이션의 소스로 사용 AWS DMS](#)
- [AWS관리형 PostgreSQL 데이터베이스를에서 동종 데이터 마이그레이션의 소스로 사용 AWS DMS](#)
- [PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항](#)

자체 관리형 PostgreSQL 데이터베이스를에서 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

이 섹션에서는 온프레미스나 Amazon EC2 인스턴스에서 호스팅되는 PostgreSQL 데이터베이스를 구성하는 방법에 대해 설명합니다.

소스 PostgreSQL 데이터베이스의 버전을 확인하세요. 에서에 설명된 대로 소스 PostgreSQL 데이터베이스 버전을 AWS DMS 지원하는지 확인합니다 [DMS 동종 데이터 마이그레이션이 지원하는 소스](#).

동종 데이터 마이그레이션은 논리적 복제를 사용하여 CDC(변경 데이터 캡처)를 지원합니다. 자체 관리형 PostgreSQL 소스 데이터베이스에서 논리적 복제를 활성화하려면 `postgresql.conf` 구성 파일에서 다음 파라미터와 값을 설정합니다.

- `wal_level`을 `logical`로 설정합니다.
- `max_replication_slots`를 1보다 큰 값으로 설정합니다.

실행할 작업 개수에 따라 `max_replication_slots` 값을 설정합니다. 예를 들어, 5개 작업을 실행하려면 최소 5개의 슬롯을 설정해야 합니다. 작업이 시작된 즉시 슬롯이 자동으로 열리고 작업이 더 이상 실행되지 않더라도 계속 열려 있습니다. 열린 슬롯은 반드시 수동으로 삭제하세요.

- `max_wal_senders`를 1보다 큰 값으로 설정합니다.

`max_wal_senders` 파라미터는 실행 가능한 동시 작업 개수를 설정합니다.

- `wal_sender_timeout` 파라미터는 지정된 밀리초보다 오래 사용되지 않는 복제 연결을 종료합니다. 기본값은 60,000밀리초(60초)입니다. 값을 0으로 설정하면 시간 제한 메커니즘이 비활성화되며, 이는 유효한 DMS 설정입니다.

일부 파라미터는 정적이며 서버 시작 시에만 설정할 수 있습니다. 구성 파일의 해당 항목에 대한 모든 변경 사항은 서버를 다시 시작할 때까지 무시됩니다. 자세한 내용은 [PostgreSQL 설명서](#)를 참조하세요.

AWS관리형 PostgreSQL 데이터베이스를에서 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

이 섹션에서는 Amazon RDS for PostgreSQL 데이터베이스 인스턴스를 구성하는 방법에 대해 설명합니다.

PostgreSQL DB 인스턴스의 AWS 마스터 사용자 계정을 동종 데이터 마이그레이션을 위한 PostgreSQL 소스 데이터 공급자의 사용자 계정으로 사용합니다 AWS DMS. 마스터 사용자 계정에는 CDC 설정을 허용하는 데 필요한 역할이 있습니다. 마스터 사용자 계정 이외의 계정을 사용하는 경우, 이 계정에는 `rds_superuser` 역할과 `rds_replication` 역할이 있어야 합니다. `rds_replication` 역할은 논리적 슬롯을 관리하고 논리적 슬롯을 사용하여 데이터를 스트리밍할 수 있는 권한을 부여합니다.

다음 코드 예제를 사용하여 `rds_superuser` 및 `rds_replication` 역할을 부여하세요.

```
GRANT rds_superuser to your_user;  
GRANT rds_replication to your_user;
```

이전 예제에서 *your_user*를 데이터베이스 사용자 이름으로 바꾸세요.

논리적 복제를 활성화하려면 DB 파라미터 그룹의 `rds.logical_replication` 파라미터를 1로 설정합니다. 이 정적 파라미터를 적용하려면 DB 인스턴스를 재부팅해야 합니다.

PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항

PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 시 적용되는 제한 사항은 다음과 같습니다.

- 데이터 소스에 연결하는 데 사용하는 사용자 이름에는 다음과 같은 제한이 있습니다.
 - 길이는 2~64자일 수 있습니다.
 - 스페이스를 포함할 수 없습니다.
 - a-z, A-Z, 0-9, 밑줄(_) 문자를 포함할 수 있습니다.
 - a-z 또는 A-Z로 시작해야 합니다.
- 데이터 소스에 연결하는 데 사용하는 암호에는 다음과 같은 제한이 있습니다.

- 길이는 1~128자일 수 있습니다.
- 작은따옴표('), 큰따옴표("), 세미콜론(;) 또는 스페이스는 포함할 수 없습니다.

에서 MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

AWS DMS에서 MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용할 수 있습니다. 이 경우 소스 데이터 공급자는 온프레미스, Amazon EC2 for MongoDB 데이터베이스 또는 Amazon DocumentDB(MongoDB 호환) 데이터베이스일 수 있습니다.

지원되는 데이터베이스 버전은 [DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자](#) 섹션을 참조하세요.

다음 섹션에서는 자체 관리형 MongoDB 데이터베이스 및 AWS관리형 Amazon DocumentDB 데이터베이스의 특정 구성 사전 조건을 설명합니다.

주제

- [자체 관리형 MongoDB 데이터베이스를에서 동종 데이터 마이그레이션의 소스로 사용 AWS DMS](#)
- [에서 동종 데이터 마이그레이션을 위한 소스로 Amazon DocumentDB 데이터베이스 사용 AWS DMS](#)
- [MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 기능](#)
- [MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항](#)
- [MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 모범 사례](#)

자체 관리형 MongoDB 데이터베이스를에서 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

이 섹션에서는 온프레미스나 Amazon EC2 인스턴스에서 호스팅되는 MongoDB 데이터베이스를 구성하는 방법에 대해 설명합니다.

소스 MongoDB 데이터베이스의 버전을 확인하세요. 에서에 설명된 대로 소스 MongoDB 데이터베이스 버전을 AWS DMS 지원하는지 확인합니다 [DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자](#).

MongoDB 소스를 이용한 동종 데이터 마이그레이션을 실행하려면 루트 권한을 가진 사용자 계정을 생성하거나 마이그레이션할 데이터베이스에서만 권한이 있는 사용자를 생성할 수 있습니다. 사용자 생성에 대한 자세한 내용은 [MongoDB를의 소스로 사용할 때 필요한 권한 AWS DMS](#) 섹션을 참조하세요.

MongoDB에서 지속적 복제 또는 CDC를 사용하려면 MongoDB 작업 로그(oplog)에 대한 액세스 권한이 AWS DMS 필요합니다. 자세한 내용은 [CDC를 위한 MongoDB 복제본 세트 구성](#) 단원을 참조하십시오.

MongoDB 인증 방법에 대한 자세한 내용은 [MongoDB를 소스로 사용할 때의 보안 요구 사항 AWS DMS](#) 섹션을 참조하세요.

MongoDB를 소스로 사용할 경우 동종 데이터 마이그레이션은 Amazon DocumentDB가 지원하는 모든 데이터 유형을 지원합니다.

MongoDB를 소스로 사용하는 경우 Secrets Manager에 사용자 자격 증명을 저장하려면 기타 유형의 보안 암호 유형을 사용하여 일반 텍스트로 제공해야 합니다. 자세한 내용은 [보안 암호를 사용하여 AWS Database Migration Service 엔드포인트 액세스](#) 단원을 참조하십시오.

다음 코드 샘플은 일반 텍스트를 사용하여 데이터베이스 보안 암호를 저장하는 방법을 보여줍니다.

```
{
  "username": "dbuser",
  "password": "dbpassword"
}
```

에서 동종 데이터 마이그레이션을 위한 소스로 Amazon DocumentDB 데이터베이스 사용 AWS DMS

이 섹션에서는 동종 데이터 마이그레이션의 소스로 사용할 Amazon DocumentDB 데이터베이스 인스턴스를 구성하는 방법을 설명합니다.

Amazon DocumentDB 인스턴스의 마스터 사용자 이름을 AWS DMS의 동종 데이터 마이그레이션을 위한 MongoDB 호환 소스 데이터 공급자의 사용자 계정으로 사용합니다. 마스터 사용자 계정에는 CDC 설정을 허용하는 데 필요한 역할이 있습니다. 마스터 사용자 계정 이외의 계정을 사용하는 경우, 이 계정에는 루트 역할이 있어야 합니다. 루트 계정으로 사용자 생성에 대한 자세한 내용은 [Amazon DocumentDB를 소스로 사용하기 위한 권한 설정](#) 섹션을 참조하세요.

논리적 복제를 활성화하려면 데이터베이스 파라미터 그룹의 `change_stream_log_retention_duration` 파라미터를 트랜잭션 워크로드에 적합한 설정으로 설정합니다. 이 정적 파라미터의 변경을 적용하려면 DB 인스턴스를 재부팅해야 합니다. 전체 로드 전 용을 포함한 모든 태스크 유형에 대해 데이터 마이그레이션을 시작하기 전에 지정된 데이터베이스 내의 모든 컬렉션에 대해 Amazon DocumentDB 변경 스트림을 활성화하거나 선택한 컬렉션에 대해서만 스트림을 활성화합니다. Amazon DocumentDB에 대한 변경 스트림 활성화에 대한 자세한 내용은 Amazon DocumentDB 개발자 안내서의 [변경 스트림 활성화](#)를 참조하세요.

Note

AWS DMS 는 Amazon DocumentDB 변경 스트림을 사용하여 지속적 복제 중에 변경 사항을 캡처합니다. DMS가 레코드를 읽기 전에 Amazon DocumentDB가 변경 스트림에서 레코드를 삭제하면 태스크가 실패합니다. 변경 사항을 최소 24시간 동안 유지하도록 `change_stream_log_retention_duration` 파라미터를 설정하는 것이 좋습니다.

동종 데이터 마이그레이션에 Amazon DocumentDB를 사용하려면 Secrets Manager의 Amazon DocumentDB 데이터베이스용 자격 증명에 사용자 자격 증명을 저장합니다.

MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 기능

- 전체 로드 단계에서 Amazon DocumentDB가 지원하는 모든 보조 인덱스를 마이그레이션할 수 있습니다.
- AWS DMS 는 컬렉션을 병렬로 마이그레이션합니다. 동종 데이터 마이그레이션은 최대 성능을 위해 컬렉션에 있는 각 문서의 평균 크기를 기반으로 런타임 시 세그먼트를 계산합니다.
- DMS는 CDC 단계에서 생성한 보조 인덱스를 복제할 수 있습니다. DMS는 MongoDB 버전 6.0에서 이 기능을 지원합니다.
- DMS는 중첩 수준이 97보다 큰 문서를 지원합니다.

MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항

- 문서는 \$ 접두사가 있는 필드 이름을 사용할 수 없습니다.
- AWS DMS 는 시계열 컬렉션 마이그레이션을 지원하지 않습니다.
- AWS DMS 는 CDC 단계에서 `createdrop`, 또는 `rename collection` DDL 이벤트를 지원하지 않습니다.
- AWS DMS 는 `_id` 필드에 대한 컬렉션에서 일관되지 않은 데이터 유형을 지원하지 않습니다. 예를 들어 다음 지원되지 않는 컬렉션에는 `_id` 필드에 대한 여러 데이터 유형이 있습니다.

```
rs0 [direct: primary] test> db.collection1.aggregate([
...   {
...     $group: {
...       _id: { $type: "$_id" },
...       count: { $sum: 1 }
...     }
...   }
... ]
```

```
...    }
...  }
... ])
[ { _id: 'string', count: 6136 }, { _id: 'objectId', count: 848033 } ]
```

- CDC 전용 작업의 경우는 `immediate` 시작 모드 AWS DMS 만 지원합니다.
- AWS DMS 는 잘못된 UTF8 문자가 있는 문서를 지원하지 않습니다.
- AWS DMS 는 샤딩된 컬렉션을 지원하지 않습니다.

MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 모범 사례

- 동일한 MongoDB 인스턴스에 호스팅되는 여러 대규모 데이터베이스 및 컬렉션의 경우 각 데이터베이스 및 컬렉션에 대한 선택 규칙을 사용하여 여러 데이터 마이그레이션 작업과 프로젝트 간에 작업을 분할하는 것이 좋습니다. 데이터베이스 및 컬렉션 분할을 조정하여 성능을 극대화할 수 있습니다.

AWS DMS 스키마 변환에 사용할 대상 데이터베이스 생성 및 설정

MySQL 호환, PostgreSQL 및 Amazon DocumentDB 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션을 위한 대상 데이터 공급자로 사용할 수 있습니다.

지원되는 데이터베이스 버전은 [DMS 동종 데이터 마이그레이션이 지원하는 대상 데이터 공급자](#) 섹션을 참조하세요.

대상 데이터 공급자는 Amazon RDS DB 인스턴스 또는 Amazon Aurora DB 클러스터일 수 있습니다. 대상 데이터 공급자의 데이터베이스 버전은 소스 데이터 공급자의 데이터베이스 버전과 같거나 그 이후 버전이어야 합니다.

주제

- [에서 동종 데이터 마이그레이션의 대상으로 MySQL 호환 데이터베이스 사용 AWS DMS](#)
- [에서 동종 데이터 마이그레이션의 대상으로 PostgreSQL 데이터베이스 사용 AWS DMS](#)
- [에서 동종 데이터 마이그레이션의 대상으로 Amazon DocumentDB 데이터베이스 사용 AWS DMS](#)

에서 동종 데이터 마이그레이션의 대상으로 MySQL 호환 데이터베이스 사용 AWS DMS

MySQL 호환 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션의 대상으로 사용할 수 있습니다.

AWS DMS 대상 Amazon RDS for MySQL, MariaDB 또는 Amazon Aurora MySQL 데이터베이스로 데이터를 마이그레이션하려면 특정 권한이 필요합니다. 다음 스크립트를 사용하여 MySQL 대상 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다.

이 예제에서 각 `### ## ## ###`를 자신의 정보로 바꿉니다. 대상 MariaDB 데이터베이스 버전이 10.50보다 낮으면 GRANT SLAVE MONITOR 명령을 건너뛸 수 있습니다.

```
CREATE USER 'your_user'@'%' IDENTIFIED BY 'your_password';

GRANT ALTER, CREATE, DROP, INDEX, INSERT, UPDATE, DELETE, SELECT, CREATE VIEW, CREATE
ROUTINE, ALTER ROUTINE, EVENT, TRIGGER, EXECUTE, REFERENCES ON *.* TO 'your_user'@'%' ;
GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'your_user'@'%' ; GRANT SLAVE
MONITOR ON *.* TO 'your_user'@'%' ;
```

이전 예제에서는 자신의 정보로 각각의 `### ## ## ###`를 바꿉니다.

다음 스크립트를 사용하여 MariaDB 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다. 마이그레이션하는 모든 데이터베이스에 대해 GRANT 쿼리를 실행합니다 AWS.

```
CREATE USER 'your_user'@'%' IDENTIFIED BY 'your_password';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER, CREATE VIEW, CREATE
ROUTINE, ALTER ROUTINE, EVENT, TRIGGER, EXECUTE, SLAVE MONITOR, REPLICATION SLAVE ON
*.* TO 'your_user'@'%' ;
```

이전 예제에서는 자신의 정보로 각각의 `### ## ## ###`를 바꿉니다.

Note

Amazon RDS에서 MySQL/Maria 데이터베이스 인스턴스의 자동 백업을 활성화하면 바이너리 로깅도 활성화됩니다. 이러한 설정을 활성화하면 대상 데이터베이스에서 함수, 프로시저, 트리거와 같은 보조 객체를 생성하는 동안 다음 오류가 발생하여 데이터 마이그레이션 작업이 실패할 수 있습니다. 대상 데이터베이스에 바이너리 로깅이 활성화되어 있는 경우 작업을 시작하기 전에 데이터베이스 파라미터 그룹에서 `log_bin_trust_function_creators`를 `true`로 설정하세요.

```
ERROR 1419 (HY000): You don't have the SUPER privilege and binary logging is
enabled (you might want to use the less safe log_bin_trust_function_creators
variable)
```

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용하는 것에 대한 제한 사항

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용 시 적용되는 제한 사항은 다음과 같습니다.

- 데이터 소스에 연결하는 데 사용하는 사용자 이름에는 다음과 같은 제한이 있습니다.
 - 길이는 2~64자일 수 있습니다.
 - 스페이스를 포함할 수 없습니다.
 - a-z, A-Z, 0-9, 밑줄(_) 문자를 포함할 수 있습니다.
 - 하이픈(-)을 포함할 수 없습니다.
 - a-z 또는 A-Z로 시작해야 합니다.
- 데이터 소스에 연결하는 데 사용하는 암호에는 다음과 같은 제한이 있습니다.
 - 길이는 1~128자일 수 있습니다.
 - 작은따옴표('), 큰따옴표("), 세미콜론(;) 또는 스페이스는 포함할 수 없습니다.

에서 동종 데이터 마이그레이션의 대상으로 PostgreSQL 데이터베이스 사용 AWS DMS

PostgreSQL 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션의 대상으로 사용할 수 있습니다.

AWS DMS 대상 Amazon RDS for PostgreSQL 또는 Amazon Aurora PostgreSQL 데이터베이스로 데이터를 마이그레이션하려면 특정 권한이 필요합니다. 다음 스크립트를 사용하여 PostgreSQL 대상 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다.

```
CREATE USER your_user WITH LOGIN PASSWORD 'your_password';
GRANT USAGE ON SCHEMA schema_name TO your_user;
GRANT CONNECT ON DATABASE db_name TO your_user;
GRANT CREATE ON DATABASE db_name TO your_user;
GRANT CREATE ON SCHEMA schema_name TO your_user;
```

```
GRANT UPDATE, INSERT, SELECT, DELETE, TRUNCATE ON ALL TABLES IN SCHEMA schema_name
  TO your_user;
      #For "Full load and change data capture (CDC)" and "Change data capture
(CDC)" data migrations, setting up logical replication requires rds_superuser
privileges
GRANT rds_superuser TO your_user;
```

이전 예제에서는 자신의 정보로 각각의 `### ## ## ###`를 바꿉니다.

RDS for PostgreSQL 대상에 대한 논리적 복제를 활성화하려면 DB 파라미터 그룹의 `rds.logical_replication` 파라미터를 1로 설정합니다. 이 정적 파라미터를 적용하려면 DB 인스턴스 또는 DB 클러스터를 재부팅해야 합니다. 일부 파라미터는 정적이므로 서버를 다시 시작할 때까지 서버 start. AWS DMS ignores에서만 DB 파라미터 그룹의 항목에 대한 변경 사항을 설정할 수 있습니다.

PostgreSQL은 트리거를 사용하여 외래 키 제약 조건을 구현합니다. 전체 로드 단계에서는 각 테이블을 한 번에 하나씩 AWS DMS 로드합니다. 전체 로드 중에는 대상 데이터베이스의 외래 키 제약 조건을 해제하는 것이 좋습니다. 이렇게 하려면 다음 방법 중 하나를 사용하세요.

- 인스턴스에서 모든 트리거를 임시로 비활성화하고 전체 로드를 완료합니다.
- PostgreSQL 내에서 `session_replication_role` 파라미터의 값을 변경하세요.

해당 시점에 트리거는 `origin`, `replica`, `always` 또는 `disabled` 상태 중 하나일 수 있습니다. `session_replication_role` 파라미터를 `replica`로 설정하면 해당 `replica` 상태의 트리거만 활성화됩니다. 그렇지 않으면, 트리거가 비활성 상태로 유지됩니다.

PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용하는 것에 대한 제한 사항

PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용 시 적용되는 제한 사항은 다음과 같습니다.

- 데이터 소스에 연결하는 데 사용하는 사용자 이름에는 다음과 같은 제한이 있습니다.
 - 길이는 2~64자일 수 있습니다.
 - 스페이스를 포함할 수 없습니다.
 - a-z, A-Z, 0-9, 밑줄(_) 문자를 포함할 수 있습니다.
 - a-z 또는 A-Z로 시작해야 합니다.
- 데이터 소스에 연결하는 데 사용하는 암호에는 다음과 같은 제한이 있습니다.

- 길이는 1~128자일 수 있습니다.
- 작은따옴표('), 큰따옴표("), 세미콜론(;) 또는 스페이스는 포함할 수 없습니다.

에서 동종 데이터 마이그레이션의 대상으로 Amazon DocumentDB 데이터베이스 사용 AWS DMS

Amazon DocumentDB(MongoDB 호환) 데이터베이스와 DocumentDB Elastic 클러스터를 AWS DMS의 동종 데이터 마이그레이션을 위한 마이그레이션 대상으로 사용할 수 있습니다.

Amazon DocumentDB 대상에 대해 동종 데이터 마이그레이션을 실행하려면 관리자 권한이 있는 사용자 계정 또는 마이그레이션할 데이터베이스에 대해서만 읽기/쓰기 권한이 있는 사용자를 생성할 수 있습니다.

동종 데이터 마이그레이션은 Amazon DocumentDB가 지원하는 모든 BSON 데이터 유형을 지원합니다. 이러한 데이터 유형의 목록은 Amazon DocumentDB 개발자 안내서의 [데이터 유형](#)을 참조하세요.

소스에서 샤딩되지 않은 컬렉션을 마이그레이션하기 위해 DocumentDB Elastic 클러스터의 샤드 기능을 사용하려면 데이터 마이그레이션 작업을 시작하기 전에 마이그레이션할 샤드 컬렉션을 생성합니다. Amazon DocumentDB Elastic 클러스터의 샤드 컬렉션에 대한 자세한 내용은 Amazon DocumentDB 개발자 안내서의 [5단계: 컬렉션 샤드](#)를 참조하세요.

Amazon DocumentDB 대상의 경우는 none 또는 require SSL 모드를 AWS DMS 지원합니다.

에서 동종 데이터 마이그레이션 실행 AWS DMS

[동종 데이터베이스 마이그레이션](#)에서 AWS DMS 를 사용하여 소스 데이터베이스의 데이터를 Amazon Relational Database Service(RDS) , Amazon Aurora 또는 Amazon DocumentDB의 동등한 엔진으로 마이그레이션할 수 있습니다.는 소스 및 대상 데이터베이스의 기본 데이터베이스 도구를 사용하여 데이터 마이그레이션 프로세스를 AWS DMS 자동화합니다.

동종 데이터 마이그레이션을 위한 인스턴스 프로파일과 호환 가능한 데이터 공급자를 만든 후 마이그레이션 프로젝트를 생성하세요. 자세한 내용은 [마이그레이션 프로젝트 생성](#) 섹션을 참조하세요.

다음 섹션에서는 동종 데이터 마이그레이션을 생성하고 구성하며 실행하는 방법에 대해 설명합니다.

주제

- [에서 데이터 마이그레이션 생성 AWS DMS](#)

- [동종 데이터 마이그레이션을 위한 선택 규칙](#)
- [에서 데이터 마이그레이션 관리 AWS DMS](#)
- [에서 데이터 마이그레이션 모니터링 AWS DMS](#)
- [에서 동종 데이터 마이그레이션의 상태 AWS DMS](#)
- [에서 동종 데이터 마이그레이션을 사용하여 MySQL 데이터베이스에서 데이터 마이그레이션 AWS DMS](#)
- [에서 동종 데이터 마이그레이션을 사용하여 PostgreSQL 데이터베이스에서 데이터 마이그레이션 AWS DMS](#)
- [에서 동종 데이터 마이그레이션을 사용하여 MongoDB 데이터베이스에서 데이터 마이그레이션 AWS DMS](#)
- [대상 테이블 준비 모드](#)

에서 데이터 마이그레이션 생성 AWS DMS

동일한 유형의 호환 가능한 데이터 공급자와 함께 마이그레이션 프로젝트를 만든 후 이 프로젝트를 사용하여 동종 데이터 마이그레이션을 수행할 수 있습니다. 자세한 내용은 [마이그레이션 프로젝트 생성](#) 섹션을 참조하세요.

먼저 동종 데이터 마이그레이션을 사용하려면 새 데이터 마이그레이션을 생성하세요. 단일 마이그레이션 프로젝트에서 여러 유형의 동종 데이터 마이그레이션을 여러 개 생성할 수 있습니다.

AWS DMS 에는에 대해 생성할 수 있는 최대 동종 데이터 마이그레이션 수가 있습니다 AWS 계정. AWS DMS 서비스 할당량에 대한 자세한 내용은 다음 섹션을 참조하세요 [에 대한 할당량AWS Database Migration Service](#).

데이터 마이그레이션을 생성하기 전에 소스 및 대상 데이터베이스, IAM 정책 및 역할, 인스턴스 프로파일, 데이터 공급자와 같은 필수 리소스를 설정해야 합니다. 자세한 내용은 [IAM 리소스 생성](#), [인스턴스 프로파일 생성](#), [데이터 공급자 생성](#) 섹션을 참조하세요.

또한 상위 데이터베이스 버전에서 하위 데이터베이스 버전으로 데이터를 마이그레이션할 때는 동종 데이터 마이그레이션을 사용하지 않는 것이 좋습니다. 소스 및 대상 데이터 공급자에 사용하는 데이터베이스 버전을 확인하고 필요하다면 대상 데이터베이스 버전을 업그레이드하세요.

마이그레이션 작업을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택하고 데이터 마이그레이션 탭에서 데이터 마이그레이션 생성을 선택합니다.
4. 이름에서 데이터 마이그레이션 이름을 입력합니다. 데이터 마이그레이션을 쉽게 식별할 수 있도록 데이터 마이그레이션에 고유한 이름을 사용해야 합니다.
5. 복제 유형에서 구성하려는 데이터 마이그레이션 유형을 선택합니다. 다음 옵션 중 하나를 선택할 수 있습니다.
 - 전체 로드 - 기존 소스 데이터를 마이그레이션합니다.
 - 전체 로드 및 변경 데이터 캡처(CDC) - 기존 소스 데이터를 마이그레이션하고 지속적인 변경 내용을 복제합니다.
 - 변경 데이터 캡처(CDC) - 지속적인 변경 사항을 복제합니다.
6. Amazon CloudWatch에 데이터 마이그레이션 로그를 저장하기 위한 CloudWatch 로그 활성화 확인란을 선택합니다. 이 옵션을 선택하지 않은 경우, 데이터 마이그레이션 실패 시 로그 파일을 볼 수 없습니다.
7. (선택 사항) 고급 설정을 확장합니다. 작업 수에가 소스 데이터를 대상으로 마이그레이션하는 데 사용할 AWS DMS 수 있는 병렬 스레드 수를 입력합니다.
8. IAM 서비스 역할의 경우, 사전 요구 사항에서 생성한 IAM 역할을 선택합니다. 자세한 내용은 [에서 동종 데이터 마이그레이션을 위한 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.
9. 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션을 위한 시작 모드를 구성합니다. 다음 옵션 중 하나를 선택할 수 있습니다.
 - 즉시 - 데이터 마이그레이션을 시작하면 지속적 복제를 시작합니다.
 - 기본 시작점 사용 - 지정된 시점부터 지속적 복제를 시작합니다.

PostgreSQL 데이터베이스의 경우, 슬롯 이름에 논리적 복제 슬롯의 이름을 입력하고 기본 시작점에는 트랜잭션 로그 시퀀스 번호를 입력합니다.

MySQL 데이터베이스의 경우, 로그 시퀀스 번호(LSN)에 트랜잭션 로그 시퀀스 번호를 입력합니다.
10. 변경 데이터 캡처(CDC) 또는 전체 로드 및 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션에 대해 중지 모드를 구성합니다. 다음 옵션 중 하나를 선택할 수 있습니다.
 - CDC 중지 안 함 - 데이터 마이그레이션을 중지할 때까지 지속적인 복제를 AWS DMS 계속합니다.
 - 서버 시점 사용 - 지정된 시간에 진행 중인 복제를 AWS DMS 중지합니다.

이 옵션을 선택하는 경우, 지속적 복제를 자동으로 중지하려는 날짜와 시간을 중지 날짜 및 시간에 입력합니다.

11. 데이터 마이그레이션 생성을 선택합니다.

AWS DMS 는 데이터 마이그레이션을 생성하여 마이그레이션 프로젝트의 데이터 마이그레이션 탭에 있는 목록에 추가합니다. 여기에서 데이터 마이그레이션 상태를 확인할 수 있습니다. 자세한 내용은 [마이그레이션 상태](#) 단원을 참조하십시오.

Important

전체 로드 및 전체 로드 및 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션의 경우는 대상 데이터베이스의 모든 데이터, 테이블 및 기타 데이터베이스 객체를 AWS DMS 삭제합니다. 대상 데이터베이스의 백업이 있는지 확인하세요.

가 데이터 마이그레이션을 AWS DMS 생성하면 데이터 마이그레이션의 상태가 준비됨으로 설정됩니다. 데이터를 마이그레이션하려면 데이터 마이그레이션을 수동으로 시작해야 합니다. 이렇게 하려면 목록에서 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 시작을 선택합니다. 자세한 내용은 [데이터 마이그레이션 관리](#) 단원을 참조하십시오.

동종 데이터 마이그레이션을 처음 시작하려면 몇 가지 setup. AWS DMS creates가 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 이 프로세스는 최대 15분이 걸립니다. 데이터 마이그레이션을 중지했다가 다시 시작한 후에는 AWS DMS 가 환경을 다시 생성하지 않으므로 데이터 마이그레이션에 더 빠르게 액세스할 수 있습니다.

동종 데이터 마이그레이션을 위한 선택 규칙

선택 규칙을 사용하여 복제에 포함할 스키마, 테이블 또는 둘 다를 선택할 수 있습니다.

데이터 마이그레이션 태스크를 생성할 때 선택 규칙 추가를 선택합니다.

규칙 설정에 다음 값을 입력합니다.

- 스키마: 스키마 입력을 선택합니다.
- 스키마 이름: 복제할 스키마의 이름을 제공하거나 %를 와일드카드로 사용합니다.
- 테이블 이름: 복제할 테이블의 이름을 제공하거나 %를 와일드카드로 사용합니다.

기본적으로 DMS가 지원하는 유일한 규칙 작업은 Include이고 DMS가 지원하는 유일한 와일드카드 문자는 %입니다.

Note

동종 데이터 마이그레이션을 AWS DMS 위한 선택 규칙 지원은 소스 데이터베이스 엔진과 선택한 마이그레이션 유형의 조합에 따라 달라집니다. PostgreSQL 및 MongoDB 호환 소스는 모든 마이그레이션 유형에 대한 선택 규칙을 허용하는 반면, MySQL 소스는 전체 로드 마이그레이션 유형에 대한 선택 규칙만 지원합니다.

Example스키마에서 모든 테이블 마이그레이션

다음은 소스에서 이름이 dmsst인 스키마의 모든 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "dmsst",
        "table-name": "%"
      },
      "filters": [],
      "rule-id": "1",
      "rule-name": "1"
    }
  ]
}
```

Example스키마에서 일부 테이블 마이그레이션

다음은 소스에서 이름이 dmsst인 스키마에서 collectionTest로 시작하는 이름을 가진 모든 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```
{
  "rules": [
    {
```

```

        "rule-type": "selection",
        "rule-action": "include",
        "object-locator": {
            "schema-name": "dmsst",
            "table-name": "collectionTest%"
        },
        "filters": [],
        "rule-id": "1",
        "rule-name": "1"
    }
]
}

```

Example 여러 스키마에서 특정 테이블 마이그레이션

다음은 소스에서 이름이 dmsst 및 Test인 여러 스키마의 일부 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "dmsst",
        "table-name": "collectionTest1"
      },
      "filters": [],
      "rule-id": "1",
      "rule-name": "1"
    },
    {
      "rule-type": "selection",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "products"
      },
      "filters": [],
      "rule-id": "2",
      "rule-name": "2"
    }
  ]
}

```

```
}
```

에서 데이터 마이그레이션 관리 AWS DMS

데이터 마이그레이션을 생성한 후 AWS DMS 는 데이터 마이그레이션을 자동으로 시작하지 않습니다. 필요하다면 수동으로 데이터 마이그레이션을 시작합니다.

데이터 마이그레이션을 시작하기 전에 데이터 마이그레이션의 모든 설정을 수정할 수 있습니다. 데이터 마이그레이션을 시작한 후에는 복제 유형을 변경할 수 없습니다. 다른 복제 유형을 사용하려면 새 데이터 마이그레이션을 생성하세요.

데이터 마이그레이션을 시작하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택합니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 데이터 마이그레이션의 요약 페이지가 열립니다.
4. 작업에서 시작을 선택합니다.

그런 다음은 데이터 마이그레이션을 위한 서버리스 환경을 AWS DMS 생성합니다. 이 프로세스는 최대 15분이 걸립니다.

데이터 마이그레이션을 시작한 후는 상태를 시작 중으로 AWS DMS 설정합니다. 가 데이터 마이그레이션에 AWS DMS 사용하는 다음 상태는 데이터 마이그레이션 설정에서 선택하는 복제 유형에 따라 다릅니다. 자세한 내용은 [마이그레이션 상태](#) 단원을 참조하십시오.

마이그레이션 작업을 수정하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택합니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 데이터 마이그레이션의 요약 페이지가 열립니다.
4. 수정을 선택합니다.
5. 데이터 마이그레이션을 위한 설정을 구성합니다.

⚠ Important

데이터 마이그레이션을 시작한 후에는 복제 유형을 변경할 수 없습니다.

6. Amazon CloudWatch에서 데이터 마이그레이션 로그를 보려면 CloudWatch 로그 활성화 확인란을 선택합니다.
7. 변경 사항 저장을 선택합니다.

가 데이터 마이그레이션을 AWS DMS 시작한 후 중지할 수 있습니다. 중지하려면 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 중지를 선택합니다.

데이터 마이그레이션을 중지한 후는 상태를 중지 중으로 AWS DMS 설정합니다. 그런 다음, AWS DMS 는 이 데이터 마이그레이션의 상태를 중지됨으로 설정합니다. 가 데이터 마이그레이션을 AWS DMS 중지한 후 데이터 마이그레이션을 수정, 재개, 다시 시작 또는 삭제할 수 있습니다.

데이터 복제를 계속하려면 데이터 마이그레이션 탭에서 중지한 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 다시 시작 처리 중을 선택합니다.

데이터 로드를 다시 시작하려면 데이터 마이그레이션 탭에서 중지한 데이터 마이그레이션을 선택합니다. 그런 다음 작업에서 재시작을 선택합니다. 대상 데이터베이스에서 모든 데이터를 AWS DMS 삭제하고 처음부터 데이터 마이그레이션을 시작합니다.

중지했거나 아직 시작하지 않은 데이터 마이그레이션을 삭제할 수 있습니다. 데이터 마이그레이션을 삭제하려면 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 삭제를 선택합니다. 마이그레이션 프로젝트를 삭제하려면 모든 데이터 마이그레이션을 중지하고 삭제하세요.

에서 데이터 마이그레이션 모니터링 AWS DMS

동종 데이터 마이그레이션을 시작한 후 상태 및 진행 상황을 모니터링할 수 있습니다. 수백 기가바이트에 달하는 대규모 데이터세트의 데이터 마이그레이션은 완료하는 데 몇 시간이 걸립니다. 데이터 마이그레이션의 신뢰성, 가용성 및 고성능을 유지하려면 데이터 마이그레이션 진행 상태를 정기적으로 모니터링하세요.

데이터 마이그레이션의 상태 및 진행 상황을 확인하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택하고 데이터 마이그레이션 탭으로 이동합니다.
4. 데이터 마이그레이션에 대해서는 상태 열을 참조하세요. 이 열에 관한 자세한 내용은 [마이그레이션 상태](#)를 참조하세요.
5. 실행 중인 데이터 마이그레이션의 경우, 마이그레이션 진행 상황 열에 마이그레이션된 데이터의 비율(%)이 표시됩니다.

데이터 마이그레이션의 세부 정보를 확인하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택합니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다.
4. 세부 정보 탭에서 마이그레이션 진행 상황을 확인할 수 있습니다. 특히 다음 지표를 확인할 수 있습니다.
 - 퍼블릭 IP 주소 - 데이터 마이그레이션의 퍼블릭 IP 주소입니다. 네트워크를 구성하려면 이 값이 필요합니다. 자세한 내용은 [네트워크 설정](#) 섹션을 참조하세요.
 - 로드된 테이블 - 성공적으로 로드된 테이블의 수입입니다.
 - 로드 중인 테이블 - 현재 로드 중인 테이블의 수입입니다.
 - 대기 중인 테이블 - 현재 로드 대기 중인 테이블의 수입입니다.
 - 오류 발생 테이블 - 로드 실패한 테이블의 수입입니다.
 - 경과 시간 - 데이터 마이그레이션이 시작된 후 경과한 시간입니다.
 - CDC 지연 시간 - 소스 테이블에서 변경이 발생한 시점과가이 변경 사항을 대상 테이블에 AWS DMS 적용한 시점 사이의 평균 시간입니다.
 - 마이그레이션 시작됨 - 이 데이터 마이그레이션을 시작한 시간입니다.
 - 마이그레이션 중지됨 - 이 데이터 마이그레이션을 중지한 시간입니다.
5. 데이터 마이그레이션에 대한 로그 파일을 보려면 동종 데이터 마이그레이션 설정에서 CloudWatch 로그 보기를 선택합니다. 데이터 마이그레이션을 생성하거나 수정할 때 CloudWatch를 활성화할 수 있습니다. 자세한 내용은 [데이터 마이그레이션 생성](#) 및 [데이터 마이그레이션 관리](#)를 참조하세요.

Amazon CloudWatch 경보 또는 이벤트를 사용하여 데이터 마이그레이션을 더욱 면밀하게 추적할 수 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 [Amazon CloudWatch](#), [Amazon CloudWatch Events](#) 및 [Amazon CloudWatch Logs](#)란 무엇입니까?를 참조하세요. Amazon CloudWatch 사용 시 비용이 청구됩니다.

동종 데이터 마이그레이션의 경우 Amazon CloudWatch에 다음 지표가 AWS DMS 포함됩니다.

지표	설명
OverallCDCLatency	CDC 단계 중 전체 지연 시간. MySQL 데이터베이스의 경우, 이 지표는 원본 바이너리 로그의 변경 시점부터 이 변경 사항의 복제 시점까지 경과한 시간(초)을 나타냅니다. PostgreSQL 데이터베이스의 경우, 이 지표는 pg_stat_subscription 뷰의 last_msg_receipt_time 부터 last_msg_send_time 까지 경과한 시간(초)을 나타냅니다. 단위: 초
StorageConsumption	데이터 마이그레이션에 사용되는 스토리지. 단위: 바이트

에서 동종 데이터 마이그레이션의 상태 AWS DMS

실행하는 각 데이터 마이그레이션에 대해 AWS DMS 콘솔에 상태가 AWS DMS 표시됩니다. 다음 목록은 사용 가능한 상태를 포함합니다.

- **Creating** - AWS DMS 가 데이터 마이그레이션을 생성하고 있습니다.
- **Ready** - 데이터 마이그레이션을 시작할 준비가 되었습니다.
- **Starting** - AWS DMS 는 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 이 프로세스는 최대 15분이 걸립니다.
- **Load running** - AWS DMS 가 전체 로드 마이그레이션을 수행하고 있습니다.
- **Load complete, replication ongoing** - 전체 로드를 AWS DMS 완료하고 현재 진행 중인 변경 사항을 복제합니다.는 전체 로드 및 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션에만 이 상태를 AWS DMS 사용합니다.

- **Replication ongoing** - AWS DMS 진행 중인 변경 사항을 복제하고 있습니다.는 변경 데이터 캡처(CDC) 유형의 마이그레이션에만이 상태를 AWS DMS 사용합니다.
- **Reloading target** - 데이터 마이그레이션을 AWS DMS 다시 시작하고 지정된 마이그레이션 유형을 수행합니다.
- **Stopping** - AWS DMS 가 데이터 마이그레이션을 중지하고 있습니다. 작업 메뉴에서 데이터 마이그레이션을 중지하도록 선택한 후이 상태를 AWS DMS 설정합니다.
- **Stopped** - AWS DMS 가 데이터 마이그레이션을 중지했습니다.
- **Failed** - 데이터 마이그레이션이 실패했습니다. 자세한 내용은 로그 파일을 참조하세요.

로그 파일을 보려면 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 그런 다음, 동종 데이터 마이그레이션 설정에서 CloudWatch 로그 보기를 선택합니다.

Important

데이터 마이그레이션을 생성할 때 CloudWatch 로그 활성화 확인란을 선택하면 로그 파일을 볼 수 있습니다.

- **Deleting** - AWS DMS 는 데이터 마이그레이션을 삭제합니다. 작업 메뉴에서 데이터 마이그레이션을 삭제하도록 선택한 후이 상태를 AWS DMS 설정합니다.
- **Maintenance** - 데이터 마이그레이션 작업과 연결된 기본 서버리스 컨테이너에 새 이미지가 배포될 때 작업을 유지 관리 모드 상태로 AWS DMS 둡니다.

에서 동종 데이터 마이그레이션을 사용하여 MySQL 데이터베이스에서 데이터 마이그레이션 AWS DMS

[동종 데이터베이스 마이그레이션](#)를 사용하여 자체 관리형 MySQL 데이터베이스를 RDS for MySQL 또는 RDS for Aurora MySQL로 마이그레이션할 수 있습니다. AWS DMS 는 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 다양한 유형의 데이터 마이그레이션의 경우는 다양한 기본 MySQL 데이터베이스 도구를 AWS DMS 사용합니다.

전체 로드 유형의 동종 데이터 마이그레이션의 경우 mydumper를 AWS DMS 사용하여 소스 데이터베이스에서 데이터를 읽고 서버리스 환경에 연결된 디스크에 저장합니다. 가 모든 소스 데이터를 AWS DMS 읽은 후 대상 데이터베이스의 myloader를 사용하여 데이터를 복원합니다.

전체 로드 및 변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션의 경우 mydumper를 AWS DMS 사용하여 소스 데이터베이스에서 데이터를 읽고 서버리스 환경에 연결된 디스크에 저장합니다.

가 모든 소스 데이터를 AWS DMS 읽은 후 대상 데이터베이스의 myloader를 사용하여 데이터를 복원합니다. 가 전체 로드를 AWS DMS 완료한 후 binlog 위치를 전체 로드의 시작으로 설정하여 binlog 복제를 설정합니다.

변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션에서 AWS DMS 는 복제를 시작하기 위한 기본 CDC 시작점이 필요합니다. 네이티브 CDC 시작점을 제공하면는 해당 지점의 변경 사항을 AWS DMS 캡처합니다. 또는 데이터 마이그레이션 설정에서 즉시를 선택하여 실제 데이터 마이그레이션이 시작될 때 복제 시작점을 자동으로 캡처할 수도 있습니다.

 Note

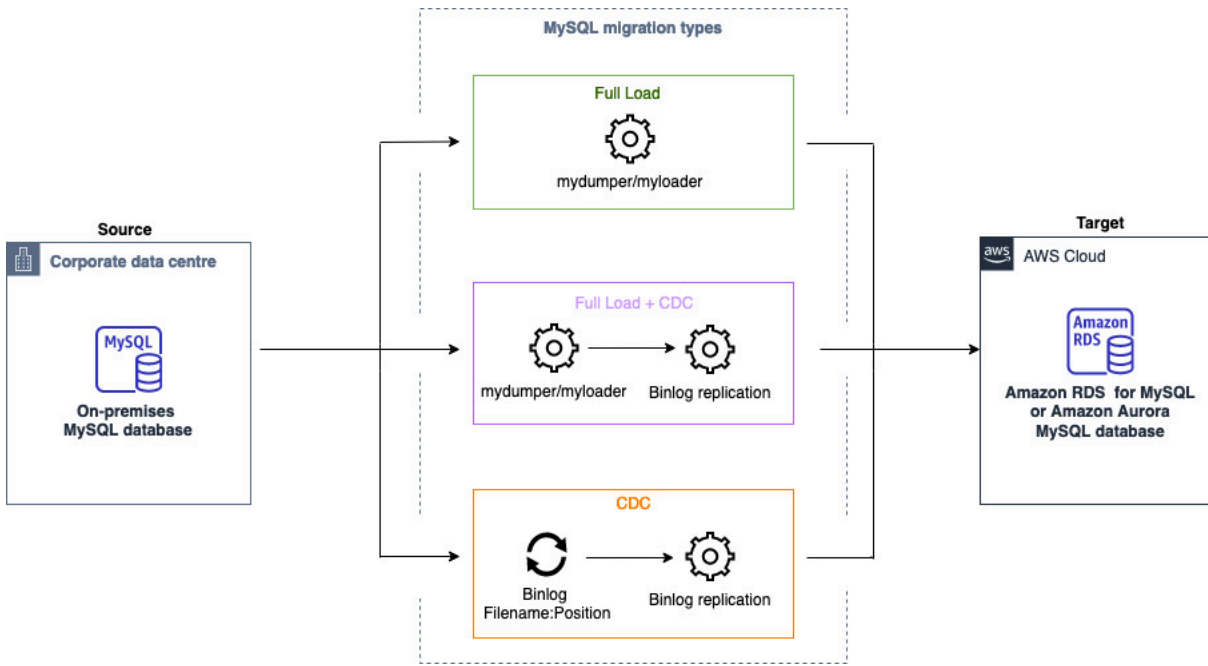
CDC 전용 마이그레이션이 제대로 작동하려면 모든 소스 데이터베이스 스키마와 객체가 대상 데이터베이스에 이미 있어야 합니다. 그러나 대상에는 원본에 없는 객체가 있을 수 있습니다.

다음 코드 예제를 사용하여 MySQL 데이터베이스의 현재 로그 시퀀스 번호(LSN)를 가져올 수 있습니다.

```
show master status
```

이 쿼리는 binlog 파일 이름과 위치를 반환합니다. 기본 시작점의 경우, binlog 파일 이름과 위치의 조합을 사용합니다. 예: mysql-bin-changelog.000024:373. 이 예제에서 mysql-bin-changelog.000024는 binlog 파일 이름이고 373는가 변경 사항 캡처를 AWS DMS 시작하는 위치입니다.

다음 다이어그램은에서 동종 데이터 마이그레이션 AWS DMS 을 사용하여 MySQL 데이터베이스를 RDS for MySQL 또는 Aurora MySQL로 마이그레이션하는 프로세스를 보여줍니다.



에서 동종 데이터 마이그레이션을 사용하여 PostgreSQL 데이터베이스에서 데이터 마이그레이션 AWS DMS

[동종 데이터베이스 마이그레이션](#)를 사용하여 자체 관리형 PostgreSQL 데이터베이스를 RDS for PostgreSQL 또는 RDS for Aurora PostgreSQL로 마이그레이션할 수 있습니다. AWS DMS 는 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 다양한 유형의 데이터 마이그레이션에서 AWS DMS 는 여러 가지 기본 PostgreSQL 데이터베이스 도구를 사용합니다.

전체 로드 유형의 동종 데이터 마이그레이션의 경우 pg_dump를 AWS DMS 사용하여 소스 데이터베이스에서 데이터를 읽고 서버리스 환경에 연결된 디스크에 저장합니다. 가 모든 소스 데이터를 AWS DMS 읽은 후 대상 데이터베이스의 pg_restore를 사용하여 데이터를 복원합니다.

전체 로드 및 변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션의 경우는 pg_dump를 AWS DMS 사용하여 소스 데이터베이스의 테이블 데이터 없이 스키마 객체를 읽고 서버리스 환경에 연결된 디스크에 저장합니다. 그런 다음 대상 데이터베이스에서 pg_restore를 사용하여 스키마 객체를 복원합니다. 가 pg_restore 프로세스를 AWS DMS 완료하면 원본 데이터베이스에서 대상 데이터베이스로 직접 초기 테이블 데이터를 복사하는 Initial Data Synchronization 옵션과 함께 논리적 복제를 위해 게시자 및 구독자 모델로 자동 전환된 다음 지속적인 복제를 시작합니다. 이 모델에서는 한 명 이상의 구독자가 게시자 노드에 있는 하나 이상의 발행물을 구독합니다.

변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션의 경우 복제를 시작하려면 네이티브 시작점이 AWS DMS 필요합니다. 기본 시작점을 제공하면가 해당 지점의 변경 사항을 AWS DMS 캡처합니

다. 또는 데이터 마이그레이션 설정에서 즉시를 선택하여 실제 데이터 마이그레이션이 시작될 때 복제 시작점을 자동으로 캡처할 수도 있습니다.

 Note

CDC 전용 마이그레이션이 제대로 작동하려면 모든 소스 데이터베이스 스키마와 객체가 대상 데이터베이스에 이미 있어야 합니다. 그러나 대상에는 원본에 없는 객체가 있을 수 있습니다.

다음 코드 예제를 사용하여 PostgreSQL 데이터베이스의 기본 시작점을 가져올 수 있습니다.

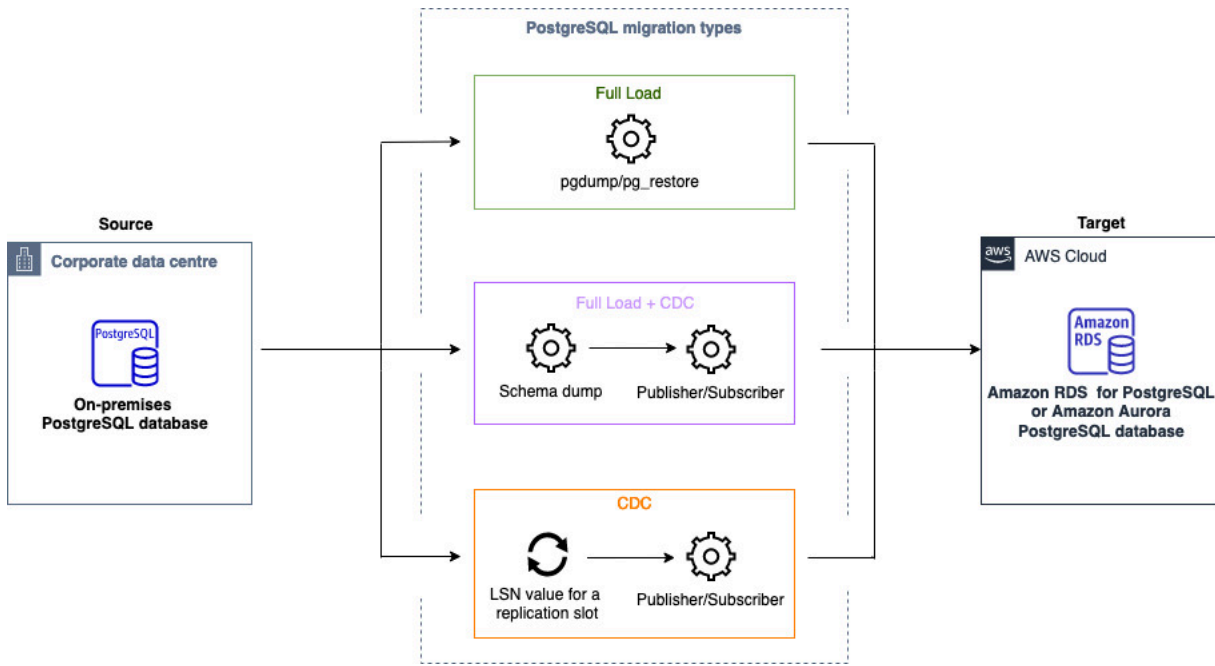
```
select confirmed_flush_lsn from pg_replication_slots where  
slot_name='migrate_to_target';
```

이 쿼리는 PostgreSQL 데이터베이스의 pg_replication_slots 뷰를 사용하여 로그 시퀀스 번호 (LSN) 값을 캡처합니다.

가 PostgreSQL 동종 데이터 마이그레이션의 상태를 중지됨, 실패 또는 삭제됨으로 AWS DMS 설정한 후에는 게시자와 복제가 제거되지 않습니다. 마이그레이션을 재개하지 않으려면 다음 명령을 사용하여 복제 슬롯과 게시자를 삭제하세요.

```
SELECT pg_drop_replication_slot('migration_subscriber_{ARN}');  
DROP PUBLICATION publication_{ARN};
```

다음 다이어그램은 동종 데이터 마이그레이션 AWS DMS 을 사용하여 PostgreSQL 데이터베이스를 RDS for PostgreSQL 또는 Aurora PostgreSQL로 마이그레이션하는 프로세스를 보여줍니다.



PostgreSQL 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 모범 사례

- FLCDC 태스크에 대한 구독자 측의 초기 데이터 동기화 속도를 높이려면 `max_logical_replication_workers` 및 `max_sync_workers_per_subscription`을 조정해야 합니다. 이러한 값을 늘리면 테이블 동기화 속도가 향상됩니다.
- `max_logical_replication_workers` - 논리적 복제 작업자의 최대 수를 지정합니다. 여기에는 구독자 측의 적용 작업자와 테이블 동기화 작업자가 모두 포함됩니다.
- `max_sync_workers_per_subscription` - `max_sync_workers_per_subscription`을 늘리면 테이블당 작업자 수가 아니라 병렬로 동기화된 테이블 수에만 영향을 줍니다.

Note

`max_logical_replication_workers`는 `max_worker_processes`를 초과해서는 안 되며, `max_sync_workers_per_subscription`은 `max_logical_replication_workers`보다 작거나 같아야 합니다.

- 대규모 테이블을 마이그레이션하려면 선택 규칙을 사용하여 테이블을 별도의 태스크로 나누는 것이 좋습니다. 예를 들어 대규모 테이블을 별도의 개별 태스크로 나누고 작은 테이블을 다른 단일 태스크로 나눌 수 있습니다.
- 구독자 측의 디스크 및 CPU 사용량을 모니터링하여 최적의 성능을 유지합니다.

에서 동종 데이터 마이그레이션을 사용하여 MongoDB 데이터베이스에서 데이터 마이그레이션 AWS DMS

[동종 데이터베이스 마이그레이션](#)을 사용하여 자체 관리형 MongoDB 데이터베이스를 Amazon DocumentDB로 마이그레이션할 수 있습니다. AWS DMS 는 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 다양한 유형의 데이터 마이그레이션에서 AWS DMS 는 여러 가지 기본 MongoDB 데이터베이스 도구를 사용합니다.

전체 로드 유형의 동종 데이터 마이그레이션의 경우는 `mongodump`를 AWS DMS 사용하여 소스 데이터베이스에서 데이터를 읽고 서버리스 환경에 연결된 디스크에 저장합니다. 가 모든 소스 데이터를 AWS DMS 읽은 후 대상 데이터베이스 `mongorestore`에서를 사용하여 데이터를 복원합니다.

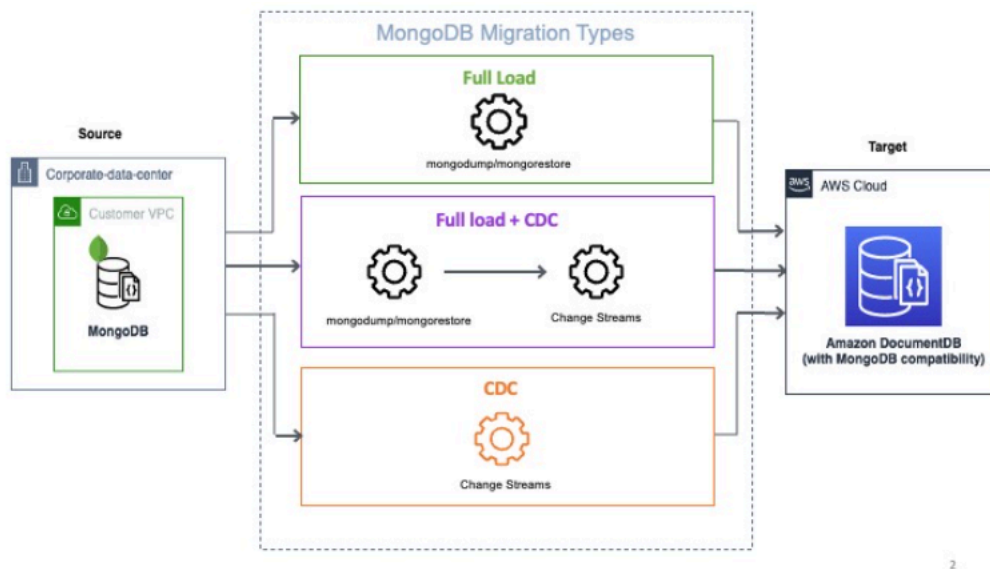
전체 로드 및 변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션의 경우는 `mongodump`를 AWS DMS 사용하여 소스 데이터베이스에서 데이터를 읽고 서버리스 환경에 연결된 디스크에 저장합니다. 가 모든 소스 데이터를 AWS DMS 읽은 후 대상 데이터베이스 `mongorestore`에서를 사용하여 데이터를 복원합니다. 가 전체 로드를 AWS DMS 완료하면 논리적 복제를 위해 게시자 및 구독자 모델로 자동 전환됩니다. 이 모델에서는 24시간 이상 변경 사항이 유지되도록 `oplog`의 크기를 조정하는 것이 좋습니다.

변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션의 경우 데이터 마이그레이션 설정에서 `immediately`를 선택하여 실제 데이터 마이그레이션이 시작될 때 복제 시작점을 자동으로 캡처합니다.

Note

새 컬렉션 또는 이름이 변경된 컬렉션의 경우 해당 컬렉션에 대해 동종 데이터 마이그레이션으로 새 데이터 마이그레이션 태스크를 생성해야 합니다. MongoDB 호환 소스의 경우 `create`, 및 `rename drop collection` 작업을 지원하지 AWS DMS 않습니다.

다음 다이어그램은에서 동종 데이터 마이그레이션 AWS DMS 을 사용하여 MongoDB 데이터베이스를 Amazon DocumentDB로 마이그레이션하는 프로세스를 보여줍니다.



대상 테이블 준비 모드

PostgreSQL, MongoDB 및 Amazon DocumentDB 마이그레이션용 AWS DMS 콘솔의 고급 설정 탭에서 데이터 마이그레이션 생성 작업을 선택하면 대상 테이블 준비 모드를 선택할 수 있습니다.

대상에 테이블 삭제

대상 모드의 테이블 삭제에서 AWS DMS 동종 마이그레이션은 마이그레이션을 시작하기 전에 대상 테이블을 삭제하고 다시 생성합니다. 이 접근 방식을 사용하면 마이그레이션 시작 시 대상 테이블이 비어 있습니다. 동종 마이그레이션 중에는 효율적인 데이터 마이그레이션을 위해 데이터를 로드하기 전에 소스 테이블 메타데이터에 정의된 인덱스를 포함한 모든 보조 객체를 AWS DMS 생성합니다.

대상에서 테이블 삭제 모드를 사용하는 경우 대상 데이터베이스를 구성해야 할 수 있습니다. 예를 들어 PostgreSQL 대상을 사용하면 보안상의 이유로 스키마 사용자를 생성할 AWS DMS 수 없습니다. 이 경우 소스와 일치하도록 스키마 사용자를 미리 생성하여가 마이그레이션이 시작될 때 테이블을 AWS DMS 생성하고 소스와 유사한 역할에 할당할 수 있도록 해야 합니다.

자르기

잘라내기 모드에서 AWS DMS 동종 마이그레이션은 마이그레이션이 시작되기 전에 모든 기존 대상 테이블을 잘라냅니다. 이렇게 하면 테이블 구조가 보존됩니다. 이 모드는 대상 스키마가 미리 생성되는 전체 로드 또는 전체 로드 및 CDC 마이그레이션에 적합합니다. Amazon DocumentDB 대상의 경우 컬렉션이 없는 경우 AWS DMS 는 마이그레이션 중에 인덱스 없이 컬렉션을 생성합니다.

아무 작업 안 함

아무 작업 안 함 모드에서 AWS DMS 동종 마이그레이션은 대상 테이블이 미리 생성되었다고 가정합니다. 대상 테이블이 비어 있지 않으면 마이그레이션 중에 데이터 충돌이 발생하여 DMS 작업 오류가 발생할 수 있습니다. 이 모드에서는 테이블 구조가 변경되지 않고 기존 데이터가 보존됩니다. 아무 작업 안 함 모드는 소스에서 대상 테이블로 백필 작업이 완료된 상태에서 CDC 작업만 수행하고 지속적인 복제를 통해 소스와 대상을 동기화해야 하는 경우에 적합합니다. Amazon DocumentDB 대상의 경우 컬렉션이 없는 경우는 보조 인덱스 없이 컬렉션을 AWS DMS 생성합니다. 또한 MongoDB 샤딩된 컬렉션에서 Amazon DocumentDB로 데이터를 마이그레이션할 때 전체 로드 단계에서 아무 작업 안 함 모드를 사용할 수 있습니다.

AWS DMS에서 동종 데이터 마이그레이션 문제 해결

AWS DMS에서 동종 데이터 마이그레이션과 관련하여 문제가 발생할 때 취해야 할 조치는 다음 목록에서 확인할 수 있습니다.

주제

- [AWS DMS에서 동종 데이터 마이그레이션을 생성할 수 없습니다.](#)
- [AWS DMS에서 동종 데이터 마이그레이션을 생성할 수 없습니다.](#)
- [AWS DMS에서 데이터 마이그레이션을 실행할 때 대상 데이터베이스에 연결할 수 없습니다.](#)
- [AWS DMS는 PostgreSQL에서 뷰를 테이블로 마이그레이션합니다.](#)

AWS DMS에서 동종 데이터 마이그레이션을 생성할 수 없습니다.

데이터 마이그레이션 생성을 선택한 후 AWS DMS가 데이터 공급자에 연결할 수 없다는 오류 메시지가 표시되면 필요한 IAM 역할을 구성했는지 확인하세요. 자세한 내용은 [IAM 역할 생성](#) 섹션을 참조하세요.

IAM 역할을 구성했는데도 이 오류 메시지가 계속 표시되면 AWS KMS 키 구성에서 이 IAM 역할을 키 사용자에게 추가하세요. 자세한 내용은 AWS Key Management Service 개발자 안내서에서 [키 사용자가 KMS 키를 사용하도록 허용](#) 섹션을 참조하세요.

AWS DMS에서 동종 데이터 마이그레이션을 생성할 수 없습니다.

마이그레이션 프로젝트에서 데이터 마이그레이션을 시작할 때 Failed 상태가 표시되면 소스 및 대상 데이터 공급자의 버전을 확인하세요. 버전을 확인하려면 MySQL 또는 PostgreSQL 데이터베이스에서 `SELECT VERSION();` 쿼리를 실행하십시오. 지원되는 데이터베이스 버전을 사용해야 합니다.

지원되는 소스 데이터베이스의 목록은 [DMS 동종 데이터 마이그레이션이 지원하는 소스](#) 섹션을 참조하세요.

지원되는 데이터 형식의 목록은 [DMS 동종 데이터 마이그레이션이 지원하는 대상](#) 섹션을 참조하세요.

지원되지 않는 데이터베이스 버전을 사용하는 경우, 소스 또는 대상 데이터베이스를 업그레이드하고 다시 시도하십시오.

AWS DMS 콘솔에서 데이터 마이그레이션에 대한 오류 메시지를 확인하세요. 이렇게 하려면 마이그레이션 프로젝트 열고 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 일반 아래에서 마지막 실패 메시지를 확인합니다.

마지막으로 CloudWatch 로그를 분석하십시오. 이렇게 하려면 마이그레이션 프로젝트를 열고 데이터 마이그레이션을 선택합니다. 세부 정보 탭에서 CloudWatch 로그 보기를 선택합니다.

AWS DMS에서 데이터 마이그레이션을 실행할 때 대상 데이터베이스에 연결할 수 없습니다.

대상에 연결할 수 없음 오류 메시지가 표시되면 다음 작업을 수행하십시오.

1. 소스 및 대상 데이터베이스에 연결된 보안 그룹에 모든 인바운드 및 아웃바운드 트래픽에 대한 규칙이 포함되어 있는지 확인하십시오. 자세한 내용은 [the section called “지속적 복제”](#) 섹션을 참조하세요.
2. 네트워크 액세스 제어 목록(ACL)과 라우팅 테이블 규칙을 확인하십시오.
3. 생성한 VPC에서 데이터베이스에 액세스할 수 있어야 합니다. VPC 보안 그룹에 퍼블릭 IP 주소를 추가하고 방화벽에서 입력 연결을 허용합니다.
4. 마이그레이션 프로젝트의 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 연결 및 보안에서 퍼블릭 IP 주소를 기록해 둡니다. 그런 다음, 소스 및 대상 데이터베이스에서 데이터 마이그레이션의 퍼블릭 IP 주소를 통한 액세스를 허용하십시오.
5. 지속적인 데이터 복제를 위해서는 소스 및 대상 데이터베이스가 서로 통신할 수 있어야 합니다.

자세한 내용을 알아보려면 Amazon Virtual Private Cloud 사용 설명서의 [보안 그룹을 사용하여 리소스에 대한 트래픽 제어](#)를 참조하세요.

AWS DMS는 PostgreSQL에서 뷰를 테이블로 마이그레이션합니다.

동종 데이터 마이그레이션은 PostgreSQL에서 뷰를 뷰로 마이그레이션하는 것을 지원하지 않습니다. PostgreSQL에서 AWS DMS는 뷰를 테이블로 마이그레이션합니다.

에서 데이터 공급자, 인스턴스 프로파일 및 마이그레이션 프로젝트 작업 AWS DMS

에서 DMS Schema Conversion 및 동종 데이터 마이그레이션을 사용하는 경우 마이그레이션 프로젝트로 작업 AWS Database Migration Service합니다. AWS DMS 마이그레이션 프로젝트에서는 서브넷 그룹, 인스턴스 프로파일, 데이터 공급자를 사용하게 됩니다.

서브넷은 VPC의 IP 주소 범위입니다. 복제 서브넷 그룹에는 인스턴스 프로파일에서 사용할 수 있는 다양한 가용 영역의 서브넷이 포함됩니다. 복제 서브넷 그룹은 DMS 리소스이며 Amazon VPC 및 Amazon RDS가 사용하는 서브넷 그룹과는 다릅니다.

인스턴스 프로파일은 마이그레이션 프로젝트가 실행되는 서버리스 환경의 네트워크 및 보안 설정을 지정합니다.

데이터 공급자는 데이터 저장소 유형과 데이터베이스에 대한 위치 정보를 저장합니다. 마이그레이션 프로젝트에 데이터 공급자를 추가한 후 from AWS Secrets Manager. AWS DMS uses의 데이터베이스 자격 증명을 제공하여 데이터베이스에 연결합니다.

데이터 공급자, 인스턴스 프로파일 및 기타 AWS 리소스를 생성한 후 마이그레이션 프로젝트를 생성할 수 있습니다. 마이그레이션 프로젝트는 인스턴스 프로파일, 소스 및 대상 데이터 공급자, 보안 암호를 설명합니다 AWS Secrets Manager. 다양한 소스 및 대상 데이터 공급자에 대한 여러 마이그레이션 프로젝트를 만들 수 있습니다.

대부분의 작업은 마이그레이션 프로젝트에서 수행합니다. DMS Schema Conversion의 경우 마이그레이션 프로젝트를 사용하여 소스 데이터 공급자의 객체를 평가하고 대상 데이터베이스와 호환되는 형식으로 변환합니다. 그런 다음 변환된 코드를 대상 데이터 공급자에 적용하거나 SQL 스크립트로 저장할 수 있습니다. 동종 데이터 마이그레이션의 경우 마이그레이션 프로젝트를 사용하여 소스 데이터베이스의 데이터를 AWS 클라우드에서 동일한 유형의 대상 데이터베이스로 마이그레이션합니다.

의 마이그레이션 프로젝트 AWS DMS 는 서버리스 전용입니다.는 마이그레이션 프로젝트에 클라우드 리소스를 AWS DMS 자동으로 프로비저닝합니다.

AWS DMS 에는에 대해 생성할 수 있는 인스턴스 프로파일, 데이터 공급자 및 마이그레이션 프로젝트의 최대 수가 있습니다 AWS 계정. AWS DMS 서비스 할당량 [에 대한 할당량AWS Database Migration Service](#)에 관한 자세한 내용은 다음 섹션을 참조하세요.

주제

- [AWS DMS 마이그레이션 프로젝트에 대한 서브넷 그룹 생성](#)

- [에 대한 인스턴스 프로파일 생성 AWS Database Migration Service](#)
- [에서 데이터 공급자 생성 AWS Database Migration Service](#)
- [에서 마이그레이션 프로젝트 생성 AWS Database Migration Service](#)
- [에서 마이그레이션 프로젝트 관리 AWS Database Migration Service](#)

AWS DMS 마이그레이션 프로젝트에 대한 서브넷 그룹 생성

인스턴스 프로파일을 생성하기 전에 인스턴스 프로파일의 서브넷 그룹을 구성하세요.

서브넷 그룹 생성

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 서브넷을 선택한 다음 서브넷 생성을 선택합니다.
3. 이름에 서브넷 그룹의 고유한 이름을 입력합니다.
4. 설명에 서브넷 그룹에 대한 설명을 입력합니다.
5. VPC에서 두 개 이상의 가용 영역에 하나 이상의 서브넷이 있는 VPC를 선택합니다.
6. 서브넷 추가에서 서브넷 그룹에 포함할 서브넷을 선택합니다. 두 개 이상의 가용 영역에서 서브넷을 선택해야 합니다.

Amazon RDS 데이터베이스에 연결하려면 서브넷 그룹에 퍼블릭 서브넷을 추가합니다. 온프레미스 데이터베이스에 연결하려면 서브넷 그룹에 프라이빗 서브넷을 추가합니다.

7. [서브넷 그룹 생성]을 선택합니다.

에 대한 인스턴스 프로파일 생성 AWS Database Migration Service

AWS DMS 콘솔에서 여러 인스턴스 프로파일을 생성할 수 있습니다. AWS DMS에서 생성하는 마이그레이션 프로젝트마다 사용할 인스턴스 프로파일을 하나 선택해야 합니다.

인스턴스 프로파일 생성

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 인스턴스 프로파일을 선택합니다.
3. 인스턴스 프로파일 생성을 선택합니다.

4. 인스턴스 프로파일 생성 페이지에서 인스턴스 프로파일의 이름을 설명하는 값을 입력합니다.
5. 네트워크 유형에서, IPv4 및 IPv6 주소 지정을 지원하는 인스턴스 프로파일을 생성하려면 이중 스택 모드를 선택합니다. IPv4 주소 지정만 지원하는 인스턴스 프로파일을 생성하려면 기본 옵션을 그대로 유지합니다.
6. 다음으로, 가상 프라이빗 클라우드(VPC)를 선택하여 선택한 네트워크 유형의 인스턴스를 실행합니다. 그런 다음 인스턴스 프로파일의 서브넷 그룹 및 VPC 보안 그룹을 선택합니다.

Amazon RDS 데이터베이스에 연결하려면 퍼블릭이든 프라이빗이든 RDS 설정에 적합한 서브넷 구성을 사용합니다. 온프레미스 데이터베이스에 연결하는 경우 프라이빗 서브넷 그룹을 사용하고 NAT 게이트웨이의 퍼블릭 IP 주소를 통해 소스 온프레미스 데이터베이스에 대한 AWS DMS 액세스를 허용하도록 네트워크가 구성되어 있는지 확인합니다. 자세한 내용은 [Amazon VPC를 기반으로 VPC 생성](#) 단원을 참조하십시오.

7. (선택 사항) DMS Schema Conversion용 마이그레이션 프로젝트를 생성한 다음, 스키마 변환 설정 - 선택 사항에서 마이그레이션 프로젝트의 정보를 저장할 Amazon S3 버킷을 선택합니다. 그런 다음이 Amazon S3 버킷에 대한 액세스를 제공하는 AWS Identity and Access Management (IAM) 역할을 선택합니다. 자세한 내용은 [Amazon S3 버킷 생성](#) 단원을 참조하십시오.
8. 인스턴스 프로파일 생성을 선택합니다.

생성한 인스턴스 프로파일은 수정하거나 삭제할 수 있습니다.

인스턴스 프로파일을 수정하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 인스턴스 프로파일을 선택합니다. 인스턴스 프로파일 페이지가 열립니다.
3. 인스턴스 프로파일을 선택한 다음 수정을 선택합니다.
4. 인스턴스 프로파일의 이름을 업데이트하고, VPC 또는 Amazon S3 버킷 설정을 편집합니다.
5. 변경 사항 저장을 선택합니다.

인스턴스 프로파일을 삭제하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 인스턴스 프로파일을 선택합니다. 인스턴스 프로파일 페이지가 열립니다.
3. 인스턴스 프로파일을 선택한 다음 삭제를 선택합니다.

4. 삭제를 선택하여 선택을 확인합니다.

에서 데이터 공급자 생성 AWS Database Migration Service

데이터 공급자를 생성하여 AWS DMS 마이그레이션 프로젝트에서 사용할 수 있습니다. 데이터베이스 공급자는 온프레미스 또는 Amazon EC2 인스턴스에서 실행되는 자체 관리형 엔진일 수 있습니다. 또한 데이터 공급자는 Amazon Relational Database Service(Amazon RDS) 또는 Amazon Aurora와 같은 완전관리형 엔진일 수 있습니다.

각 데이터베이스에 대해 단일 데이터 공급자를 생성할 수 있습니다. 여러 마이그레이션 프로젝트에서 하나의 데이터 공급자를 사용할 수 있습니다.

마이그레이션 프로젝트를 생성하려면 데이터 공급자가 두 개 이상 생성되어 있어야 합니다. 데이터 공급자 중 하나는 AWS 서비스에 있어야 합니다. AWS DMS 를 사용하여 스키마를 변환하거나 데이터를 온프레미스 데이터베이스로 마이그레이션할 수 없습니다.

다음 절차에서는 AWS DMS 콘솔 마법사에서 데이터 공급자를 생성하는 방법을 보여줍니다.

데이터 공급자를 생성하려면

1. 에 로그인 AWS Management Console한 다음 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 데이터 공급자를 선택합니다. 데이터 공급자 페이지가 열립니다.
3. 데이터 공급자 생성을 선택합니다. 다음 표는 설정에 대한 설명입니다.

옵션	작업
구성	수동으로 데이터 공급자에 대한 정보를 입력할지 아니면 Amazon RDS DB 인스턴스를 사용할지 선택합니다.
이름	데이터 공급자의 이름을 입력합니다. 데이터 공급자를 쉽게 식별할 수 있도록 데이터 공급자에 고유한 이름을 사용해야 합니다.
엔진 유형	데이터 공급자의 데이터베이스 엔진 유형을 선택합니다.
[서버 이름]	데이터베이스 서버의 DNS(Domain Name Service) 이름 또는 IP 주소를 입력합니다. 동종 복제에 사용되는 데이터 공급자의

옵션	작업
	서버 이름은 영숫자로 시작해야 하며 영숫자, 하이픈(-), 마침표(.) 또는 밑줄(_)만 포함할 수 있습니다.
포트	데이터베이스 서버에 연결하는 데 사용되는 포트를 입력합니다.
서비스 ID(SID) 또는 서비스 이름	Oracle SID(SID)를 입력합니다. Oracle SID를 확인하려면 Oracle 데이터베이스에 다음 쿼리를 제출합니다. <pre>SELECT sys_context('userenv','instance_name') AS SID FROM dual;</pre>
데이터베이스 이름	이 데이터 공급자의 데이터베이스 이름을 입력합니다. 동종 복제에 사용되는 데이터 공급자의 데이터베이스 이름은 최대 63 자일 수 있으며 스페이스를 포함할 수 없습니다.
Secure Socket Layer(SSL) 모드	이 데이터 공급자에서 연결 암호화를 활성화하려면 SSL 모드를 선택합니다. 선택하는 모드에 따라 인증서와 서버 인증서 정보를 제공해야 할 수도 있습니다. 자세한 내용은 SSL 사용을 참조하세요 AWS Database Migration Service .
인증 모드	MongoDB 소스의 경우 AWS DMS 가 엔드포인트 연결을 인증하는 데 사용하는 인증 모드입니다.
인증 소스	MongoDB 소스의 경우 인증을 위한 자격 증명을 검증하는 데 사용할 MongoDB 데이터베이스의 이름입니다.
인증 메커니즘	MongoDB 소스의 경우 MongoDB가 암호를 암호화하는 데 사용하는 인증 방법입니다.

4. 데이터 공급자 생성을 선택합니다.

데이터 공급자를 생성한 후에는 AWS Secrets Manager에서 데이터베이스 연결 자격 증명 정보를 추가해야 합니다.

에서 마이그레이션 프로젝트 생성 AWS Database Migration Service

에서 마이그레이션 프로젝트를 생성하기 전에 다음 리소스를 생성해야 AWS DMS합니다.

- 소스 및 대상 데이터베이스를 설명하는 데이터 공급자
- 에 저장된 데이터베이스 자격 증명이 있는 보안 암호 AWS Secrets Manager
- Secrets Manager에 대한 액세스를 제공하는 AWS Identity and Access Management (IAM) 역할
- 네트워크 및 보안 설정이 포함된 인스턴스 프로파일

마이그레이션 프로젝트를 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트 생성을 선택합니다. 다음 표는 설정에 대한 설명입니다.

옵션	작업
이름	마이그레이션 프로젝트의 이름을 입력합니다. 마이그레이션 프로젝트를 쉽게 식별할 수 있도록 데이터 마이그레이션에 고유한 이름을 사용해야 합니다.
인스턴스 프로파일	마이그레이션 프로젝트에 사용할 인스턴스 프로파일을 선택합니다.
소스	찾아보기를 선택한 다음 소스 데이터 공급자를 선택합니다.
비밀 ID	Secrets Manager에서 소스 데이터베이스 자격 증명 정보가 저장된 비밀의 Amazon 리소스 이름(ARN)을 선택합니다.
IAM 역할	Secrets Manager에서 소스 데이터베이스 자격 증명 정보에 대한 액세스를 제공할 IAM 역할을 선택합니다.
대상	찾아보기를 선택한 다음 대상 데이터 공급자를 선택합니다.
비밀 ID	Secrets Manager에서 대상 데이터베이스 자격 증명 정보가 저장된 비밀의 ARN을 선택합니다.

옵션	작업
IAM 역할	Secrets Manager에서 대상 데이터베이스 자격 증명 정보에 대한 액세스를 제공할 IAM 역할을 선택합니다.
변환 규칙	(선택 사항) DMS Schema Conversion용 마이그레이션 프로젝트를 생성하는 경우 변환 규칙 추가를 선택하여 변환 규칙을 설정합니다. 변환 규칙을 사용하면 지정한 규칙에 따라 객체 이름을 변경할 수 있습니다. 자세한 내용은 변환 규칙 설정 단원을 참조하십시오.

4. 마이그레이션 프로젝트 생성을 선택합니다.

가 마이그레이션 프로젝트를 AWS DMS 생성한 후 DMS Schema Conversion 또는 동종 데이터 마이그레이션에서이 프로젝트를 사용할 수 있습니다. 마이그레이션 프로젝트 작업을 시작하려면 마이그레이션 프로젝트 페이지의 목록에서 프로젝트를 선택합니다.

에서 마이그레이션 프로젝트 관리 AWS Database Migration Service

생성한 마이그레이션 프로젝트는 수정하거나 삭제할 수 있습니다. 예를 들어 소스 또는 대상 데이터 공급자를 변경하려면 마이그레이션 프로젝트를 수정합니다.

스키마 변환 또는 데이터 마이그레이션 작업을 종료한 후에만 마이그레이션 프로젝트를 수정하거나 삭제할 수 있습니다. 이렇게 하려면 목록에서 마이그레이션 프로젝트를 선택하고 스키마 변환 또는 데이터 마이그레이션을 선택합니다. 그런 다음 DMS Schema Conversion에 대해 스키마 변환 종료를 선택하고 선택을 확인합니다. 동종 데이터 마이그레이션의 경우 데이터 마이그레이션을 선택한 다음 작업 메뉴에서 종지를 선택합니다. 마이그레이션 프로젝트를 편집한 후에 스키마 변환을 시작하거나 데이터 마이그레이션을 다시 시작할 수 있습니다.

마이그레이션 프로젝트를 수정하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음 수정을 선택합니다.
4. 프로젝트의 이름을 업데이트하거나, 인스턴스 프로파일을 편집하거나, 소스 및 대상 데이터 공급자를 변경합니다. 선택적으로, 변환 중에 객체 이름을 변경하는 마이그레이션 규칙을 추가하거나 편집할 수도 있습니다.

5. 변경 사항 저장을 선택합니다.

마이그레이션 프로젝트를 삭제하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음 삭제를 선택합니다.
4. 삭제를 선택하여 선택을 확인합니다.

AWS 자체 관리형 데이터베이스 소스를 위한 제로 ETL 통합

AWS 제로 ETL 통합은 여러 운영 및 트랜잭션 데이터베이스 소스의 Amazon Redshift, Amazon S3 및 Amazon S3 Tables에서 트랜잭션 및 운영 데이터를 사용할 수 있도록 하는 완전관리형 솔루션입니다. 제로 ETL을 사용하면 기존 AWS Database Migration Service (AWS DMS) 소스 엔드포인트를 통해 MySQL, PostgreSQL, SQL Server 및 Oracle과 같은 자체 관리형 소스 데이터베이스에서 Amazon Redshift로 데이터를 복제할 수 있습니다. 자동 동기화는 기존 추출, 전환, 적재(ETL) 프로세스를 방지합니다. 또한 실시간 분석 및 AI 워크로드를 지원합니다. 자세한 내용은 Amazon Redshift 관리 안내서의 [제로 ETL 통합](#)을 참조하세요.

제로 ETL 통합은 다음을 제공합니다.

- 실시간 데이터 복제 - 지연 시간을 최소화하면서 Oracle 데이터베이스에서 Amazon Redshift로의 지속적인 데이터 동기화.
- 복잡한 ETL 파이프라인 제거 - 사용자 지정 데이터 통합 솔루션을 구축하고 유지 관리할 필요가 없습니다.
- 운영 오버헤드 감소 - AWS APIs 통한 자동 설정 및 관리.
- 간소화된 데이터 통합 아키텍처 - 자체 관리형 데이터베이스와 AWS 분석 서비스 간의 원활한 통합.
- 보안 강화 - 기본 제공 암호화 및 IAM 액세스 제어.

자체 관리형 데이터베이스 소스에서 제로 ETL 통합이 작동하는 방식

이전에 자체 관리형 데이터베이스에 대해 생성한 기존 AWS DMS 엔드포인트를 사용하거나 새 엔드포인트를 생성할 수 있습니다.

- AWS Glue 콘솔 또는 [CLI](#)를 사용하여 AWS Glue 카탈로그에서 대상으로 Amazon Redshift와의 제로 ETL 통합을 생성합니다. 제로 ETL 통합을 생성할 때 스키마 및 테이블 필터를 지정할 수 있습니다.
- 통합과 관련된 추가 읽기 전용 리소스는 AWS DMS 서비스 내에 자동으로 생성됩니다. 제로 ETL 엔진을 포함한 이러한 리소스는 전체 로드 및 지속적인 데이터 변경 프로세스를 시작하여 데이터를 Amazon Redshift 대상 데이터베이스와 동기화하는 데 사용됩니다.
- 통합 소스를 생성할 때, 제로 ETL 통합을 생성할 때, Amazon Redshift 데이터 웨어하우스를 생성할 때 데이터의 암호화를 제어합니다.
- 통합은 데이터 파이프라인의 상태를 모니터링하고 가능한 경우 문제로부터 복구합니다.

- 동일한 유형의 소스에서 단일 Amazon Redshift 데이터 웨어하우스로 통합을 생성하여 여러 애플리케이션에 걸쳐 전체적인 인사이트를 도출할 수 있습니다.

데이터가 복제되면 Amazon Redshift의 분석 기능을 사용할 수 있습니다. 제공되는 분석 기능에는 기본 제공 기계 학습(ML), 구체화된 뷰, 데이터 공유, 여러 데이터 저장소 및 데이터 레이크에 대한 직접 액세스 등이 있습니다. 데이터 엔지니어의 경우 제로 ETL 통합을 통해 복잡한 데이터 파이프라인의 간헐적 오류로 인해 지연될 수 있는 시간에 민감한 데이터에 액세스할 수 있습니다. 트랜잭션 데이터에서 분석 쿼리와 ML 모델을 실행하여 시간에 민감한 이벤트와 비즈니스 의사 결정에 대한 시기적절한 인사이트를 도출할 수 있습니다.

제로 ETL 통합에 문제가 발생할 때 자동으로 알림을 받을 Amazon Redshift 이벤트 알림 구독을 생성할 수 있습니다. 통합 관련 이벤트 알림 목록을 보려면 [Amazon EventBridge를 사용한 제로 ETL 통합 이벤트 알림](#)을 참조하세요. 구독을 생성하는 가장 간단한 방법은 Amazon Simple Notification Service 콘솔을 사용하는 것입니다. Amazon SNS 주제를 새로 만들어 구독하는 방법에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 안내서의 [Amazon SNS 시작하기](#) 섹션을 참조하세요.

통합을 생성하기 전에 소스 데이터베이스 구성

제로 ETL 통합을 구성하기 전에 데이터베이스 엔진의 요구 사항에 따라 소스 데이터베이스를 올바르게 구성해야 합니다. 각 엔진에는 특정 구성 요구 사항 및 제한 사항이 있습니다.

SQL Server

- 구성 요구 사항은 [Microsoft SQL Server 데이터베이스를 소스로 사용을 참조하세요 AWS DMS](#).
- 변경 데이터 캡처(CDC) 요구 사항은 [SQL Server에서 지속적 복제를 위한 데이터 변경 캡처](#)를 참조하세요.

Note

RDS SQL Server 및 Azure SQL Server는 자체 관리형 제로 ETL 통합의 소스로 사용할 수 없습니다.

Oracle

- 구성 요구 사항 및 제한 사항은 [Oracle 데이터베이스를 소스로 사용을 참조하세요 AWS DMS](#).

MySQL

- 구성 요구 사항 및 제한 사항은 [MySQL 호환 데이터베이스를 소스로 사용을 참조하세요 AWS DMS](#).

PostgreSQL

- 구성 요구 사항 및 제한 사항은 [PostgreSQL 데이터베이스를 AWS DMS 소스로 사용을 참조하세요](#).
- 자체 관리형 PostgreSQL 데이터베이스를 소스로 사용하는 경우 다음 제한이 적용됩니다.
 - 자체 관리형 PostgreSQL 소스의 경우 소스 엔드포인트를 생성할 때 PostgreSQL 엔드포인트 설정 `false`에서 `CaptureDdl`로 설정해야 합니다. 이 파라미터를 설정하려면 소스 엔드포인트를 생성할 때 다음을 사용합니다.

```
--postgres-sql-settings '{"CaptureDdl": false}'
```

CaptureDdl을 `false`로 설정하지 않으면 작업이 성공적으로 시작되지 않을 수 있습니다.

- CaptureDdl을 `로` 설정하면 복제 중에 소스 데이터베이스에서 수행되는 `falseDDL` 작업(예: CREATE TABLE, ALTER TABLE 및 DROP TABLE)이 캡처되거나 대상으로 복제되지 않습니다.

제로 ETL 통합을 위한 IAM 권한 및 암호화 설정

제로 ETL 통합을 생성하고 관리하려면 적절한 IAM 권한, AWS Key Management Service (AWS KMS) 암호화 키 및 리소스 정책을 구성해야 합니다. 이 섹션에서는 필수 보안 구성 요소 설정에 대한 지침을 제공합니다.

사전 조건

제로 ETL 통합을 생성하기 전에 다음이 있는지 확인합니다.

- 통합을 생성하고 관리할 수 있는 적절한 권한이 있는 IAM 사용자 또는 역할
- 자체 관리형 데이터베이스의 AWS DMS 소스 엔드포인트를 구성합니다. 자세한 내용은 [the section called “통합을 생성하기 전에 소스 데이터베이스 구성”](#) 단원을 참조하십시오.
- Amazon Redshift 프로비저닝된 클러스터 또는 서버리스 네임스페이스를 대상으로 사용
- VPC 서브넷 및 보안 그룹을 포함한 네트워크 구성

KMS 키 생성

먼저 고객 관리형 AWS KMS 키를 생성하여 제로 ETL 통합에서 데이터를 암호화합니다. 다음 예제에서는 대칭 암호화 키를 생성합니다.

```
aws kms create-key \
  --description "On-prem Zero-ETL Integration Encryption Key" \
  --key-usage ENCRYPT_DECRYPT \
  --key-spec SYMMETRIC_DEFAULT \
  --region region
```

키 정책을 구성KeyId하고 통합을 생성할 때 필요하므로 응답Arn의 값을 기록해 둡니다.

출력 예시:

```
{
  "KeyMetadata": {
    "AWSAccountId": "account-id",
    "KeyId": "4e2c14f8-7abe-4aec-851a-379f6ed973a8",
    "Arn": "arn:aws:kms:region:account-id:key/4e2c14f8-7abe-4aec-851a-379f6ed973a8",
    "CreationDate": 1763155061.148,
    "Enabled": true,
    "Description": "Zero-ETL Integration Encryption Key",
    "KeyUsage": "ENCRYPT_DECRYPT",
    "KeyState": "Enabled",
    "Origin": "AWS_KMS",
    "KeyManager": "CUSTOMER",
    "CustomerMasterKeySpec": "SYMMETRIC_DEFAULT",
    "KeySpec": "SYMMETRIC_DEFAULT",
    "EncryptionAlgorithms": [
      "SYMMETRIC_DEFAULT"
    ],
    "MultiRegion": false
  }
}
```

KMS 키 정책 구성

KMS 키를 생성한 후 Amazon Redshift 및 AWS Glue 서비스가 암호화 및 복호화 작업에 키를 사용할 수 있도록 키 정책을 구성합니다. 키 정책은 서비스 보안 주체에게 필요한 권한을 부여하고 통합 생성 중에 사용할 암호화 컨텍스트를 포함해야 합니다.

다음 예제에서는 제로 ETL 통합에 대한 키 정책을 보여줍니다.

```
{
  "Version": "2012-10-17",
  "Id": "key-default-1",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::account-id:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allows the Redshift and glue service principal to add a grant to a
KMS key",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "redshift.amazonaws.com",
          "glue.amazonaws.com"
        ]
      },
      "Action": "kms:CreateGrant",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:context-key": "context-value"
        },
        "ForAllValues:StringEquals": {
          "kms:GrantOperations": [
            "Decrypt",
            "GenerateDataKey",
            "CreateGrant",
            "GenerateDataKeyWithoutPlaintext",
            "ReEncryptTo"
          ]
        }
      }
    }
  ]
}
```

```
}
```

kms:EncryptionContext 조건은 통합을 생성할 때 지정한 추가 암호화 컨텍스트와 일치해야 합니다. AWS KMS 콘솔 또는 다음 CLI 명령을 사용하여 키 정책을 업데이트할 수 있습니다.

```
aws kms put-key-policy \
  --key-id key-id \
  --policy-name default \
  --policy file://kms-key-policy.json
```

IAM 사용자 생성 및 AWS CLI 구성

제로 ETL 통합을 관리하는 데 사용할 IAM 사용자를 생성하고 적절한 자격 증명으로 AWS CLI를 구성합니다.

1. IAM 사용자 생성:

```
aws iam create-user --user-name cli-user
```

2. 사용자에게 대한 액세스 키를 생성합니다.

```
aws iam create-access-key --user-name cli-user
```

AWS CLI를 구성하는 데 필요하므로 출력SecretAccessKey에서 AccessKeyId 및를 저장합니다.

IAM 정책 생성 및 연결

제로 ETL 통합 작업에 대한 권한을 부여하는 IAM 정책을 생성합니다. 정책에는 AWS Glue, AWS DMS, Amazon Redshift AWS KMS 및에 대한 권한이 포함되어야 합니다.

다음 정책을 파일에 저장합니다(예: /tmp/zetl-policy.json).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ZetlGlueIntegrationAccess",
      "Effect": "Allow",
      "Action": [
```

```

        "glue:CreateIntegration",
        "glue:ModifyIntegration",
        "glue>DeleteIntegration",
        "glue:DescribeIntegrations",
        "glue:DescribeInboundIntegrations"
    ],
    "Resource": "*"
},
{
    "Sid": "DMSIntegrationAccess",
    "Effect": "Allow",
    "Action": [
        "dms:CreateOutboundIntegration",
        "dms:ModifyOutboundIntegration",
        "dms:CreateEndpoint",
        "dms:DescribeEndpoints",
        "dms:ModifyEndpoint",
        "dms>DeleteEndpoint",
        "dms:TestConnection"
    ],
    "Resource": "*"
},
{
    "Sid": "ZetlRedshiftFullAccess",
    "Effect": "Allow",
    "Action": [
        "redshift:*",
        "redshift-serverless:*",
        "ec2:DescribeAccountAttributes",
        "ec2:DescribeAddresses",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcs",
        "ec2:DescribeInternetGateways",
        "sns:CreateTopic",
        "sns:Get*",
        "sns:List*",
        "cloudwatch:Describe*",
        "cloudwatch:Get*",
        "cloudwatch:List*",
        "cloudwatch:PutMetricAlarm",
        "cloudwatch:EnableAlarmActions",
        "cloudwatch:DisableAlarmActions",

```



```

        "tag:GetResources",
        "tag:UntagResources",
        "tag:GetTagValues",
        "tag:GetTagKeys",
        "tag:TagResources"
    ],
    "Resource": "*"
},
{
    "Sid": "ZetlRedshiftDataAPI",
    "Effect": "Allow",
    "Action": [
        "redshift-data:ExecuteStatement",
        "redshift-data:CancelStatement",
        "redshift-data:ListStatements",
        "redshift-data:GetStatementResult",
        "redshift-data:DescribeStatement",
        "redshift-data:ListDatabases",
        "redshift-data:ListSchemas",
        "redshift-data:ListTables",
        "redshift-data:DescribeTable"
    ],
    "Resource": "*"
},
{
    "Sid": "ZetlKMSAccess",
    "Effect": "Allow",
    "Action": [
        "kms:CreateKey",
        "kms:DescribeKey",
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:ListKeys",
        "kms:CreateAlias",
        "kms:ListAliases",
        "kms:CreateGrant"
    ],
    "Resource": "*"
},
{
    "Sid": "ZetlSecretsManagerAccess",
    "Effect": "Allow",
    "Action": [

```

```

        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "secretsmanager:CreateSecret",
        "secretsmanager:UpdateSecret",
        "secretsmanager>DeleteSecret",
        "secretsmanager:DescribeSecret",
        "secretsmanager:ListSecrets",
        "secretsmanager:GetResourcePolicy",
        "secretsmanager:PutResourcePolicy",
        "secretsmanager:ValidateResourcePolicy"
    ],
    "Resource": "*"
}
]
}

```

정책을 생성하고 IAM 사용자에게 연결합니다.

```

aws iam create-policy \
  --policy-name ZetlCustomPolicy \
  --policy-document file:///tmp/zetl-policy.json

aws iam attach-user-policy \
  --policy-arn arn:aws:iam::account-id:policy/ZetlCustomPolicy \
  --user-name cli-user

```

AWS CLI 프로파일 구성

이전 단계에서 생성한 사용자 자격 증명으로 AWS CLI 프로파일을 구성합니다.

```

aws configure set aws_access_key_id ACCESS_KEY_ID --profile cli-user
aws configure set aws_secret_access_key SECRET_ACCESS_KEY --profile cli-user
aws configure set region region --profile cli-user
aws configure set output json --profile cli-user

```

프로파일 구성을 테스트합니다.

```

aws sts get-caller-identity --profile cli-user

```

출력에 사용자의 계정 ID, 사용자 ID 및 ARN이 표시되어야 프로필이 올바르게 구성되었는지 확인할 수 있습니다.

Redshift 리소스 정책 생성

Amazon Redshift 리소스 정책을 생성하여 AWS DMS 소스 엔드포인트가 Amazon Redshift 네임스페이스와의 인바운드 통합을 생성하도록 승인합니다. 이 정책은 Amazon Redshift 네임스페이스에 연결되며 데이터를 복제할 수 있는 소스를 제어합니다.

다음 예제에서는 Amazon Redshift 네임스페이스에 대한 리소스 정책을 생성하는 방법을 보여줍니다.

```
aws redshift put-resource-policy \
  --policy '{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Principal": {
          "Service": ["redshift.amazonaws.com"]
        },
        "Action": ["redshift:AuthorizeInboundIntegration"],
        "Condition": {
          "StringEquals": {
            "aws:SourceArn": "arn:aws:dms:region:account-id:endpoint:endpoint-id"
          }
        }
      },
      {
        "Effect": "Allow",
        "Principal": {
          "AWS": "account-id"
        },
        "Action": [
          "redshift:CreateInboundIntegration",
          "redshift:ModifyInboundIntegration"
        ]
      }
    ]
  }' \
  --resource-arn arn:aws:redshift:region:account-id:namespace:namespace-id \
  --region region
```

다음과 같이 자리 표시자를 바꿉니다.

- *region* - 리소스가 위치한 AWS 리전

- *account-id* - AWS 계정 ID
- *endpoint-id* - AWS DMS 소스 엔드포인트의 ID입니다.
- *namespace-id* - Amazon Redshift 네임스페이스의 ID입니다.

예: 암호화와 제로 ETL 통합 생성

필요한 권한 및 암호화 키를 설정한 후 AWS Glue API를 사용하여 제로 ETL 통합을 생성할 수 있습니다. 다음 예제에서는 KMS 암호화를 사용하여 MySQL 소스에서 Amazon Redshift 대상으로 통합을 생성하는 방법을 보여줍니다.

```
aws glue create-integration \
  --integration-name mysql-onprem-integration \
  --source-arn arn:aws:dms:region:account-id:endpoint:source-endpoint-id \
  --target-arn arn:aws:redshift:region:account-id:namespace:namespace-id \
  --description "MySQL to Redshift integration" \
  --integration-config '{"SourceProperties":{"SubnetIds":["subnet-id1,subnet-id2,subnet-id3"],"VpcSecurityGroupIds":["sg-id"]}}' \
  --data-filter "include: mysql.*" \
  --kms-key-id arn:aws:kms:region:account-id:key/key-id \
  --additional-encryption-context '{"context-key": "context-value"}' \
  --profile cli-user \
  --region region
```

명령에는 다음과 같은 키 파라미터가 포함됩니다.

- *--integration-name* - 통합의 고유한 이름
- *--source-arn* - AWS DMS 소스 엔드포인트의 ARN
- *--target-arn* - Amazon Redshift 네임스페이스의 ARN
- *--integration-config* - 서브넷 IDs 및 보안 그룹을 포함한 네트워크 구성
- *--data-filter* - 복제할 스키마 및 테이블을 지정합니다.
- *--kms-key-id* - 암호화를 위한 AWS KMS 키의 ARN
- *--additional-encryption-context* - KMS 키 정책과 일치해야 하는 암호화 컨텍스트 키-값 페어(예: {"*context-key*": "*context-value*"})
- *--profile* - 사용할 AWS CLI 프로파일(이전에 생성된 *cli-user* 프로파일)

성공적으로 생성되면 명령은 통합 ARN, 상태 및 구성 파라미터를 포함한 통합 세부 정보를 반환합니다. 출력 예시:

```
{
  "SourceArn": "arn:aws:dms:region:account-id:endpoint:endpoint-id",
  "TargetArn": "arn:aws:redshift:region:account-id:namespace:namespace-id",
  "IntegrationName": "mysql-onprem-integration",
  "IntegrationArn": "arn:aws:glue:region:account-id:integration:integration-id",
  "KmsKeyId": "arn:aws:kms:region:account-id:key/key-id",
  "AdditionalEncryptionContext": {
    "context-key": "context-value"
  },
  "Status": "CREATED",
  "CreateTime": 1763234086.001,
  "DataFilter": "include: mysql.*",
  "IntegrationConfig": {
    "SourceProperties": {
      "SubnetIds": "subnet-id1,subnet-id2,subnet-id3",
      "VpcSecurityGroupIds": "sg-id"
    }
  }
}
```

보안 모범 사례

제로 ETL 통합을 설정할 때 다음 보안 모범 사례를 따르세요.

- 최소 권한 액세스 사용 - 통합을 생성하고 관리하는 데 필요한 최소 권한만 부여합니다. 가능하면 리소스 수준 권한을 사용하는 것이 좋습니다.
- 암호화 활성화 - 항상 고객 관리형 AWS KMS 키를 사용하여 통합 데이터를 암호화합니다. 추가 보안을 위해 암호화 컨텍스트를 지정합니다.
- 자격 증명 정기적으로 교체 - IAM 사용자 액세스 키를 사용하는 경우 정기적으로 교체합니다. 대신 임시 자격 증명과 함께 IAM 역할을 사용하는 것이 좋습니다.
- 액세스 모니터링 - 통합 생성 및 관리와 관련된 API 호출을 모니터링하는 AWS CloudTrail 데 사용합니다.
- 네트워크 액세스 제한 - 네트워크 액세스를 필요한 리소스로만 제한하도록 VPC 보안 그룹을 구성합니다.
- 리소스 정책 사용 - Amazon Redshift 리소스 정책을 구현하여 데이터 웨어하우스와의 통합을 생성할 수 있는 소스를 제어합니다.

- 리소스 태그 지정 - 통합 및 관련 리소스에 태그를 적용하여 조직 및 비용 추적을 개선합니다.

의 모범 사례 AWS Database Migration Service

AWS Database Migration Service (AWS DMS)를 가장 효과적으로 사용하려면 데이터를 마이그레이션 하는 가장 효율적인 방법에 대한이 섹션의 권장 사항을 참조하세요.

주제

- [에 대한 마이그레이션 계획 AWS Database Migration Service](#)
- [스키마 변환](#)
- [AWS DMS 공개 설명서 검토](#)
- [개념 증명 실행](#)
- [AWS DMS 마이그레이션 성능 개선](#)
- [자체 온프레미스 이름 서버 사용](#)
- [대용량 이진 객체\(LOB\) 마이그레이션](#)
- [행 필터링을 사용하여 큰 테이블 마이그레이션 시 성능 향상](#)
- [지속적 복제](#)
- [소스 데이터베이스의 로드 감소](#)
- [대상 데이터베이스에서 병목 현상 감소](#)
- [마이그레이션 중에 데이터 유효성 검사 사용](#)
- [지표를 사용하여 AWS DMS 작업 모니터링](#)
- [이벤트 및 알림](#)
- [마이그레이션 문제 해결을 위해 작업 로그 사용](#)
- [Time Travel을 통한 복제 태스크 문제 해결](#)
- [Oracle 대상에서 사용자 및 스키마 변경](#)
- [Oracle 대상에 대한 테이블 및 인덱스 테이블스페이스 변경](#)
- [복제 인스턴스 버전 업그레이드](#)
- [마이그레이션 비용 이해](#)

에 대한 마이그레이션 계획 AWS Database Migration Service

를 사용하여 데이터베이스 마이그레이션을 계획할 때는 다음 사항을 AWS Database Migration Service고려하세요.

- 소스 및 대상 데이터베이스를 AWS DMS 복제 인스턴스에 연결하려면 네트워크를 구성합니다. 이 작업은 복제 인스턴스와 동일한 가상 프라이빗 클라우드(VPC)에 있는 AWS 리소스 두 개를 연결하는 것만큼 간단할 수도 있습니다. 가상 프라이빗 네트워크(VPN)을 통해 온프레미스 데이터베이스를 Amazon RDS DB 인스턴스에 연결하는 등 더 복잡한 구성까지 다양할 수 있습니다. 자세한 내용은 [데이터베이스 마이그레이션을 위한 네트워크 구성](#) 단원을 참조하십시오.
- 소스 및 대상 엔드포인트 - 소스 데이터베이스의 어떤 정보와 테이블을 대상 데이터베이스로 마이그레이션해야 하는지 알아야 합니다. 테이블 및 기본 키 생성을 포함한 기본 스키마 마이그레이션을 AWS DMS 지원합니다. 그러나 대상 데이터베이스에 보조 인덱스, 외래 키, 사용자 계정 등을 자동으로 생성하지 AWS DMS 않습니다. 소스 및 대상 데이터베이스 엔진에 따라 소스 또는 대상 데이터베이스에 추가 로깅을 설정하거나 기타 설정을 수정해야 할 수 있습니다. 자세한 내용은 [데이터 마이그레이션용 소스 및 마이그레이션에 적합한 대상](#) 섹션을 참조하세요.
- 스키마 및 코드 마이그레이션 - 스키마 또는 코드 변환 AWS DMS 을 수행하지 않습니다. Oracle SQL Developer, MySQL Workbench, pgAdmin III와 같은 도구를 사용하여 스키마를 변환할 수 있습니다. 기존 스키마를 다른 데이터베이스 엔진으로 변환하려면 AWS Schema Conversion Tool ()를 사용할 수 있습니다. 대상 스키마를 생성하고 전체 스키마(테이블, 인덱스, 보기 등)을 생성할 수 있습니다. 또한 PL/SQL 또는 TSQL을 PostgreSQL 및 기타 형식으로 변환하는 도구를 사용할 수도 있습니다. 에 대한 자세한 내용은 [AWS SCT 사용 설명서](#)를 AWS SCT참조하세요.
- 지원되지 않는 데이터 형식 - 소스 데이터 형식을 대상 데이터베이스의 해당 데이터 형식으로 변환해야 합니다. 지원되는 데이터 형식에 대한 자세한 내용은 해당 데이터 저장소 설명서의 소스 또는 대상 섹션을 참조하세요.
- 진단 지원 스크립트 결과 - 마이그레이션을 계획할 때는 진단 지원 스크립트를 실행하는 것이 좋습니다. 이러한 스크립트의 결과를 통해 잠재적 마이그레이션 실패에 대한 사전 정보를 찾을 수 있습니다.

데이터베이스에 사용할 수 있는 지원 스크립트가 있는 경우 다음 섹션의 해당 스크립트 주제에 있는 링크를 사용하여 다운로드하세요. 스크립트를 확인하고 검토한 후 로컬 환경에서 스크립트 주제에 설명된 절차에 따라 스크립트를 실행할 수 있습니다. 스크립트 실행이 완료되면 결과를 검토할 수 있습니다. 문제 해결의 첫 단계로 이러한 스크립트를 실행하는 것이 좋습니다. 결과는 AWS Support 팀과 함께 작업할 때 유용할 수 있습니다. 자세한 내용은 [에서 진단 지원 스크립트 작업 AWS DMS](#) 섹션을 참조하세요.

- 마이그레이션 전 평가 - 마이그레이션 전 평가는 데이터베이스 마이그레이션 태스크의 특정 구성 요소를 평가하여 마이그레이션 태스크가 예상대로 실행되는 데 방해가 될 수 있는 문제를 식별하는 데 도움이 됩니다. 이 평가를 사용하면 새 태스크 또는 수정된 태스크를 실행하기 전에 잠재적 문제를 식별할 수 있습니다. 마이그레이션 전 평가 작업에 대한 자세한 내용은 [작업에 대한 마이그레이션 전 평가 활성화 및 활용](#) 섹션을 참조하세요.

스키마 변환

AWS DMS 는 스키마 또는 코드 변환을 수행하지 않습니다. 기존 스키마를 다른 데이터베이스 엔진으로 변환하려는 경우를 사용할 수 있습니다 AWS SCT.는 소스 객체, 테이블, 인덱스, 뷰, 트리거 및 기타 시스템 객체를 대상 데이터 정의 언어(DDL) 형식으로 AWS SCT 변환합니다. AWS SCT 를 사용하여 PL/SQL 또는 TSQL과 같은 대부분의 애플리케이션 코드를 동등한 대상 언어로 변환할 수도 있습니다.

를 AWS SCT 에서 무료로 다운로드할 수 있습니다 AWS. 에 대한 자세한 내용은 [AWS SCT 사용 설명서](#)를 AWS SCT참조하세요.

소스 및 대상 엔드포인트가 동일한 데이터베이스 엔진인 경우 Oracle SQL Developer, MySQL Workbench 또는 PgAdmin4와 같은 도구를 사용하여 스키마를 이동 수 있습니다.

AWS DMS 공개 설명서 검토

첫 번째 마이그레이션 전에 소스 및 대상 엔드포인트에 대한 AWS DMS 공개 설명서 페이지를 살펴보는 것이 좋습니다. 이 설명서는 마이그레이션을 시작하기 전에 마이그레이션을 위한 사전 요구 사항을 식별하고 현재 제한 사항을 이해하는 데 도움이 될 수 있습니다. 자세한 내용은 [AWS DMS 엔드포인트 작업](#) 단원을 참조하십시오.

마이그레이션 중에 공개 설명서는 문제를 해결하는 데 도움이 될 수 있습니다 AWS DMS. 설명서의 페이지 문제를 해결하면 AWS DMS 및 선택한 엔드포인트 데이터베이스를 모두 사용하여 일반적인 문제를 해결하는 데 도움이 될 수 있습니다. 자세한 내용은 [에서 마이그레이션 작업 문제 해결 AWS Database Migration Service](#) 단원을 참조하십시오.

개념 증명 실행

데이터베이스 마이그레이션 초기 단계에서 환경 문제를 발견하는 데 도움이 되도록 소규모 테스트 마이그레이션을 실행하는 것이 좋습니다. 이렇게 하면 마이그레이션 일정을 보다 현실적으로 설정하는 데도 도움이 될 수 있습니다. 또한 전체 규모 테스트 마이그레이션을 실행하여가 네트워크를 통해 데이터베이스 처리량을 처리할 AWS DMS 수 있는지 여부를 측정해야 할 수 있습니다. 이 동안에는 초기 전체 로드 및 지속적 복제를 벤치마킹하고 최적화하는 것이 좋습니다. 이렇게 하면 네트워크 지연 시간을 파악하고 전반적인 성능을 측정하는 데 도움이 될 수 있습니다.

또한 이 시점에서 다음을 포함하여 데이터 프로파일 및 데이터베이스 규모를 파악할 수 있습니다.

- 대형, 중형 및 소형 테이블의 수.
- 가 데이터 유형 및 문자 집합 변환을 AWS DMS 처리하는 방법.

- 대형 객체(LOB) 열이 있는 테이블의 수.
- 테스트 마이그레이션을 실행하는 데 걸리는 시간.

AWS DMS 마이그레이션 성능 개선

AWS DMS 마이그레이션 성능에는 다음과 같은 여러 요인이 영향을 미칩니다.

- 소스의 리소스 가용성.
- 가용 네트워크 처리량.
- 복제 서버의 리소스 용량.
- 대상이 변경 사항을 수집하는 능력.
- 소스 데이터의 형식 및 분포.
- 마이그레이션할 객체의 수.

아래에 언급된 모범 사례 중 일부 또는 전부를 사용하여 성능을 개선할 수 있습니다. 이러한 방법을 사용할 수 있는지 여부는 구체적인 사용 사례에 따라 다릅니다. 다음과 같은 몇 가지 제한 사항을 확인할 수 있습니다.

적절한 복제 서버를 프로비저닝

AWS DMS는 Amazon EC2 인스턴스에서 실행되는 관리형 서비스입니다. 이 서비스는 소스 데이터베이스에 연결하고, 소스 데이터를 읽고, 대상 데이터베이스에서 사용할 수 있도록 데이터 형식을 지정하고, 대상 데이터베이스에 데이터를 로드합니다.

이 절차 대다수는 메모리에서 진행됩니다. 그렇지만, 대규모 트랜잭션은 디스크에서 일부 버퍼링이 필요할 수 있습니다. 캐시된 트랜잭션과 로그 파일도 디스크에 기록됩니다. 다음 섹션에서 복제 서버를 선택할 때 고려해야 할 사항을 확인할 수 있습니다.

CPU

AWS DMS는 이기종 마이그레이션을 위해 설계되었지만 동종 마이그레이션도 지원합니다. 동종 마이그레이션을 수행하려면 먼저 각 소스 데이터 유형을 동등한 AWS DMS 데이터 유형으로 변환합니다. 그런 다음 각 AWS DMS 형식 데이터를 대상 데이터 형식으로 변환합니다. AWS DMS 사용 설명서에서 각 데이터베이스 엔진의 이러한 변환에 대한 참조를 찾을 수 있습니다.

가 이러한 변환을 최적으로 수행 AWS DMS 하려면 변환이 발생할 때 CPU를 사용할 수 있어야 합니다. CPU에 과부하가 걸리고 CPU 리소스가 충분하지 않으면 마이그레이션 속도가 느려지고 다른 부작용도 발생할 수 있습니다.

복제 인스턴스 클래스

소형 인스턴스 클래스 중에는 서비스를 테스트하거나 소규모 마이그레이션을 하기에 충분한 것이 있습니다. 마이그레이션에 대량의 테이블이 포함되어 있는 경우 또는 여러 동시 복제 태스크를 실행해야 하는 경우, 대형 인스턴스 중 하나를 사용하는 것이 좋습니다. 서비스가 상당한 양의 메모리 및 CPU를 소비하기 때문에 더 큰 용량의 인스턴스를 사용하는 것이 좋습니다.

AWS DMS R5 및 C5 인스턴스 클래스 지원

R5 인스턴스 클래스는 메모리에서 대규모 데이터를 처리하는 워크로드에 대해 빠른 성능을 제공할 수 있도록 설계된 메모리 최적화 인스턴스입니다. 이를 사용하여 처리량이 많은 트랜잭션 시스템을 지속적으로 마이그레이션하거나 복제하면 때때로 많은 양의 CPU와 메모리를 소비 AWS DMS 할 수 있습니다. R5 인스턴스는 R4보다 vCPU당 5%의 추가 메모리를 제공하며 가장 큰 크기는 768GiB의 메모리를 제공합니다. 또한 R5 인스턴스는 R4에 비해 GiB당 가격이 10% 향상되고 CPU 성능이 최대 20% 향상되었습니다.

C5 인스턴스 클래스는 컴퓨팅 집약적 워크로드에 최적화되어 있으며 높은 컴퓨팅 가성비로 비용 효율적인 고성능을 제공합니다. 네트워크 성능이 크게 향상되었습니다. Elastic Network Adapter(ENA)는 C5 인스턴스에 Amazon EBS에 최대 25Gbps의 네트워크 대역폭과 최대 14Gbps의 전용 대역폭을 제공합니다. 특히 Oracle에서 PostgreSQL로 마이그레이션하는 것과 같은 이기종 마이그레이션 및 복제를 수행할 때 CPU 집약적일 AWS DMS 수 있습니다. C5 인스턴스는 이러한 상황에서 훌륭한 선택이 될 수 있습니다.

스토리지

인스턴스 클래스에 따라 복제 인스턴스에는 50GB 또는 100GB의 데이터 스토리지가 포함되어 있습니다. 이 스토리지는 로그 파일과 로드 중에 수집되는 캐시된 변경 사항을 저장하는 데 사용됩니다. 소스 시스템이 사용 빈도가 높거나 대규모 트랜잭션을 처리하는 경우 스토리지를 늘려야 할 수 있습니다. 복제 서버에서 여러 태스크를 실행하는 경우 스토리지를 늘려야 할 수도 있습니다. 하지만 일반적으로 기본 용량으로도 충분합니다.

의 모든 스토리지 볼륨 AWS DMS 은 GP2 또는 범용 SSD(SSDs. GP2 볼륨의 기본 성능은 3IOPS(초당 I/O 작업)이며 크레딧 기준으로 최대 3,000IOPS까지 버스트할 수 있습니다. 경험상 복제 인스턴스의 ReadIOPS 및 WriteIOPS 지표를 확인하는 것이 좋습니다. 이러한 값의 합계가 해당 볼륨의 기본 성능을 초과하지 않는지 확인하세요.

다중 AZ

다중 AZ 인스턴스를 선택하면 스토리지 장애로부터 마이그레이션을 보호할 수 있습니다. 대부분의 마이그레이션은 일시적이며 장기간 실행되지 않습니다. 지속적 복제 AWS DMS 를 위해 사용하는 경우 스토리지 문제가 발생할 경우 다중 AZ 인스턴스를 선택하면 가용성이 향상될 수 있습니다.

단일 AZ 또는 다중 AZ 복제 인스턴스를 사용하여 전체 로드를 수행하는 동안 장애 조치 또는 호스트 교체가 발생하는 경우 전체 로드 태스크는 실패할 것으로 예상됩니다. 완료되지 않았거나 오류 상태인 나머지 테이블의 경우 실패 지점부터 태스크를 다시 시작할 수 있습니다.

여러 개의 테이블을 병렬로 로드

기본적으로는 한 번에 8개의 테이블을 AWS DMS 로드합니다. dms.c4.xlarge 이상 인스턴스와 같은 매우 큰 복제 서버를 사용하는 경우 이 값을 약간 높이면 성능이 어느 정도 향상될 수 있습니다. 하지만 어느 시점에서는 이 병렬 처리를 높이면 성능이 저하됩니다. 복제 서버가 dms.t2.medium과 같이 비교적 소형인 경우에는 병렬로 로드되는 테이블 수를 줄이는 것이 좋습니다.

에서이 번호를 변경하려면 콘솔을 AWS Management Console 열고 작업을 선택한 다음 작업을 생성하거나 수정하도록 선택한 다음 고급 설정을 선택합니다. 튜닝 설정에서 병렬로 로드할 최대 테이블 수 옵션을 변경합니다.

를 사용하여이 번호를 변경하려면에서 MaxFullLoadSubTasks 파라미터를 AWS CLI변경합니다 TaskSettings.

병렬 전체 로드 사용

Oracle, Microsoft SQL Server, MySQL, Sybase 및 IBM DB2 LUW 소스에서 파티션 및 하위 파티션을 기반으로 병렬 로드를 사용할 수 있습니다. 이렇게 하면 전체 로드 기간을 전반적으로 개선할 수 있습니다. 또한 AWS DMS 마이그레이션 태스크를 실행하는 동안 대형 테이블 또는 파티셔닝된 테이블의 마이그레이션 속도를 높일 수 있습니다. 이렇게 하려면 테이블을 세그먼트로 분할하고 동일한 마이그레이션 태스크에서 세그먼트를 병렬로 로드하세요.

병렬 로드를 사용하려면 parallel-load 옵션을 사용하여 table-settings 유형의 테이블 매핑 규칙을 생성합니다. table-settings 규칙 내에서 병렬로 로드하려는 테이블의 선택 기준을 지정할 수 있습니다. 선택 기준을 지정하려면 parallel-load의 type 요소를 다음 중 하나로 설정합니다.

- partitions-auto
- subpartitions-auto
- partitions-list
- ranges
- none

이러한 설정에 대한 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 섹션을 참조하세요.

인덱스, 트리거 및 참조 무결성 제약 조건 사용

인덱스, 트리거, 참조 무결성 제약 조건은 마이그레이션 성능에 영향을 미치고 마이그레이션 실패를 초래할 수 있습니다. 이러한 제약 조건이 마이그레이션에 미치는 영향은 복제 태스크가 전체 로드 태스크인지 지속적 복제(변경 데이터 캡처·CDC) 태스크인지에 따라 달라집니다.

전체 로드 태스크의 경우 프라이머리 키 인덱스, 보조 인덱스, 참조 무결성 제약 조건 및 데이터 조작 언어(DML) 트리거를 삭제하는 것이 좋습니다. 또는 전체 로드 태스크가 완료될 때까지 생성을 연기할 수도 있습니다. 전체 로드 태스크 중에는 인덱스가 필요하지 않으며 인덱스가 있는 경우 유지 관리 오버헤드가 발생합니다. 전체 로드 태스크는 테이블 그룹을 한 번에 로드하므로 참조 무결성 제약 조건이 위반됩니다. 마찬가지로 삽입, 업데이트 및 삭제 트리거로 인해 오류가 발생할 수 있습니다(예: 이전에 대량으로 로드된 테이블에 대해 행 삽입이 트리거되는 경우). 다른 유형의 트리거도 추가 처리로 인해 성능에 영향을 줍니다.

데이터 볼륨이 비교적 작고 추가 마이그레이션 시간이 문제가 되지 않는 경우 전체 로드 태스크 전에 프라이머리 키와 세컨더리 인덱스를 구축할 수 있습니다. 참조 무결성 제약 조건 및 트리거는 항상 비활성화하세요.

전체 로드 + CDC 태스크의 경우 CDC 단계 전에 보조 색인을 추가하는 것이 좋습니다. AWS DMS는 논리적 복제를 사용하므로 전체 테이블 스캔을 방지하기 위해 DML 작업을 지원하는 보조 인덱스가 있는지 확인합니다. CDC 단계가 시작되기 전에 복제 태스크를 일시 중지하여 인덱스를 작성하고 참조 무결성 제약 조건을 생성한 다음 태스크를 다시 시작할 수 있습니다.

트리거는 전환 직전에 활성화해야 합니다.

백업 및 트랜잭션 로깅을 비활성화하세요.

Amazon RDS 데이터베이스로 마이그레이션하려면 전환 준비가 될 때까지 대상에서 백업과 다중 AZ를 비활성화하는 것이 좋습니다. 마찬가지로, Amazon RDS가 아닌 다른 시스템으로 마이그레이션할 때는 일반적으로 전환 이후까지 대상에서의 로깅을 비활성화하는 것이 좋습니다.

여러 개의 태스크 사용

경우에 따라 단일 마이그레이션에 여러 태스크를 사용하면 성능이 향상될 수 있습니다. 일반적인 트랜잭션에 참여하지 않는 테이블 집합이 있는 경우 마이그레이션을 여러 태스크로 나눌 수 있습니다. 트랜잭션 일관성은 태스크 내에서 유지되므로 별도의 태스크에 있는 테이블이 일반적인 트랜잭션에 참여하지 않도록 하는 것이 중요합니다. 또한 각 태스크는 독립적으로 트랜잭션 스트림을 읽으므로 소스 데이터베이스에 너무 많은 부하가 걸리지 않도록 주의해야 합니다.

여러 태스크를 사용하여 별도의 복제 스트림을 생성할 수 있습니다. 이렇게 하면 소스에서의 읽기, 복제 인스턴스의 프로세스, 대상 데이터베이스에 대한 쓰기를 병렬화할 수 있습니다.

변경 처리 최적화

기본적으로는 트랜잭션 모드에서 변경 사항을 AWS DMS 처리하여 트랜잭션 무결성을 유지합니다. 트랜잭션 무결성이 일시적으로 저하될 여지가 있다면 배치 최적화 적용 옵션을 대신 사용할 수 있습니다. 이 옵션은 효율성을 위해 트랜잭션을 효율적으로 그룹화하고 일괄적으로 적용합니다. 배치 최적화 적용 옵션을 사용하면 거의 항상 참조 무결성 제약 조건을 위반합니다. 따라서 마이그레이션 프로세스 중에 이러한 제약 조건을 해제한 다음, 전환 프로세스의 일부로 다시 설정하는 것이 좋습니다.

자체 온프레미스 이름 서버 사용

일반적으로 AWS DMS 복제 인스턴스는 Amazon EC2 인스턴스의 도메인 이름 시스템(DNS) 해석기를 사용하여 도메인 엔드포인트를 확인합니다. 그러나 Amazon Route 53 Resolver를 사용하는 경우 자체 온프레미스 이름 서버를 사용하여 특정 엔드포인트를 확인할 수 있습니다. 이 도구를 사용하면 인바운드 및 아웃바운드 엔드포인트, 전달 규칙 및 프라이빗 연결을 AWS 사용하여 온프레미스와 간에 쿼리할 수 있습니다. 온프레미스 이름 서버를 사용하면 보안이 향상되고 방화벽 뒤에서의 사용 편의성이 향상된다는 이점이 있습니다.

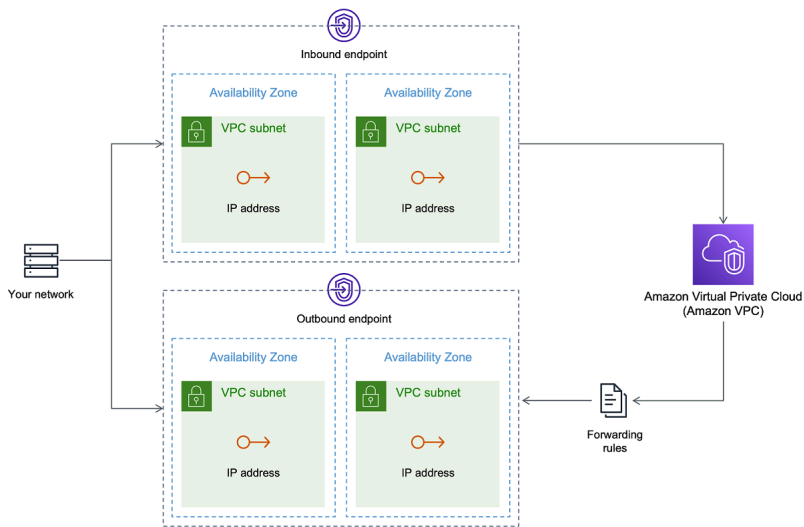
인바운드 엔드포인트가 있는 경우 온프레미스에서 시작되는 DNS 쿼리를 사용하여 AWS호스팅 도메인을 확인할 수 있습니다. 엔드포인트를 구성하려면 해석기를 제공하려는 각 서브넷의 IP 주소를 할당합니다. 온프레미스 DNS 인프라와 간에 연결을 설정하려면 Direct Connect 또는 가상 프라이빗 네트워크(VPN)를 AWS사용합니다.

아웃바운드 엔드포인트는 온프레미스 이름 서버에 연결됩니다. 이름 서버는 허용 목록에 포함되고 아웃바운드 엔드포인트에 설정된 IP 주소에만 액세스 권한을 부여합니다. 이름 서버의 IP 주소는 대상 IP 주소입니다. 아웃바운드 엔드포인트에 대한 보안 그룹을 선택할 때는 복제 인스턴스에서 사용하는 것과 동일한 보안 그룹을 선택하세요.

선택된 도메인을 이름 서버로 전달하려면 다음 규칙을 사용합니다. 아웃바운드 엔드포인트는 여러 전달 규칙을 처리할 수 있습니다. 전달 규칙의 범위는 가상 프라이빗 클라우드(VPC)입니다. VPC와 연결된 전달 규칙을 사용하여 AWS 클라우드의 논리적으로 격리된 섹션을 프로비저닝할 수 있습니다. 이 논리적으로 격리된 섹션에서 가상 네트워크에서 AWS 리소스를 시작할 수 있습니다.

온프레미스 DNS 인프라 내에서 호스팅되는 도메인을 아웃바운드 DNS 쿼리를 설정하는 조건부 전달 규칙으로 구성할 수 있습니다. 이러한 도메인 중 하나에 대한 쿼리가 수행되면 규칙은 규칙으로 구성된 서버로 DNS 요청을 전달하려는 시도를 트리거합니다. 다시 말하지만 Direct Connect 또는 VPN을 통한 프라이빗 연결이 필요합니다.

다음 다이어그램은 Route 53 Resolver 아키텍처입니다.



Route 53 DNS Resolver에 대한 자세한 내용은 Amazon Route 53 개발자 안내서의 [Route 53 Resolver 시작하기](#)를 참조하세요.

에서 Amazon Route 53 Resolver 사용 AWS DMS

를 사용하여 엔드포인트를 해결하기 AWS DMS 위해 용 온프레미스 이름 서버를 생성할 수 있습니다 [Amazon Route 53 Resolver](#).

Route 53을 AWS DMS 기반으로 용 온프레미스 이름 서버를 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/route53/> Route 53 콘솔을 엽니다.
2. Route 53 콘솔에서 Route 53 Resolver를 구성할 AWS 리전을 선택합니다. Route 53 Resolver는 리전별 서비스입니다.
3. 쿼리 방향(인바운드, 아웃바운드 또는 둘 다)을 선택합니다.
4. 인바운드 쿼리 구성을 제공합니다.
 - a. 엔드포인트 이름을 입력하고 VPC를 선택합니다.
 - b. VPC 내에서 하나 이상의 서브넷을 할당합니다(예: 가용성을 위해 2개 선택).
 - c. 엔드포인트로 사용할 특정 IP 주소를 할당하거나 Route 53 Resolver가 자동으로 할당하도록 합니다.
5. VPC 내부의 워크로드가 DNS 쿼리를 DNS 인프라로 라우팅할 수 있도록 온프레미스 도메인에 대한 규칙을 생성합니다.
6. 온프레미스 DNS 서버에 대한 IP 주소를 하나 이상 입력합니다.

7. 규칙을 제출합니다.

모든 것이 생성되면 VPC가 인바운드 및 아웃바운드 규칙과 연결되어 트래픽 라우팅을 시작할 수 있습니다.

Route 53 Resolver에 대한 자세한 내용은 Amazon Route 53 개발자 안내서의 [Route 53 Resolver 시작하기](#)를 참조하세요.

대용량 이진 객체(LOB) 마이그레이션

일반적으로는 LOB 데이터를 다음 두 단계로 AWS DMS 마이그레이션합니다.

1. AWS DMS 는 대상 테이블에 새 행을 생성하고 연결된 LOB 값을 제외한 모든 데이터로 행을 채웁니다.
2. AWS DMS 는 대상 테이블의 행을 LOB 데이터로 업데이트합니다.

이 LOB 마이그레이션 프로세스를 수행하려면 마이그레이션 중에 대상 테이블의 모든 LOB 열에 Null 을 허용해야 합니다. 이는 소스 테이블에서 LOB 열을 Null로 설정할 수 없는 경우에도 마찬가지입니다. 가 대상 테이블을 AWS DMS 생성하면 기본적으로 LOB 열이 null로 설정됩니다. 경우에 따라 가져오기 또는 내보내기와 같은 다른 메커니즘을 사용하여 대상 테이블을 생성할 수 있습니다. 이러한 경우 마이그레이션 작업을 시작하기 전에 LOB 열이 Null 허용인지 확인하세요.

이 요구 사항에는 한 가지 예외가 있습니다. Oracle 소스에서 Oracle 대상으로 동종 마이그레이션을 수행하고 제한적 LOB 모드를 선택한다고 가정해 보겠습니다. 이 경우 LOB 값을 포함하여 전체 행이 한 번에 채워집니다. 이러한 경우 AWS DMS 는 필요한 경우 null할 수 없는 제약 조건이 있는 대상 테이블 LOB 열을 생성할 수 있습니다.

제한적 LOB 모드 사용

AWS DMS 는 마이그레이션에 LOB 값이 포함된 경우 성능과 편의성의 균형을 맞추는 두 가지 방법을 사용합니다.

1. 제한적 LOB 모드에서는 모든 LOB 값을 사용자가 지정한 크기 제한(기본값은 32KB)까지 마이그레이션합니다. 크기 제한보다 큰 LOB 값은 수동으로 마이그레이션해야 합니다. 모든 마이그레이션 태스크의 기본값인 제한적 LOB 모드가 일반적으로 최상의 성능을 제공합니다. 하지만 최대 LOB 크기 파라미터 설정이 올바른지 확인해야 합니다. 이 파라미터를 모든 테이블에서 가장 큰 LOB 크기로 설정하세요.

2. 전체 LOB 모드는 크기에 관계없이 테이블의 모든 LOB 데이터를 마이그레이션합니다. 전체 LOB 모드를 사용하면 테이블의 모든 LOB 데이터를 편리하게 이동할 수 있지만 프로세스가 성능에 상당한 영향을 미칠 수 있습니다.

PostgreSQL과 같은 일부 데이터베이스 엔진의 경우 LOBs와 같은 JSON 데이터 형식을 AWS DMS 처리합니다. 제한적 LOB 모드를 선택한 경우 최대 LOB 크기 옵션이 JSON 데이터가 잘리지 않는 값으로 설정되어 있는지 확인하세요.

AWS DMS 는 대규모 객체 데이터 유형(BLOBs, CLOBs 및 NCLOBs) 사용을 완벽하게 지원합니다. 다음 소스 엔드포인트는 전체 LOB 지원을 제공합니다.

- Oracle
- Microsoft SQL Server
- ODBC

다음 대상 엔드포인트는 전체 LOB 지원을 제공합니다.

- Oracle
- Microsoft SQL Server

다음 대상 엔드포인트는 제한적 LOB 지원을 제공합니다. 이 대상 엔드포인트에는 무제한 LOB 크기를 사용할 수 없습니다.

- Amazon Redshift
- Amazon S3

전체 LOB를 지원하는 엔드포인트의 경우 LOB 데이터 형식에 대한 크기 제한을 설정할 수도 있습니다.

개선된 LOB 성능

LOB 데이터를 마이그레이션하는 동안 다음과 같은 다양한 LOB 최적화 설정을 지정할 수 있습니다.

테이블별 LOB 설정

테이블별 LOB 설정을 사용하여 테이블 일부 또는 전부에 대한 태스크 수준 LOB 설정을 재정의할 수 있습니다. 이렇게 하려면 `table-settings` 규칙에서 `lob-settings`를 정의합니다. 다음은 몇 가지 큰 LOB 값이 포함된 예제 테이블입니다.

```

SET SERVEROUTPUT ON
CREATE TABLE TEST_CLOB
(
ID NUMBER,
C1 CLOB,
C2 VARCHAR2(4000)
);
DECLARE
bigtextstring CLOB := '123';
iINT;
BEGIN
WHILE Length(bigtextstring) <= 60000 LOOP
bigtextstring := bigtextstring || '00000000000000000000000000000000';
END LOOP;
INSERT INTO TEST_CLOB (ID, C1, C2) VALUES (0, bigtextstring, 'AnyValue');
END;
/
SELECT * FROM TEST_CLOB;
COMMIT

```

다음으로 마이그레이션 태스크를 생성하고 새 lob-settings 규칙을 사용하여 테이블의 LOB 처리를 수정합니다. bulk-max-siz 값에 따라 최대 LOB 크기(KB)가 결정됩니다. 지정된 크기보다 크면 잘립니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "HR",
      "table-name": "TEST_CLOB"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "table-settings",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
      "schema-name": "HR",
      "table-name": "TEST_CLOB"
    }
  }
]
}

```

```

    },
    "lob-settings": {
      "mode": "limited",
      "bulk-max-size": "16"
    }
  }
]
}

```

이 AWS DMS 작업이 생성되더라도 테이블 FullLobMode : true별 LOB 설정은이 특정 테이블의 LOB 데이터를 16,000 AWS DMS 으로 자르도록 지시합니다. 태스크 로그를 확인하여 이를 확인할 수 있습니다.

```

721331968: 2018-09-11T19:48:46:979532 [SOURCE_UNLOAD] W: The value of column 'C' in
table
'HR.TEST_CLOB' was truncated to length 16384

```

인라인 LOB 설정

AWS DMS 작업을 생성할 때 LOB 모드에 따라 LOBs 방법이 결정됩니다.

전체 LOB 모드와 제한적 LOB 모드는 각각 장단점이 있습니다. 인라인 LOB 모드는 전체 LOB 모드와 제한적 LOB 모드의 장점을 결합합니다.

큰 LOB와 작은 LOB를 모두 복제해야 하지만 대부분의 LOB가 작은 경우 인라인 LOB 모드를 사용할 수 있습니다. 이 옵션을 선택하면 전체 로드 중에 AWS DMS 작업이 작은 LOBs 인라인으로 전송하므로 더 효율적입니다. AWS DMS 작업은 소스 테이블에서 조회를 수행하여 대용량 LOBs를 전송합니다.

변경 처리 중에는 소스 테이블에서 검색을 수행하여 작은 LOB와 큰 LOB를 모두 복제합니다.

인라인 LOB 모드를 사용하는 경우 AWS DMS 작업은 모든 LOB 크기를 확인하여 인라인으로 전송할 LOB 크기를 결정합니다. 지정된 크기보다 큰 LOB는 전체 LOB 모드를 사용하여 복제됩니다. 따라서 대부분의 LOB가 지정된 설정보다 큰 경우 이 옵션을 사용하지 않는 것이 좋습니다. 대신 LOB 크기를 무제한으로 허용하세요.

이 옵션은 태스크 설정의 InlineLobMaxSize 속성을 사용하여 구성합니다. 이 속성은 FullLobMode가 true로 설정된 경우에만 사용할 수 있습니다. InlineLobMaxSize의 기본값은 0이고 범위는 1~102400킬로바이트(100MB)입니다. InlineLobMaxSize를 32Kb보다 큰 값으로 설정하면 워크로드에 따라 메모리 압력과 성능 저하가 발생할 수 있습니다.

예를 들어 다음 AWS DMS 작업 설정을 사용할 수 있습니다. 여기서 `InlineLobMaxSize` 값을 5로 설정하면 5KiB(5,120바이트)보다 작거나 같은 모든 LOB가 인라인으로 전송됩니다.

```
{
  "TargetMetadata": {
    "TargetSchema": "",
    "SupportLobs": true,
    "FullLobMode": true,
    "LobChunkSize": 64,
    "LimitedSizeLobMode": false,
    "LobMaxSize": 32,
    "InlineLobMaxSize": 5,
    "LoadMaxFileSize": 0,
    "ParallelLoadThreads": 0,
    "ParallelLoadBufferSize": 0,
    "BatchApplyEnabled": false,
    "TaskRecoveryTableEnabled": false},
  . . .
}
```

리소스 할당

- 적절한 복제 인스턴스 크기를 선택하고 LOB 처리에 충분한 스토리지를 할당해야 합니다. 자세한 내용은 [복제 인스턴스에 가장 적합한 크기 선택](#)을 참조하세요.
- 대상에 LOB 열이 포함된 테이블에 대한 충분한 여유 공간이 있는지 확인합니다.
- 성능을 최적화하기 위해 특성이 다른 LOB에 대해 개별 태스크를 생성하세요.
 - 자주 업데이트된 LOB와 거의 수정되지 않는 LOB 비교
 - 대형 LOB 소형 LOB 비교
 - 자주 액세스하는 테이블과 자주 액세스하지 않는 테이블 비교

행 필터링을 사용하여 큰 테이블 마이그레이션 시 성능 향상

대형 테이블을 마이그레이션할 때 성능을 개선하려면 마이그레이션을 두 개 이상의 태스크로 나누세요. 행 필터링을 사용하여 마이그레이션을 여러 태스크로 나누려면 키 또는 파티션 키를 사용합니다. 예를 들어 1에서 8,000,000까지의 정수 프라이머리 키 ID가 있는 경우 행 필터링을 사용하여 8개의 태스크를 생성하여 각각 1백만 개의 레코드를 마이그레이션할 수 있습니다.

콘솔에서 행 필터링을 적용하려면

1. 를 엽니다 AWS Management Console.
2. 태스크를 선택하고 새 태스크를 생성합니다.
3. 테이블 매핑 탭을 선택하고 선택 규칙을 확장합니다.
4. 새 선택 규칙 추가를 선택합니다. 이제 작거나 같음, 크거나 같음, 같음 또는 두 값 사이의 범위 조건을 사용하여 열 필터를 추가할 수 있습니다. 열 필터링에 대한 자세한 내용은 [콘솔에서 테이블 선택 및 변환 규칙 지정](#) 섹션을 참조하세요.

날짜별로 분할된 큰 파티션된 테이블이 있는 경우 날짜를 기준으로 데이터를 마이그레이션할 수 있습니다. 예를 들어, 월별로 분할된 테이블이 있고 현재 월의 데이터만 업데이트된다고 가정합니다. 이 경우 각 고정 월별 파티션에 대해 전체 로드 태스크를 생성하고 현재 업데이트된 파티션에 대해 전체 로드 + CDC 태스크를 생성할 수 있습니다.

테이블에 단일 열 기본 키 또는 고유 인덱스가 있는 경우 작업에서 범위 유형의 병렬 로드를 사용하여 테이블을 AWS DMS 분할하여 데이터를 병렬로 로드하도록 할 수 있습니다. 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 단원을 참조하십시오.

지속적 복제

AWS DMS 는 데이터를 지속적으로 복제하여 소스 데이터베이스와 대상 데이터베이스를 동기화된 상태로 유지합니다. 제한된 양의 데이터 정의 언어(DDL) 문만 복제됩니다. AWS DMS 는 인덱스, 사용자, 권한, 저장 프로시저, 기타 테이블 데이터와 직접 관련되지 않은 데이터베이스 변경 사항과 같은 항목을 전파하지 않습니다.

지속적 복제를 사용하려는 경우 복제 인스턴스를 생성할 때 다중 AZ 옵션을 설정하세요. 다중 AZ를 선택하면 복제 인스턴스의 고가용성과 장애 조치 기능을 확보할 수 있습니다. 하지만 이 옵션은 성능에 영향을 줄 수 있으며 대상 시스템에 변경 사항을 적용하는 동안 복제 속도가 느려질 수 있습니다.

소스 또는 대상 데이터베이스를 업그레이드하기 전에 이러한 데이터베이스에서 실행 중인 모든 AWS DMS 태스크를 중지하는 것이 좋습니다. 업그레이드가 완료된 후 태스크를 재개하세요.

지속적 복제 중에는 소스 데이터베이스 시스템과 AWS DMS 복제 인스턴스 간의 네트워크 대역폭을 식별하는 것이 중요합니다. 복제가 진행되는 동안 네트워크로 인해 병목 현상이 발생하지 않는지 확인해야 합니다.

소스 데이터베이스 시스템의 시간당 변경 속도 및 아카이브 로그 생성을 파악하는 것도 중요합니다. 이렇게 하면 복제가 진행되는 동안 발생할 수 있는 처리량을 이해하는 데 도움이 될 수 있습니다.

소스 데이터베이스의 로드 감소

AWS DMS 는 소스 데이터베이스에서 일부 리소스를 사용합니다. 전체 로드 태스크 중에 AWS DMS 는 소스 테이블에서 병렬로 처리되는 각 테이블에 대해 전체 테이블 스캔을 수행합니다. 또한 마이그레이션의 일부로 생성하는 각 태스크는 CDC 프로세스의 일환으로 소스에 변경 사항을 쿼리합니다. AWS DMS 가 Oracle과 같은 일부 소스에 대해 CDC를 수행하려면 데이터베이스의 변경 로그에 기록된 데이터의 양을 늘려야 할 수 있습니다.

소스 데이터베이스에 과부하가 걸린다면 마이그레이션할 태스크 수 또는 각 태스크의 테이블 수를 줄이세요. 각 태스크마다 소스 변경 사항이 독립적으로 적용되므로 태스크를 통합하면 변경 사항 캡처 워크로드를 줄일 수 있습니다.

대상 데이터베이스에서 병목 현상 감소

마이그레이션하는 동안 대상 데이터베이스의 쓰기 리소스를 경합하는 프로세스를 모두 제거해 보세요.

- 불필요한 트리거를 비활성화하세요.
- 초기 로드 중에는 보조 인덱스를 비활성화하고 나중에 복제가 진행되는 동안 다시 활성화하세요.
- Amazon RDS 데이터베이스의 경우 전환이 완료될 때까지 백업 및 다중 AZ를 비활성화하는 것이 좋습니다.
- 비 RDS 시스템으로 마이그레이션할 때는 전환 시까지 대상의 모든 로깅을 비활성화하는 것이 좋습니다.

마이그레이션 중에 데이터 유효성 검사 사용

데이터가 소스에서 대상으로 정확히 마이그레이션되었는지 확인하기 위해 데이터 유효성 검사를 사용하는 것이 좋습니다. 작업에 대한 데이터 검증을 켜면 테이블에 대한 전체 로드가 수행된 직후 소스 데이터와 대상 데이터를 비교하기 AWS DMS 시작합니다.

데이터 검증은 소스 및 대상 엔드포인트로 AWS DMS 지원하는 모든 곳에서 다음 데이터베이스와 함께 작동합니다.

- Oracle
- PostgreSQL
- MySQL
- MariaDB

- Microsoft SQL Server
- Amazon Aurora MySQL 호환 버전
- Amazon Aurora PostgreSQL 호환 에디션
- IBM DB2 LUW
- Amazon Redshift

자세한 내용은 [AWS DMS 데이터 검증](#) 단원을 참조하십시오.

지표를 사용하여 AWS DMS 작업 모니터링

AWS DMS 콘솔을 사용하여 태스크의 지표를 모니터링할 수 있는 몇 가지 옵션이 있습니다.

호스트 지표

CloudWatch 지표 탭에서 각 특정 복제 인스턴스에 대한 호스트 지표를 찾을 수 있습니다. 여기에서 복제 인스턴스의 크기가 적절한지 모니터링할 수 있습니다.

복제 작업 지표

수신 및 커밋된 변경 사항, 복제 호스트와 소스/대상 데이터베이스 간의 지연 시간을 비롯한 복제 태스크 지표는 각 태스크에 대한 CloudWatch Metrics 탭에서 확인할 수 있습니다.

테이블 지표

개별 테이블 지표는 각 태스크에 대한 테이블 통계 탭에서 확인할 수 있습니다. 이러한 지표에는 다음과 같은 수치가 포함됩니다.

- 전체 로드 중에 로드된 행 수.
- 태스크 시작 이후 삽입, 업데이트 및 삭제 횟수.
- 태스크 시작 이후 DDL 작업 수.

지표 모니터링에 대한 자세한 정보는 [AWS DMS 작업 모니터링](#) 섹션을 참조하세요.

이벤트 및 알림

AWS DMS 는 Amazon SNS를 사용하여 복제 인스턴스 생성 또는 삭제와 같은 AWS DMS 이벤트가 발생할 때 알림을 제공합니다. AWS 리전에 대해 Amazon SNS에서 지원하는 모든 형식으로 이러한 알

림을 사용할 수 있습니다. 여기에는 이메일 메시지, 문자 메시지 또는 HTTP 엔드포인트에 대한 호출이 포함될 수 있습니다.

자세한 내용은 [에서 Amazon SNS 이벤트 및 알림 작업 AWS Database Migration Service](#) 섹션을 참조하세요.

마이그레이션 문제 해결을 위해 작업 로그 사용

경우에 따라에서 경고 또는 오류 메시지가 작업 로그에만 나타나는 문제가 AWS DMS 발생할 수 있습니다. 특히 데이터 잘림 문제나 외래 키 위반으로 인한 행 거부는 태스크 로그에만 기록됩니다. 따라서 데이터베이스를 마이그레이션할 때는 태스크 로그를 검토해야 합니다. 태스크 로그를 보려면 Amazon CloudWatch를 태스크 생성의 일부로 구성하세요.

자세한 내용은 [Amazon CloudWatch를 사용하여 복제 지표 모니터링](#)을 참조하세요.

Time Travel을 통한 복제 태스크 문제 해결

AWS DMS 마이그레이션 문제를 해결하기 위해 Time Travel을 사용할 수 있습니다. Time Travel에 대한 자세한 내용은 [Time Travel 작업 설정](#) 섹션을 참조하세요.

Time Travel로 작업할 때 다음 사항을 고려해야 합니다.

- DMS 복제 인스턴스의 오버헤드를 방지하려면 디버깅이 필요한 태스크에만 Time Travel을 활성화합니다.
- Time Travel을 사용하여 며칠 동안 실행될 수 있는 복제 태스크의 문제를 해결할 때는 복제 인스턴스 지표에서 리소스 오버헤드를 모니터링합니다. 이 접근 방식은 특히 소스 데이터베이스에서 장시간 동안 높은 트랜잭션 로드가 실행되는 경우에 적용됩니다. 자세한 내용은 [AWS DMS 작업 모니터링](#) 섹션을 참조하세요.
- Time Travel 태스크 설정 EnableRawData를 true로 설정하면 DMS 복제 중 태스크 메모리 사용량이 Time Travel이 활성화되어 있지 않을 때보다 많을 수 있습니다. Time Travel을 장시간 활성화하는 경우 태스크를 모니터링하세요.
- 현재 Time Travel은 태스크 수준에서만 활성화할 수 있습니다. 모든 테이블의 변경 사항이 Time Travel 로그에 기록됩니다. 트랜잭션 볼륨이 많은 데이터베이스의 특정 테이블에 대한 문제를 해결하려면 별도의 태스크를 생성하세요.

Oracle 대상에서 사용자 및 스키마 변경

Oracle을 대상으로 사용하는 경우는 대상 엔드포인트의 사용자가 소유한 스키마로 데이터를 AWS DMS 마이그레이션합니다.

예를 들어 명명된 PERFDATA 스키마를 Oracle 대상 엔드포인트로 마이그레이션하는 중이고 대상 엔드포인트 사용자 이름은 MASTER이라고 가정해 보겠습니다. AWS DMS 는 MASTER로 Oracle 대상에 연결하고 MASTER 스키마를 PERFDATA의 데이터베이스 객체로 채웁니다.

이 동작을 재정의하려면 스키마 변환을 제공합니다. 예를 들어 PERFDATA 스키마 객체를 대상 엔드포인트의 PERFDATA 스키마로 마이그레이션하려면 다음 변환을 사용합니다.

```
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "object-locator": {
    "schema-name": "PERFDATA"
  },
  "rule-target": "schema",
  "rule-action": "rename",
  "value": "PERFDATA"
}
```

변환에 대한 자세한 내용은 [JSON을 사용하여 테이블 선택 및 변환 지정](#) 섹션을 참조하세요.

Oracle 대상에 대한 테이블 및 인덱스 테이블스페이스 변경

Oracle을 대상으로 사용하는 경우는 모든 테이블과 인덱스를 대상의 기본 테이블스페이스로 AWS DMS 마이그레이션합니다. 예를 들어 소스가 Oracle이 아닌 다른 데이터베이스 엔진이라고 가정해 보겠습니다. 모든 대상 테이블 및 인덱스가 동일한 기본 테이블스페이스로 마이그레이션됩니다.

이 동작을 재정의하려면 해당하는 테이블스페이스 변환을 제공합니다. 예를 들어, 테이블 및 인덱스를 소스의 스키마를 따라 명명된 Oracle 대상의 테이블 및 인덱스 테이블스페이스로 마이그레이션하려고 한다고 가정해 보겠습니다. 이 경우 다음과 같은 변형을 사용할 수 있습니다. 여기서 소스의 스키마는 이름이 INVENTORY이고 대상의 해당 테이블 및 인덱스 테이블스페이스는 이름이 각각 INVENTORYTBL 및 INVENTORYIDX입니다.

```
{
```

```

    "rule-type": "transformation",
    "rule-id": "3",
    "rule-name": "3",
    "rule-action": "rename",
    "rule-target": "table-tablespace",
    "object-locator": {
      "schema-name": "INVENTORY",
      "table-name": "%",
      "table-tablespace-name": "%"
    },
    "value": "INVENTORYTBL"
  },
  {
    "rule-type": "transformation",
    "rule-id": "4",
    "rule-name": "4",
    "rule-action": "rename",
    "rule-target": "index-tablespace",
    "object-locator": {
      "schema-name": "INVENTORY",
      "table-name": "%",
      "index-tablespace-name": "%"
    },
    "value": "INVENTORYIDX"
  }
}

```

변환에 대한 자세한 내용은 [JSON을 사용하여 테이블 선택 및 변환 지정](#) 섹션을 참조하세요.

Oracle이 소스와 대상인 경우 Oracle 소스 추가 접속 속성인 `enableHomogenousTablespace=true`를 설정하여 기존 테이블 또는 인덱스 테이블스페이스 할당을 유지할 수 있습니다. 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요.

복제 인스턴스 버전 업그레이드

AWS 는 새로운 기능 및 성능 개선과 함께 새로운 버전의 AWS DMS 복제 엔진 소프트웨어를 정기적으로 릴리스합니다. 복제 엔진 소프트웨어의 각 버전마다 고유한 버전 번호가 있습니다. AWS DMS 복제 인스턴스를 최신 버전으로 업그레이드하기 전에 프로덕션 워크로드를 실행하는 복제 인스턴스의 기존 버전을 테스트하는 것이 중요합니다. 사용 가능한 버전 업그레이드에 대한 자세한 내용은 [AWS DMS 릴리스 정보](#) 섹션을 참조하세요.

마이그레이션 비용 이해

AWS Database Migration Service 를 사용하면 저렴한 비용으로 데이터베이스를 로 AWS 쉽고 안전하게 마이그레이션할 수 있습니다. 복제 인스턴스 및 추가 로그 스토리지에 대한 비용만 지불하면 됩니다. 각 데이터베이스 마이그레이션 인스턴스에는 대부분의 복제에 충분한 스왑 공간, 복제 로그 및 데이터 캐시용 스토리지가 포함되어 있으며 인바운드 데이터 전송은 무료입니다.

초기 로드 또는 피크 로드 시간에는 더 많은 리소스가 필요할 수 있습니다. CloudWatch 지표를 사용하여 복제 인스턴스 리소스 사용률을 면밀히 모니터링할 수 있습니다. 그런 다음 사용량에 따라 복제 인스턴스 크기를 늘리거나 줄일 수 있습니다.

마이그레이션 비용 추정에 대한 자세한 내용은 다음을 참조하세요.

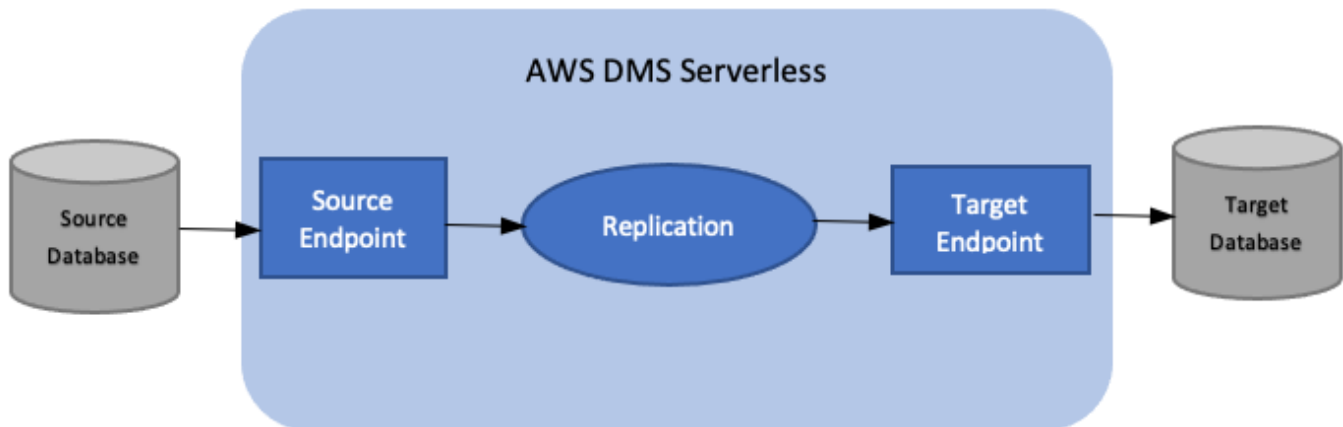
- [AWS Database Migration Service 요금](#)
- [AWS 요금 계산기](#)

AWS DMS 서버리스 작업

AWS DMS Serverless는 자동 프로비저닝, 규모 조정, 기본 제공 고가용성 및 pay-for-use 결제 모델을 제공하여 운영 민첩성을 높이고 비용을 최적화하는 기능입니다. 서버리스 기능을 사용하면 용량 예측, 프로비저닝, 비용 최적화, 복제 엔진 버전 관리, 패치 적용과 같은 복제 인스턴스 관리 태스크가 필요하지 않습니다.

AWS DMS Serverless에서는의 현재 기능 AWS DMS (이 문서에서 AWS DMS 표준이라고 함)과 마찬가지로 엔드포인트를 사용하여 소스 및 대상 연결을 생성합니다. 소스 및 대상 엔드포인트를 생성한 후, 제공된 복제에 대한 구성 설정이 포함된 복제 구성을 생성합니다. 복제를 시작, 중지, 수정 또는 삭제하여 복제를 관리할 수 있습니다. 각 복제 작업에는 데이터베이스 마이그레이션의 요구 사항에 따라 구성할 수 있는 설정이 있습니다. JSON 파일 또는 AWS DMS 섹션을 사용하여 이러한 설정을 지정합니다 AWS Management Console. 복제 설정에 대한 자세한 내용은 [AWS DMS 엔드포인트 작업을 참조](#)하십시오. 복제가 시작되면 AWS DMS 서버리스는 소스 데이터베이스에 연결하고, 데이터베이스 메타데이터를 수집하여 복제 워크로드를 분석합니다. 이 메타데이터를 사용하여 필요한 용량을 AWS DMS 계산 및 프로비저닝하고 데이터 복제를 시작합니다.

다음 다이어그램은 AWS DMS 서버리스 복제 프로세스를 보여줍니다.



Note

AWS DMS Serverless의 현재 엔진 버전은 [3.5.4입니다](#).

다음 주제를 보고 AWS DMS Serverless에 대한 자세한 내용을 알아보세요.

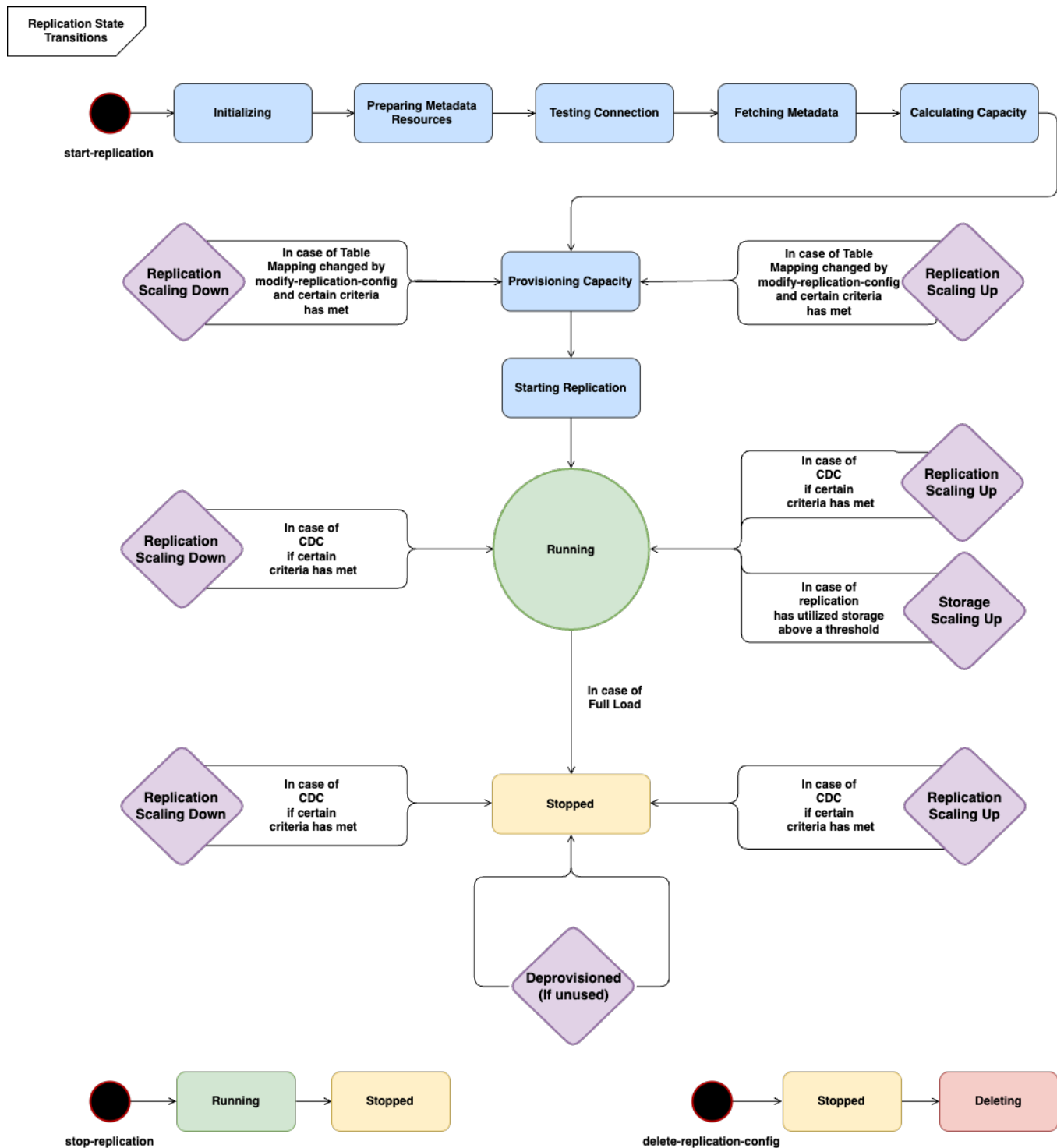
주제

- [AWS DMS 서버리스 구성 요소](#)
- [AWS DMS 서버리스 제한 사항](#)
- [AWS DMS 서버리스 사전 마이그레이션.](#)

AWS DMS 서버리스 구성 요소

복제를 수행하는 데 필요한 리소스를 관리하기 위해 AWS DMS 서버리스에는 서비스가 수행하는 다양한 내부 작업을 보여주는 세분화된 상태가 있습니다. 복제를 시작하면 AWS DMS 서버리스는 용량 부하를 계산하고, 계산된 용량을 프로비저닝한 후 다음 복제 상태에 따라 데이터 복제를 시작합니다.

다음 다이어그램은 AWS DMS 서버리스 복제의 상태 전환을 보여줍니다.



- 복제를 시작한 후 첫 번째 상태는 초기화 중입니다. 이 상태에서는 모든 필수 파라미터가 초기화됩니다.

- 바로 다음 상태에는 메타데이터 리소스 준비 중, 연결 테스트 중, 메타데이터 가져오는 중이 포함됩니다. 이러한 상태에서 AWS DMS Serverless는 소스 데이터베이스에 연결하여 필요한 용량을 예측하는 데 필요한 정보를 얻습니다.
- 복제 상태가 연결 테스트인 경우 AWS DMS 서버리스는 소스 및 대상 데이터베이스에 대한 연결이 성공적으로 설정되었는지 확인합니다.
- 연결 테스트 중 이후의 복제 상태는 메타데이터 가져오는 중입니다. 여기서는 용량을 계산하는 데 필요한 정보를 AWS DMS 검색합니다.
- 가 필요한 정보를 AWS DMS 검색하면 다음 상태는 용량 계산입니다. 이 상태에서 시스템은 복제를 수행하는 데 필요한 기본 리소스의 크기를 계산합니다.
- 용량 계산 중 다음에는 상태가 용량 프로비저닝 중으로 전환됩니다. 복제 작업이 이 상태인 경우, AWS DMS 서버리스는 기본 컴퓨팅 리소스를 초기화합니다.
- 모든 리소스가 성공적으로 프로비저닝된 후의 복제 상태는 복제 시작 중입니다. 이 상태에서 AWS DMS 서버리스는 데이터 복제를 시작합니다. 복제 단계는 다음과 같습니다.
 - 전체 로드: 이 단계에서 DMS는 복제가 시작될 때와 마찬가지로 소스 데이터 저장소를 복제합니다.
 - CDC(초기): 이 단계에서 DMS는 전체 로드 단계에서 발생한 소스 데이터 저장소의 변경 사항을 복제합니다. DMS는 `StopTaskCachedChangesNotApplied` 태스크 설정이 `false`인 경우에만 이 단계를 실행합니다.
 - CDC(진행 중): 초기 CDC 단계 후 DMS는 소스 데이터베이스의 변경 사항이 발생할 때 이를 복제합니다. DMS는 `StopTaskCachedChangesApplied` 태스크 설정이 `false`인 경우에만 초기 CDC 단계 이후에 복제를 계속 실행합니다.
- 마지막 상태는 실행 중입니다. 실행 중 상태에서는 데이터 복제가 진행되는 중입니다.
- 중지하는 복제는 중지됨 상태로 전환됩니다. 성공적으로 완료된 전체 로드 전용 복제 작업의 경우 복제가 중지 상태가 될 수 있습니다. 중지되거나 실패한 상태에서 복제를 재개하거나 다시 시작하는 경우 이러한 상황을 고려해야 합니다.
 - 가 리소스를 프로비저닝 AWS DMS 해제하므로 48시간 이내에 시작되지 않은 복제는 다시 시작할 수 없습니다.

이 주제는 다음 섹션을 포함하고 있습니다.

- [지원 엔드포인트](#)
- [서버리스 애플리케이션 생성](#)
- [AWS DMS 서버리스 복제 수정](#)
- [컴퓨팅 구성](#)

- [AWS DMS 서버리스의 오토 스케일링 이해](#)
- [AWS DMS 서버리스 복제 모니터링](#)
- [Oracle에서 Amazon Redshift 및 Amazon S3로의 전체 로드 마이그레이션을 위한 향상된 처리량](#)
- [AWS DMS 서버리스의 스토리지 Autoscaling 이해](#)

AWS DMS 서버리스의 경우 AWS DMS 콘솔의 왼쪽 탐색 패널에 서버리스 복제라는 새로운 옵션이 있습니다. 서버리스 복제의 경우 복제 인스턴스 유형이나 태스크 대신, 복제를 지정하여 복제를 정의합니다. 또한 복제를 위해 DMS로 프로비저닝할 최대 및 최소 DMS 용량 단위(DCU)를 지정합니다. DCU는 2GB의 RAM입니다.는 복제가 현재 사용 중인 각 DCU에 대해 계정에 AWS DMS 청구합니다. AWS DMS 요금에 대한 자세한 내용은 [AWS Database Migration Service 요금](#)을 참조하세요.

AWS DMS 그런 다음은 테이블 매핑과 워크로드의 예측 크기에 따라 복제 리소스를 자동으로 프로비저닝합니다. 이 용량 단위는 지정한 최소 및 최대 용량 단위 값 범위에 속하는 값입니다.

지원 엔드포인트

AWS DMS 서버리스에서는 서비스가 해당 설정을 처리하기 때문에 엔진 버전을 선택하고 관리할 필요가 없습니다. AWS DMS Serverless는 다음 소스를 지원합니다.

- MongoDB
- Amazon DocumentDB(MongoDB 호환)
- Microsoft SQL Server
- PostgreSQL 호환 데이터베이스
- MySQL 호환 데이터베이스
- MariaDB
- Oracle
- Amazon S3
- IBM DB2

AWS DMS Serverless는 다음 대상을 지원합니다.

- Microsoft SQL Server
- PostgreSQL
- MySQL 호환 데이터베이스

- Oracle
- Amazon S3
- Amazon Redshift
- Amazon DynamoDB
- Amazon Kinesis Data Streams
- Amazon Managed Streaming for Apache Kafka
- Amazon OpenSearch Service
- Amazon DocumentDB(MongoDB 호환)
- Amazon Neptune

AWS DMS 서버리스의 일부로 AWS DMS 서버리스 복제를 생성, 구성, 시작 및 관리할 수 있는 콘솔 명령에 액세스할 수 있습니다. 콘솔의 서버리스 복제 섹션을 사용하여 이러한 명령을 실행하려면 다음 중 하나를 수행해야 합니다.

- 정책을 연결할 새 AWS Identity and Access Management (IAM) 정책 및 IAM 역할을 설정합니다.
- AWS CloudFormation 템플릿을 사용하여 필요한 액세스 권한을 제공합니다.

AWS DMS Serverless를 사용하려면 계정에 서비스 연결 역할(SLR)이 있어야 합니다. 이 역할의 생성 및 사용을 AWS DMS 관리합니다. 필요한 SLR이 있는지 확인하는 방법에 대한 자세한 내용은 [에 대한 서비스 연결 역할 AWS DMS](#) 섹션을 참조하세요.

서버리스 애플리케이션 생성

두 기존 AWS DMS 엔드포인트 간에 서버리스 복제를 생성하려면 다음을 수행합니다. AWS DMS 엔드포인트 생성에 대한 자세한 내용은 섹션을 참조하세요 [소스 및 대상 엔드포인트 생성](#).

서버리스 애플리케이션 생성

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 서버리스 복제를 선택한 다음, 복제 생성을 선택합니다.
3. 복제 생성 페이지에서 서버리스 복제 구성을 지정합니다.

옵션	작업
이름	복제를 식별할 이름을 입력합니다(예: DMS-replication).
설명이 포함된 Amazon 리소스 이름 (ARN) - 선택 사항	이 선택적 파라미터를 사용하여 복제에 대한 설명을 제공할 수 있습니다.
소스 데이터베이스 엔드포인트	계정의 기존 엔드포인트를 선택합니다. AWS DMS Serverless는 AWS DMS 표준이 지원하는 엔드포인트 유형의 하위 집합만 지원합니다.
대상 데이터베이스 엔드포인트	계정의 기존 엔드포인트를 선택합니다. AWS DMS Serverless는 AWS DMS 표준이 지원하는 엔드포인트 유형의 하위 집합만 지원합니다.
복제 유형	요구 사항에 따라 복제 유형을 선택합니다. - 전체 로드: 기존 데이터만 AWS DMS 마이그레이션합니다. - 전체 로드 및 변경 데이터 캡처(CDC): 복제 중에 발생하는 기존 데이터와 변경 사항을 AWS DMS 마이그레이션합니다. - 변경 데이터 캡처(CDC): 복제를 시작한 후 발생하는 변경 사항 AWS DMS 만 마이그레이션합니다.

설정 섹션에서 복제에 필요한 설정을 지정합니다.

테이블 매핑 섹션에서 테이블 매핑을 설정하여 복제 중인 데이터를 선택하고 필터링하는 규칙을 정의합니다. 매핑을 지정하기 전에 소스 및 대상 데이터베이스의 데이터 형식 매핑에 대한 관련 설명서 섹션을 참조하세요. 소스 및 대상 데이터베이스의 데이터 유형 매핑에 대한 자세한 내용은 [AWS DMS 엔드포인트 작업](#) 주제의 소스 및 대상 엔드포인트 유형에 대한 데이터 유형 섹션을 참조하세요.

컴퓨팅 설정 섹션에서 다음과 같은 설정을 지정합니다. 컴퓨팅 구성 설정에 대한 내용은 [컴퓨팅 구성](#) 섹션을 참조하세요.

옵션	작업
VPC	기존 VPC를 선택합니다.
서브넷 그룹	기존 서브넷 그룹을 선택합니다.
VPC 보안 그룹	아직 선택하지 않은 경우 기본값을 선택합니다.
AWS KMS 키	적절한 KMS 키를 선택합니다. KMS 키에 대한 자세한 내용은 AWS Key Management Service API 참조의 키 생성 을 참조하세요.
배포	그대로 둡니다.
가용 영역	그대로 둡니다.
최소 DMS 용량 단위(DCU) - (선택 사항)	기본값인 1 DCU를 사용하려면 비워둡니다.
최대 DMS 용량 단위(DCU)	16 DCU를 선택합니다.

유지 관리 설정을 그대로 둡니다.

4. 복제 생성을 선택합니다.

AWS DMS 는 마이그레이션을 수행하기 위해 서버리스 복제를 생성합니다.

AWS DMS 서버리스 복제 수정

복제 구성을 수정하려면 `modify-replication-config` 작업을 사용합니다. CREATED, STOPPED 또는 FAILED 상태의 AWS DMS 복제 구성만 수정할 수 있습니다. `modify-replication-config` 작업에 대한 자세한 내용은 AWS Database Migration Service API 참조의 [ModifyReplicationConfig](#)를 참조하세요.

를 사용하여 서버리스 복제 구성을 수정하려면 AWS Management Console

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

2. 탐색 창에서 서버리스 복제를 선택합니다.
3. 수정할 복제를 선택합니다. 다음 표에는 복제의 현재 상태에 따라 수정할 수 있는 내용이 설명되어 있습니다.

설정	설명	허용되는 상태
이름	복제의 이름을 변경할 수 있습니다. 인쇄 가능한 ASCII 문자(/, " 및 @ 제외) 8~16자를 포함하는 복제의 이름을 입력합니다. 이름은 선택한 AWS 리전의 계정에서 고유해야 합니다. 수행 중인 AWS 리전 및 작업 포함과 같은 몇 가지 세부 정보를 이름에 추가하도록 선택할 수 있습니다. 예: west2-mysql2mysql-config1 .	ReplicationState 가 CREATED, STOPPED, FAILED인 경우.
소스 데이터베이스 엔드포인트	새 기존 소스 엔드포인트를 복제할 소스로 선택합니다.	ProvisionState 가 null일 때 ReplicationState 가 CREATED 또는 FAILED인 경우.
대상 데이터베이스 엔드포인트	새 기존 대상 엔드포인트를 복제할 대상으로 선택합니다.	ProvisionState 가 null일 때 ReplicationState 가 CREATED 또는 FAILED인 경우.
복제 유형	서버리스 복제 유형을 수정할 수 있습니다.	ProvisionState 가 null일 때 ReplicationState 가 CREATED 또는 FAILED인 경우.
복제 설정	대상 테이블 준비 모드, 복제에 LOB 열을 포함할지 여부, 최대 LOB 크기, 검증, 로깅을 포함한 복제 설정을 수정할 수 있습니다. 자세한 내용은 작업 설정 단원을 참조하십시오.	ReplicationState 가 CREATED, STOPPED, FAILED인 경우.

설정	설명	허용되는 상태
테이블 매핑	선택 규칙 및 변환 규칙을 포함하여, 서버리스 복제의 테이블 매핑 설정을 수정할 수 있습니다. 자세한 내용은 테이블 매핑 단원을 참조하십시오.	ReplicationState 가 CREATED, STOPPED, FAILED인 경우.

설정	설명	허용되는 상태
컴퓨팅 구성	네트워킹 설정, 크기 조정 설정, 유지 관리 설정을 포함하여 서버리스 복제의 컴퓨팅 구성 설정을 수정할 수 있습니다. 컴퓨팅 구성 설정에 대한 내용은 컴퓨팅 구성 섹션을 참조하세요.	• ReplicationState 가 CREATED, STOPPED 또는 FAILED인 경우 다음과 같은 크기 조정, 유지 관리, 네트워크 설정을 수정할 수 있습니다. • MinCapacityUnits • MaxCapacityUnits • MultiAZ • PreferredMaintenanceWindow • VpcSecurityGroupIds • ProvisionState 가 null일 때 FAILED 또는 ReplicationState 가 CREATED인 경우 다음과 같은 네트워킹 및 보안 설정을 수정할 수 있습니다. • AvailabilityZone • DnsNameServers • KmsKeyId

설정	설명	허용되는 상태
		ReplicationSubnetGroupId

Note

작업 상태가 시작되거나 실행 중인 경우 DMS 작업과 관련된 엔드포인트를 수정할 수 없습니다.

컴퓨팅 구성

컴퓨팅 구성 파라미터 또는 콘솔 섹션을 사용하여 복제 프로비저닝을 구성합니다. 컴퓨팅 구성 객체의 필드에는 다음 내용이 포함됩니다.

옵션	설명
MinCapacityUnits	프로비저닝 AWS DMS 할 최소 DMS 용량 단위(DCU) 수입니다. 이는 오토 스케일링으로 스케일 다운할 수 있는 최소 DCU이기도 합니다.
MaxCapacityUnits	복제 용량 예측에 따라 AWS DMS 가 프로비저닝할 수 있는 최대 DMS 용량 단위(DCU)입니다. 이는 오토 스케일링으로 스케일 업할 수 있는 최대 DCU이기도 합니다.
KmsKeyId	복제 스토리지와 연결 정보를 암호화하기 위해 사용할 암호화 키입니다. (기본값) aws/dms를 선택하면 계정 및와 연결된 기본 KMS 키를 AWS DMS 사용합니다 AWS 리전. 설명 및 계정 번호가 키의 ARN과 함께 표시됩니다. 암호화 키 사용에 대한 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정 섹션을 참조하세요. 이 튜토리얼에서는 (기본값) aws/dms를 선택한 상태로 둡니다.
ReplicationSubnetGroupId	복제를 생성할 위치인 선택된 VPC에 있는 복제 서브넷 그룹입니다. 소스 데이터베이스가 VPC에 있는 경우, 소스 데

옵션	설명
	이터베이스가 포함된 서브넷 그룹을 복제할 위치로 선택합니다. 복제 서브넷에 대한 자세한 내용은 복제 서브넷 그룹 생성 섹션을 참조하세요.
VpcSecurityGroupIds	모든 복제 인스턴스가 VPC에 생성됩니다. 소스 데이터베이스가 VPC에 있는 경우, 데이터베이스가 위치한 DB 인스턴스에 대한 액세스 권한을 부여하는 VPC 보안 그룹을 선택합니다.
PreferredMaintenanceWindow	이 파라미터는 시스템 유지 관리를 실행할 수 있는 주 단위 기간(UTC, 협정 세계시)을 정의합니다. 기본값은 당 8시간 블록의 시간 중에서 임의로 선택한 30분 기간으로 AWS 리전, 요일에 무작위로 발생합니다.
MultiAZ	이 선택적 파라미터를 설정하면 다른 가용 영역에 복제의 대기 복제본이 생성되어 장애 조치를 지원합니다. 변경 데이터 캡처(CDC) 또는 지속적 복제를 사용하려면, 이 옵션을 켜야 합니다.

AWS DMS 서버리스의 오토 스케일링 이해

복제를 프로비저닝하고 RUNNING 상태가 되면 AWS DMS 서비스는 변화하는 워크로드에 맞게 기본 리소스의 용량을 관리합니다. 이러한 관리 기능은 다음과 같은 복제 설정에 따라 복제 리소스의 크기를 조정합니다.

- MinCapacityUnits
- MaxCapacityUnits

복제는 사용률 임계값 상한을 초과하는 기간이 지나면 스케일 업되고, 용량 사용률이 오랫동안 최소 용량 사용률 임계값 이하로 떨어지면 스케일 다운됩니다.

Note

전체 로드가 진행 중인 동안에는 서버리스 복제를 자동으로 축소할 수 없습니다.

AWS DMS 서버리스에서 오토 스케일링 조정

복제 Autoscaling 파라미터를 조정하려면 최대값 MaxCapacityUnits으로 설정하고가 리소스 프로비저닝을 AWS DMS 관리하도록 하는 것이 좋습니다. 오토 스케일링의 이점을 최대한 활용하고 트랜잭션 볼륨의 급증에 대비하려면 가장 큰 DCU 최대 용량 설정을 선택하는 것이 좋습니다. 요금 계산기는 복제 시 지속적으로 최대 DCU를 사용할 경우 최대 월별 비용을 보여줍니다. 사용한 용량에 대해서만 비용을 지불하므로, 최대 DCU는 실제 비용을 나타내지 않습니다.

복제가 전체 용량으로 리소스를 사용하지 않는 경우는 리소스를 AWS DMS 점진적으로 프로비저닝 해제하여 비용을 절감합니다. 하지만 리소스 프로비저닝 및 프로비저닝 해제에는 시간이 걸리므로, 복제 워크로드가 갑자기 급증할 것으로 예상되는 경우 MinCapacityUnits 설정을 이를 처리할 수 있는 값으로 설정하는 것이 좋습니다. 이렇게 하면가 더 높은 워크로드 수준에 맞게 리소스를 AWS DMS 프로비저닝하는 동안 복제가 과소 프로비저닝되는 것을 방지할 수 있습니다.

데이터 요구 사항에 비해 너무 낮은 최대 용량 설정 또는 복제 워크로드의 갑작스러운 급증을 처리하기에 너무 낮은 최소 용량으로 복제를 과소 프로비저닝할 경우, CapacityUtilization 지표가 계속 최대값으로 표시될 수 있습니다. 이 경우 복제가 실패할 수 있습니다. 과소 프로비저닝된 리소스로 인해 복제가 실패하면는 복제 로그에 out-of-memory 이벤트를 AWS DMS 생성합니다. 복제 워크로드 또는 조정된 구성의 갑작스러운 급증으로 인해 메모리 부족(OOM) 조건이 발생하는 경우 시스템에는 상황을 해결하고 처리를 재개할 수 있는 오토 스케일링 기능이 기본 제공되어 있습니다. 그러나이 자동 복구 메커니즘은 즉각적이지 않으며 유효해지는 데 다소 시간이 걸릴 수 있습니다. 더 빠른 복구를 위해 특히 MinCapacityUnits 값을 늘린 다음 작업을 재개하여 작업 구성을 수정하여 수동 작업을 수행할 수 있습니다. 이 수동 개입은 자동 크기 조정 프로세스를 기다리는 것보다 메모리 부족 오류를 더 빠르게 해결할 수 있습니다.

AWS DMS 서버리스 복제 모니터링

AWS 는 AWS DMS 서버리스 복제를 모니터링하고 잠재적 인시던트에 대응하기 위한 몇 가지 도구를 제공합니다.

- [AWS DMS 서버리스 복제 지표](#)
- [AWS DMS 서버리스 복제 로그](#)

AWS DMS 서버리스 복제 지표

서버리스 복제 모니터링에는 다음 통계의 Amazon CloudWatch 지표가 포함됩니다. 이러한 통계는 각 서버리스 복제별로 그룹화됩니다.

지표	단위	설명
CapacityUtilization	%	서버리스 복제에 사용된 메모리 비율
CDCIncomingChanges	%	대상에 적용되기를 기다리고 있는 특정 시점의 총 변경 이벤트 수입니다. 이 값은 소스 엔드포인트의 트랜잭션 변경 비율 측정치와 같지 않습니다. 이 지표의 숫자가 많으면 일반적으로 AWS DMS가 캡처된 변경 사항을 적시에 적용할 수 없기 때문에 대상 지연 시간이 길어집니다.
CDCLatencySource	초	소스 엔드포인트에서 캡처한 마지막 이벤트와 AWS DMS 인스턴스의 현재 시스템 타임스탬프 간의 간격(초)입니다. CDCLatencySource는 소스 인스턴스와 복제 인스턴스 간의 지연 시간을 나타냅니다. CDCLatencySource가 높으면 소스에서 변경 사항을 캡처하는 프로세스가 지연되었음을 나타냅니다. 지속적 복제의 지연 시간을 식별하기 위해 이 지표를 CDCLatencyTarget과 함께 확인할 수 있습니다. CDCLatencySource와 CDCLatencyTarget이 모두 높으면 CDCLatencySource를 먼저 조사하세요. 소스와 복제 간에 복제 지연이 없는 경우 CDCLatencySource는 0이 될 수 있습니다. 복제 작업이 소스의 트랜잭션 로그에서 다음 이벤트를 읽으려고 시도하고 소스에서 마지막으로 읽은 때와 비교하여 새 이벤트가 없는 경우에도 CDCLatencySource는 0이 될 수 있습니다. 이 경우 복제는 CDCLatencySource를 0으로 재설정합니다.
CDCLatencyTarget	초	대상에 커밋되기를 대기하는 첫 이벤트 타임스탬프와 AWS DMS 인스턴스의 현재 타임스탬프 사이의 간격(초)입니다. 대상 지연 시간은 복제 인스턴스 서버 시간과 대상 구성 요소에 전달된 확인되지 않은 가장 오래된 이벤트 ID 간의 차이입니다. 즉, 대상 지연 시간은 복제 인스턴스와 적용되었지만 TRG 엔드포인트에서 확인되지 않은 가장 오래된 이벤트 간의 타임스탬프 차이

지표	단위	설명
		입니다(99%). CDCLatencyTarget이 높으면 대상에 변경 이벤트를 적용하는 프로세스가 지연되었음을 나타냅니다. 지속적 복제의 지연 시간을 식별하기 위해 이 지표를 CDCLatencySource와 함께 확인할 수 있습니다. CDCLatencyTarget은 높지만 CDCLatencySource는 높지 않은 경우 다음을 조사하세요. • 대상에 프라이머리 키 또는 인덱스가 없음 • 대상 또는 복제 인스턴스에서 리소스 병목 현상이 발생함 • 복제와 대상 사이에 네트워크 문제가 있음
CDCThroughputBandwidthTarget	초당 KB	대상의 발신 데이터(초당 KB 단위). CDCThroughputBandwidth는 샘플링 포인트에서 전송된 송신 데이터를 기록합니다. 어떠한 네트워크 트래픽도 찾을 수 없다면 이 값은 0입니다. CDC는 장시간 트랜잭션을 할 수 없기 때문에 네트워크 트래픽은 기록되지 않을 수 있습니다.
CDCThroughputRowsSource	초당 행 수	소스의 수신 변경 사항 개수(초당 행 수 단위).
CDCThroughputRowsTarget	초당 행 수	대상의 발신 변경 사항 개수(초당 행 수 단위).
FullLoadThroughputBandwidthTarget	초당 KB	대상 전체 로드의 발신 데이터(초당 KB 단위).
FullLoadThroughputRowsTarget	초당 행 수	대상 전체 로드의 발신 변경 사항(초당 행 수 단위).

AWS DMS 서버리스 복제 로그

Amazon CloudWatch를 사용하여 AWS DMS 마이그레이션 프로세스 중에 복제 정보를 로깅할 수 있습니다. 복제 설정을 선택할 때 로깅을 활성화합니다.

서버리스 복제는 상태 로그를 CloudWatch 계정에 업로드하여 복제 진행 상황에 대한 가시성을 향상하고, 문제 해결을 지원합니다.

AWS DMS 는 접두사가 인 전용 로그 그룹에 서버리스 연결 로그를 업로드합니다 `dms-serverless-replication-<your replication config resource ID>`. 이 로그 그룹에는 `dms-serverless-replication-orchestrator-<your replication config resource ID>`라는 로그 스트림이 있습니다. 이 로그 스트림은 복제의 복제 상태를 보고하고, 이 단계에서 진행 중인 작업에 대한 추가 세부 정보를 제공하는 관련 메시지를 보고합니다. 로그 항목의 예는 아래의 [서버리스 복제 로그 예제](#) 섹션을 참조하세요.

Note

AWS DMS 는 복제를 실행할 때까지 로그 그룹 또는 스트림을 생성하지 않습니다. 는 복제만 생성하는 경우 로그 그룹 또는 스트림을 생성하지 않습니다.

실행된 복제의 로그를 보려면 다음 단계를 따르세요.

1. AWS DMS 콘솔을 열고 탐색 창에서 서버리스 복제를 선택합니다. 서버리스 복제 대화 상자가 나타납니다.
2. 구성 섹션으로 이동하고 일반 열에서 서버리스 로그 보기를 선택합니다. CloudWatch 로그 그룹이 열립니다.

복제에 실패하면는 복제 상태가 인 로그 항목 `failed`과 실패 이유를 설명하는 메시지를 AWS DMS 생성합니다. 실패한 복제 문제를 해결하려면 첫 번째 단계로 CloudWatch 로그를 확인해야 합니다.

Note

AWS DMS 표준과 마찬가지로 데이터 마이그레이션 자체의 진행 상황, 즉 기본 복제 작업에서 내보낸 로그에 대해 보다 세분화된 로깅을 활성화할 수 있습니다. 아래의 JSON 예제에 나온 것처럼, Logging 필드에서 `EnableLogging`을 `true`로 설정하면 복제 설정에서 이러한 로그를 활성화할 수 있습니다.

```
{
```

```
"Logging": {
  "EnableLogging": true
}
```

이러한 로그를 활성화하면 서버리스 복제의 running 단계에서만 로그가 나타나기 시작합니다. 로그는 이전 로그 스트림과 동일한 로그 그룹 아래에 표시되지만, 새 로그 스트림인 `dms-serverless-serv-res-id-{unique identifier}` 아래에 표시됩니다. 서버리스 복제 로그를 해석하는 방법에 대한 내용은 다음 섹션을 참조하세요.

서버리스 복제 로그 예제

이 섹션에는 서버리스 복제에 대한 로그 항목의 예가 포함되어 있습니다.

예: 복제 시작

서버리스 복제를 실행하면는 다음과 유사한 로그 항목을 AWS DMS 생성합니다.

```
{'replication_state':'initializing', 'message': 'Initializing the replication workflow.'}
```

예: 복제 실패

복제의 엔드포인트 중 하나가 올바르게 구성되지 않은 경우는 다음과 유사한 로그 항목을 AWS DMS 생성합니다.

```
{'replication_state':'failed', 'message': 'Test connection failed for endpoint X.', 'failure_message': 'X'}
```

실패 후 로그에 이 메시지가 표시되면 지정된 엔드포인트가 정상이고 올바르게 구성되어 있는지 확인하세요.

Oracle에서 Amazon Redshift 및 Amazon S3로의 전체 로드 마이그레이션을 위한 향상된 처리량

AWS DMS 는 Oracle에서 Amazon Redshift 및 Amazon S3로의 전체 로드 마이그레이션을 위해 처리량 성능을 크게 개선했습니다. DMS는 테이블 매핑의 사용자 지정 `parallel-load` 옵션 없이 테이블에 대해 이 기능을 자동으로 활성화합니다. 사용자 지정 병렬 로드 옵션이 있는 테이블의 경우 DMS

Serverless는 지정된 테이블 매핑 구성을 기반으로 테이블 로드를 분산합니다. 향상된 처리량을 사용하려면 다음을 수행합니다.

- 파티션 또는 경계를 참조하지 않는 선택 규칙을 제공합니다. 예를 들어 테이블 매핑의 테이블 설정에 `parallel-load`가 포함된 경우 DMS Serverless는 향상된 처리량 기능을 사용하지 않습니다. 자세한 내용은 [선택 규칙 및 작업](#) 단원을 참조하십시오.
- `MaxFileSize` 및 `WriteBufferSize`를 64MB로 설정합니다. 자세한 내용은 [Amazon Redshift를 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.
- 데이터가 희소한 데이터 저장소의 경우 `CompressCsvFiles`를 `true`로 설정하고, 데이터 밀도가 높은 데이터 저장소의 경우 `false`로 설정하는 것이 좋습니다.
- 다음 태스크 설정을 0으로 설정합니다.
 - `ParallelLoadThreads`
 - `ParallelLoadQueuesPerThread`
 - `ParallelApplyThreads`
 - `ParallelApplyQueuesPerThread`
 - `ParallelLoadBufferSize`
- 병렬 데이터 마이그레이션을 지원하려면 `MaxFullLoadSubTasks`를 49로 설정합니다.
- LOB mode를 `inline`으로 설정합니다. 자세한 내용은 [AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정](#) 단원을 참조하십시오.

AWS DMS 는 다음 복제에 대해 향상된 처리량 성능을 제공하지 않습니다.

- 병렬 로드를 사용하는 테이블을 사용한 복제. 자세한 내용은 [선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용](#) 단원을 참조하십시오.
- 데이터 변환 규칙을 사용한 복제.
- 필터 규칙을 사용한 복제.
- 변환 규칙을 사용한 복제.
- Amazon Redshift Serverless를 대상으로 사용하는 복제.

AWS DMS 서버리스의 스토리지 Autoscaling 이해

복제 프로세스를 시작하면 AWS DMS 서버리스는 복제에 100GB의 초기 스토리지를 할당합니다. 스토리지는 주로 로그 파일과 캐시된 트랜잭션에서 소모합니다. 캐시 트랜잭션에서 스토리지는 캐시된 트

랜잭션을 디스크에 기록해야 할 때에만 사용됩니다. 따라서 AWS DMS Serverless는 상당한 양의 스토리지를 사용하지 않습니다. 일부 예외 사항은 다음과 같습니다.

- 상당한 트랜잭션 로드를 발생하는 매우 큰 테이블. 큰 테이블을 로드하면 다소 시간이 걸릴 수 있으므로, 캐시된 트랜잭션은 큰 테이블 로드 중에 디스크에 기록될 가능성이 더 높습니다.
- 캐시된 트랜잭션을 로드하기 전에 일시 중지되도록 구성되는 작업입니다. 이 경우에 모든 테이블에서 전체 로드가 완료될 때까지 모든 트랜잭션이 캐시됩니다. 이 구성에서 상당량의 스토리지가 캐시된 트랜잭션에서 소모될 수 있습니다.
- Amazon Redshift로 로드되고 있는 테이블을 사용하여 구성된 작업입니다. 이 구성은 Amazon Aurora가 대상일 때는 문제가 아닙니다.

따라서 AWS DMS 서버리스는 15분마다 스토리지 사용률을 모니터링합니다. 할당된 스토리지가 90% 사용되면 AWS DMS 서버리스는 추가 스토리지로 복제를 확장합니다. 스케일링 프로세스 이전 또는 도중에 복제의 100% 스토리지가 사용되고 복제 작업이 실패하는 경우 DMS Serverless는 스케일링이 성공적으로 완료되면 작업을 재개합니다.

Note

- 이전에 중지된 작업을 재개할 때 미완료 테이블에 대해 처음부터 전체 로드 작업이 다시 시작됩니다.
- 스토리지 스케일링 이벤트 중에는 DMS 작업 성능에 영향이 미치지 않습니다.
- 두 스토리지 오토 스케일링 이벤트 사이에는 냉각 기간이 없습니다.

AWS DMS 서버리스 제한 사항

AWS DMS Serverless에는 다음과 같은 제한 사항이 있습니다.

- CREATED, STOPPED 또는 FAILED 상태의 AWS DMS 복제 구성만 수정할 수 있습니다. 어떤 조건에서 어떤 설정을 변경할 수 있는지에 대한 자세한 내용은 [AWS DMS 서버리스 복제 수정](#) 섹션을 참조하세요.
- STOPPED 또는 FAILED 상태에 있는 AWS DMS 복제 구성만 삭제할 수 있습니다.
- 복제 인스턴스와 달리 AWS DMS 서버리스 복제에는 관리 작업을 위한 퍼블릭 IP 주소가 없습니다. 콘솔을 사용하여 서버리스 복제를 관리합니다.

- 이 AWS DMS 서버리스 릴리스는 AWS DMS 표준이 지원하는 모든 소스 및 대상 엔드포인트 유형을 지원하지 않습니다. 지원되는 엔진 유형의 목록은 [AWS DMS 서버리스 구성 요소](#) 섹션을 참조하세요.
- 서버리스 복제는 VPC 엔드포인트를 사용하여 종속성에 액세스해야 합니다. VPC 엔드포인트를 사용하여 아래와 같은 엔드포인트 유형에 액세스해야 합니다.
 - Amazon Amazon S3
 - Amazon Kinesis
 - AWS Secrets Manager
 - Amazon DynamoDB
 - Amazon Redshift
 - Amazon OpenSearch Service

VPC 엔드포인트 설정에 대한 내용은 [에 대한 VPC 엔드포인트 구성 AWS DMS](#) 섹션을 참조하세요.

- AWS DMS 서버리스는 뷰를 지원하지 않습니다.
- AWS DMS Serverless는 DB2 엔드포인트에 대한 SSL 연결을 지원하지 않습니다.
- AWS DMS Serverless는 사용자 지정 CDC 시작점 설정을 지원하지 않습니다.
- 복제 작업이 프로비저닝 해제된 상태이면 테이블 및 복제 통계와 관련된 메타데이터가 손실됩니다.

AWS DMS 서버리스 사전 마이그레이션.

AWS DMS Serverless에는 데이터베이스 마이그레이션을 시작하기 전에 잠재적 문제를 식별하는 데 도움이 되는 마이그레이션 전 평가 기능이 포함되어 있습니다. 마이그레이션 전 평가를 실행하면 성공적인 서버리스 복제를 방해할 수 있는 구성 문제 또는 호환성 문제를 감지하고 해결할 수 있습니다. 자세한 내용은 [개별 평가](#)를 참조하세요.

AWS DMS 표준과 달리 AWS DMS 서버리스는 사전 마이그레이션 평가 결과를 시스템 관리형 Amazon S3 버킷에 자동으로 저장하므로 사용자 지정 버킷을 지정할 필요가 없습니다.

AWS DMS Serverless는 마이그레이션 전 평가를 지원하기 위해 다음과 같은 선택적 설정을 제공합니다.

- **ResultLocationFolder:** 이 평가 실행의 결과를 AWS DMS 저장할 Amazon S3 버킷 내의 폴더입니다.

- **ResultEncryptionMode**: 지원되는 값은 SSE_KMS 및 SSE_S3입니다. 이러한 값이 제공되지 않으면 저장 시 파일이 암호화되지 않습니다. 자세한 내용은 [Amazon S3 대상 객체를 암호화하기 위한 AWS KMS 키 생성](#)을 참조하세요.
- **ResultKmsKeyArn**: ResultEncryptionMode를 SSE_KMS로 설정할 때 지정하는 고객 KMS 암호화 키의 ARN입니다.
- **IncludeOnly**: 포함하려는 특정 개별 평가에 대해 공백으로 구분된 이름 목록입니다. 이러한 이름은 AWS DMS가 관련된 마이그레이션에 대해 지원하는 개별 평가의 기본 목록에서 가져옵니다.
- **Exclude**: 제외하려는 특정 개별 평가에 대해 공백으로 구분된 이름 목록입니다. 이러한 이름은 연결된 마이그레이션에 대해 AWS DMS가 지원하는 개별 평가의 기본 목록에서 가져옵니다.
- **FailOnAssessmentFailure**: true(기본 설정) 또는 false로 설정할 수 있는 구성 가능한 설정입니다. 평가에 실패할 경우 복제가 자동으로 시작되지 않도록 이 설정을 사용합니다. 이렇게 하면 복제가 성공적으로 실행되지 않는 문제를 평가하는 데 도움이 될 수 있습니다.

KMS 키를 사용하여 파일 암호화

고객 관리형 키를 사용하여 DMS Serverless 마이그레이션 전 평가를 위해 SSE-KMS를 구성하려면 KMS 키에 DMS 서비스 연결 역할을 부여하는 정책 설명을 추가하여 평가 프로세스 중에 데이터의 보안 암호화 및 복호화를 활성화합니다. kms:GenerateDataKey 및 kms:Decrypt 권한을 구성해야 합니다. 아래 예를 참조하세요.

```
{
  "Sid": "AccessForDMSServerlessPremigration",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::<CustomerAccountId>:role/aws-service-role/dms.amazonaws.com/AWSServiceRoleForDMSServerless"
  },
  "Action": [
    "kms:Decrypt",
    "kms:GenerateDataKey*"
  ],
  "Resource": "*"
}
```

제한 사항

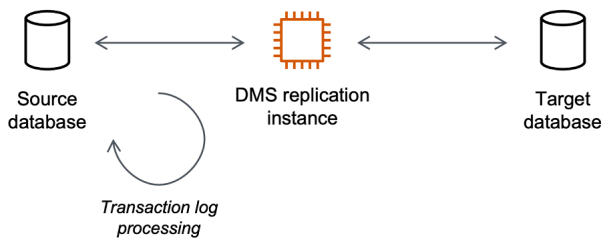
서버리스 마이그레이션 전 평가에는 다음과 같은 제한 사항이 있습니다.

- AWS DMS 서버리스는 describe-replications API를 호출할 때 가장 최근의 마이그레이션 전 평가 결과만 유지합니다. 이전 평가 실행은 즉시 표시에서 제거되지만 해당 결과 파일은 S3 결과 버킷에서 계속 액세스할 수 있습니다.
- 평가 결과를 저장하도록 사용자 지정 S3 버킷을 선택할 수 없습니다.
- 스키마, 테이블, 열 재매핑에 대한 변환은 사전 배포에서 지원되지 않습니다.

AWS DMS 복제 인스턴스 작업

AWS DMS 복제 인스턴스를 생성할 때는 Amazon VPC 서비스를 기반으로 Virtual Private Cloud(VPC)의 Amazon EC2 인스턴스에서 복제 인스턴스를 AWS DMS 생성합니다. 이 복제 인스턴스를 사용하여 데이터베이스 마이그레이션을 수행합니다. 복제 인스턴스를 사용하면 다중 AZ 옵션을 선택할 때 다중 AZ 배포로고가용성 및 장애 조치 지원을 확보할 수 있습니다.

다중 AZ 배포에서는 다른 가용 영역에 복제 인스턴스의 동기 대기 복제본을 AWS DMS 자동으로 프로비저닝하고 유지합니다. 기본 복제 인스턴스는 가용 영역 전체에서 대기 복제본으로 동기식으로 복제됩니다. 이 접근 방식에서는 데이터 중복을 제공하고 I/O 중지를 없애며 지연 시간 스파이크를 최소화합니다.



AWS DMS 는 복제 인스턴스를 사용하여 소스 데이터 스토어에 연결하고, 소스 데이터를 읽고, 대상 데이터 스토어에서 사용할 데이터의 형식을 지정합니다. 복제 인스턴스는 또한 데이터를 대상 데이터 저장소에 로드합니다. 이 절차 대다수는 메모리에서 진행됩니다. 그렇지만, 대규모 트랜잭션은 디스크에서 일부 버퍼링이 필요할 수 있습니다. 캐시된 트랜잭션과 로그 파일도 디스크에 기록됩니다.

다음 AWS 리전에서 AWS DMS 복제 인스턴스를 생성할 수 있습니다.

리전 이름	리전	엔드포인트	프로토콜
미국 동부 (오하이오)	us-east-2	dms.us-east-2.amazonaws.com	HTTPS
		dms-fips.us-east-2.amazonaws.com	HTTPS
미국 동부 (버지니아 북부)	us-east-1	dms.us-east-1.amazonaws.com	HTTPS
		dms-fips.us-east-1.amazonaws.com	HTTPS
미국 서부 (캘리포니아 북부)	us-west-1	dms.us-west-1.amazonaws.com	HTTPS
		dms-fips.us-west-1.amazonaws.com	HTTPS

리전 이름	리전	엔드포인트	프로토콜	
미국 서부 (오리곤)	us-west-2	dms.us-west-2.amazonaws.com	HTTPS	
		dms-fips.us-west-2.amazonaws.com	HTTPS	
아프리카 (케이프타운)	af-south-1	dms.af-south-1.amazonaws.com	HTTPS	
아시아 태평양(홍콩)	ap-east-1	dms.ap-east-1.amazonaws.com	HTTPS	
아시아 태평양(하이데라바드)	ap-south-2	dms.ap-south-2.amazonaws.com	HTTPS	
아시아 태평양(자카르타)	ap-southeast-3	dms.ap-southeast-3.amazonaws.com	HTTPS	
아시아 태평양(말레이시아)	ap-southeast-5	dms.ap-southeast-5.amazonaws.com	HTTPS	
아시아 태평양(멜버른)	ap-southeast-4	dms.ap-southeast-4.amazonaws.com	HTTPS	
아시아 태평양(뭄바이)	ap-south-1	dms.ap-south-1.amazonaws.com	HTTPS	
아시아 태평양(뉴질랜드)	ap-southeast-6	dms.ap-southeast-6.amazonaws.com	HTTPS	

리전 이름	리전	엔드포인트	프로토콜	
아시아 태평양(오사카)	ap-northeast-3	dms.ap-northeast-3.amazonaws.com	HTTPS	
아시아 태평양(서울)	ap-northeast-2	dms.ap-northeast-2.amazonaws.com	HTTPS	
아시아 태평양(싱가포르)	ap-southeast-1	dms.ap-southeast-1.amazonaws.com	HTTPS	
아시아 태평양(시드니)	ap-southeast-2	dms.ap-southeast-2.amazonaws.com	HTTPS	
아시아 태평양(타이베이)	ap-east-2	dms.ap-east-2.amazonaws.com	HTTPS	
아시아 태평양(태국)	ap-southeast-7	dms.ap-southeast-7.amazonaws.com	HTTPS	
아시아 태평양(도쿄)	ap-northeast-1	dms.ap-northeast-1.amazonaws.com	HTTPS	
캐나다(중부)	ca-central-1	dms.ca-central-1.amazonaws.com	HTTPS	
		dms-fips.ca-central-1.amazonaws.com	HTTPS	
캐나다 서부(캘거리)	ca-west-1	dms.ca-west-1.amazonaws.com	HTTPS	
		dms-fips.ca-west-1.amazonaws.com	HTTPS	

리전 이름	리전	엔드포인트	프로토콜	
유럽(프랑크푸르트)	eu-central-1	dms.eu-central-1.amazonaws.com	HTTPS	
유럽(아일랜드)	eu-west-1	dms.eu-west-1.amazonaws.com	HTTPS	
유럽(런던)	eu-west-2	dms.eu-west-2.amazonaws.com	HTTPS	
유럽(밀라노)	eu-south-1	dms.eu-south-1.amazonaws.com	HTTPS	
유럽(파리)	eu-west-3	dms.eu-west-3.amazonaws.com	HTTPS	
유럽(스페인)	eu-south-2	dms.eu-south-2.amazonaws.com	HTTPS	
유럽(스톡홀름)	eu-north-1	dms.eu-north-1.amazonaws.com	HTTPS	
유럽(취리히)	eu-central-2	dms.eu-central-2.amazonaws.com	HTTPS	
이스라엘(텔아비브)	il-central-1	dms.il-central-1.amazonaws.com	HTTPS	
멕시코(중부)	mx-central-1	dms.mx-central-1.amazonaws.com	HTTPS	
중동(바레인)	me-south-1	dms.me-south-1.amazonaws.com	HTTPS	
중동(UAE)	me-central-1	dms.me-central-1.amazonaws.com	HTTPS	

리전 이름	리전	엔드포인트	프로토콜
남아메리카(상파울루)	sa-east-1	dms.sa-east-1.amazonaws.com	HTTPS
AWS GovCloud(미국 동부)	us-gov-east-1	dms.us-gov-east-1.amazonaws.com	HTTPS
AWS GovCloud(미국 서부)	us-gov-west-1	dms.us-gov-west-1.amazonaws.com	HTTPS

AWS DMS 는 미국 정부 기관 및 고객이 민감한 워크로드를 클라우드로 이동할 수 있도록 설계된 라는 특수 AWS 리전을 지원합니다. AWS GovCloud (US) AWS GovCloud (US) 는 미국 정부의 특정 규제 및 규정 준수 요구 사항을 해결합니다. 에 대한 자세한 내용은 [GovCloud\(미국\)란 AWS 무엇입니까?](#)를 AWS GovCloud (US)참조하십시오.

아래에서 복제 인스턴스에 대한 자세한 정보를 확인할 수 있습니다.

주제

- [마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택](#)
- [복제 인스턴스에 가장 적합한 크기 선택](#)
- [복제 엔진 버전 작업](#)
- [퍼블릭 및 프라이빗 복제 인스턴스](#)
- [IP 주소 지정 및 네트워크 유형](#)
- [복제 인스턴스용으로 네트워크 설정](#)
- [복제 인스턴스의 암호화 키 설정](#)
- [복제 인스턴스 생성](#)
- [복제 인스턴스 수정](#)
- [복제 인스턴스 재부팅](#)
- [복제 인스턴스 삭제](#)

• [AWS DMS 유지 관리 기간 작업](#)

마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택

AWS DMS 는 Amazon EC2 인스턴스에서 복제 인스턴스를 생성합니다.는 AWS DMS 현재 복제 인스턴스에 대한 T3, C5, C6i, R5 및 R6i Amazon EC2 인스턴스 클래스를 지원합니다.

- T3 인스턴스는 차세대 버스트 가능 범용 인스턴스 유형입니다. 이 유형은 기본 수준의 성능과 함께, 필요할 때는 언제든지 CPU 사용량을 버스트할 수 있는 기능을 제공합니다. T3 인스턴스는 컴퓨팅, 메모리 및 네트워크 리소스를 균형 있게 제공하고 CUP 사용량이 중간 정도이지만 일시적으로 사용량이 급증하는 워크로드에 적합합니다. T3 인스턴스는 워크로드가 기준 임계값 이하로 작동할 때 CPU 크레딧을 누적합니다. 획득한 각 CPU 크레딧은 T3 인스턴스가 필요할 때 1분 동안 전체 CPU 코어 성능으로 버스트할 수 있는 기회를 제공합니다.

T3 인스턴스는 언제든지 unlimited 모드에서 필요한 기간 동안 버스트할 수 있습니다.

unlimited 모드에 대한 자세한 내용은 [성능 버스트 가능 인스턴스의 무제한 모드 사용](#) 섹션을 참조하세요.

- C5 인스턴스는 고급 컴퓨팅 집약적 워크로드를 실행하기 위해 높은 컴퓨팅 가성비로 비용 효율적인 고성능을 제공하는 차세대 인스턴스 유형입니다. 여기에는 고성능 웹 서버, 고성능 컴퓨팅(HPC), 배치 처리, 광고 서비스, 고확장성 멀티플라이어 게임, 비디오 인코딩 등의 워크로드가 포함됩니다. C5 인스턴스가 적합한 다른 워크로드에는 과학 모델링, 분산 분석, 기계 학습, 딥 러닝 추론 등이 있습니다. C5 인스턴스는 Intel 및 AMD의 다양한 프로세서를 사용할 수 있습니다.
- C6i 인스턴스는 다양한 워크로드에 대해 동급 5세대 인스턴스보다 최대 15% 더 우수한 컴퓨팅 가격 성능과 상시 메모리 암호화를 제공합니다. C6i 인스턴스는 배치 처리, 분산 분석, 고성능 컴퓨팅(HPC), 광고 서비스, 고확장성 멀티플라이어 게임, 비디오 인코딩과 같은 컴퓨팅 집약적 워크로드에 매우 적합합니다.
- R5 인스턴스는 Amazon EC2를 위한 차세대 메모리 최적화 인스턴스 유형입니다. R5 인스턴스는 고성능 데이터베이스, 분산형 웹 스케일 인 메모리 캐시, 중간 규모 인 메모리 데이터베이스, 실시간 빅 데이터 분석 및 기타 엔터프라이즈 애플리케이션 등 메모리 집약적 애플리케이션에 매우 적합합니다. 를 사용하여 처리량이 많은 트랜잭션 시스템을 지속적으로 마이그레이션하거나 복제하면 많은 양의 CPU와 메모리를 소비 AWS DMS 할 수도 있습니다.
- R6i 인스턴스는 다양한 워크로드에 대해 동급 5세대 인스턴스보다 최대 15% 더 우수한 컴퓨팅 가격 성능과 상시 메모리 암호화를 제공합니다. R6i 인스턴스는 SAP 인증을 받았으며 SQL 및 noSQL 데이터베이스와 같은 워크로드, Memcached 및 Redis OSS와 같은 분산형 웹 스케일 인 메모리 캐시, SAP HANA와 같은 인 메모리 데이터베이스, Hadoop 및 Spark 클러스터와 같은 실시간 빅 데이터 분석에 적합합니다.

- C7i 인스턴스는 비슷한 이전 세대 인스턴스에 비해 더 나은 컴퓨팅 성능을 제공합니다. AWS DMS 워크로드의 경우 C7i 인스턴스는 데이터 변환 프로세스를 가속화하고, 컴퓨팅 집약적인 스키마 변환을 처리하고, 대용량 마이그레이션 작업 중에 일관된 처리량을 유지하는 데 뛰어납니다. 이러한 인스턴스는 지속적인 CPU 성능이 필요한 컴퓨팅 성능의 이상적인 균형을 제공합니다.
- R7i 인스턴스는 메모리 집약적인 워크로드를 위한 높은 메모리 용량과 결합된 유사한 이전 세대 인스턴스보다 더 나은 컴퓨팅 성능을 제공합니다. AWS DMS 워크로드의 경우 R7i 인스턴스는 대용량의 동시 데이터베이스 트랜잭션을 처리하는 대규모 데이터베이스와 관련된 작업에 특히 적합하므로 메모리 집약적인 복제 시나리오와 상당한 메모리 버퍼가 필요한 복잡한 데이터 검증 프로세스를 효율적으로 처리할 수 있습니다.

각 복제 인스턴스에는 특정 메모리 및 vCPU 구성이 있습니다. 다음 표에는 각 복제 인스턴스 유형별 구성이 나와 있습니다. 요금 정보는 [AWS Database Migration Service 서비스 요금 페이지](#)를 참조하세요.

범용 복제 인스턴스 유형

Type	vCPU	메모리(GiB)
dms.t3.micro	2	1
dms.t3.small	2	2
dms.t3.medium	2	4
dms.t3.large	2	8

컴퓨팅 최적화 복제 인스턴스 유형

Type	vCPU	메모리(GiB)
dms.c5.large	2	4
dms.c5.xlarge	4	8
dms.c5.2xlarge	8	16
dms.c5.4xlarge	16	32

Type	vCPU	메모리(GiB)
dms.c5.9xlarge	36	72
dms.c5.12xlarge	48	96
dms.c5.18xlarge	72	144
dms.c5.24xlarge	96	192
dms.c6i.large	2	4
dms.c6i.xlarge	4	8
dms.c6i.2xlarge	8	16
dms.c6i.4xlarge	16	32
dms.c6i.8xlarge	32	64
dms.c6i.12xlarge	48	96
dms.c6i.16xlarge	64	128
dms.c6i.24xlarge	96	192
dms.c6i.32xlarge	128	256
dms.c7i.large	2	4
dms.c7i.xlarge	4	8
dms.x7i.2xlarge	8	16
dms.x7i.4xlarge	16	32
dms.x7i.8xlarge	32	64
dms.x7i.12xlarge	48	96
dms.x7i.16xlarge	64	128

Type	vCPU	메모리(GiB)
dms.x7i.24xlarge	96	192
dms.x7i.48xlarge	192	384

메모리 최적화 복제 인스턴스 유형

Type	vCPU	메모리(GiB)
dms.r5.large	2	16
dms.r4.xlarge	4	32
dms.r5.2xlarge	8	64
dms.r5.4xLarge	16	128
dms.r5.8xlarge	32	256
dms.r5.12xLarge	48	384
dms.r5.16 x Large	64	512
dms.r5.24 xlarge	96	768
dms.r6i.large	2	16
dms.r6i.xlarge	4	32
dms.r6i.2xlarge	8	64
dms.r6i.4xlarge	16	128
dms.r6i.8xlarge	32	256
dms.r6i.12xlarge	48	384
dms.r6i.16xlarge	64	512

Type	vCPU	메모리(GiB)
dms.r6i.24xlarge	96	768
dms.r6i.32xlarge	128	1024
dms.r7i.large	2	16
dms.r7i.xlarge	4	32
dms.r7i.2xlarge	8	64
dms.r7i.4xlarge	16	128
dms.r7i.8xlarge	32	256
dms.r7i.12xlarge	48	384
dms.r7i.16xlarge	64	512
dms.r7i.24xlarge	96	768
dms.r7i.48xlarge	192	1536

위의 표에는 모든 AWS DMS 복제 인스턴스 유형이 나열되어 있지만 해당 리전에서 사용할 수 있는 유형은 다를 수 있습니다. 다음 [AWS CLI](#) 명령을 실행하여 해당 리전에서 사용 가능한 복제 인스턴스 유형을 확인할 수 있습니다.

```
aws dms describe-orderable-replication-instances --region your_region_name
```

주제

- [사용할 인스턴스 클래스 결정](#)
- [성능 버스트 가능 인스턴스의 무제한 모드 사용](#)

사용할 인스턴스 클래스 결정

어떤 복제 인스턴스 클래스가 가장 적합한지 결정하는 데 도움이 되도록에서 AWS DMS 사용하는 변경 데이터 캡처(CDC) 프로세스를 살펴보겠습니다.

사용자가 전체 로드 및 CDC 작업(대량 로드 및 지속적 복제)을 실행 중이라고 가정해 보시다. 이 경우 해당 작업에는 자체 SQLite 리포지토리가 있어 메타데이터 및 기타 정보를 저장할 수 있습니다. 가 전체 로드를 AWS DMS 시작하기 전에 다음 단계를 수행합니다.

- AWS DMS 는 소스 엔진의 트랜잭션 로그에서 마이그레이션하는 테이블에 대한 변경 사항 캡처를 시작합니다(이러한 캐시된 변경 사항이라고 함). 전체 로드가 완료되고 나면 이러한 캐시된 변경 사항이 수집되어 대상에 적용됩니다. 캐시된 변경 사항의 볼륨에 따라 이러한 변경 사항은 메모리로부터 직접 적용될 수 있습니다. 이 경우 변경 사항은 적용되기 전에 먼저 설정된 임계값에 이를 때까지 수집됩니다. 또는 디스크로부터 적용될 수 있습니다. 이 경우 변경 사항은 메모리에 유지될 수 없을 때 기록됩니다.
- 캐시된 변경 사항이 적용된 후 기본적으로는 대상 인스턴스에서 트랜잭션 적용 프로세스를 AWS DMS 시작합니다.

적용된 캐시된 변경 단계 및 지속적 복제 단계에서는 수신 및 발신 데이터에 대해 각각 하나씩 두 개의 스트림 버퍼를 AWS DMS 사용합니다. AWS DMS 또한 또 다른 메모리 버퍼인 정렬기라는 중요한 구성 요소를 사용합니다. 다음은 분류기 구성 요소의 두 가지 중요 용도입니다(다른 용도도 있음).

- 분류기는 모든 트랜잭션을 추적하고 발신 버퍼에 관련이 있는 트랜잭션만 전달합니다.
- 이를 통해 트랜잭션이 원본에서와 동일한 커밋 순서로 전달됩니다.

보시다시피 이와 같은 AWS DMS의 CDC용 아키텍처에는 세 가지 중요 메모리 버퍼가 있습니다. 이 버퍼 중에 메모리 부족을 겪는 것이 있다면 해당 마이그레이션에는 장애를 유발할 가능성이 있는 성능 문제가 있을 수 있습니다.

초당 트랜잭션(TPS)이 높은 과중한 워크로드를 이 아키텍처에 풀러징하는 경우 R5 및 R6i 인스턴스에서 제공하는 추가 메모리가 유용할 수 있습니다. R5 및 R6i 인스턴스를 사용하여 메모리에 대규모 트랜잭션을 유지하고 지속적 복제 중에 메모리 부족 문제를 방지할 수 있습니다.

성능 버스트 가능 인스턴스의 무제한 모드 사용

T3 인스턴스와 같이 unlimited로 구성된 성능 버스트 가능 인스턴스는 필요한 경우 언제든지 원하는 기간 동안 높은 CPU 사용률을 유지할 수 있습니다. 시간당 인스턴스 가격은 모든 CPU 사용량 급증을 자동으로 충당할 수 있습니다. 이는 24시간 또는 인스턴스 수명 중 더 짧은 기간 동안 인스턴스의 평균 CPU 사용률이 기준 이하인 경우에 적용됩니다.

대부분의 범용 워크로드에서 unlimited로 구성된 인스턴스는 추가 요금 없이 충분한 성능을 제공합니다. 인스턴스 실행에 장기간 높은 CPU 사용률이 필요한 경우, vCPU-시간당 추가 고정 요금으로

인스턴스를 실행할 수 있습니다. T3 인스턴스 요금에 대한 자세한 내용은 [AWS Database Migration Service](#)의 'T3 CPU 크레딧'을 참조하세요.

T3 인스턴스의 unlimited 모드에 대한 자세한 내용은 Amazon EC2 사용 설명서에서 [성능 버스트 가능 인스턴스의 무제한 모드](#)를 참조하세요.

Important

[AWS 프리 티어](#) 혜택이 적용되는 dms.t3.micro 인스턴스를 unlimited 모드에서 사용하는 경우에는 요금이 부과될 수 있습니다. 특히, 24시간 동안 평균 사용률이 인스턴스의 기준 사용률을 초과하면 요금이 적용될 수 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 [기준 사용률](#)을 참조하세요.

T3 인스턴스는 unlimited로 시작하도록 기본 설정되어 있습니다. 24시간 동안 평균 CPU 사용량이 기준을 초과하면 잉여 크레딧에 대한 요금이 발생합니다. 경우에 따라 T3 스팟 인스턴스를 unlimited로 시작하고 즉각적으로 단기간 사용하려는 경우가 있습니다. CPU 크레딧을 누적할 유효 시간 없이 이렇게 하면 잉여 크레딧에 대한 요금이 발생합니다. 더 높은 비용을 지불하지 않으려면 표준 모드에서 T3 스팟 인스턴스를 시작하는 것이 좋습니다. 자세한 내용은 Amazon EC2 사용 설명서의 [잉여 크레딧으로 요금 발생 가능](#), [T3 스팟 인스턴스](#) 및 [성능 버스트 가능 인스턴스의 표준 모드](#)를 참조하세요.

복제 인스턴스에 가장 적합한 크기 선택

적절한 복제 인스턴스를 선택하는 것은 사용 사례의 여러 요소에 따라 달라집니다. 복제 인스턴스 리소스가 사용되는 방식을 이해하는 데 도움이 되도록 다음 설명을 참조하세요. 여기서는 전체 로드 + CDC 작업의 일반적인 시나리오를 다룹니다.

전체 로드 작업 중에는 테이블을 개별적으로 AWS DMS 로드합니다. 기본적으로 한 번에 8개의 테이블이 로드됩니다. 전체 로드 작업 중에 소스에 대한 지속적인 변경 사항을 AWS DMS 캡처하므로 나중에 대상 엔드포인트에 변경 사항을 적용할 수 있습니다. 변경 사항은 메모리에 캐시되며, 사용 가능한 메모리가 모두 사용되면 변경 사항이 디스크에 캐시됩니다. 테이블에 대한 전체 로드 작업이 완료되면 캐시된 변경 사항을 대상 테이블에 AWS DMS 즉시 적용합니다.

테이블에 대한 모든 캐시 변경 사항이 적용된 후에는 대상 엔드포인트는 트랜잭션 일치 상태가 됩니다. 이 시점에서 대상은 마지막으로 캐시된 변경 사항과 관련하여 소스 엔드포인트와 동기화됩니다. AWS DMS 그런 다음은 소스와 대상 간의 지속적인 복제를 시작합니다. 이를 위해서는 소스 트랜잭션 로그에서 변경 작업을 AWS DMS 가져와 트랜잭션 일관성 있는 방식으로 대상에 적용합니다. (이 프로세스는 배치 최적화 적용이 선택되지 않은 것으로 가정합니다.)는 가능하면 복제 인스턴스의 메모리를 통해 지

속적인 변경 사항을 AWS DMS 스트리밍합니다. 그렇지 않으면 대상에 적용할 수 있을 때까지 복제 인스턴스의 디스크에 변경 사항을 AWS DMS 씁니다.

복제 인스턴스의 변경 처리 방식과 해당 프로세스에서 메모리가 사용되는 방식을 어느 정도 제어할 수 있습니다. 변경 처리를 조정하는 방법에 대한 자세한 내용은 [변경 처리 튜닝 설정](#) 섹션을 참조하세요.

고려해야 할 요소

메모리 및 디스크 공간은 사용 사례에 적합한 복제 인스턴스를 선택할 때 가장 중요한 요소입니다. 이 하에서는 복제 인스턴스를 선택하기 위해 분석해야 할 사용 사례 특성에 대한 설명합니다.

- 데이터베이스 및 테이블 크기

데이터 볼륨은 전체 로드 성능을 최적화하기 위한 태스크 구성을 결정하는 데 도움이 됩니다. 예를 들어 1TB 스키마 2개에 대해 테이블을 500GB 태스크 4개로 파티션하여 병렬로 실행할 수 있습니다. 가능한 병렬 처리는 복제 인스턴스에서 사용할 수 있는 CPU 리소스에 따라 달라집니다. 따라서 전체 로드 성능을 최적화하려면 데이터베이스와 테이블의 크기를 이해하는 것이 좋습니다. 이는 수행할 수 있는 태스크의 수를 결정하는 데 도움이 됩니다.

- 대형 객체

마이그레이션 범위에 있는 데이터 형식은 성능에 영향을 줄 수 있습니다. 특히 대형 객체(LOB)는 성능과 메모리 소비에 영향을 미칩니다. LOB 값을 마이그레이션하기 위해서는 2단계 프로세스를 AWS DMS 수행합니다. 먼저 LOB 값 없이 행을 대상에 AWS DMS 삽입합니다. 둘째, 행을 LOB 값으로 AWS DMS 업데이트합니다. 이는 메모리에 영향을 미치므로 소스에서 LOB 열을 식별하고 크기를 분석하는 것이 중요합니다.

- 로드 빈도 및 트랜잭션 크기

로드 빈도와 초당 트랜잭션 수(TPS)는 메모리 사용량에 영향을 줍니다. TPS 또는 데이터 조작 언어(DML) 활동이 많으면 메모리 사용량이 많아집니다. DMS는 변경 사항이 대상에 적용될 때까지 변경 사항을 캐시하기 때문입니다. CDC 중에 이로 인해 스와핑(메모리 오버플로로 인한 물리적 디스크에 쓰기)이 발생하여 지연 시간이 발생합니다.

- 테이블 키 및 참조 무결성

테이블의 키에 대한 정보에 따라 데이터 마이그레이션에 사용하는 CDC 모드(배치 적용 또는 트랜잭션 적용)가 결정됩니다. 일반적으로 트랜잭션 적용은 배치 적용보다 느립니다. 장기 실행 트랜잭션의 경우 마이그레이션해야 할 변경 사항이 많을 수 있습니다. 트랜잭션 적용을 사용하는 경우 일괄 적용에 비해 변경 사항을 저장하는 데 더 많은 메모리가 AWS DMS 필요할 수 있습니다. 프라이머리 키가 없는 테이블을 마이그레이션하는 경우 배치 적용이 실패하고 DMS 작업이 트랜잭션 적용 모드로 전

한됩니다. CDC 중에 테이블 간에 참조 무결성이 활성화되면 기본적으로 트랜잭션 AWS DMS 이 적용됩니다. 배치 적용과 트랜잭션 적용의 비교에 대한 자세한 내용은 [DMS 배치 적용 기능을 사용하여 CDC 복제 성능을 개선하려면 어떻게 해야 하나요?](#)를 참조하세요.

이러한 지표를 사용하여 복제 인스턴스를 컴퓨팅 최적화해야 하는지 아니면 메모리 최적화해야 하는지 결정하세요.

일반적인 문제

마이그레이션 중에 다음과 같이 복제 인스턴스에서 리소스 경합을 유발하는 일반적인 문제가 발생할 수 있습니다. 복제 인스턴스 지표에 대한 자세한 내용은 [복제 인스턴스 지표](#) 섹션을 참조하세요.

- 복제 인스턴스의 메모리가 부족해지면 데이터가 디스크에 쓰여집니다. 디스크에서 읽을 때 지연 시간이 발생할 수 있는데, 충분한 메모리로 복제 인스턴스 크기를 조정하면 지연 시간을 피할 수 있습니다.
- 복제 인스턴스에 할당된 디스크 크기는 필요한 것보다 작을 수 있습니다. 디스크 크기는 메모리에서 데이터가 넘칠 때 사용되며 태스크 로그를 저장하는 데도 사용됩니다. 최대 IOPS도 이에 따라 달라집니다.
- 여러 태스크를 실행하거나 병렬성이 높은 태스크를 실행하면 복제 인스턴스의 CPU 사용량에 영향을 줍니다. 이로 인해 태스크 처리 속도가 느려지고 지연 시간이 길어집니다.

모범 사례

복제 인스턴스의 크기를 조정할 때는 다음과 같이 가장 일반적인 2가지 모범 사례를 고려해야 합니다. 자세한 내용은 [의 모범 사례 AWS Database Migration Service](#) 단원을 참조하십시오.

1. 워크로드의 규모를 파악하고 컴퓨팅 집약적인지 메모리 집약적인지 판단하세요. 이를 기반으로 복제 인스턴스의 클래스 및 크기를 결정할 수 있습니다.
 - AWS DMS 는 메모리에서 LOBs 처리합니다. 이 작업에는 상당한 양의 메모리가 필요합니다.
 - 태스크 수와 스레드 수는 CPU 사용량에 영향을 줍니다. 전체 로드 작업 중에는 MaxFullLoadSubTasks를 8개 이상 사용하지 마세요.
2. 전체 로드 중에 워크로드가 많으면 복제 인스턴스에 할당된 디스크 공간을 늘리세요. 이렇게 하면 복제 인스턴스가 할당된 최대 IOPS를 사용할 수 있습니다.

위 가이드라인에서 가능한 시나리오를 모두 다루는 것은 아닙니다. 복제 인스턴스의 크기를 결정할 때 특정 사용 사례의 구체적인 내용을 고려하는 것이 중요합니다.

이전 테스트에서 CPU와 메모리가 워크로드에 따라 달라지는 것으로 나타났습니다. 특히 LOB는 메모리에 영향을 미치고 태스크 수 또는 병렬 처리는 CPU에 영향을 미칩니다. 마이그레이션을 실행한 후에는 복제 인스턴스의 CPU, 여유 메모리, 여유 스토리지, IOPS를 모니터링하세요. 수집한 데이터를 기반으로 필요에 따라 복제 인스턴스의 크기를 늘리거나 줄일 수 있습니다.

복제 엔진 버전 작업

복제 엔진은 복제 인스턴스에서 실행되고 지정한 마이그레이션 작업을 수행하는 핵심 AWS DMS 소프트웨어입니다. AWS는 새로운 기능 및 성능 개선과 함께 AWS DMS 복제 엔진 소프트웨어의 새 버전을 정기적으로 릴리스합니다. 복제 엔진 소프트웨어의 각 버전마다 고유한 버전 번호가 있어서 버전 간에 구분됩니다.

새 복제 인스턴스를 시작하면 달리 지정하지 않는 한 최신 AWS DMS 엔진 버전이 실행됩니다. 자세한 내용은 [AWS DMS 복제 인스턴스 작업](#) 단원을 참조하십시오.

현재 실행 중인 복제 인스턴스가 있는 경우 최신 엔진 버전으로 업그레이드할 수 있습니다(엔진 버전 다운그레이드 AWS DMS는 지원하지 않음). 복제 엔진 버전에 대한 자세한 내용은 [AWS DMS 릴리스 정보](#) 섹션을 참조하세요.

콘솔을 사용하여 엔진 버전 업그레이드

를 사용하여 AWS DMS 복제 인스턴스를 업그레이드할 수 있습니다 AWS Management Console.

콘솔을 사용하여 복제 인스턴스를 업그레이드하려면

1. <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 복제 인스턴스를 선택합니다.
3. 복제 엔진을 선택한 다음 수정을 선택합니다.
4. 엔진 버전에서 원하는 버전 번호를 선택한 다음 수정을 선택합니다.

Note

복제 인스턴스를 업그레이드하기 전에 모든 태스크를 중지하는 것이 좋습니다. 작업을 중지하지 않으면 AWS DMS는 업그레이드 전에 작업을 자동으로 중지합니다. 태스크를 수동으로 중지하는 경우 업그레이드가 완료된 후 태스크를 수동으로 시작해야 합니다. 복제 인스턴스를 업그레이드하는 데는 몇 분이 걸립니다. 인스턴스가 준비되면 인스턴스의 상태가 사용 가능으로 변경됩니다.

를 사용하여 엔진 버전 업그레이드 AWS CLI

다음과 AWS CLI같이 사용하여 AWS DMS 복제 인스턴스를 업그레이드할 수 있습니다.

를 사용하여 복제 인스턴스를 업그레이드하려면 AWS CLI

1. 다음 명령을 사용하여 복제 인스턴스의 Amazon 리소스 이름(ARN)을 확인합니다.

```
aws dms describe-replication-instances \
--query "ReplicationInstances[*].
[ReplicationInstanceIdentifier,ReplicationInstanceArn,ReplicationInstanceClass]"
```

출력에서 업그레이드할 복제 인스턴스에 대한 ARN을 적어둡니다. 예: `arn:aws:dms:us-east-1:123456789012:rep:6EFQQ06U6EDPRCPKLNPL2SCEEY`

2. 다음 명령을 사용하여 사용할 수 있는 복제 인스턴스 버전을 확인합니다.

```
aws dms describe-orderable-replication-instances \
--query "OrderableReplicationInstances[*].[ReplicationInstanceClass,EngineVersion]"
```

출력에서 복제 인스턴스 클래스에 사용할 수 있는 엔진 버전 번호를 적어둡니다. 1단계의 출력에 이 정보가 표시됩니다.

3. 다음 명령을 사용하여 복제 인스턴스를 업그레이드합니다.

```
aws dms modify-replication-instance \
--replication-instance-arn arn \
--engine-version n.n.n
```

앞의 *arn*을 전 단계에서 확인한 실제 복제 인스턴스 ARN으로 바꿉니다.

*n.n.n*을 원하는 엔진 버전으로 바꿉니다. 예: 3.4.5

Note

복제 인스턴스를 업그레이드하는 데는 몇 분이 걸립니다. 다음 명령을 사용하여 복제 인스턴스 상태를 볼 수 있습니다.

```
aws dms describe-replication-instances \
```

```
--query "ReplicationInstances[*].  
[ReplicationInstanceIdentifier,ReplicationInstanceStatus]"
```

복제 인스턴스가 준비되면 인스턴스의 상태가 사용 가능으로 변경됩니다.

퍼블릭 및 프라이빗 복제 인스턴스

복제 인스턴스가 원본과 대상 데이터베이스에 연결하는 데 사용할 퍼블릭 또는 프라이빗 IP 주소 여부를 지정할 수 있습니다.

프라이빗 복제 인스턴스에는 복제 네트워크 외부에서 액세스할 수 없는 프라이빗 IP 주소가 있습니다. 소스 및 대상 데이터베이스가 모두 복제 인스턴스의 가상 프라이빗 클라우드(VPC)에 연결된 동일한 네트워크에 있는 경우 프라이빗 인스턴스를 사용합니다. 가상 프라이빗 네트워크(VPN) Direct Connect 또는 VPC 피어링을 사용하여 네트워크를 VPC에 연결할 수 있습니다.

VPC 피어링 연결은 두 VPC 간의 네트워킹 연결입니다. 이를 통해 각 VPC의 프라이빗 IP 주소를 동일한 네트워크에 있는 것처럼 사용하여 라우팅할 수 있습니다. VPC 피어링에 관한 자세한 내용은 Amazon VPC 사용 설명서에서 [VPC 피어링](#)을 참조하세요.

퍼블릭 복제 인스턴스는 복제 인스턴스의 VPC 보안 그룹과 복제 인스턴스의 퍼블릭 IP 주소 또는 NAT 게이트웨이의 퍼블릭 IP 주소를 사용할 수 있습니다. 이 연결은 데이터 마이그레이션에서 사용하는 네트워크를 구성합니다.

IP 주소 지정 및 네트워크 유형

AWS DMS 는 항상 Amazon Virtual Private Cloud(VPC)에서 복제 인스턴스를 생성합니다. VPC를 생성할 때 사용할 IP 주소 지정(IPv4, IPv6 또는 둘 다)을 결정할 수 있습니다. 그런 다음 복제 인스턴스를 생성하거나 수정할 때 이중 스택 모드를 사용하여 IPv4 주소 프로토콜 또는 IPv6 주소 프로토콜의 사용을 지정할 수 있습니다.

IPv4 주소

VPC를 생성할 때 VPC의 IPv4 주소 범위를 CIDR(Classless Inter-Domain Routing) 블록 형태로 지정할 수 있습니다(예: 10.0.0.0/16). 서브넷 그룹은 이 CIDR 블록에서 IP 주소의 범위를 정의합니다. 이 IP 주소는 프라이빗 또는 퍼블릭일 수 있습니다.

프라이빗 IPv4 주소는 인터넷을 통해 연결할 수 없는 IP 주소입니다. 프라이빗 IPv4 주소는 복제 인스턴스 및 같은 VPC 내의 Amazon EC2 인스턴스와 같은 기타 리소스 간의 통신을 위해 사용될 수 있습니다. 각 복제 인스턴스에는 VPC 내 통신을 위한 프라이빗 IP 주소가 있습니다.

퍼블릭 IP 주소는 인터넷을 통해 연결할 수 있는 IPv4 주소입니다. 퍼블릭 주소는 복제 인스턴스와 인터넷 상의 리소스 사이의 통신을 위해 사용될 수 있습니다. 복제 인스턴스가 퍼블릭 IP 주소를 수신할지 여부를 제어할 수 있습니다.

이중 스택 모드 및 IPv6 주소

IPv6를 통해 복제 인스턴스와 통신해야 하는 리소스가 있는 경우 이중 스택 모드를 사용합니다. 이중 스택 모드를 사용하려면 복제 인스턴스와 연결하는 DB 서브넷 그룹의 각 서브넷에 IPv6 CIDR 블록이 연결되어 있어야 합니다. 새 복제 서브넷 그룹을 생성하거나 기존 복제 서브넷 그룹을 수정하여 이 요구 사항을 충족하도록 수정할 수 있습니다. 각 IPv6 주소는 전역적으로 고유합니다. VPC에 대한 IPv6 CIDR 블록은 Amazon의 IPv6 주소 풀에서 자동으로 할당됩니다. 범위를 직접 선택할 수는 없습니다.

DMS는 프라이빗 이중 스택 모드 복제 인스턴스의 IPv6 엔드포인트에 대한 인터넷 게이트웨이 액세스를 비활성화합니다. DMS는 IPv6 엔드포인트가 프라이빗이며 VPC 내에서만 액세스할 수 있도록 하기 위해 이 작업을 수행합니다.

AWS DMS 콘솔을 사용하여 복제 인스턴스를 생성 또는 수정하고 네트워크 유형 섹션에서 듀얼 스택 모드를 지정할 수 있습니다. 다음 이미지는 콘솔의 네트워크 유형 섹션을 보여 줍니다.

Connectivity and security

Network type - new [Info](#)

To use dual-stack mode, make sure that you associate an IPv6 CIDR block with a subnet in the VPC you specify.

☐ IPv4

Replication instance with an IPv4 network type that supports IPv4 addressing.

☒ Dual-stack mode

Replication instance with a dual network type that supports both IPv4 and IPv6 addressing.

참조

- IPv4 및 IPv6 주소에 대한 자세한 내용은 Amazon VPC 사용 설명서의 [IP 주소 지정](#)을 참조하세요.
- 이중 스택 모드를 사용한 복제 인스턴스 생성에 대한 자세한 내용은 [복제 인스턴스 생성](#) 섹션을 참조하세요.
- 복제 인스턴스 수정에 대한 자세한 내용은 [복제 인스턴스 수정](#) 섹션을 참조하세요.

복제 인스턴스용으로 네트워크 설정

AWS DMS는 항상 Amazon VPC를 기반으로 VPC에 복제 인스턴스를 생성합니다. 복제 인스턴스의 위치할 VPC는 사용자가 지정합니다. 계정 및 AWS 리전에 기본 VPC를 사용하거나 새 VPC를 생성할 수 있습니다.

복제 인스턴스의 VPC에 할당된 탄력적 네트워크 인터페이스가 보안 그룹과 연결되어 있는지 확인해야 합니다. 또한 이 보안 그룹의 규칙이 모든 포트에서 모든 트래픽이 VPC를 벗어나는 것(송신)을 허용해야 합니다. 이 접근 방식을 통해 엔드포인트에서 올바른 수신 규칙이 활성화되어 있는 한 통신을 복제 인스턴스에서 소스 및 대상 데이터베이스 엔드포인트로 전달할 수 있습니다. 모든 포트에서 모든 주소로의 송신을 허용할 수 있도록 엔드포인트에서 기본 설정을 사용하는 것이 좋습니다.

원본과 대상 엔드포인트는 VPC에 연결하거나 VPC 내부에 있어 VPC 내부에 있는 복제 인스턴스에 액세스합니다. 데이터베이스 엔드포인트는 복제 인스턴스에서 수신되는 액세스를 허용하는 네트워크 ACL(액세스 제어 목록)과 보안 그룹 규칙(해당하는 경우)을 포함해야 합니다. 설정 방법은 사용하는 네트워크 구성에 따라 다릅니다. 복제 인스턴스 VPC 보안 그룹, 복제 인스턴스의 프라이빗 또는 퍼블릭 IP 주소 또는 NAT 게이트웨이의 퍼블릭 IP 주소를 사용할 수 있습니다. 이 연결은 데이터 마이그레이션에서 사용하는 네트워크를 구성합니다.

Note

기본 인프라 변경으로 인해 IP 주소가 달라질 수 있으므로 VPC CIDR 범위를 사용하거나 NAT GW 연결 탄력적 IP를 통해 복제 인스턴스 아웃바운드 트래픽을 라우팅하는 것이 좋습니다. CIDR 블록을 포함하여 VPC 생성에 대한 자세한 내용은 Amazon Virtual Private Cloud 사용 설명서의 [VPC 및 서브넷 사용](#)을 참조하세요. 탄력적 IP 주소에 대한 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 [탄력적 IP 주소](#)를 참조하세요.

데이터베이스 마이그레이션을 위한 네트워크 구성

AWS Database Migration Service에서 여러 가지 네트워크 구성을 사용할 수 있습니다. 다음은 데이터베이스 마이그레이션에 사용되는 네트워크의 일반적인 구성입니다.

주제

- [하나의 VPC에 모든 데이터베이스 마이그레이션 구성 요소가 있는 구성](#)
- [여러 VPC를 사용한 구성](#)
- [공유 VPC를 사용한 구성](#)
- [Direct Connect 또는 VPN을 사용하여 네트워크를 VPC로 구성](#)

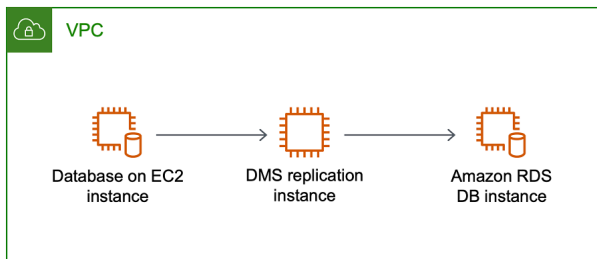
- [인터넷을 사용한 네트워크와 VPC 연결 구성](#)
- [ClassicLink를 사용하여 VPC에 없는 RDS DB 인스턴스를 VPC에 있는 DB 인스턴스와 연결하여 구성](#)
- [AWS 서비스에 연결하는 네트워크에 대한 구성](#)
- [VPC 엔드포인트를 사용하여 AWS 서비스에 연결하는 네트워크에 대한 구성](#)
- [인터넷을 사용하여 AWS 서비스에 연결하는 네트워크에 대한 구성](#)

가능하면 대상 엔드포인트와 동일한 리전, 대상 엔드포인트와 동일한 VPC 또는 서브넷에 DMS 복제 인스턴스를 생성하는 것이 좋습니다.

하나의 VPC에 모든 데이터베이스 마이그레이션 구성 요소가 있는 구성

데이터베이스 마이그레이션에서 가장 간단한 네트워크는 원본 엔드포인트, 복제 인스턴스, 대상 엔드포인트가 모두 동일한 VPC에 있도록 하는 것입니다. 이 구성은 소스 및 대상 엔드포인트가 Amazon RDS DB 인스턴스 또는 Amazon EC2 인스턴스에 있는 경우에 잘 맞습니다.

다음 그림은 Amazon EC2 인스턴스의 데이터베이스가 복제 인스턴스에 연결되고 데이터가 Amazon RDS DB 인스턴스에 마이그레이션되는 경우의 구성을 보여줍니다.



이 구성에서 사용되는 VPC 보안 그룹은 복제 인스턴스의 데이터베이스 포트에서 수신을 허용해야 합니다. 이 작업은 두 가지 방법으로 수행할 수 있습니다. 복제 인스턴스에서 사용하는 보안 그룹이 엔드포인트에서 수신을 허용하도록 할 수 있습니다. 또는 복제 인스턴스의 VPC CIDR 범위, NAT GW 탄력적 IP 또는 프라이빗 IP 주소(사용 중인 경우)를 허용할 수 있습니다. 하지만 복제 인스턴스의 프라이빗 IP 주소는 사용하지 않는 것이 좋습니다. 복제 IP 주소가 변경되면 복제가 중단될 수 있기 때문입니다.

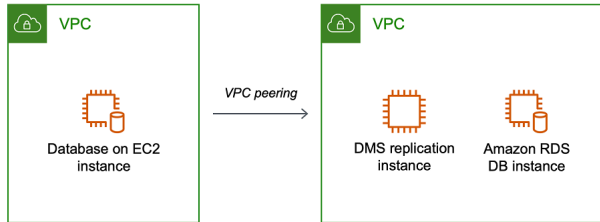
여러 VPC를 사용한 구성

소스 엔드포인트와 대상 엔드포인트가 다른 VPC에 있는 경우, VPC 중 하나에서 복제 인스턴스를 생성할 수 있습니다. 그런 다음 VPC 피어링을 사용하여 두 VPC를 연결할 수 있습니다.

VPC 피어링 연결은 두 VPC가 동일한 네트워크에 있는 것처럼 각 VPC의 프라이빗 IP 주소를 사용하여 라우팅을 지원하는 두 VPC 간 네트워크 연결입니다. 자체 VPC, 다른 AWS 계정 VPCs 또는 다른 AWS

리전의 VPC 간에 VPC 피어링 연결을 생성할 수 있습니다. VPC 피어링에 관한 자세한 내용은 Amazon VPC 사용 설명서에서 [VPC 피어링](#)을 참조하세요.

다음 그림에 VPC 피어링을 사용한 구성 예가 나와 있습니다. 여기서, VPC의 Amazon EC2 인스턴스에 있는 소스 데이터베이스는 VPC 피어링에 의해 VPC에 연결됩니다. 이 VPC에는 Amazon RDS DB 인스턴스의 복제 인스턴스 및 대상 데이터베이스가 들어 있습니다.



VPC 피어링을 구현하려면 Amazon Virtual Private Cloud, VPC 피어링 설명서에 있는 [VPC 피어링 연결 사용](#)의 지침을 따르세요. 한 VPC의 라우팅 테이블에 다른 VPC의 CIDR 블록이 포함되어야 합니다. 예를 들어 VPC A가 대상 10.0.0.0/16을 사용하고 VPC B가 대상 172.31.0.0을 사용하는 경우 VPC A의 라우팅 테이블에는 172.31.0.0이 포함되고 VPC B의 라우팅 테이블에는 10.0.0.0/16이 포함되어야 합니다. 자세한 내용은 Amazon Virtual Private Cloud, VPC 피어링 설명서의 [VPC 피어링 연결에 대한 라우팅 테이블 업데이트](#)를 참조하세요.

이 구성에 사용되는 VPC 보안 그룹은 복제 인스턴스의 데이터베이스 포트에서 수신을 허용하거나 피어링되는 VPC의 CIDR 블록에서 수신을 허용해야 합니다.

공유 VPC를 사용한 구성

AWS DMS 는 조직의 참여 고객 계정과 공유되는 서브넷을 동일한 계정의 일반 서브넷과 동일하게 처리합니다. 다음은 VPCs, 서브넷을 AWS DMS 처리하는 방법과 공유 VPCs.

ReplicationSubnetGroup 객체를 생성하여 사용자 지정 서브넷 또는 VPC에서 작동하도록 네트워크 구성을 구성할 수 있습니다. ReplicationSubnetGroup을 생성할 때 계정에 있는 특정 VPC의 서브넷을 지정할 수 있습니다. 지정하는 서브넷 목록에는 별도의 가용 영역에 있는 서브넷이 두 개 이상 포함되어야 하며 모든 서브넷은 동일한 VPC에 있어야 합니다. 를 생성할 때 ReplicationSubnetGroup객체는 서브넷만 지정합니다. AWS DMS 는 각 서브넷이 정확히 하나의 VPC에 연결되어 있으므로 사용자를 대신하여 VPC를 결정합니다.

또는 AWS DMS ReplicationInstance 생성할 때 ReplicationInstance 또는 서브넷이 복제가 작동하는 ReplicationSubnetGroup 및/또는 VPC 보안 그룹을 지정하도록 AWS DMS ReplicationConfig선택할 수 있습니다. 지정하지 않으면 고객 기본 ReplicationSubnetGroup(기본 VPC의 모든 서브넷에 지정하지 않은 경우에서 자동으로 AWS DMS 생성)과 기본 VPC 보안 그룹을 AWS DMS 선택합니다.

지정한 가용 영역 또는 ReplicationSubnetGroup의 가용 영역 중 하나에서 마이그레이션을 실행하도록 선택할 수 있습니다. 가 복제 인스턴스를 생성하거나 서버리스 복제를 시작 AWS DMS 하려고 하면 서브넷의 가용 영역을 코어 서비스 계정의 가용 영역으로 변환하여 가용 영역 매핑이 두 계정 간에 동일하지 않더라도 올바른 가용 영역에서 인스턴스를 시작할 수 있도록 합니다.

공유 VPC를 사용하는 경우 공유 VPC에서 사용하려는 서브넷에 매핑되는 ReplicationSubnetGroup 객체를 생성해야 합니다. ReplicationInstance 또는 ReplicationConfig를 생성할 때 공유 VPC에 대해 ReplicationSubnetGroup을 지정하고, Create 요청을 통해 공유 VPC용으로 생성한 VPC 보안 그룹을 지정해야 합니다.

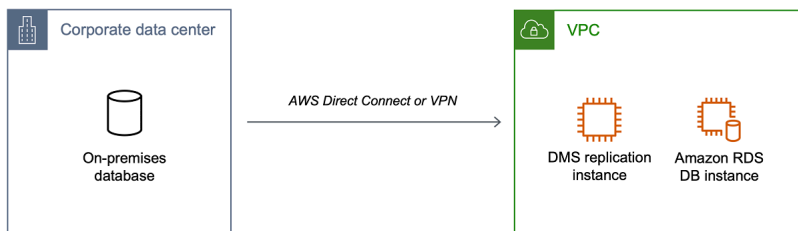
공유 VPC 사용 시 다음에 유의하세요.

- VPC 소유자는 참여자와 리소스를 공유할 수 없지만 참여자는 소유자의 서브넷에서 서비스 리소스를 생성할 수 있습니다.
- 모든 리소스는 계정별이므로 VPC 소유자는 참여자가 생성한 리소스(예: 복제 인스턴스)에 액세스할 수 없습니다. 하지만 공유 VPC에서 복제 인스턴스를 생성하면 복제 엔드포인트 또는 태스크에 올바른 권한이 있는 한 소유 계정과 관계없이 VPC의 리소스에 액세스할 수 있습니다.
- 리소스는 계정별이므로 다른 참여자는 다른 계정이 소유한 리소스에 액세스할 수 없습니다. 공유 VPC에서 생성된 리소스에 자신의 계정으로 액세스할 수 있도록 다른 계정에 부여할 수 있는 권한은 없습니다.

Direct Connect 또는 VPN을 사용하여 네트워크를 VPC로 구성

원격 네트워크는 AWS Direct Connect, 소프트웨어 또는 하드웨어 VPN 연결과 같은 여러 옵션을 사용하여 VPC에 연결할 수 있습니다. 이 옵션은 흔히 내부 네트워크를 AWS 클라우드로 확장함으로써 모니터링, 인증, 보안, 데이터 또는 기타 시스템과 같은 기존 현장 서비스를 통합하는 데 사용됩니다. 이러한 유형의 네트워크 확장을 활용하여 VPC와 같이 AWS호스팅 리소스에 완벽하게 연결할 수 있습니다.

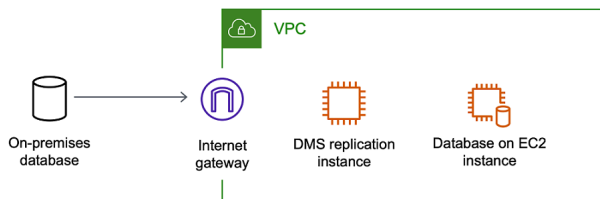
다음 그림은 원본 엔드포인트가 기업 데이터 센터에서 온프레미스 데이터베이스인 구성을 나타냅니다. Amazon RDS DB 인스턴스에서 복제 인스턴스와 대상 데이터베이스를 포함하는 Direct Connect 또는 VPN-VPC를 사용하여 연결합니다.



이 구성에서 VPC 보안 그룹에는 VPC CIDR 범위 또는 특정 IP 주소로 지정된 트래픽을 호스트로 전송하는 라우팅 규칙이 포함되어야 합니다. 이 호스트는 트래픽을 VPC에서 온프레미스 VPN으로 연결할 수 있어야 합니다. 이 경우, NAT 호스트에는 자체 보안 그룹 설정이 포함됩니다. 이러한 설정은 복제 인스턴스의 VPC CIDR 범위, 프라이빗 IP 주소 또는 보안 그룹에서 NAT 인스턴스로 들어오는 트래픽을 허용해야 합니다. 하지만 복제 인스턴스의 프라이빗 IP 주소는 사용하지 않는 것이 좋습니다. 복제 IP 주소가 변경되면 복제가 중단될 수 있기 때문입니다.

인터넷을 사용한 네트워크와 VPC 연결 구성

VPN 또는를 사용하여 AWS 리소스 Direct Connect 에 연결하지 않는 경우 인터넷을 사용하여 데이터베이스를 마이그레이션할 수 있습니다. 이 경우 Amazon EC2 인스턴스 또는 Amazon RDS DB 인스턴스로 마이그레이션할 수 있습니다. 이 구성은 대상 엔드포인트와 복제 인스턴스를 포함하는 인터넷 게이트웨이를 지원하는 VPC에 퍼블릭 복제 인스턴스를 포함합니다.



VPC에 인터넷 게이트웨이를 추가하려면 Amazon VPC 사용 설명서의 [인터넷 게이트웨이 연결](#)을 참조하세요.

VPC 라우팅 테이블에는 기본적으로 VPC로 향하지 않는 트래픽을 인터넷 게이트웨이로 보내는 라우팅 규칙이 포함되어야 합니다. 이 구성에서 엔드포인트와의 연결이 프라이빗 IP 주소가 아닌 복제 인스턴스의 퍼블릭 IP 주소에서 오는 것으로 나타납니다. 자세한 내용은 Amazon VPC 사용 설명서의 [VPC 라우팅 테이블](#)을 참조하세요.

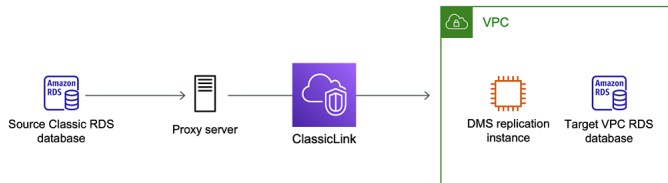
ClassicLink를 사용하여 VPC에 없는 RDS DB 인스턴스를 VPC에 있는 DB 인스턴스와 연결하여 구성

EC2-Classic은 2022년 8월 15일에 사용 중지될 예정입니다. EC2-Classic에서 VPC로 마이그레이션하는 것이 좋습니다. 자세한 내용은 Amazon EC2 사용 설명서의 [EC2-Classic에서 VPC로 마이그레이션](#) 및 [EC2-Classic 네트워킹 지원 중단에 대비하는 방법](#) 블로그 게시물을 참조하세요.

VPC에 없는 Amazon RDS DB 인스턴스를 VPC에 있는 DMS 복제 서버 및 DB 인스턴스에 연결하려면 ClassicLink를 프록시 서버와 함께 사용할 수 있습니다.

ClassicLink를 사용하면 EC2-Classic DB 인스턴스를 동일한 AWS 리전 내 계정의 VPC에 연결할 수 있습니다. 링크를 생성하고 난 후 원본 DB 인스턴스는 프라이빗 IP 주소를 사용하여 VPC 내부에 있는 복제 인스턴스와 통신할 수 있습니다.

VPC의 복제 인스턴스는 ClassicLink를 사용하여 EC2-Classic 플랫폼의 소스 DB 인스턴스에 직접 액세스할 수 없기 때문에 프록시 서버를 사용합니다. 프록시 서버는 원본 DB 인스턴스를 VPC에 연결하며, 여기에는 복제 인스턴스와 대상 DB 인스턴스가 포함됩니다. 프록시 서버는 ClassicLink를 VPC와 연결합니다. 프록시 서버의 포트 전달은 VPC에서 원본 DB 인스턴스와 대상 DB 인스턴스 통신을 허용합니다.



AWS 데이터베이스 마이그레이션 서비스와 함께 ClassicLink 사용

VPC에 없는 Amazon RDS DB 인스턴스를 VPC에 있는 AWS DMS 복제 서버 및 DB 인스턴스에 연결할 수 있습니다. 이를 위해 Amazon EC2 ClassicLink를 프록시 서버와 함께 사용할 수 있습니다.

다음 절차에서는 ClassicLink를 이러한 용도로 사용하는 방법을 보여줍니다. 이 절차에서는 VPC에 없는 Amazon RDS 소스 DB 인스턴스를 AWS DMS 복제 인스턴스와 대상 DB 인스턴스가 포함된 VPC에 연결합니다.

- VPC에서 AWS DMS 복제 인스턴스를 생성합니다. (모든 복제 인스턴스는 VPC에서 생성됩니다).
- VPC 보안 그룹을 복제 인스턴스와 대상 DB 인스턴스에 연결합니다. 두 인스턴스가 VPC 보안 그룹을 공유할 때에는 기본적으로 이 두 인스턴스가 서로 통신할 수 있습니다.
- EC2 Classic 인스턴스에서 프록시 서버를 설정합니다.
- ClassicLink를 사용하여 프록시 서버와 VPC를 연결합니다.
- 소스 및 대상 데이터베이스에 대한 AWS DMS 엔드포인트를 생성합니다.
- AWS DMS 작업을 생성합니다.

ClassicLink를 사용하여 VPC에 없는 DB 인스턴스의 데이터베이스를 VPC에 있는 DB 인스턴스의 데이터베이스로 마이그레이션하려면

1. AWS DMS 복제 인스턴스를 생성하고 VPC 보안 그룹을 할당합니다.
 - a. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

AWS Identity and Access Management (IAM) 사용자로 로그인한 경우에 액세스할 수 있는 적절한 권한이 있는지 확인합니다 AWS DMS. 데이터베이스 마이그레이션에 필요한 권한에 대한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

- b. 대시보드 페이지에서 복제 인스턴스를 선택합니다. [1단계: AWS DMS 콘솔을 사용하여 복제 인스턴스 생성](#)의 지침에 따라 복제 인스턴스를 생성합니다.
 - c. AWS DMS 복제 인스턴스를 생성한 후 EC2 서비스 콘솔을 엽니다. 탐색 창에서 네트워크 인스턴스를 선택합니다.
 - d. DMSNetworkInterface를 선택한 다음 작업 메뉴에서 보안 그룹 변경을 선택합니다.
 - e. 복제 인스턴스와 대상 DB 인스턴스에서 사용할 보안 그룹을 선택합니다.
2. 마지막 단계에서 선택한 보안 그룹을 대상 DB 인스턴스와 연결합니다.
- a. Amazon RDS 서비스 콘솔을 엽니다. 탐색 창에서 인스턴스를 선택합니다.
 - b. 대상 DB 인스턴스를 선택합니다. 인스턴스 작업에서 수정을 선택합니다.
 - c. 보안 그룹 파라미터에서 이전 단계에서 사용한 보안 그룹을 선택합니다.
 - d. 계속을 선택하고 DB 인스턴스 수정을 선택합니다.
3. 3단계: NGINX를 사용하여 EC2 Classic 인스턴스에서 프록시 서버를 설정합니다. 선택한 AMI를 사용하여 EC2 Classic 인스턴스를 시작합니다. 다음 예제는 AMI Ubuntu Server 14.04 LTS(HVM)를 기반으로 합니다.

EC2 Classic 인스턴스에서 프록시 서버를 설정하려면

- a. 다음 명령을 사용하여 EC2 Classic 인스턴스에 연결하고 NGINX를 설치합니다.

```
Prompt> sudo apt-get update
Prompt> sudo wget http://nginx.org/download/nginx-1.9.12.tar.gz
Prompt> sudo tar -xvzf nginx-1.9.12.tar.gz
Prompt> cd nginx-1.9.12
Prompt> sudo apt-get install build-essential
Prompt> sudo apt-get install libpcre3 libpcre3-dev
Prompt> sudo apt-get install zlib1g-dev
Prompt> sudo ./configure --with-stream
Prompt> sudo make
Prompt> sudo make install
```

- b. 다음 코드를 사용하여 NGINX 대몬 파일 `/etc/init/nginx.conf`를 편집합니다.

```
# /etc/init/nginx.conf - Upstart file

description "nginx http daemon"
author "email"

start on (filesystem and net-device-up IFACE=lo)
stop on runlevel [!2345]

env DAEMON=/usr/local/nginx/sbin/nginx
env PID=/usr/local/nginx/logs/nginx.pid

expect fork
respawn
respawn limit 10 5

pre-start script
    $DAEMON -t
    if [ $? -ne 0 ]
        then exit $?
    fi
end script

exec $DAEMON
```

- c. `/usr/local/nginx/conf/nginx.conf`에 NGINX 구성 파일을 만듭니다. 구성 파일에서 다음을 추가합니다.

```
# /usr/local/nginx/conf/nginx.conf - NGINX configuration file

worker_processes 1;

events {
    worker_connections 1024;
}

stream {
    server {
        listen DB instance port number;
        proxy_pass DB instance identifier:DB instance port number;
    }
}
```

```
}
}
```

- d. 명령줄에서 다음 명령을 사용하여 NGINX를 시작합니다.

```
Prompt> sudo initctl reload-configuration
Prompt> sudo initctl list | grep nginx
Prompt> sudo initctl start nginx
```

4. 대상 DB 인스턴스와 복제 인스턴스를 포함하는 대상 VPC와 프록시 서버를 ClassicLink로 연결합니다.
 - a. EC2 콘솔을 열고 프록시 서버를 실행하는 EC2 Classic 인스턴스를 선택합니다.
 - b. 작업에서 ClassicLink를 선택한 후 VPC에 연결을 선택합니다.
 - c. 이 절차에서 앞서 사용한 보안 그룹을 선택합니다.
 - d. VPC에 연결을 선택합니다.
5. 5단계:의 절차를 사용하여 AWS DMS 엔드포인트를 생성합니다 [2단계: 소스 및 대상 엔드포인트 지정](#). 소스 엔드포인트를 지정할 때에는 서버 이름으로 프록시의 내부 EC2 DNS 호스트 이름을 사용해야 합니다.
6. 의 절차를 사용하여 AWS DMS 작업을 생성합니다 [3단계: 태스크 생성 및 데이터 마이그레이션](#).

AWS 서비스에 연결하는 네트워크에 대한 구성

AWS 서비스에 연결하려면 인터넷 연결 또는 Virtual Private Cloud(VPC) 엔드포인트를 사용합니다. 이는 다음과 같은 경우에 적용됩니다.

소스 또는 대상 엔드포인트는 다음과 같은 AWS 서비스를 사용합니다.

- AWS Secrets Manager
- Amazon Simple Storage Service

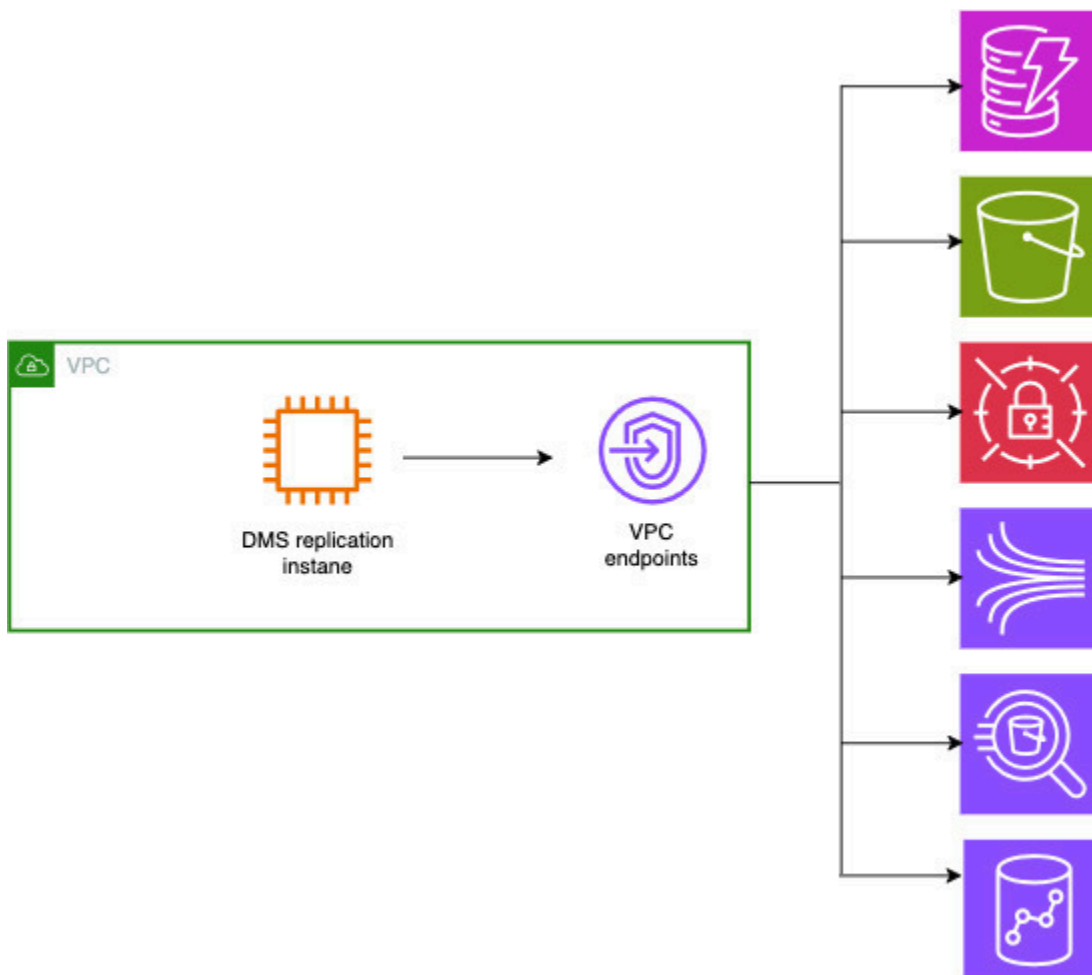
대상 엔드포인트는 다음과 같은 AWS 서비스입니다.

- Amazon S3
- Amazon Kinesis

- Amazon DynamoDB
- Amazon Redshift
- Amazon OpenSearch Service
- Amazon Athena

VPC 엔드포인트를 사용하여 AWS 서비스에 연결하는 네트워크에 대한 구성

VPC 엔드포인트는 AWS 리소스 간에 보안 연결을 제공하여 인터넷 액세스 없이 VPC 리소스를 AWS 서비스에 연결합니다. 프라이빗 서브넷의 애플리케이션은 AWS 네트워크 내에 있는 동안 AWS 서비스에 액세스하여 보안을 개선하고 지연 시간을 줄일 수 있습니다. 아래 이미지를 참조하세요.



자세한 내용은 [VPC 엔드포인트를 AWS DMS 소스 및 대상 엔드포인트로 구성](#) 및 [AWS DMS 보안 암호 관리자 VPC 엔드포인트 구성](#)을 참조하세요.

인터넷을 사용하여 AWS 서비스에 연결하는 네트워크에 대한 구성

복제 인스턴스는 데이터 마이그레이션 중에 AWS 리소스에 연결하려면 인터넷 액세스가 필요합니다.

VPC 내의 프라이빗 및 퍼블릭 서브넷에 대한 자세한 내용은 Amazon Virtual Private Cloud 사용 설명서의 [예제: 프라이빗 서브넷에 서버가 있는 VPC 및 NAT](#)를 참조하세요. 필요한 서비스와의 연결을 위해 네트워크 구성을 테스트해야 합니다.

복제 서브넷 그룹 생성

데이터베이스 마이그레이션에서 사용하는 네트워크의 일부로서 가상 프라이빗 클라우드(VPC)에서 사용할 서브넷을 지정해야 합니다. 이 VPC는 Amazon VPC 서비스를 기반으로 해야 합니다. 서브넷은 지정된 가용 영역에 있는 VPC의 IP 주소 범위입니다. 이러한 서브넷은 VPC가 위치한 AWS 리전의 가용 영역 간에 분산될 수 있습니다.

AWS DMS 콘솔에서 복제 인스턴스 또는 인스턴스 프로파일을 생성할 때 선택한 서브넷을 사용할 수 있습니다.

복제 서브넷 그룹을 생성하여 사용할 서브넷을 정의합니다. 가용 영역 2개 이상에서 서브넷을 지정해야 합니다.

복제 서브넷 그룹을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

IAM 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다. 데이터베이스 마이그레이션에 필요한 권한에 대한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

2. 탐색 창에서 서브넷 그룹을 선택합니다.
3. 서브넷 그룹 생성을 선택합니다.
4. 복제 서브넷 그룹 생성 페이지에서 복제 서브넷 그룹 정보를 지정합니다. 다음 표는 설정에 대한 설명입니다.

옵션	작업
이름	인쇄 가능한 ASCII 문자(/, " 및 @ 제외) 8-16자를 포함하는 복제 서브넷 그룹 이름을 입력합니다. 이름은 선택한 AWS 리전의 계정에 대해 고유해야 합니다. 와 같

옵션	작업
	이 수행 중인 AWS 리전 및 작업 포함과 같은 일부 인텔 리전스를 이름에 추가하도록 선택할 수 있습니다 DMS-default-VPC .
설명	복제 인스턴스 그룹에 대한 간략한 설명을 입력합니다.
VPC	데이터베이스 마이그레이션에 사용할 VPC를 선택합니다. VPC는 최소 2개의 가용 영역에 서브넷 1개 이상이 있어야 합니다.
서브넷 추가	복제 서브넷 그룹에 포함할 서브넷을 선택합니다. 두 개 이상의 가용 영역에서 서브넷을 선택해야 합니다.

5. 서브넷 그룹 생성을 선택합니다.

DNS를 사용하여 도메인 엔드포인트 확인

일반적으로 AWS DMS 복제 인스턴스는 Amazon EC2 인스턴스의 도메인 이름 시스템(DNS) 해석기를 사용하여 도메인 엔드포인트를 확인합니다. DNS 확인이 필요한 경우 Amazon Route 53 Resolver를 사용할 수 있습니다. Route 53 DNS Resolver 사용에 대한 자세한 내용은 [Route 53 Resolver 시작하기](#)를 참조하세요.

자체 온프레미스 이름 서버를 사용하여 Amazon Route 53 Resolver로 특정 엔드포인트를 확인하는 방법에 대한 자세한 내용은 [자체 온프레미스 이름 서버 사용](#) 섹션을 참조하세요.

복제 인스턴스의 암호화 키 설정

AWS DMS는 복제 인스턴스에서 사용하는 스토리지와 엔드포인트 연결 정보를 암호화합니다. 복제 인스턴스에서 사용하는 스토리지를 암호화하기 위해 AWS DMS AWS KMS key 는 AWS 계정에 고유한 것을 사용합니다. AWS Key Management Service ()를 사용하여이 KMS 키를 보고 관리할 수 있습니다 AWS KMS. 계정의 기본 KMS 키(aws/dms) 또는 사용자가 생성한 KMS 키를 사용할 수 있습니다. 기존 AWS KMS 암호화 키가 있는 경우 해당 키를 암호화에 사용할 수도 있습니다.

KMS 키 식별자를 제공하여 AWS DMS 리소스를 암호화하여 자체 암호화 키를 지정할 수 있습니다. 자체 암호화 키를 지정할 때 데이터베이스 마이그레이션을 수행하는 데 사용되는 사용자 계정에는 키에 대한 액세스 권한이 있어야 합니다. 자체 암호화 키를 생성하고 암호화 키에 사용자 액세스 권한을 부여하는 방법에 대한 자세한 내용은 [AWS KMS 개발자 안내서](#)를 참조하세요.

KMS 키 식별자를 지정하지 않으면 AWS DMS는 기본 암호화 키를 사용합니다. KMS는 계정 AWS의 AWS DMS에 대한 기본 암호화 키를 생성합니다. AWS 계정에는 AWS 리전마다 다른 기본 암호화 키가 있습니다.

AWS DMS 리소스를 암호화하는 데 사용되는 키를 관리하려면 AWS KMS를 사용합니다. 탐색 AWS KMS 창에서 KMS를 검색 AWS Management Console 하에서 찾을 수 있습니다.

AWS KMS는 안전하고 가용성이 높은 하드웨어와 소프트웨어를 결합하여 클라우드에 맞게 확장된 키 관리 시스템을 제공합니다. 이를 사용하면 암호화 키를 AWS KMS에서 생성하고 이러한 키를 사용할 수 있는 방법을 제어하는 정책을 정의할 수 있습니다. 이를 AWS KMS 지원 AWS CloudTrail하므로 키 사용을 감사하여 키가 적절하게 사용되고 있는지 확인할 수 있습니다. AWS KMS 키는 AWS DMS 및 기타 지원되는 AWS 서비스와 함께 사용할 수 있습니다. 지원되는 AWS 서비스로는 Amazon RDS, Amazon S3, Amazon Elastic Block Store(Amazon EBS), Amazon Redshift가 있습니다.

특정 암호화 키로 AWS DMS 리소스를 생성한 경우 해당 리소스의 암호화 키를 변경할 수 없습니다. AWS DMS 리소스를 생성하기 전에 암호화 키 요구 사항을 결정해야 합니다.

복제 인스턴스 생성

데이터베이스를 마이그레이션하는 첫 번째 작업은 복제 인스턴스를 생성하는 것입니다. 이 복제 인스턴스에는 소스 데이터베이스에서 대상 데이터베이스로 데이터를 할당하고 마이그레이션하는 작업을 수행하는 데 충분한 스토리지 및 처리 능력이 필요합니다. 이 인스턴스의 필요한 크기는 마이그레이션해야 하는 데이터 양과 인스턴스 실행에 필요한 작업에 따라 달라집니다. 복제 인스턴스에 대한 자세한 내용은 [AWS DMS 복제 인스턴스 작업](#) 섹션을 참조하세요.

AWS 콘솔을 사용하여 복제 인스턴스를 생성하려면

1. AWS DMS 콘솔의 탐색 창에서 복제 인스턴스를 선택한 다음 복제 인스턴스 생성을 선택합니다.
2. 복제 인스턴스 생성 페이지에서 복제 인스턴스 정보를 지정합니다. 다음 표는 사용자가 지정할 수 있는 설정에 대한 설명입니다.

옵션	작업
이름	인쇄 가능한 ASCII 문자(/, " 및 @ 제외) 8-16자를 포함하는 복제 인스턴스의 이름을 입력합니다. 이름은 선택한 AWS 리전의 계정에서 고유해야 합니다. 와 같이 수행 중인 AWS 리전 및 작업 포함과 같은 일부 인텔리전

옵션	작업
	스를 이름에 추가하도록 선택할 수 있습니다 west2-mysql2mysql-instance1 .
설명이 포함된 Amazon 리소스 이름 (ARN) - 선택 사항	기본 DMS ARN을 재정의하기 위한 친숙한 이름입니다. 생성 후에는 수정할 수 없습니다.
설명	복제 인스턴스에 대한 간략한 설명을 입력합니다.
인스턴스 클래스	마이그레이션에 필요한 구성과 함께 인스턴스 클래스를 선택합니다. 마이그레이션을 성공적으로 완료할 수 있도록 인스턴스의 스토리지, 네트워크 및 처리 능력은 충분해야 합니다. 마이그레이션에 최적인 인스턴스 클래스를 결정하는 방법에 대한 자세한 내용은 AWS DMS 복제 인스턴스 작업 섹션을 참조하세요.
엔진 버전	AWS DMS 콘솔에서 원하는 지원되는 엔진 버전을 선택할 수 있습니다. 에서 다른 엔진 버전을 지정하지 않는 한 AWS CLI복제 인스턴스는 복제 엔진의 AWS DMS 최신 비베타 버전을 실행합니다 AWS CLI.
고가용성	이 선택적 파라미터를 사용하여 다른 가용 영역에 복제 인스턴스의 대기 복제본을 생성하여 장애 조치를 지원합니다. 변경 데이터 캡처(CDC) 또는 지속적 복제를 사용하려면, 이 옵션을 활성화해야 합니다.

옵션	작업
할당된 스토리지(GiB)	스토리지는 주로 로그 파일과 캐시된 트랜잭션에서 소모합니다. 캐시 트랜잭션에서 스토리지는 캐시된 트랜잭션을 디스크에 기록해야 할 때에만 사용됩니다. 따라서 AWS DMS는 상당한 양의 스토리지를 사용하지 않습니다. 일부 예외 사항은 다음과 같습니다. • 상당한 트랜잭션 로드를 발생하는 매우 큰 테이블. 큰 테이블을 로드하면 다소 시간이 걸릴 수 있으므로, 캐시된 트랜잭션은 큰 테이블 로드 중에 디스크에 기록될 가능성이 더 높습니다. • 캐시된 트랜잭션을 로드하기 전에 일시 중지되도록 구성되는 작업입니다. 이 경우에 모든 테이블에서 전체 로드가 완료될 때까지 모든 트랜잭션이 캐시됩니다. 이 구성에서 상당량의 스토리지가 캐시된 트랜잭션에서 소모될 수 있습니다. • Amazon Redshift로 로드되고 있는 테이블을 사용하여 구성된 작업입니다. 그렇지만, 이 구성은 Amazon Aurora가 대상일 때는 문제가 아닙니다. 대다수의 경우에 스토리지의 기본 할당량은 충분합니다. 하지만 항상 스토리지 관련 지표에 주의를 기울이는 것이 좋습니다. 기본 할당량보다 많이 소비하는 경우 스토리지를 스케일 업해야 합니다.
네트워크 유형	DMS는 IPv4 주소 지정 프로토콜 네트워크 유형을 지원하고 이중 스택 모드에서 IPv4 및 IPv6 주소 지정 프로토콜 네트워크 유형을 모두 지원합니다. IPv6 주소 지정 프로토콜 네트워크 유형을 사용하여 복제 인스턴스와 통신해야 하는 리소스가 있는 경우 이중 스택 모드를 사용합니다. 이중 스택 모드의 제한 사항에 대한 자세한 내용은 Amazon Relational Database Service 사용 설명서의 이중 스택 네트워크 DB 인스턴스 제한 사항을 참조하세요.

옵션	작업
VPC	사용할 VPC를 선택합니다. 원본 또는 대상 데이터베이스가 VPC에 있는 경우, 해당 VPC를 선택합니다. 소스 데이터베이스와 대상 데이터베이스가 다른 VPC에 있으면, 모두 퍼블릭 서브넷에 있고 공개적으로 액세스 가능해야 합니다. 그런 다음 복제 인스턴스를 배치할 VPC를 선택합니다. 복제 인스턴스는 원본 VPC에 있는 데이터에 액세스할 수 있어야 합니다. 소스 또는 대상 데이터베이스가 VPC에 없으면, 복제 인스턴스가 배치될 VPC를 선택합니다.
복제 서브넷 그룹	복제 인스턴스를 생성할 위치에 있는 선택된 VPC에서 복제 서브넷 그룹을 선택합니다. 원본 데이터베이스가 VPC에 있는 경우, 원본 데이터베이스가 포함된 서브넷 그룹을 복제 인스턴스 위치로 선택합니다. 복제 서브넷에 대한 자세한 내용은 복제 서브넷 그룹 생성 섹션을 참조하세요.
공개적으로 액세스할 수 있음	인터넷에서 복제 인스턴스에 액세스하려면 이 옵션을 선택합니다. 기본값은 공개적으로 액세스할 수 있으며, 옵션을 선택한 후에는 생성한 복제 인스턴스를 수정할 수 없습니다.

3. 고급 탭을 선택하고 네트워크와 암호화 설정이 필요하면 값을 설정합니다. 다음 표는 설정에 대한 설명입니다.

옵션	작업
가용 영역	원본 데이터베이스가 있는 가용 영역을 선택합니다.
VPC 보안 그룹	모든 복제 인스턴스가 VPC에 생성됩니다. 소스 데이터베이스가 VPC에 있는 경우, 데이터베이스가 상주하는 곳에 있는 DB 인스턴스에 대한 액세스 권한을 부여하는 VPC 보안 그룹을 선택합니다.
KMS 키	복제 스토리지와 연결 정보를 암호화하기 위해 사용할 암호화 키를 선택합니다. (기본값) aws/dms를 선택

옵션	작업
	택하면 계정 및 AWS 리전과 연결된 기본 AWS Key Management Service (AWS KMS) 키가 사용됩니다. 설명 및 계정 번호가 키의 ARN과 함께 표시됩니다. 암호화 키 사용에 대한 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정를 참조하세요.

4. 유지 관리 설정을 지정합니다. 다음 표는 설정에 대한 설명입니다. 유지 관리 설정에 대한 자세한 내용은 [AWS DMS 유지 관리 기간 작업](#) 섹션을 참조하세요.

옵션	작업
자동 버전 업그레이드	AWS DMS 는 메이저 버전과 마이너 버전을 구분하지 않습니다. 예를 들어 버전 3.4.x에서 3.5.x로 업그레이드 하는 것은 메이저 업그레이드로 간주되지 않으며, 따라서 모든 변경 사항은 이전 버전과 호환됩니다. 자동 버전 업그레이드가 활성화된 경우 DMS는 유지 관리 기간 동안 더 이상 사용되지 않는 복제 인스턴스 버전을 자동으로 업그레이드합니다. AutoMinorVersionUpgrade가 활성화된 경우 DMS는 복제 인스턴스를 생성할 때 현재 기본 엔진 버전을 사용합니다. 예를 들어 엔진 버전을 현재 기본 버전보다 낮은 버전 번호로 설정하면 DMS는 기본 버전을 사용합니다. AutoMinorVersionUpgrade가 활성화되지 않은 상태에서 복제 인스턴스를 생성하면 DMS는 엔진 버전 파라미터로 지정된 엔진 버전을 사용합니다.
유지 관리 윈도우	시스템 유지 관리를 실행할 수 있는 주 단위 기간(UTC, 협정 세계시)을 선택합니다. 기본값: AWS 리전당 8시간 블록 시간 중에서 임의로 선택되는 30분 기간으로, 요일에 무작위로 발생합니다.

5. 복제 인스턴스 생성을 선택합니다.

복제 인스턴스 수정

복제 인스턴스에 대한 설정을 수정하여 예컨대 인스턴스 클래스를 변경하거나 스토리지를 늘릴 수 있습니다.

복제 인스턴스를 수정하는 경우 변경 사항을 즉시 적용할 수 있습니다. 변경 사항을 즉시 적용하려면 AWS Management Console에서 변경 사항 즉시 적용 옵션을 선택합니다. 또는를 호출할 때 --apply-immediately 파라미터를 사용하거나 DMS API를 사용할 때 true 때 ApplyImmediately 파라미터를 로 AWS CLI 설정합니다.

변경 사항을 즉시 적용하지 않기로 선택하면 변경 사항이 보류 중 수정 사항 대기열로 보내집니다. 다음 유지 관리 기간에 대기열에 있는 보류 중 변경 사항이 적용됩니다.

Note

변경 사항을 즉시 적용하기로 선택하면 보류 중 수정 사항 대기열에 있는 변경 사항까지 모두 적용됩니다. 보류 중 수정 사항이 하나라도 가동 중지 시간을 필요로 하는 경우 변경 사항 즉시 적용을 선택하면 예기치 못한 가동 중지 시간이 발생할 수 있습니다.

AWS 콘솔을 사용하여 복제 인스턴스를 수정하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 복제 인스턴스를 선택합니다.
3. 수정할 복제 인스턴스를 선택합니다. 다음 표에서는 수행할 수 있는 수정 사항을 설명합니다.

옵션	작업
이름	복제 인스턴스의 이름을 변경할 수 있습니다. 인쇄 가능한 ASCII 문자(/, " 및 @ 제외) 8-16자를 포함하는 복제 인스턴스의 이름을 입력합니다. 이름은 선택한 AWS 리전의 계정에서 고유해야 합니다. 와 같이 수행 중인 AWS 리전 및 작업 포함과 같은 일부 인텔리전스를 이름에 추가하도록 선택할 수 있습니다 west2-mysql1mysql-instance1 .

옵션	작업
설명	복제 인스턴스에 대한 간략한 설명을 수정하거나 입력합니다.
인스턴스 클래스	인스턴스 클래스를 변경할 수 있습니다. 마이그레이션에 필요한 구성과 함께 인스턴스 클래스를 선택합니다. 인스턴스 클래스를 변경하면 복제 인스턴스가 재부팅됩니다. 이 재부팅은 다음 유지 관리 기간 중에 발생하지만 변경 사항 즉시 적용 옵션을 선택하는 경우 즉시 발생할 수 있습니다. 마이그레이션에 최적인 인스턴스 클래스를 결정하는 방법에 대한 자세한 내용은 AWS DMS 복제 인스턴스 작업 섹션을 참조하세요.
엔진 버전	복제 인스턴스에 사용되는 엔진 버전을 업그레이드할 수 있습니다. 복제 엔진 버전을 업그레이드하면 업그레이드 중에 복제 인스턴스가 종료됩니다.
다중 AZ	이 옵션을 변경하여 다른 가용 영역에 장애 조치를 지원할 복제 인스턴스의 예비 복제본을 생성하거나 이 옵션을 제거합니다. 데이터 캡처(CDC) 또는 지속적 복제를 사용하려면, 이 옵션을 활성화해야 합니다.

옵션	작업
할당된 스토리지(GiB)	스토리지는 주로 로그 파일과 캐시된 트랜잭션에서 소모합니다. 캐시 트랜잭션에서 스토리지는 캐시된 트랜잭션을 디스크에 기록해야 할 때에만 사용됩니다. 따라서 AWS DMS는 상당한 양의 스토리지를 사용하지 않습니다. 일부 예외 사항은 다음과 같습니다. • 상당한 트랜잭션 로드를 발생하는 매우 큰 테이블. 큰 테이블을 로드하면 다소 시간이 걸릴 수 있으므로, 캐시된 트랜잭션은 큰 테이블 로드 중에 디스크에 기록될 가능성이 더 높습니다. • 캐시된 트랜잭션을 로드하기 전에 일시 중지되도록 구성되는 작업입니다. 이 경우에 모든 테이블에서 전체 로드가 완료될 때까지 모든 트랜잭션이 캐시됩니다. 이 구성에서 상당량의 스토리지가 캐시된 트랜잭션에서 소모될 수 있습니다. • Amazon Redshift로 로드되고 있는 테이블을 사용하여 구성된 작업입니다. 그렇지만, 이 구성은 Amazon Aurora가 대상일 때는 문제가 아닙니다. 대다수의 경우에 스토리지의 기본 할당량은 충분합니다. 그렇지만, 항상 스토리지 관련 지표에 유의하고 기본 할당량보다 더 많이 소모하게 되면 스토리지를 확장하는 것이 좋습니다.
네트워크 유형	DMS는 IPv4 주소 지정 프로토콜 네트워크 유형을 지원하고 이중 스택 모드에서 IPv4 및 IPv6 주소 지정 프로토콜 네트워크 유형을 모두 지원합니다. IPv6 주소 지정 프로토콜 네트워크 유형을 사용하여 복제 인스턴스와 통신해야 하는 리소스가 있는 경우 이중 스택 모드를 선택합니다. 이중 스택 모드의 제한 사항에 대한 자세한 내용은 Amazon Relational Database Service 사용 설명서의 이중 스택 네트워크 DB 인스턴스 제한 사항을 참조하세요.

옵션	작업
VPC 보안 그룹	모든 복제 인스턴스가 VPC에 생성됩니다. 소스 데이터베이스가 VPC에 있는 경우, 데이터베이스가 상주하는 곳에 있는 DB 인스턴스에 대한 액세스 권한을 부여하는 VPC 보안 그룹을 선택합니다.
자동 버전 업그레이드	AWS DMS 는 메이저 버전과 마이너 버전을 구분하지 않습니다. 예를 들어 버전 3.4.x에서 3.5.x로 업그레이드 하는 것은 메이저 업그레이드로 간주되지 않으며, 따라서 모든 변경 사항은 이전 버전과 호환됩니다. 자동 버전 업그레이드가 활성화된 경우 DMS는 유지 관리 기간 동안 더 이상 사용되지 않는 복제 인스턴스 버전을 자동으로 업그레이드합니다. 자동 버전 업그레이드가 활성화된 경우 DMS는 복제 인스턴스를 생성할 때 현재 기본 엔진 버전을 사용합니다. 예를 들어 엔진 버전을 현재 기본 버전보다 낮은 버전 번호로 설정하면 DMS는 기본 버전을 사용합니다. 자동 버전 업그레이드가 활성화되지 않은 상태에서 복제 인스턴스를 생성하면 DMS는 엔진 버전 파라미터로 지정된 엔진 버전을 사용합니다.
유지 관리 윈도우	시스템 유지 관리를 실행할 수 있는 주 단위 기간(UTC, 협정 세계시)을 선택합니다. 기본값: AWS 리전당 8시간 블록 시간 중에서 임의로 선택되는 30분 기간으로, 요일에 무작위로 발생합니다.
변경 내용 즉시 적용	변경 내용을 즉시 적용하려면 이 옵션을 선택합니다. 선택하는 설정에 따라 이 옵션을 선택하면 복제 인스턴스가 즉시 재부팅될 수 있습니다. AWS DMS 가 변경 사항을 적용하는 동안 연결 테스트를 선택하면 오류 메시지가 표시됩니다. 가 복제 인스턴스에 변경 사항을 AWS DMS 적용한 후 연결 테스트를 다시 선택합니다.

옵션	작업
다음 유지 관리 기간에 변경 사항 적용	DMS가 다음 예정된 유지 관리 기간까지 기다려 변경 사항을 적용하도록 하려면 이 옵션을 선택합니다.

복제 인스턴스 수정 모범 사례

복제 인스턴스를 수정할 때 다음 모범 사례를 따르면 마이그레이션 워크플로에 미치는 영향을 최소화 하면서 성공적인 업데이트를 보장하는 데 도움이 됩니다. 수정 전, 중, 후에 다음 주요 단계를 수행하여 프로세스 전반에 걸쳐 데이터 무결성과 운영 효율성을 유지합니다.

계획 수정 타이밍:

- 변경 사항을 즉시 적용하거나 다음 유지 관리 기간에 예약할 수 있습니다.
- 영향을 최소화하기 위해 트래픽이 적은 기간 동안 일정을 잡습니다.

사전 수정 단계:

- 모든 활성 복제 작업을 중지합니다.
- 모든 작업이 성공적으로 중지되었는지 확인합니다.
- 현재 구성 작업 설정을 문서화합니다.

수정 중:

- 수정 진행 상황 모니터링.
- 계속하기 전에 "사용 가능" 상태를 기다립니다.

수정 후 단계:

- 이전에 중지된 모든 작업을 재개합니다.
- 작업이 올바르게 실행되고 있는지 확인합니다.

복제 인스턴스 재부팅

AWS DMS 복제 인스턴스를 재부팅하여 복제 엔진을 다시 시작할 수 있습니다. 재부팅하면 인스턴스 상태가 재부팅 중으로 설정되면서 복제 인스턴스가 잠시 중단됩니다. AWS DMS 인스턴스가 다중 AZ에 대해 구성된 경우 장애 조치를 통해 재부팅을 수행할 수 있습니다. 재부팅이 완료되면 AWS DMS 이벤트가 생성됩니다.

AWS DMS 인스턴스가 다중 AZ 배포인 경우 재부팅할 때 한 AWS 가용 영역에서 다른 가용 영역으로 계획된 장애 조치를 강제로 수행할 수 있습니다. AWS DMS 인스턴스의 계획된 장애 조치를 강제로 적

용하면는 다른 가용 영역의 대기 인스턴스로 자동 전환하기 전에 현재 인스턴스의 활성 연결을 AWS DMS 종료합니다. 계획된 장애 조치로 재부팅하면 복제 AWS DMS 인스턴스 클래스를 조정할 때와 같이 인스턴스의 계획된 장애 조치 이벤트를 시뮬레이션할 수 있습니다.

Note

재부팅할 때 한 가용 영역에서 다른 가용 영역으로 장애 조치를 강제로 실시하면 가용 영역 변경 사항이 몇 분 동안 반영되지 않을 수 있습니다. 이 지연은 AWS Management Console 및 AWS DMS API에 대한 호출 AWS CLI 및에 표시됩니다.

복제 인스턴스에서 마이그레이션 태스크가 실행 중일 때 재부팅이 발생하면 데이터 손실은 발생하지 않지만 태스크가 중지되고 태스크 상태는 오류 상태로 바뀝니다.

마이그레이션 태스크의 테이블이 대량 로드 중에 있고(전체 로드 단계) 아직 시작되지 않은 경우 오류 상태로 전환됩니다. 하지만 그 시점에 완료된 테이블은 완료 상태로 남아 있습니다. 전체 로드 단계에서 재부팅이 발생하는 경우 아래 단계 중 하나를 수행하는 것이 좋습니다.

- 태스크에서 완료 상태인 테이블을 제거하고 나머지 테이블을 사용하여 태스크를 다시 시작합니다.
- 오류 상태의 테이블과 보류 중인 테이블로 새 태스크를 생성합니다.

마이그레이션 작업의 테이블이 지속적인 복제 단계에 있는 경우 재부팅이 완료된 후 작업이 다시 시작됩니다.

상태가 사용 가능 상태가 아니면 AWS DMS 복제 인스턴스를 재부팅할 수 없습니다. 이전에 요청한 수정 또는 유지 관리 기간 작업과 같은 여러 가지 이유로 AWS DMS 인스턴스를 사용할 수 없습니다. AWS DMS 복제 인스턴스를 재부팅하는 데 필요한 시간은 일반적으로 짧습니다(5분 미만).

AWS 콘솔을 사용하여 복제 인스턴스 재부팅

복제 인스턴스를 재부팅하려면 AWS 콘솔을 사용합니다.

AWS 콘솔을 사용하여 복제 인스턴스를 재부팅하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 복제 인스턴스를 선택합니다.
3. 재부팅할 복제 인스턴스를 선택합니다.

4. 재부팅을 선택합니다. 복제 인스턴스 재부팅 대화 상자가 열립니다.
5. 다중 AZ 배포에 대해 복제 인스턴스를 구성한 경우 다른 AWS 가용 영역으로 장애 조치하려면 계획된 장애 조치로 재부팅하시겠습니까? 확인란을 선택합니다.
6. 재부팅을 선택합니다.

CLI를 사용하여 복제 인스턴스 재부팅

복제 인스턴스를 재부팅하려면 다음 파라미터와 함께 명령을 사용합니다 AWS CLI [reboot-replication-instance](#).

- `--replication-instance-arn`

Example 간단한 재부팅 예제

다음 AWS CLI 예시에서는 복제 인스턴스를 재부팅합니다.

```
aws dms reboot-replication-instance \  
--replication-instance-arn arn of my rep instance
```

Example 장애 조치로 간단한 재부팅 예제

다음 AWS CLI 예시에서는 장애 조치로 복제 인스턴스를 재부팅합니다.

```
aws dms reboot-replication-instance \  
--replication-instance-arn arn of my rep instance \  
--force-planned-failover
```

API를 사용하여 복제 인스턴스 재부팅

복제 인스턴스를 재부팅하려면 다음 파라미터와 함께 AWS DMS API [RebootReplicationInstance](#) 작업을 사용합니다.

- `ReplicationInstanceArn` = *arn of my rep instance*

Example 간단한 재부팅 예제

다음 코드 예제에서는 복제 인스턴스를 재부팅합니다.

```
https://dms.us-west-2.amazonaws.com/
?Action=RebootReplicationInstance
&DBInstanceArn=arn of my rep instance
&SignatureMethod=HmacSHA256
&SignatureVersion=4
&Version=2014-09-01
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIADQKE4SARGYLE/20140425/us-east-1/dms/aws4_request
&X-Amz-Date=20140425T192732Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
&X-Amz-Signature=1dc9dd716f4855e9bdf188c70f1cf9f6251b070b68b81103b59ec70c3e7854b3
```

Example 장애 조치로 간단한 재부팅 예제

다음 코드 예제에서는 복제 인스턴스를 재부팅하고 다른 AWS 가용 영역으로 장애 조치합니다.

```
https://dms.us-west-2.amazonaws.com/
?Action=RebootReplicationInstance
&DBInstanceArn=arn of my rep instance
&ForcePlannedFailover=true
&SignatureMethod=HmacSHA256
&SignatureVersion=4
&Version=2014-09-01
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIADQKE4SARGYLE/20140425/us-east-1/dms/aws4_request
&X-Amz-Date=20140425T192732Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
&X-Amz-Signature=1dc9dd716f4855e9bdf188c70f1cf9f6251b070b68b81103b59ec70c3e7854b3
```

복제 인스턴스 삭제

사용을 마치면 AWS DMS 복제 인스턴스를 삭제할 수 있습니다. 복제 인스턴스를 사용하는 마이그레이션 작업이 있는 경우 복제 인스턴스를 삭제하기 전에 작업을 중지한 후 삭제해야 합니다.

AWS 계정을 해지하면 2일 후에 계정과 연결된 모든 AWS DMS 리소스 및 구성이 삭제됩니다. 이 리소스에는 모든 복제 인스턴스, 원본 및 대상 엔드포인트 구성, 복제 작업, SSL 인증서가 포함됩니다. 2일 후에 AWS DMS 다시 사용하기로 결정한 경우 필요한 리소스를 다시 생성합니다.

복제 인스턴스가 모든 삭제 기준을 충족하고 장기간 DELETING 상태를 유지하는 경우 지원 팀에 문의하여 문제를 해결하세요.

AWS 콘솔을 사용하여 복제 인스턴스 삭제

복제 인스턴스를 삭제하려면 AWS 콘솔을 사용합니다.

AWS 콘솔을 사용하여 복제 인스턴스를 삭제하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 복제 인스턴스를 선택합니다.
3. 삭제할 복제 인스턴스를 선택합니다.
4. 삭제를 선택합니다.
5. 대화 상자에서 삭제를 선택합니다.

CLI를 사용하여 복제 인스턴스 삭제

복제 인스턴스를 삭제하려면 다음 파라미터와 AWS CLI [delete-replication-instance](#) 함께 명령을 사용합니다.

- `--replication-instance-arn`

Example 삭제 예제

다음 AWS CLI 예시에서는 복제 인스턴스를 삭제합니다.

```
aws dms delete-replication-instance \  
--replication-instance-arn arn of my rep instance
```

API를 사용하여 복제 인스턴스 삭제

복제 인스턴스를 삭제하려면 API AWS DMS [DeleteReplicationInstance](#) 작업을 다음 파라미터와 함께 사용합니다.

- `ReplicationInstanceArn` = *arn of my rep instance*

Example 삭제 예제

다음 코드 예시에서는 복제 인스턴스를 삭제합니다.

```
https://dms.us-west-2.amazonaws.com/
?Action=DeleteReplicationInstance
&DBInstanceArn=arn of my rep instance
&SignatureMethod=HmacSHA256
&SignatureVersion=4
&Version=2014-09-01
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIADQKE4SARGYLE/20140425/us-east-1/dms/aws4_request
&X-Amz-Date=20140425T192732Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
&X-Amz-Signature=1dc9dd716f4855e9bdf188c70f1cf9f6251b070b68b81103b59ec70c3e7854b3
```

AWS DMS 유지 관리 기간 작업

모든 AWS DMS 복제 인스턴스에는 사용 가능한 시스템 변경 사항이 적용되는 주간 유지 관리 기간이 있습니다. 유지 관리 기간은 수정 사항 및 소프트웨어 패치 적용 시점을 조절할 수 있는 기회로 생각하면 좋습니다.

에서 지정된 주 동안 유지 관리가 필요하다고 AWS DMS 판단하면 복제 인스턴스를 생성할 때 선택한 30분 유지 관리 기간 동안 유지 관리가 수행됩니다. 30분 유지 관리 기간 동안 대부분의 유지 관리를 AWS DMS 완료합니다. 단, 대규모 변경 사항의 경우 보다 긴 시간이 필요할 수 있습니다.

유지 관리가 기존 마이그레이션 작업에 미치는 영향

인스턴스에서 AWS DMS 마이그레이션 작업이 실행 중이면 패치가 적용될 때 다음 이벤트가 발생합니다.

- 마이그레이션 작업의 테이블이 진행 중 변경 사항 복제 단계(CDC)에 있는 경우 AWS DMS 는 태스크를 잠시 동안 중지했다가 패치가 적용된 후에 다시 시작합니다. 그리고 나면 패치가 적용될 때 중단되었던 지점부터 마이그레이션이 계속됩니다.
- AWS DMS 가 기존 데이터 마이그레이션 또는 기존 데이터 마이그레이션의 일부로 테이블을 마이그레이션하고 진행 중인 변경 작업을 복제하는 경우 DMS는 패치가 적용되는 동안 전체 로드 단계에 있는 모든 테이블에 대해 마이그레이션을 중지했다가 다시 시작합니다. 또한 DMS는 패치가 적용되는 동안 CDC 단계에 있는 모든 테이블을 중지했다가 다시 시작합니다.

유지 관리 기간 설정 변경

AWS Management Console AWS CLI, 또는 AWS DMS API를 사용하여 유지 관리 기간 기간을 변경할 수 있습니다.

콘솔을 사용하여 유지 관리 기간 설정 변경

AWS Management Console을 사용하여 유지 관리 기간 프레임을 변경할 수 있습니다.

콘솔을 사용하여 기본 유지 관리 기간을 변경하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 복제 인스턴스를 선택합니다.
3. 수정을 수정 및 선택할 복제 인스턴스를 선택합니다.
4. 유지 관리 섹션을 확장하고 유지 관리 기간의 날짜 및 시간을 선택합니다.
5. 변경 내용 즉시 적용을 선택합니다.
6. 수정을 선택합니다.

CLI를 사용하여 유지 관리 기간 설정 변경

원하는 유지 관리 기간을 조정하려면 다음 파라미터와 함께 명령을 사용합니다 AWS CLI [modify-replication-instance](#).

- `--replication-instance-identifier`
- `--preferred-maintenance-window`

Example

다음 AWS CLI 예제에서는 유지 관리 기간을 화요일 오전 4시~오전 4시 30분으로 설정합니다. UTC 기준입니다.

```
aws dms modify-replication-instance \  
--replication-instance-identifier myrepinstance \  
--preferred-maintenance-window Tue:04:00-Tue:04:30
```

API를 사용하여 유지 관리 기간 설정 변경

기본 유지 관리 기간을 조정하려면 다음 파라미터와 함께 AWS DMS API [ModifyReplicationInstance](#) 작업을 사용합니다.

- `ReplicationInstanceIdentifier` = *myrepinstance*
- `PreferredMaintenanceWindow` = *Tue:04:00-Tue:04:30*

Example

다음은 유지 관리 기간을 화요일 오전 4:00~4:30으로 설정하는 코드 예제입니다. UTC 기준입니다.

```
https://dms.us-west-2.amazonaws.com/  
?Action=ModifyReplicationInstance  
&DBInstanceIdentifier=myrepinstance  
&PreferredMaintenanceWindow=Tue:04:00-Tue:04:30  
&SignatureMethod=HmacSHA256  
&SignatureVersion=4  
&Version=2014-09-01  
&X-Amz-Algorithm=AWS4-HMAC-SHA256  
&X-Amz-Credential=AKIADQKE4SARGYLE/20140425/us-east-1/dms/aws4_request  
&X-Amz-Date=20140425T192732Z  
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date  
&X-Amz-Signature=1dc9dd716f4855e9bdf188c70f1cf9f6251b070b68b81103b59ec70c3e7854b3
```

AWS DMS 엔드포인트 작업

엔드포인트는 데이터 스토어에 대한 연결, 데이터 스토어 유형 및 위치 정보를 제공합니다. AWS 데이터베이스 마이그레이션 서비스는 이 정보를 사용하여 데이터 스토어에 연결하고 소스 엔드포인트에서 대상 엔드포인트로 데이터를 마이그레이션합니다. 엔드포인트 설정을 사용하여 엔드포인트에 대한 추가 연결 속성을 지정할 수 있습니다. 이러한 설정은 로깅, 파일 크기 및 기타 파라미터를 제어할 수 있습니다. 엔드포인트 설정에 대한 자세한 내용은 해당 데이터 저장소의 설명서 단원을 참조하세요.

아래에서 엔드포인트에 대한 자세한 정보를 확인할 수 있습니다.

주제

- [소스 및 대상 엔드포인트 생성](#)
- [데이터 마이그레이션용 소스](#)
- [마이그레이션에 적합한 대상](#)
- [에 대한 VPC 엔드포인트 구성 AWS DMS](#)
- [에서 지원하는 DDL 문 AWS DMS](#)
- [고급 엔드포인트 구성](#)

소스 및 대상 엔드포인트 생성

복제 인스턴스를 생성할 때 원본 및 대상 엔드포인트를 생성할 수 있습니다. 또는 복제 인스턴스가 생성된 후에 엔드포인트를 생성할 수 있습니다. 원본과 대상 데이터 저장소는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, Amazon Relational Database Service(Amazon RDS) DB 인스턴스 또는 온프레미스 데이터베이스에 있을 수 있습니다. (엔드포인트 중 하나가 AWS 서비스에 있어야 합니다. AWS DMS를 사용하여 온프레미스 데이터베이스에서 다른 온프레미스 데이터베이스로 마이그레이션할 수 없습니다.)

다음 절차에서는 AWS DMS 콘솔 마법사를 선택했다고 가정합니다. 참고로 AWS DMS 콘솔의 탐색 창에서 엔드포인트를 선택한 다음 엔드포인트 생성을 선택하여 이 단계를 수행할 수도 있습니다. 콘솔 마법사를 사용할 때에는 동일한 페이지에서 원본과 대상 엔드포인트를 모두 생성합니다. 콘솔 마법사를 사용하지 않을 때에는 각 엔드포인트를 별도로 생성합니다.

AWS 콘솔을 사용하여 소스 또는 대상 데이터베이스 엔드포인트를 지정하려면

1. 소스 및 타겟 데이터베이스 엔드포인트 연결 페이지에서 원본 또는 대상 데이터베이스에 대한 연결 정보를 지정합니다. 다음 표는 설정에 대한 설명입니다.

구성 옵션	구성 설정
[엔드포인트 유형]	이 엔드포인트가 원본 엔드포인트인지, 아니면 대상 엔드포인트인지 선택합니다.
RDS DB 인스턴스 선택	엔드포인트가 Amazon RDS 데이터베이스 인스턴스인 경우 이 옵션을 선택합니다.
[엔드포인트 식별자]	엔드포인트를 식별하는 데 사용할 이름을 입력합니다. 이름에 oracle-source 또는 PostgreSQL-target 과 같은 엔드포인트 유형을 포함해야 합니다. 모든 복제 인스턴스에서 이 이름은 고유해야 합니다.
설명이 포함된 Amazon 리소스 이름 (ARN) - 선택 사항	기본 DMS ARN을 재정의하기 위해 이름을 제공합니다. 이 설정은 선택 사항입니다.
소스 엔진 및 타겟 엔진	엔드포인트가 되는 데이터베이스 엔진의 유형을 선택합니다.
엔드포인트 데이터베이스에 대한 액세스	엔드포인트 데이터베이스 자격 증명 정보 지정에 사용할 옵션을 선택합니다. • AWS Secrets Manager선택 - 다음과 같이 AWS Secrets Manager에 정의된 비밀을 사용하여 자격 증명 정보를 비밀리에 제공합니다. 이러한 보안 암호 생성 및에서 해당 보안 암호에 액세스할 수 있는 보안 액세스 역할에 AWS DMS 대한 자세한 내용은 섹션을 참조하세요 보안 암호를 사용하여 AWS Database Migration Service 엔드포인트 액세스. • 수동으로 액세스 정보 제공 - 다음과 같이 직접 입력하는 일반 텍스트 자격 증명 정보를 사용합니다. • IAM 인증 - Amazon RDS 데이터베이스 인스턴스의 사용자 이름 및 암호 대신 인증 유형으로 IAM을 사용합니다.
AWS Secrets Manager선택	다음 비밀 자격 증명 정보를 설정합니다.

구성 옵션	구성 설정
비밀 ID	AWS Secrets Manager 에서 엔드포인트 데이터베이스 액세스를 위해 생성한 비밀의 전체 Amazon 리소스 이름(ARN), 부분 ARN 또는 기억하기 쉬운 이름을 입력합니다.
IAM 역할	사용자를 대신하여 보안 암호 ID로 식별되는 보안 암호에 대한 액세스를 제공하기 위해 IAM에서 생성한 보안 암호 AWS DMS 액세스 역할의 ARN을 입력합니다. 비밀 액세스 역할 생성에 대한 자세한 내용은 보안 암호를 사용하여 AWS Database Migration Service 엔드포인트 액세스 단원을 참조하세요.
Oracle Automatic Storage Management(ASM)용 비밀 ID	(Oracle ASM을 사용하는 Oracle 소스 엔드포인트에만 해당) Oracle ASM 액세스를 위해 AWS Secrets Manager 에서 생성한 비밀의 전체 Amazon 리소스 이름(ARN), 부분 ARN 또는 기억하기 쉬운 이름을 입력합니다. 이 비밀은 일반적으로 보안 ID로 식별되는 비밀과 동일한 서버에서 Oracle ASM에 액세스하기 위해 생성됩니다.
Oracle ASM용 IAM 역할	(Oracle ASM을 사용하는 Oracle 소스 엔드포인트에만 해당) 사용자를 대신하여 Oracle 자동 스토리지 관리(ASM)용 보안 암호 ID로 식별되는 보안 암호에 대한 액세스를 제공하기 위해 IAM에서 생성한 보안 암호 AWS DMS 액세스 역할의 ARN을 입력합니다.
수동으로 액세스 정보 제공	다음과 같은 일반 텍스트 자격 증명 정보를 설정합니다.
[서버 이름]	서버 이름을 입력합니다. 온프레미스 데이터베이스의 경우, 이것은 IP 주소 또는 퍼블릭 호스트 이름이 될 수 있습니다. Amazon RDS DB 인스턴스의 경우, 이것은 mysqlsrvinst.abcd12345678.us-west-2.rds.amazonaws.com 과 같이 DB 인스턴스의 엔드포인트(DNS 이름)가 될 수 있습니다.
포트	데이터베이스가 사용하는 포트를 입력합니다.

구성 옵션	구성 설정
Secure Socket Layer(SSL) 모드	이 엔드포인트에서 연결 암호화를 활성화해야 하는 경우 SSL 모드를 선택합니다. 선택하는 모드에 따라 인증서와 서버 인증서 정보를 제공해야 할 수도 있습니다.
사용자 이름	데이터 마이그레이션을 허용하는 데 필요한 권한을 가진 사용자 이름을 입력합니다. 필요한 권한에 대한 정보는 이 사용 설명서의 원본 또는 대상 데이터베이스 엔진의 보안 섹션을 참조하세요.
암호	필요한 권한을 가진 계정의 암호를 입력합니다. AWS DMS 소스 및 대상 엔드포인트의 암호에는 데이터베이스 엔진에 따라 문자 제한이 있습니다. 자세한 정보는 다음 표를 참조하세요.
데이터베이스 이름	일부 데이터베이스 엔진의 경우, 엔드포인트 데이터베이스로 사용하려는 데이터베이스의 이름.

다음 표에는 나열된 데이터베이스 엔진의 엔드포인트 암호 및 암호 관리자 보안 암호에서 지원되지 않는 문자가 나열되어 있습니다. 엔드포인트 암호에 쉼표(,)를 사용하려면 제공된 Secrets Manager 지원을 사용하여 AWS DMS 인스턴스에 대한 액세스를 AWS DMS 인증합니다. 자세한 내용은 [보안 암호를 사용하여 AWS Database Migration Service 엔드포인트 액세스](#) 단원을 참조하십시오.

이 데이터베이스 엔진의 경우	엔드포인트 암호 및 암호 관리자 보안 암호에는 다음 문자가 지원되지 않습니다.
모두	{ }
Microsoft Azure(소스로 사용되는 경우에만)	;
Microsoft SQL Server	, ;
MySQL 호환(MySQL, MariaDB, Amazon Aurora MySQL 포함)	;

이 데이터베이스 엔진의 경우	엔드포인트 암호 및 암호 관리자 보안 암호에는 다음 문자가 지원되지 않습니다.
Oracle	,
PostgreSQL, Amazon Aurora PostgreSQL 호환 에디션, Amazon Aurora Serverless(Aurora PostgreSQL 호환 에디션용 대상으로 사용되는 경우에만)	; + %
Amazon Redshift(대상으로 사용되는 경우에만)	, ;

2. 엔드포인트 설정을 선택하고 필요한 경우 AWS KMS key 을 선택합니다. [테스트 실행]을 선택하여 엔드포인트 연결을 테스트할 수 있습니다. 다음 표는 설정에 대한 설명입니다.

구성 옵션	구성 설정
엔드포인트 설정	여기에서 추가 연결 파라미터를 선택합니다. 엔드포인트 설정에 대한 자세한 내용은 1단계에서 지정한 소스 엔진 또는 대상 엔진의 설명서를 참조하세요. Oracle ASM을 사용하는 Oracle 소스 엔드포인트의 경우 1단계에서 수동으로 액세스 정보 제공을 선택한 경우 엔드포인트 설정에 입력하여 Oracle ASM 사용자 자격 증명 정보를 지정해야 할 수도 있습니다. 이러한 Oracle ASM 엔드포인트 설정에 대한 자세한 내용은 CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용 섹션을 참조하세요.
AWS KMS key	복제 스토리지와 연결 정보를 암호화하기 위해 사용할 암호화 키를 선택합니다. (기본값) aws/dms를 선택하면 계정 및 AWS 리전과 연결된 기본 AWS Key Management Service(AWS KMS) 키가 사용됩니다. 암호화 키 사용에 대한 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정를 참조하세요.

구성 옵션	구성 설정
엔드포인트 연결 테스트(선택 사항)	VPC 및 복제 인스턴스 이름을 추가합니다. 연결을 테스트하려면 테스트 실행을 선택합니다.

에서 Amazon RDS 엔드포인트에 대한 IAM 인증 사용 AWS DMS

AWS Identity and Access Management(IAM) 데이터베이스 인증은 IAM 자격 증명을 통해 AWS 데이터베이스 액세스를 관리하여 Amazon RDS 데이터베이스에 대한 보안을 강화합니다. IAM 인증은 기존 데이터베이스 암호를 사용하는 대신 AWS 자격 증명을 사용하여 15분 동안 유효한 수명이 짧은 인증 토큰을 생성합니다. 이 접근 방식은 애플리케이션 코드에 데이터베이스 암호를 저장할 필요가 없고, 자격 증명 노출 위험을 줄이고, IAM을 통해 중앙 집중식 액세스 관리를 제공하여 보안을 크게 개선합니다. 또한 기존 AWS IAM 역할 및 정책을 활용하여 액세스 관리를 간소화하므로 다른 AWS 서비스에 사용하는 것과 동일한 IAM 프레임워크를 사용하여 데이터베이스 액세스를 제어할 수 있습니다.

AWS DMS 는 이제 Amazon RDS에서 MySQL, PostgreSQL, Aurora PostgreSQL, Aurora MySQL 또는 MariaDB 엔드포인트에 연결할 때 DMS 버전 3.6.1 이상을 실행하는 복제 인스턴스에 대한 IAM 인증을 지원합니다. 이러한 엔진에 대한 새 엔드포인트를 생성할 때 데이터베이스 자격 증명을 제공하는 대신 IAM 인증을 선택하고 IAM 역할을 지정할 수 있습니다. 이 통합은 마이그레이션 작업에 대한 데이터베이스 암호를 관리하고 저장할 필요가 없으므로 보안을 강화합니다.

에서 Amazon RDS 엔드포인트에 대한 IAM 인증 구성 AWS DMS

엔드포인트를 생성할 때 Amazon RDS 데이터베이스에 대한 IAM 인증을 구성할 수 있습니다. IAM 인증을 구성하려면 다음을 수행합니다.

DMS 콘솔

1. Amazon RDS와 데이터베이스 사용자에게 IAM 인증이 활성화되어 있는지 확인합니다. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 [IAM 데이터베이스 인증 활성화 및 비활성화](#)를 참조하세요.
2. IAM 콘솔로 이동하여 아래 정책을 사용하여 IAM 역할을 생성합니다.

정책

JSON

```
{
```

```

    "Version": "2012-10-17",
    "Statement": [
      {
        "Sid": "VisualEditor0",
        "Effect": "Allow",
        "Action": [
          "rds-db:connect"
        ],
        "Resource": "*"
      }
    ]
  }
}

```

신뢰 정책:

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "dms.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

3. [AWS DMS 콘솔](#)의 엔드포인트 구성 중에 엔드포인트 데이터베이스 액세스 섹션으로 이동하여 IAM 인증을 선택합니다.
4. RDS 데이터베이스 인증에 대한 IAM 역할 드롭다운 메뉴에서 데이터베이스에 액세스할 수 있는 적절한 권한이 있는 IAM 역할을 선택합니다.

자세한 내용은 [소스 및 대상 엔드포인트 생성](#)을 참조하세요.

AWS CLI

1. Amazon RDS와 데이터베이스 사용자에게 IAM 인증이 활성화되어 있는지 확인합니다. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 [IAM 데이터베이스 인증 활성화 및 비활성화](#)를 참조하세요.
2. AWS CLI로 이동하여 IAM 역할을 생성하고 DMS가 역할을 수입하도록 허용합니다.

정책:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "rds-db:connect"
      ],
      "Resource": "*"
    }
  ]
}
```

신뢰 정책:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "dms.amazonaws.com"
        ]
      }
    }
  ]
}
```

```

    },
    "Action": "sts:AssumeRole"
  }
]
}

```

3. 다음 명령을 실행하여 인증서를 가져오고 PEM 파일을 다운로드합니다. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 [Amazon RDS용 인증서 번들 다운로드](#)를 참조하세요.

```
aws dms import-certificate --certificate-identifier rdsglobal --certificate-pem
file://~/global-bundle.pem
```

4. 다음 명령을 실행하여 IAM 엔드포인트를 만들 수 있습니다.

- PostgreSQL/Aurora PostgreSQL 엔드포인트의 경우(sslmode이 로 설정된 경우 required --certificate-arn 플래그가 필요하지 않음):

```
aws dms create-endpoint --endpoint-identifier <endpoint-name> --endpoint-
type <source/target> --engine-name <postgres/aurora-postgres> --username <db
username with iam auth privileges> --server-name <db server name> --port <port
number> --ssl-mode <required/verify-ca/verify-full> --postgres-sql-settings
{"\"ServiceAccessRoleArn\": \"role arn created from step 2 providing permissions
for iam authentication\", \"AuthenticationMethod\": \"iam\", \"DatabaseName\":
\"database name\"}" --certificate-arn <if sslmode is verify-ca/verify full use
cert arn generated in step 3, otherwise this parameter is not required>
```

- MySQL, MariaDB 또는 Aurora MySQL 엔드포인트의 경우:

```
aws dms create-endpoint --endpoint-identifier <endpoint-name> --endpoint-type
<source/target> --engine-name <mysql/mariadb/aurora> --username <db username
with iam auth privileges> --server-name <db server name> --port <port number>
--ssl-mode <verify-ca/verify-full> --my-sql-settings {"\"ServiceAccessRoleArn
\": \"role arn created from step 2 providing permissions for iam authentication
\", \"AuthenticationMethod\": \"iam\", \"DatabaseName\": \"database name\"}" --
certificate-arn <cert arn from previously imported cert in step 3>
```

5. 원하는 복제 인스턴스에 대해 테스트 연결을 실행하여 인스턴스 엔드포인트 연결을 생성하고 모든 것이 올바르게 설정되었는지 확인합니다.

```
aws dms test-connection --replication-instance-arn <replication instance arn> --
endpoint-arn <endpoint arn from previously created endpoint in step 4>
```

Note

IAM 인증을 사용하는 경우 테스트 연결에 제공된 복제 인스턴스는 AWS DMS 버전 3.6.1 이상이어야 합니다.

제한 사항

AWS DMS에는 Amazon RDS 엔드포인트에서 IAM 인증을 사용할 때 다음과 같은 제한 사항이 있습니다.

- 현재 Amazon RDS PostgreSQL 및 Amazon Aurora PostgreSQL 인스턴스는 IAM 인증을 통한 CDC 연결을 지원하지 않습니다. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 [IAM 데이터베이스 인증 제한 사항을](#) 참조하세요.

데이터 마이그레이션용 소스

AWS Database Migration Service (AWS DMS)는 가장 인기 있는 많은 데이터 엔진을 데이터 복제의 소스로 사용할 수 있습니다. 데이터베이스 소스는 Amazon EC2 인스턴스 또는 온프레미스 데이터베이스에서 실행되는 자체 관리형 엔진이 될 수 있습니다. 또는 Amazon RDS 또는 Amazon S3와 같은 AWS 서비스의 데이터 소스일 수 있습니다.

유효한 소스의 전체 목록은 [AWS DMS용 소스](#) 섹션을 참조하세요.

주제

- [Oracle 데이터베이스를의 소스로 사용 AWS DMS](#)
- [Microsoft SQL Server 데이터베이스를의 소스로 사용 AWS DMS](#)
- [Microsoft Azure SQL 데이터베이스를의 소스로 사용 AWS DMS](#)
- [Microsoft Azure SQL 관리형 인스턴스를의 소스로 사용 AWS DMS](#)
- [Microsoft Azure Database for PostgreSQL 유연한 서버를의 소스로 사용 AWS DMS](#)
- [Microsoft Azure Database for MySQL 유연한 서버를의 소스로 사용 AWS DMS](#)
- [OCI MySQL Heatwave를의 소스로 사용 AWS DMS](#)
- [MySQL용 Google Cloud를의 소스로 사용 AWS DMS](#)
- [Google Cloud for PostgreSQL을의 소스로 사용 AWS DMS](#)
- [PostgreSQL 데이터베이스를 AWS DMS 소스로 사용](#)

- [MySQL 호환 데이터베이스들의 소스로 사용 AWS DMS](#)
- [SAP ASE 데이터베이스들의 소스로 사용 AWS DMS](#)
- [MongoDB들의 소스로 사용 AWS DMS](#)
- [Amazon DocumentDB\(MongoDB 호환\)를 소스로 사용 AWS DMS](#)
- [Amazon S3들의 소스로 사용 AWS DMS](#)
- [Linux, Unix, Windows 및 Amazon RDS 데이터베이스\(Db2 LUW\)용 IBM Db2들의 소스로 사용 AWS DMS](#)
- [IBM Db2 for z/OS 데이터베이스들의 소스로 사용 AWS DMS](#)

Oracle 데이터베이스들의 소스로 사용 AWS DMS

를 사용하여 하나 이상의 Oracle 데이터베이스에서 데이터를 마이그레이션할 수 있습니다 AWS DMS. Oracle 데이터베이스를 소스로 사용하여 AWS DMS가 지원하는 대상으로 데이터를 마이그레이션할 수 있습니다.

AWS DMS 는 다음 Oracle 데이터베이스 에디션을 지원합니다.

- Oracle Enterprise Edition
- Oracle Standard Edition
- Oracle Express Edition
- Oracle Personal Edition

가 소스로 AWS DMS 지원하는 Oracle 데이터베이스 버전에 대한 자세한 내용은 [섹션을 참조하세요](#) [소스 AWS DMS](#).

Secure Sockets Layer(SSL)를 사용하여 Oracle 엔드포인트와 복제 인스턴스 간의 연결을 암호화할 수 있습니다. Oracle 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 [Oracle 엔드포인트용 SSL 지원](#) 섹션을 참조하세요.

AWS DMS 는 Oracle 투명 데이터 암호화(TDE)를 사용하여 소스 데이터베이스의 저장 데이터를 암호화할 수 있도록 지원합니다. Oracle 소스 엔드포인트에서 Oracle TDE를 사용하는 방법에 대한 자세한 내용은 [Oracle을 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS](#) 섹션을 참조하세요.

AWS 는 Oracle 엔드포인트(및 기타 모든 엔드포인트 유형)에서 TLS 버전 1.2 이상의 사용을 지원하며 TLS 버전 1.3 이상을 사용할 것을 권장합니다.

다음 단계에 따라 Oracle 데이터베이스를 AWS DMS 소스 엔드포인트로 구성합니다.

1. 가 Oracle 소스 데이터베이스에 액세스할 수 AWS DMS 있는 적절한 권한을 가진 Oracle 사용자를 생성합니다.
2. 선택한 Oracle 데이터베이스 구성을 준수하는 Oracle 소스 엔드포인트를 생성합니다. 전체 로드 전용 작업을 생성하기 위한 추가 구성이 필요하지 않습니다.
3. 변경 데이터 캡처를 처리하는 작업(CDC 전용 또는 전체 로드 및 CDC 작업)을 생성하려면 Oracle LogMiner 또는 AWS DMS Binary Reader를 선택하여 데이터 변경 사항을 캡처합니다. LogMiner 또는 Binary Reader 중 어느 것을 선택하는지에 따라 이후 권한 및 구성 옵션의 일부가 결정됩니다. LogMiner와 Binary Reader를 비교하려면 다음 섹션을 참조하세요.

 Note

전체 로드 작업, CDC 전용 작업, 전체 로드 및 CDC 작업에 대한 자세한 내용은 [작업 생성](#)을 참조하세요.

Oracle 소스 데이터베이스 및 작업에 대한 자세한 내용은 다음 섹션을 AWS DMS참조하세요.

주제

- [CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용](#)
- [에 대한 자체 관리형 또는 AWS관리형 Oracle 소스 데이터베이스를 구성하기 위한 워크플로 AWS DMS Oracle 소스 데이터베이스 구성](#)
- [의 소스로 자체 관리형 Oracle 데이터베이스 작업 AWS DMS](#)
- [AWS의 소스로 관리형 Oracle 데이터베이스 작업 AWS DMS](#)
- [Oracle을의 소스로 사용할 때의 제한 사항 AWS DMS](#)
- [Oracle 엔드포인트용 SSL 지원](#)
- [Oracle을의 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS](#)
- [Oracle을의 소스로 사용하는 데 지원되는 압축 방법 AWS DMS](#)
- [Oracle을의 소스로 사용하여 중첩 테이블 복제 AWS DMS](#)
- [Oracle을의 소스로 사용할 때 Oracle ASM에 REDO 저장 AWS DMS](#)
- [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#)
- [Oracle용 소스 데이터 형식](#)

CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용

에는 Oracle에 대한 변경 데이터 캡처(CDC)를 소스로 수행할 때 다시 실행 로그를 읽는 두 AWS DMS 가지 방법이 있습니다. Oracle LogMiner와 AWS DMS Binary Reader입니다. LogMiner는 온라인 다시 실행 로그와 보관된 다시 실행 로그 파일을 읽는 Oracle API입니다. Binary Reader는 원시 다시 실행 로그 파일을 직접 읽고 구문 분석하는 AWS DMS 방법입니다. 이 메서드의 특징은 다음과 같습니다.

기능	LogMiner	Binary Reader
간편한 구성	예	아니요
소스 시스템 I/O 및 CPU에 미치는 영향 감소	아니요	예
향상된 CDC 성능	아니요	예
Oracle 테이블 클러스터 지원	예	아니요
모든 유형의 Oracle HCC(Hybrid Columnar Compression) 지원	예	부분적으로 Binary Reader는 CDC 작업에서 QUERY LOW를 지원하지 않습니다. 다른 모든 HCC 유형은 완벽하게 지원됩니다.
Oracle 12c에서만 LOB 열 지원	아니요(Oracle 12c에서는 LogMiner에 LOB 지원을 사용할 수 없습니다.)	예
LOB 열에만 영향을 미치는 UPDATE 문 지원	아니요	예
Oracle 투명한 데이터 암호화(TDE) 지원	부분적으로 Oracle LogMiner를 사용하는 경	부분적으로 Binary Reader는 자체 관리형

기능	LogMiner	Binary Reader
	우 Amazon RDS for Oracle의 열 수준에서 TDE 암호화를 지원하지 AWS DMS 않습니다.	Oracle 데이터베이스에서만 TDE를 지원합니다.
모든 Oracle 압축 방법 지원	예	아니요
XA 트랜잭션 지원	아니요	예
RAC	예	예
	성능상의 이유와 일부 내부 DMS 제한으로 인해 권장되지 않습니다.	적극적으로 권장됨

Note

기본적으로는 (CDC)용 Oracle LogMiner를 AWS DMS 사용합니다. AWS DMS 는 Oracle 소스 데이터베이스로 작업할 때 투명한 데이터 암호화(TDE) 방법을 지원합니다. 지정한 TDE 자격 증명이 올바르지 않아도 AWS DMS 마이그레이션 작업이 실패하지 않으므로 암호화된 테이블의 지속적 복제에 영향을 미칠 수 있습니다. TDE 자격 증명에 대한 자세한 내용은 [Oracle을의 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS](#)을 참조하세요.

LogMiner들과 함께 사용할 때의 주요 이점은 다음과 AWS DMS 같습니다.

- LogMiner는 암호화 옵션과 압축 옵션과 같은 대다수 Oracle 옵션을 지원합니다. Binary Reader는 모든 Oracle 옵션을 지원하지 않고, 특히 압축과 대부분의 암호화 옵션을 지원합니다.
- LogMiner는 특히 Binary Reader 직접 액세스 설정과 비교할 때 또는 Oracle ASM(Automatic Storage Management)을 사용하여 다시 실행 로그를 관리하는 경우 구성이 더 단순합니다.

- LogMiner에서 사용할 테이블 클러스터를 지원합니다 AWS DMS. Binary Reader는 테이블 클러스터를 지원하지 않습니다.

에서 Binary Reader를 사용할 때의 주요 이점은 다음과 AWS DMS 같습니다.

- 대용량 변경 사항으로 마이그레이션할 때 LogMiner의 일부 I/O 또는 CPU가 Oracle 원본 데이터베이스를 호스팅하는 컴퓨터에 영향을 줄 수도 있습니다. Binary Reader는 여러 데이터베이스 쿼리를 수행하는 대신 로그를 직접 마이닝하기 때문에 I/O 또는 CPU에 영향을 미칠 가능성이 적습니다.
- 대용량 변경 사항으로 마이그레이션할 때 Binary Reader를 사용하면 Oracle LogMiner를 사용하는 경우에 비해 CDC 성능이 대개 훨씬 더 우수합니다.
- Binary Reader는 Oracle 버전 12c에서 LOB용 CDC를 지원합니다. 하지만 LogMiner는 그렇지 않습니다.

일반적으로 다음 상황에 있는 경우 Oracle 데이터베이스를 마이그레이션할 때 Oracle LogMiner를 사용합니다.

- 원본 Oracle 데이터베이스에서 여러 마이그레이션 작업을 실행해야 합니다.
- 소스 Oracle 데이터베이스의 변경 사항 볼륨 또는 다시 실행 로그 볼륨이 크거나 변경 사항이 있고 Oracle ASM도 사용 중입니다.

Note

Oracle LogMiner와 AWS DMS Binary Reader 사용 간에 변경하는 경우 CDC 작업을 다시 시작해야 합니다.

Oracle 소스 데이터베이스에서의 CDC 구성

Oracle 소스 엔드포인트가 변경 데이터 캡처(CDC) 작업을 위해 데이터베이스에 연결하려면 추가 연결 속성을 지정해야 할 수 있습니다. 이는 전체 로드 및 CDC 작업이나 CDC 전용 작업에도 해당될 수 있습니다. 지정하는 추가 연결 속성은 다시 실행 로그에 액세스하는 데 사용하는 방법인 Oracle LogMiner 또는 AWS DMS Binary Reader에 따라 다릅니다.

소스 엔드포인트를 생성할 때 추가 연결 속성을 지정합니다. 연결 속성 설정이 여러 개인 경우, 추가 공백 없이 세미콜론으로 구분하세요(예: `oneSetting;thenAnother`).

AWS DMS 는 기본적으로 LogMiner를 사용합니다. LogMiner를 사용하기 위해 추가 연결 속성을 지정할 필요는 없습니다.

Binary Reader를 사용하여 다시 실행 로그에 액세스하려면 다음 추가 연결 속성을 추가합니다.

```
useLogMinerReader=N;useBfile=Y;
```

다음 형식을 추가 연결 속성에 사용하여 Binary Reader와 함께 ASM을 사용하는 서버에 액세스합니다.

```
useLogMinerReader=N;useBfile=Y;asm_user=asm_username;asm_server=RAC_server_ip_address:port_number
+ASM;
```

소스 엔드포인트 Password 요청 파라미터를 다음과 같이 쉼표로 구분된 Oracle 사용자 암호 및 ASM 암호로 설정합니다.

```
oracle_user_password,asm_user_password
```

Oracle 소스가 ASM을 사용하는 경우, 대규모 트랜잭션 처리를 위해 Binary Reader의 고성능 옵션을 사용할 수 있습니다. 이러한 옵션에는 병렬 스레드 수(parallelASMReadThreads)와 미리 읽기 버퍼 수(readAheadBlocks)를 지정하는 추가 연결 속성이 포함됩니다. 이러한 속성을 함께 설정하면 CDC 작업의 성능을 크게 개선할 수 있습니다. 다음 설정은 대부분의 ASM 구성에서 양호한 결과를 제공합니다.

```
useLogMinerReader=N;useBfile=Y;asm_user=asm_username;asm_server=RAC_server_ip_address:port_number
+ASM;
parallelASMReadThreads=6;readAheadBlocks=150000;
```

추가 연결 속성이 지원하는 값에 대한 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요.

또한 ASM을 사용하는 Oracle 소스가 있는 CDC 작업의 성능은 선택하는 다른 설정에 따라 달라집니다. 이러한 설정에는 AWS DMS 추가 연결 속성 및 Oracle 소스를 구성하기 위한 SQL 설정이 포함됩니다. ASM을 사용하는 Oracle 소스의 추가 연결 속성에 대한 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#)을 참조하세요.

또한 적절한 CDC 시작점을 선택해야 합니다. 일반적으로 이 경우, CDC를 시작할 가장 이른 열린 트랜잭션을 캡처하는 트랜잭션 처리 지점을 식별해야 합니다. 그렇지 않으면 CDC 작업에서 이전의 열린

트랜잭션이 누락될 수 있습니다. Oracle 소스 데이터베이스의 경우, Oracle SCN(시스템 변경 번호)을 기준으로 CDC 네이티브 시작점을 선택하면 가장 이른 열린 트랜잭션을 식별할 수 있습니다. 자세한 내용은 [CDC 시작 지점에서 복제를 시작](#) 단원을 참조하십시오.

소스로 사용되는 자체 관리형 Oracle 데이터베이스에서 CDC를 구성하는 방법에 대한 자세한 내용은 [Oracle LogMiner를 사용하여 다시 실행 로그에 액세스할 때 필요한 계정 권한](#), [AWS DMS Binary Reader를 사용하여 다시 실행 로그에 액세스할 때 필요한 계정 권한](#), [Oracle ASM과 함께 Binary Reader를 사용할 때 필요한 추가 계정 권한](#) 섹션을 참조하세요.

AWS관리형 Oracle 데이터베이스에 대한 CDC를 소스로 구성하는 방법에 대한 자세한 내용은 [용 RDS for Oracle 소스와 함께 Binary Reader를 사용하도록 CDC 작업 구성 AWS DMS](#) 및 섹션을 참조하세요. [요에서 CDC용 Binary Reader와 함께 Amazon RDS Oracle Standby\(읽기 전용 복제본\)를 소스로 사용 AWS DMS](#).

에 대한 자체 관리형 또는 AWS관리형 Oracle 소스 데이터베이스를 구성하기 위한 워크플로 AWS DMS

에 대한 자체 관리형 또는 AWS관리형 Oracle 소스 데이터베이스를 구성하기 위한 워크플로 AWS DMS

자체 관리형 소스 데이터베이스 인스턴스를 구성하려면 CDC 수행 방식에 따라 다음 워크플로 단계를 사용하세요.

이 워크플로 단계의 경우	LogMiner를 사용하여 CDC를 수행하는 경우 다음을 수행	Binary Reader를 사용하여 CDC를 수행하는 경우 다음을 수행
Oracle 계정 권한을 부여합니다.	용 자체 관리형 Oracle 소스에 필요한 사용자 계정 권한 AWS DMS 를 참조하세요.	용 자체 관리형 Oracle 소스에 필요한 사용자 계정 권한 AWS DMS 을 참조하세요.
CDC를 사용하여 복제할 소스 데이터베이스를 준비합니다.	를 사용하여 CDC용 Oracle 자체 관리형 소스 데이터베이스 준비 AWS DMS 을 참조하세요.	를 사용하여 CDC용 Oracle 자체 관리형 소스 데이터베이스 준비 AWS DMS 을 참조하세요.
CDC에 필요한 추가 Oracle 사용자 권한을 부여합니다.	Oracle LogMiner를 사용하여 다시 실행 로그에 액세스할 때 필요한 계정 권한 을 참조하세요.	AWS DMS Binary Reader를 사용하여 다시 실행 로그에 액세스할 때 필요한 계정 권한 를 참조하세요.

이 워크플로 단계의 경우	LogMiner를 사용하여 CDC를 수행하는 경우 다음을 수행	Binary Reader를 사용하여 CDC를 수행하는 경우 다음을 수행
ASM을 사용하는 Oracle 인스턴스의 경우, CDC를 위해 ASM에 액세스하는 데 필요한 추가 사용자 계정 권한을 부여합니다.	추가 작업 없음.는 추가 계정 권한 없이 Oracle ASM을 AWS DMS 지원합니다.	Oracle ASM과 함께 Binary Reader를 사용할 때 필요한 추가 계정 권한을(를) 참조하세요.
아직 구성하지 않았다면 CDC에 LogMiner 또는 Binary Reader를 사용하도록 작업을 구성합니다.	CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용 을 참조하세요.	CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용 을 참조하세요.
Oracle Standby를 CDC의 소스로 구성합니다.	AWS DMS 는 Oracle Standby를 소스로 지원하지 않습니다.	에서 CDC용 Binary Reader를 사용하여 자체 관리형 Oracle Standby를 소스로 사용 AWS DMS 을(를) 참조하세요.

다음 워크플로 단계를 사용하여 AWS관리형 Oracle 소스 데이터베이스 인스턴스를 구성합니다.

이 워크플로 단계의 경우	LogMiner를 사용하여 CDC를 수행하는 경우 다음을 수행	Binary Reader를 사용하여 CDC를 수행하는 경우 다음을 수행
Oracle 계정 권한을 부여합니다.	자세한 내용은 용 AWS관리형 Oracle 소스에 필요한 사용자 계정 권한 AWS DMS 단원을 참조하십시오.	자세한 내용은 용 AWS관리형 Oracle 소스에 필요한 사용자 계정 권한 AWS DMS 단원을 참조하십시오.
CDC를 사용하여 복제할 소스 데이터베이스를 준비합니다.	자세한 내용은 AWS에 대한 관리형 Oracle 소스 구성 AWS DMS 단원을 참조하십시오.	자세한 내용은 AWS에 대한 관리형 Oracle 소스 구성 AWS DMS 단원을 참조하십시오.
CDC에 필요한 추가 Oracle 사용자 권한을 부여합니다.	추가 계정 권한은 필요하지 않습니다.	자세한 내용은 용 RDS for Oracle 소스와 함께 Binary

이 워크플로 단계의 경우	LogMiner를 사용하여 CDC를 수행하는 경우 다음을 수행	Binary Reader를 사용하여 CDC를 수행하는 경우 다음을 수행
		Reader를 사용하도록 CDC 작업 구성 AWS DMS 단원을 참조하십시오.
아직 구성하지 않았다면 CDC에 LogMiner 또는 Binary Reader를 사용하도록 작업을 구성합니다.	자세한 내용은 CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용 단원을 참조하십시오.	자세한 내용은 CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용 단원을 참조하십시오.
Oracle Standby를 CDC의 소스로 구성합니다.	AWS DMS는 Oracle Standby를 소스로 지원하지 않습니다.	자세한 내용은 에서 CDC용 Binary Reader와 함께 Amazon RDS Oracle Standby(읽기 전용 복제본)를 소스로 사용 AWS DMS 단원을 참조하십시오.

의 소스로 자체 관리형 Oracle 데이터베이스 작업 AWS DMS

자체 관리형 데이터베이스는 사용자가 구성 및 제어하는 데이터베이스로, 로컬 온프레미스 데이터베이스 인스턴스이거나 Amazon EC2의 데이터베이스입니다. 아래에서 자체 관리형 Oracle 데이터베이스를 함께 사용할 때 필요한 권한 및 구성에 대해 알아볼 수 있습니다 AWS DMS.

용 자체 관리형 Oracle 소스에 필요한 사용자 계정 권한 AWS DMS

에서 Oracle 데이터베이스를 소스로 사용하려면 Oracle 엔드포인트 연결 설정에 지정된 Oracle 사용자에게 다음 권한을 AWS DMS부여합니다.

Note

권한을 부여할 때는 각 객체의 동의어가 아닌 객체의 실제 이름을 사용하세요. 예를 들어 밀줄이 없는 V_\$OBJECT가 아니라 밀줄을 포함하여 V\$OBJECT를 사용하세요.

```
GRANT CREATE SESSION TO dms_user;
```

```

GRANT SELECT ANY TRANSACTION TO dms_user;
GRANT SELECT ON V_$ARCHIVED_LOG TO dms_user;
GRANT SELECT ON V_$LOG TO dms_user;
GRANT SELECT ON V_$LOGFILE TO dms_user;
GRANT SELECT ON V_$LOGMNR_LOGS TO dms_user;
GRANT SELECT ON V_$LOGMNR_CONTENTS TO dms_user;
GRANT SELECT ON V_$DATABASE TO dms_user;
GRANT SELECT ON V_$THREAD TO dms_user;
GRANT SELECT ON V_$PARAMETER TO dms_user;
GRANT SELECT ON V_$NLS_PARAMETERS TO dms_user;
GRANT SELECT ON V_$TIMEZONE_NAMES TO dms_user;
GRANT SELECT ON V_$TRANSACTION TO dms_user;
GRANT SELECT ON V_$CONTAINERS TO dms_user;
GRANT SELECT ON ALL_INDEXES TO dms_user;
GRANT SELECT ON ALL_OBJECTS TO dms_user;
GRANT SELECT ON ALL_TABLES TO dms_user;
GRANT SELECT ON ALL_USERS TO dms_user;
GRANT SELECT ON ALL_CATALOG TO dms_user;
GRANT SELECT ON ALL_CONSTRAINTS TO dms_user;
GRANT SELECT ON ALL_CONS_COLUMNS TO dms_user;
GRANT SELECT ON ALL_TAB_COLS TO dms_user;
GRANT SELECT ON ALL_IND_COLUMNS TO dms_user;
GRANT SELECT ON ALL_ENCRYPTED_COLUMNS TO dms_user;
GRANT SELECT ON ALL_LOG_GROUPS TO dms_user;
GRANT SELECT ON ALL_TAB_PARTITIONS TO dms_user;
GRANT SELECT ON SYS.DBA_REGISTRY TO dms_user;
GRANT SELECT ON SYS.OBJ$ TO dms_user;
GRANT SELECT ON DBA_TABLESPACES TO dms_user;
GRANT SELECT ON DBA_OBJECTS TO dms_user; -- Required if the Oracle version is earlier
    than 11.2.0.3.
GRANT SELECT ON SYS.ENC$ TO dms_user; -- Required if transparent data encryption (TDE)
    is enabled. For more information on using Oracle TDE with AWS DMS, see Oracle## ### ##
    ## ## ##### ### ## AWS DMS.
GRANT SELECT ON GV_$TRANSACTION TO dms_user; -- Required if the source database is
    Oracle RAC in AWS DMS versions 3.4.6 and higher.
GRANT SELECT ON V_$DATAGUARD_STATS TO dms_user; -- Required if the source database
    is Oracle Data Guard and Oracle Standby is used in the latest release of DMS version
    3.4.6, version 3.4.7, and higher.
GRANT SELECT ON V_$DATABASE_INCARNATION TO dms_user;

```

특정 테이블 목록을 사용하는 경우, 복제된 각 테이블에 대해 다음 추가 권한을 부여합니다.

```
GRANT SELECT on any-replicated-table to dms_user;
```

검증 기능을 사용할 수 있는 다음 추가 권한을 부여합니다.

```
GRANT EXECUTE ON SYS.DBMS_CRYPT0 TO dms_user;
```

LogMiner 대신 Binary Reader를 사용하는 경우, 다음 추가 권한을 부여합니다.

```
GRANT SELECT ON SYS.DBA_DIRECTORIES TO dms_user;
```

뷰를 노출할 수 있는 다음 추가 권한을 부여합니다.

```
GRANT SELECT on ALL_VIEWS to dms_user;
```

뷰를 노출하려면 소스 엔드포인트에 `exposeViews=true` 추가 연결 속성도 추가해야 합니다.

서버리스 복제를 사용할 경우, 다음 추가 권한을 부여합니다.

```
GRANT SELECT on dba_segments to dms_user;  
GRANT SELECT on v_$tablespace to dms_user;  
GRANT SELECT on dba_tab_subpartitions to dms_user;  
GRANT SELECT on dba_extents to dms_user;
```

서버리스 복제에 대해서는 [AWS DMS 서버리스 작업을](#) 참조하세요.

Oracle 관련 마이그레이션 전 평가를 사용할 경우, 다음 추가 권한을 부여합니다.

```
GRANT SELECT on gv_$parameter to dms_user;  
GRANT SELECT on v_$instance to dms_user;  
GRANT SELECT on v_$version to dms_user;  
GRANT SELECT on gv_$ASM_DISKGROUP to dms_user;  
GRANT SELECT on gv_$database to dms_user;  
GRANT SELECT on dba_db_links to dms_user;  
GRANT SELECT on gv_$log_History to dms_user;  
GRANT SELECT on gv_$log to dms_user;  
GRANT SELECT ON DBA_TYPES TO dms_user;  
GRANT SELECT ON DBA_USERS to dms_user;  
GRANT SELECT ON DBA_DIRECTORIES to dms_user;  
GRANT EXECUTE ON SYS.DBMS_XMLGEN TO dms_user;
```

Oracle 관련 마이그레이션 전 평가에 대해서는 [Oracle 평가](#)를 참조하세요.

Oracle Standby 열린 트랜잭션 처리를 위한 사전 요구 사항

AWS DMS 버전 3.4.6 이상을 사용하는 경우 다음 단계를 수행하여 Oracle Standby에 대한 열린 트랜잭션을 처리합니다.

1. 기본 데이터베이스에 AWSDMS_DBLINK라는 데이터베이스 링크를 생성합니다. **DMS_USER**는 이 데이터베이스 링크를 사용하여 기본 데이터베이스에 연결합니다. 이 데이터베이스 링크는 대기 인스턴스에서 실행되어 기본 데이터베이스에서 실행 중인 열린 트랜잭션을 쿼리합니다. 다음 예제를 참조하세요.

```
CREATE PUBLIC DATABASE LINK AWSDMS_DBLINK
CONNECT TO DMS_USER IDENTIFIED BY DMS_USER_PASSWORD
USING '(DESCRIPTION=
      (ADDRESS=(PROTOCOL=TCP)(HOST=PRIMARY_HOST_NAME_OR_IP)(PORT=PORT))
      (CONNECT_DATA=(SERVICE_NAME=SID))
    )';
```

2. 다음 예제와 같이 **DMS_USER**를 사용하여 데이터베이스 링크에 연결되었는지 확인합니다.

```
select 1 from dual@AWSDMS_DBLINK
```

를 사용하여 CDC용 Oracle 자체 관리형 소스 데이터베이스 준비 AWS DMS

다음을 수행하여 자체 관리형 Oracle 데이터베이스를 CDC 작업을 실행할 소스로 준비합니다.

- [가 소스 데이터베이스 버전을 AWS DMS 지원하는지 확인.](#)
- [ARCHIVELOG 모드가 켜져 있는지 확인.](#)
- [보충 로깅 설정.](#)

가 소스 데이터베이스 버전을 AWS DMS 지원하는지 확인

다음과 같은 쿼리를 실행하여 AWS DMS가 현재 버전의 Oracle 데이터베이스를 지원하는지 확인합니다.

```
SELECT name, value, description FROM v$parameter WHERE name = 'compatible';
```

여기서 name, value 및 description은 name의 값을 기준으로 쿼리되는 데이터베이스의 어딘가에 있는 열입니다. 이 쿼리가 오류 없이 실행되는 경우는 데이터베이스의 현재 버전을 AWS DMS 지원하며 마이그레이션을 계속할 수 있습니다. 쿼리에서 오류가 발생하면 AWS DMS는 데이터베이스의 현

재 버전을 지원하지 않습니다. 마이그레이션을 계속하려면 먼저 Oracle 데이터베이스를에서 지원하는 버전으로 변환합니다 AWS DMS.

ARCHIVELOG 모드가 켜져 있는지 확인

Oracle은 ARCHIVELOG 모드와 NOARCHIVELOG 모드 두 가지 모드로 실행할 수 있습니다. CDC 작업을 실행하려면 데이터베이스를 ARCHIVELOG 모드에서 실행합니다. 데이터베이스가 ARCHIVELOG 모드 상태인지 확인하려면 다음 쿼리를 실행합니다.

```
SQL> SELECT log_mode FROM v$database;
```

NOARCHIVELOG 모드가 반환되면 Oracle 지침에 따라 데이터베이스를 ARCHIVELOG로 설정합니다.

보충 로깅 설정

진행 중인 변경 사항을 캡처하려면 Oracle 소스 데이터베이스에서 최소 보충 로깅을 활성화 AWS DMS 해야 합니다. 또한 데이터베이스의 복제된 각 테이블에서 보충 로깅을 활성화해야 합니다.

기본적으로는 복제된 모든 테이블에 PRIMARY KEY 보충 로깅을 AWS DMS 추가합니다. AWS DMS 가 PRIMARY KEY 보충 로깅을 추가하도록 허용하려면 복제된 각 테이블에 대해 다음 권한을 부여합니다.

```
ALTER on any-replicated-table;
```

추가 연결 속성을 AWS DMS 사용하여 추가된 기본 PRIMARY KEY 보충 로깅을 비활성화할 수 있습니다 `addSupplementalLogging`. 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

복제 작업에서 WHERE 절을 사용하여 프라이머리 키 열을 참조하지 않는 테이블을 업데이트하는 경우, 보충 로깅을 켜야 합니다.

수동으로 보충 로깅을 설정하려면

1. 다음 쿼리를 실행하여 데이터베이스에 보충 로깅이 이미 활성화되어 있는지 확인합니다.

```
SELECT supplemental_log_data_min FROM v$database;
```

반환된 결과가 YES 또는 IMPLICIT인 경우, 데이터베이스에 보충 로깅이 활성화되어 있습니다.

활성화되어 있지 않다면 다음 명령을 실행하여 데이터베이스에 보충 로깅을 활성화합니다.


```
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
```

2. 복제된 각 테이블에 필요한 보충 로깅이 추가되었는지 확인합니다.

다음을 고려하세요.

- 테이블에 ALL COLUMNS 보충 로깅이 추가되면 로깅을 더 추가할 필요가 없습니다.
- 프라이머리 키가 있는 경우 프라이머리 키에 대한 보충 로깅을 추가합니다. 형식을 사용하여 프라이머리 키 자체에 보충 로깅을 추가하거나 데이터베이스에서 프라이머리 키 열에 보충 로깅을 추가하여 이 작업을 수행할 수 있습니다.

```
ALTER TABLE Tablename ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;  
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;
```

- 프라이머리 키가 없고 테이블에 고유 인덱스가 하나만 있는 경우, 고유 인덱스의 모든 열을 보충 로그에 추가합니다.

```
ALTER TABLE TableName ADD SUPPLEMENTAL LOG GROUP LogGroupName  
(UniqueIndexColumn1[, UniqueIndexColumn2] ...) ALWAYS;
```

SUPPLEMENTAL LOG DATA (UNIQUE INDEX) COLUMNS를 사용해도 고유 인덱스 열이 로그에 추가되지 않습니다.

- 프라이머리 키가 없고 테이블에 여러 고유 인덱스가 있는 경우는 알파벳순으로 정렬된 오름차순 목록에서 첫 번째 고유 인덱스를 AWS DMS 선택합니다. 이전 항목에서처럼 선택한 인덱스의 열에 보충 로깅을 추가해야 합니다.

SUPPLEMENTAL LOG DATA (UNIQUE INDEX) COLUMNS를 사용해도 고유 인덱스 열이 로그에 추가되지 않습니다.

- 프라이머리 키가 없고 고유 인덱스가 없는 경우 모든 열에 보충 로깅을 추가합니다.

```
ALTER TABLE TableName ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
```

일부의 경우, 대상 테이블 프라이머리 키 또는 고유 색인이 원본 테이블 프라이머리 키 또는 고유 색인과 다릅니다. 이러한 경우, 대상 테이블 프라이머리 키 또는 고유 인덱스를 구성하는 소스 테이블 열에 보충 로깅을 수동으로 추가합니다.

또한 대상 테이블 프라이머리 키를 변경하는 경우, 원래 프라이머리 키 또는 고유 인덱스 대신 대상 고유 인덱스에 보충 로깅을 추가해야 합니다.

테이블에 대해 필터 또는 변환이 정의된 경우 추가 로깅을 활성화해야 할 수 있습니다.

다음을 고려하세요.

- 테이블에 ALL COLUMNS 보충 로깅이 추가되면 로깅을 더 추가할 필요가 없습니다.
- 테이블에 고유 인덱스 또는 프라이머리 키가 있다면 필터 또는 변환에 포함된 각 열에 보충 로깅을 추가합니다. 하지만 해당 열이 프라이머리 키 또는 고유 인덱스 열과 다른 경우에만 그렇게 하세요.
- 변환에 하나의 열만 포함되는 경우, 이 열을 보충 로깅 그룹에 추가하지 마세요. 예를 들어 A+B 변환의 경우 열 A 및 B 모두에 보충 로깅을 추가합니다. 그러나 substring(A,10) 변환의 경우 열 A에 보충 로깅을 추가하지 마세요.
- 프라이머리 키 또는 고유 인덱스 열과 필터링되거나 변환되는 그 밖의 열에 보충 로깅을 설정하려면 USER_LOG_GROUP 보충 로깅을 설정하면 됩니다. 프라이머리 키 또는 고유 인덱스 열과 필터링되거나 변환되는 그 밖의 특정 열 모두에 이 보충 로깅을 추가합니다.

예를 들어, 프라이머리 키 ID와 열 NAME 기준 필터를 사용하여 TEST.LOGGING이라는 테이블을 복제하려면 다음과 비슷한 명령을 실행하여 로그 그룹 보충 로깅을 만들 수 있습니다.

```
ALTER TABLE TEST.LOGGING ADD SUPPLEMENTAL LOG GROUP TEST_LOG_GROUP (ID, NAME) ALWAYS;
```

Oracle LogMiner를 사용하여 다시 실행 로그에 액세스할 때 필요한 계정 권한

Oracle LogMiner를 사용하여 다시 실행 로그에 액세스하려면 Oracle 엔드포인트 연결 설정에 지정된 Oracle 사용자에게 다음 권한을 부여합니다.

```
GRANT EXECUTE on DBMS_LOGMNR to dms_user;
GRANT SELECT on V_$LOGMNR_LOGS to dms_user;
GRANT SELECT on V_$LOGMNR_CONTENTS to dms_user;
GRANT LOGMINING to dms_user; -- Required only if the Oracle version is 12c or higher.
```

AWS DMS Binary Reader를 사용하여 다시 실행 로그에 액세스할 때 필요한 계정 권한

AWS DMS Binary Reader를 사용하여 다시 실행 로그에 액세스하려면 Oracle 엔드포인트 연결 설정에 지정된 Oracle 사용자에게 다음 권한을 부여합니다.

```
GRANT SELECT on v_$transportable_platform to dms_user;    -- Grant this privilege if the
redo logs are stored in Oracle Automatic Storage Management (ASM) and AWS DMS accesses
them from ASM.
GRANT CREATE ANY DIRECTORY to dms_user;                  -- Grant this privilege to
allow AWS DMS to use Oracle BFILE read file access in certain cases. This access is
```

```
required when the replication instance does not have file-level access to the redo
logs and the redo logs are on non-ASM storage.
GRANT EXECUTE on DBMS_FILE_TRANSFER to dms_user;           -- Grant this privilege to
copy the redo log files to a temporary folder using the CopyToTempFolder method.
GRANT EXECUTE on DBMS_FILE_GROUP to dms_user;
```

Binary Reader는 Oracle 디렉터리를 포함하는 Oracle 파일 기능과 함께 작동합니다. 각 Oracle 디렉터리 객체에는 처리할 다시 실행 로그 파일이 들어 있는 폴더의 이름이 포함됩니다. 이러한 Oracle 디렉터리는 파일 시스템 수준에서 표시되지 않습니다. 대신, Oracle 데이터베이스 수준에서 생성되는 논리적 디렉터리입니다. Oracle ALL_DIRECTORIES 뷰에서 볼 수 있습니다.

이러한 Oracle 디렉터리 AWS DMS 를 생성하려면 앞에서 지정한 권한을 부여합니다. CREATE ANY DIRECTORY는 DMS_ 접두사가 있는 디렉터리 이름을 AWS DMS 생성합니다. CREATE ANY DIRECTORY 권한을 부여하지 않는 경우 수동으로 해당 디렉터리를 생성해야 합니다. 수동으로 Oracle 디렉터리를 만들 때 Oracle 소스 엔드포인트에 지정된 Oracle 사용자가 이러한 디렉터리를 만든 사용자가 아닌 경우도 있습니다. 이런 경우에도 READ on DIRECTORY 권한을 부여하세요.

Note

AWS DMS CDC는 자동 다시 실행 전송 서비스를 사용하도록 구성되지 않은 Active Dataguard Standby를 지원하지 않습니다.

경우에 따라 Oracle Managed Files(OMF)를 사용하여 로그를 저장할 수 있습니다. 또는 소스 엔드포인트가 ADG에 있고 따라서 CREATE ANY DIRECTORY 권한을 부여할 수 없습니다. 이러한 경우 AWS DMS 복제 작업을 시작하기 전에 가능한 모든 로그 위치를 사용하여 디렉터리를 수동으로 생성합니다. AWS DMS 가 예상한 미리 생성된 디렉터를 찾지 못하면 작업이 중지됩니다. 또한 AWS DMS 는 ALL_DIRECTORIES 뷰에서 생성한 항목을 삭제하지 않으므로 수동으로 삭제합니다.

Oracle ASM과 함께 Binary Reader를 사용할 때 필요한 추가 계정 권한

Binary Reader를 사용하여 ASM(Automatic Storage Management)의 다시 실행 로그에 액세스하려면 Oracle 엔드포인트 연결 설정에 지정된 Oracle 사용자에게 다음 권한을 부여합니다.

```
SELECT ON v_$transportable_platform
SYSASM -- To access the ASM account with Oracle 11g Release 2 (version 11.2.0.2) and
higher, grant the Oracle endpoint user the SYSASM privilege. For older supported
Oracle versions, it's typically sufficient to grant the Oracle endpoint user the
SYSDBA privilege.
```

명령 프롬프트를 열고 앞서 지정한 Oracle 버전에 따라 다음 명령문 중 하나를 간접적으로 호출하여 ASM 계정 액세스를 검증할 수 있습니다.

SYSDBA 권한이 필요한 경우 다음을 사용하세요.

```
sqlplus asmuser/asmpassword@asmserver as sysdba
```

SYSASM 권한이 필요한 경우 다음을 사용하세요.

```
sqlplus asmuser/asmpassword@asmserver as sysasm
```

에서 CDC용 Binary Reader를 사용하여 자체 관리형 Oracle Standby를 소스로 사용 AWS DMS

CDC용 Binary Reader를 사용할 때 Oracle Standby 인스턴스를 소스로 구성하려면 다음 사전 요구 사항부터 시작합니다.

- AWS DMS 는 현재 Oracle Active Data Guard Standby만 지원합니다.
- Oracle Data Guard 구성에서 다음을 사용하는지 확인하세요.
 - 다시 실행 데이터의 자동 전송을 위한 다시 실행 전송 서비스.
 - 대기 데이터베이스에 다시 실행을 자동으로 적용하는 서비스 적용.

이러한 요구 사항이 충족되는지 확인하려면 다음 쿼리를 실행합니다.

```
SQL> select open_mode, database_role from v$database;
```

해당 쿼리의 출력에서 대기 데이터베이스가 읽기 전용 모드로 열리고 다시 실행이 자동으로 적용되는지 확인합니다. 예제:

OPEN_MODE	DATABASE_ROLE
-----	-----
READ ONLY WITH APPLY	PHYSICAL STANDBY

CDC용 Binary Reader를 사용할 때 Oracle Standby 인스턴스를 소스로 구성하려면

1. 대기 로그 파일에 액세스하는 데 필요한 추가 권한을 부여합니다.

```
GRANT SELECT ON v_$standby_log TO dms_user;
```

2. AWS Management Console 또는 AWS CLI를 사용하여 Oracle Standby의 소스 엔드포인트를 생성합니다. 엔드포인트를 생성할 때 다음과 같은 추가 연결 속성을 지정합니다.

```
useLogminerReader=N;useBfile=Y;
```

Note

에서 추가 연결 속성을 사용하여 다시 실행 로그 대신 아카이브 로그에서 마이그레이션할 지 여부를 지정할 AWS DMS 수 있습니다. 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

3. 아카이브된 로그 대상을 구성합니다.

ASM이 없는 Oracle 소스용 DMS Binary Reader는 Oracle 디렉토리를 사용하여 아카이빙된 다시 실행 로그에 액세스합니다. 데이터베이스가 FRA(Fast Recovery Area)를 아카이브 로그 대상으로 사용하도록 구성된 경우, 아카이브 다시 실행 파일의 위치는 일정하지 않습니다. 아카이빙된 다시 실행 로그가 생성되는 날마다 FRA에 디렉터리 이름 형식 YYYY_MM_DD를 사용하여 새 디렉터리가 생성됩니다. 예제:

```
DB_RECOVERY_FILE_DEST/SID/archivelog/YYYY_MM_DD
```

DMS가 새로 생성된 FRA 디렉터리의 아카이빙된 다시 실행 파일에 액세스해야 하고 기본 읽기-쓰기 데이터베이스를 소스로 사용하는 경우, DMS는 다음과 같이 새 Oracle 디렉터리를 생성하거나 기존 Oracle 디렉터를 대체합니다.

```
CREATE OR REPLACE DIRECTORY dmsrep_taskid AS 'DB_RECOVERY_FILE_DEST/SID/archivelog/YYYY_MM_DD';
```

대기 데이터베이스를 소스로 사용하는 경우, 데이터베이스가 읽기 전용 모드이기 때문에 DMS는 Oracle 디렉터를 생성하거나 바꿀 수 없습니다. 하지만 다음 추가 단계 중 하나를 수행할 수 있습니다.

- a. Oracle이 일일 하위 디렉터를 생성하지 않는 구성에서는 FRA 대신 실제 경로를 사용하도록 `log_archive_dest_id_1`을 수정합니다.

```
ALTER SYSTEM SET log_archive_dest_1='LOCATION=full directory path'
```

그런 다음 DMS가 사용할 Oracle 디렉터리 객체를 생성합니다.

```
CREATE OR REPLACE DIRECTORY dms_archived_logs AS 'full directory path';
```

- b. 추가 아카이브 로그 대상과 해당 대상을 가리키는 Oracle 디렉터리 객체를 생성합니다. 예제:

```
ALTER SYSTEM SET log_archive_dest_3='LOCATION=full directory path';
CREATE DIRECTORY dms_archived_log AS 'full directory path';
```

그런 다음 작업 소스 엔드포인트에 추가 연결 속성을 추가합니다.

```
archivedLogDestId=3
```

- c. DMS가 사용할 Oracle 디렉터리 객체를 수동으로 사전 생성합니다.

```
CREATE DIRECTORY dms_archived_log_20210301 AS 'DB_RECOVERY_FILE_DEST/SID/
archivelog/2021_03_01';
CREATE DIRECTORY dms_archived_log_20210302 AS 'DB_RECOVERY_FILE_DEST>/SID>/
archivelog/2021_03_02';
...
```

- d. 매일 실행되고 필요한 디렉터리를 생성하는 Oracle 스케줄러 작업을 생성합니다.

4. 온라인 로그 대상을 구성합니다.

대기 재실행 로그를 사용하여 OS 디렉터리를 가리키는 Oracle 디렉터리를 생성합니다.

```
CREATE OR REPLACE DIRECTORY STANDBY_REDO_DIR AS '<full directory path>';
GRANT READ ON DIRECTORY STANDBY_REDO_DIR TO <dms_user>;
```

Oracle 클라우드 인프라(OCI)의 사용자 관리형 데이터베이스를의 CDC 소스로 사용 AWS DMS

사용자 관리형 데이터베이스는 가상 머신(VM), 베어 메탈 또는 Exadata 서버에서 만든 Oracle 데이터베이스와 같이 사용자가 구성하고 제어하는 데이터베이스입니다. 또는 OCI(Oracle Cloud Infrastructure)와 같은 전용 인프라에서 실행되고 사용자가 구성하고 제어하는 데이터베이스인 경우도 있습니다. 다음 정보는 AWS DMS에서 OCI의 Oracle 사용자 관리형 데이터베이스를 변경 데이터 캡처(CDC) 소스로 사용할 때 필요한 권한과 구성을 설명합니다.

OCI에서 호스팅하는 사용자 관리형 Oracle 데이터베이스를 변경 데이터 캡처의 소스로 구성하려면

- OCI의 사용자 관리형 Oracle 소스 데이터베이스에 필요한 사용자 계정 권한을 부여합니다. 자세한 내용은 [자체 관리형 Oracle 소스 엔드포인트에 대한 계정 권한](#)을 참조하세요.

2. Binary Reader를 사용하여 다시 실행 로그에 액세스할 때 필요한 계정 권한을 부여합니다. 자세한 내용은 [Binary Reader를 사용할 때 필요한 계정 권한](#)을 참조하세요.
3. Oracle ASM(Automatic Storage Management)에서 Binary Reader를 사용할 때 필요한 계정 권한을 추가합니다. 자세한 내용은 [Binary Reader를 Oracle ASM과 함께 사용할 때 필요한 추가 계정 권한](#)을 참조하세요.
4. 보충 로깅을 설정합니다. 자세한 내용은 [보충 로깅 설정](#)을 참조하세요.
5. TDE 암호화를 설정합니다. 자세한 내용은 [Oracle 데이터베이스를 소스 엔드포인트로 사용할 때 암호화 방법](#)을 참조하세요.

OCI(Oracle Cloud Infrastructure)의 Oracle 소스 데이터베이스에서 데이터를 복제할 때는 다음과 같은 제한 사항이 적용됩니다.

제한 사항

- DMS는 Oracle LogMiner를 사용한 다시 실행 로그 액세스를 지원하지 않습니다.
- DMS는 자율 DB를 지원하지 않습니다.

AWS의 소스로 관리형 Oracle 데이터베이스 작업 AWS DMS

AWS관리형 데이터베이스는 Amazon RDS, Amazon Aurora 또는 Amazon S3와 같은 Amazon 서비스에 있는 데이터베이스입니다. 아래에서 AWS관리형 Oracle 데이터베이스를 사용할 때 설정해야 하는 권한과 구성을 찾을 수 있습니다 AWS DMS.

용 AWS관리형 Oracle 소스에 필요한 사용자 계정 권한 AWS DMS

Oracle 소스 엔드포인트 정의에 지정된 Oracle 사용자 계정에 다음 권한을 부여합니다.

Important

*dms_user*와 *any-replicated-table* 같은 모든 파라미터 값에서 대소문자를 구분하는 식별자로 값을 지정하지 않는 한 Oracle은 값이 모두 대문자인 것으로 가정합니다. 예를 들어 CREATE USER *myuser* 또는 CREATE USER MYUSER에서처럼 따옴표를 사용하지 않고 *dms_user* 값을 생성한다고 가정해 보겠습니다. 이 경우 Oracle은 값을 모두 대문자(MYUSER)로 식별하고 저장합니다. CREATE USER "MyUser" 또는 CREATE USER 'MyUser'에서처럼 따옴표를 사용하면 Oracle은 사용자가 지정한 대소문자를 구별하는 값(MyUser)을 식별하고 저장합니다.

```
GRANT CREATE SESSION to dms_user;
GRANT SELECT ANY TRANSACTION to dms_user;
GRANT SELECT on DBA_TABLESPACES to dms_user;
GRANT SELECT ON any-replicated-table to dms_user;
GRANT EXECUTE on rdsadmin.rdsadmin_util to dms_user;
-- For Oracle 12c or higher:
GRANT LOGMINING to dms_user; - Required only if the Oracle version is 12c or higher.
```

또한 아래 나온 것처럼 Amazon RDS 절차 `rdsadmin.rdsadmin_util.grant_sys_object`를 사용하여 SYS 객체에 대한 SELECT 및 EXECUTE 권한을 부여합니다. 자세한 내용은 [SYS 객체에 SELECT 또는 EXECUTE 권한 부여](#)를 참조하세요.

```
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_VIEWS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_TAB_PARTITIONS', 'dms_user',
'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_INDEXES', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_OBJECTS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_TABLES', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_USERS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_CATALOG', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_CONSTRAINTS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_CONS_COLUMNS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_TAB_COLS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_IND_COLUMNS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_LOG_GROUPS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$ARCHIVED_LOG', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOG', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGFILE', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$DATABASE', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$THREAD', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$PARAMETER', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$NLS_PARAMETERS', 'dms_user',
'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$TIMEZONE_NAMES', 'dms_user',
'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$TRANSACTION', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$CONTAINERS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_REGISTRY', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('OBJ$', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_ENCRYPTED_COLUMNS', 'dms_user',
'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGMNR_LOGS', 'dms_user', 'SELECT');
```



```

exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGMNR_CONTENTS', 'dms_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_LOGMNR', 'dms_user', 'EXECUTE');

-- (as of Oracle versions 12.1 and higher)
exec rdsadmin.rdsadmin_util.grant_sys_object('REGISTRY$SQLPATCH', 'dms_user',
'SELECT');

-- (for Amazon RDS Active Dataguard Standby (ADG))
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$STANDBY_LOG', 'dms_user', 'SELECT');

-- (for transparent data encryption (TDE))

exec rdsadmin.rdsadmin_util.grant_sys_object('ENC$', 'dms_user', 'SELECT');

-- (for validation with LOB columns)
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_CRYPT0', 'dms_user', 'EXECUTE');

-- (for binary reader)
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_DIRECTORIES', 'dms_user', 'SELECT');

-- Required when the source database is Oracle Data guard, and Oracle Standby is used
in the latest release of DMS version 3.4.6, version 3.4.7, and higher.

exec rdsadmin.rdsadmin_util.grant_sys_object('V_$DATAGUARD_STATS', 'dms_user',
'SELECT');

```

AWS DMS 에서 Amazon RDS Active Dataguard Standby(ADG)를 사용하는 방법에 대한 자세한 내용은 [에서 CDC용 Binary Reader와 함께 Amazon RDS Oracle Standby\(읽기 전용 복제본\)를 소스로 사용 AWS DMS](#) 섹션을 참조하세요.

에서 Oracle TDE를 사용하는 방법에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [Oracle을의 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS](#).

Oracle Standby 열린 트랜잭션 처리를 위한 사전 요구 사항

AWS DMS 버전 3.4.6 이상을 사용하는 경우 다음 단계를 수행하여 Oracle Standby에 대한 열린 트랜잭션을 처리합니다.

1. 기본 데이터베이스에 AWSDMS_DBLINK라는 데이터베이스 링크를 생성합니다. **DMS_USER**는 이 데이터베이스 링크를 사용하여 기본 데이터베이스에 연결합니다. 이 데이터베이스 링크는 대기 인스턴스에서 실행되어 기본 데이터베이스에서 실행 중인 열린 트랜잭션을 쿼리합니다. 다음 예를 참조하세요.

```
CREATE PUBLIC DATABASE LINK AWS_DMS_DBLINK
CONNECT TO DMS_USER IDENTIFIED BY DMS_USER_PASSWORD
USING '(DESCRIPTION=
      (ADDRESS=(PROTOCOL=TCP)(HOST=PRIMARY_HOST_NAME_OR_IP)(PORT=PORT))
      (CONNECT_DATA=(SERVICE_NAME=SID)))
)';
```

2. 다음 예제와 같이 **DMS_USER**를 사용하여 데이터베이스 링크에 연결되었는지 확인합니다.

```
select 1 from dual@AWS_DMS_DBLINK
```

AWS에 대한 관리형 Oracle 소스 구성 AWS DMS

AWS관리형 Oracle 데이터베이스를 소스로 사용하기 전에 Oracle 데이터베이스에 대해 다음 작업을 AWS DMS수행합니다.

- 자동 백업을 활성화합니다. 자동 백업 활성화에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [자동 백업 활성화](#) 섹션을 참조하세요.
- 보충 로깅을 설정합니다.
- 보관을 설정합니다. Amazon RDS for Oracle DB 인스턴스에 대한 다시 실행 로그를 보관하면 AWS DMS 가 Oracle LogMiner 또는 Binary Reader를 사용하여 로그 정보를 검색할 수 있습니다.

보관을 설정하려면

1. `rdsadmin.rdsadmin_util.set_configuration` 명령을 실행하여 보관을 설정합니다.

예를 들어 보관된 다시 실행 로그를 24시간 동안 유지하려면 다음 명령을 실행합니다.

```
exec rdsadmin.rdsadmin_util.set_configuration('archivelog retention hours',24);
commit;
```

Note

변경 사항을 적용하려면 커밋해야 합니다.

2. 지정된 보존 기간 동안 보관된 다시 실행 로그를 위한 스토리지 공간이 충분한지 확인해야 합니다. 예를 들어 보존 기간이 24시간인 경우, 일반적인 트랜잭션 처리 시간 동안 보관된 다시 실행 로그

의 누적된 총 크기를 계산하고 그 합계에 24를 곱합니다. 계산된 이 24시간 합계를 사용 가능한 스토리지 공간과 비교하여 전체 24시간 트랜잭션 처리를 처리할 수 있는 충분한 스토리지 공간이 있는지 결정합니다.

보충 로깅을 설정하려면

1. 데이터베이스 수준에서 보충 로깅을 활성화하려면 다음 명령을 실행합니다.

```
exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD');
```

2. 다음 명령을 실행하여 프라이머리 키 보충 로깅을 활성화합니다.

```
exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD','PRIMARY KEY');
```

3. (선택 사항) 테이블 수준에서 키 수준 보충 로깅을 활성화합니다.

키 수준 보충 로깅을 활성화하면 소스 데이터베이스에 약간의 오버헤드가 발생합니다. 따라서 테이블 하위 집합만을 마이그레이션할 경우, 테이블 수준에서 키 수준 보충 로깅을 활성화하는 것이 좋습니다. 테이블 수준에서 키 수준 보충 로깅을 활성화하려면 다음 명령을 실행합니다.

```
alter table table_name add supplemental log data (PRIMARY KEY) columns;
```

용 RDS for Oracle 소스와 함께 Binary Reader를 사용하도록 CDC 작업 구성 AWS DMS

CDC용 Binary Reader를 사용하여 소스 Amazon RDS for Oracle 인스턴스 다시 실행 로그에 액세스 AWS DMS 하도록 구성할 수 있습니다.

Note

Oracle LogMiner를 사용하려면 최소 필수 사용자 계정 권한으로 충분합니다. 자세한 내용은 [용 AWS관리형 Oracle 소스에 필요한 사용자 계정 권한 AWS DMS](#) 단원을 참조하십시오.

AWS DMS Binary Reader를 사용하려면 버전에 AWS DMS 따라 Oracle 소스 엔드포인트에 대한 추가 설정 및 추가 연결 속성을 지정합니다.

Binary Reader는 다음 버전의 Amazon RDS for Oracle에서 지원됩니다.

- Oracle 11.2 – 버전 11.2.0.4V11 이상

- Oracle 12.1 – 버전 12.1.0.2.V7 이상
- Oracle 12.2 – 모든 버전
- Oracle 18.0 – 모든 버전
- Oracle 19.0 – 모든 버전

Binary Reader를 사용하여 CDC를 구성하려면

1. Amazon RDS for Oracle 소스 데이터베이스에 마스터 사용자로 로그인하고 다음 저장 프로시저를 실행하여 서버 수준 디렉터리를 생성합니다.

```
exec rdsadmin.rdsadmin_master_util.create_archivelog_dir;
exec rdsadmin.rdsadmin_master_util.create_onlinelog_dir;
```

2. Oracle 소스 엔드포인트에 액세스하는 데 사용되는 Oracle 사용자 계정에 다음 권한을 부여합니다.

```
GRANT READ ON DIRECTORY ONLINELOG_DIR TO dms_user;
GRANT READ ON DIRECTORY ARCHIVELOG_DIR TO dms_user;
```

3. Amazon RDS Oracle 소스 엔드포인트에서 다음 추가 연결 속성을 설정합니다.

- RDS Oracle 버전 11.2 및 12.1의 경우, 다음을 설정합니다.

```
useLogminerReader=N;useBfile=Y;accessAlternateDirectly=false;useAlternateFolderForOnline=
oraclePathPrefix=/rdsdbdata/db/{${DATABASE_NAME}}_A/;usePathPrefix=/rdsdbdata/
log/;replacePathPrefix=true;
```

- RDS Oracle 버전 12.2, 18.0, 19.0의 경우, 다음을 설정합니다.

```
useLogminerReader=N;useBfile=Y;
```

Note

여러 속성 설정의 경우, 세미콜론 구분자(;) 뒤에 공백이 오면 안 됩니다(예: oneSetting;thenAnother).

CDC 작업 구성에 대한 자세한 내용은 [Oracle 소스 데이터베이스에서의 CDC 구성](#) 섹션을 참조하세요.

에서 CDC용 Binary Reader와 함께 Amazon RDS Oracle Standby(읽기 전용 복제본)를 소스로 사용
AWS DMS

AWS DMS에서 CDC용 Binary Reader를 사용할 때 Amazon RDS for Oracle Standby를 소스로 사용하려면 다음 사전 요구 사항을 확인하세요.

- Oracle 마스터 사용자를 사용하여 Binary Reader를 설정합니다.
- 가 AWS DMS 현재 Oracle Active Data Guard Standby만 사용하도록 지원하는지 확인합니다.

그런 다음 CDC용 Binary Reader를 사용할 때 다음 절차를 사용하여 RDS for Oracle Standby를 소스로 사용합니다.

CDC용 Binary Reader를 사용할 때 RDS for Oracle Standby를 소스로 구성하려면

1. RDS for Oracle 기본 인스턴스에 마스터 사용자로 로그인합니다.
2. Amazon RDS 사용 설명서에 설명된 대로 다음 저장 프로시저를 실행하여 서버 수준 디렉터리를 생성합니다.

```
exec rdsadmin.rdsadmin_master_util.create_archivelog_dir;
exec rdsadmin.rdsadmin_master_util.create_onlinelog_dir;
```

3. 2단계에서 생성한 디렉터리를 식별합니다.

```
SELECT directory_name, directory_path FROM all_directories
WHERE directory_name LIKE ( 'ARCHIVELOG_DIR_%' )
      OR directory_name LIKE ( 'ONLINELOG_DIR_%' )
```

예를 들어 위 코드는 다음과 같은 디렉터리 목록을 표시합니다.

⚡ DIRECTORY_NAME	⚡ DIRECTORY_PATH
ARCHIVELOG_DIR_A	/rdsdbdata/db/ORCL_A/arch
ARCHIVELOG_DIR_B	/rdsdbdata/db/ORCL_B/arch
ONLINELOG_DIR_A	/rdsdbdata/db/ORCL_A/onlineolog
ONLINELOG_DIR_B	/rdsdbdata/db/ORCL_B/onlineolog

4. Oracle Standby에 액세스하는 데 사용되는 Oracle 사용자 계정에 이전 디렉터리에 대한 Read 권한을 부여합니다.

```
GRANT READ ON DIRECTORY ARCHIVELOG_DIR_A TO dms_user;
GRANT READ ON DIRECTORY ARCHIVELOG_DIR_B TO dms_user;
GRANT READ ON DIRECTORY ONLINELOG_DIR_A TO dms_user;
GRANT READ ON DIRECTORY ONLINELOG_DIR_B TO dms_user;
```

5. 기본 인스턴스에서 아카이브 로그 스위치를 수행합니다. 이렇게 하면 ALL_DIRECTORIES에 대한 변경 사항이 Oracle Standby에도 포팅되도록 할 수 있습니다.
6. Oracle Standby에서 ALL_DIRECTORIES 쿼리를 실행하여 변경 사항이 적용되었는지 확인합니다.
7. Management Console 또는 AWS Command Line Interface ()를 사용하여 Oracle Standby에 대한 소스 엔드포인트를 AWS DMS 생성합니다AWS CLI. 엔드포인트를 생성할 때 다음과 같은 추가 연결 속성을 지정합니다.

```
useLogminerReader=N;useBfile=Y;archivedLogDestId=1;additionalArchivedLogDestId=2
```

8. 엔드포인트를 생성한 후 콘솔의 엔드포인트 생성 페이지에서 엔드포인트 연결 테스트 또는 AWS CLI test-connection 명령을 사용하여 연결이 설정되었는지 확인합니다.

Oracle을의 소스로 사용할 때의 제한 사항 AWS DMS

AWS DMS용 소스로 Oracle 데이터베이스 사용 시 다음 제한 사항이 적용됩니다.

- AWS DMS 는 AWS DMS 버전 3.5.0 이상의 Oracle 확장 데이터 형식을 지원합니다.
- AWS DMS 는 긴 객체 이름(30바이트 이상)을 지원하지 않습니다.
- AWS DMS 는 함수 기반 인덱스를 지원하지 않습니다.
- 보충 로깅을 관리하고 임의의 열에서 변환을 수행하는 경우, 모든 필드 및 열에 대해 보충 로깅이 활성화되어 있어야 합니다. 보충 로깅 설정에 대한 자세한 내용은 다음 주제를 참조하세요.
 - 자체 관리형 Oracle 소스 데이터베이스는 [보충 로깅 설정](#) 섹션을 참조하세요.
 - AWS관리형 Oracle 소스 데이터베이스는 섹션을 참조하세요[AWS에 대한 관리형 Oracle 소스 구성 AWS DMS](#).
- AWS DMS 는 다중 테넌트 컨테이너 루트 데이터베이스(CDB\$ROOT)를 지원하지 않습니다. DMS는 Binary Reader를 사용하여 PDB를 지원합니다.
- AWS DMS 는 연기된 제약 조건을 지원하지 않습니다.
- AWS DMS 버전 3.5.3 이상은 보안 LOBs 완전히 지원합니다.

- AWS DMS 는 지원되는 모든 Oracle 버전 `rename table table-name to new-table-name` 11 이상에 대한 구문을 지원합니다. 이 구문은 Oracle 버전 10 소스 데이터베이스에서는 지원되지 않습니다.
- AWS DMS 는 DDL 문의 결과를 복제하지 않습니다 `ALTER TABLE ADD column data_type DEFAULT default_value. default_value`를 대상에 복제하는 대신 새 열을 NULL로 설정합니다.
- AWS DMS 버전 3.4.7 이상을 사용하는 경우 파티션 또는 하위 파티션 작업으로 인한 변경 사항을 복제하려면 DMS 작업을 시작하기 전에 다음을 수행합니다.
 - 파티셔닝된 테이블 구조(DDL)를 수동으로 생성합니다.
 - Oracle 소스와 Oracle 대상 모두에서 DDL이 동일해야 합니다.
 - 추가 연결 속성 `enableHomogenousPartitionOps=true`를 설정합니다.

`enableHomogenousPartitionOps`에 대한 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요. 또한 FULL+CDC 작업에서 DMS는 캐시된 변경 사항의 일부로 캡처된 데이터 변경 사항을 복제하지 않는다는 점에 유의하세요. 이 사용 사례에서는 Oracle 대상에서 테이블 구조를 다시 만들고 해당 테이블을 다시 로드하세요.

AWS DMS 버전 3.4.7 이전:

DMS는 파티션 또는 하위 파티션 작업(ADD, DROP, EXCHANGE, TRUNCATE)으로 인해 발생하는 데이터 변경 사항을 복제하지 않습니다. 이러한 업데이트로 인해 복제 중에 다음과 같은 오류가 발생할 수 있습니다.

- ADD 작업의 경우 추가된 데이터에 대한 업데이트 및 삭제로 인해 “적용된 행 없음” 경고가 발생할 수 있습니다.
- DROP 및 TRUNCATE 작업의 경우 새 삽입은 “중복” 오류를 발생시킬 수 있습니다.
- EXCHANGE 작업을 수행하면 “적용된 행 없음” 경고와 “중복” 오류가 발생할 수 있습니다.

파티션 또는 하위 파티션 작업으로 인해 발생하는 변경 사항을 복제하려면 해당 테이블을 다시 로드합니다. 빈 파티션을 새로 추가한 후 새로 추가된 파티션에 대한 작업이 정상적으로 대상에 복제됩니다.

- AWS DMS 3.4 이전 버전은 소스에서 CREATE TABLE AS 문을 실행하여 발생하는 대상의 데이터 변경을 지원하지 않습니다. 하지만, 새 테이블이 대상에서 생성됩니다.
- AWS DMS 는 테이블 메타데이터 및 OBJECT_ID 필드와 같은 Oracle DBMS_REDEFINITION 패키지의 변경 사항을 캡처하지 않습니다.
- 제한적인 크기의 LOB 모드가 활성화되면 Oracle 소스의 빈 BLOB/CLOB 열이 NULL 값으로 복제됩니다. 전체 LOB 모드가 활성화되면 빈 문자열(' ')로 복제됩니다.

- Oracle 11 LogMiner를 사용하여 변경 사항을 캡처할 경우 문자열 길이가 1982보다 큰 CLOB 열에 대한 업데이트가 손실되고 대상이 업데이트되지 않습니다.
- 변경 데이터 캡처(CDC) 중에는 AWS DMS 가 기본 키로 정의된 숫자 열에 대한 배치 업데이트를 지원하지 않습니다.
- AWS DMS 는 특정 UPDATE 명령을 지원하지 않습니다. 다음은 지원되지 않는 UPDATE 명령의 예입니다.

```
UPDATE TEST_TABLE SET KEY=KEY+1;
```

여기서, TEST_TABLE은 테이블 이름이고 KEY는 프라이머리 키로 정의 된 숫자 열입니다.

- AWS DMS 는 LONG 및 LONG RAW 열을 로드하기 위한 전체 LOB 모드를 지원하지 않습니다. 대신 제한된 LOB 모드를 사용하여 이러한 데이터 형식을 Oracle 대상으로 마이그레이션할 수 있습니다. 제한된 LOB 모드에서는 64KB보다 긴 LONG 또는 LONG RAW 열로 설정한 모든 데이터를 64KB로 AWS DMS 잘라냅니다.
- AWS DMS 는 XMLTYPE 열을 로드하기 위한 전체 LOB 모드를 지원하지 않습니다. 대신 제한된 LOB 모드를 사용하여 XMLTYPE 열을 Oracle 대상으로 마이그레이션할 수 있습니다. 제한된 LOB 모드에서 DMS는 사용자가 정의한 '최대 LOB 크기' 변수보다 큰 데이터를 모두 잘라냅니다. '최대 LOB 크기'의 최대 권장 값은 100MB입니다.
- AWS DMS 는 이름에 아포스트로피가 포함된 테이블을 복제하지 않습니다.
- AWS DMS 는 구체화된 뷰에서 CDC를 지원합니다. 하지만 DMS는 그 밖의 뷰에서는 CDC를 지원하지 않습니다.
- AWS DMS 는 오버플로 세그먼트가 있는 인덱스 구성 테이블에 대해 CDC를 지원하지 않습니다.
- AWS DMS 는가 로 enableHomogenousPartitionOps 설정된 참조로 분할된 테이블에 대한 Drop Partition 작업을 지원하지 않습니다true.
- Oracle LogMiner를 사용하여 다시 실행 로그에 액세스하는 경우 다음과 같은 제한 AWS DMS 사항이 있습니다.
 - Oracle 12의 경우에만는 LOB 열에 변경 사항을 복제하지 AWS DMS 않습니다.
 - AWS DMS 는 Oracle LogMiner를 사용하는 동안 복제 중인 XA 트랜잭션을 지원하지 않습니다.
 - Oracle LogMiner는 플러그형 데이터베이스(PDB)에 대한 연결을 지원하지 않습니다. PDB에 연결하려면 Binary Reader를 사용하여 다시 실행 로그에 액세스합니다.
 - SHRINK SPACE 작업은 지원되지 않습니다.
- Binary Reader를 사용하는 경우 AWS DMS 에는 다음과 같은 제한 사항이 있습니다.
 - 테이블 클러스터를 지원하지 않습니다.

- 테이블 수준 SHRINK SPACE 작업만 지원합니다. 이 수준에는 전체 테이블, 파티션, 하위 파티션이 포함됩니다.
- 키 압축을 사용하는 인덱스 구성 테이블에 대한 변경 사항은 지원하지 않습니다.
- 원시 디바이스에서 온라인 다시 실행 로그 구현은 지원하지 않습니다.
- RDS for Oracle이 TDE 암호화 키를 위한 지갑 암호 검색을 지원하지 않으므로 Binary Reader는 자체 관리형 Oracle 데이터베이스에서만 TDE를 지원합니다.
- AWS DMS 는 Oracle Automatic Storage Management(ASM) 프록시를 사용하여 Amazon RDS Oracle 소스에 대한 연결을 지원하지 않습니다.
- AWS DMS 는 가상 열을 지원하지 않습니다.
- AWS DMS 는 ROWID 열을 기반으로 하는 ROWID 데이터 유형 또는 구체화된 뷰를 지원하지 않습니다.

AWS DMS 는 Oracle 구체화된 뷰를 부분적으로 지원합니다. 전체 로드인 경우, DMS는 Oracle 구체화된 뷰의 전체 로드 복사를 수행할 수 있습니다. DMS는 구체화된 뷰를 대상 시스템에 기본 테이블로 복사하고 구체화된 뷰의 모든 ROWID 열을 무시합니다. 지속적 복제(CDC)의 경우 DMS는 구체화된 뷰 데이터에 대한 변경 사항을 복제하려고 시도하지만 결과는 이상적이지 않을 수 있습니다. 특히, 구체화된 뷰가 완전히 새로 고쳐지면 DMS는 모든 행에 개별 삭제를 복제한 후 모든 행에 개별 삽입을 복제합니다. 이는 리소스를 매우 많이 사용하는 작업이며, 행 수가 많은 구체화된 뷰에서는 성능이 떨어질 수 있습니다. 구체화된 뷰가 빠른 새로 고침을 수행하는 지속적인 복제의 경우, DMS는 빠른 새로 고침 데이터 변경을 처리하고 복제하려고 합니다. 어느 경우든 DMS는 구체화된 뷰의 ROWID 열을 모두 건너뜁니다.

- AWS DMS 는 글로벌 임시 테이블을 로드하거나 캡처하지 않습니다.
- 복제를 사용하는 S3 대상의 경우, 소스 행 업데이트가 모든 열 값을 캡처할 수 있도록 모든 열에서 보충 로깅을 활성화하세요. 예제는 다음과 같습니다. `alter table yourtablename add supplemental log data (all) columns;`
- null을 포함하는 복합 고유 키가 있는 행에 대한 업데이트는 대상에서 복제될 수 없습니다.
- AWS DMS 는 동일한 소스 엔드포인트에서 여러 Oracle TDE 암호화 키 사용을 지원하지 않습니다. 각 엔드포인트는 TDE 암호화 키 이름 “securityDbEncryptionName“에 대한 속성 하나와 이 키에 대한 하나의 TDE 암호만 가질 수 있습니다.
- Amazon RDS for Oracle에서 복제하는 경우, TDE는 암호화된 테이블스페이스와 Oracle LogMiner를 사용하는 경우에만 지원됩니다.
- AWS DMS 는 여러 테이블 이름 바꾸기 작업을 빠르게 연속해서 지원하지 않습니다.
- Oracle 19.0을 소스로 사용하는 경우 AWS DMS 는 다음 기능을 지원하지 않습니다.
 - 데이터가드 DML 리디렉션

- 파티셔닝된 하이브리드 테이블
- 스키마 전용 Oracle 계정
- AWS DMS 는 BIN\$ 또는 유형의 테이블 또는 뷰 마이그레이션을 지원하지 않습니다DR\$.
- Oracle 18.x부터 AWS DMS 는 Oracle Express Edition(Oracle Database XE)의 변경 데이터 캡처 (CDC)를 지원하지 않습니다.
- CHAR 열에서 데이터를 마이그레이션할 때 DMS는 모든 후행 공백을 잘라냅니다.
- AWS DMS 는 애플리케이션 컨테이너에서의 복제를 지원하지 않습니다.
- AWS DMS 는 Oracle Flashback 데이터베이스 및 복원 지점 수행을 지원하지 않습니다. 이러한 작업 은 Oracle Redo Log 파일의 일관성에 영향을 미치기 때문입니다.
- AWS DMS 버전 3.5.3 이전에는 다음과 같은 경우 병렬 실행 옵션이 있는 직접 로드 INSERT 절차가 지원되지 않습니다.
 - 열이 255개를 넘는 압축되지 않은 테이블
 - 행 크기가 8K를 초과
 - Exadata HCC 테이블
 - Big Endian 플랫폼에서 실행되는 데이터베이스
- 프라이머리 키도 고유 키도 없는 소스 테이블에는 ALL COLUMN 보충 로깅을 활성화해야 합니다. 이 로 인해 더 많은 다시 실행 로그 활동이 생성되고 DMS CDC 대기 시간이 늘어날 수 있습니다.
- AWS DMS 는 소스 데이터베이스의 보이지 않는 열에서 데이터를 마이그레이션하지 않습니다. 이러한 열을 마이그레이션 범위에 포함하려면 ALTER TABLE 문을 사용하여 이러한 열을 표시하세요.
- 모든 Oracle 버전에서는 XMLTYPE 및 LOB 열에 대한 UPDATE 작업 결과를 복제하지 AWS DMS 않습니다.
- AWS DMS 는 시간 유효성 제약 조건이 있는 테이블에서의 복제를 지원하지 않습니다.
- 전체 로드 작업 중에 Oracle 소스를 사용할 수 없게 되면 데이터 마이그레이션이 불완전하게 유지되더라도 여러 번의 재연결 시도 후 작업을 완료된 것으로 표시할 수 AWS DMS 있습니다. 이 시나리오에서 대상 테이블에는 연결 손실 전에 마이그레이션된 레코드만 포함되어 소스와 대상 시스템 간에 데이터 불일치가 발생할 수 있습니다. 데이터 완전성을 보장하려면 전체 로드 작업을 완전히 다시 시작하거나 연결 중단의 영향을 받는 특정 테이블을 다시 로드해야 합니다.

Oracle 엔드포인트용 SSL 지원

AWS DMS Oracle 엔드포인트는 none 및 SSL 모드에 대해 verify-ca SSL V3를 지원합니다.

Oracle 엔드포인트에서 SSL을 사용하려면, .pem 인증서 파일이 아닌 엔드포인트용 Oracle Wallet을 업로드합니다.

주제

- [Oracle SSL용 기존 인증서 사용](#)
- [Oracle SSL용 자체 서명 인증서 사용](#)

Oracle SSL용 기존 인증서 사용

기존 Oracle 클라이언트 설치를 사용하여 CA 인증서 파일에서 Oracle wallet 파일을 생성하려면, 다음 단계를 진행합니다.

에서 Oracle SSL에 기존 Oracle 클라이언트 설치를 사용하려면 AWS DMS

1. 다음 명령을 실행하여 ORACLE_HOME 시스템 변수를 dbhome_1 디렉터리 위치로 설정합니다.

```
prompt>export ORACLE_HOME=/home/user/app/user/product/12.1.0/dbhome_1
```

2. LD_LIBRARY_PATH 시스템 변수에 \$ORACLE_HOME/lib을 추가합니다.

```
prompt>export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib
```

3. \$ORACLE_HOME/ssl_wallet에 Oracle Wallet을 위한 디렉터를 만듭니다.

```
prompt>mkdir $ORACLE_HOME/ssl_wallet
```

4. CA 인증서 .pem 파일을 ssl_wallet 디렉터리에 배치합니다. Amazon RDS를 사용하는 경우, Amazon RDS에서 호스팅되는 rds-ca-2015-root.pem 루트 CA 인증서 파일을 다운로드할 수 있습니다. 이 파일 다운로드에 자세한 내용은 Amazon RDS 사용 설명서에서 [SSL/TLS를 사용하여 DB 인스턴스에 대한 연결 암호화](#)를 참조하세요.
5. CA 인증서에 둘 이상의 PEM 파일(예: Amazon RDS 글로벌 또는 리전 번들)이 포함된 경우 다음 bash 스크립트를 사용하여 별도의 파일로 분할하고 Oracle Wallet에 추가해야 합니다. 이 스크립트에는 CA 인증서 경로와 이전에 생성한 Oracle Wallet의 폴더 경로라는 두 가지 파라미터 입력이 필요합니다.

```
#!/usr/bin/env bash
```

```

certnum=$(grep -c BEGIN <(cat $1))

cnt=0
temp_cert=""
while read line
do
if [ -n "$temp_cert" -a "$line" == "-----BEGIN CERTIFICATE-----" ]
then
cnt=$(expr $cnt + 1)
printf "\rImporting certificate # $cnt of $certnum"
orapki wallet add -wallet "$2" -trusted_cert -cert <(echo -n "${temp_cert}") -
auto_login_only 1>/dev/null 2>/dev/null
temp_cert=""
fi
temp_cert+="$line"$'\n'
done < <(cat $1)

cnt=$(expr $cnt + 1)
printf "\rImporting certificate # $cnt of $certnum"
orapki wallet add -wallet "$2" -trusted_cert -cert <(echo -n "${temp_cert}") -
auto_login_only 1>/dev/null 2>/dev/null
echo ""

```

이전 단계를 완료했으면, certificate-wallet 매개변수를 지정하여 ImportCertificate API 직접 호출을 사용하여 wallet 파일을 가져올 수 있습니다. 그런 다음 Oracle 엔드포인트를 생성하거나 수정할 때 SSL 모드로서 verify-ca를 선택하면 가져온 wallet 인증서를 사용할 수 있습니다.

Note

Oracle Wallet은 바이너리 파일입니다. AWS DMS는 이러한 파일을 있는 그대로 허용합니다.

Oracle SSL용 자체 서명 인증서 사용

Oracle SSL용 자체 서명 인증서를 사용하려면 Oracle wallet 암호가 oracle123이라고 가정하고 다음 단계를 수행하세요.

에서 Oracle SSL에 자체 서명된 인증서를 사용하려면 AWS DMS

1. 자체 서명 인증서를 사용할 디렉터리를 생성합니다.

```
mkdir -p /u01/app/oracle/self_signed_cert
```

2. 이전 단계에서 생성한 디렉터리로 변경합니다.

```
cd /u01/app/oracle/self_signed_cert
```

3. 루트 키를 생성합니다.

```
openssl genrsa -out self-rootCA.key 2048
```

4. 이전 단계에서 생성한 루트 키를 루트 인증서를 자체 서명합니다.

```
openssl req -x509 -new -nodes -key self-rootCA.key  
-sha256 -days 3650 -out self-rootCA.pem
```

다음과 같은 입력 파라미터를 사용하세요.

- Country Name (2 letter code) [XX], 예: AU
- State or Province Name (full name) [], 예: NSW
- Locality Name (e.g., city) [Default City], 예: Sydney
- Organization Name (e.g., company) [Default Company Ltd], 예: AmazonWebService
- Organizational Unit Name (e.g., section) [], 예: DBeng
- Common Name (e.g., your name or your server's hostname) [], 예: aws
- Email Address [], 예: abcd.efgh@amazonweb-service.com

5. Oracle 데이터베이스용 Oracle wallet 디렉터리를 만듭니다.

```
mkdir -p /u01/app/oracle/wallet
```

6. 새 Oracle wallet을 만듭니다.

```
orapki wallet create -wallet "/u01/app/oracle/wallet" -pwd oracle123 -  
auto_login_local
```

7. 루트 인증서를 Oracle wallet에 추가합니다.

```
orapki wallet add -wallet "/u01/app/oracle/wallet" -pwd oracle123 -trusted_cert
```

```
-cert /u01/app/oracle/self_signed_cert/self-rootCA.pem
```

8. Oracle wallet의 콘텐츠를 나열합니다. 이 목록에는 루트 인증서가 포함되어야 합니다.

```
orapki wallet display -wallet /u01/app/oracle/wallet -pwd oracle123
```

예를 들어 다음과 비슷하게 표시될 수 있습니다.

```
Requested Certificates:
User Certificates:
Trusted Certificates:
Subject:          CN=aws,OU=DBEng,O= AmazonWebService,L=Sydney,ST=NSW,C=AU
```

9. ORAPKI 유틸리티를 사용한 인증서 서명 요청(CSR)을 생성합니다.

```
orapki wallet add -wallet "/u01/app/oracle/wallet" -pwd oracle123
-dn "CN=aws" -keysize 2048 -sign_alg sha256
```

10. 다음 명령을 실행합니다.

```
openssl pkcs12 -in /u01/app/oracle/wallet/ewallet.p12 -nodes -out /u01/app/oracle/
wallet/nonoracle_wallet.pem
```

이 명령의 출력은 다음과 같습니다.

```
Enter Import Password:
MAC verified OK
Warning unsupported bag type: secretBag
```

11. 'dms'를 일반 이름으로 둡니다.

```
openssl req -new -key /u01/app/oracle/wallet/nonoracle_wallet.pem -out certdms.csr
```

다음과 같은 입력 파라미터를 사용하세요.

- Country Name (2 letter code) [XX], 예: AU
- State or Province Name (full name) [], 예: NSW
- Locality Name (e.g., city) [Default City], 예: Sydney

- Organization Name (e.g., company) [Default Company Ltd], 예: AmazonWebService
- Organizational Unit Name (e.g., section) [], 예: aws
- Common Name (e.g., your name or your server's hostname) [], 예: aws
- Email Address [], 예: abcd.efgh@amazonweb-service.com

이 단계는 4단계와 달라야 합니다. 예를 들어 다음과 같이 조직 단위 이름을 다른 이름으로 변경하면 됩니다.

인증서 요청과 함께 전송될 다음 추가 속성을 입력합니다.

- A challenge password [], 예: oracle123
- An optional company name [], 예: aws

12. 인증서 서명을 가져옵니다.

```
openssl req -noout -text -in certdms.csr | grep -i signature
```

이 게시물의 서명 키는 sha256WithRSAEncryption입니다.

13. 다음 명령을 실행하여 인증서(.crt) 파일을 생성합니다.

```
openssl x509 -req -in certdms.csr -CA self-rootCA.pem -CAkey self-rootCA.key  
-CAcreateserial -out certdms.crt -days 365 -sha256
```

그러면 다음과 같은 출력이 표시됩니다.

```
Signature ok  
subject=/C=AU/ST=NSW/L=Sydney/O=awsweb/OU=DBGeng/CN=aws  
Getting CA Private Key
```

14. 인증서를 wallet에 추가합니다.

```
orapki wallet add -wallet /u01/app/oracle/wallet -pwd oracle123 -user_cert -cert  
certdms.crt
```

15. wallet을 봅니다. 두 항목이 있어야 합니다. 다음 코드를 확인합니다.

```
orapki wallet display -wallet /u01/app/oracle/wallet -pwd oracle123
```

16. sqlnet.ora 파일(\$ORACLE_HOME/network/admin/sqlnet.ora)을 구성합니다.

```
WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = /u01/app/oracle/wallet/)
    )
  )

SQLNET.AUTHENTICATION_SERVICES = (NONE)
SSL_VERSION = 1.0
SSL_CLIENT_AUTHENTICATION = FALSE
SSL_CIPHER_SUITES = (SSL_RSA_WITH_AES_256_CBC_SHA)
```

17. Oracle 리스너를 중지합니다.

```
lsnrctl stop
```

18. listener.ora 파일(\$ORACLE_HOME/network/admin/listener.ora)에 SSL에 대한 항목을 추가합니다.

```
SSL_CLIENT_AUTHENTICATION = FALSE
WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = /u01/app/oracle/wallet/)
    )
  )

SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (GLOBAL_DBNAME = SID)
      (ORACLE_HOME = ORACLE_HOME)
      (SID_NAME = SID)
    )
  )

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
```



```

        (ADDRESS = (PROTOCOL = TCP)(HOST = localhost.localdomain)(PORT = 1521))
        (ADDRESS = (PROTOCOL = TCPS)(HOST = localhost.localdomain)(PORT = 1522))
        (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
    )
)

```

19. tnsnames.ora 파일(\$ORACLE_HOME/network/admin/tnsnames.ora)을 구성합니다.

```

<SID>=
(DESCRIPTION=
    (ADDRESS_LIST =
        (ADDRESS=(PROTOCOL = TCP)(HOST = localhost.localdomain)(PORT =
1521))
    )
    (CONNECT_DATA =
        (SERVER = DEDICATED)
        (SERVICE_NAME = <SID>)
    )
)

<SID>_ssl=
(DESCRIPTION=
    (ADDRESS_LIST =
        (ADDRESS=(PROTOCOL = TCPS)(HOST = localhost.localdomain)(PORT =
1522))
    )
    (CONNECT_DATA =
        (SERVER = DEDICATED)
        (SERVICE_NAME = <SID>)
    )
)

```

20. Oracle 리스너를 다시 시작합니다.

```
lsnrctl start
```

21. Oracle 리스너 상태를 표시합니다.

```
lsnrctl status
```

22. sqlplus 및 SSL tnsnames 항목을 사용하여 로컬 호스트에서 데이터베이스와의 SSL 연결을 테스트합니다.

```
sqlplus -L ORACLE_USER@SID_ssl
```

23. SSL을 사용하여 잘 연결되었는지 확인합니다.

```
SELECT SYS_CONTEXT('USERENV', 'network_protocol') FROM DUAL;

SYS_CONTEXT('USERENV', 'NETWORK_PROTOCOL')
-----
tcps
```

24. 자체 서명 인증서를 사용하여 디렉토리를 이 디렉터리로 변경합니다.

```
cd /u01/app/oracle/self_signed_cert
```

25. 에서 사용할 새 클라이언트 Oracle Wallet AWS DMS 을 생성합니다.

```
orapki wallet create -wallet ./ -auto_login_only
```

26. Oracle wallet에 자체 서명 루트 인증서를 추가합니다.

```
orapki wallet add -wallet ./ -trusted_cert -cert self-rootCA.pem -auto_login_only
```

27. 에서 사용할 Oracle Wallet의 내용을 나열 AWS DMS 합니다. 이 목록에는 자체 서명 루트 인증서가 포함되어야 합니다.

```
orapki wallet display -wallet ./
```

이 명령의 출력은 다음과 같습니다.

```
Trusted Certificates:
Subject:          CN=aws,OU=DBeng,O=AmazonWebService,L=Sydney,ST=NSW,C=AU
```

28. 방금 생성한 Oracle Wallet을 업로드합니다 AWS DMS.

Oracle을의 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS

다음 표에서는가 Oracle 소스 데이터베이스로 작업할 때 AWS DMS 지원하는 투명한 데이터 암호화 (TDE) 방법을 확인할 수 있습니다.

다시 실행 로그 액세스 방법	TDE 테이블스페이스	TDE 열
Oracle LogMiner	예	예
Binary Reader	예	예

AWS DMS 는 열 수준과 테이블스페이스 수준 모두에서 Binary Reader를 사용할 때 Oracle TDE를 지원합니다. 에서 TDE 암호화를 사용하려면 AWS DMS 먼저 TDE 암호화 키와 TDE 암호가 저장되는 Oracle Wallet 위치를 식별합니다. 그런 다음 Oracle 소스 엔드포인트의 올바른 TDE 암호화 키와 암호를 식별합니다.

TDE 암호화를 위한 암호화 키와 암호를 식별하고 지정하려면

1. 다음 쿼리를 실행하여 Oracle 데이터베이스 호스트에서 Oracle 암호화 wallet을 찾습니다.

```
SQL> SELECT WRL_PARAMETER FROM V$ENCRYPTION_WALLET;
```

```
WRL_PARAMETER
```

```
-----  
/u01/oracle/product/12.2.0/dbhome_1/data/wallet/
```

여기서는 /u01/oracle/product/12.2.0/dbhome_1/data/wallet/이 wallet 위치입니다.

2. 다음과 같이 비CDB 또는 CDB 소스의 마스터 키 ID를 가져옵니다.

- a. 비 CDB 소스의 경우 다음 쿼리를 실행하여 마스터 암호화 키 ID를 검색합니다.

```
SQL> select rownum, key_id, activation_time from v$encryption_keys;
```

```
ROWNUM KEY_ID ACTIVATION_TIME
-----
```

1	AeKask0XZU+Nvysf1CYBEVwAAAAAAAAAAAAAAAAAAAAAAAAAAAA	04-SEP-24
10.20.56.605200 PM +00:00		
2	AV7WU9uhoU8rv8daE/HNnSwAAAAAAAAAAAAAAAAAAAAAAAAAAAA	10-AUG-21
07.52.03.966362 PM +00:00		
3	AckpoJ/f+k8xvzJ+gSuoVH4AAAAAAAAAAAAAAAAAAAAAAAAAAAA	14-SEP-20
09.26.29.048870 PM +00:00		

활성화 시간은 과거 어느 시점에서 CDC를 시작하려는 경우에 유용합니다. 예를 들어 위의 결과를 사용하면 ROWNUM 10-08-21일 오후 7.52.03부터 2014-09-20 오후 09.26.29까지 CDC

를 시작할 수 있습니다. 작업이 09.26.29년 14-09-20 오후 또는 그 이후에 생성된 재실행에 도달하면 실패하므로 소스 엔드포인트를 수정하고 ROWNUM 3에 마스터 키 ID를 입력한 다음 작업을 재개해야 합니다.

- b. CDB 소스 DMS의 경우 CDB\$ROOT 마스터 암호화 키가 필요합니다. CDB\$ROOT에 연결하고 다음 쿼리를 실행합니다.

```
SQL> select rownum, key_id, activation_time from v$encryption_keys where con_id
= 1;
```

ROWNUM	KEY_ID	ACTIVATION_TIME
1	Aa2E/Vwb5U+zv5hCncS5ErMAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	29-AUG-24
	12.51.19.699060 AM +00:00	

3. 명령줄을 사용하여 소스 Oracle 데이터베이스 호스트에서 암호화 wallet 항목을 나열합니다.

```
$ mkstore -wrl /u01/oracle/product/12.2.0/dbhome_1/data/wallet/ -list
Oracle Secret Store entries:
ORACLE.SECURITY.DB.ENCRYPTION.AWGDC9g1Sk8Xv+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.DB.ENCRYPTION.AY1mRA80XU9Qvzo3idU40H4AAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.DB.ENCRYPTION.MASTERKEY
ORACLE.SECURITY.ID.ENCRYPTION.
ORACLE.SECURITY.KB.ENCRYPTION.
ORACLE.SECURITY.KM.ENCRYPTION.AY1mRA80XU9Qvzo3idU40H4AAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

2단계에서 찾은 마스터 키 ID(AWGDC9g1Sk8Xv+3bVveiVSg)가 포함된 항목을 찾습니다. 이 항목이 TDE 암호화 키 이름입니다.

4. 이전 단계에서 찾은 항목의 세부 정보를 확인합니다.

```
$ mkstore -wrl /u01/oracle/product/12.2.0/dbhome_1/data/wallet/ -viewEntry
ORACLE.SECURITY.DB.ENCRYPTION.AWGDC9g1Sk8Xv+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Oracle Secret Store Tool : Version 12.2.0.1.0
Copyright (c) 2004, 2016, Oracle and/or its affiliates. All rights reserved.
Enter wallet password:
ORACLE.SECURITY.DB.ENCRYPTION.AWGDC9g1Sk8Xv+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAAA
= AEMAASAASGYs0phWHfNt9J5mEMkkegGFid4LLfQszDojgDzbfoYDEACv0x3pJC+UGD/
PdtE2jLIcBQcAeHgJChQGLA==
```

결과를 보려면 wallet 암호를 입력합니다.

여기서는 '='의 오른쪽에 있는 값이 TDE 암호입니다.

5. securityDbEncryptionName 추가 연결 속성을 설정하여 Oracle 소스 엔드포인트의 TDE 암호화 키 이름을 지정합니다.

```
securityDbEncryptionName=ORACLE.SECURITY.DB.ENCRIPTION.AWGDC9g1Sk8Xv
+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

6. 콘솔에서 이 키에 연결된 TDE 암호를 Oracle 소스 암호 값의 일부로 제공합니다. TDE 암호 값으로 끝나는 쉼표로 구분된 암호 값의 형식을 지정하려면 다음 순서를 사용합니다.

```
Oracle_db_password,ASM_Password,AEMAASAAGYs0phWHfNt9J5mEMkkegGFid4LLfQszDojgDzbfoYDEACv0x3
+UGD/PdtE2jLIcBQcAeHgJChQGLA==
```

Oracle 데이터베이스 구성에 관계없이 이 순서로 암호 값을 지정합니다. 예를 들어 TDE를 사용하고 있지만 Oracle 데이터베이스가 ASM을 사용하지 않는 경우, 암호 값을 다음과 같은 쉼표로 구분된 순서로 지정합니다.

```
Oracle_db_password,,AEMAASAAGYs0phWHfNt9J5mEMkkegGFid4LLfQszDojgDzbfoYDEACv0x3pJC
+UGD/PdtE2jLIcBQcAeHgJChQGLA==
```

지정한 TDE 자격 증명이 올바르지 않으면 AWS DMS 마이그레이션 작업이 실패하지 않습니다. 하지만 이 작업은 진행 중인 대상 데이터베이스에 대한 복제 변경 사항을 읽거나 적용하지도 않습니다. 작업을 시작한 후에 콘솔 마이그레이션 작업 페이지에서 테이블 통계를 모니터링하여 변경 사항이 복제되었는지 확인하세요.

작업이 실행되는 동안 DBA가 Oracle 데이터베이스의 TDE 자격 증명 정보 값을 변경하면 작업이 실패합니다. 오류 메시지에는 새 TDE 암호화 키 이름이 포함됩니다. 새 값을 지정하고 작업을 다시 시작하려면 이전 절차를 사용하세요.

Important

cp, mv, orapki, mkstore 같은 OS 수준 명령으로 인해 ASM 위치에 저장된 wallet 파일이 손상되므로 Oracle ASM(Automatic Storage Management) 위치에 생성된 TDE wallet을 조작할 수 없습니다. 이 제한은 ASM 위치에 저장된 TDE wallet 파일에만 적용되며, 로컬 OS 디렉터리에 저장된 TDE wallet 파일에는 적용되지 않습니다.

OS 수준 명령으로 ASM에 저장된 TDE wallet을 조작하려면 다음과 같이 로컬 키스토어를 생성하고 ASM 키스토어를 로컬 키스토어에 병합하세요.

1. 로컬 키스토어를 생성합니다.

```
ADMINISTER KEY MANAGEMENT create keystore file system wallet location
identified by wallet password;
```

2. ASM 키스토어를 로컬 키스토어에 병합합니다.

```
ADMINISTER KEY MANAGEMENT merge keystore ASM wallet location identified
by wallet password into existing keystore file system wallet location
identified by wallet password with backup;
```

그런 다음 암호화 wallet 항목과 TDE 암호를 나열하려면 로컬 키스토어에 대해 3단계와 4단계를 실행합니다.

Oracle을의 소스로 사용하는 데 지원되는 압축 방법 AWS DMS

다음 표에서는 Oracle 소스 데이터베이스로 작업할 때에서 AWS DMS 지원하는 압축 방법을 확인할 수 있습니다. 표에서 볼 수 있듯이 압축 지원은 Oracle 데이터베이스 버전과 DMS가 Oracle LogMiner를 사용하여 다시 실행 로그에 액세스하도록 구성되어 있는지 여부에 따라 달라집니다.

버전	기본	OLTP	HCC(Oracle 11g R2 이상부터)	기타
Oracle 10	아니요	해당 사항 없음	해당 사항 없음	아니요
Oracle 11 이상 – Oracle LogMiner	예	예	예	예 - Oracle LogMiner가 지원하는 모든 압축 방법.
Oracle 11 이상 – Binary Reader	예	예	예 - 자세한 내용은 다음 참고 사항을 참조하세요.	예

Note

Oracle 소스 엔드포인트가 Binary Reader를 사용하도록 구성된 경우 HCC 압축 방법의 Query Low 수준은 전체 로드 작업에 대해서만 지원됩니다.

Oracle을의 소스로 사용하여 중첩 테이블 복제 AWS DMS

AWS DMS 는 중첩 테이블 또는 정의된 유형인 열이 포함된 Oracle 테이블의 복제를 지원합니다. 이 기능을 활성화하려면 Oracle 소스 엔드포인트에 다음 추가 연결 속성 설정을 추가합니다.

```
allowSelectNestedTables=true;
```

AWS DMS 는 Oracle 중첩 테이블의 대상 테이블을 고유한 제약 조건 없이 대상의 일반 상위 및 하위 테이블로 생성합니다. 대상에서 올바른 데이터에 액세스하려면 상위 및 하위 테이블을 조인합니다. 이렇게 하려면 먼저 대상 하위 테이블의 NESTED_TABLE_ID 열에 고유하지 않은 인덱스를 수동으로 만듭니다. 그런 다음 조인 ON 절의 NESTED_TABLE_ID 열을 자식 테이블 이름에 해당하는 상위 열과 함께 사용할 수 있습니다. 또한 이러한 인덱스를 생성하면 대상 하위 테이블 데이터가 업데이트되거나 삭제될 때 성능이 향상됩니다 AWS DMS. 예제는 [대상의 상위 및 하위 테이블에 대한 조인 예](#) 섹션을 참조하세요.

전체 로드가 완료되면 작업을 중지하도록 구성하는 것이 좋습니다. 그런 다음 대상의 복제된 모든 하위 테이블에 대해 고유하지 않은 인덱스를 만들고 작업을 재개합니다.

캡처된 중첩 테이블이 기존 상위 테이블(캡처됨 또는 캡처되지 않음)에 추가되면가 이를 올바르게 AWS DMS 처리합니다. 그러나 해당 대상 테이블에 대한 고유하지 않은 인덱스는 생성되지 않습니다. 이 경우 대상 하위 테이블이 매우 커지면 성능이 영향을 받을 수 있습니다. 이런 경우에는 작업을 중지하고 인덱스를 만든 다음 작업을 재개하는 것이 좋습니다.

중첩 테이블이 대상에 복제된 후 DBA가 상위 및 해당 하위 테이블에 대한 조인을 실행하여 데이터를 병합하도록 합니다.

Oracle 중첩 테이블을 소스로 복제하기 위한 사전 요구 사항

복제된 모든 중첩 테이블의 상위 테이블을 복제해야 합니다. 테이블 매핑에 상위 테이블(중첩된 테이블 열을 포함하는 테이블)과 하위 AWS DMS 테이블(즉, 중첩된 테이블)을 모두 포함합니다.

소스로 지원되는 Oracle 중첩 테이블 유형

AWS DMS 는 다음과 같은 Oracle 중첩 테이블 유형을 소스로 지원합니다.

- 데이터 유형
- 사용자 정의 객체

소스로서 Oracle 중첩 테이블에 대한 AWS DMS 지원 제한 사항

AWS DMS에는 Oracle 중첩 테이블을 소스로 지원하는 데 다음과 같은 제한 사항이 있습니다.

- AWS DMS는 한 수준의 테이블 중첩만 지원합니다.
- AWS DMS 테이블 매핑은 복제를 위해 상위 테이블과 하위 테이블이 모두 선택되었는지 확인하지 않습니다. 즉, 하위 테이블이 없는 상위 테이블 또는 상위 테이블이 없는 하위 테이블을 선택할 수 있습니다.

가 Oracle 중첩 테이블을 소스로 AWS DMS 복제하는 방법

AWS DMS는 다음과 같이 상위 및 중첩 테이블을 대상에 복제합니다.

- AWS DMS는 원본과 동일한 상위 테이블을 생성합니다. 그런 다음 상위 테이블의 중첩 열을 RAW(16)로 정의하고 상위 테이블의 중첩 테이블에 대한 참조를 해당 NESTED_TABLE_ID 열에 포함합니다.
- AWS DMS는 중첩 소스와 동일하지만 라는 추가 열을 사용하여 하위 테이블을 생성합니다 NESTED_TABLE_ID. 이 열은 해당 상위 중첩 열과 동일한 유형 및 값을 가지며 동일한 의미를 갖습니다.

대상의 상위 및 하위 테이블에 대한 조인 예

상위 테이블을 병합하려면 다음 예제와 같이 상위 테이블과 하위 테이블 간에 조인을 실행합니다.

1. Type 테이블을 만듭니다.

```
CREATE OR REPLACE TYPE NESTED_TEST_T AS TABLE OF VARCHAR(50);
```

2. 앞서 정의한 유형 NESTED_TEST_T의 열을 사용하여 상위 테이블을 만듭니다.

```
CREATE TABLE NESTED_PARENT_TEST (ID NUMBER(10,0) PRIMARY KEY, NAME NESTED_TEST_T)
  NESTED TABLE NAME STORE AS NAME_KEY;
```

3. CHILD.NESTED_TABLE_ID가 PARENT.NAME과 일치하는 NAME_KEY 하위 테이블과의 조인을 사용하여 테이블 NESTED_PARENT_TEST를 병합합니다.


```
SELECT ... FROM NESTED_PARENT_TEST PARENT, NAME_KEY CHILD WHERE CHILD.NESTED_
TABLE_ID = PARENT.NAME;
```

Oracle을의 소스로 사용할 때 Oracle ASM에 REDO 저장 AWS DMS

REDO가 많이 생성되는 Oracle 소스의 경우, Oracle ASM에 REDO를 저장하면 특히 RAC 구성에서 성능이 향상될 수 있습니다. ASM REDO 읽기를 모든 ASM 노드에 분산하도록 DMS를 구성할 수 있기 때문입니다.

이 구성을 활용하려면 `asmServer` 연결 속성을 사용하세요. 예를 들어 다음 연결 문자열은 DMS REDO 읽기를 ASM 노드 3개에 분산합니다.

```
asmServer=(DESCRIPTION=(CONNECT_TIMEOUT=8)(ENABLE=BROKEN)(LOAD_BALANCE=ON)(FAILOVER=ON)
(ADDRESS_LIST=
(ADDRESS=(PROTOCOL=tcp)(HOST=asm_node1_ip_address)(PORT=asm_node1_port_number))
(ADDRESS=(PROTOCOL=tcp)(HOST=asm_node2_ip_address)(PORT=asm_node2_port_number))
(ADDRESS=(PROTOCOL=tcp)(HOST=asm_node3_ip_address)(PORT=asm_node3_port_number)))
(CONNECT_DATA=(SERVICE_NAME=+ASM)))
```

NFS를 사용하여 Oracle REDO를 저장하는 경우, 해당 DNFS(Direct NFS) 클라이언트 패치, 특히 Oracle 버그 25224242를 해결하는 모든 패치를 적용해야 합니다. 자세한 내용은 Direct NFS 클라이언트 관련 패치에 관한 다음 Oracle 간행물, [Recommended Patches for Direct NFS Client](#)를 참조하세요.

또한 NFS 읽기 성능을 향상시키려면 다음 예와 같이 NFS 볼륨의 `fstab`에서 `rsz` 및 `wsz`의 값을 늘리는 것이 좋습니다.

```
NAS_name_here:/ora_DATA1_archive /u09/oradata/DATA1 nfs
rw,bg,hard,nointr,tcp,nfsvers=3,_netdev,
timeo=600,rsz=262144,wsz=262144
```

또한 다음과 같이 `tcp-max-xfer-size` 값을 조정합니다.

```
vserver nfs modify -vserver vserver -tcp-max-xfer-size 262144
```

Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Oracle 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 `--oracle-`

settings '{"*EndpointSetting*": "*value*", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

Oracle을 소스로 하여 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

이름	설명
AccessAlternateDirectly	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하기 위해서는 이 속성을 false로 설정합니다. 그러면 DMS 인스턴스가 직접 파일 액세스를 사용하여 지정된 경로 접두사 교체를 통해 다시 실행 로그에 액세스하지 않습니다. 자세한 내용은 용 RDS for Oracle 소스와 함께 Binary Reader를 사용하도록 CDC 작업 구성 AWS DMS 단원을 참조하십시오. 기본값: true 유효값: true/false 예시: --oracle-settings '{"AccessAlternateDirectly": false}'
AdditionalArchivedLogDestId	프라이머리-스탠바이 설정에서 ArchivedLogDestId 로 이 속성을 설정합니다. 이 설정은 Oracle Data Guard 데이터베이스가 소스로 사용될 때 전환에 유용합니다. 이 경우 AWS DMS 는 변경 사항을 읽기 위해 아카이브 다시 실행 로그를 가져올 대상을 알아야 합니다. 이것은 전환 후에 이전 프라이머리가 이제는 대기 인스턴스이기 때문입니다. 는 Oracle RESETLOGS 옵션을 사용하여 데이터베이스를 열 수 있도록 AWS DMS 지원하지만 필요한 RESETLOGS 경우가 아니면 사용하지 마십시오. RESETLOGS 에 대한 자세한 내용을 알아보려면 Oracle® Database Backup and Recovery User's Guide의 RMAN Data Repair Concepts를 참조하세요. 유효한 값: 아카이브 대상 ID 예시: --oracle-settings '{"AdditionalArchivedLogDestId": 2}'

이름	설명
AddSupplementalLogging	이 속성을 설정하면 Oracle 데이터베이스의 테이블 수준 보충 로깅을 설정할 수 있습니다. 이 속성은 테이블 메타데이터에 따라 마이그레이션 작업을 위해 선택한 모든 테이블에서 다음 중 하나를 활성화합니다. • PRIMARY KEY COLUMNS 보충 로깅 • UNIQUE KEY COLUMNS 보충 로깅 • ALL COLUMNS 보충 로깅 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"AddSupplementalLogging": false}'</code>  Note 이 옵션을 사용할 경우, 앞에서 설명한 대로 데이터베이스 수준 보충 로깅을 계속 활성화해야 합니다.
AllowSelectNestedTables	이 속성을 true로 설정하면 중첩 테이블 또는 정의된 유형인 열을 포함하는 Oracle 테이블의 복제를 활성화합니다. 자세한 내용은 Oracle을의 소스로 사용하여 중첩 테이블 복제 AWS DMS 단원을 참조하십시오. 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"AllowSelectNestedTables": true}'</code>

이름	설명
ArchivedLogDestId	보관된 다시 실행 로그의 대상 ID를 지정합니다. 이 값은 v\$archived_log 보기의 dest_id 열의 숫자와 같아야 합니다. 추가 다시 실행 로그 대상으로 작업하는 경우, AdditionalArchivedLogDestId 속성을 사용하여 추가 대상 ID를 지정하는 것이 좋습니다. 그러면 시작 단계부터 올바른 로그에 액세스하도록 보장하여 성능이 개선됩니다. 기본값: 1 유효한 값: 숫자 예시: --oracle-settings '{"ArchivedLogDestId": 1}'
ArchivedLogsOnly	이 필드를 Y로 설정하면는 보관된 다시 실행 로그 AWS DMS 에만 액세스합니다. 보관된 다시 실행 로그가 Oracle ASM에만 저장되는 경우 AWS DMS 사용자 계정에 ASM 권한을 부여해야 합니다. 기본값: N 유효한 값: Y/N 예시: --oracle-settings '{"ArchivedLogsOnly": Y}'

이름	설명
asmUsePLS QLArray (ECA만 해 당)	BinaryReader로 소스 변경 내용을 캡처할 때는 이 추가 연결 속성(ECA)을 사용하세요. 이 설정을 통해 DMS는 parallelASMReadThreads 속성을 사용하여 스레드 수를 제어하면서 단일 읽기 스레드당 ASM 수준에서 50개의 읽기를 버퍼링할 수 있습니다. 이 속성을 설정하면이 AWS DMS 진 리더는 익명 PL/SQL 블록을 사용하여 다시 실행 데이터를 캡처하고 복제 인스턴스로 큰 버퍼로 다시 보냅니다. 이렇게 하면 소스로의 왕복 횟수가 줄어듭니다. 이렇게 하면 소스 캡처 성능이 크게 향상될 수 있지만 ASM 인스턴스의 PGA 메모리 사용량이 증가합니다. 메모리 대상이 충분하지 않을 경우, 안정성 문제가 발생할 수 있습니다. 다음 수식을 사용하여 단일 DMS 작업의 총 ASM 인스턴스 PGA 메모리 사용량을 추정할 수 있습니다. $\text{number\_of\_redo\_threads} * \text{parallelASMReadThreads} * 7 \text{ MB}$ 기본값: false 유효값: true/false ECA 예제: asmUsePLSQLArray=true;
ConvertTi mestampWi thZoneToUTC	'TIMESTAMP WITH TIME ZONE' 및 'TIMESTAMP WITH LOCAL TIME ZONE' 열의 타임스탬프 값을 UTC로 변환하려면 이 속성을 true로 설정합니다. 기본적으로 이 속성의 값은 'false'이며, 데이터는 소스 데이터베이스 시간대를 사용하여 복제됩니다. 기본값: false 유효값: true/false 예시: --oracle-settings '{"ConvertTimestampWithZoneToUTC": true}'

이름	설명
EnableHomogenousPartitionOps	Oracle 동종 마이그레이션을 위한 Oracle 파티션 및 하위 파티션 DDL 작업의 복제를 활성화하려면 이 속성을 true로 설정합니다. 이 기능과 개선 사항은 AWS DMS 버전 3.4.7에 도입되었습니다. 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"EnableHomogenousPartitionOps": true}'</code>
EnableHomogenousTablespace	동종 테이블스페이스 복제를 활성화하고 대상의 동일한 테이블스페이스 아래에 기존 테이블이나 인덱스를 생성하려면 이 속성을 설정합니다. 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"EnableHomogenousTablespace": true}'</code>
EscapeCharacter	이 속성을 이스케이프 문자로 설정합니다. 이 이스케이프 문자를 사용하면 단일 와일드카드 문자가 테이블 매핑 표현식에서 일반 문자처럼 동작하게 할 수 있습니다. 자세한 내용은 테이블 매핑의 와일드카드 단원을 참조하십시오. 기본값: Null 유효한 값: 와일드카드 문자를 제외한 모든 문자 예시: <code>--oracle-settings '{"EscapeCharacter": "#"}'</code>  Note escapeCharacter 는 테이블 이름에만 사용할 수 있습니다. 스키마 이름이나 열 이름에서 문자를 이스케이프 처리하지 않습니다.

이름	설명
ExposeViews	이 속성은 뷰에서 데이터를 한 번 끌어오는 데 사용하세요. 지속적 복제에는 사용할 수 없습니다. 보기에서 데이터를 추출하면 보기가 대상 스키마에서 테이블로 표시됩니다. 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"ExposeViews": true}'</code>
ExtraArchivedLogDestIds	하나 이상의 아카이브된 다시 실행 로그에 대한 하나 이상의 대상 ID를 지정합니다. 이 ID는 v\$archived_log 뷰에서 dest_id 열의 값입니다. 프라이머리 대 단일 설정 또는 프라이머리 대 다중 대기 설정에서 ArchivedLogDestId 추가 연결 속성과 함께 이 설정을 사용합니다. 이 설정은 Oracle Data Guard 데이터베이스를 소스로 사용할 때 전환에 유용합니다. 이 경우는 변경 사항을 읽기 위해 아카이브 다시 실행 로그를 가져올 대상에 대한 정보가 AWS DMS 필요합니다.는 이전 기본 인스턴스를 전환한 후 대기 인스턴스이기 때문에이 AWS DMS 정보가 필요합니다. 유효한 값: 아카이브 대상 ID 예시: <code>--oracle-settings '{"ExtraArchivedLogDestIds": 1}'</code>
FailTasksOnLobTruncation	true로 설정할 경우 LOB 열의 실제 크기가 지정된 LobMaxSize 보다 큰 경우 작업이 실패합니다. 작업이 제한된 LOB 모드로 설정되어 있고 이 옵션이 true로 설정된 경우, LOB 데이터가 잘리지 않는 대신에 작업이 실패합니다. 기본값: false 유효 값: 부울 예시: <code>--oracle-settings '{"FailTasksOnLobTruncation": true}'</code>

이름	설명
filterTransactionsOfUser (ECA만 해당)	이 추가 연결 속성(ECA)을 사용하면 LogMiner를 사용하여 Oracle에서 데이터를 복제할 때 DMS가 지정된 사용자의 트랜잭션을 무시할 수 있습니다. 심표로 구분된 사용자 이름 값을 전달할 수 있지만 값은 모두 대문자여야 합니다. ECA 예제: filterTransactionsOfUser= USERNAME;
NumberDataTypeScale	숫자 크기를 지정합니다. 스케일을 최대 38로 선택하거나 FLOAT의 경우 -1, VARCHAR의 경우 -2를 선택할 수 있습니다. 기본적으로 NUMBER 데이터 형식은 정밀도 38, 크기 10으로 변환됩니다. 기본값: 10 유효한 값: -2~38(VARCHAR의 경우 -2, FLOAT의 경우 -1) 예시: --oracle-settings '{"NumberDataTypeScale": 12}'  Note 정밀도-스케일 조합, -1(FLOAT) 또는 -2(VARCHAR)를 선택합니다. DMS는 Oracle에서 지원하는 모든 정밀도-스케일 조합을 지원합니다. 정밀도가 39 이상인 경우, -2(VARCHAR)를 선택합니다. Oracle 데이터베이스의 NumberDataTypeScale 설정은 NUMBER 데이터 형식에만 사용됩니다(명시적인 정밀도 및 스케일 정의 없이). 이 설정이 잘못 구성된 경우 정밀도 손실이 발생할 수 있다는 점에 유의해야 합니다.

이름	설명
OpenTransactionWindow	CDC 전용 작업을 위해 진행 중인 트랜잭션이 있는지 확인할 수 있는 분 단위 기간을 제공합니다.  Note OpenTransactionWindow 를 1 이상으로 설정하면 DMS는 SCN_TO_TIMESTAMP 를 사용하여 SCN 값을 타임스탬프 값으로 변환합니다. Oracle Database 제한으로 인해 CDC 시작점으로 너무 오래된 SCN을 지정하면 SCN_TO_TIMESTAMP가 ORA-08181 오류와 함께 실패하며 CDC 전용 태스크를 시작할 수 없습니다. 기본값: 0 유효한 값: 0~240 사이의 정수 예시: openTransactionWindow=15;
OraclePathPrefix	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하려면 이 문자열 속성을 필수 값으로 설정하세요. 이 값은 다시 실행 로그에 액세스하는 데 사용되는 기본 Oracle 루트를 지정합니다. 자세한 내용은 용 RDS for Oracle 소스와 함께 Binary Reader를 사용하도록 CDC 작업 구성 AWS DMS 단원을 참조하십시오. 기본값: none 유효한 값: /rdsdbdata/db/ORCL_A/ 예시: --oracle-settings '{"OraclePathPrefix": " rdsdbdata/db/ORCL_A/"}'

이름	설명
ParallelASMReadThreads	Oracle ASM(Automatic Storage Management)을 사용하여 변경 데이터 캡처(CDC)를 수행하기 위해 DMS가 구성하는 스레드 수를 변경하려면 이 속성을 설정합니다. 2(기본값) ~ 8(최대값) 사이의 정수 값을 지정할 수 있습니다. 이 속성을 ReadAheadBlocks 속성과 함께 사용합니다. 자세한 내용은 용 RDS for Oracle 소스와 함께 Binary Reader를 사용하도록 CDC 작업 구성 AWS DMS 단원을 참조하십시오. 기본값: 2 유효한 값: 2~8 사이의 정수 예시: <code>--oracle-settings '{"ParallelASMReadThreads": 6;}'</code>
ReadAheadBlocks	Oracle ASM(Automatic Storage Management) 및 ASM이 아닌 NAS 스토리지를 사용하여 CDC를 수행하기 위해 DMS가 구성하는 미리 읽기 블록 수를 변경하려면 이 속성을 설정합니다. 1000(기본값)에서 2,000,000(최대값) 사이의 정수 값을 지정할 수 있습니다. 이 속성을 ParallelASMReadThreads 속성과 함께 사용합니다. 자세한 내용은 용 RDS for Oracle 소스와 함께 Binary Reader를 사용하도록 CDC 작업 구성 AWS DMS 단원을 참조하십시오. 기본값: 1000 유효한 값: 1000~2,000,000 사이의 정수 예시: <code>--oracle-settings '{"ReadAheadBlocks": 150000}'</code>
ReadTableSpaceName	true로 설정할 경우, 이 속성에서 테이블스페이스 복제를 지원합니다. 기본값: false 유효 값: 부울 예시: <code>--oracle-settings '{"ReadTableSpaceName": true}'</code>

이름	설명
ReplacePathPrefix	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하기 위해서는 이 속성을 true로 설정합니다. 이렇게 설정하면 DMS 인스턴스가 다시 실행 로그에 액세스하기 위해 지정된 UsePathPrefix 설정을 기본 Oracle 루트로 대체합니다. 자세한 내용은 용 RDS for Oracle 소스와 함께 Binary Reader를 사용하도록 CDC 작업 구성 AWS DMS 단원을 참조하십시오. 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"ReplacePathPrefix": true}'</code>
RetryInterval	시스템이 쿼리를 다시 전송하기까지 대기하는 시간(초)을 지정합니다. 기본값: 5 유효한 값: 1부터 시작하는 숫자 예시: <code>--oracle-settings '{"RetryInterval": 6}'</code>
SecurityDbEncryptionName	Oracle 원본 데이터베이스의 열 및 테이블스페이스의 TDE(Transparent Data Encryption)에 사용되는 키의 이름을 지정합니다. Oracle 소스 엔드포인트에서 이 속성 및 연결된 암호를 설정하는 방법에 대한 자세한 내용은 Oracle을의 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS 섹션을 참조하세요. 기본값: "" 유효값: 문자열 예시: <code>--oracle-settings '{"SecurityDbEncryptionName": "ORACLE.SECURITY.DB.ENCRYPTION.Adg8m2dhkU/0v/m5QUaaNJEAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"}'</code>

이름	설명
SpatialSdo2GeoJsonFunctionName	PostgreSQL 대상으로 마이그레이션하는 Oracle 버전 12.1 이전 소스의 경우 이 속성을 사용하여 SDO_GEOMETRY를 GEOJSON 형식으로 변환합니다. 기본적으로는 AWS DMS 사용자가 존재하고 액세스할 수 있어야 하는 SD02GEOJSON 사용자 지정 함수를 AWS DMS 호출합니다. 또는 SD0GEOJSON 의 작업을 모방하는 자체 사용자 정의 함수를 생성하여 SpatialSdo2GeoJsonFunctionName 이 대신 호출하도록 설정할 수 있습니다. 기본값: SD02GEOJSON 유효값: 문자열 예시: <code>--oracle-settings '{"SpatialSdo2GeoJsonFunctionName": "myCustomSD02GEOJSONFunction"}'</code>
StandbyDelayTime	사용 이 속성을 사용하여 스탠바이 동기화에 대한 시간(분)을 지정합니다. 소스가 Active Data Guard 대기 데이터베이스인 경우, 이 속성을 사용하여 프라이머리 데이터베이스와 대기 데이터베이스 간의 시간 지연을 지정합니다. 에서 진행 중인 변경 사항을 복제하기 위한 소스로 Active Data Guard 대기 인스턴스를 사용하는 Oracle CDC 작업을 생성할 AWS DMS 수 있습니다. 이렇게 하면 프로덕션 단계에 있을 수 있는 활성 데이터베이스에 연결할 필요가 없습니다. 기본값: 0 유효한 값: 숫자 예시: <code>--oracle-settings '{"StandbyDelayTime": 1}'</code> 참고: DMS 3.4.6, 3.4.7 및 그 이상을 사용하는 경우 이 연결 설정의 사용은 선택 사항입니다. DMS 3.4.6 최신 버전과 버전 3.4.7에서 <i>dms_user</i>이 V_\$DATAGUARD_STATS 에 select 권한이 있어야 DMS가 대기 지연 시간을 계산할 수 있습니다.

이름	설명
UseAlternateFolderForOnline	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하기 위해서는 이 속성을 true로 설정합니다. 그러면 DMS 인스턴스가 지정된 접두사 교환을 사용해 모든 온라인 다시 실행 로그에 액세스합니다. 자세한 내용은 용 RDS for Oracle 소스와 함께 Binary Reader를 사용하는 CDC 작업 구성 AWS DMS 단원을 참조하십시오. 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"UseAlternateFolderForOnline": true}'</code>
UseBfile	Binary Reader 유틸리티를 사용하여 변경 데이터를 캡처하려면 이 속성을 Y로 설정합니다. 이 속성을 Y로 설정하려면 UseLogminerReader를 N으로 설정합니다. Amazon RDS for Oracle을 소스로 하여 Binary Reader를 사용하려면 추가 속성을 설정합니다. 이 설정 및 Oracle Automatic Storage Management(ASM) 사용에 대한 자세한 내용은 CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용 섹션을 참조하세요. 참고: 이 값을 추가 연결 속성(ECA)으로 설정할 때 유효한 값은 'Y'와 'N'입니다. 이 값을 엔드포인트 설정으로 설정할 때 유효한 값은 true와 false입니다. 기본값: N 유효한 값: Y/N(이 값을 ECA로 설정하는 경우), true/false(이 값을 엔드포인트 설정으로 설정하는 경우). 예시: <code>--oracle-settings '{"UseBfile": Y}'</code>

이름	설명
UseLogminerReader	이 속성을 Y로 설정하면 LogMiner 유틸리티를 사용하여 변경 데이터를 캡처합니다(기본값). AWS DMS 가 이진 파일에서 다시 실행 로그에 액세스하도록 하려면 이 옵션을 N으로 설정합니다. 이 옵션을 N으로 설정하는 경우, useBfile=Y 설정도 추가하세요. 이 설정 및 Oracle ASM(Automated Storage Management) 사용에 대한 자세한 내용은 CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용 섹션을 참조하세요. 참고: 이 값을 추가 연결 속성(ECA)으로 설정할 때 유효한 값은 'Y'와 'N'입니다. 이 값을 엔드포인트 설정으로 설정할 때 유효한 값은 true와 false입니다. 기본값: Y 유효한 값: Y/N(이 값을 ECA로 설정하는 경우), true/false(이 값을 엔드포인트 설정으로 설정하는 경우). 예시: --oracle-settings '{"UseLogminerReader": Y}'
UsePathPrefix	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하려면 이 문자열 속성을 필수 값으로 설정하세요. 이 값은 다시 실행 로그에 액세스하기 위해 기본 Oracle 루트를 대체하는 데 사용되는 경로 접두사를 지정합니다. 자세한 내용은 용 RDS for Oracle 소스와 함께 Binary Reader를 사용하도록 CDC 작업 구성 AWS DMS 단원을 참조하십시오. 기본값: none 유효한 값: /rdsdbdata/log/ 예시: --oracle-settings '{"UsePathPrefix": " /rdsdbdata/log/ "}'

Oracle용 소스 데이터 형식

용 Oracle 엔드포인트는 대부분의 Oracle 데이터 유형을 AWS DMS 지원합니다. 다음 표에는를 사용할 때 지원되는 Oracle 소스 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식에 대한 기본 매핑이 나와 있습니다.

Note

LONG 및 LONG RAW 데이터 형식을 제외하고 Oracle 소스에서 Oracle 대상으로 복제(동종 복제)하는 경우, 모든 소스 및 대상 데이터 형식이 동일합니다. 하지만 LONG 데이터 형식은 CLOB에 매핑되고 LONG RAW 데이터 형식은 BLOB에 매핑됩니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

Oracle 데이터 형식	AWS DMS 데이터 유형
BINARY_FLOAT	REAL4
BINARY_DOUBLE	REAL8
BINARY	BYTES
FLOAT (P)	정밀도가 24 이하인 경우, REAL4를 사용합니다. 정밀도가 24보다 큰 경우, REAL8을 사용합니다.
NUMBER (P,S)	스케일이 0보다 크면 NUMERIC을 사용합니다. 크기가 0인 경우: • 그리고 정밀도가 2 이하인 경우, INT1을 사용합니다. • 그리고 정밀도가 2보다 크고 4 이하인 경우, INT2를 사용합니다. • 그리고 정밀도가 4보다 크고 9 이하인 경우, INT4를 사용합니다. • 그리고 정밀도가 9보다 큰 경우, NUMERIC을 사용합니다. • 그리고 정밀도가 크기 이상인 경우, NUMERIC을 사용합니다. 스케일이 0 미만인 경우, REAL8을 사용합니다.
DATE	DATETIME

Oracle 데이터 형식	AWS DMS 데이터 유형
INTERVAL_YEAR TO MONTH	STRING(간격 year_to_month 표시 포함)
INTERVAL_DAY TO SECOND	STRING(간격 day_to_second 표시 포함)
TIMESTAMP	DATETIME
TIMESTAMP(시간대 사용)	STRING(timestamp_with_timezone 표시 포함)
TIMESTAMP WITH LOCAL TIME ZONE	STRING(timestamp_with_local_timezone 표시 포함)
CHAR	STRING
VARCHAR2	• 길이가 4,000바이트를 초과하는 경우 CLOB • 길이가 4,000바이트 이하인 경우 STRING
NCHAR	WSTRING
NVARCHAR2	• 길이가 4,000바이트를 초과하는 경우 NCLOB • 길이가 4,000바이트 이하인 경우 WSTRING
RAW	BYTES
REAL	REAL8
BLOB	BLOB 이 데이터 형식들과 함께 사용하려면 특정 작업에 대해 BLOB 데이터 형식 사용을 활성화 AWS DMS해야 합니다.는 기본 키가 포함된 테이블에서만 BLOB 데이터 형식을 AWS DMS 지원합니다.

Oracle 데이터 형식	AWS DMS 데이터 유형
CLOB	CLOB 이 데이터 형식과 함께 사용하려면 특정 작업에 대해 CLOB 데이터 형식 사용을 활성화 AWS DMS해야 합니다. CDC 중에는 기본 키가 포함된 테이블에서만 CLOB 데이터 형식을 AWS DMS 지원합니다.
NCLOB	NCLOB 이 데이터 형식과 함께 사용하려면 특정 작업에 대해 NCLOB 데이터 형식 사용을 활성화 AWS DMS해야 합니다. CDC 중에는 기본 키가 포함된 테이블에서만 NCLOB 데이터 형식을 AWS DMS 지원합니다.
LONG	CLOB LONG 데이터 형식은 배치 최적화된 적용 모드(TurboStream CDC 모드)에서 지원되지 않습니다. 이 데이터 형식과 함께 사용하려면 특정 작업에 대해 LOBs 사용을 AWS DMS활성화합니다. CDC 또는 전체 로드 중에는 기본 키가 있는 테이블에서만 LOB 데이터 유형을 AWS DMS 지원합니다. 또한 AWS DMS 는 LONG 열을 로드하기 위한 전체 LOB 모드를 지원하지 않습니다. 대신 제한된 LOB 모드를 사용하여 LONG 열을 Oracle 대상으로 마이그레이션할 수 있습니다. 제한된 LOB 모드에서는 64KB보다 긴 LONG 열로 설정한 모든 데이터를 64KB로 AWS DMS 잘라냅니다. 의 LOB 지원에 대한 자세한 내용은 섹션을 AWS DMS참조하세요. AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정

Oracle 데이터 형식	AWS DMS 데이터 유형
LONG RAW	BLOB LONG RAW 데이터 형식은 배치 최적화된 적용 모드(TurboStream CDC 모드)에서 지원되지 않습니다. 이 데이터 형식과 함께 사용하려면 특정 작업에 대해 LOBs 사용을 AWS DMS 활성화합니다. CDC 또는 전체 로드 중에는 기본 키가 있는 테이블에서만 LOB 데이터 유형을 AWS DMS 지원합니다. 또한 LONG RAW 열을 로드하기 위한 전체 LOB 모드를 지원하지 AWS DMS 않습니다. 대신 제한된 LOB 모드를 사용하여 LONG RAW 열을 Oracle 대상으로 마이그레이션할 수 있습니다. 제한된 LOB 모드에서 AWS DMS 는 LONG RAW 열로 설정한 64KB보다 긴 모든 데이터를 64KB로 잘라냅니다. 의 LOB 지원에 대한 자세한 내용은 섹션을 AWS DMS 참조하세요. AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정
XMLTYPE	CLOB
SDO_GEOMETRY	BLOB(Oracle에서 Oracle로 마이그레이션하는 경우) CLOB(Oracle에서 PostgreSQL로 마이그레이션하는 경우)

다음 데이터 형식의 열과 함께 소스로 사용되는 Oracle 테이블은 지원되지 않으며 복제될 수도 없습니다. 이 데이터 형식을 사용하는 열을 복제하면 null 열이 됩니다.

- BFILE
- ROWID
- REF
- UROWID
- 사용자 정의 데이터 형식
- ANYDATA
- VARRAY

Note

가상 열은 지원되지 않습니다.

Oracle 공간 데이터 형식 마이그레이션

공간 데이터는 공간에서 객체 또는 위치의 지오메트리 정보를 식별합니다. Oracle 데이터베이스에서 공간 객체의 기하학적 설명은 SDO_GEOMETRY 형식 객체에 저장됩니다. 이 객체 내에서 기하학적 설명은 사용자 정의 테이블의 단일 열에 있는 단일 행에 저장됩니다.

AWS DMS 는 Oracle 소스에서 Oracle 또는 PostgreSQL 대상으로 Oracle 유형 SDO_GEOMETRY 마이그레이션을 지원합니다.

를 사용하여 Oracle 공간 데이터 형식을 마이그레이션할 때는 다음 고려 사항에 유의 AWS DMS 하세요.

- Oracle 대상으로 마이그레이션하는 경우 형식 정보를 포함하는 USER_SDO_GEOM_METADATA 항목을 수동으로 전송해야 합니다.
- Oracle 소스 엔드포인트에서 PostgreSQL 대상 엔드포인트로 마이그레이션할 때는 대상 열을 AWS DMS 생성합니다. 이러한 열에는 2D 차원과 SRID(공간 참조 식별자)가 0인 기본 geometry 및 geography 형식 정보가 있습니다. 예를 들면, GEOMETRY, 2, 0입니다.
- PostgreSQL 대상으로 마이그레이션하는 Oracle 버전 12.1 이전 소스의 경우 SD02GE0JSON 함수 또는 spatialSdo2GeoJsonFunctionName 추가 연결 속성을 사용하여 SDO_GEOMETRY 객체를 GE0JSON 형식으로 변환합니다. 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.
- AWS DMS 는 전체 LOB 모드에서만 Oracle 공간 열 마이그레이션을 지원합니다. AWS DMS 는 제한된 LOB 또는 인라인 LOB 모드를 지원하지 않습니다. LOB 모드에 대한 자세한 내용은 [AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정](#) 섹션을 참조하세요.
- 는 Oracle 공간 열을 마이그레이션하기 위한 전체 LOB 모드 AWS DMS 만 지원하므로 열의 테이블에는 기본 키와 고유 키가 필요합니다. 테이블에 프라이머리 키와 고유 키가 없는 경우 마이그레이션에서 테이블을 건너뜁니다.

Microsoft SQL Server 데이터베이스를의 소스로 사용 AWS DMS

를 사용하여 하나 이상의 Microsoft SQL Server 데이터베이스에서 데이터를 마이그레이션합니다. AWS DMS, SQL Server 데이터베이스를 소스로 사용하면 데이터를 다른 SQL Server 데이터베이스 또는 AWS DMS 지원되는 다른 데이터베이스 중 하나로 마이그레이션할 수 있습니다.

가 소스로 AWS DMS 지원하는 SQL Server 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

원본 SQL Server 데이터베이스는 네트워크 상의 어느 컴퓨터에나 설치할 수 있습니다. AWS DMS 에서 사용하려면 사용자가 선택한 작업 유형에 적절한 원본 데이터베이스 액세스 권한이 있는 SQL Server 계정이 필요합니다. 자세한 내용은 [SQL Server 태스크에 대한 권한](#) 단원을 참조하십시오.

AWS DMS 는 SQL Server의 명명된 인스턴스에서 데이터 마이그레이션을 지원합니다. 원본 엔드포인트를 생성하는 경우 서버 이름에 다음 표기법을 사용할 수 있습니다.

```
IPAddress\InstanceName
```

예를 들어 올바른 원본 엔드포인트 서버 이름은 다음과 같습니다. 여기서, 이름의 첫 번째 부분은 서버의 IP 주소이며, 두 번째 부분은 SQL Server 인스턴스 이름(이 예에서는 SQLTest)입니다.

```
10.0.0.25\SQLTest
```

또한 SQL Server의 명명된 인스턴스가 수신 대기하는 포트 번호를 가져와 소스 AWS DMS 엔드포인트를 구성하는 데 사용합니다.

Note

포트 1433은 Microsoft SQL Server의 기본값입니다. 그러나 SQL Server가 시작될 때마다 변경되는 동적 포트와 방화벽을 통해 SQL Server에 연결하는 데 사용되는 특정 정적 포트 번호도 자주 사용됩니다. 따라서 AWS DMS 소스 엔드포인트를 생성할 때 SQL Server의 명명된 인스턴스의 실제 포트 번호를 알고 싶습니다.

SSL을 사용하여 SQL Server 엔드포인트와 복제 인스턴스 사이의 연결을 암호화할 수 있습니다. SQL Server 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 [에서 SSL 사용 AWS Database Migration Service](#) 섹션을 참조하세요.

CDC를 사용하여 SQL Server 데이터베이스에서 지속적으로 마이그레이션할 수 있습니다. CDC용 소스 SQL Server 데이터베이스 구성에 대한 자세한 내용은 [SQL Server에서 지속적인 복제를 위한 데이터 변경 사항 캡처](#) 섹션을 참조하세요.

SQL Server 소스 데이터베이스 작업에 대한 자세한 내용은 다음을 AWS DMS참조하세요.

주제

- [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#)
- [SQL Server 태스크에 대한 권한](#)
- [SQL Server 소스에서 지속적 복제\(CDC\) 사용을 위한 사전 요구 사항](#)
- [SQL Server에 지원되는 압축 방법](#)
- [자체 관리형 SQL Server AlwaysOn 가용성 그룹 사용](#)
- [SQL Server를 소스로 사용할 때 엔드포인트 설정 AWS DMS](#)
- [SQL Server용 소스 데이터 형식](#)
- [SQL Server에서 지속적인 복제를 위한 데이터 변경 사항 캡처](#)

SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS

다음 제한 사항은 SQL Server 데이터베이스를 AWS DMS용 소스로 사용 시 적용됩니다.

- 열의 자격 증명 속성은 대상 데이터베이스 열로 마이그레이션되지 않습니다.
- SQL Server 엔드포인트는 희소 열을 가진 테이블 사용을 지원하지 않습니다.
- Windows 인증은 지원되지 않습니다.
- SQL Server에서 컴퓨팅된 필드의 변경 사항은 복제되지 않습니다.
- 임시 테이블은 지원되지 않습니다.
- SQL Server 파티션 전환은 지원되지 않습니다.
- WRITETEXT 및 UPDATETEXT 유틸리티를 사용하는 경우 소스 데이터베이스에 적용된 이벤트를 캡처하지 AWS DMS 않습니다.
- 다음 데이터 조작 언어(DML) 패턴은 지원되지 않습니다.

```
SELECT * INTO new_table FROM existing_table
```

- SQL Server를 소스로 사용할 경우, 열 수준 암호화는 지원되지 않습니다.

- AWS DMS 는 SQL Server 2008 또는 SQL Server 2008 R2에 대한 서버 수준 감사를 소스로 지원하지 않습니다. 이는 SQL Server 2008 및 2008 R2의 알려진 문제 때문입니다. 예를 들어 다음 명령을 실행하면 실패 AWS DMS 합니다.

```
USE [master]
GO
ALTER SERVER AUDIT [my_audit_test-20140710] WITH (STATE=on)
GO
```

- SQL Server를 소스로 사용하는 경우 Geometry 및 Geography 열은 전체 LOB 모드에서 지원되지 않습니다. 대신, 제한적 LOB 모드를 사용하거나 인라인 LOB 모드를 사용하도록 InlineLobMaxSize 작업 설정을 설정하세요.
- 복제 작업에서 Microsoft SQL Server 소스 데이터베이스를 사용하는 경우, 작업을 제거하면 SQL Server Replication Publisher 정의가 제거되지 않습니다. Microsoft SQL Server 시스템 관리자는 Microsoft SQL Server에서 해당 정의를 삭제해야 합니다.
- 전체 로드 전용 작업의 경우, 스키마 바운딩된 뷰와 스키마 바인딩되지 않은 뷰에서 데이터를 마이그레이션할 수 있습니다.
- sp_rename을 사용하여 테이블 이름을 바꾸는 것은 지원되지 않습니다(예: sp_rename 'Sales.SalesRegion', 'SalesReg;')).
- sp_rename을 사용하여 열 이름을 바꾸는 것은 지원되지 않습니다(예: sp_rename 'Sales.Sales.Region', 'RegID', 'COLUMN';).
- AWS DMS 는 열 기본값을 설정 및 설정 해제하기 위한 변경 처리를 지원하지 않습니다(ALTER TABLE문과 함께 ALTER COLUMN SET DEFAULT 절 사용).
- AWS DMS 는 열 null성을 설정하기 위한 변경 처리를 지원하지 않습니다(ALTER TABLE문과 함께 ALTER COLUMN [SET|DROP] NOT NULL 절 사용).
- SQL Server 2012 및 SQL Server 2014에서는 가용성 그룹과 함께 DMS 복제를 사용하는 경우, 배포 데이터베이스를 가용성 그룹에 배치할 수 없습니다. SQL 2016은 병합, 양방향 또는 P2P 복제 토폴로지에 사용되는 배포 데이터베이스를 제외하고 배포 데이터베이스를 가용성 그룹에 배치하는 것을 지원합니다.
- 분할된 테이블의 경우 AWS DMS 는 각 파티션에 대해 서로 다른 데이터 압축 설정을 지원하지 않습니다.
- SQL Server 공간 데이터 유형(GEOGRAPHY 및 GEOMETRY)에 값을 삽입할 때 공간 참조 시스템 식별자(SRID) 속성을 무시하거나 다른 숫자를 지정할 수 있습니다. 공간 데이터 형식으로 테이블을 복제할 때는 SRID를 기본 SRID(GEOMETRY의 경우 0, GEOGRAPHY의 경우 4326)로 바 AWS DMS 끕니다.

- 데이터베이스가 MS-REPLICATION 또는 MS-CDC에 대해 구성되지 않은 경우에도 프라이머리 키가 없는 테이블을 캡처할 수 있지만 INSERT/DELETE DML 이벤트만 캡처됩니다. UPDATE 및 TRUNCATE TABLE 이벤트는 무시됩니다.
- Columnstore 인덱스는 지원되지 않습니다.
- 메모리 최적화 테이블(인 메모리 OLTP 사용)은 지원되지 않습니다.
- 여러 열로 구성된 프라이머리 키가 있는 테이블을 복제할 때 전체 로드 중에 프라이머리 키 열을 업데이트하는 것은 지원되지 않습니다.
- 지연된 지속성은 지원되지 않습니다.
- readBackupOnly=true 엔드포인트 설정(추가 연결 속성)은 RDS의 백업 수행 방식 때문에 RDS for SQL Server 소스 인스턴스에서 작동하지 않습니다.
- RDS 사용자에게는 SQL Server 저장 프로시저 sp_repldone을 실행할 수 있는 액세스 권한이 없기 때문에 EXCLUSIVE_AUTOMATIC_TRUNCATION은 Amazon RDS SQL Server 소스 인스턴스에서는 작동하지 않습니다.
- AWS DMS 는 잘라내기 명령을 캡처하지 않습니다.
- AWS DMS 는 가속 데이터베이스 복구(ADR)가 켜져 있는 데이터베이스에서의 복제를 지원하지 않습니다.
- AWS DMS 는 단일 트랜잭션 내에서 데이터 정의 언어(DDL) 및 데이터 조작 언어(DML) 문 캡처를 지원하지 않습니다.
- AWS DMS 는 데이터 계층 애플리케이션 패키지(DACPAC)의 복제를 지원하지 않습니다.
- 프라이머리 키나 고유 인덱스를 포함하고 여러 데이터 행을 업데이트하는 UPDATE 문은 대상 데이터베이스에 변경 내용을 적용할 때 충돌을 일으킬 수 있습니다. 예를 들어 대상 데이터베이스가 업데이트를 단일 UPDATE 문 대신 INSERT 문과 DELETE 문으로 적용하는 경우, 이런 일이 발생할 수 있습니다. 일괄 최적화 적용 모드에서는 테이블이 무시될 수 있습니다. 트랜잭션 적용 모드에서는 UPDATE 작업 시 제약 조건 위반이 발생할 수 있습니다. 이 문제를 방지하려면 관련 테이블을 다시 로드하세요. 또는 Apply Exceptions 제어 테이블(dmslogs.aws_dms_apply_exceptions)에서 문제가 되는 레코드를 찾아 대상 데이터베이스에서 수동으로 편집합니다. 자세한 내용은 [변경 처리 튜닝 설정](#) 단원을 참조하십시오.
- AWS DMS 는 테이블 및 스키마 복제를 지원하지 않습니다. 여기서 이름에는 다음 세트의 특수 문자가 포함됩니다.

\\ -- \n \" \b \r ' \t ;
- 데이터 마스킹은 지원되지 않습니다. 마스킹 없이 마스킹된 데이터를 AWS DMS 마이그레이션합니다.

- AWS DMS 는 기본 키가 있는 최대 32,767개의 테이블과 각 테이블에 대해 최대 1,000개의 열을 복제합니다. 이가 복제된 각 테이블에 대해 SQL Server 복제 문서를 AWS DMS 생성하고 SQL Server 복제 문서에 이러한 제한이 있기 때문입니다.
- CDC(변경 데이터 캡처)를 사용할 때는 고유 인덱스를 구성하는 모든 열을 NOT NULL로 정의해야 합니다. 이 요구 사항이 충족되지 않으면 SQL Server 시스템 오류 22838이 발생합니다.
- SQL Server가 활성 트랜잭션 로그에서 백업 로그로 아카이브하거나 활성 트랜잭션 로그에서 잘라내면 이벤트가 손실될 수 있습니다.

백업 트랜잭션 로그에 액세스할 때는 다음 제한 사항이 적용됩니다.

- 암호화된 백업은 지원되지 않습니다.
- URL 또는 Windows Azure에 저장된 백업은 지원되지 않습니다.
- AWS DMS doe snot는 대체 공유 폴더의 파일 수준에서 트랜잭션 로그 백업의 직접 처리를 지원하지 않습니다.
- Amazon RDS for Microsoft SQL Server 이외의 클라우드 SQL Server 소스의 경우는 활성 트랜잭션 로그로만 지속적 복제(CDC)를 AWS DMS 지원합니다. CDC에서는 백업 로그를 사용할 수 없습니다. SQL Server가 활성 트랜잭션 로그에서 백업 로그로 아카이브하거나 DMS가 읽기 전에 활성 트랜잭션 로그에서 잘라내면 이벤트가 손실될 수 있습니다.
- Amazon RDS for Microsoft SQL Server 소스의 경우 AWS DMS 3.5.2 이하에서는 DMS가 CDC를 통해 백업 로그에 액세스할 수 없으므로 활성 트랜잭션 로그로만 지속적 복제(CDC)를 지원합니다. RDS for SQL Server가 활성 트랜잭션 로그에서 백업 로그로 아카이브하거나 DMS가 읽기 전에 활성 트랜잭션 로그에서 잘라내면 이벤트가 손실될 수 있습니다. 이 제한은 AWS DMS 버전 3.5.3 이상에는 적용되지 않습니다.
- AWS DMS 는 Amazon RDS Proxy for SQL Server에 대한 CDC를 소스로 지원하지 않습니다.
- 전체 로드 작업 중에 SQL Server 소스를 사용할 수 없게 되면 AWS DMS 는 데이터 마이그레이션이 불완전하게 유지되더라도 여러 번의 재연결 시도 후 작업을 완료된 것으로 표시할 수 있습니다. 이 시나리오에서 대상 테이블에는 연결 손실 전에 마이그레이션된 레코드만 포함되어 소스와 대상 시스템 간에 데이터 불일치가 발생할 수 있습니다. 데이터 완전성을 보장하려면 전체 로드 작업을 완전히 다시 시작하거나 연결 중단의 영향을 받는 특정 테이블을 다시 로드해야 합니다.

SQL Server 태스크에 대한 권한

주제

- [전체 로드 전용 작업에 대한 권한](#)

• [복제가 진행 중인 태스크에 대한 권한](#)

전체 로드 전용 작업에 대한 권한

전체 로드 전용 작업을 수행하려면 다음 권한이 필요합니다. AWS DMS 는 `dms_user` 로그인을 생성하지 않습니다. SQL Server 로그인 생성에 대한 자세한 내용은 Microsoft 설명서의 [데이터베이스 사용자 생성](#) 주제를 참조하세요.

```
USE db_name;

CREATE USER dms_user FOR LOGIN dms_user;
ALTER ROLE [db_datareader] ADD MEMBER dms_user;
GRANT VIEW DATABASE STATE to dms_user;
GRANT VIEW DEFINITION to dms_user;

USE master;

GRANT VIEW SERVER STATE TO dms_user;
```

복제가 진행 중인 태스크에 대한 권한

자체 관리형 SQL Server 인스턴스는 `sysadmin` 역할을 사용하거나 사용하지 않고도 DMS를 사용하여 계속 복제하도록 구성할 수 있습니다. `sysadmin` 역할을 부여할 수 없는 SQL Server 인스턴스의 경우 DMS 사용자에게 다음과 같은 권한이 있는지 확인합니다.

자체 관리형 SQL Server 데이터베이스에서 지속적인 복제를 위한 권한 설정

1. SQL Server Management Studio(SSMS)를 사용하여 또는 [전체 로드 전용 작업에 대한 권한](#)에서 이전에 설명한 대로 암호 인증이 있는 새 SQL Server 계정을 생성합니다(예: `self_managed_user`).
2. 다음 GRANT 명령을 실행합니다.

```
GRANT VIEW SERVER STATE TO self_managed_user;

USE msdb;
GRANT SELECT ON msdb.dbo.backupset TO self_managed_user;
GRANT SELECT ON msdb.dbo.backupmediafamily TO self_managed_user;
GRANT SELECT ON msdb.dbo.backupfile TO self_managed_user;

USE db_name;
CREATE USER self_managed_user FOR LOGIN self_managed_user;
```

```
ALTER ROLE [db_owner] ADD MEMBER self_managed_user;
GRANT VIEW DEFINITION to self_managed_user;
```

3. 이전 권한 외에도 사용자는 다음 중 하나가 필요합니다.

- 사용자는 sysadmin 고정된 서버 역할의 구성원이어야 합니다.
- 소스 구성에 따라 [가용성 그룹 환경의 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음](#) 또는 [독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음](#)에 설명된 구성 및 권한.

클라우드 SQL Server 데이터베이스에서 지속적인 복제를 위한 권한 설정

클라우드에 호스팅된 SQL Server 인스턴스는 Amazon RDS for Microsoft SQL Server, Azure SQL Managed Instance 또는 DMS에서 지원하는 기타 관리형 클라우드 SQL Server 인스턴스에서 실행되는 인스턴스입니다.

SQL Server Management Studio(SSMS)를 사용하여 또는 [전체 로드 전용 작업에 대한 권한](#)에서 이전에 설명한 대로 암호 인증이 있는 새 SQL Server 계정을 생성합니다(예: rds_user).

다음 GRANT 명령을 실행합니다.

```
GRANT VIEW SERVER STATE TO rds_user;
```

Amazon RDS for Microsoft SQL Server 소스의 경우 DMS 버전 3.5.3 이상은 트랜잭션 로그 백업에서 읽기를 지원합니다. DMS가 로그 백업에 액세스할 수 있도록 하려면 위의 항목 외에도 RDS SQL Server 소스에 대한 master 사용자 권한 또는 다음 권한을 부여합니다.

```
USE msdb;
GRANT EXEC ON msdb.dbo.rds_dms_tlog_download TO rds_user;
GRANT EXEC ON msdb.dbo.rds_dms_tlog_read TO rds_user;
GRANT EXEC ON msdb.dbo.rds_dms_tlog_list_current_lsn TO rds_user;
GRANT EXEC ON msdb.dbo.rds_task_status TO rds_user;

USE db_name;
CREATE USER rds_user FOR LOGIN rds_user;
ALTER ROLE [db_owner] ADD MEMBER rds_user;
GRANT VIEW DEFINITION to rds_user;
```

Amazon Azure SQL 관리형 인스턴스의 경우 다음 권한을 부여합니다.

```
GRANT SELECT ON msdb.dbo.backupset TO rds_user;  
GRANT SELECT ON msdb.dbo.backupmediafamily TO rds_user;  
GRANT SELECT ON msdb.dbo.backupfile TO rds_user;
```

SQL Server 소스에서 지속적 복제(CDC) 사용을 위한 사전 요구 사항

온프레미스 또는 Amazon EC2의 자체 관리형 SQL Server 데이터베이스 또는 Amazon RDS나 Microsoft Azure SQL 관리형 인스턴스 같은 클라우드 데이터베이스에 지속적 복제(변경 데이터 캡처, 즉 CDC)를 사용할 수 있습니다.

특히 SQL Server 데이터베이스를 AWS DMS의 소스로 이용하는 지속적 복제를 사용할 때 다음 요구 사항이 적용됩니다.

- SQL Server는 전체 백업에 맞게 구성되어야 하며, 사용자는 데이터 복제를 시작하기 전에 백업을 수행해야 합니다.
- 복구 모델은 대량 로그 또는 전체로 설정되어야 합니다.
- 여러 디스크로의 SQL Server 백업은 지원되지 않습니다. 백업이 서로 다른 디스크의 여러 파일에 데이터베이스 백업을 쓰도록 정의된 경우는 데이터를 읽을 AWS DMS 수 없으며 AWS DMS 작업이 실패합니다.
- 자체 관리형 SQL Server 소스의 경우, 작업을 제거할 때 DMS CDC 작업에 사용되는 소스의 SQL Server Replication Publisher 정의는 제거되지 않습니다. SQL Server 시스템 관리자는 자체 관리형 원본을 위해 SQL Server에서 이 정의를 삭제해야 합니다.
- CDC 중에는 변경 사항을 읽기 위해 SQL Server 트랜잭션 로그 백업을 조희 AWS DMS 해야 합니다. AWS DMS 는 네이티브 형식이 아닌 타사 백업 소프트웨어를 사용하여 생성된 SQL Server 트랜잭션 로그 백업을 지원하지 않습니다. 네이티브 형식이고 타사 백업 소프트웨어를 사용하여 생성된 트랜잭션 로그 백업을 지원하려면 소스 엔드포인트에 use3rdPartyBackupDevice=Y 연결 속성을 추가하세요.
- 자체 관리형 SQL Server 원본의 경우, SQL Server는 변경 사항이 게시될 때까지 새로 생성된 테이블에서 변경 사항을 캡처하지 않음에 유의하세요. 테이블이 SQL Server 소스에 추가되면가 게시 생성을 AWS DMS 관리합니다. 그렇지만, 이 프로세스에는 몇 분이 걸릴 수 있습니다. 이 지연 시간 동안 새로 생성된 테이블에 적용된 작업은 대상에 캡처되거나 복제되지 않습니다.
- AWS DMS 변경 데이터 캡처를 사용하려면 SQL Server에서 전체 트랜잭션 로깅을 활성화해야 합니다. SQL Server에서 전체 트랜잭션 로깅을 켜려면 MS-REPLICATION 또는 CHANGE DATA CAPTURE(CDC)를 활성화하세요.
- SQL Server tlog 항목은 MS CDC 캡처 작업이 해당 변경 내용을 처리할 때까지는 재사용으로 표시되지 않습니다.

- CDC 작업은 메모리 최적화된 테이블에서 지원되지 않습니다. 이 제한 사항은 이 기능이 처음 도입된 SQL Server 2014 이상에 적용됩니다.
- AWS DMS 변경 데이터 캡처에는 기본적으로 Amazon EC2 또는 온프레미스 SQL 서버에서 소스로 배포 데이터베이스가 필요합니다. 따라서 프라이머리 키가 있는 테이블의 MS 복제를 구성하는 동안 배포자를 활성화했는지 확인하세요.

SQL Server에 지원되는 압축 방법

AWS DMS에서의 SQL Server 압축 방법 지원에 대해서는 다음을 참고하세요.

- AWS DMS 는 SQL Server 버전 2008 이상에서 행/페이지 압축을 지원합니다.
- AWS DMS 는 Vardecimal 스토리지 형식을 지원하지 않습니다.
- AWS DMS 는 희소 열 및 열 구조 압축을 지원하지 않습니다.

자체 관리형 SQL Server AlwaysOn 가용성 그룹 사용

SQL Server AlwaysOn 가용성 그룹은 데이터베이스 미러링의 엔터프라이즈 수준 대안으로서고가용성과 재해 복구를 제공합니다.

에서는 단일 기본 또는 보조 가용성 그룹 복제본에서 변경 사항을 마이그레이션 AWS DMS할 수 있습니다.

기본 가용성 그룹 복제본 사용

에서 기본 가용성 그룹을 소스로 사용하려면 다음을 AWS DMS수행합니다.

1. 가용성 복제본의 모든 SQL Server 인스턴스에서 배포 옵션을 켭니다. 자세한 내용은 [자체 관리형 SQL Server에서 지속적 복제 설정](#) 단원을 참조하십시오.
2. AWS DMS 콘솔에서 SQL Server 소스 데이터베이스 설정을 엽니다. 서버 이름에서 가용성 그룹 리스너에 맞게 구성된 DNS(Domain Name Service) 이름이나 IP 주소를 지정합니다.

AWS DMS 작업을 처음 시작하면 평소보다 시작하는 데 시간이 오래 걸릴 수 있습니다. 이렇게 느려지는 이유는 가용성 그룹 서버가 테이블 항목 생성을 복제하고 있기 때문입니다.

보조 가용성 그룹 복제본 사용

에서 보조 가용성 그룹을 소스로 사용하려면 다음을 AWS DMS수행합니다.

1. AWS DMS 소스 엔드포인트 사용자가 사용하는 것과 동일한 자격 증명을 사용하여 개별 복제본에 연결합니다.
2. AWS DMS 복제 인스턴스가 모든 기존 복제본의 DNS 이름을 확인하고 연결할 수 있는지 확인합니다. 다음 SQL 쿼리를 사용하여 모든 복제본의 DNS 이름을 가져올 수 있습니다.

```
select ar.replica_server_name, ar.endpoint_url from sys.availability_replicas ar
JOIN sys.availability_databases_cluster adc
ON adc.group_id = ar.group_id AND adc.database_name = '<source_database_name>';
```

3. 소스 엔드포인트를 만들 때 엔드포인트의 서버 이름 또는 엔드포인트 암호의 서버 주소에 가용성 그룹 리스너의 DNS 이름을 지정합니다. 가용성 그룹 리스너에 대한 자세한 내용은 SQL Server 설명서의 [What is an availability group listner?](#)를 참조하세요.

공용 DNS 서버 또는 온프레미스 DNS 서버를 사용하여 가용성 그룹 리스너, 기본 복제본, 보조 복제본을 확인할 수 있습니다. 온프레미스 DNS 서버를 사용하려면 Amazon Route 53 Resolver를 구성하세요. 자세한 내용은 [자체 온프레미스 이름 서버 사용](#) 단원을 참조하십시오.

4. 다음 추가 연결 속성을 소스 엔드포인트에 추가합니다.

추가 연결 속성	값	설명
applicationIntent	ReadOnly	이 ODBC 설정이 없으면 복제 작업이 기본 가용성 그룹 복제본으로 라우팅됩니다. 자세한 내용은 SQL Server 설명서의 SQL Server 네이티브 클라이언트의 고가용성과 재해 복구 지원 을 참조하세요.
multiSubnetFailover	yes	자세한 내용은 SQL Server 설명서의 SQL Server 네이티브 클라이언트의 고가용성과 재해 복구 지원 을 참조하세요.

추가 연결 속성	값	설명
alwaysOnS haredSync hedBackup IsEnabled	false	자세한 내용은 SQL Server를의 소스로 사용할 때 엔드포인트 설정 AWS DMS 단원을 참조하십시오.
activateSafeguard	false	자세한 내용은 제한 사항 섹션을 참조 하세요.
setUpMsCd cForTables	false	자세한 내용은 제한 사항 섹션을 참조 하세요.

- 가용성 그룹의 모든 복제본에서 배포 옵션을 활성화합니다. 배포자 목록에 모든 노드를 추가합니
다. 자세한 내용은 [배포를 설정하려면](#) 단원을 참조하십시오.
- 기본 읽기-쓰기 복제본에서 다음 쿼리를 실행하여 데이터베이스를 게시할 수 있도록 합니다. 이 쿼
리는 데이터베이스에 대해 한 번만 실행합니다.

```
sp_replicationdboption @dbname = N'<source DB name>', @optname = N'publish', @value  
= N'true';
```

제한 사항

다음은 보조 가용성 그룹 복제본 작업의 제한 사항입니다.

- AWS DMS 는 읽기 전용 가용성 그룹 복제본을 소스로 사용할 때 Safeguard를 지원하지 않습니다.
자세한 내용은 [SQL Server를의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.
- AWS DMS 는 읽기 전용 가용성 그룹 복제본을 소스로 사용할 때 setUpMsCdcForTables 추가 연
결 속성을 지원하지 않습니다. 자세한 내용은 [SQL Server를의 소스로 사용할 때 엔드포인트 설정
AWS DMS](#) 단원을 참조하십시오.
- AWS DMS 는 버전 3.4.7부터 자체 관리형 보조 가용성 그룹 복제본을 지속적 복제(변경 데이터 캡처
또는 CDC)를 위한 소스 데이터베이스로 사용할 수 있습니다. Cloud SQL Server 다중 AZ 읽기 전용
복제본은 지원되지 않습니다. 이전 버전의를 사용하는 경우 기본 가용성 그룹 복제본을 CDC의 소스
데이터베이스로 사용해야 AWS DMS합니다.

다른 노드로의 장애 조치

엔드포인트에 대한 `ApplicationIntent` 추가 연결 속성을 로 설정하면 `ReadOnly` AWS DMS 태스크가 읽기 전용 라우팅 우선 순위가 가장 높은 읽기 전용 노드에 연결됩니다. 우선 순위가 가장 높은 읽기 전용 노드를 사용할 수 없는 경우, 가용성 그룹에 있는 다른 읽기 전용 노드로 장애 조치됩니다. 를 설정하지 않으면 `ApplicationIntent` AWS DMS 작업은 가용성 그룹의 기본(읽기/쓰기) 노드에만 연결됩니다.

SQL Server를의 소스로 사용할 때 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 SQL Server 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 -- `microsoft-sql-server-settings` '{"*EndpointSetting*': "*value*", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

SQL Server를 소스로 할 때 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

이름	설명
ActivateSafeguard	이 속성은 Safeguard를 켜거나 끕니다. Safeguard에 대한 자세한 내용은 다음 <code>SafeguardPolicy</code> 를 참조하세요. 기본값: <code>true</code> 유효값: <code>{false, true}</code> 예시: '{"ActivateSafeguard": true}'
AlwaysOnSharedSync hedBackupIsEnabled	이 속성은 Always On 가용성 그룹 클러스터의 일부로 호스팅되는 SQL Server 소스 데이터베이스에서 마이그레이션할 AWS DMS 때의 동작을 조정합니다. AWS DMS 는 Always On 클러스터에서 실행되도록 구성된 SQL Server 소스 데이터베이스에 대한 지원이 항상되었습니다. 이 경우 AWS DMS 는 소스 데이터베이스 인스턴스가 호스팅되는 노드가 아닌 Always On 클러스터의 노드에서 트랜잭션 백업이 실행되고 있는지 추적하려고 시도합니다. 마이그레이션 작업을 시작할 때는 클러스터의 각 노드에 연결을 AWS DMS 시도하지만 노드 중 하나에 연결할 수 없는 경우 실패합니다.

이름	설명
	트랜잭션 백업 AWS DMS 을 위해 Always On 클러스터의 모든 노드를 폴링해야 하는 경우 이 속성을 로 설정합니다 false. 기본값: true 유효값: true 또는 false 예시: '{"AlwaysOnSharedSynchedBackupIsEnabled": false}'
"ApplicationIntent": "readonly"	SQL Server는 이 ODBC 드라이버 속성 설정을 통해 우선 순위가 가장 높은 읽기 전용 노드로 복제 작업을 라우팅합니다. 이 설정이 없으면 SQL Server는 복제 작업을 기본 읽기-쓰기 노드로 라우팅합니다.
ConnectionTimeout	이 추가 연결 속성(ECA)을 사용하여 SQL Server 인스턴스의 엔드포인트 연결 제한 시간을 초 단위로 설정합니다. 기본값은 10초입니다. ECA 예제: ConnectionTimeout= 30 .
EnableNonSysadminWrapper	sysadmin 사용자 없이 독립 실행형 SQL 서버에서 지속적 복제를 설정하는 경우, 이 엔드포인트 설정을 사용하세요. 이 파라미터는 AWS DMS 버전 3.4.7 이상에서 지원됩니다. 독립 실행형 SQL Server에서 지속적 복제를 설정하는 방법에 대한 자세한 내용은 SQL Server에서 지속적인 복제를 위한 데이터 변경 사항 캡처 섹션을 참조하세요. 기본값: false 유효값: true, false 예시: '{"EnableNonSysadminWrapper": true}'

이름	설명
ExecuteTimeout	이 추가 연결 속성(ECA)을 사용하여 SQL Server 인스턴스의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본 값은 60초입니다. 예시: '{"ExecuteTimeout": 100}'
FatalOnSimpleModel	이 설정을 true로 설정하면 SQL Server 데이터베이스 복구 모델이 simple로 설정된 경우 치명적 오류가 발생합니다. 기본값: false 유효값: true 또는 false 예시: '{"FatalOnSimpleModel": true}'
ForceLobLookup	인라인 LOB에서 LOB 조회를 강제 실행합니다. 기본값: false 유효값: true, false 예시: '{"ForceLobLookup": false}'
"MultiSubnetFailover": "Yes"	이 ODBC 드라이버 속성은 가용성 그룹 장애 조치 시 DMS가 새 기본 가용 영역에 연결하는 데 도움이 됩니다. 이 속성은 연결이 끊어지거나 리스너 IP 주소가 잘못된 상황을 위해 설계되었습니다. 이러한 상황에서는 가용 그룹 리스너와 연결된 모든 IP 주소에 연결하려고 AWS DMS 시도합니다.

이름	설명
ReadBackupOnly	이 속성을 사용하려면 sysadmin 권한이 필요합니다. 이 속성이 true로 설정되면 AWS DMS 는 지속적 복제 중에 변경 사항을 트랜잭션 로그 백업에서만 읽고 활성 트랜잭션 로그 파일에서는 읽지 않습니다. 이 파라미터를 true로 설정하면 전체 로드 및 지속적 복제 작업 중에 활성 트랜잭션 로그 파일 증가를 제어할 수 있습니다. 하지만 약간의 소스 지연 시간이 지속적 복제에 추가될 수 있습니다. 유효한 값: true 또는 true. 기본값은 false입니다. 예시: '{"ReadBackupOnly": true}'  Note RDS가 백업을 수행하는 방식 때문에 Amazon RDS SQL Server 소스 인스턴스에서는 이 파라미터가 작동하지 않습니다.

이름	설명
SafeguardPolicy	최적의 성능을 위해서는 활성 트랜잭션 로그(TLOG)에서 읽지 않은 모든 변경 사항을 캡처하려고 AWS DMS 시도합니다. 하지만 경우에 따라 잘라내기로 인해 활성 TLOG에 읽지 않은 변경 사항 전체가 포함되지 않을 수도 있습니다. 이 경우는 로그 백업에 AWS DMS 액세스하여 누락된 변경 사항을 캡처합니다. 로그 백업에 액세스할 필요성을 최소화하기 위해서는 다음 방법 중 하나를 사용하여 잘림을 AWS DMS 방지합니다. 1. RELY_ON_SQL_SERVER_REPLICATION_AGENT (데이터베이스에서 트랜잭션 시작): 기본값입니다 AWS DMS. 이 설정을 사용하는 경우, AWS DMS 은 SQL Server Log Reader Agent가 실행 중이어야 AWS DMS 가 복제하도록 표시된 트랜잭션을 활성 TLOG에서 이동할 수 있습니다. Log Reader Agent가 실행되고 있지 않으면 활성 TLOG가 가득 차게 되어 문제를 해결할 수 있을 때까지 소스 데이터베이스가 읽기 전용 모드로 전환될 수 있습니다. 이외의 목적으로 데이터베이스에서 Microsoft Replication을 활성화해야 하는 경우 이 설정을 선택해야 AWS DMS합니다. - 이 설정을 사용하면는 라는 테이블을 생성하여 로그 백업 읽기를 AWS DMS 최소화awsdms_truncation_safeguard 하고 데이터베이스에서 열린 트랜잭션을 모방하여 TLOG 잘림을 방지합니다. 이렇게 하면 데이터베이스에서 이벤트가 잘리고 5분(기본값) 동안 백업 로그로 이동되는 것을 방지할 수 있습니다. 유지 관리 작업 실패를 초래할 수 있으므로 유지 관리 계획에 테이블이 포함되지 않도록 하세요. Start Transactions 데이터베이스 옵션으로 구성된 작업이 없는 경우, 테이블을 안전하게 삭제할 수 있습니다. 2. EXCLUSIVE_AUTOMATIC_TRUNCATION (sp_repldone 단일 작업에만 사용):이 설정을 사용

이름	설명
	하면 로그 항목을 사용으로 표시하는 복제 에이전트 프로세스를 AWS DMS 완전히 제어할 ready for truncation 수 있습니다sp_repldone . 이 설정을 사용하면 (RELY_ON_SQL_SERVER_REPLICATION_AGENT 기본값) 설정과 같이 더미 트랜잭션을 사용하지 AWS DMS 않습니다. MS Replication이 소스 데이터베이스 이외의 다른 용도로 사용되지 않는 경우에만이 설정을 사용할 수 AWS DMS 있습니다. 또한이 설정을 사용하는 경우 하나의 AWS DMS 작업만 데이터베이스에 액세스할 수 있습니다. 동일한 데이터베이스에 대해 병렬 AWS DMS 작업을 실행해야 하는 경우를 사용합니다RELY_ON_SQL_SERVER_REPLICATION_AGENT . • 이 설정을 사용하려면 데이터베이스에서 Log Reader Agent를 중지해야 합니다. 작업이 시작될 때 Log Reader Agent가 실행 중인 경우 AWS DMS 작업은 강제로 중지합니다. 또는 작업을 시작하기 전에 수동으로 Log Reader Agent를 중지할 수도 있습니다. • MS-CDC에 이 방법을 사용하는 경우, MS-CDC 캡처 및 MS-CDC 정리 작업을 중지하고 비활성화해야 합니다. • 는 원격 시스템에 액세스할 수 없으므로 Microsoft SQL Server 마이그레이션 작업이 원격 Distributor 시스템에서 실행되는 경우이 설정을 사용할 수 AWS DMS 없습니다. • Amazon RDS 사용자에게는 sp_repldone 저장 프로시저를 실행할 수 있는 액세스 권한이 없기 때문에 EXCLUSIVE_AUTOMATIC_TRUNCATION 는 Amazon RDS for SQL Server 소스 인스턴스에서는 작동하지 않습니다. • sysadmin 역할을 사용하지 않고 Safeguard Policy 를 EXCLUSIVE_AUTOMATIC_TRUNCATION 으로 설정하는 경우, dmsuser 사용자에게

이름	설명
	dbo.syscategories 및 dbo.sysjobs 객체에 대한 권한을 부여해야 합니다. 기본값: RELY_ON_SQL_SERVER_REPLICAT ION_AGENT 유효값: {EXCLUSIVE_AUTOMATIC_TRUNCATION , RELY_ON_SQL_SERVER_REPLICATION_AGENT } 예시: '{"SafeguardPolicy": "EXCLUSIV E_AUTOMATIC_TRUNCATION"}'
SetUpMsCdcForTables	이 속성은 MS-Replication이 활성화되지 않은 작업 매핑의 테이블과 소스 데이터베이스에서 MS-CDC를 캡니다. 이 값을 true로 설정하면 소스 데이터베이스에서 sp_cdc_enable_db 저장 프로시저가 실행되고, 소스 데이터베이스에서 MS-Replication이 활성화되지 않은 작업의 각 테이블에서 sp_cdc_enable_table 저장 프로시저가 실행됩니다. 배포를 켜는 방법에 대한 자세한 내용은 자체 관리형 SQL Server에서 지속적 복제 설정 섹션을 참조하세요. 유효값: {true, false} 예시: '{"SetUpMsCdcForTables": true}'
TlogAccessMode	CDC 데이터를 가져오는 데 사용되는 모드를 나타냅니다. 기본값: PreferTlog 유효값: BackupOnly , PreferBackup , PreferTlog , TlogOnly 예시: '{"TlogAccessMode": "PreferTlog"}'
Use3rdPartyBackupDevice	이 속성이 Y로 설정된 경우, AWS DMS 는 네이티브 형식으로 생성된 타사 트랜잭션 로그 백업을 처리합니다.

SQL Server용 소스 데이터 형식

SQL Server를 소스로 사용하는 데이터 마이그레이션은 대부분의 SQL Server 데이터 유형을 AWS DMS 지원합니다. 다음 표에는 사용 시 지원되는 SQL Server 소스 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

SQL Server 데이터 형식	AWS DMS 데이터 유형
BIGINT	INT8
BIT	BOOLEAN
DECIMAL	NUMERIC
INT	INT4
MONEY	NUMERIC
NUMERIC(p,s)	NUMERIC
SMALLINT	INT2
SMALLMONEY	NUMERIC
TINYINT	UINT1
REAL	REAL4
FLOAT	REAL8
DATETIME	DATETIME
DATETIME2(SQL Server 2008 이상)	DATETIME
SMALLDATETIME	DATETIME

SQL Server 데이터 형식	AWS DMS 데이터 유형
DATE	DATE
TIME	TIME
DATETIMEOFFSET	WSTRING
CHAR	STRING
VARCHAR	STRING
VARCHAR(최대)	CLOB TEXT 이 데이터 형식들과 함께 사용하려면 특정 작업에 대해 CLOB 데이터 형식 사용을 활성화 AWS DMS해야 합니다. SQL Server 테이블의 경우 SQL Server의 LOB 열 값을 변경하지 않는 UPDATE 문에도 대상의 LOB 열을 AWS DMS 업데이트합니다. CDC 중에는 기본 키가 포함된 테이블에서만 CLOB 데이터 형식을 AWS DMS 지원합니다.
NCHAR	WSTRING
NVARCHAR(길이)	WSTRING

SQL Server 데이터 형식	AWS DMS 데이터 유형
NVARCHAR(최대)	NCLOB NTEXT 이 데이터 형식들과 함께 사용하려면 특정 작업에 대해 SupportLobs 사용을 활성화 AWS DMS 해야 합니다. Lob 지원 활성화에 대한 자세한 내용은 AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정 섹션을 참조하세요. SQL Server 테이블의 경우 SQL Server의 LOB 열 값을 변경하지 않는 UPDATE 문에도 대상의 LOB 열을 AWS DMS 업데이트합니다. CDC 중에는 기본 키가 포함된 테이블에서만 CLOB 데이터 형식을 AWS DMS 지원합니다.
BINARY	BYTES
VARBINARY	BYTES
VARBINARY(최대)	BLOB IMAGE SQL Server 테이블의 경우 SQL Server의 LOB 열 값을 변경하지 않는 UPDATE 문에도 대상의 LOB 열을 AWS DMS 업데이트합니다. 이 데이터 형식들과 함께 사용하려면 특정 작업에 대해 BLOB 데이터 형식 사용을 활성화 AWS DMS해야 합니다. AWS DMS 는 기본 키가 포함된 테이블에서만 BLOB 데이터 형식을 지원합니다.
TIMESTAMP	BYTES

SQL Server 데이터 형식	AWS DMS 데이터 유형
UNIQUEIDENTIFIER	STRING
HIERARCHYID	SQL Server 대상 엔드포인트에 복제할 때 HIERARCHYID를 사용합니다. 다른 모든 대상 엔드포인트에 복제할 때에는 WSTRING(250)을 사용합니다.
XML	NCLOB SQL Server 테이블의 경우 SQL Server의 LOB 열 값을 변경하지 않는 UPDATE 문에도 대상의 LOB 열을 AWS DMS 업데이트합니다. 이 데이터 형식과 함께 사용하려면 특정 작업에 대해 NCLOB 데이터 형식 사용을 활성화 AWS DMS해야 합니다. CDC 중에는 기본 키가 포함된 테이블에서만 NCLOB 데이터 형식을 AWS DMS 지원합니다.
GEOMETRY	이 데이터 형식을 지원하는 대상 엔드포인트에 복제할 때에는 GEOMETRY를 사용합니다. 이 데이터 형식을 지원하지 않는 대상 엔드포인트에 복제할 때에는 CLOB를 사용합니다.
GEOGRAPHY	이 데이터 형식을 지원하는 대상 엔드포인트에 복제할 때에는 GEOGRAPHY를 사용합니다. 이 데이터 형식을 지원하지 않는 대상 엔드포인트에 복제할 때에는 CLOB를 사용합니다.

AWS DMS 는 다음 데이터 형식의 필드가 포함된 테이블을 지원하지 않습니다.

- CURSOR
- SQL_VARIANT

• TABLE

Note

사용자 정의 데이터 형식은 그 기반 유형에 따라 지원됩니다. 예를 들어, DATETIME을 기반으로 한 사용자 정의 데이터는 DATETIME 데이터 형식으로 처리됩니다.

SQL Server에서 지속적인 복제를 위한 데이터 변경 사항 캡처

이 주제에서는 SQL Server 소스에서 CDC 복제를 설정하는 방법을 설명합니다.

주제

- [온프레미스 또는 Amazon EC2에서 자체 관리형 SQL Server의 데이터 변경 캡처](#)
- [클라우드 SQL Server DB 인스턴스에서 지속적 복제 설정](#)

온프레미스 또는 Amazon EC2에서 자체 관리형 SQL Server의 데이터 변경 캡처

소스 Microsoft SQL Server 데이터베이스에서 변경 내용을 캡처하려면 데이터베이스가 전체 백업을 수행하도록 구성되어 있어야 합니다. 데이터베이스를 전체 복구 모드 또는 대량 로그 모드로 구성합니다.

자체 관리형 SQL Server 소스의 경우 다음을 AWS DMS 사용합니다.

MS-REPLICATION

프라이머리 키가 있는 테이블의 변경 사항을 캡처합니다. 소스 SQL Server 인스턴스의 AWS DMS 엔드포인트 사용자에게 sysadmin 권한을 부여하여 이를 자동으로 구성할 수 있습니다. 또는 이 섹션의 단계에 따라 소스를 준비하고 AWS DMS 엔드포인트에 대한 sysadmin 권한이 없는 사용자를 사용할 수 있습니다.

MS-CDC

프라이머리 키가 없는 테이블의 변경 사항을 캡처합니다. 데이터베이스 수준에서 모든 테이블에 대해 개별적으로 MS-CDC를 활성화합니다.

지속적 복제(CDC)를 위해 SQL Server 데이터베이스를 설정할 때 다음 중 하나를 수행할 수 있습니다.

- sysadmin 역할을 사용하여 지속적 복제를 설정합니다.

- sysadmin 역할을 사용하지 않도록 지속적 복제를 설정합니다.

Note

다음 스크립트를 사용하여 프라이머리 키 또는 고유 키가 없는 모든 테이블을 찾을 수 있습니다.

```
USE [DBname]
SELECT SCHEMA_NAME(schema_id) AS schema_name, name AS table_name
FROM sys.tables
WHERE OBJECTPROPERTY(object_id, 'TableHasPrimaryKey') = 0
      AND OBJECTPROPERTY(object_id, 'TableHasUniqueCnst') = 0
ORDER BY schema_name, table_name;
```

자체 관리형 SQL Server에서 지속적 복제 설정

이 섹션에는 sysadmin 역할을 사용하거나 사용하지 않고 자체 관리형 SQL 서버에서 지속적 복제를 설정하는 방법에 대한 정보가 나와 있습니다.

주제

- [자체 관리형 SQL Server에서 지속적 복제 설정: sysadmin 역할 사용](#)
- [독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음](#)
- [가용성 그룹 환경의 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음](#)

자체 관리형 SQL Server에서 지속적 복제 설정: sysadmin 역할 사용

AWS DMS SQL Server에 대한 지속적 복제는 기본 키가 있는 테이블에는 네이티브 SQL Server 복제를 사용하고 기본 키가 없는 테이블에는 변경 데이터 캡처(CDC)를 사용합니다.

지속적 복제를 설정하기 전에 [SQL Server 소스에서 지속적 복제\(CDC\) 사용을 위한 사전 요구 사항](#) 섹션을 참조하세요.

기본 키가 있는 테이블의 경우 AWS DMS 는 일반적으로 소스에 필요한 아티팩트를 구성할 수 있습니다. 하지만 자체 관리형인 SQL Server 소스 인스턴스의 경우, 먼저 SQL Server 배포를 수동으로 구성해야 합니다. 이렇게 하면 sysadmin 권한이 있는 AWS DMS 소스 사용자가 기본 키가 있는 테이블에 대한 게시를 자동으로 생성할 수 있습니다.

배포가 이미 구성되어 있는지 확인하려면 다음 명령을 실행합니다.

```
sp_get_distributor
```

열 배포 결과가 NULL인 경우 배포가 구성되지 않은 것입니다. 다음 절차에 따라 배포를 설정할 수 있습니다.

배포를 설정하려면

1. SQL Server Management Studio(SSMS) 도구를 사용하여 SQL Server 소스 데이터베이스에 연결합니다.
2. 복제 폴더의 컨텍스트 메뉴(마우스 오른쪽 버튼 클릭)를 열고 배포 구성을 선택합니다. 배포 구성 마법사가 나타납니다.
3. 마법사에 따라 기본값을 입력하고 배포를 생성합니다.

CDC를 설정하려면

AWS DMS 버전 3.4.7 이상에서는 읽기 전용 복제본을 사용하지 않는 경우 데이터베이스 및 모든 테이블에 대해 MS CDC를 자동으로 설정할 수 있습니다. 이 기능을 사용하려면 `SetUpMsCdcForTables` ECA를 true로 설정합니다. ECA에 대한 자세한 내용은 [엔드포인트 설정](#) 섹션을 참조하세요.

3.4.7 AWS DMS 이전 버전 또는 소스로서 읽기 전용 복제본의 경우 다음 단계를 수행합니다.

1. 프라이머리 키가 없는 테이블의 경우, 데이터베이스에 MS-CDC를 설정합니다. 이렇게 하려면 `sysadmin` 역할이 할당된 계정을 사용하고 다음 명령을 실행합니다.

```
use [DBname]
EXEC sys.sp_cdc_enable_db
```

2. 다음으로, 각 소스 테이블마다 MS-CDC를 설정합니다. 고유 키는 있지만 프라이머리 키가 없는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 설정합니다.

```
exec sys.sp_cdc_enable_table
@source_schema = N'schema_name',
@source_name = N'table_name',
@index_name = N'unique_index_name',
@role_name = NULL,
@supports_net_changes = 1
GO
```

3. 프라이머리 키가 없거나 고유 키가 없는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 설정합니다.

```
exec sys.sp_cdc_enable_table
@source_schema = N'schema_name',
@source_name = N'table_name',
@role_name = NULL
GO
```

특정 테이블에서 MS-CDC를 설정하는 방법에 대한 자세한 내용은 [SQL Server 설명서](#)를 참조하세요.

독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음

이 섹션에서는 사용자 계정에 sysadmin 권한이 필요 없는 SQL Server 데이터베이스 소스에 대해 지속적 복제를 설정하는 방법을 설명합니다.

Note

sysadmin이 아닌 DMS 사용자는 이 섹션의 단계를 실행한 후 다음을 수행할 수 있는 권한을 갖게 됩니다.

- 온라인 트랜잭션 로그 파일에서 변경 사항 읽기
- 트랜잭션 로그 백업 파일의 변경 사항을 읽을 수 있는 디스크 액세스
- DMS에서 사용하는 게시물 추가 또는 변경
- 게시물에 문서 추가

1. [SQL Server에서 지속적인 복제를 위한 데이터 변경 사항 캡처](#)에 설명된 대로 복제를 위해 Microsoft SQL Server를 설정합니다.
2. 소스 데이터베이스에서 MS-REPLICATION을 활성화합니다. 이 작업은 수동으로 수행하거나, sysadmin 사용자로 태스크를 한 번 실행하여 수행할 수 있습니다.
3. 아래의 스크립트를 사용하여 소스 데이터베이스에서 awsdms 스키마를 생성합니다.

```
use master
go
create schema awsdms
go

-- Create the table valued function [awsdms].[split_partition_list] on the Master
database, as follows:
```

```
USE [master]
GO

set ansi_nulls on
go

set quoted_identifier on
go

if (object_id('[awsdms].[split_partition_list]','TF')) is not null

drop function [awsdms].[split_partition_list];

go

create function [awsdms].[split_partition_list]

(

@plist varchar(8000), --A delimited list of partitions

@dmlm nvarchar(1) --Delimiting character

)

returns @partitionsTable table --Table holding the BIGINT values of the string
    fragments

(

    pid bigint primary key

)

as

begin

declare @partition_id bigint;

declare @dmlm_pos integer;

declare @dmlm_len integer;
```

```

set @dml_len = len(@dml);

while (charindex(@dml,@plist)>0)

begin

set @dml_pos = charindex(@dml,@plist);

set @partition_id = cast( ltrim(rtrim(substring(@plist,1,@dml_pos-1))) as bigint);

insert into @partitionsTable (pid) values (@partition_id)

set @plist = substring(@plist,@dml_pos+@dml_len,len(@plist));

end

set @partition_id = cast (ltrim(rtrim(@plist)) as bigint);

insert into @partitionsTable (pid) values ( @partition_id );

return

end

GO

```

4. 아래의 스크립트를 사용하여 마스터 데이터베이스에서 [awsdms].[rtm_dump_dblog] 프로시저를 생성합니다.

```

use [MASTER]

go

if (object_id('[awsdms].[rtm_dump_dblog]','P')) is not null drop procedure
[awsdms].[rtm_dump_dblog];
go

set ansi_nulls on
go

set quoted_identifier on
GO

```

```
CREATE procedure [awsdms].[rtm_dump_dblog]

(

@start_lsn varchar(32),

@seqno integer,

@filename varchar(260),

@partition_list varchar(8000), -- A comma delimited list: P1,P2,... Pn

@programmed_filtering integer,

@minPartition bigint,

@maxPartition bigint

)

as begin

declare @start_lsn_cmp varchar(32); -- Stands against the GT comparator

SET NOCOUNT ON -- - Disable "rows affected display"

set @start_lsn_cmp = @start_lsn;

if (@start_lsn_cmp) is null

set @start_lsn_cmp = '00000000:00000000:0000';

if (@partition_list is null)

begin

RAISERROR ('Null partition list waspassed',16,1);

return

end
```



```
if (@start_lsn) is not null

set @start_lsn = '0x'+@start_lsn;

if (@programmed_filtering=0)


SELECT

[Current LSN],

[operation],

[Context],

[Transaction ID],

[Transaction Name],

[Begin Time],

[End Time],

[Flag Bits],

[PartitionID],

[Page ID],

[Slot ID],

[RowLog Contents 0],

[Log Record],

[RowLog Contents 1]

FROM

fn_dump_dblog (

@start_lsn, NULL, N'DISK', @seqno, @filename,
```

```
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default)

where [Current LSN] collate SQL_Latin1_General_CP1_CI_AS > @start_lsn_cmp collate
SQL_Latin1_General_CP1_CI_AS

and

(

( [operation] in ('LOP_BEGIN_XACT','LOP_COMMIT_XACT','LOP_ABORT_XACT') )

or

( [operation] in ('LOP_INSERT_ROWS','LOP_DELETE_ROWS','LOP_MODIFY_ROW')

and

( ( [context] in ('LCX_HEAP','LCX_CLUSTERED','LCX_MARK_AS_GHOST') ) or ([context] =
'LCX_TEXT_MIX' and (datalength([RowLog Contents 0])) in (0,1))))

and [PartitionID] in ( select * from master.awsdfs.split_partition_list
(@partition_list,',') )

)

or
```

```
([operation] = 'LOP_HOBT_DDL')

)

else

SELECT

[Current LSN],

[operation],

[Context],

[Transaction ID],

[Transaction Name],

[Begin Time],

[End Time],

[Flag Bits],

[PartitionID],

[Page ID],

[Slot ID],

[RowLog Contents 0],

[Log Record],

[RowLog Contents 1] -- After Image

FROM

fn_dump_dblog (

@start_lsn, NULL, N'DISK', @seqno, @filename,
```

```
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default,
default, default, default, default, default, default, default)

where [Current LSN] collate SQL_Latin1_General_CP1_CI_AS > @start_lsn_cmp collate
SQL_Latin1_General_CP1_CI_AS

and

(

( [operation] in ('LOP_BEGIN_XACT','LOP_COMMIT_XACT','LOP_ABORT_XACT') )

or

( [operation] in ('LOP_INSERT_ROWS','LOP_DELETE_ROWS','LOP_MODIFY_ROW')

and

( ( [context] in ('LCX_HEAP','LCX_CLUSTERED','LCX_MARK_AS_GHOST') ) or ([context] =
'LCX_TEXT_MIX' and (datalength([RowLog Contents 0]) in (0,1))))

and ([PartitionID] is not null) and ([PartitionID] >= @minPartition and
[PartitionID]<=@maxPartition)

)

or

([operation] = 'LOP_HOBT_DDL')
```

```
)  
  
SET NOCOUNT OFF -- Re-enable "rows affected display"  
  
end  
  
GO
```

5. 아래의 스크립트를 사용하여 마스터 데이터베이스에서 인증서를 생성합니다.

```
Use [master]  
Go  
  
CREATE CERTIFICATE [awsdms_rtm_dump_dblog_cert] ENCRYPTION BY PASSWORD =  
    N'@5trongpassword'  
  
WITH SUBJECT = N'Certificate for FN_DUMP_DBLOG Permissions';
```

6. 아래의 스크립트를 사용하여 인증서에서 로그인을 생성합니다.

```
Use [master]  
Go  
  
CREATE LOGIN awsdms_rtm_dump_dblog_login FROM CERTIFICATE  
    [awsdms_rtm_dump_dblog_cert];
```

7. 아래의 스크립트를 사용하여 sysadmin 서버 역할에 로그인을 추가합니다.

```
ALTER SERVER ROLE [sysadmin] ADD MEMBER [awsdms_rtm_dump_dblog_login];
```

8. 아래의 스크립트를 사용하여 [master].[awsdms].[rtm_dump_dblog]에 서명을 추가합니다.

```
Use [master]  
GO  
ADD SIGNATURE  
TO [master].[awsdms].[rtm_dump_dblog] BY CERTIFICATE [awsdms_rtm_dump_dblog_cert]  
    WITH PASSWORD = '@5trongpassword';
```

Note

저장된 프로시저를 다시 생성할 경우 서명을 다시 추가해야 합니다.

9. 아래의 스크립트를 사용하여 마스터 데이터베이스에 [awsdms].[rtm_position_1st_timestamp]를 생성합니다.

```
use [master]
if object_id('[awsdms].[rtm_position_1st_timestamp]','P') is not null
DROP PROCEDURE [awsdms].[rtm_position_1st_timestamp];
go
create procedure [awsdms].[rtm_position_1st_timestamp]
(
    @dbname          sysname,          -- Database name
    @seqno           integer,          -- Backup set sequence/position number
    @filename         varchar(260),    -- The backup filename
    @1stTimeStamp     varchar(40)      -- The timestamp to position by
)
as begin

    SET NOCOUNT ON          -- Disable "rows affected display"

    declare @firstMatching table
    (
        cLsn varchar(32),
        bTim datetime
    )

    declare @sql nvarchar(4000)
    declare @nl          char(2)
    declare @tb          char(2)
    declare @fnameVar    nvarchar(254) = 'NULL '

    set @nl = char(10); -- New line
    set @tb = char(9)   -- Tab separator

    if (@filename is not null)
    set @fnameVar = '''+@filename +'''

    set @sql='use ['+@dbname+'];'+@nl+
    'select top 1 [Current LSN],[Begin Time]'+@nl+
```

```
'FROM fn_dump_dblog (NULL, NULL, NULL, '+' cast(@seqno as varchar(10))+','+@fnameVar+', '+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    @tb+'default, default, default, default, default, default, default, default,'+@nl+'
    where operation=''LOP_BEGIN_XACT'' '+@nl+'
    'and [Begin Time]>= cast(''+'''+@1stTimeStamp+''''+ ' as datetime)'+@nl+'

--print @sql
delete from @firstMatching
insert into @firstMatching exec sp_executesql @sql      -- Get them all

select top 1 cLsn as [matching LSN],convert(varchar,bTim,121) as [matching
Timestamp] from @firstMatching;

SET NOCOUNT OFF          -- Re-enable "rows affected display"

end
GO
```

10. 아래의 스크립트를 사용하여 마스터 데이터베이스에서 인증서를 생성합니다.

```
Use [master]
Go
CREATE CERTIFICATE [awsdms_rtm_position_1st_timestamp_cert]
ENCRYPTION BY PASSWORD = '@5trongpassword'
WITH SUBJECT = N'Certificate for FN_POSITION_1st_TIMESTAMP Permissions';
```

11. 아래의 스크립트를 사용하여 인증서에서 로그인을 생성합니다.

```
Use [master]
Go
CREATE LOGIN awsdms_rtm_position_1st_timestamp_login FROM CERTIFICATE
    [awsdms_rtm_position_1st_timestamp_cert];
```

12. 아래의 스크립트를 사용하여 sysadmin 역할에 로그인을 추가합니다.

```
ALTER SERVER ROLE [sysadmin] ADD MEMBER [awsdms_rtm_position_1st_timestamp_login];
```

13. 아래의 스크립트를 사용하여 인증서를 통해 [master].[awsdms].[rtm_position_1st_timestamp]에 서명을 추가합니다.

```
Use [master]
GO
ADD SIGNATURE
TO [master].[awsdms].[rtm_position_1st_timestamp]
BY CERTIFICATE [awsdms_rtm_position_1st_timestamp_cert]
WITH PASSWORD = '@5trongpassword';
```

14. 아래의 스크립트를 사용하여 DMS 사용자에게 새 저장 프로시저에 대한 실행 액세스 권한을 부여합니다.

```
use master
go
GRANT execute on [awsdms].[rtm_position_1st_timestamp] to dms_user;
```

15. 다음 각 데이터베이스에서 다음과 같은 권한 및 역할을 보유한 사용자를 생성합니다.

Note

각 복제본에서 동일한 SID를 사용하여 dmsnosysadmin 사용자 계정을 생성해야 합니다. 아래의 SQL 쿼리는 각 복제본에서 dmsnosysadmin 계정 SID 값을 확인하는 데 도움이 될 수 있습니다. 사용자 생성에 대한 자세한 내용은 [Microsoft SQL 서버 설명서의 CREATE USER \(Transact-SQL\)](#) 섹션을 참조하세요. Azure SQL 데이터베이스의 SQL 사용자 계정 생성에 대한 자세한 내용은 [활성 geo-replication](#) 섹션을 참조하세요.

```
use master
go
grant select on sys.fn_dblog to [DMS_user]
grant view any definition to [DMS_user]
grant view server state to [DMS_user]--(should be granted to the login).
grant execute on sp_repldone to [DMS_user]
grant execute on sp_replincrementlsn to [DMS_user]
grant execute on sp_addpublication to [DMS_user]
grant execute on sp_addarticle to [DMS_user]
grant execute on sp_articlefilter to [DMS_user]
```



```
grant select on [awsdms].[split_partition_list] to [DMS_user]
grant execute on [awsdms].[rtm_dump_dblog] to [DMS_user]
```

```
use msdb
go
grant select on msdb.dbo.backupset to self_managed_user
grant select on msdb.dbo.backupmediafamily to self_managed_user
grant select on msdb.dbo.backupfile to self_managed_user
```

소스 데이터베이스에서 다음과 같은 스크립트를 실행합니다.

```
use Source_DB
Go
EXEC sp_addrolemember N'db_owner', N'DMS_user'
```

16. 마지막으로, 소스 SQL Server 엔드포인트에 추가 연결 속성(ECA)을 추가합니다.

```
enableNonSysadminWrapper=true;
```

가용성 그룹 환경의 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음

이 섹션에서는 사용자 계정에 sysadmin 권한이 필요 없는 가용성 그룹 환경의 SQL Server 데이터베이스 소스에 대해 지속적 복제를 설정하는 방법을 설명합니다.

Note

sysadmin이 아닌 DMS 사용자는 이 섹션의 단계를 실행한 후 다음을 수행할 수 있는 권한을 갖게 됩니다.

- 온라인 트랜잭션 로그 파일에서 변경 사항 읽기
- 트랜잭션 로그 백업 파일의 변경 사항을 읽을 수 있는 디스크 액세스
- DMS에서 사용하는 게시물 추가 또는 변경
- 게시물에 문서 추가

가용성 그룹 환경에서 sysadmin 사용자를 사용하지 않고 지속적 복제를 설정하려면

1. [SQL Server에서 지속적인 복제를 위한 데이터 변경 사항 캡처](#)에 설명된 대로 복제를 위해 Microsoft SQL Server를 설정합니다.
2. 소스 데이터베이스에서 MS-REPLICATION을 활성화합니다. 이 작업은 수동으로 수행하거나, sysadmin 사용자를 통해 태스크를 한 번 실행하여 수행할 수 있습니다.

 Note

MS-REPLICATION 배포자를 로컬로 구성하거나, 연결된 서버를 통해 sysadmin 이외의 사용자가 액세스할 수 있는 방식으로 구성해야 합니다.

3. 단일 태스크 내에서 sp_repldone을 단독으로 사용 엔드포인트 옵션이 활성화된 경우, MS-REPLICATION Log Reader 작업을 중지합니다.
4. 각각의 복제에서 다음 단계를 수행합니다.

1. 마스터 데이터베이스에서 [awsdms][awsdms] 스키마를 생성합니다.

```
CREATE SCHEMA [awsdms]
```

2. 마스터 데이터베이스에서 [awsdms].[split_partition_list] 테이블 값 함수를 생성합니다.

```
USE [master]
GO

SET ansi_nulls on
GO

SET quoted_identifier on
GO

IF (object_id('[awsdms].[split_partition_list]','TF')) is not null
    DROP FUNCTION [awsdms].[split_partition_list];
GO

CREATE FUNCTION [awsdms].[split_partition_list]
(
    @plist varchar(8000),    --A delimited list of partitions
    @dlim nvarchar(1)      --Delimiting character
)

```

```

RETURNS @partitionsTable table --Table holding the BIGINT values of the string
    fragments
(
    pid bigint primary key
)
AS
BEGIN
    DECLARE @partition_id bigint;
    DECLARE @dlm_pos integer;
    DECLARE @dlm_len integer;
    SET @dlm_len = len(@dlm);
    WHILE (charindex(@dlm,@plist)>0)
    BEGIN
        SET @dlm_pos = charindex(@dlm,@plist);
        SET @partition_id = cast( ltrim(rtrim(substring(@plist,1,@dlm_pos-1))) as
        bigint);
        INSERT into @partitionsTable (pid) values (@partition_id)
        SET @plist = substring(@plist,@dlm_pos+@dlm_len,len(@plist));
    END
    SET @partition_id = cast (ltrim(rtrim(@plist)) as bigint);
    INSERT into @partitionsTable (pid) values ( @partition_id );
    RETURN
END
GO

```

3. 마스터 데이터베이스에서 [awsdms].[rtm_dump_dblog] 프로시저를 생성합니다.

```

USE [MASTER]
GO

IF (object_id('[awsdms].[rtm_dump_dblog]','P')) is not null
    DROP PROCEDURE [awsdms].[rtm_dump_dblog];
GO

SET ansi_nulls on
GO

SET quoted_identifier on
GO

CREATE PROCEDURE [awsdms].[rtm_dump_dblog]
(
    @start_lsn          varchar(32),
    @seqno              integer,

```

```

@filename          varchar(260),
@partition_list    varchar(8000), -- A comma delimited list: P1,P2,... Pn
@programmed_filtering integer,
@minPartition      bigint,
@maxPartition      bigint
)
AS
BEGIN

    DECLARE @start_lsn_cmp varchar(32); -- Stands against the GT comparator

    SET NOCOUNT ON -- Disable "rows affected display"

    SET @start_lsn_cmp = @start_lsn;
    IF (@start_lsn_cmp) is null
        SET @start_lsn_cmp = '00000000:00000000:0000';

    IF (@partition_list is null)
    BEGIN
        RAISERROR ('Null partition list was passed',16,1);
        return
        --set @partition_list = '0,';    -- A dummy which is never matched
    END

    IF (@start_lsn) is not null
        SET @start_lsn = '0x'+@start_lsn;

    IF (@programmed_filtering=0)
    SELECT
        [Current LSN],
        [operation],
        [Context],
        [Transaction ID],
        [Transaction Name],
        [Begin Time],
        [End Time],
        [Flag Bits],
        [PartitionID],
        [Page ID],
        [Slot ID],
        [RowLog Contents 0],
        [Log Record],
        [RowLog Contents 1] -- After Image
    FROM

```

```

fn_dump_dblog (
    @start_lsn, NULL, N'DISK', @seqno, @filename,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default)
WHERE
    [Current LSN] collate SQL_Latin1_General_CP1_CI_AS > @start_lsn_cmp collate
SQL_Latin1_General_CP1_CI_AS -- This aims for implementing FN_DBLOG based on GT
comparator.
    AND
    (
        ( [operation] in ('LOP_BEGIN_XACT','LOP_COMMIT_XACT','LOP_ABORT_XACT') )
        OR
        ( [operation] in ('LOP_INSERT_ROWS','LOP_DELETE_ROWS','LOP_MODIFY_ROW')
            AND
            ( ( [context] in ('LCX_HEAP','LCX_CLUSTERED','LCX_MARK_AS_GHOST') )
or ([context] = 'LCX_TEXT_MIX') )
            AND
            [PartitionID] in ( select * from master.awsdms.split_partition_list
(@partition_list,',') )
        )
        OR
        ([operation] = 'LOP_HOBT_DDL')
    )
ELSE
    SELECT
        [Current LSN],
        [operation],
        [Context],
        [Transaction ID],
        [Transaction Name],
        [Begin Time],
        [End Time],
        [Flag Bits],
        [PartitionID],
        [Page ID],
        [Slot ID],
        [RowLog Contents 0],

```

```

[Log Record],
[RowLog Contents 1] -- After Image
FROM
fn_dump_dblog (
    @start_lsn, NULL, N'DISK', @seqno, @filename,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default,
    default, default, default, default, default, default, default, default)
WHERE [Current LSN] collate SQL_Latin1_General_CP1_CI_AS > @start_lsn_cmp
collate SQL_Latin1_General_CP1_CI_AS -- This aims for implementing FN_DBLOG
based on GT comparator.
AND
(
    ( [operation] in ('LOP_BEGIN_XACT','LOP_COMMIT_XACT','LOP_ABORT_XACT') )
    OR
    ( [operation] in ('LOP_INSERT_ROWS','LOP_DELETE_ROWS','LOP_MODIFY_ROW')
      AND
        ( ( [context] in ('LCX_HEAP','LCX_CLUSTERED','LCX_MARK_AS_GHOST') )
        or ([context] = 'LCX_TEXT_MIX') )
      AND ([PartitionID] is not null) and ([PartitionID] >= @minPartition and
[PartitionID]<=@maxPartition)
    )
    OR
    ([operation] = 'LOP_HOBT_DDL')
)
SET NOCOUNT OFF -- Re-enable "rows affected display"
END
GO

```

4. 마스터 데이터베이스에서 인증서를 생성합니다.

```

USE [master]
GO
CREATE CERTIFICATE [awsdms_rtm_dump_dblog_cert]
    ENCRYPTION BY PASSWORD = N'@hardpassword1'
    WITH SUBJECT = N'Certificate for FN_DUMP_DBLOG Permissions'

```

5. 인증서에서 로그인을 생성합니다.

```
USE [master]
GO
CREATE LOGIN awsdms_rtm_dump_dblog_login FROM CERTIFICATE
[awsdms_rtm_dump_dblog_cert];
```

6. sysadmin 서버 역할에 로그인을 추가합니다.

```
ALTER SERVER ROLE [sysadmin] ADD MEMBER [awsdms_rtm_dump_dblog_login];
```

7. 인증서를 사용하여 [master].[awsdms].[rtm_dump_dblog] 프로시저에 서명을 추가합니다.

```
USE [master]
GO

ADD SIGNATURE
TO [master].[awsdms].[rtm_dump_dblog]
BY CERTIFICATE [awsdms_rtm_dump_dblog_cert]
WITH PASSWORD = '@hardpassword1';
```

Note

저장된 프로시저를 다시 생성할 경우 서명을 다시 추가해야 합니다.

8. 마스터 데이터베이스에서 [awsdms].[rtm_position_1st_timestamp] 프로시저를 생성합니다.

```
USE [master]
IF object_id('[awsdms].[rtm_position_1st_timestamp]','P') is not null
    DROP PROCEDURE [awsdms].[rtm_position_1st_timestamp];
GO
CREATE PROCEDURE [awsdms].[rtm_position_1st_timestamp]
(
    @dbname                sysname,          -- Database name
    @seqno                 integer,          -- Backup set sequence/position number
    within file
    @filename               varchar(260),    -- The backup filename
    @1stTimeStamp           varchar(40)      -- The timestamp to position by
)
AS
BEGIN
    SET NOCOUNT ON          -- Disable "rows affected display"
```

```

DECLARE @firstMatching table
(
    cLsn varchar(32),
    bTim datetime
)
DECLARE @sql nvarchar(4000)
DECLARE @nl          char(2)
DECLARE @tb          char(2)
DECLARE @fnameVar    sysname = 'NULL'

SET @nl = char(10); -- New line
SET @tb = char(9)   -- Tab separator

IF (@filename is not null)
    SET @fnameVar = '''+@filename +'''
SET @filename = '''+@filename +'''
SET @sql='use ['+@dbname+'];'+@nl+
    'SELECT TOP 1 [Current LSN],[Begin Time]'+@nl+
    'FROM fn_dump_dblog (NULL, NULL, NULL, '+ cast(@seqno as varchar(10))+','+
@filename +','+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    'WHERE operation=''LOP_BEGIN_XACT''' +@nl+
    'AND [Begin Time]>= cast(''+'''+@1stTimeStamp+'''+'' as datetime)'+@nl

--print @sql
DELETE FROM @firstMatching
INSERT INTO @firstMatching exec sp_executesql @sql      -- Get them all
SELECT TOP 1 cLsn as [matching LSN],convert(varchar,bTim,121) AS[matching
Timestamp] FROM @firstMatching;

SET NOCOUNT OFF      -- Re-enable "rows affected display"

END
GO

```


9. 마스터 데이터베이스에서 인증서를 생성합니다.

```
USE [master]
GO
CREATE CERTIFICATE [awsdms_rtm_position_1st_timestamp_cert]
    ENCRYPTION BY PASSWORD = N'@hardpassword1'
    WITH SUBJECT = N'Certificate for FN_POSITION_1st_TIMESTAMP Permissions';
```

10. 인증서에서 로그인을 생성합니다.

```
USE [master]
GO
CREATE LOGIN awsdms_rtm_position_1st_timestamp_login FROM CERTIFICATE
    [awsdms_rtm_position_1st_timestamp_cert];
```

11. sysadmin 서버 역할에 로그인을 추가합니다.

```
ALTER SERVER ROLE [sysadmin] ADD MEMBER
    [awsdms_rtm_position_1st_timestamp_login];
```

12. 인증서를 사용하여 [master].[awsdms].[rtm_position_1st_timestamp] 프로시저에 서명을 추가합니다.

```
USE [master]
GO
ADD SIGNATURE
    TO [master].[awsdms].[rtm_position_1st_timestamp]
    BY CERTIFICATE [awsdms_rtm_position_1st_timestamp_cert]
    WITH PASSWORD = '@hardpassword1';
```

Note

저장된 프로시저를 다시 생성할 경우 서명을 다시 추가해야 합니다.

13. 다음 각 데이터베이스에서 다음과 같은 권한/역할을 보유한 사용자를 생성합니다.

Note

각 복제본에서 동일한 SID를 사용하여 dmsnosysadmin 사용자 계정을 생성해야 합니다. 아래의 SQL 쿼리는 각 복제본에서 dmsnosysadmin 계정 SID 값을 확인하는 데도

움이 될 수 있습니다. 사용자 생성에 대한 자세한 내용은 [Microsoft SQL 서버 설명서의 CREATE USER \(Transact-SQL\)](#) 섹션을 참조하세요. Azure SQL 데이터베이스의 SQL 사용자 계정 생성에 대한 자세한 내용은 [활성 geo-replication](#) 섹션을 참조하세요.

```
SELECT @@servername servername, name, sid, create_date, modify_date
FROM sys.server_principals
WHERE name = 'dmsnosysadmin';
```

14.각 복제에서 마스터 데이터베이스에 대한 권한을 부여합니다.

```
USE master
GO

GRANT select on sys.fn_dblog to dmsnosysadmin;
GRANT view any definition to dmsnosysadmin;
GRANT view server state to dmsnosysadmin -- (should be granted to the login).
GRANT execute on sp_repldone to dmsnosysadmin;
GRANT execute on sp_replincrementlsn to dmsnosysadmin;
GRANT execute on sp_addpublication to dmsnosysadmin;
GRANT execute on sp_addarticle to dmsnosysadmin;
GRANT execute on sp_articlefilter to dmsnosysadmin;
GRANT select on [awsdms].[split_partition_list] to dmsnosysadmin;
GRANT execute on [awsdms].[rtm_dump_dblog] to dmsnosysadmin;
GRANT execute on [awsdms].[rtm_position_1st_timestamp] to dmsnosysadmin;
```

15.각 복제에서 msdb 데이터베이스에 대한 권한을 부여합니다.

```
USE msdb
GO
GRANT select on msdb.dbo.backupset TO self_managed_user
GRANT select on msdb.dbo.backupmediafamily TO self_managed_user
GRANT select on msdb.dbo.backupfile TO self_managed_user
```

16.소스 데이터베이스에서 db_owner 역할을 dmsnosysadmin에 추가합니다. 데이터베이스가 동기화되었으므로, 기본 복제에만 역할을 추가할 수 있습니다.

```
use <source DB>
GO
EXEC sp_addrolemember N'db_owner', N'dmsnosysadmin'
```

클라우드 SQL Server DB 인스턴스에서 지속적 복제 설정

이 섹션에서는 클라우드에 호스팅된 SQL Server 데이터베이스 인스턴스에 CDC를 설정하는 방법을 설명합니다. 클라우드에 호스팅된 SQL 서버 인스턴스는 Amazon RDS for SQL Server, Azure SQL Managed Instance 또는 기타 관리형 클라우드 SQL Server 인스턴스에서 실행되는 인스턴스입니다. 각 데이터베이스 유형에 따른 지속적 복제 제한 사항에 대한 자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요.

지속적 복제를 설정하기 전에 [SQL Server 소스에서 지속적 복제\(CDC\) 사용을 위한 사전 요구 사항](#) 섹션을 참조하세요.

자체 관리형 Microsoft SQL Server 소스와 달리 Amazon RDS for SQL Server는 MS-Replication을 지원하지 않습니다. 따라서 기본 키가 있거나 없는 테이블에 MS-CDC를 사용해야 AWS DMS 합니다.

Amazon RDS는 소스 SQL Server 인스턴스의 지속적인 변경에 AWS DMS 가 사용하는 복제 아티팩트를 설정할 수 있는 sysadmin 권한을 부여하지 않습니다. 다음 절차와 같이 Amazon RDS 인스턴스에서 MS-CDC를 켜야 합니다(마스터 사용자 권한 사용).

클라우드 SQL Server DB 인스턴스에서 MS-CDC를 켜려면

1. 데이터베이스 수준에서 다음 쿼리 중 하나를 실행합니다.

RDS for SQL Server DB 인스턴스에는 다음 쿼리를 사용하세요.

```
exec msdb.dbo.rds_cdc_enable_db 'DB_name'
```

Azure SQL 관리형 DB 인스턴스에는 다음 쿼리를 사용하세요.

```
USE DB_name
GO
EXEC sys.sp_cdc_enable_db
GO
```

2. 프라이머리 키가 있는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 켭니다.

```
exec sys.sp_cdc_enable_table
@source_schema = N'schema_name',
@source_name = N'table_name',
@role_name = NULL,
@supports_net_changes = 1
GO
```

고유 키는 있지만 프라이머리 키가 없는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 켭니다.

```
exec sys.sp_cdc_enable_table
@source_schema = N'schema_name',
@source_name = N'table_name',
@index_name = N'unique_index_name',
@role_name = NULL,
@supports_net_changes = 1
GO
```

프라이머리 키도 없고 고유 키도 없는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 켭니다.

```
exec sys.sp_cdc_enable_table
@source_schema = N'schema_name',
@source_name = N'table_name',
@role_name = NULL
GO
```

3. 보존 기간 설정:

- DMS 버전 3.5.3 이상을 사용하여 복제하는 RDS for SQL Server 인스턴스의 경우 보존 기간이 기본값인 5초로 설정되어 있는지 확인합니다. DMS 3.5.2 이하에서 DMS 3.5.3 이상으로 업그레이드하거나 이동하는 경우 새 인스턴스 또는 업그레이드된 인스턴스에서 작업이 실행된 후 폴링 간격 값을 변경합니다. 다음 스크립트는 보존 기간을 5초로 설정합니다.

```
use dbname
EXEC sys.sp_cdc_change_job @job_type = 'capture' ,@pollinginterval = 5
exec sp_cdc_stop_job 'capture'
exec sp_cdc_start_job 'capture'
```

- @pollinginterval 파라미터는 초 단위로 측정되며 권장 값은 86,399로 설정되어 있습니다. 즉, @pollinginterval = 86399일 경우 트랜잭션 로그는 86,399초(하루) 동안 변경 사항을 보존합니다. 프로시저 `exec sp_cdc_start_job 'capture'`에서 설정을 시작합니다.

Note

일부 버전의 SQL Server에서는 pollinginterval 값을 3,599초 이상으로 설정하면 값이 기본값인 5초로 재설정됩니다. 이 경우가 항목을 AWS DMS 읽기 전에 T-Log

항목이 제거됩니다. 이 알려진 문제의 영향을 받는 SQL Server 버전을 확인하려면 [이 Microsoft KB 문서](#)를 참조하세요.

다중 AZ에서 Amazon RDS를 사용하는 경우, 장애 조치 시 보조 AZ도 올바른 값을 갖도록 설정해야 합니다.

```
exec rdsadmin..rds_set_configuration 'cdc_capture_pollinginterval' , <5 or preferred value>
```

AWS DMS 복제 작업이 1시간 이상 중지될 때 보존 기간을 유지하려면

Note

다음 단계는 DMS 3.5.3 이상을 사용하여 RDS for SQL Server 소스를 복제하는 데 필요하지 않습니다.

1. 다음 명령을 사용하여 트랜잭션 로그를 잘라내는 작업을 중지합니다.

```
exec sp_cdc_stop_job 'capture'
```

2. AWS DMS 콘솔에서 작업을 찾아 작업을 재개합니다.
3. 모니터링 탭을 선택하고 CDCLatencySource 지표를 확인합니다.
4. CDCLatencySource 지표가 0이고 그대로 유지되면 다음 명령을 사용하여 트랜잭션 로그를 잘라내는 작업을 다시 시작합니다.

```
exec sp_cdc_start_job 'capture'
```

SQL Server 트랜잭션 로그를 잘라내는 작업을 시작해야 한다는 점을 명심하세요. 그렇지 않으면 SQL Server 인스턴스의 스토리지가 꽉 찰 수 있습니다.

RDS for SQL Server를 소스로 사용할 때 권장되는 설정 AWS DMS

For AWS DMS 3.5.3 이상

Note

RDS for SQL Server 로그 백업 기능의 초기 릴리스는 DMS 버전 3.5.3 릴리스 후 생성하거나 수정한 엔드포인트에 대해 기본적으로 활성화됩니다. 기존 엔드포인트에 이 기능을 사용하려면 변경하지 않고 엔드포인트를 수정합니다.

AWS DMS 버전 3.5.3에는 로그 백업에서 읽기에 대한 지원이 도입되었습니다. DMS는 주로 활성 트랜잭션 로그의 읽기에 의존하여 이벤트를 복제합니다. DMS가 활성 로그에서 트랜잭션을 읽기 전에 백업되는 경우 작업은 온디맨드 방식으로 RDS 백업에 액세스하고 활성 트랜잭션 로그를 따라잡을 때까지 후속 백업 로그에서 읽습니다. DMS가 로그 백업에 액세스할 수 있도록 하려면 RDS 자동 백업 보존 기간을 최소 1일로 설정합니다. 자동 백업 보존 기간 설정에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [백업 보존 기간](#)을 참조하세요.

로그 백업에 액세스하는 DMS 작업은 RDS 인스턴스의 스토리지를 활용합니다. 태스크는 복제에 필요한 로그 백업에만 액세스합니다. Amazon RDS는 다운로드한 백업을 몇 시간 내에 제거합니다. 이 제거는 Amazon S3 또는 Amazon RDS RESTORE DATABASE 기능에 보관된 Amazon RDS 백업에는 영향을 주지 않습니다. DMS를 사용하여 복제하려는 경우 RDS for SQL Server 소스에 추가 스토리지를 할당하는 것이 좋습니다. 필요한 스토리지 양을 추정하는 한 가지 방법은 DMS가 복제를 시작하거나 재개할 백업을 식별하고 RDS tlog backup 메타데이터 함수를 사용하여 모든 후속 백업의 파일 크기를 추가하는 것입니다. tlog backup 함수에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [사용 가능한 트랜잭션 로그 백업 나열](#)을 참조하세요.

또는 Amazon RDS 인스턴스의 CloudWatch FreeStorageSpace 지표를 기반으로 스토리지 자동 크기 조정을 활성화하거나 스토리지 크기 조정을 트리거하도록 선택할 수 있습니다.

트랜잭션 로그 백업의 너무 먼 지점에서 시작하거나 재개하지 않는 것이 좋습니다. SQL Server 인스턴스에 스토리지가 채워질 수 있습니다. 이러한 경우 전체 로드를 시작하는 것이 좋습니다. 트랜잭션 로그 백업에서 복제하는 속도가 활성 트랜잭션 로그에서 읽는 것보다 느립니다. 자세한 내용은 [RDS for SQL Server에 대한 트랜잭션 로그 백업 처리](#) 단원을 참조하십시오.

로그 백업에 액세스하려면 추가 권한이 필요합니다. 자세한 내용은 [클라우드 SQL Server 데이터베이스에서 지속적인 복제를 위한 권한 설정](#) 섹션의 설명을 참조하세요. 태스크 복제를 시작하기 전에 이러한 권한을 부여했는지 확인하세요.

For AWS DMS 3.5.2 이하

Amazon RDS for SQL Server를 소스로 사용하는 경우, MS-CDC 캡처 작업은 `maxscans` 및 `maxtrans` 파라미터를 사용합니다. 이러한 파라미터는 MS-CDC 캡처가 트랜잭션 로그에서 수행하는 최대 스캔 수와 각 스캔별로 처리되는 트랜잭션 수를 제어합니다.

트랜잭션 수가 `maxtrans*maxscans`보다 많은 데이터베이스의 경우, `polling_interval` 값을 늘리면 활성 트랜잭션 로그 레코드가 누적될 수 있습니다. 결과적으로 이러한 누적으로 인해 트랜잭션 로그 크기가 증가할 수 있습니다.

AWS DMS는 MS-CDC 캡처 작업에 의존하지 않습니다. MS-CDC 캡처 작업은 트랜잭션 로그 항목을 처리된 것으로 표시합니다. 이렇게 하면 트랜잭션 로그 백업 작업이 트랜잭션 로그에서 항목을 제거할 수 있습니다.

트랜잭션 로그의 크기와 MS-CDC 작업의 성공 여부를 모니터링하는 것이 좋습니다. MS-CDC 작업이 실패하면 트랜잭션 로그가 과도하게 증가하여 AWS DMS 복제가 실패할 수 있습니다. 소스 데이터베이스의 `sys.dm_cdc_errors` 동적 관리 뷰를 사용하여 MS-CDC 캡처 작업 오류를 모니터링할 수 있습니다. `DBCC SQLPERF(LOGSPACE)` 관리 명령을 사용하여 트랜잭션 로그 크기를 모니터링할 수 있습니다.

MS-CDC로 인한 트랜잭션 로그 증가를 해결하려면

1. Log Space Used % 데이터베이스 AWS DMS 가에서 복제되고 있는지 확인하고 지속적으로 증가하는지 확인합니다.

```
DBCC SQLPERF(LOGSPACE)
```

2. 트랜잭션 로그 백업 프로세스를 방해하는 요인을 식별합니다.

```
Select log_reuse_wait, log_reuse_wait_desc, name from sys.databases where name = db_name();
```

`log_reuse_wait_desc` 값이 `REPLICATION`과 같으면 MS-CDC의 지연 시간으로 인해 로그 백업 보존이 발생합니다.

3. `maxtrans` 및 `maxscans` 파라미터 값을 늘려 캡처 작업에서 처리되는 이벤트 수를 늘립니다.

```
EXEC sys.sp_cdc_change_job @job_type = 'capture' ,@maxtrans = 5000, @maxscans = 20
exec sp_cdc_stop_job 'capture'
exec sp_cdc_start_job 'capture'
```

이 문제를 해결하려면 `maxtrans*maxscans`가 매일 소스 데이터베이스에서 AWS DMS 복제되는 테이블에 대해 생성된 평균 이벤트 수와 같도록 `maxtrans` 및 `maxscans` 값을 설정합니다.

이러한 파라미터를 권장 값보다 높게 설정하면 캡처 작업은 트랜잭션 로그의 모든 이벤트를 처리합니다. 이러한 파라미터를 권장 값보다 낮게 설정하면 MS-CDC 지연 시간이 늘어나고 트랜잭션 로그가 증가합니다.

워크로드의 변경으로 이벤트 수가 달라지기 때문에 적절한 `maxscans` 값과 `maxtrans` 값을 식별하는 것이 어려울 수 있습니다. 이 경우 MS-CDC 지연 시간에 대한 모니터링을 설정하는 것이 좋습니다. 자세한 내용은 SQL Server 설명서에서 [Monitor the process](#)를 참조하세요. 그런 다음 모니터링 결과를 기반으로 `maxtrans`와 `maxscans`를 동적으로 구성합니다.

AWS DMS 작업이 작업을 재개하거나 계속하는 데 필요한 로그 시퀀스 번호(LSNs)를 찾을 수 없는 경우 작업이 실패하고 전체 재로드가 필요할 수 있습니다.

Note

AWS DMS 를 사용하여 RDS for SQL Server 소스에서 데이터를 복제하는 경우 Amazon RDS 인스턴스의 중지-시작 이벤트 후 복제를 재개하려고 할 때 오류가 발생할 수 있습니다. 이는 중지-시작 이벤트 후 다시 시작할 때 SQL Server Agent 프로세스가 캡처 작업 프로세스를 다시 시작하기 때문입니다. 이는 MS-CDC 폴링 간격을 우회합니다.

따라서 트랜잭션 볼륨이 MS-CDC 캡처 작업 처리보다 낮은 데이터베이스에서는가 중지된 위치에서 재개하기 전에 데이터가 처리되거나 복제 및 백업된 것으로 표시되어 다음 오류가 발생할 AWS DMS 수 있습니다.

```
[SOURCE_CAPTURE ]E: Failed to access LSN '0000dbd9:0006f9ad:0003' in
the backup log sets since BACKUP/LOG-s are not available. [1020465]
(sqlserver_endpoint_capture.c:764)
```

이 문제를 완화하려면 앞서 권장한 대로 `maxtrans` 및 `maxscans` 값을 설정하세요.

Microsoft Azure SQL 데이터베이스를의 소스로 사용 AWS DMS

를 사용하면 Microsoft Azure SQL 데이터베이스를 SQL Server와 거의 동일한 방식으로 소스로 사용할 수 있습니다. 온프레미스 또는 Amazon EC2 인스턴스에서 실행되는 SQL Server에 대해 지원되는 것과 동일한 데이터베이스 버전 목록을 소스로 AWS DMS 지원합니다.

자세한 내용은 [Microsoft SQL Server 데이터베이스를의 소스로 사용 AWS DMS](#) 단원을 참조하십시오.

Note

AWS DMS 는 Azure SQL Database에서 변경 데이터 캡처 작업(CDC)을 지원하지 않습니다.

Microsoft Azure SQL 관리형 인스턴스의 소스로 사용 AWS DMS

를 사용하면 SQL Server와 거의 동일한 방식으로 Microsoft Azure SQL 관리형 인스턴스를 소스로 사용할 AWS DMS 수 있습니다. 온프레미스 또는 Amazon EC2 인스턴스에서 실행되는 SQL Server에 대해 지원되는 것과 동일한 데이터베이스 버전 목록을 소스로 AWS DMS 지원합니다.

자세한 내용은 [Microsoft SQL Server 데이터베이스의 소스로 사용 AWS DMS](#) 단원을 참조하십시오.

Microsoft Azure Database for PostgreSQL 유연한 서버의 소스로 사용 AWS DMS

를 사용하면 PostgreSQL과 거의 동일한 방식으로 Microsoft Azure Database for PostgreSQL 유연한 서버를 소스로 사용할 AWS DMS 수 있습니다.

가 소스로 AWS DMS 지원하는 Microsoft Azure Database for PostgreSQL 유연한 서버 버전에 대한 자세한 내용은 [섹션을 참조하세요](#) [의 소스 AWS DMS](#).

논리적 복제 및 디코딩을 위한 Microsoft Azure Database for PostgreSQL 유연한 서버 설정

데이터베이스 마이그레이션 중에 Microsoft Azure Database for PostgreSQL 유연한 서버에서 논리적 복제 및 디코딩 기능을 사용할 수 있습니다.

논리적 디코딩을 위해 DMS는 `test_decoding` 또는 `pglogical` 플러그인 중 하나를 사용합니다. 소스 PostgreSQL 데이터베이스에서 `pglogical` 플러그인을 사용할 수 있는 경우, DMS는 `pglogical`을 사용하여 복제 슬롯을 생성합니다. 그렇지 않을 경우에는 `test_decoding` 플러그인을 사용합니다.

Microsoft Azure Database for PostgreSQL 유연한 서버를 DMS의 소스 엔드포인트로 구성하려면 다음 단계를 수행합니다.

1. 포털에서 서버 파라미터 페이지를 엽니다.
2. `wal_level` 서버 파라미터를 LOGICAL로 설정합니다.

3. pglogical 확장을 사용하려면 `shared_preload_libraries` 및 `azure.extensions` 파라미터를 `pglogical`로 설정합니다.
4. `max_replication_slots` 파라미터를 동시에 실행하려는 최대 DMS 태스크 수로 설정합니다. Microsoft Azure의 경우, 이 파라미터의 기본값은 10입니다. 이 파라미터의 최대값은 PostgreSQL 인스턴스의 사용 가능한 메모리에 따라 달라지며, 메모리의 GB당 2~8개의 복제 슬롯이 허용됩니다.
5. `max_wal_senders` 파라미터를 1보다 큰 값으로 설정합니다. `max_wal_senders` 파라미터는 실행 가능한 동시 작업 개수를 설정합니다. 기본값은 10입니다.
6. `max_worker_processes` 파라미터 값을 최소 16으로 설정합니다. 그렇지 않을 경우, 다음과 같은 오류가 발생할 수 있습니다.

WARNING: out of background worker slots.

7. 변경 사항을 저장합니다. 서버를 재시작하여 변경 사항을 적용합니다.
8. PostgreSQL 인스턴스가 연결 리소스의 네트워크 트래픽을 허용하는지 확인합니다.
9. 아래의 명령을 사용하여 기존 사용자에게 복제 권한을 부여하거나, 복제 권한이 있는 새 사용자를 생성합니다.

- 아래의 명령을 사용하여 기존 사용자에게 복제 권한을 부여합니다.

```
ALTER USER <existing_user> WITH REPLICATION;
```

- 아래의 명령을 사용하여 복제 권한이 있는 새 사용자를 생성합니다.

```
CREATE USER aws_dms_user PASSWORD 'aws_dms_user_password';
GRANT azure_pg_admin to aws_dms_user;
ALTER ROLE aws_dms_user REPLICATION LOGIN;
```

PostgreSQL을 사용한 논리적 복제에 대한 자세한 내용은 아래 주제를 참조하세요.

- [논리적 복제를 사용하여 변경 데이터 캡처\(CDC\) 활성화](#)
- [네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정](#)
- [Azure Database for PostgreSQL 설명서의 Logical replication and logical decoding in Azure Database for PostgreSQL - Flexible Server.](#)

Microsoft Azure Database for MySQL 유연한 서버를의 소스로 사용 AWS DMS

를 사용하면 MySQL과 거의 동일한 방식으로 Microsoft Azure Database for MySQL 유연한 서버를 소스로 사용할 AWS DMS 수 있습니다.

가 소스로 AWS DMS 지원하는 Microsoft Azure Database for MySQL 유연한 서버 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

에서 고객 관리형 MySQL 호환 데이터베이스를 사용하는 방법에 대한 자세한 내용은 섹션을 AWS DMS 참조하세요 [자체 관리형 MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS](#).

Azure MySQL을의 소스로 사용할 때의 제한 사항 AWS Database Migration Service

- Azure MySQL 유연한 서버 시스템 변수의 기본값은 `sql_generate_invisible_primary_key`이며 ON 서버는 생성된 보이지 않는 기본 키(GIPK)를 명시적 기본 키 없이 생성된 테이블에 자동으로 추가합니다. AWS DMS 는 GIPK 제약 조건이 있는 MySQL 테이블에 대한 지속적인 복제를 지원하지 않습니다.

OCI MySQL Heatwave를의 소스로 사용 AWS DMS

를 사용하면 MySQL과 거의 동일한 방식으로 OCI MySQL Heatwave를 소스로 사용할 AWS DMS 수 있습니다. OCI MySQL Heatwave를 소스로 사용하려면 몇 가지 추가 구성을 변경해야 합니다.

가 소스로 AWS DMS 지원하는 OCI MySQL Heatwave 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

논리적 복제를 위한 OCI MySQL Heatwave 설정

OCI MySQL Heatwave 인스턴스를 DMS의 소스 엔드포인트로 구성하려면 다음을 수행합니다.

1. OCI 콘솔에 로그인하고 왼쪽 상단의 기본 햄버거 메뉴(≡)를 엽니다.
2. 데이터베이스, DB 시스템을 선택합니다.
3. 구성 메뉴를 엽니다.
4. 구성 생성을 선택합니다.
5. 구성 이름(예: **dms_configuration**)을 입력합니다.
6. 현재 OCI MySQL Heatwave 인스턴스의 형태를 선택합니다. DB 시스템 구성:형태 섹션 아래에 있는 인스턴스의 DB 시스템 구성 속성 탭에서 형태를 찾을 수 있습니다.

7. 시스템 변수 섹션에서 `binlog_row_value_options` 시스템 변수를 선택합니다. 기본값은 `PARTIAL_JSON`입니다. 값을 지웁니다.
8. 생성 버튼을 선택합니다.
9. OCI MySQLHeatwave 인스턴스를 열고 편집 버튼을 선택합니다.
10. 구성 섹션에서 구성 변경 버튼을 선택하고, 4단계에서 생성한 형태 구성을 선택합니다.
11. 변경 사항이 적용되면 인스턴스는 논리적 복제를 수행할 준비가 완료됩니다.

MySQL용 Google Cloud를의 소스로 사용 AWS DMS

를 사용하면 MySQL과 거의 동일한 방식으로 Google Cloud for MySQL을 소스로 사용할 AWS DMS 수 있습니다.

가 소스로 AWS DMS 지원하는 GCP MySQL 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

자세한 내용은 [MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS](#) 단원을 참조하십시오.

Note

소스로서 GCP MySQL 8.0에 대한 지원은 AWS DMS 버전 3.4.6에서 사용할 수 있습니다. AWS DMS 는 MySQL 인스턴스 `verify-full`용 GCP에 대한 SSL 모드를 지원하지 않습니다. GCP MySQL 보안 설정은 서버 및 클라이언트 인증서 확인이 모두 필요하므로 지원되지 `Allow only SSL connections` 않습니다.는 서버 인증서 확인 AWS DMS 만 지원합니다. AWS DMS 는 `binlog_checksum` 데이터베이스 플래그에 CRC32에 대한 기본 GCP CloudSQL for MySQL 값을 지원합니다.

Google Cloud for PostgreSQL을의 소스로 사용 AWS DMS

를 사용하면 자체 관리형 PostgreSQL 데이터베이스를 수행하는 것과 거의 동일한 방식으로 Google Cloud for PostgreSQL을 소스로 사용할 AWS DMS 수 있습니다.

가 소스로 AWS DMS 지원하는 GCP PostgreSQL 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

자세한 내용은 [PostgreSQL 데이터베이스를 AWS DMS 소스로 사용](#) 단원을 참조하십시오.

논리적 복제 및 디코딩을 위해 Google Cloud for PostgreSQL 설정

데이터베이스 마이그레이션 중에 Google Cloud SQL for PostgreSQL에서 논리적 복제 및 디코딩 기능을 사용할 수 있습니다.

논리적 디코딩을 위해 DMS는 다음 플러그인 중 하나를 사용합니다.

- `test_decoding`
- `pglogical`

소스 PostgreSQL 데이터베이스에서 `pglogical` 플러그인을 사용할 수 있는 경우, DMS는 `pglogical`을 사용하여 복제 슬롯을 생성합니다. 그렇지 않을 경우에는 `test_decoding` 플러그인을 사용합니다.

에서 논리적 디코딩을 사용하는 방법은 AWS DMS 다음과 같습니다.

1. Google Cloud SQL for PostgreSQL을 사용할 경우, `cloudsql.logical_decoding` 플래그를 `on`으로 설정하여 논리적 디코딩을 활성화합니다.
2. `pglogical`을 활성화하려면 `cloudsql.enable_pglogical` 플래그를 `on`으로 설정하고, 데이터베이스를 다시 시작합니다.
3. 논리적 디코딩 기능을 사용하려면 REPLICATION 속성을 보유한 PostgreSQL 사용자를 생성합니다. `pglogical` 확장을 사용할 경우, 사용자에게 `cloudsqlsuperuser` 역할이 있어야 합니다. `cloudsqlsuperuser` 역할이 있는 사용자를 생성하려면 다음을 수행합니다.

```
CREATE USER new_aws_dms_user WITH REPLICATION
IN ROLE cloudsqlsuperuser LOGIN PASSWORD 'new_aws_dms_user_password';
```

기존 사용자에게 이 속성을 설정하려면 다음 작업을 수행합니다.

```
ALTER USER existing_user WITH REPLICATION;
```

4. `max_replication_slots` 파라미터를 동시에 실행하려는 최대 DMS 태스크 수로 설정합니다. Google Cloud SQL에서 이 파라미터의 기본값은 10입니다. 이 파라미터의 최대값은 PostgreSQL 인스턴스의 사용 가능한 메모리에 따라 달라지며, 메모리의 GB당 2~8개의 복제 슬롯이 허용됩니다.

PostgreSQL을 사용한 논리적 복제에 대한 자세한 내용은 아래 주제를 참조하세요.

- [논리적 복제를 사용하여 변경 데이터 캡처\(CDC\) 활성화](#)
- [네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정](#)
- [Cloud SQL for PostgreSQL 설명서의 논리 복제 및 디코딩 설정](#).

PostgreSQL 데이터베이스를 AWS DMS 소스로 사용

를 사용하여 하나 이상의 PostgreSQL 데이터베이스에서 데이터를 마이그레이션할 수 있습니다 AWS DMS. PostgreSQL 데이터베이스를 소스로 사용하여 다른 PostgreSQL 데이터베이스나 지원되는 다른 데이터베이스 중 하나로 데이터를 마이그레이션할 수 있습니다.

가 소스로 AWS DMS 지원하는 PostgreSQL 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

AWS DMS 는 다음과 같은 유형의 데이터베이스에 대해 PostgreSQL을 지원합니다.

- 온프레미스 데이터베이스
- Amazon EC2 인스턴스의 데이터베이스
- Amazon RDS DB 인스턴스의 데이터베이스
- Amazon Aurora PostgreSQL 호환 버전 기반 DB 인스턴스의 데이터베이스
- Amazon Aurora PostgreSQL 호환 서버리스 버전 기반 DB 인스턴스의 데이터베이스

Note

DMS는 전체 로드와 함께 Amazon Aurora PostgreSQL Serverless V1을 소스로 지원합니다. 하지만 Amazon Aurora PostgreSQL Serverless V2는 전체 로드, 전체 로드 및 CDC, CDC 전용 작업의 소스로 사용할 수 있습니다.

Secure Sockets Layer(SSL)을 사용하여 PostgreSQL 엔드포인트와 복제 인스턴스 간 연결을 암호화할 수 있습니다. PostgreSQL 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 [에서 SSL 사용 AWS Database Migration Service](#) 섹션을 참조하세요.

PostgreSQL을 소스로 사용 시 추가 보안 요구 사항으로, 지정된 사용자 계정은 PostgreSQL 데이터베이스에서 등록된 사용자여야 합니다.

PostgreSQL 데이터베이스를 AWS DMS 소스 엔드포인트로 구성하려면 다음을 수행합니다.

- PostgreSQL 소스 데이터베이스에 대한 AWS DMS 액세스를 제공할 수 있는 적절한 권한이 있는 PostgreSQL 사용자를 생성합니다.

Note

- PostgreSQL 소스 데이터베이스가 자체 관리형인 경우, 자세한 내용은 [에서 소스로 자체 관리형 PostgreSQL 데이터베이스 작업 AWS DMS](#) 섹션을 참조하세요.
- PostgreSQL 소스 데이터베이스를 Amazon RDS가 관리하는 경우, 자세한 내용은 [AWS관리형 PostgreSQL 데이터베이스를 DMS 소스로 사용](#) 섹션을 참조하세요.

- 선택한 PostgreSQL 데이터베이스 구성을 준수하는 PostgreSQL 소스 엔드포인트를 생성합니다.
- 작업 또는 작업 세트를 생성하여 테이블을 마이그레이션합니다.

전체 로드 전용 작업을 생성하기 위한 추가 엔드포인트 구성이 필요하지 않습니다.

변경 데이터 캡처 작업(CDC 전용 또는 전체 로드 및 CDC 작업)을 생성하기 전에 [자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화](#) 또는 [를 사용하여 AWS관리형 PostgreSQL DB 인스턴스로 CDC 활성화 AWS DMS](#) 섹션을 참조하세요.

주제

- [에서 소스로 자체 관리형 PostgreSQL 데이터베이스 작업 AWS DMS](#)
- [AWS관리형 PostgreSQL 데이터베이스를 DMS 소스로 사용](#)
- [논리적 복제를 사용하여 변경 데이터 캡처\(CDC\) 활성화](#)
- [네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정](#)
- [를 사용하여 PostgreSQL에서 PostgreSQL로 마이그레이션 AWS DMS](#)
- [를 사용하여 Babelfish for Amazon Aurora PostgreSQL에서 마이그레이션 AWS DMS](#)
- [PostgreSQL 소스 데이터베이스에서 AWS DMS 아티팩트 제거](#)
- [PostgreSQL 데이터베이스를 DMS 소스로 사용 시 추가 구성 설정](#)
- [읽기 전용 복제본을 PostgreSQL의 소스로 사용](#)
- [MapBooleanAsBoolean PostgreSQL 엔드포인트 설정 사용](#)
- [DMS 소스로 PostgreSQL 사용 시 엔드포인트 설정 및 추가 연결 속성\(ECA\)](#)
- [PostgreSQL 데이터베이스를 DMS 소스로 사용 시 제한 사항](#)
- [PostgreSQL용 소스 데이터 형식](#)

에서 소스로 자체 관리형 PostgreSQL 데이터베이스 작업 AWS DMS

자체 관리형 PostgreSQL 데이터베이스를 소스로 사용하면 다른 PostgreSQL 데이터베이스 또는에서 지원하는 다른 대상 데이터베이스 중 하나로 데이터를 마이그레이션할 수 있습니다 AWS DMS. 온프레미스 데이터베이스나 Amazon EC2 인스턴스에서 실행되는 자체 관리형 엔진이 데이터베이스 소스가 될 수 있습니다. 전체 로드 작업 및 변경 데이터 캡처(CDC) 작업 모두에 DB 인스턴스를 사용할 수 있습니다.

자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하기 위한 사전 조건

자체 관리형 PostgreSQL 소스 데이터베이스에서 데이터를 마이그레이션하기 전에 다음을 수행합니다.

- 버전 9.4.x 이상의 PostgreSQL 데이터베이스를 사용해야 합니다.
- 전체 로드 및 변경 데이터 캡처(CDC) 작업 또는 CDC 전용 작업의 경우, PostgreSQL 소스 데이터베이스에 지정된 사용자 계정에 슈퍼유저 권한을 부여합니다. 사용자 계정이 소스의 복제 관련 함수에 액세스하려면 슈퍼유저 권한이 필요합니다. DMS 사용자 계정은 테이블을 성공적으로 마이그레이션하려면 모든 열에 대한 SELECT 권한이 필요합니다. 열에 대한 권한이 누락된 경우 DMS는 일반 DMS 데이터 형식 매핑을 사용하여 대상 테이블을 생성하여 메타데이터 차이와 작업 실패를 초래합니다.
- AWS DMS 복제 서버의 IP 주소를 pg_hba.conf 구성 파일에 추가하고 복제 및 소켓 연결을 활성화합니다. 예를 들면 다음과 같습니다.

```
# Replication Instance
host all all 12.3.4.56/00 md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
host replication dms 12.3.4.56/00 md5
```

PostgreSQL의 pg_hba.conf 구성 파일은 클라이언트 인증을 제어합니다. (HBA는 호스트 기반 인증을 의미합니다.) 이 파일은 일반적으로 데이터베이스 클러스터의 데이터 디렉터리에 저장됩니다.

- 를 사용하여 데이터베이스를 논리적 복제의 소스로 구성하는 경우 섹션을 AWS DMS 참조하세요.

[자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화](#)

Note

일부 AWS DMS 트랜잭션은 DMS 엔진이 다시 사용하기 전에 잠시 유향 상태입니다. PostgreSQL 버전 9.6 이상에서 `idle_in_transaction_session_timeout` 파라미터를 사용하면 유향 트랜잭션이 시간 초과되어 실패하도록 할 수 있습니다. AWS DMS를 사용할 때는 유향 트랜잭션을 종료하지 마세요.

자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화

AWS DMS 는 논리적 복제를 사용하여 변경 데이터 캡처(CDC)를 지원합니다. 자체 관리형 PostgreSQL 소스 데이터베이스의 논리적 복제를 활성화하려면 `postgresql.conf` 구성 파일에서 다음 파라미터와 값을 설정합니다.

- `wal_level = logical`을 설정합니다.
- `max_replication_slots`를 1보다 큰 값으로 설정합니다.

실행할 작업 개수에 따라 `max_replication_slots` 값을 설정합니다. 예를 들어, 5개 작업을 실행하려면 최소 5개의 슬롯을 설정해야 합니다. 작업이 시작된 즉시 슬롯이 자동으로 열리고 작업이 더 이상 실행되지 않더라도 계속 열려 있습니다. 열린 슬롯은 반드시 수동으로 삭제하세요. 참고로 DMS에서 슬롯을 생성한 경우 작업이 삭제되면 DMS가 복제 슬롯을 자동으로 삭제합니다.

- `max_wal_senders`를 1보다 큰 값으로 설정합니다.

`max_wal_senders` 파라미터는 실행 가능한 동시 작업 개수를 설정합니다.

- `wal_sender_timeout` 파라미터는 지정된 밀리초보다 오래 사용되지 않는 복제 연결을 종료합니다. 온프레미스 PostgreSQL 데이터베이스의 기본값은 60,000밀리초(60초)입니다. 값을 0으로 설정하면 시간 제한 메커니즘이 비활성화되며, 이는 유효한 DMS 설정입니다.

`wal_sender_timeout`을 0이 아닌 값으로 설정하는 경우, CDC가 있는 DMS 작업에는 최소 10,000밀리초(10초)가 필요하며, 값이 10,000보다 작으면 작업이 실패합니다. DMS 복제 인스턴스의 다중 AZ 장애 조치 중에 지연이 발생하지 않도록 하려면 값을 5분 미만으로 유지하세요.

일부 파라미터는 정적이며 서버 시작 시에만 설정할 수 있습니다. 구성 파일(자체 관리형 데이터베이스의 경우) 또는 DB 파라미터 그룹(RDS for PostgreSQL 데이터베이스의 경우)의 항목에 대한 모든 변경 사항은 서버를 다시 시작할 때까지 무시됩니다. 자세한 내용은 [PostgreSQL 설명서](#)를 참조하세요.

CDC 활성화에 대한 자세한 내용은 [논리적 복제를 사용하여 변경 데이터 캡처\(CDC\) 활성화](#) 섹션을 참조하세요.

AWS관리형 PostgreSQL 데이터베이스를 DMS 소스로 사용

AWS관리형 PostgreSQL DB 인스턴스를 소스로 사용할 수 있습니다 AWS DMS. AWS관리형 PostgreSQL 소스를 사용하여 전체 로드 작업과 변경 데이터 캡처(CDC) 작업을 모두 수행할 수 있습니다.

AWS관리형 PostgreSQL 데이터베이스를 DMS 소스로 사용하기 위한 사전 조건

AWS관리형 PostgreSQL 소스 데이터베이스에서 데이터를 마이그레이션하기 전에 다음을 수행합니다.

- PostgreSQL DB 인스턴스에 필요한 최소 권한이 있는 AWS 사용자 계정을 PostgreSQL 소스 엔드 포인트의 사용자 계정으로 사용하는 것이 좋습니다 AWS DMS. 마스터 계정을 사용하는 것은 권장되지 않습니다. 계정에는 `rds_superuser` 역할과 `rds_replication` 역할이 있어야 합니다. `rds_replication` 역할은 논리적 슬롯을 관리하고 논리적 슬롯을 사용하여 데이터를 스트리밍할 수 있는 권한을 부여합니다.

사용하는 계정의 마스터 사용자 계정에서 여러 객체를 생성해야 합니다. 객체 생성에 대한 자세한 내용은 [마스터 사용자 계정을 사용하지 않고 Amazon RDS for PostgreSQL 데이터베이스 마이그레이션](#) 섹션을 참조하세요.

- 소스 데이터베이스가 가상 프라이빗 클라우드(VPC)에 있는 경우, 데이터베이스가 상주하는 DB 인스턴스에 대한 액세스 권한을 부여하는 VPC 보안 그룹을 선택합니다. 이는 DMS 복제 인스턴스를 소스 DB 인스턴스에 성공적으로 연결하는 데 필요합니다. 데이터베이스와 DMS 복제 인스턴스가 동일한 VPC에 있는 경우, 적절한 보안 그룹을 자체 인바운드 규칙에 추가합니다.

Note

일부 AWS DMS 트랜잭션은 DMS 엔진이 다시 사용하기 전에 잠시 유향 상태입니다.

PostgreSQL 버전 9.6 이상에서 `idle_in_transaction_session_timeout` 파라미터를 사용하면 유향 트랜잭션이 시간 초과되어 실패하도록 할 수 있습니다. AWS DMS를 사용할 때는 유향 트랜잭션을 종료하지 마세요.

를 사용하여 AWS관리형 PostgreSQL DB 인스턴스로 CDC 활성화 AWS DMS

AWS DMS 는 DB 인스턴스가 논리적 복제를 사용하도록 구성된 경우 Amazon RDS PostgreSQL 데이터베이스에서 CDC를 지원합니다. 다음 표에는 각 AWS관리형 PostgreSQL 버전의 논리적 복제 호환성이 요약되어 있습니다.

PostgreSQL 버전	AWS DMS 전체 로드 지원	AWS DMS CDC 지원
PostgreSQL 10.5(또는 이하)와 호환되는 Aurora PostgreSQL 버전 2.1	예	아니요
PostgreSQL 10.6(또는 이상)과 호환되는 Aurora PostgreSQL 버전 2.2	예	예
PostgreSQL 10.21(또는 이상)과 호환되는 RDS for PostgreSQL	예	예

RDS PostgreSQL DB 인스턴스에 대한 논리적 복제를 활성화하려면

1. PostgreSQL DB 인스턴스의 AWS 마스터 사용자 계정을 PostgreSQL 소스 엔드포인트의 사용자 계정으로 사용합니다. 마스터 사용자 계정에는 CDC 설정을 허용하는 데 필요한 역할이 있습니다.

마스터 사용자 계정 이외의 계정을 사용하는 경우, 사용하는 계정의 마스터 사용자 계정에서 여러 객체를 생성해야 합니다. 자세한 내용은 [마스터 사용자 계정을 사용하지 않고 Amazon RDS for PostgreSQL 데이터베이스 마이그레이션](#) 단원을 참조하십시오.

2. DB 클러스터 파라미터 그룹의 `rds.logical_replication` 파라미터를 1로 설정합니다. 이 정적 파라미터를 적용하려면 DB 인스턴스를 재부팅해야 합니다. 이 파라미터를 적용하는 중에 AWS DMS에서는 `wal_level`, `max_wal_senders`, `max_replication_slots`, `max_connections` 파라미터를 설정합니다. 이러한 파라미터 변경은 WAL(Write Ahead Log) 생성을 강화하므로 논리적 복제 슬롯을 사용할 때 `rds.logical_replication`만 설정하면 됩니다.
3. `wal_sender_timeout` 파라미터는 지정된 밀리초보다 오래 사용되지 않는 복제 연결을 종료합니다. AWS관리형 PostgreSQL 데이터베이스의 기본값은 30,000밀리초(30초)입니다. 값을 0으로 설정하면 시간 제한 메커니즘이 비활성화되며, 이는 유효한 DMS 설정입니다.

`wal_sender_timeout`을 0이 아닌 값으로 설정하는 경우, CDC가 있는 DMS 작업에는 최소 10,000밀리초(10초)가 필요하며, 값이 0에서 10,000 사이이면 작업이 실패합니다. DMS 복제 인스턴스의 다중 AZ 장애 조치 중에 지연이 발생하지 않도록 하려면 값을 5분 미만으로 유지하세요.

4. DB 클러스터 파라미터 그룹의 `max_worker_processes` 파라미터 값이 `max_logical_replication_workers`, `autovacuum_max_workers`, `max_parallel_workers`의 총 합계 값과 같거나 더 큰지 확인합니다. 백그라운드 작업자 프로세스 수가 많으면 소규모 인스턴스의 애플리케이션 워크로드에 영향을 미칠 수 있습니다. 따라서 `max_worker_processes`를 기본값보다 높게 설정한 경우, 데이터베이스의 성능을 모니터링하세요.
5. Aurora PostgreSQL을 CDC의 소스로 사용하는 경우 `synchronous_commit`를 ON으로 설정합니다.

CDC용 PostgreSQL MultiAZ DB 클러스터 읽기 전용 복제본을 사용하려면(지속 복제)

1. DB 클러스터 파라미터 그룹의 `rds.logical_replication` 및 `sync_replication_slots` 파라미터를 1로 설정합니다. 이 정적 파라미터를 적용하려면 DB 인스턴스를 재부팅해야 합니다.
2. 다음 명령을 실행하여 Writer에서 `awsdms_ddl_audit` 테이블을 생성하고를 사용할 스키마의 `objects_schema` 이름으로 바꿉니다.

```
CREATE TABLE objects_schema.awsdms_ddl_audit
(
  c_key      bigserial primary key,
  c_time     timestamp,      -- Informational
  c_user     varchar(64),    -- Informational: current_user
  c_txn      varchar(16),    -- Informational: current transaction
  c_tag      varchar(24),    -- Either 'CREATE TABLE' or 'ALTER TABLE' or 'DROP TABLE'
  c_oid      integer,       -- For future use - TG_OBJECTID
  c_name     varchar(64),    -- For future use - TG_OBJECTNAME
  c_schema   varchar(64),    -- For future use - TG_SCHEMANAME. For now - holds
  current_schema
  c_ddlqry   text           -- The DDL query associated with the current DDL event
);
```

3. 다음 명령을 실행하여 `awsdms_intercept_ddl` 함수를 생성하고 `objects_schema`를 다음 스키마의 이름으로 바꿔서 다음을 사용합니다.

```
CREATE OR REPLACE FUNCTION objects_schema.awsdms_intercept_ddl()
  RETURNS event_trigger
LANGUAGE plpgsql
SECURITY DEFINER
AS $$
  declare _qry text;
BEGIN
```

```

if (tg_tag='CREATE TABLE' or tg_tag='ALTER TABLE' or tg_tag='DROP TABLE' or
tg_tag = 'CREATE TABLE AS') then
    SELECT current_query() into _qry;
    insert into objects_schema.awsdms_ddl_audit
    values
    (
        default,current_timestamp,current_user,cast(TXID_CURRENT()as
varchar(16)),tg_tag,0,' ',current_schema,_qry
    );
    delete from objects_schema.awsdms_ddl_audit;
end if;
END;
$$;

```

4. 다음 명령을 실행하여 이벤트 트리거 awsdms_intercept_ddl을 생성합니다.

```

CREATE EVENT TRIGGER awsdms_intercept_ddl ON ddl_command_end EXECUTE PROCEDURE
objects_schema.awsdms_intercept_ddl();

```

이러한 이벤트에 액세스하는 모든 사용자와 역할에 필요한 DDL 권한이 있는지 확인합니다. 예제:

```

grant all on public.awsdms_ddl_audit to public;
grant all on public.awsdms_ddl_audit_c_key_seq to public;

```

5. 라이터에서 복제 슬롯 생성:

```

SELECT * FROM pg_create_logical_replication_slot('dms_read_replica_slot',
'test_decoding', false, true);

```

6. Reader에서 복제 슬롯을 사용할 수 있는지 확인합니다.

```

select * from pg_catalog.pg_replication_slots where slot_name =
'dms_read_replica_slot';

slot_name          |plugin          |slot_type|datoid|database|temporary|
active|active_pid|xmin|catalog_xmin|restart_lsn|confirmed_flush_lsn|
wal_status|safe_wal_size|two_phase|inactive_since          |conflicting|
invalidation_reason|failover|synced|
-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+

```

```
dms_read_replica_slot|test_decoding|logical | 5|postgres|false |false |
| |3559 |0/180011B8 |0/180011F0 |reserved |
|true |2025-02-10 15:45:04.083 +0100|false | |false |
false |
```

- 읽기 전용 복제본에 대한 DMS 소스 엔드포인트를 생성하고 추가 연결 속성을 통해 논리적 복제 슬롯 이름을 설정합니다.

```
slotName=dms_read_replica_slot;
```

- CDC/FL+CDC 작업을 생성하고 시작합니다.

Note

CDC/FL+CDC 마이그레이션의 경우 DMS는 작업 시작 시간을 CDC 시작 위치로 간주합니다. 복제 슬롯의 모든 이전 LSNs은 무시됩니다.

마스터 사용자 계정을 사용하지 않고 Amazon RDS for PostgreSQL 데이터베이스 마이그레이션

경우에 따라 소스로 사용 중인 Amazon RDS for PostgreSQL DB 인스턴스에 마스터 사용자 계정을 사용하지 못할 수 있습니다. 이러한 경우, 데이터 정의 언어(DDL) 이벤트를 캡처하기 위해 몇 개의 객체를 생성합니다. 마스터 계정 이외의 계정에서 이 객체를 생성하고 나서 마스터 사용자 계정에서 트리거를 생성합니다.

Note

소스 엔드포인트에서 CaptureDdlIs 엔드포인트 설정을 false로 설정하는 경우, 소스 데이터베이스에 다음 테이블 및 트리거를 생성할 필요가 없습니다.

다음 절차에 따라 이 객체를 생성합니다.

객체를 생성하려면

- 객체가 생성되는 위치에 있는 스키마를 선택합니다. 기본 스키마는 public입니다. 스키마가 있는지와 이 스키마를 *OtherThanMaster* 계정으로 액세스할 수 있는지 확인합니다.
- 마스터 계정이 아닌 사용자 계정(여기서는 *OtherThanMaster* 계정)을 사용하여 PostgreSQL DB 인스턴스에 로그인합니다.

3. 다음 명령을 실행하여 테이블 `awsdms_ddl_audit`를 생성하고, 다음 코드의 `objects_schema`를 사용할 스키마 이름으로 대체합니다.

```
CREATE TABLE objects_schema.awsdms_ddl_audit
(
  c_key      bigserial primary key,
  c_time     timestamp,      -- Informational
  c_user     varchar(64),    -- Informational: current_user
  c_txn      varchar(16),    -- Informational: current transaction
  c_tag      varchar(24),    -- Either 'CREATE TABLE' or 'ALTER TABLE' or 'DROP TABLE'
  c_oid      integer,        -- For future use - TG_OBJECTID
  c_name     varchar(64),    -- For future use - TG_OBJECTNAME
  c_schema   varchar(64),    -- For future use - TG_SCHEMA_NAME. For now - holds
  current_schema
  c_ddlqry   text            -- The DDL query associated with the current DDL event
);
```

4. 다음 코드의 `awsdms_intercept_ddl`를 사용할 스키마의 이름으로 바뀌어서 다음 명령을 실행하여 `objects_schema` 함수를 생성합니다.

```
CREATE OR REPLACE FUNCTION objects_schema.awsdms_intercept_ddl()
  RETURNS event_trigger
LANGUAGE plpgsql
SECURITY DEFINER
AS $$
  declare _qry text;
BEGIN
  if (tg_tag='CREATE TABLE' or tg_tag='ALTER TABLE' or tg_tag='DROP TABLE' or
  tg_tag = 'CREATE TABLE AS') then
    SELECT current_query() into _qry;
    insert into objects_schema.awsdms_ddl_audit
    values
    (
      default,current_timestamp,current_user,cast(TXID_CURRENT()as
      varchar(16)),tg_tag,0,'',current_schema,_qry
    );
    delete from objects_schema.awsdms_ddl_audit;
  end if;
END;
```

```
$$;
```

5. *OtherThanMaster* 계정에서 로그아웃하고 `rds_superuser` 역할이 할당된 계정으로 로그인합니다.
6. 다음 명령을 실행하여 이벤트 트리거 `awsdms_intercept_ddl`을 생성합니다.

```
CREATE EVENT TRIGGER awsdms_intercept_ddl ON ddl_command_end
EXECUTE PROCEDURE objects_schema.awsdms_intercept_ddl();
```

7. 이러한 이벤트에 액세스하는 모든 사용자와 역할에 필요한 DDL 권한이 있는지 확인합니다. 예제:

```
grant all on public.awsdms_ddl_audit to public;
grant all on public.awsdms_ddl_audit_c_key_seq to public;
```

이전 절차를 완료하면, *OtherThanMaster* 계정을 사용하여 AWS DMS 소스 엔드포인트를 생성할 수 있습니다.

Note

이러한 이벤트는 CREATE TABLE, ALTER TABLE, DROP TABLE 문에 의해 트리거됩니다.

논리적 복제를 사용하여 변경 데이터 캡처(CDC) 활성화

PostgreSQL의 네이티브 논리적 복제 기능을 사용하여 PostgreSQL 소스의 데이터베이스 마이그레이션 중에 변경 데이터 캡처(CDC)를 활성화할 수 있습니다. 이 기능은 자체 관리형 PostgreSQL 및 Amazon RDS for PostgreSQL SQL DB 인스턴스에 사용할 수 있습니다. 이 접근 방식은 가동 중지 시간을 줄이고 대상 데이터베이스를 소스 PostgreSQL 데이터베이스와 동기화하는 데 도움이 됩니다.

AWS DMS는 기본 키가 있는 PostgreSQL 테이블용 CDC를 지원합니다. 테이블에 프라이머리 키가 없는 경우, 미리 쓰기 로그(WAL)에 데이터베이스 행의 이전 이미지가 포함되지 않습니다. 이 경우 DMS는 테이블을 업데이트할 수 없습니다. 여기서는 추가 구성 설정과 테이블 복제본 ID를 해결 방법으로 사용할 수 있습니다. 하지만 이 접근 방식을 사용하면 추가 로그가 생성될 수 있습니다. 신중하게 테스트한 후에만 테이블 복제본 ID를 해결 방법으로 사용하는 것이 좋습니다. 자세한 내용은 [PostgreSQL 데이터베이스를 DMS 소스로 사용 시 추가 구성 설정](#) 단원을 참조하십시오.

Note

REPLICA IDENTITY FULL은 논리적 디코딩 플러그인에서는 지원되지만 pglogical 플러그인에서는 지원되지 않습니다. 자세한 내용은 [pglogical 설명서](#)를 참조하세요.

전체 로드 및 CDC 및 CDC 전용 작업의 경우는 논리적 복제 슬롯을 AWS DMS 사용하여 로그가 디코딩될 때까지 복제용 WAL 로그를 유지합니다. 전체 로드 및 CDC 작업 또는 CDC 작업을 다시 시작(재개가 아님)하면 복제 슬롯이 다시 생성됩니다.

Note

DMS는 논리적 디코딩에 test_decoding 또는 pglogical 플러그인을 사용합니다. 소스 PostgreSQL 데이터베이스에서 pglogical 플러그인을 사용할 수 있는 경우, DMS는 pglogical을 사용하여 복제 슬롯을 생성하고, 그렇지 않으면 test_decoding 플러그인을 사용합니다. test_decoding 플러그인에 대한 자세한 내용은 [PostgreSQL 설명서](#)를 참조하세요. 데이터베이스 파라미터 max_slot_wal_keep_size가 기본값이 아닌 값으로 설정되고, 복제 슬롯의 restart_lsn이 현재 LSN보다 이 크기 이상 모자라는 경우, 필요한 WAL 파일이 제거되기 때문에 DMS 작업이 실패합니다.

pglogical 플러그인 구성

PostgreSQL 확장으로 구현된 pglogical 플러그인은 선택적 데이터 복제를 위한 논리적 복제 시스템 및 모델입니다. 다음 표에는 pglogical 플러그인을 지원하는 소스 PostgreSQL 데이터베이스 버전이 나와 있습니다.

PostgreSQL 소스	pglogical 지원
자체 관리형 PostgreSQL 9.4 이상	예
Amazon RDS PostgreSQL 9.5 이하	아니요
Amazon RDS PostgreSQL 9.6 이상	예
Aurora PostgreSQL 1.x~2.5.x	아니요
Aurora PostgreSQL 2.6.x 이상	예

PostgreSQL 소스	pglogical 지원
Aurora PostgreSQL 3.3.x 이상	예

와 함께 사용하도록 pglogical을 구성하기 전에 AWS DMS 먼저 PostgreSQL 소스 데이터베이스에서 변경 데이터 캡처(CDC)에 대한 논리적 복제를 활성화합니다.

- 자체 관리형 PostgreSQL 소스 데이터베이스에서 CDC의 논리적 복제를 활성화하는 방법에 대한 자세한 내용은 [자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화](#) 섹션을 참조하세요.
- AWS 관리형 PostgreSQL 소스 데이터베이스에서 CDC의 논리적 복제를 활성화하는 방법에 대한 자세한 내용은 [를 사용하여 AWS 관리형 PostgreSQL DB 인스턴스로 CDC 활성화 AWS DMS](#) 섹션을 참조하세요.

PostgreSQL 소스 데이터베이스에서 논리적 복제를 활성화한 후 다음 단계를 사용하여 DMS에 사용할 pglogical을 구성합니다.

를 사용하여 PostgreSQL 소스 데이터베이스에서 논리적 복제에 pglogical 플러그인을 사용하려면 AWS DMS

1. 소스 PostgreSQL 데이터베이스에서 pglogical 확장을 생성합니다.
 - a. 올바른 파라미터를 설정하세요.
 - 자체 관리형 PostgreSQL 데이터베이스의 경우, 데이터베이스 파라미터 `shared_preload_libraries= 'pglogical'`을 설정합니다.
 - Amazon RDS 기반 PostgreSQL과 Amazon Aurora PostgreSQL 호환 버전 데이터베이스의 경우, 동일한 RDS 파라미터 그룹에서 `shared_preload_libraries` 파라미터를 `pglogical`로 설정합니다.
 - b. PostgreSQL 소스 데이터베이스를 다시 시작합니다.
 - c. PostgreSQL 데이터베이스에서 `create extension pglogical;` 명령을 실행합니다.
2. 다음 명령을 실행하여 pglogical이 성공적으로 설치되었는지 확인합니다.

```
select * FROM pg_catalog.pg_extension
```

이제 PostgreSQL 소스 데이터베이스 엔드포인트에 대한 변경 데이터 캡처를 수행하는 AWS DMS 작업을 생성할 수 있습니다.

Note

PostgreSQL 소스 데이터베이스에서 `pglogical`을 활성화하지 않으면 AWS DMS 는 기본적으로 `test_decoding` 플러그인을 사용합니다. 논리적 디코딩에 `pglogical`이 활성화된 경우 기본적으로 `pglogical`을 AWS DMS 사용합니다. 하지만 `test_decoding` 플러그인을 대신 사용하도록 추가 연결 속성 `PluginName`을 설정할 수 있습니다.

네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정

PostgreSQL을 소스로 사용하여 네이티브 CDC 시작 지점을 활성화하려면 엔드포인트를 생성할 때 `slotName` 추가 연결 속성을 기존 논리적 복제 슬롯의 이름으로 설정합니다. 이 논리적 복제 슬롯에는 엔드포인트 생성 시점부터 지속적인 변경 사항이 포함되므로 이전 시점의 복제를 지원합니다.

PostgreSQL은 AWS DMS 가 논리적 복제 슬롯에서 변경 사항을 성공적으로 읽은 후에만 삭제되는 WAL 파일에 데이터베이스 변경 사항을 기록합니다. 논리적 복제 슬롯을 사용하면 기록된 변경 사항이 복제 엔진에서 사용되기 전에 삭제되지 않도록 보호할 수 있습니다.

그러나 변경 비율과 사용 비율에 따라 논리적 복제 슬롯에 변경 사항이 포함되어 디스크 사용량이 증가할 수 있습니다. 논리적 복제 슬롯을 사용할 때 소스 PostgreSQL 인스턴스에서 공간 사용 경보를 설정하는 것이 좋습니다. `slotName` 추가 연결 속성 설정에 대한 자세한 내용은 [DMS 소스로 PostgreSQL 사용 시 엔드포인트 설정 및 추가 연결 속성\(ECA\)](#) 섹션을 참조하세요.

다음 절차에서는 이 접근 방식에 대해 자세히 설명합니다.

고유 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드를 설정하려면

1. 시작 지점으로 사용할 이전 복제 작업(상위 작업)에서 사용하는 논리적 복제 슬롯을 식별합니다. 그런 다음 원본 데이터베이스에서 `pg_replication_slots` 뷰를 쿼리하여 이 슬롯에 활성 연결이 없는지 확인합니다. 활성 연결이 있다면 계속하기 전에 연결을 확인하고 닫습니다.

다음 단계에서는 논리적 복제 슬롯이

`abc1d2efghijk_34567890_z0yx98w7_6v54_32ut_1srq_1a2b34c5d67ef`라고 가정합니다.

2. 다음 추가 연결 속성 설정을 포함하는 새 소스 엔드포인트를 만듭니다.

```
slotName=abc1d2efghijk_34567890_z0yx98w7_6v54_32ut_1srq_1a2b34c5d67ef;
```

3. 콘솔 AWS CLI 또는 AWS DMS API를 사용하여 새 CDC 전용 작업을 생성합니다. 예를 들어 CLI를 사용하여 다음 `create-replication-task` 명령을 실행할 수 있습니다.

```
aws dms create-replication-task --replication-task-identifier postgresql-slot-name-test
--source-endpoint-arn arn:aws:dms:us-west-2:012345678901:endpoint:ABCD1EFGHIJK2LMNOPQRST3UV4
--target-endpoint-arn arn:aws:dms:us-west-2:012345678901:endpoint:ZYX9WVUTSRQ0NM8LKJIHGF7ED6
--replication-instance-arn arn:aws:dms:us-west-2:012345678901:rep:AAAAAAAAAAAA5BB4CCC3DDDD2EE
--migration-type cdc --table-mappings "file://mappings.json" --cdc-start-position
"4AF/B00000D0"
--replication-task-settings "file://task-pg.json"
```

위의 명령에서는 다음 옵션이 설정됩니다.

- `source-endpoint-arn` 옵션은 2단계에서 만든 새 값으로 설정됩니다.
- `replication-instance-arn` 옵션은 1단계의 상위 작업과 동일한 값으로 설정됩니다.
- `table-mappings` 및 `replication-task-settings` 옵션은 1단계의 상위 작업과 동일한 값으로 설정됩니다.
- `cdc-start-position` 옵션은 시작 위치 값으로 설정됩니다. 이 시작 위치를 찾으려면 원본 데이터베이스의 `pg_replication_slots` 뷰를 쿼리하거나 1단계에서 상위 작업에 대한 콘솔 세부 정보를 확인합니다. 자세한 내용은 [CDC 기본 시작점 결정](#) 단원을 참조하십시오.

AWS DMS 콘솔을 사용하여 새 CDC 전용 작업을 생성할 때 사용자 지정 CDC 시작 모드를 활성화하려면 다음을 수행합니다.

- 작업 설정 섹션의 소스 트랜잭션의 CDC 시작 모드에서 사용자 지정 CDC 시작 모드 활성화를 선택합니다.
- 소스 트랜잭션의 사용자 지정 CDC 시작 지점에서 로그 시퀀스 번호 지정을 선택합니다. 시스템 변경 번호를 지정하거나 복구 체크포인트 지정을 선택하고 복구 체크포인트를 제공합니다.

이 CDC 태스크가 실행되면 지정된 논리적 복제 슬롯이 없는 경우에서 오류가 AWS DMS 발생합
니다. 또한 작업이 cdc-start-position에 올바른 설정으로 생성되지 않은 경우에도 오류가 발
생합니다.

pglogical 플러그인과 함께 네이티브 CDC 시작 지점을 사용하고 새 복제 슬롯을 사용하려는 경우,
CDC 작업을 생성하기 전에 다음 설정 단계를 완료하세요.

이전에 다른 DMS 작업의 일부로 생성하지 않은 새 복제 슬롯을 사용하려면

1. 다음과 같이 복제 슬롯을 생성합니다.

```
SELECT * FROM pg_create_logical_replication_slot('replication_slot_name',  
'pglogical');
```

2. 데이터베이스에서 복제 슬롯이 생성된 후 해당 슬롯의 restart_lsn 및 confirmed_flush_lsn 값을 가
져와 적어 둡니다.

```
select * from pg_replication_slots where slot_name like 'replication_slot_name';
```

복제 슬롯 이후에 생성된 CDC 작업의 네이티브 CDC 시작 위치는 confirmed_flush_lsn 값보다 이
전일 수 없습니다.

restart_lsn 및 confirmed_flush_lsn 값에 대한 자세한 내용은 [pg_replication_slots](#)를 참조하세요.

3. pglogical 노드를 생성합니다.

```
SELECT pglogical.create_node(node_name := 'node_name', dsn := 'your_dsn_name');
```

4. pglogical.create_replication_set 함수를 사용하여 두 개의 복제 세트를 생성합니다. 첫
번째 복제 세트는 프라이머리 키가 있는 테이블의 업데이트 및 삭제를 추적합니다. 두 번째 복제
세트는 삽입만 추적하며, 첫 번째 복제 세트와 같은 이름에 접두사 'i'가 추가됩니다.

```
SELECT pglogical.create_replication_set('replication_slot_name', false, true, true,  
false);  
SELECT pglogical.create_replication_set('ireplication_slot_name', true, false,  
false, true);
```

5. 복제본 세트에 테이블을 추가합니다.

```
SELECT pglogical.replication_set_add_table('replication_slot_name',
'schemaname.tablename', true);
SELECT pglogical.replication_set_add_table('ireplication_slot_name',
'schemaname.tablename', true);
```

6. 소스 엔드포인트를 생성할 때 추가 연결 속성(ECA)을 다음과 같이 설정합니다.

```
PluginName=PGLLOGICAL;slotName=slot_name;
```

이제 새 복제 슬롯을 사용하여 PostgreSQL 네이티브 시작 지점이 있는 CDC 전용 작업을 생성할 수 있습니다. pglogical 플러그인에 대한 자세한 내용은 [pglogical 3.7 설명서](#)를 참조하세요.

를 사용하여 PostgreSQL에서 PostgreSQL로 마이그레이션 AWS DMS

PostgreSQL 이외의 데이터베이스 엔진에서 PostgreSQL 데이터베이스로 마이그레이션하는 경우 AWS DMS 는 거의 항상 사용할 수 있는 최상의 마이그레이션 도구입니다. 하지만 PostgreSQL 데이터베이스에서 PostgreSQL 데이터베이스로 마이그레이션하는 경우, PostgreSQL 도구가 더 효과적입니다.

PostgreSQL 네이티브 도구를 사용하여 데이터 마이그레이션

다음과 같은 경우에는 pg_dump와 같은 PostgreSQL 데이터베이스 마이그레이션 도구를 사용하는 것이 좋습니다.

- 같은 유형의 마이그레이션을 수행합니다. 이 경우 PostgreSQL 데이터베이스에서 대상 PostgreSQL 데이터베이스로 마이그레이션합니다.
- 전체 데이터베이스를 마이그레이션합니다.
- 기본 도구를 사용하여 최소한의 가동 중지 시간으로 데이터를 마이그레이션할 수 있습니다.

pg_dump 유틸리티는 COPY 명령을 사용하여 PostgreSQL 데이터베이스의 스키마와 데이터 덤프를 만듭니다. pg-dump에 의해 생성되는 덤프 스크립트는 같은 이름을 가진 데이터베이스로 데이터를 로드하고 테이블, 인덱스, 외래 키를 다시 만듭니다. 데이터를 다른 이름의 데이터베이스로 복구하려면 pg_restore 명령과 -d 파라미터를 사용하세요.

EC2에서 실행되는 PostgreSQL 소스 데이터베이스에서 Amazon RDS for PostgreSQL 대상으로 데이터를 마이그레이션하는 경우, pglogical 플러그인을 사용할 수 있습니다.

PostgreSQL 데이터베이스를 Amazon RDS for PostgreSQL 또는 Amazon Aurora PostgreSQL 호환 버전으로 가져오기에 대한 자세한 내용은 <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/PostgreSQL.Procedural.Importing.html> 섹션을 참조하세요.

DMS를 사용하여 PostgreSQL에서 PostgreSQL로 데이터 마이그레이션

AWS DMS 는 예를 들어 온프레미스에 있는 소스 PostgreSQL 데이터베이스에서 대상 Amazon RDS for PostgreSQL 또는 Aurora PostgreSQL 인스턴스로 데이터를 마이그레이션할 수 있습니다. 코어 또는 베이직 PostgreSQL 데이터 유형은 대부분은 성공적으로 마이그레이션됩니다.

 Note

파티션된 테이블을 PostgreSQL 소스에서 PostgreSQL 대상으로 복제할 때는 DMS 작업에서 선택 기준의 일부로 상위 테이블을 언급하지 않아도 됩니다. 상위 테이블을 언급하면 대상의 하위 테이블에서 데이터가 중복되어 PK 위반이 발생할 수 있습니다. 테이블 매핑 선택 기준에서 하위 테이블만 선택하면 상위 테이블이 자동으로 채워집니다.

소스 데이터베이스에서 지원되지만 대상에서 지원되지 않는 데이터 형식은 성공적으로 마이그레이션되지 않을 수 있습니다.는 데이터 형식을 알 수 없는 경우 일부 데이터 형식을 문자열로 AWS DMS 스트리밍합니다. XML 또는 JSON과 같은 어떤 데이터 유형은 작업 파일을 성공적으로 마이그레이션할 수 있지만 대형 문서의 경우 실패할 수 있습니다.

데이터 형식 마이그레이션을 수행할 때는 다음 사항에 유의하세요.

- PostgreSQL NUMERIC(p,s) 데이터 형식이 정밀도와 스케일을 지정하지 않는 경우도 있습니다. DMS 버전 3.4.2 이하 버전에서 DMS는 기본적으로 정밀도 28과 스케일 6인 NUMERIC(28,6)을 사용합니다. 예를 들어 소스의 0.611111104488373 값은 PostgreSQL 대상에서 0.611111로 변환됩니다.
- ARRAY 데이터 형식이 있는 테이블에는 프라이머리 키가 있어야 합니다. 프라이머리 키가 없는 ARRAY 데이터 형식이 있는 테이블은 전체 로드 중에 일시 중단됩니다.

다음 표는 소스 PostgreSQL 데이터 유형과 이 유형들이 성공적으로 마이그레이션되었는지 나타냅니다.

데이터 유형	마이그레이션 성공	부분적으로 마이그레이션	마이그레이션 하지 않습니 다	설명
INTEGER	X			
SMALLINT	X			
BIGINT	X			
NUMERIC/DECIMAL(p,s)		X		여기서, 0<p<39 및 0<s
NUMERIC/DECIMAL		X		여기서, p>38 또는 p=s=0
REAL	X			
DOUBLE	X			
SMALLSERIAL	X			
SERIAL	X			
BIGSERIAL	X			
MONEY	X			
CHAR		X		지정한 정밀 도 없이
CHAR(n)	X			
VARCHAR		X		지정한 정밀 도 없이
VARCHAR(n)	X			
TEXT	X			

데이터 유형	마이그레이션 성공	부분적으로 마이그레이션	마이그레이션 하지 않습니 다	설명
BYTEA	X			
TIMESTAMP	X			양과 음의 무한대 값은 각각 '9999-12-31 23:59:59'와 '4713-01-01 00:00:00 BC'로 잘립니다.
TIMESTAMP(시간대 사용)		X		
DATE	X			
TIME	X			
TIME WITH TIME ZONE	X			
INTERVAL		X		
BOOLEAN	X			
ENUM			X	
CIDR	X			
INET			X	
MACADDR			X	
TSVECTOR			X	
TSQUERY			X	
XML		X		

데이터 유형	마이그레이션 성공	부분적으로 마이그레이션	마이그레이션 하지 않습니 다	설명
POINT	X			PostGIS 공간 데이터 형식
LINE			X	
LSEG			X	
BOX			X	
PATH			X	
POLYGON	X			PostGIS 공간 데이터 형식
CIRCLE			X	
JSON		X		
ARRAY	X			프라이머 리 키 필요
COMPOSITE			X	
RANGE			X	
LINESTRING	X			PostGIS 공간 데이터 형식
MULTIPOINT	X			PostGIS 공간 데이터 형식
MULTILINESTRING	X			PostGIS 공간 데이터 형식
MULTIPOLYGON	X			PostGIS 공간 데이터 형식

데이터 유형	마이그레이션 성공	부분적으로 마이그레이션	마이그레이션 하지 않습니 다	설명
GEOMETRYCOLLECTION	X			PostGIS 공간 데이터 형식

PostGIS 공간 데이터 형식 마이그레이션

공간 데이터는 공간에서 객체 또는 위치의 지오메트리 정보를 식별합니다. PostgreSQL 객체 관계형 데이터베이스는 PostGIS 공간 데이터 형식을 지원합니다.

PostgreSQL 공간 데이터 객체를 마이그레이션하기 전에 PostGIS 플러그인이 전역 수준에서 활성화되어 있는지 확인하세요. 이렇게 하면 PostgreSQL 대상 DB 인스턴스에 대한 정확한 소스 공간 데이터 열을 AWS DMS 생성합니다.

PostgreSQL에서 PostgreSQL로의 동종 마이그레이션의 경우는 PostGIS 기하학적 및 지리적(지리적 좌표) 데이터 객체 유형과 다음과 같은 하위 유형의 마이그레이션을 AWS DMS 지원합니다.

- POINT
- LINESTRING
- POLYGON
- MULTIPOINT
- MULTILINESTRING
- MULTIPOLYGON
- GEOMETRYCOLLECTION

를 사용하여 Babelfish for Amazon Aurora PostgreSQL에서 마이그레이션 AWS DMS

를 사용하여 Babelfish for Aurora PostgreSQL 소스 테이블을 지원되는 모든 대상 엔드포인트로 마이그레이션할 수 있습니다 AWS DMS.

DMS 콘솔, API 또는 CLI 명령을 사용하여 AWS DMS 소스 엔드포인트를 생성할 때 소스를 Amazon Aurora PostgreSQL로 설정하고 데이터베이스 이름을 로 설정합니다 **babelfish_db**. 엔드포인트 설정 섹션에서 DatabaseMode가 Babelfish로 설정되어 있고 BabelfishDatabaseName이 소스 Babelfish T-SQL 데이터베이스의 이름으로 설정되어 있는지 확인합니다. Babelfish TCP 포트 **1433**을 사용하는 대신 Aurora PostgreSQL TCP 포트 **5432**를 사용합니다.

데이터를 마이그레이션하기 전에 테이블을 생성하여 DMS가 올바른 데이터 유형과 테이블 메타데이터를 사용하도록 해야 합니다. 마이그레이션을 실행하기 전에 대상에 테이블을 생성하지 않으면 DMS는 잘못된 데이터 유형 및 권한으로 테이블을 생성할 수 있습니다.

마이그레이션 작업에 변환 규칙 추가

Babelfish 소스에 대한 마이그레이션 태스크를 생성할 때는 DMS가 사전 생성된 대상 테이블을 사용하도록 하는 변환 규칙을 포함해야 합니다.

Babelfish for PostgreSQL 클러스터를 정의할 때 다중 데이터베이스 마이그레이션 모드를 설정한 경우 스키마 이름을 T-SQL 스키마로 바꾸는 변환 규칙을 추가합니다. 예를 들어 T-SQL 스키마 이름이 `dbo`이고 Babelfish for PostgreSQL 스키마 이름이 `mydb_dbo`인 경우, 변환 규칙을 사용하여 스키마 이름을 `dbo`로 바꾸세요. PostgreSQL 스키마 이름을 찾으려면 Amazon Aurora 사용 설명서의 [Babelfish 아키텍처](#)를 참조하세요.

단일 데이터베이스 모드를 사용하는 경우, 데이터베이스 스키마 이름을 바꾸는 변환 규칙을 사용할 필요가 없습니다. PostgreSQL 스키마 이름에는 T-SQL 데이터베이스의 스키마 이름에 대한 일대일 매핑이 있습니다.

다음 변환 규칙 예제에서는 스키마 이름을 `mydb_dbo`에서 `dbo`로 바꾸는 방법을 보여줍니다.

```
{
  "rules": [
    {
      "rule-type": "transformation",
      "rule-id": "566251737",
      "rule-name": "566251737",
      "rule-target": "schema",
      "object-locator": {
        "schema-name": "mydb_dbo"
      },
      "rule-action": "rename",
      "value": "dbo",
      "old-value": null
    },
    {
      "rule-type": "selection",
      "rule-id": "566111704",
      "rule-name": "566111704",
      "object-locator": {
        "schema-name": "mydb_dbo",
        "table-name": "%"
      }
    }
  ]
}
```

```

    },
    "rule-action": "include",
    "filters": []
  }
]
}

```

PostgreSQL 소스 엔드포인트를 Babelfish 테이블과 함께 사용할 때 적용되는 제한 사항

PostgreSQL 소스 엔드포인트를 Babelfish 테이블과 함께 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- DMS는 Babelfish 버전 16.2/15.6 이상 및 DMS 버전 3.5.3 이상에서만 마이그레이션을 지원합니다.
- DMS는 대상 엔드포인트에 Babelfish 테이블 정의 변경 사항을 복제하지 않습니다. 이 제한의 해결 방법은 먼저 대상에 테이블 정의 변경 사항을 적용한 다음 Babelfish 소스에서 테이블 정의를 변경하는 것입니다.
- BYTEA 데이터 유형으로 Babelfish 테이블을 생성하면 DMS는 SQL Server로 대상을 마이그레이션할 때 해당 테이블을 varbinary(max) 데이터 유형으로 변환합니다.
- DMS는 바이너리 데이터 유형에 대한 전체 LOB 모드를 지원하지 않습니다. 이진 데이터 유형에는 대신 제한된 LOB 모드를 사용합니다.
- DMS는 Babelfish에 대한 데이터 검증을 소스로 지원하지 않습니다.
- 대상 테이블 준비 모드 태스크 설정의 경우, 아무것도 안 함 또는 잘라내기 모드만 사용하세요. 대상에서 테이블 삭제 모드를 사용하지 마세요. 대상에서 삭제 테이블을 사용하는 경우 DMS는 잘못된 데이터 유형을 사용하여 테이블을 생성할 수 있습니다.
- 지속적인 복제(CDC 또는 전체 로드 및 CDC)를 사용하는 경우 PluginName 추가 연결 속성(ECA)을 TEST_DECODING으로 설정합니다.
- DMS는 Babelfish를 소스로 분할한 테이블의 복제(CDC 또는 전체 로드 및 CDC)를 지원하지 않습니다.

PostgreSQL 소스 데이터베이스에서 AWS DMS 아티팩트 제거

DDL 이벤트를 캡처하기 위해 마이그레이션 작업이 시작될 때 PostgreSQL 데이터베이스에서 다양한 아티팩트를 AWS DMS 생성합니다. 작업이 완료되면 이 아티팩트를 제거해야 할 수도 있습니다.

아티팩트를 제거하려면 다음 문을 (표시되는 순서대로) 발행합니다. 여기서

{AmazonRDSMigration}은 아티팩트가 생성되는 스키마입니다. 스키마 삭제를 수행할 때는 매우 주의해야 합니다. 절대로 연산 스키마(특히 퍼블릭 스키마)를 삭제해서는 안 됩니다.

```
drop event trigger awsdms_intercept_ddl;
```

이벤트 트리거는 특정 스키마에 속하지 않습니다.

```
drop function {AmazonRDSMigration}.awsdms_intercept_ddl()
drop table {AmazonRDSMigration}.awsdms_ddl_audit
drop schema {AmazonRDSMigration}
```

PostgreSQL 데이터베이스를 DMS 소스로 사용 시 추가 구성 설정

다음 두 가지 방법으로 PostgreSQL 데이터베이스에서 데이터를 마이그레이션할 때 구성 설정을 추가할 수 있습니다.

- 연결 속성을 추가하여 DDL 이벤트를 캡처하고 연산 DDL 데이터베이스 아티팩트가 생성된 스키마를 지정할 수 있습니다. 자세한 내용은 [DMS 소스로 PostgreSQL 사용 시 엔드포인트 설정 및 추가 연결 속성\(ECA\)](#) 단원을 참조하십시오.
- 연결 문자열 파라미터를 재정의할 수 있습니다. 다음 중 하나를 수행하려면 이 옵션을 선택합니다.
 - 내부 AWS DMS 파라미터를 지정합니다. 이러한 파라미터는 거의 필요하지 않으므로 사용자 인터페이스에 표시되지 않습니다.
 - 특정 데이터베이스 클라이언트에 대한 패스스루(패스스루) 값을 지정합니다. 데이터베이스 클라이언트에 전달되는 연결 스템에 패스스루 파라미터를 AWS DMS 포함합니다.
- PostgreSQL 버전 9.4 이상에서 테이블 수준 파라미터 REPLICATION IDENTITY를 사용하면 미리 쓰기 로그(WAL)에 기록되는 정보를 제어할 수 있습니다. 특히 업데이트되거나 삭제된 행을 식별하는 WAL의 경우 더욱 그렇습니다. REPLICATION IDENTITY FULL은 행에 있는 모든 열의 이전 값을 기록합니다. FULL은 필요하지 않을 수도 있는 WAL을 추가로 생성하므로 각 테이블에 REPLICATION IDENTITY FULL을 신중하게 사용하세요. 자세한 내용은 [ALTER TABLE-REPLICATION IDENTITY](#)를 참조하세요.

읽기 전용 복제본을 PostgreSQL의 소스로 사용

PostgreSQL 읽기 전용 복제본을의 CDC 소스로 사용하여 기본 데이터베이스 로드를 AWS DMS 줄입니다. 이 기능은 PostgreSQL 16.x에서 사용할 수 있으며 AWS DMS 버전 3.6.1 이상이 필요합니다. CDC 처리에 읽기 전용 복제본을 사용하면 기본 데이터베이스에 대한 운영 영향이 줄어듭니다.

Note

Amazon RDS PostgreSQL 버전 16.x에는 3AZ(TAZ) 구성의 읽기 전용 복제본 논리적 복제에 대한 제한이 있습니다. TAZ 배포에서 전체 읽기 전용 복제본 논리적 복제를 지원하려면 PostgreSQL 버전 17.x 이상을 사용해야 합니다.

사전 조건

읽기 전용 복제본을 CDC 소스로 사용하기 전에 기본 데이터베이스 인스턴스와 읽기 전용 복제본 모두에서 논리적 복제를 활성화하여 읽기 전용 복제본에 논리적 디코딩을 생성 AWS DMS해야 합니다. 다음 조치 중 하나를 취하세요.

- 다른 필수 데이터베이스 파라미터와 함께 기본 데이터베이스 인스턴스 및 읽기 전용 복제본 모두에서 논리적 복제를 활성화합니다. 자세한 내용은 [DMS 소스로 AWS관리형 PostgreSQL 데이터베이스 작업을 참조하세요](#).
- CDC 전용 작업의 경우 기본(라이터) 인스턴스에 복제 슬롯을 생성합니다. 자세한 내용은 [네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정](#)을 참조하세요. 읽기 전용 복제본은 복제 슬롯 생성을 지원하지 않으므로 이 작업이 필요합니다.
- PostgreSQL 버전 16의 경우 읽기 전용 복제본에서 슬롯을 수동으로 생성해야 합니다.
- PostgreSQL 버전 17 이상의 경우 복제 슬롯을 기본에 생성해야 하며 읽기 전용 복제본과 자동으로 동기화됩니다.
- 전체 로드 + CDC 또는 CDC 전용 작업을 사용하는 경우 AWS DMS 는 읽기 전용 복제본이 아닌 기본 인스턴스의 논리적 복제 슬롯을 자동으로 관리할 수 있습니다. PostgreSQL 버전 16 읽기 전용 복제본의 경우 작업을 다시 시작하기 전에(다시 시작하지 않고) 복제 슬롯을 수동으로 삭제하고 다시 생성해야 합니다. 이 단계를 건너뛰면 작업 실패 또는 잘못된 CDC 시작 위치가 발생할 수 있습니다. PostgreSQL 버전 17 이상에서는 기본 인스턴스의 논리적 슬롯 동기화가이 프로세스를 자동화합니다.

사전 조건을 완료한 후 엔드포인트 설정에서 SlotName 읽기 전용 복제본 AWS DMS 소스를 복제하여 소스 엔드포인트를 설정하고 네이티브 CDC 시작점을 사용하여 AWS DMS 작업을 구성할 수 있습니다. 자세한 내용은 [PostgreSQL을 DMS 소스로 사용할 때 엔드포인트 설정 및 추가 연결 속성\(ECAs\)](#)과 [기본 CDC 시작점을 사용하여 PostgreSQL 소스의 CDC 로드 설정](#)을 참조하세요.

MapBooleanAsBoolean PostgreSQL 엔드포인트 설정 사용

PostgreSQL 엔드포인트 설정을 사용하여 PostgreSQL 소스의 불을 Amazon Redshift 대상에 매핑할 수 있습니다. 기본적으로 부울 형식은 varchar(5)로 마이그레이션됩니다. 다음 예제에 나온 것처럼 PostgreSQL이 부울 형식을 불로 마이그레이션하도록 MapBooleanAsBoolean을 지정할 수 있습니다.

```
--postgre-sql-settings '{"MapBooleanAsBoolean": true}'
```

단, 이 설정이 적용되려면 소스 엔드포인트와 대상 엔드포인트 모두에서 이 설정을 지정해야 합니다.

MySQL에는 부울 형식이 없으므로 부울 데이터를 MySQL로 마이그레이션할 때는 이 설정 대신 변환 규칙을 사용하세요.

DMS 소스로 PostgreSQL 사용 시 엔드포인트 설정 및 추가 연결 속성(ECA)

엔드포인트 설정 및 추가 연결 속성(ECA)을 사용하여 PostgreSQL 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --postgre-sql-settings '{"EndpointSetting": "value", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 소스 엔드포인트를 생성할 때 엔드포인트 설정을 지정합니다.

다음 표에는 PostgreSQL을 소스로 사용할 수 있는 엔드포인트 설정 및 ECA가 나와 있습니다.

속성 이름	설명
CaptureDdl	DDL 이벤트를 캡처하기 위해서는 작업이 시작될 때 PostgreSQL 데이터베이스에서 다양한 아티팩트를 AWS DMS 생성합니다. PostgreSQL 소스 데이터베이스에서 AWS DMS 아티팩트 제거의 설명에 따라 이 아티팩트를 제거할 수 있습니다. 이 값을 false로 설정하면 소스 데이터베이스에 테이블이나 트리거를 생성할 필요가 없습니다. 스트리밍된 DDL 이벤트가 캡처됩니다. 기본값: true 유효값: true/false

속성 이름	설명
	예시: <code>--postgre-sql-settings '{"CaptureDdls": true}'</code>
ConsumeMonotonicEvents	로그 시퀀스 번호(LSN)가 중복된 모놀리식 트랜잭션이 복제되는 방식을 제어하는 데 사용됩니다. 이 파라미터가 <code>false</code>인 경우, LSN이 중복된 이벤트가 대상에서 소비되고 복제됩니다. 이 파라미터가 <code>true</code>인 경우, 첫 번째 이벤트만 복제되고 LSN이 중복된 이벤트는 대상에서 소비되거나 복제되지 않습니다. 기본값: <code>false</code> 유효값: <code>false/true</code> 예시: <code>--postgre-sql-settings '{"ConsumeMonotonicEvents": true}'</code>
DdlArtifactsSchema	연산 DDL 데이터베이스 아티팩트가 생성되는 스키마를 설정합니다. 기본값: 퍼블릭 유효값: 문자열 예시: <code>--postgre-sql-settings '{"DdlArtifactsSchema": "xyzddlschema"}'</code>

속성 이름	설명
DisableUnicodeSourceFilter	소스 엔드포인트 열 값의 선택 규칙 필터에 전달된 값에 대해 PostgreSQL을 사용하여 유니코드 소스 필터를 비활성화합니다. 기본적으로 DMS는 유니코드 문자열을 사용하여 소스 필터 비교를 수행하므로 조회 시 텍스트 열의 인덱스를 무시하고 마이그레이션 속도를 늦출 수 있습니다. 유니코드 지원은 인덱싱된 소스 데이터베이스의 텍스트 열에 선택 규칙 필터를 사용하는 경우에만 비활성화해야 합니다. 기본값: false 유효값: true/false 예시: --postgre-sql-settings '{"DisableUnicodeSourceFilter": " true"}'
ExecuteTimeout	PostgreSQL 인스턴스의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값은 60초입니다. 예시: --postgre-sql-settings '{"ExecuteTimeout": 100}'
FailTasksOnLobTruncation	이 값을 true로 설정하면 LOB 열의 실제 크기가 지정된 LobMaxSize 보다 클 경우 작업이 실패합니다. 작업이 제한된 LOB 모드로 설정되어 있고 이 옵션이 true로 설정된 경우, LOB 데이터가 잘리지 않는 대신에 작업이 실패합니다. 기본값: false 유효 값: 부울 예시: --postgre-sql-settings '{"FailTasksOnLobTruncation": true}'

속성 이름	설명
fetchCacheSize	이 추가 연결 속성(ECA)은 전체 로드 작업 중에 커서가 가져올 행 수를 설정합니다. 복제 인스턴스에서 사용할 수 있는 리소스에 따라 값을 높이거나 낮출 수 있습니다. 기본값: 10000 유효한 값: 숫자 ECA 예제: <code>fetchCacheSize=10000;</code>
HeartbeatEnable	WAL 하트비트 기능은 더미 트랜잭션을 모방하므로 유효 논리적 복제 슬롯이 이전 WAL 로그를 유지하지 않아 소스에 스토리지 전체 상황이 발생합니다. 이 하트비트는 <code>restart_lsn</code> 이동 재시작을 유지하고 스토리지 전체 시나리오를 방지합니다. 기본값: false 유효값: true/false 예시: <code>--postgre-sql-settings '{"HeartbeatEnable": true}'</code>
HeartbeatFrequency	WAL 하트비트 빈도(분)를 설정합니다. 기본값: 5 유효한 값: 숫자 예시: <code>--postgre-sql-settings '{"HeartbeatFrequency": 1}'</code>

속성 이름	설명
HeartbeatSchema	하트비트 아티팩트가 생성되는 스키마를 설정합니다. 기본값: public 유효값: 문자열 예시: <code>--postgre-sql-settings '{"HeartbeatSchema": "xyzheartbeatSchema"}'</code>
MapJsonbAsClob	기본적으로는 JSONB을 NCLOB으로 AWS DMS 매핑합니다. PostgreSQL이 JSONB 형식을 CLOB로 마이그레이션하도록 MapJsonbAsClob 을 지정할 수 있습니다. 예시: <code>--postgre-sql-settings='{"MapJsonbAsClob": "true"}'</code>
MapLongVarcharAs	기본적으로는 VARCHAR를 WSTRING에 AWS DMS 매핑합니다. PostgreSQL이 VARCHAR(N) 형식(여기서 N은 16387보다 큼)을 다음 형식으로 마이그레이션하도록 MapLongVarcharAs 를 지정할 수 있습니다. • WSTRING • CLOB • NCLOB 예시: <code>--postgre-sql-settings='{"MapLongVarcharAs": "CLOB"}'</code>

속성 이름	설명
MapUnboundedNumericAsString	이 파라미터는 숫자 값의 정밀도를 유지하면서 성공적으로 마이그레이션할 수 있도록 제한이 없는 숫자 데이터 형식의 열을 문자열로 취급합니다. 이 파라미터는 PostgreSQL 소스에서 PostgreSQL 대상 또는 PostgreSQL과 호환되는 데이터베이스로 복제하는 경우에만 사용하세요. 기본값: false 유효값: false/true 예시: <code>--postgre-sql-settings '{"MapUnboundedNumericAsString": true}'</code> 이 파라미터를 사용하면 숫자에서 문자열로, 다시 숫자로 변환되므로 복제 성능이 일부 저하될 수 있습니다. 이 파라미터는 DMS 버전 3.4.4 이상에서 사용할 수 있습니다.  Note 단지 MapUnboundedNumericAsString 을 PostgreSQL 소스 및 대상 엔드포인트에서 함께 사용해야 합니다. 소스 PostgreSQL 엔드포인트에서 MapUnboundedNumericAsString 을 사용하면 CDC 중에 정밀도가 28로 제한됩니다. 대상 엔드포인트에서 MapUnboundedNumericAsString 을 사용하면 정밀도 28, 스케일 6으로 데이터를 마이그레이션할 수 있습니다. PostgreSQL이 아닌 대상에 MapUnboundedNumericAsString 을 사용하지 마세요.

속성 이름	설명
PluginName	복제 슬롯을 생성하는 데 사용할 플러그인을 지정합니다. 유효값: pglogical , test_decoding 예시: --postgre-sql-settings '{"Plugin Name": "test_decoding"}'
SlotName	PostgreSQL 원본 인스턴스의 CDC 로드와 대해 이전에 만든 논리적 복제 슬롯의 이름을 설정합니다. AWS DMS API CdcStartPosition 요청 파라미터와 함께 사용하는 경우이 속성은 네이티브 CDC 시작점을 사용할 수도 있습니다. DMS는 CDC 로드 작업을 시작하기 전에 지정된 논리적 복제 슬롯이 존재하는지 확인합니다. 또한 작업이 CdcStartPosition에 대한 유효한 설정으로 생성되었는지 확인합니다. 지정된 슬롯이 존재하지 않거나 유효한 CdcStartPosition 설정이 작업에 없으면 DMS에서 오류가 발생합니다. CdcStartPosition 요청 파라미터 설정에 대한 자세한 내용은 CDC 기본 시작점 결정 섹션을 참조하세요. CdcStartPosition 사용에 대한 자세한 내용은 AWS Database Migration Service API 참조의 CreateReplicationTask, StartReplicationTask 및 ModifyReplicationTask API 작업 설명서를 참조하세요. 유효값: 문자열 예시: --postgre-sql-settings '{"SlotName": "abc1d2efghijk_34567890_z0yx98w7_6v54_32ut_1srq_1a2b34c5d67ef"}'
unboundedVarcharMaxSize	이 추가 연결 속성(ECA)은 최대 길이 지정자 없이 VarChar 유형으로 정의된 데이터 열의 최대 크기를 정의합니다. 기본값은 8000(바이트)입니다. 최대값은 10,485,760바이트입니다.

PostgreSQL 데이터베이스를 DMS 소스로 사용 시 제한 사항

PostgreSQL을 AWS DMS에서 원본으로 사용하는 경우 다음 제한 사항이 적용됩니다.

- AWS DMS 는 Amazon RDS for PostgreSQL 10.4 또는 Amazon Aurora PostgreSQL 10.4를 소스 또는 대상으로 사용할 수 없습니다.
- 캡처된 테이블에는 프라이머리 키가 있어야 합니다. 테이블에 기본 키가 없는 경우는 해당 테이블에 대한 DELETE 및 UPDATE 레코드 작업을 AWS DMS 무시합니다. 해결 방법은 [논리적 복제를 사용한 변경 데이터 캡처\(CDC\) 활성화](#)를 참조하세요.

참고: 프라이머리 키/고유 인덱스 없이 마이그레이션하는 것은 권장하지 않습니다. 마이그레이션할 경우, “NO” Batch 적용 기능, 전체 LOB 기능, 데이터 검증 및 Redshift 대상으로 효율적으로 복제할 수 없는 등의 추가 제한이 적용됩니다.

- AWS DMS 는 기본 키 세그먼트를 업데이트하려는 시도를 무시합니다. 이 경우, 대상에서 아무 행도 업데이트하지 않는 것으로 업데이트를 식별합니다. 그러나, PostgreSQL의 프라이머리 키 업데이트 결과는 예측할 수 없기 때문에 예외 테이블에 레코드가 기록되지 않습니다.
- AWS DMS 는 타임스탬프 실행에서 프로세스 변경 시작 옵션을 지원하지 않습니다.
- AWS DMS 는 파티션 또는 하위 파티션 작업(ADD, 또는 DROPTRUNCATE)으로 인한 변경 사항을 복제하지 않습니다.
- 이름이 같지만 각 이름의 대소문자가 다른 여러 테이블(예: table1, TABLE1, Table1)을 복제하면 예측할 수 없는 동작이 발생할 수 있습니다. 이 문제로 인해 AWS DMS 는 이러한 유형의 복제를 지원하지 않습니다.
- 대부분의 경우는 테이블에 대한 CREATE, ALTER 및 DROP DDL 명령문의 변경 처리를 AWS DMS 지원합니다.는 테이블이 내부 함수 또는 프로시 AWS DMS 저 본문 블록 또는 다른 중첩된 구문에 유지되는 경우이 변경 처리를 지원하지 않습니다.

예를 들어 다음 변경은 캡처되지 않습니다.

```
CREATE OR REPLACE FUNCTION attu.create_distributors1() RETURNS void
LANGUAGE plpgsql
AS $$
BEGIN
create table attu.distributors1(did serial PRIMARY KEY,name
varchar(40) NOT NULL);
END;
$$;
```

- 현재 PostgreSQL 소스의 boolean 데이터 형식은 값이 일치하지 않는 bit 데이터 형식으로 SQL Server 대상에 마이그레이션됩니다. 해결 방법으로 열에 대한 VARCHAR(1) 데이터 유형을 사용하여 테이블을 미리 생성(또는 테이블 AWS DMS 생성)합니다. 그런 다음, 다운스트림 처리에서 'F'를 False로 처리하고 'T'를 True로 처리합니다.
- AWS DMS 는 TRUNCATE 작업의 변경 처리를 지원하지 않습니다.
- OID LOB 데이터 형식은 대상으로 마이그레이션되지 않습니다.
- AWS DMS 는 동종 마이그레이션에 대해서만 PostGIS 데이터 형식을 지원합니다.
- 소스가 온프레미스 또는 Amazon EC2 인스턴스에 있는 PostgreSQL 데이터베이스인 경우, 소스 엔드포인트에 test_decoding 출력 플러그인이 설치되어 있는지 확인합니다. 이 플러그인은 PostgreSQL contrib 패키지에서 찾을 수 있습니다. test-decoding 플러그인에 대한 자세한 내용은 [PostgreSQL 설명서](#)를 참조하세요.
- AWS DMS 는 열 기본값을 설정 및 설정 해제하는 변경 처리를 지원하지 않습니다(ALTER TABLE 문에서 ALTER COLUMN SET DEFAULT 절 사용).
- AWS DMS 는 열 null성을 설정하는 변경 처리를 지원하지 않습니다(ALTER TABLE 문에서 ALTER COLUMN [SET|DROP] NOT NULL 절 사용).
- 논리적 복제가 활성화된 경우, 트랜잭션당 메모리에 보관되는 최대 변경 사항 수는 4MB입니다. 그 후에는 변경 사항이 디스크로 유출됩니다. 따라서 트랜잭션이 완료 또는 중지되고 롤백이 완료될 때까지 ReplicationSlotDiskUsage가 증가하며 restart_lsn이 진행되지 않습니다. 트랜잭션이 길기 때문에 롤백하는 데 시간이 오래 걸릴 수 있습니다. 따라서 논리적 복제가 활성화된 경우, 장기 실행 트랜잭션 또는 많은 하위 트랜잭션은 피하세요. 대신 트랜잭션을 여러 개의 작은 트랜잭션으로 나누세요.

Aurora PostgreSQL 버전 13 이상에서는 DMS 유출 시 디스크로 데이터가 변경될 때 제어할 logical_decoding_work_mem 파라미터를 조정할 수 있습니다. 자세한 내용은 [Aurora PostgreSQL에서 파일 유출](#) 단원을 참조하십시오.

- ARRAY 데이터 형식이 있는 테이블에는 프라이머리 키가 있어야 합니다. 프라이머리 키가 없는 ARRAY 데이터 형식이 있는 테이블은 전체 로드 중에 일시 중단됩니다.
- AWS DMS 는 테이블 파티셔닝 또는 [테이블 상속](#)과 관련된 테이블 메타데이터 마이그레이션을 지원하지 않습니다. 가 분할된 테이블 또는 상속을 사용하는 테이블을 AWS DMS 발견하면 다음 동작이 관찰됩니다.
 - AWS DMS 는 소스 데이터베이스의 파티셔닝 또는 상속과 관련된 상위 테이블과 하위 테이블을 모두 식별하고 보고합니다.
 - 대상 테이블 생성: 대상 데이터베이스에서는 테이블을 표준(파티션되지 않은, 상속되지 않은) 테이블로 AWS DMS 생성하여 선택한 테이블의 구조와 속성을 보존하지만 파티셔닝 또는 상속 로직은 보존하지 않습니다.

- 상속된 테이블의 레코드 구분: 상속을 사용하는 테이블의 경우 상위 테이블을 채울 때는 하위 테이블에 속하는 레코드를 구분하지 AWS DMS 않습니다. 따라서 `SELECT * FROM ONLY parent_table_name`와 같은 구문으로 SQL 쿼리를 활용하지 않습니다.
- 파티션된 테이블을 PostgreSQL 소스에서 PostgreSQL 대상으로 복제하려면, 먼저 대상에 상위 및 하위 테이블을 수동으로 생성해야 합니다. 그런 다음 이 테이블에 복제할 별도의 작업을 정의합니다. 이 경우에 작업 구성을 로딩 전 자르기로 설정합니다.
- PostgreSQL NUMERIC 데이터 형식의 크기는 고정되어 있지 않습니다. NUMERIC 데이터 형식이지만 정밀도 및 배율이 없는 데이터를 전송하는 경우, DMS는 기본적으로 NUMERIC(28,6)(정밀도 28, 배율 6)을 사용합니다. 예를 들어 소스의 0.6111111104488373 값은 PostgreSQL 대상에서 0.611111로 변환됩니다.
- AWS DMS 는 Aurora PostgreSQL Serverless V1을 전체 로드 작업의 소스로만 지원합니다.는 Aurora PostgreSQL Serverless V2를 전체 로드, 전체 로드 및 CDC, CDC 전용 작업의 소스로 AWS DMS 지원합니다.
- AWS DMS 는 병합 함수로 생성된 고유 인덱스가 있는 테이블의 복제를 지원하지 않습니다.
- 소스와 대상의 프라이머리 키 정의가 일치하지 않는 경우 복제 결과를 예측할 수 없습니다.
- 병렬 로드 기능을 사용하는 경우, 파티션 또는 하위 파티션에 따른 테이블 분할은 지원되지 않습니다. 병렬 로드와 관련한 자세한 내용은 [선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용](#) 섹션을 참조하세요.
- AWS DMS 는 연기된 제약 조건을 지원하지 않습니다.
- AWS DMS 버전 3.4.7은 PostgreSQL 14.x를 다음과 같은 제한 사항이 있는 소스로 지원합니다.
 - AWS DMS 는 2단계 커밋의 변경 처리를 지원하지 않습니다.
 - AWS DMS 는 진행 중인 긴 트랜잭션을 스트리밍하기 위한 논리적 복제를 지원하지 않습니다.
- AWS DMS 는 Amazon RDS Proxy for PostgreSQL에 대한 CDC를 소스로 지원하지 않습니다.
- 프라이머리 키 열을 포함하지 않는 [소스 필터](#)를 사용하면 DELETE 작업이 캡처되지 않습니다.
- 소스 데이터베이스가 다른 타사 복제 시스템의 대상이기도 한 경우, CDC 중에는 DDL 변경 내용이 마이그레이션되지 않을 수 있습니다. 이러한 상황으로 인해 `awsdms_intercept_ddl` 이벤트 트리거가 실행되지 않을 수 있기 때문입니다. 이 상황을 해결하려면 소스 데이터베이스에서 해당 트리거를 다음과 같이 수정하세요.

```
alter event trigger awsdms_intercept_ddl enable always;
```

- AWS DMS 는 소스 데이터베이스의 기본 키 정의에 대한 변경 사항 복제를 지원하지 않습니다. 활성 복제 작업 중에 프라이머리 키 구조가 변경되면 영향을 받는 테이블에 대한 후속 변경 사항이 대상에 복제되지 않습니다.

- 스크립트의 일부로 DDL 복제에서 스크립트당 최대 총 DDL 명령 수는 8192개이고 스크립트당 최대 총 줄 수는 8192개입니다.
- AWS DMS 는 구체화된 보기를 지원하지 않습니다.
- 읽기 전용 복제본을 소스로 사용하는 전체 로드 및 CDC 작업의 경우 읽기 전용 복제본에 복제 슬롯을 생성할 AWS DMS 수 없습니다.

PostgreSQL용 소스 데이터 형식

다음 표에는 사용 시 지원되는 PostgreSQL 소스 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식에 대한 기본 매핑이 나와 있습니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

PostgreSQL 데이터 형식	DMS 데이터 형식
INTEGER	INT4
SMALLINT	INT2
BIGINT	INT8
NUMERIC(p,s)	정밀도 범위가 0~38인 경우, NUMERIC을 사용합니다. 정밀도가 39 이상인 경우, STRING을 사용합니다.
DECIMAL(P,S)	정밀도 범위가 0~38인 경우, NUMERIC을 사용합니다. 정밀도가 39 이상인 경우, STRING을 사용합니다.
REAL	REAL4

PostgreSQL 데이터 형식	DMS 데이터 형식
DOUBLE	REAL8
SMALLSERIAL	INT2
SERIAL	INT4
BIGSERIAL	INT8
MONEY	NUMERIC(38,4) MONEY 데이터 형식은 SQL Server에서 FLOAT에 매핑됩니다.
CHAR	WSTRING (1)
CHAR(N)	WSTRING (n)
VARCHAR(N)	WSTRING (n)
TEXT	NCLOB
CITEXT	NCLOB
BYTEA	BLOB
TIMESTAMP	DATETIME
TIMESTAMP(시간대 사용)	DATETIME
DATE	DATE
TIME	TIME
TIME WITH TIME ZONE	TIME
INTERVAL	STRING (128) - 1 YEAR, 2 MONTHS, 3 DAYS, 4 HOURS, 5 MINUTES, 6 SECONDS
BOOLEAN	CHAR (5) 거짓 혹은 참

PostgreSQL 데이터 형식	DMS 데이터 형식
ENUM	STRING (64)
CIDR	STRING (50)
INET	STRING (50)
MACADDR	STRING (18)
BIT (n)	STRING (n)
BIT VARYING (n)	STRING (n)
UUID	STRING
TSVECTOR	CLOB
TSQUERY	CLOB
XML	CLOB
POINT	STRING (255) "(x,y)"
LINE	STRING (255) "(x,y,z)"
LSEG	STRING (255) "((x1,y1),(x2,y2))"
BOX	STRING (255) "((x1,y1),(x2,y2))"
PATH	CLOB "((x1,y1),(xn,yn))"
POLYGON	CLOB "((x1,y1),(xn,yn))"
CIRCLE	STRING (255) "(x,y),r"
JSON	NCLOB
JSONB	NCLOB
ARRAY	NCLOB

PostgreSQL 데이터 형식	DMS 데이터 형식
COMPOSITE	NCLOB
HSTORE	NCLOB
INT4RANGE	STRING (255)
INT8RANGE	STRING (255)
NUMRANGE	STRING (255)
STRRANGE	STRING (255)

PostgreSQL에 LOB 소스 데이터 형식 사용

PostgreSQL 열 크기는 PostgreSQL LOB 데이터 형식의 AWS DMS 데이터 형식으로의 변환에 영향을 미칩니다. 이를 위해 다음 AWS DMS 데이터 형식의 경우, 다음 단계에 따르세요.

- BLOB - 작업 생성 시 LOB 크기를 다음으로 제한을 최대 LOB 크기(KB) 값으로 설정합니다.
- CLOB - 복제는 각 문자를 UTF8 문자로 취급합니다. 따라서 열에서 가장 긴 문자 텍스트의 길이를 찾습니다(여기에서는 `max_num_chars_text`로 표시). 이 길이를 사용하여 LOB 크기를 다음으로 제한 값을 지정합니다. 데이터에 4바이트 문자가 포함된 경우, 2를 곱해 LOB 크기를 다음으로 제한 값(바이트 단위)을 지정합니다. 이 경우, LOB 크기를 다음으로 제한은 `max_num_chars_text x 2`와 같습니다.
- NCLOB - 복제는 각 문자를 2바이트 문자로 취급합니다. 따라서 열에서 가장 긴 문자 텍스트의 길이(`max_num_chars_text`)를 찾고 2를 곱합니다. 이렇게 하여 LOB 크기를 다음으로 제한 값을 지정합니다. 이 경우, LOB 크기를 다음으로 제한은 `max_num_chars_text x 2`와 같습니다. 데이터에 4바이트 문자가 포함된 경우에는 다시 2를 곱합니다. 이 경우, LOB 크기를 다음으로 제한은 `max_num_chars_text x 4`와 같습니다.

MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS

AWS Database Migration Service를 사용하여 MySQL 호환 데이터베이스(MySQL, MariaDB 또는 Amazon Aurora MySQL)에서 데이터를 마이그레이션할 수 있습니다.

가 소스로 AWS DMS 지원하는 MySQL 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

SSL을 사용하여 MySQL 호환 엔드포인트와 복제 인스턴스 간의 연결을 암호화할 수 있습니다. MySQL 호환 엔드포인트에서 SSL 사용에 관한 자세한 내용은 [에서 SSL 사용 AWS Database Migration Service](#) 섹션을 참조하세요.

다음 섹션에서 ‘자체 관리형’이라는 용어는 온프레미스나 Amazon EC2에 설치되어 있는 데이터베이스에 적용됩니다. ‘AWS관리형’이라는 용어는 Amazon RDS, Amazon Aurora 또는 Amazon S3에 있는 모든 데이터베이스에 적용됩니다.

MySQL 호환 데이터베이스 작업에 대한 자세한 내용은 다음 섹션을 AWS DMS참조하세요.

주제

- [를 사용하여 MySQL에서 MySQL로 마이그레이션 AWS DMS](#)
- [MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS](#)
- [자체 관리형 MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS](#)
- [AWS관리형 MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS](#)
- [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#)
- [XA 트랜잭션 지원](#)
- [MySQL을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#)
- [MySQL용 소스 데이터 형식](#)

Note

AWS Database Migration Service (AWS DMS) 매핑 규칙을 구성할 때는 데이터베이스 또는 스키마 이름에 와일드카드(%)를 사용하지 않는 것이 중요합니다. 대신 마이그레이션해야 하는 사용자 생성 데이터베이스만 명시적으로 지정해야 합니다. 와일드카드 문자를 사용하면 대상 인스턴스에 필요하지 않은 시스템 데이터베이스를 포함하여 마이그레이션 프로세스의 모든 데이터베이스가 포함됩니다. MySQL Amazon RDS 마스터 사용자는 대상 시스템 데이터베이스로 데이터를 가져오는 데 필요한 권한이 없으므로 이러한 시스템 데이터베이스를 마이그레이션하지 못합니다.

를 사용하여 MySQL에서 MySQL로 마이그레이션 AWS DMS

MySQL 이외의 데이터베이스 엔진에서 MySQL 데이터베이스로 마이그레이션하는 이기종 마이그레이션의 경우는 거의 항상 사용할 수 있는 최상의 마이그레이션 도구 AWS DMS 입니다. 하지만 MySQL 데이터베이스에서 MySQL 데이터베이스로 마이그레이션하는 동종 마이그레이션의 경우, 동종 데이터

마이그레이션 프로젝트를 사용하는 것이 좋습니다. 동종 데이터 마이그레이션은 네이티브 데이터베이스 도구를 사용하여 AWS DMS와 비교할 때 향상된 데이터 마이그레이션 성능과 정확성을 제공합니다.

MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS

MySQL 데이터베이스를 소스로 사용하기 전에 다음 사전 조건이 있는지 AWS DMS 확인합니다. 이러한 사전 조건은 자체 관리형 또는 AWS 관리형 소스에 적용됩니다.

복제 관리자 역할 AWS DMS 이 있는의 계정이 있어야 합니다. 역할에는 다음 권한이 있어야 합니다.

- REPLICATION CLIENT - 이 권한은 변경 데이터 캡처(CDC) 작업에만 필요합니다. 즉, 전체 로드 전용 작업에서는 이 권한이 필요하지 않습니다.

Note

MariaDB 버전 10.5.2+의 경우 BINLOG MONITOR를 사용할 수 있습니다. BINLOG MONITOR는 REPLICATION CLIENT를 대체합니다.

- REPLICATION SLAVE - 이 권한은 변경 데이터 캡처(CDC) 작업에만 필요합니다. 즉, 전체 로드 전용 작업에서는 이 권한이 필요하지 않습니다.
- SUPER - 이 권한은 MySQL 5.6.6 이전 버전에서만 필요합니다.

또한 AWS DMS 사용자에게 복제용으로 지정된 소스 테이블에 대한 SELECT 권한이 있어야 합니다.

MySQL별 마이그레이션 전 평가를 사용하는 경우 다음 권한을 부여합니다.

```
grant select on mysql.user to <dms_user>;
grant select on mysql.db to <dms_user>;
grant select on mysql.tables_priv to <dms_user>;
grant select on mysql.role_edges to <dms_user> #only for MySQL version 8.0.11 and higher
grant select on performance_schema.replication_connection_status to <dms_user>;
#Required for primary instance validation - MySQL version 5.7 and higher only
```

RDS 소스를 사용하고 MySQL별 사전 마이그레이션 평가를 실행하려는 경우 다음 권한을 추가합니다.

```
grant select on mysql.rds_configuration to <dms_user>; #Required for binary log retention check
```

BatchEnable 파라미터가 true인 경우 다음을 부여해야 합니다.

```
grant create temporary tables on `<schema>`.* to <dms_user>;
```

자체 관리형 MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS

AWS DMS용 원본으로 다음 자체 관리형 MySQL 호환 데이터베이스를 사용할 수 있습니다.

- MySQL Community Edition
- MySQL Standard Edition
- MySQL Enterprise Edition
- MySQL Cluster Carrier Grade Edition
- MariaDB Community Edition
- MariaDB Enterprise Edition
- MariaDB Column Store

CDC를 사용하려면 이진 로깅을 활성화해야 합니다. 이진 로깅을 활성화하려면, MySQL의 my.ini(Windows) 또는 my.cnf(UNIX) 파일에서 다음 파라미터를 구성해야 합니다.

파라미터	값
server_id	이 파라미터 값을 1 이상의 값으로 설정합니다.
log-bin	log-bin=E:\MySql_Logs\BinLog 과 같은 이진 로그 파일로 이 경로를 설정합니다. 파일 확장자를 포함해서는 안 됩니다.
binlog_format	이 파라미터를 ROW으로 설정합니다. binlog_format 을 STATEMENT 로 설정하면 대상에 데이터를 복제할 때 불일치가 발생할 수 있으므로 복제 중에 이 설정을 사용하는 것이 좋습니다. binlog_format 이 MIXED로 설정되면 데이터베이스 엔진도 비슷한 일관되지 않은 데이터를 대상에 기록합니다. 대상 데이터베이스에 일관되지 않은 데이터가 기록되도록 할 수 있는 STATEMENT 기반 로깅으로 데이터베이스 엔진이 자동으로 전환되기 때문입니다.
expire_logs_days	이 파라미터 값을 1 이상의 값으로 설정합니다. 디스크 공간의 과사용을 방지하려면 기본값 0을 사용하지 않는 것이 좋습니다.

파라미터	값
binlog_checksum	DMS 버전 3.4.7 이하에 대해 이 파라미터를 NONE으로 설정합니다.
binlog_row_image	이 파라미터를 FULL으로 설정합니다.
log_slave_updates	원본으로 MySQL 또는 MariaDB 읽기 전용 복제본을 사용하고 있는 경우 이 파라미터를 TRUE로 설정합니다.

MySQL 또는 MariaDB 읽기 전용 복제본을 기존 데이터 마이그레이션 및 지속적인 변경 사항 복제 모드를 사용하여 DMS 마이그레이션 작업의 소스로 사용하는 경우 데이터 손실 가능성이 있습니다. DMS는 다음 조건에서 전체 로드 또는 CDC 중에 트랜잭션을 작성하지 않습니다.

- DMS 태스크가 시작되기 전에 트랜잭션이 기본 인스턴스에 커밋되었습니다.
- 기본 인스턴스와 복제본 간의 지연으로 인해 DMS 태스크가 시작된 후까지 트랜잭션이 복제본에 커밋되지 않았습니다.

기본 인스턴스와 복제본 간의 지연 시간이 길수록 데이터 손실 가능성이 커집니다.

원본에서 NDB(클러스터 방식) 데이터베이스 엔진을 사용하는 경우, 해당 스토리지 엔진을 사용하는 테이블에 대해 CDC를 활성화하도록 다음 파라미터를 구성해야 합니다. 이러한 변경 사항을 MySQL의 my.ini(Windows) 또는 my.cnf(UNIX) 파일에 추가합니다.

파라미터	값
ndb_log_bin	이 파라미터를 ON으로 설정합니다. 이 값은 클러스터링된 테이블의 변경 사항이 이진 로그에 로깅되도록 보장해 줍니다.
ndb_log_update_as_write	이 파라미터를 OFF으로 설정합니다. 이 값은 이진 로그에서 UPDATE 문을 INSERT 문으로 쓰지 않도록 해줍니다.
ndb_log_updated_only	이 파라미터를 OFF으로 설정합니다. 이 값은 이진 로그가 변경된 열뿐만 아니라 전체 행을 포함하도록 해줍니다.

AWS관리형 MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS

AWS DMS다음 AWS관리형 MySQL 호환 데이터베이스를의 소스로 사용할 수 있습니다.

- MySQL Community Edition
- MariaDB Community Edition
- Amazon Aurora MySQL 호환 버전

AWS관리형 MySQL 호환 데이터베이스를 소스로 사용하는 경우 CDC에 대해 다음과 같은 사전 조건이 있는지 AWS DMS확인합니다.

- RDS for MySQL 및 RDS for MariaDB에 이진 로그를 활성화하려면 인스턴스 수준에서 자동 백업을 활성화하세요. Aurora MySQL 클러스터의 이진 로그를 활성화하려면 파라미터 그룹에서 `binlog_format` 변수를 변경합니다.

자동 백업 설정에 관한 자세한 내용은 Amazon RDS 사용 설명서의 [백업 작업](#) 섹션을 참조하세요.

Amazon RDS for MySQL 데이터베이스에 이진 로깅을 설정하는 방법에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [MySQL 이진 로깅 구성](#)을 참조하세요.

Aurora MySQL 클러스터의 이진 로깅을 설정하는 방법에 대한 자세한 내용은 [Amazon Aurora MySQL 클러스터에 대한 바이너리 로깅을 활성화하려면 어떻게 해야 하나요?](#)를 참조하세요.

- CDC를 사용하려면 이진 로깅을 켭니다. Amazon RDS for MySQL 데이터베이스에 이진 로깅을 설정하는 방법에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [MySQL 이진 로깅 형식 구성](#)을 참조하세요.
- 바이너리 로그를 사용할 수 있는지 확인합니다 AWS DMS. AWS관리형 MySQL 호환 데이터베이스는 이진 로그를 최대한 빨리 제거하므로 로그를 계속 사용할 수 있는 시간을 늘려야 합니다. 예를 들어, 로그 보존 시간을 24시간으로 늘리려면 다음 명령을 실행합니다.

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

- `binlog_format` 파라미터를 "ROW"로 설정합니다.

Note

MySQL 또는 MariaDB에서는 `binlog_format`이 동적 파라미터이므로 새 값을 적용하기 위해 재부팅할 필요가 없습니다. 하지만 새 값은 새 세션에만 적용됩니다. 복제를 위해 `binlog_format`을 `ROW`로 전환해도 값을 변경하기 전에 해당 세션이 시작된 경우, 데이터베이스는 `MIXED` 형식을 사용하여 후속 이진 로그를 계속 생성할 수 있습니다. 이렇게 하면 AWS DMS가 소스 데이터베이스의 모든 변경 사항을 올바르게 캡처하지 못할 수 있습니다. MariaDB 또는 MySQL 데이터베이스에서 `binlog_format` 설정을 변경하는 경우, 데이터베이스를 다시 시작하여 기존 세션을 모두 닫거나 DML(데이터 조작 언어) 작업을 수행하는 애플리케이션을 다시 시작해야 합니다. `binlog_format` 파라미터를 로 변경한 후 데이터베이스를 강제로 모든 세션을 다시 시작하면 데이터베이스가 올바른 형식을 사용하여 모든 후속 소스 데이터베이스 변경 사항을 기록하므로 이러한 변경 사항을 올바르게 캡처할 AWS DMS 수 있습니다.

- `binlog_row_image` 파라미터를 "Full"로 설정합니다.
- DMS 버전 3.4.7 이전의 경우 `binlog_checksum` 파라미터를 "NONE"으로 설정합니다. Amazon RDS MySQL의 파라미터 설정에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [백업 작업](#)을 참조하세요.
- Amazon RDS MySQL 또는 Amazon RDS MariaDB 읽기 전용 복제본을 소스로 사용하고 있는 경우, 읽기 전용 복제본에서 백업을 활성화하고 `log_slave_updates` 파라미터를 TRUE로 설정해야 합니다.

MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS

MySQL 데이터베이스를 원본으로 사용하는 경우 다음 사항을 고려하세요.

- Amazon RDS MySQL 5.5 이하의 경우, 변경 데이터 캡처(CDC)가 지원되지 않습니다. Amazon RDS MySQL의 경우, CDC를 활성화하려면 버전 5.6, 5.7 또는 8.0을 사용해야 합니다. CDC는 자체 관리형 MySQL 5.5 소스에서 지원됩니다.
- CDC의 경우, `CREATE TABLE`, `ADD COLUMN`, `DROP COLUMN` 열 데이터 형식 변경과 `renaming a column`이 지원됩니다. 하지만 `DROP TABLE`, `RENAME TABLE` 및 열 기본값, 열 Null 허용 여부, 문자 집합 등과 같은 다른 속성에 대한 업데이트는 지원되지 않습니다.
- 소스의 분할된 테이블의 경우 대상 테이블 준비 모드를 대상의 테이블 삭제로 설정하면 MySQL 대상에 파티션이 없는 간단한 테이블을 AWS DMS 생성합니다. 파티션된 테이블을 대상의 파티션된

테이블로 마이그레이션하려면, 대상 MySQL 데이터베이스에서 파티션된 테이블을 미리 생성합니다.

- ALTER TABLE *table_name* ADD COLUMN *column_name* 문을 사용하여 테이블의 시작(FIRST) 또는 중간(AFTER)에 열을 추가하는 것은 관계형 대상에 대해 지원되지 않습니다. 열은 항상 테이블 끝에 추가됩니다. 대상이 Amazon S3 또는 Amazon Kinesis Data Streams인 경우 FIRST 또는 AFTER를 사용하여 열을 추가할 수 있습니다.
- 테이블 이름에 대/소문자가 포함되어 있고, 대/소문자를 구분하지 않는 파일 이름이 있는 운영 체제에서 소스 엔진이 호스팅되는 경우, CDC가 지원되지 않습니다. 예로는 HFS+를 사용하는 Microsoft Windows 또는 OS X가 있습니다.
- Aurora MySQL 호환 버전 서버리스 v1은 전체 로드에서 사용할 수 있지만 CDC에는 사용할 수 없습니다. MySQL 사전 요구 사항을 활성화할 수 없기 때문입니다. 자세한 내용은 [파라미터 그룹 및 Aurora Serverless v1](#)을 참조하세요.

Aurora MySQL 호환 버전 서버리스 v2는 CDC를 지원합니다.

- 열의 AUTO_INCREMENT 속성은 대상 데이터베이스 열로 마이그레이션되지 않습니다.
- 표준 블록 스토리지에 이진 로그가 저장되지 않으면 변경 사항 캡처가 지원되지 않습니다. 예를 들어 이진 로그가 Amazon S3에 저장되면 CDC가 작동하지 않습니다.
- AWS DMS 는 기본적으로 InnoDB 스토리지 엔진을 사용하여 대상 테이블을 생성합니다. InnoDB가 아닌 스토리지 엔진을 사용해야 하는 경우, 테이블을 수동으로 생성하고 이를 [아무 작업 안 함](#) 모드를 사용하여 마이그레이션해야 합니다.
- DMS 마이그레이션 작업 모드가 기존 데이터 마이그레이션인 AWS DMS 경우에만 Aurora MySQL 복제본을 소스로 사용할 수 있습니다.
- MySQL 호환 소스가 전체 로드 도중 중지되는 경우, AWS DMS 작업은 오류로 중지되지 않습니다. 작업은 성공적으로 종료되지만 대상과 원본이 동기화되지 않았을 수 있습니다. 그러한 경우, 작업을 다시 시작하거나 영향을 받은 테이블을 다시 로드하세요.
- 열의 일부에 생성된 색인 값은 마이그레이션되지 않습니다. 예를 들어 CREATE INDEX first_ten_chars ON 색인 고객(이름(10))은 대상에서 생성되지 않습니다.
- 일부 경우, LOB를 복제하지 않도록 작업이 구성됩니다(작업 설정에서 "SupportLobs"가 false이거나 작업 콘솔에서 Don't include LOB columns가 선택되지 않음). 이러한 경우는 MEDIUMBLOB, LONGBLOB, MEDIUMTEXT 및 LONGTEXT 열을 대상으로 마이그레이션하지 AWS DMS 않습니다.

BLOB, TINYBLOB, TEXT 및 TINYTEXT 열은 영향을 받지 않으며 대상으로 마이그레이션됩니다.

- 임시 데이터 테이블 또는 시스템 버전이 지정된 테이블은 MariaDB 소스 및 대상 데이터베이스에서 지원되지 않습니다.

- 두 Amazon RDS Aurora MySQL 클러스터 간에 마이그레이션하는 경우, RDS Aurora MySQL 소스 엔드포인트는 복제본 인스턴스가 아니라 읽기/쓰기 인스턴스여야 합니다.
- AWS DMS 는 현재 MariaDB에 대한 뷰 마이그레이션을 지원하지 않습니다.
- AWS DMS 는 MySQL의 분할된 테이블에 대한 DDL 변경을 지원하지 않습니다. CDC 도중 파티션 DDL 변경으로 인한 테이블 일시 종단을 건너뛰려면 `skipTableSuspensionForPartitionDdl`을 `true`로 설정합니다.
- AWS DMS 는 버전 3.5.0 이상의 XA 트랜잭션만 지원합니다. 이전 버전은 XA 트랜잭션 AWS DMS 을 지원하지 않습니다. MariaDB 버전 10.6 이상에서는 XA 트랜잭션을 지원하지 않습니다. 자세한 내용은 [the section called “XA 트랜잭션 지원”](#) 다음을 참조하세요.
- AWS DMS 는 소스 데이터에 GTIDs를 사용하지 않습니다.
- AWS DMS 는 Aurora MySQL 향상된 바이너리 로그를 지원하지 않습니다.
- AWS DMS 는 이진 로그 트랜잭션 압축을 지원하지 않습니다.
- AWS DMS 는 InnoDB 스토리지 엔진을 사용하여 MySQL 데이터베이스에 대한 ON DELETE CASCADE 및 ON UPDATE CASCADE 이벤트를 전파하지 않습니다. 이러한 이벤트의 경우, MySQL은 하위 테이블의 계단식 작업을 반영하기 위해 binlog 이벤트를 생성하지 않습니다. 따라서 AWS DMS 는 해당 변경 사항을 하위 테이블에 복제할 수 없습니다. 자세한 내용은 [인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음](#) 단원을 참조하십시오.
- AWS DMS 는 계산된 (VIRTUAL 및 GENERATED ALWAYS) 열의 변경 사항을 캡처하지 않습니다. 이러한 제한을 해결하려면 다음을 수행하세요.
 - 대상 데이터베이스에서 대상 테이블을 미리 생성하고, `DO_NOTHING` 또는 `TRUNCATE_BEFORE_LOAD` 전체 로드 작업 설정을 사용하여 AWS DMS 작업을 생성합니다.
 - 계산된 열을 작업 범위에서 제거하는 변환 규칙을 추가합니다. 변환에 대한 자세한 내용은 [변환 규칙 및 작업](#) 섹션을 참조하세요.
- 내부 MySQL 제한으로 인하여 BINLOGs 4GB 크기 이하로 처리할 AWS DMS 수 있습니다. BINLOGs 4GB보다 크면 DMS 작업 실패 또는 기타 예측할 수 없는 동작이 발생할 수 있습니다. 4GB 보다 큰 BINLOG를 방지하려면 트랜잭션 크기를 줄여야 합니다.
- AWS DMS 는 스키마, 테이블 및 열 이름에서 백틱(`) 또는 작은따옴표(')를 지원하지 않습니다.
- AWS DMS 는 소스 데이터베이스의 보이지 않는 열에서 데이터를 마이그레이션하지 않습니다. 이러한 열을 마이그레이션 범위에 포함하려면 ALTER TABLE 문을 사용하여 이러한 열을 표시하세요.

XA 트랜잭션 지원

확장 아키텍처(XA) 트랜잭션은 여러 트랜잭션 리소스의 일련의 작업을 신뢰할 수 있는 단일 전역 트랜잭션으로 그룹화하는 데 사용할 수 있는 트랜잭션입니다. XA 트랜잭션은 2단계 커밋 프로토콜을

사용합니다. 일반적으로 열린 XA 트랜잭션이 있는 동안 변경 내용을 캡처하면 데이터가 손실될 수 있습니다. 데이터베이스에서 XA 트랜잭션을 사용하지 않는 경우, 기본값 TRUE를 사용하여 이 권한과 IgnoreOpenXaTransactionsCheck 구성을 무시할 수 있습니다. XA 트랜잭션이 있는 소스에서 복제를 시작하려면 다음을 수행합니다.

- AWS DMS 엔드포인트 사용자에게 다음 권한이 있는지 확인합니다.

```
grant XA_RECOVER_ADMIN on *.* to 'userName'@'%';
```

- 엔드포인트 설정 IgnoreOpenXaTransactionsCheck를 false로 설정합니다.

Note

AWS DMS 는 MariaDB 소스 DB 버전 10.6 이상에서 XA 트랜잭션을 지원하지 않습니다.

MySQL을의 소스로 사용할 때 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 MySQL 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --my-sql-settings '{"EndpointSetting": "value", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

MySQL을 소스로 하여 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

이름	설명
ConnectionTimeout	이 추가 연결 속성(ECA)을 사용하여 MySQL 인스턴스의 엔드포인트 연결 제한 시간을 초 단위로 설정합니다. 기본값은 10초입니다. ECA 예제: ConnectionTimeout=30 .
EventsPollInterval	데이터베이스가 유휴 상태일 때 이진 로그에서 새 변경 사항/이벤트를 확인하는 빈도를 지정합니다. 기본값: 5 유효한 값: 1~60

이름	설명
	예시: <code>--my-sql-settings '{"EventsPollInterval": 5}'</code> 이 예에서는 5초마다 바이너리 로그의 변경 사항을 AWS DMS 확인합니다.
ExecuteTimeout	AWS DMS 버전 3.4.7 이상의 경우는 MySQL 소스 엔드포인트에 대한 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값: 60 예시: <code>--my-sql-settings '{"ExecuteTimeout": 1500}'</code>
ServerTimezone	원본 MySQL 데이터베이스의 시간대를 지정합니다. 예시: <code>--my-sql-settings '{"ServerTimezone": "US/Pacific"}'</code>
AfterConnectScript	가 엔드포인트에 AWS DMS 연결한 직후 실행할 스크립트를 지정합니다. SQL 문의 성공 여부에 상관없이 마이그레이션 작업이 계속 실행됩니다. 유효한 값: 하나 이상의 유효한 SQL 문(세미콜론으로 끝남). 예시: <code>--my-sql-settings '{"AfterConnectScript": "ALTER SESSION SET CURRENT_SCHEMA=system"}'</code>

이름	설명
CleanSourceMetadataOnMismatch	불일치 발생 시 복제 인스턴스에서 테이블 메타데이터 정보를 지운 후 다시 생성합니다. 예컨대 테이블에서 alter DDL을 실행하면 복제 인스턴스에 캐시된 테이블에 관해 다른 정보가 생길 수 있는 경우를 들 수 있습니다. 부울. 기본값: false 예시: <code>--my-sql-settings '{"CleanSourceMetadataOnMismatch": false}'</code>
skipTableSuspensionForPartitionDdl	AWS DMS 는 MySQL의 분할된 테이블에 대한 DDL 변경을 지원하지 않습니다. AWS DMS 버전 3.4.6 이상의 경우 CDC. AWS DMS ignores partitioned-table-related 일시 중지를 true 건너뛰고 추가 바이너리 로그 변경을 계속 처리하도록 설정합니다. 기본값: false 예시: <code>--my-sql-settings '{"skipTableSuspensionForPartitionDdl": true}'</code>
IgnoreOpenXaTransactionsCheck	AWS DMS 버전 3.5.0 이상의 경우 태스크가 시작하는 동안 열려 있는 XA 트랜잭션을 무시해야 하는지 여부를 지정합니다. 소스에 XA 트랜잭션이 있는 경우, 이 값을 false로 설정하세요. 기본값: true 예시: <code>--my-sql-settings '{"IgnoreOpenXaTransactionsCheck": false}'</code>

MySQL용 소스 데이터 형식

다음 표에는를 사용할 때 지원되는 MySQL 데이터베이스 소스 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

MySQL 데이터 형식	AWS DMS 데이터 형식
INT	INT4
BIGINT	INT8
MEDIUMINT	INT4
TINYINT	INT1
SMALLINT	INT2
UNSIGNED TINYINT	UINT1
UNSIGNED SMALLINT	UINT2
UNSIGNED MEDIUMINT	UINT4
UNSIGNED INT	UINT4
UNSIGNED BIGINT	UINT8
DECIMAL(10)	NUMERIC (10,0)
BINARY	BYTES(1)
BIT	BOOLEAN
BIT(64)	BYTES(8)
BLOB	BYTES(65535)
LONGBLOB	BLOB
MEDIUMBLOB	BLOB

MySQL 데이터 형식	AWS DMS 데이터 형식
TINYBLOB	BYTES(255)
DATE	DATE
DATETIME	DATETIME 괄호 안의 값이 없는 DATETIME은 밀리초 없이 복제됩니다. 괄호 안의 값이 1~5인 DATETIME(예: DATETIME(5))은 밀리초를 사용하여 복제됩니다. DATETIME 열을 복제할 때 대상에서 시간이 동일하게 유지됩니다. UTC로 변환되지 않습니다.
TIME	STRING
TIMESTAMP	DATETIME TIMESTAMP 열을 복제할 때는 대상에서 시간이 UTC로 변환됩니다.
YEAR	INT2
DOUBLE	REAL8
FLOAT	REAL(DOUBLE) FLOAT 값이 다음 범위에 속하지 않는 경우, 변환을 사용하여 FLOAT를 STRING에 매핑하세요. 변환에 관한 자세한 내용은 변환 규칙 및 작업 섹션을 참조하세요. 지원되는 FLOAT 범위는 -1.79E+308~-2.23E-308, 0, 그리고 2.23E-308~1.79E+308입니다.
VARCHAR (45)	WSTRING (45)
VARCHAR (2000)	WSTRING (2000)

MySQL 데이터 형식	AWS DMS 데이터 형식
VARCHAR (4000)	WSTRING (4000)
VARBINARY (4000)	BYTES (4000)
VARBINARY (2000)	BYTES (2000)
CHAR	WSTRING
TEXT	WSTRING
LONGTEXT	NCLOB
MEDIUMTEXT	NCLOB
TINYTEXT	WSTRING(255)
GEOMETRY	BLOB
POINT	BLOB
LINESTRING	BLOB
POLYGON	BLOB
MULTIPOINT	BLOB
MULTILINESTRING	BLOB
MULTIPOLYGON	BLOB
GEOMETRYCOLLECTION	BLOB
ENUM	WSTRING (<i>length</i>) 여기서 <i>length</i> 는 ENUM에서 가장 긴 값의 길이입니다.

MySQL 데이터 형식	AWS DMS 데이터 형식
SET	WSTRING (<i>length</i>) 여기서 <i>length</i> 는 쉼표를 포함하여 SET에 있는 모든 값의 총 길이입니다.
JSON	CLOB

Note

경우에 따라 '0' 값으로 DATETIME 및 TIMESTAMP 데이터 형식(즉, 0000-00-00)을 지정할 수 있습니다. 이 경우, 복제 작업의 대상 데이터베이스가 DATETIME 및 TIMESTAMP 데이터 형식으로 '0' 값을 지원하는지 확인하세요. 그렇지 않으면, 이 값이 대상에서 null로 기록됩니다.

SAP ASE 데이터베이스를의 소스로 사용 AWS DMS

이전에는 Sybase라고 했던 SAP Adaptive Server Enterprise(ASE) 데이터베이스에서 데이터를 마이그레이션할 수 있습니다 AWS DMS. SAP ASE 데이터베이스를 소스로 사용하면 지원되는 다른 AWS DMS 대상 데이터베이스로 데이터를 마이그레이션할 수 있습니다.

가 소스로 AWS DMS 지원하는 SAP ASE 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

SAP ASE 데이터베이스 및 작업에 대한 자세한 내용은 다음 섹션을 AWS DMS참조하세요.

주제

- [SAP ASE 데이터베이스를 소스로 사용하기 위한 사전 조건 AWS DMS](#)
- [SAP ASE를 소스로 사용할 때의 제한 사항 AWS DMS](#)
- [SAP ASE를의 소스로 사용하는 데 필요한 권한 AWS DMS](#)
- [잘림 지점 제거](#)
- [SAP ASE를의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#)
- [SAP ASE용 소스 데이터 형식](#)

SAP ASE 데이터베이스를 소스로 사용하기 위한 사전 조건 AWS DMS

SAP ASE 데이터베이스를 소스로 사용하려면 다음을 AWS DMS수행합니다.

- sp_setreptable 명령을 사용하여 테이블에서 SAP ASE 복제를 활성화합니다. 자세한 내용은 [Sybase Infocenter Archive](#)를 참조하세요.
- SAP ASE 데이터베이스에서 RepAgent를 비활성화합니다. 자세한 내용은 [Stop and disable the RepAgent thread in the primary database](#)를 참조하세요.
- 비라틴 문자(예: 중국어)에 대해 구성된 Windows EC2 인스턴스에 SAP ASE 버전 15.7을 복제하려면 대상 컴퓨터에 SAP ASE 15.7 SP121을 설치해야 합니다.

 Note

지속적 변경 데이터 캡처(CDC) 복제의 경우, DMS는 dbcc logtransfer와 dbcc log를 실행하여 트랜잭션 로그에서 데이터를 읽습니다.

SAP ASE를 소스로 사용할 때의 제한 사항 AWS DMS

SAP ASE 데이터베이스를 AWS DMS에서 원본으로 사용 시 다음 제한 사항이 적용됩니다.

- 각 SAP ASE 데이터베이스에 대해 지속적 복제 또는 CDC를 사용하여 하나의 AWS DMS 작업만 실행할 수 있습니다. 여러 개의 전체 로드 전용 작업을 병렬로 실행할 수 있습니다.
- 테이블의 이름을 바꿀 수 없습니다. 예를 들어 다음 명령은 실패합니다.

```
sp_rename 'Sales.SalesRegion', 'SalesReg;
```

- 열의 이름을 바꿀 수 없습니다. 예를 들어 다음 명령은 실패합니다.

```
sp_rename 'Sales.Sales.Region', 'RegID', 'COLUMN';
```

- 이진 데이터 형식 문자열 끝에 있는 0 값은 대상 데이터베이스에 복제될 때 잘립니다. 예를 들어, 원본 테이블의 0x0000000000000000000000000000000010000001000000000은 대상 테이블에서 0x0000000000000000000000000000000010000001이 됩니다.
- 데이터베이스 기본값이 NULL 값을 허용하지 않도록 설정된 경우는 NULL 값을 허용하지 않는 열이 있는 대상 테이블을 AWS DMS 생성합니다. 따라서 전체 로드 또는 CDC 복제 작업에 빈 값이 포함된 경우 오류가 AWS DMS 발생합니다. 다음 명령을 사용하여 원본 데이터베이스에서 NULL 값을 허용함으로써 이러한 오류를 방지할 수 있습니다.

```
sp_dboption database_name, 'allow nulls by default', 'true'
go
use database_name
CHECKPOINT
go
```

- reorg rebuild 인덱스 명령은 지원되지 않습니다.
- AWS DMS 는 클러스터 또는 MSA(Multi-Site Availability)/Warm Standby를 소스로 사용하는 것을 지원하지 않습니다.
- 매핑 규칙에 AR_H_TIMESTAMP 변환 헤더 표현식을 사용하는 경우, 추가된 열에서는 밀리초가 캡처되지 않습니다.
- CDC 중에 병합 작업을 실행하면 복구할 수 없는 오류가 발생합니다. 대상을 다시 동기화하려면 전체 로드를 실행하세요.
- 데이터 행 잠금 체계를 사용하는 테이블에는 롤백 트리거 이벤트가 지원되지 않습니다.
- AWS DMS 는 소스 SAP 데이터베이스에서 작업 범위 내의 테이블을 삭제한 후에는 복제 작업을 재개할 수 없습니다. DMS 복제 작업을 중지하고 DML 작업(INSERT, UPDATE, DELETE)을 수행한 후 테이블을 삭제한 경우, 복제 작업을 다시 시작해야 합니다.

SAP ASE를의 소스로 사용하는 데 필요한 권한 AWS DMS

SAP ASE 데이터베이스를 AWS DMS 작업의 소스로 사용하려면 권한을 부여해야 합니다. AWS DMS 데이터베이스 정의에 지정된 사용자 계정에 SAP ASE 데이터베이스에서 다음 권한을 부여합니다.

- sa_role
- replication_role
- sybase_ts_role
- 기본적으로 sp_setreptable 저장 프로시저를 실행할 권한이 필요한 경우 SAP ASE 복제 옵션을 AWS DMS 활성화합니다. 자체가 아닌 데이터베이스 엔드포인트에서 직접 테이블 sp_setreptable에서를 실행하려면 enableReplication 추가 연결 속성을 사용할 AWS DMS 수 있습니다. 자세한 내용은 [SAP ASE를의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

잘림 지점 제거

작업이 시작되면 syslogshold 시스템 보기에서 복제 프로세스가 진행 중임을 나타내는 \$replication_truncation_point 항목을 AWS DMS 설정합니다. AWS DMS 가 작동하는 동안 대상에 이미 복사된 데이터의 양에 따라 복제 잘림 지점을 정기적으로 진행합니다.

\$replication_truncation_point 항목이 설정되면 데이터베이스 로그가 과도하게 커지지 않도록 AWS DMS 작업을 계속 실행합니다. AWS DMS 작업을 영구적으로 중지하려면 다음 명령을 실행하여 복제 잘림 지점을 제거합니다.

```
dbcc settrunc('ltm','ignore')
```

잘림 지점을 제거한 후에는 AWS DMS 작업을 재개할 수 없습니다. 로그는 체크포인트에서 계속 자동으로 잘립니다(자동 잘림이 설정된 경우).

SAP ASE를의 소스로 사용할 때 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 SAP ASE 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --sybase-settings '{"EndpointSetting": "value", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

SAP ASE를 소스로 하여 사용할 수 있는 엔드포인트 설정은 다음 표에 나와 있습니다.

이름	설명
Charset	이 속성을 국제 문자 집합에 해당하는 SAP ASE 이름으로 설정합니다. 일부 문자 집합에서는 Provider 엔드포인트 설정을 Adaptive Server Enterprise 16.0.02.0 0 Full 로 설정해야 합니다. 기본값: iso_1 예시: --sybase-settings '{"Charset": "utf8"}' 유효한 값: acsii_8

이름	설명
	• big5hk • cp437 • cp850 • cp852 • cp852 • cp855 • cp857 • cp858 • cp860 • cp864 • cp866 • cp869 • cp874 • cp932 • cp936 • cp950 • cp1250 • cp1251 • cp1252 • cp1253 • cp1254 • cp1255 • cp1256 • cp1257 • cp1258 • deckanji • euccns • eucgb • eucjis

이름	설명
	• eucksc • gb18030 • greek8 • iso_1 • iso88592 • iso88595 • iso88596 • iso88597 • iso88598 • iso88599 • iso15 • kz1048 • koi8 • roman8 • iso88599 • sjis • tis620 • turkish8 • utf8
	SAP ASE 데이터베이스에서 지원되는 문자 집합에 대한 추가 질문은 Adaptive Server Enterprise: Supported character sets를 참조하세요.

이름	설명
EnableReplication	데이터베이스 종료부터 테이블에 sp_setreptable 대 해를 활성화하려는 경우 이 속성을 설정합니다 AWS DMS. 기본값: true 유효값: true 또는 false 예시: --sybase-settings '{"EnableReplicati on": false}'
EncryptPassword	소스 데이터베이스에서 "net password encryption reqd"를 활성화한 경우, 이 속성을 설정합니다. 기본값: 0 유효값: 0, 1 또는 2 예시: --sybase-settings '{"EncryptPassword ": 1}' 이러한 파라미터 값에 대한 자세한 내용은 Adaptive Server Enterprise: Using the EncryptPassword Connectio n string property를 참조하세요.
Provider	ASE 15.7 이상 버전에서 Transport Layer Security(TLS) 1.2 를 사용하려면 이 속성을 설정하세요. AWS 에는 TLS 버전 1.2 이상이 필요하며, 버전 1.3이 권장됩니다. 기본값: Adaptive Server Enterprise 유효값: Adaptive Server Enterprise 16.03.06 , Adaptive Server Enterprise 16.0.02.00 Full 예시: --sybase-settings '{"Provider": "Adaptive Server Enterprise 16.03.06"}'

SAP ASE용 소스 데이터 형식

사용 AWS DMS 시 지원되는 SAP ASE 소스 데이터 형식 목록과 AWS DMS 데이터 형식의 기본 매핑은 다음 표를 참조하세요. AWS DMS 는 사용자 정의 형식(UDT) 데이터 형식의 열이 있는 SAP ASE 소스 테이블을 지원하지 않습니다. 이 데이터 형식을 사용하는 복제된 열은 NULL로 생성됩니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 대상 엔드포인트의 [마이그레이션에 적합한 대상](#) 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

SAP ASE 데이터 형식	AWS DMS 데이터 형식
BIGINT	INT8
UNSIGNED BIGINT	UINT8
INT	INT4
UNSIGNED INT	UINT4
SMALLINT	INT2
UNSIGNED SMALLINT	UINT2
TINYINT	UINT1
DECIMAL	NUMERIC
NUMERIC	NUMERIC
FLOAT	REAL8
DOUBLE	REAL8
REAL	REAL4
MONEY	NUMERIC
SMALLMONEY	NUMERIC

SAP ASE 데이터 형식	AWS DMS 데이터 형식
DATETIME	DATETIME
BIGDATETIME	DATETIME(6)
SMALLDATETIME	DATETIME
DATE	DATE
TIME	TIME
BIGTIME	TIME
CHAR	STRING
UNICHAR	WSTRING
NCHAR	WSTRING
VARCHAR	STRING
UNIVARCHAR	WSTRING
NVARCHAR	WSTRING
BINARY	BYTES
VARBINARY	BYTES
BIT	BOOLEAN
TEXT	CLOB
UNITEXT	NCLOB
IMAGE	BLOB

MongoDB를의 소스로 사용 AWS DMS

가 소스로 AWS DMS 지원하는 MongoDB 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

MongoDB 버전 지원에 대한 다음 사항에 유의하세요.

- AWS DMS 3.4.5 이상 버전은 MongoDB 버전 4.2 및 4.4를 지원합니다.
- AWS DMS 3.4.5 이상 버전과 MongoDB 4.2 이상 버전은 분산 트랜잭션을 지원합니다. MongoDB 분산 트랜잭션에 대한 자세한 내용은 [MongoDB 설명서의 Transactions](#)를 참조하세요.
- AWS DMS 3.5.0 이상 버전은 3.6 이전 버전의 MongoDB를 지원하지 않습니다.
- AWS DMS 3.5.1 이상 버전은 MongoDB 버전 5.0을 지원합니다.
- AWS DMS 3.5.2 이상 버전은 MongoDB 버전 6.0을 지원합니다.
- AWS DMS 3.5.4 이상 버전은 MongoDB 버전 7.0 및 8.0을 지원합니다.

MongoDB가 생소하다면 다음과 같은 MongoDB 데이터베이스의 중요 개념을 잘 이해하시기 바랍니다.

- MongoDB의 레코드는 필드와 값으로 구성된 데이터 구조인 문서입니다. 필드의 값에는 다른 문서, 배열 및 문서의 배열이 포함될 수 있습니다. 문서는 대체로 관계형 데이터베이스 테이블의 행에 상응합니다.
- MongoDB의 모음은 문서 그룹이며 대략 관계형 데이터베이스 테이블에 상응합니다.
- MongoDB의 데이터베이스는 모음 세트이며, 대략 관계형 데이터베이스의 스키마에 해당합니다.
- 내부적으로 MongoDB 문서는 이진 JSON(BSON) 파일에 문서의 각 필드별로 형식을 포함하는 압축 형식으로 저장됩니다. 각 문서에는 고유 ID가 있습니다.

AWS DMS 는 MongoDB를 소스로 사용할 때 문서 모드 또는 테이블 모드라는 두 가지 마이그레이션 모드를 지원합니다. MongoDB 엔드포인트를 생성할 때 또는 AWS DMS 콘솔에서 메타데이터 모드 파라미터를 설정하여 사용할 마이그레이션 모드를 지정합니다. 선택에 따라 엔드포인트 구성 패널에서 `_id`를 별도의 열로 확인 표시 버튼을 선택하여 프라이머리 키 역할을 하는 `_id`라는 두 번째 열을 생성할 수 있습니다.

마이그레이션 모드 선택은 아래 설명된 대로 결과로 얻는 대상 데이터의 형식에 영향을 미칩니다.

문서 모드

문서 모드에서 MongoDB 문서는 있는 그대로 마이그레이션됩니다. 즉 해당 문서 데이터는 대상 테이블에서 `_doc`라는 단일 열로 통합됩니다. 문서 모드는 MongoDB를 소스 엔드포인트로 사용할 때의 기본 설정입니다.

예를 들어, `myCollection`이라는 MongoDB 모음의 문서를 가정해 보세요.

```
db.myCollection.find()
{ "_id" : ObjectId("5a94815f40bd44d1b02bdfe0"), "a" : 1, "b" : 2, "c" : 3 }
{ "_id" : ObjectId("5a94815f40bd44d1b02bdfe1"), "a" : 4, "b" : 5, "c" : 6 }
```

데이터를 문서 모드를 사용하는 관계형 데이터베이스 테이블로 마이그레이션 하면, 데이터의 구조는 다음과 같습니다. MongoDB 문서의 데이터 필드는 `_doc` 열로 통합됩니다.

oid_id	_doc
5a94815f40bd44d1b02bdfe0	{ "a" : 1, "b" : 2, "c" : 3 }
5a94815f40bd44d1b02bdfe1	{ "a" : 4, "b" : 5, "c" : 6 }

선택적으로 추가 연결 속성 `extractDocID`를 `true`로 설정하여 프라이머리 키 역할을 하는 `"_id"`라는 두 번째 열을 생성할 수 있습니다. CDC를 사용할 경우, 이 파라미터를 `true`로 설정합니다.

[다중 문서 트랜잭션](#)을 생성하는 소스와 함께 CDC를 사용하는 경우 `ExtractDocId` 파라미터를 `true`로 설정해야 합니다. 이 파라미터를 활성화하지 않으면 다중 문서 트랜잭션이 발생하면 AWS DMS 작업이 실패합니다.

문서 모드에서는 다음과 같이 컬렉션의 생성 및 이름 변경을 AWS DMS 관리합니다.

- 소스 데이터베이스에 새 컬렉션을 추가하는 경우는 컬렉션에 대한 새 대상 테이블을 AWS DMS 생성하고 모든 문서를 복제합니다.
- 소스 데이터베이스에서 기존 모음의 이름을 바꾸면 AWS DMS 는 대상 테이블의 이름을 바꾸지 않습니다.

대상 엔드포인트가 Amazon DocumentDB인 경우, 문서 모드에서 마이그레이션을 실행하세요.

테이블 모드

테이블 모드에서는 MongoDB 문서의 각 최상위 필드를 대상 테이블의 열로 AWS DMS 변환합니다. 필드가 중첩된 경우 중첩된 값을 단일 열로 AWS DMS 확장합니다. AWS DMS 그런 다음 대상 테이블의 열 세트에 키 필드와 데이터 유형을 추가합니다.

각 MongoDB 문서에 대해서는 각 키와 유형을 대상 테이블의 열 세트에 AWS DMS 추가합니다. 예를 들어 테이블 모드를 사용하면가 이전 예제를 다음 테이블로 AWS DMS 마이그레이션합니다.

oid_id	a	b	c
5a94815f4 0bd44d1b02bdfe0	1	2	3
5a94815f4 0bd44d1b02bdfe1	4	5	6

중첩된 값은 점으로 분리된 키 이름을 포함하는 열에 평면화됩니다. 열은 마침표로 각각 분리된 연속된 평면화된 필드 이름으로 이름이 지정됩니다. 예를 들어,와 같은 중첩 값 필드가 있는 JSON 문서를 라는 열{"a" : {"b" : {"c": 1}}}로 AWS DMS 마이그레이션합니다. a.b.c.

대상 열을 생성하기 위해서는 지정된 수의 MongoDB 문서를 AWS DMS 스캔하고 모든 필드 및 유형 세트를 생성합니다. AWS DMS 그런 다음이 세트를 사용하여 대상 테이블의 열을 생성합니다. 콘솔을 사용해 MongoDB 소스 엔드포인트를 생성하거나 수정한다면, 스캔(검사)할 문서의 수를 지정할 수 있습니다. 기본값은 1000 개 문서입니다. 를 사용하는 경우 추가 연결 속성을 사용할 AWS CLI수 있습니다docsToInvestigate.

테이블 모드에서는 다음과 같이 문서 및 컬렉션을 AWS DMS 관리합니다.

- 기존 모음에 문서를 추가할 때 문서가 복제됩니다. 대상에 없는 필드가 있으면 이 필드는 복제되지 않습니다.
- 문서를 업데이트하면 업데이트된 문서가 복제됩니다. 대상에 없는 필드가 있으면 이 필드는 복제되지 않습니다.
- 문서 삭제는 완벽하게 지원됩니다.
- 새 모음을 CDC 작업 중에 추가해도 대상에 테이블이 새로 생성되지 않습니다.
- 변경 데이터 캡처(CDC) 단계에서 AWS DMS 는 컬렉션 이름 변경을 지원하지 않습니다.

주제

- [MongoDB를의 소스로 사용할 때 필요한 권한 AWS DMS](#)
- [CDC를 위한 MongoDB 복제본 세트 구성](#)
- [MongoDB를 소스로 사용할 때의 보안 요구 사항 AWS DMS](#)
- [MongoDB 모음을 분할하고 병렬로 마이그레이션](#)
- [MongoDB를 소스로 사용할 때 여러 데이터베이스 마이그레이션 AWS DMS](#)
- [MongoDB를의 소스로 사용할 때의 제한 사항 AWS DMS](#)
- [MongoDB를의 소스로 사용할 때 엔드포인트 구성 설정 AWS DMS](#)
- [MongoDB용 소스 데이터 형식](#)

MongoDB를의 소스로 사용할 때 필요한 권한 AWS DMS

MongoDB 소스를 사용한 AWS DMS 마이그레이션의 경우 루트 권한이 있는 사용자 계정 또는 마이그레이션할 데이터베이스에 대한 권한만 있는 사용자를 생성할 수 있습니다.

다음 코드는 루트 계정이 되도록 사용자를 생성합니다.

```
use admin
db.createUser(
  {
    user: "root",
    pwd: "password",
    roles: [ { role: "root", db: "admin" } ]
  }
)
```

MongoDB 3.x 소스의 경우, 다음 코드는 마이그레이션할 데이터베이스에 대한 최소 권한이 있는 사용자를 생성합니다.

```
use database_to_migrate
db.createUser(
  {
    user: "dms-user",
    pwd: "password",
    roles: [ { role: "read", db: "local" }, "read" ]
  })
```

MongoDB 4.x 소스의 경우, 다음 코드는 최소 권한을 가진 사용자를 생성합니다.


```
{ resource: { db: "", collection: "" }, actions: [ "find", "changeStream" ] }
```

예를 들어 “admin” 데이터베이스에 다음 역할을 생성합니다.

```
use admin
db.createRole(
{
  role: "changestreamrole",
  privileges: [
    { resource: { db: "", collection: "" }, actions: [ "find","changeStream" ] }
  ],
  roles: []
}
)
```

그리고 역할이 생성되면 마이그레이션할 데이터베이스에 사용자를 생성합니다.

```
use test
> db.createUser(
{
  user: "dms-user12345",
  pwd: "password",
  roles: [ { role: "changestreamrole", db: "admin" }, "read"]
})
```

CDC를 위한 MongoDB 복제본 세트 구성

MongoDB에서 지속적 복제 또는 CDC를 사용하려면 MongoDB 작업 로그(oplog)에 대한 액세스 권한이 AWS DMS 필요합니다. oplog를 생성하려면 복제 세트를 배포하여 합니다(복제 세트가 없는 경우). 자세한 내용은 [MongoDB 설명서](#)를 참조하세요.

CDC는 소스 엔드포인트로 설정된 MongoDB 복제본 세트의 기본 또는 보조 노드와 함께 사용할 수 있습니다.

독립형 인스턴스를 복제본 집합으로 변환하려면

1. 명령줄을 사용하여 mongo에 연결합니다.

```
mongo localhost
```

2. mongod 서비스를 중단합니다.

```
service mongod stop
```

3. 다음 명령을 사용하여 mongod를 다시 시작합니다.

```
mongod --replSet "rs0" --auth -port port_number
```

4. 다음 명령을 사용하여 복제본 집합에 대한 연결을 테스트합니다.

```
mongo -u root -p password --host rs0/localhost:port_number
--authenticationDatabase "admin"
```

문서 모드 마이그레이션을 계획하고 있다면 MongoDB 엔드포인트 생성 시 `_id as a separate column` 옵션을 선택하세요. 이 옵션을 선택하면 프라이머리 키로 작동하는 두 번째 열인 `_id`가 생성됩니다. 이 두 번째 열은 데이터 조작 언어(DML) 작업을 지원하기 AWS DMS 위해에 필요합니다.

Note

AWS DMS는 작업 로그(oplog)를 사용하여 지속적 복제 중에 변경 사항을 캡처합니다. 가 레코드를 AWS DMS 읽기 전에 MongoDB가 oplog에서 레코드를 풀러시하면 작업이 실패합니다. 24시간 이상 변경 사항이 유지되도록 oplog의 크기를 조정하는 것이 좋습니다.

MongoDB를 소스로 사용할 때의 보안 요구 사항 AWS DMS

AWS DMS는 MongoDB에 대해 두 가지 인증 방법을 지원합니다. 이 두 가지 인증 메서드는 암호를 암호화하는 데 사용되므로 `authType` 파라미터를 `PASSWORD`로 설정할 때만 사용됩니다.

MongoDB 인증 메서드는 다음과 같습니다.

- MONGODB-CR - 이전 버전과의 호환성을 위해
- SCRAM-SHA-1 - MongoDB 버전 3.x 및 4.0 사용 시 기본값

인증 방법을 지정하지 않으면 AWS DMS는 MongoDB 소스 버전에 기본 방법을 사용합니다.

MongoDB 모음을 분할하고 병렬로 마이그레이션

마이그레이션 작업의 성능을 개선하기 위해 MongoDB 소스 엔드포인트는 테이블 매핑에서 병렬 전체 로드 옵션 두 가지를 지원합니다.

다시 말해 JSON 설정에서 병렬 전체 로드를 위해 테이블 매핑에 자동 분할 또는 범위 분할을 사용하여 모음을 병렬로 마이그레이션할 수 있습니다. 자동 분할을 사용하면 각 스레드에서 마이그레이션을 위해 소스를 자동으로 분할 AWS DMS 하는 기준을 지정할 수 있습니다. 범위 분할을 사용하면 DMS AWS DMS 가 각 스레드에서 마이그레이션하도록 각 세그먼트의 특정 범위를 지정할 수 있습니다. 이러한 설정에 대한 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 섹션을 참조하세요.

자동 분할 범위를 사용하여 MongoDB 데이터베이스를 병렬로 마이그레이션

AWS DMS 가 각 스레드의 데이터를 자동으로 파티셔닝(분할)하는 기준을 지정하여 문서를 병렬로 마이그레이션할 수 있습니다. 특히 스레드당 마이그레이션할 문서 수를 지정합니다. 이 접근 방식을 사용하면 스레드당 최대 성능을 위해 세그먼트 경계를 최적화하려고 AWS DMS 시도합니다.

테이블 매핑에서 다음 테이블 설정 옵션을 사용하여 분할 기준을 지정할 수 있습니다.

테이블 설정 옵션	설명
"type"	(필수) MongoDB를 소스로 사용하는 경우 "partitions-auto" 로 설정합니다.
"number-of-partitions"	(선택 사항) 마이그레이션에 사용되는 총 파티션 (세그먼트) 수. 기본값은 16입니다.
"collection-count-from-meta data"	(선택 사항) 이 옵션을 true로 설정하면 AWS DMS 는 추정 모음 개수를 사용하여 파티션 수를 결정합니다. 이 옵션이 false로 설정된 경우 실제 컬렉션 수를 AWS DMS 사용합니다. 기본값은 true입니다.
"max-records-skip-per-page"	(선택 사항) 각 파티션의 경계를 결정할 때 한 번에 건너뛴 레코드 수입니다. 페이지가 매겨진 건너뛰기 접근 방식을 AWS DMS 사용하여 파티션의 최소 경계를 결정합니다. 기본값은 10,000입니다. 상대적으로 큰 값을 설정하면 커서 시간이 초과되고 작업이 실패할 수 있습니다. 상대적으로 작은 값을 설정하면 페이지당 작업 수가 늘어나고 전체 로드 속도가 느려집니다.

테이블 설정 옵션	설명
"batch-size"	(선택 사항) 배치 하나에서 반환되는 문서 수를 제한합니다. 배치마다 서버를 왕복해야 합니다. 배치 크기가 0인 경우, 커서는 서버에서 정의한 최대 배치 크기를 사용합니다. 기본값은 0입니다.

다음 예제는 자동 분할을 위한 테이블 매핑을 보여 줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "admin",
        "table-name": "departments"
      },
      "rule-action": "include",
      "filters": []
    },
    {
      "rule-type": "table-settings",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "admin",
        "table-name": "departments"
      },
      "parallel-load": {
        "type": "partitions-auto",
        "number-of-partitions": 5,
        "collection-count-from-metadata": "true",
        "max-records-skip-per-page": 1000000,
        "batch-size": 50000
      }
    }
  ]
}
```

자동 분할에는 다음과 같은 제한이 있습니다. 각 세그먼트의 마이그레이션은 모음 개수와 모음의 최소 `_id`를 개별적으로 가져옵니다. 그런 다음 페이지 매김 건너뛰기를 사용하여 해당 세그먼트의 최소 경계를 계산합니다.

따라서 모음의 모든 세그먼트 경계를 계산할 때까지 각 모음의 최소 `_id` 값이 일정하게 유지되어야 합니다. 세그먼트 경계를 계산하는 동안 모음의 최소 `_id` 값을 변경하면 데이터가 손실되거나 행 중복 오류가 발생할 수 있습니다.

범위 분할을 사용하여 MongoDB 데이터베이스를 병렬로 마이그레이션

스레드의 각 세그먼트마다 범위를 지정하여 문서를 병렬로 마이그레이션할 수 있습니다. 이 접근 방식을 사용하면 스레드당 선택한 문서 범위에 따라 AWS DMS 특정 문서가 각 스레드에서 마이그레이션하도록 지시할 수 있습니다.

다음 예제에 나온 MongoDB 모음에는 항목 7개와 프라이머리 키 `_id`가 있습니다.

Key	Value	Type
▼ (1) ObjectId("5f805c74873173399a278d78")	{ 3 fields }	Object
_id	ObjectId("5f805c74873173399a278d78")	ObjectId
num	1	Int32
name	a	String
▼ (2) ObjectId("5f805c97873173399a278d79")	{ 3 fields }	Object
_id	ObjectId("5f805c97873173399a278d79")	ObjectId
num	2	Int32
name	b	String
▼ (3) ObjectId("5f805cb0873173399a278d7a")	{ 3 fields }	Object
_id	ObjectId("5f805cb0873173399a278d7a")	ObjectId
num	3	Int32
name	c	String
▼ (4) ObjectId("5f805cbb873173399a278d7b")	{ 3 fields }	Object
_id	ObjectId("5f805cbb873173399a278d7b")	ObjectId
num	4	Int32
name	d	String
▼ (5) ObjectId("5f805cc5873173399a278d7c")	{ 3 fields }	Object
_id	ObjectId("5f805cc5873173399a278d7c")	ObjectId
num	5	Int32
name	e	String
▼ (6) ObjectId("5f805cd0873173399a278d7d")	{ 3 fields }	Object
_id	ObjectId("5f805cd0873173399a278d7d")	ObjectId
num	6	Int32
name	f	String
▼ (7) ObjectId("5f805cdd873173399a278d7e")	{ 3 fields }	Object
_id	ObjectId("5f805cdd873173399a278d7e")	ObjectId
num	7	Int32
name	g	String

가 병렬로 마이그레이션 AWS DMS 하기 위해 컬렉션을 3개의 특정 세그먼트로 분할하려면 마이그레이션 작업에 테이블 매핑 규칙을 추가할 수 있습니다. 이 방법은 다음 JSON 예제에 나와 있습니다.

```
{ // Task table mappings:
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "testdatabase",
        "table-name": "testtable"
      },
      "rule-action": "include"
    }
  ]
}
```

```

    }, // "selection" : "rule-type"
    {
      "rule-type": "table-settings",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "testdatabase",
        "table-name": "testtable"
      },
      "parallel-load": {
        "type": "ranges",
        "columns": [
          "_id",
          "num"
        ],
        "boundaries": [
          // First segment selects documents with _id less-than-or-equal-to
          5f805c97873173399a278d79
          // and num less-than-or-equal-to 2.
          [
            "5f805c97873173399a278d79",
            "2"
          ],
          // Second segment selects documents with _id > 5f805c97873173399a278d79 and
          // _id less-than-or-equal-to 5f805cc5873173399a278d7c and
          // num > 2 and num less-than-or-equal-to 5.
          [
            "5f805cc5873173399a278d7c",
            "5"
          ],
          // Third segment is implied and selects documents with _id >
          5f805cc5873173399a278d7c.
        ] // : "boundaries"
      } // : "parallel-load"
    } // "table-settings" : "rule-type"
  ] // : "rules"
} // : Task table mappings

```

이 테이블 매핑 정의는 소스 모음을 세 개의 세그먼트로 분할하고 병렬로 마이그레이션합니다. 분할 경계는 다음과 같습니다.

```
Data with _id less-than-or-equal-to "5f805c97873173399a278d79" and num less-than-or-equal-to 2 (2 records)
Data with _id > "5f805c97873173399a278d79" and num > 2 and _id less-than-or-equal-to "5f805cc5873173399a278d7c" and num less-than-or-equal-to 5 (3 records)
Data with _id > "5f805cc5873173399a278d7c" and num > 5 (2 records)
```

마이그레이션 작업이 완료되면 다음 예제와 같이 작업 로그에서 테이블이 병렬로 로드된 것을 확인할 수 있습니다. 소스 테이블에서 각 세그먼트를 언로드하는 데 사용된 MongoDB find 절도 확인할 수 있습니다.

```
[TASK_MANAGER    ] I: Start loading segment #1 of 3 of table
'testdatabase'. 'testtable' (Id = 1) by subtask 1. Start load timestamp
0005B191D638FE86 (replicationtask_util.c:752)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is initialized.
(mongodb_unload.c:157)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is: { "_id" :
{ "$lte" : { "$oid" : "5f805c97873173399a278d79" } }, "num" : { "$lte" :
{ "$numberInt" : "2" } } } (mongodb_unload.c:328)

[SOURCE_UNLOAD   ] I: Unload finished for segment #1 of segmented table
'testdatabase'. 'testtable' (Id = 1). 2 rows sent.

[TASK_MANAGER    ] I: Start loading segment #1 of 3 of table
'testdatabase'. 'testtable' (Id = 1) by subtask 1. Start load timestamp
0005B191D638FE86 (replicationtask_util.c:752)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is initialized.
(mongodb_unload.c:157)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is: { "_id" : { "$lte" :
{ "$oid" : "5f805c97873173399a278d79" } }, "num" : { "$lte" : { "$numberInt" :
"2" } } } (mongodb_unload.c:328)

[SOURCE_UNLOAD   ] I: Unload finished for segment #1 of segmented table
'testdatabase'. 'testtable' (Id = 1). 2 rows sent.

[TARGET_LOAD     ] I: Load finished for segment #1 of segmented table
'testdatabase'. 'testtable' (Id = 1). 1 rows received. 0 rows skipped. Volume
transferred 480.
```



```
[TASK_MANAGER    ] I: Load finished for segment #1 of table
'testdatabase'. 'testtable' (Id = 1) by subtask 1. 2 records transferred.
```

현재는 다음 MongoDB 데이터 형식을 세그먼트 키 열로 AWS DMS 지원합니다.

- 배정밀도 실수
- 문자열
- ObjectId
- 32비트 정수
- 64비트 정수

MongoDB를 소스로 사용할 때 여러 데이터베이스 마이그레이션 AWS DMS

AWS DMS 버전 3.4.5 이상에서는 지원되는 모든 MongoDB 버전에 대해 단일 작업으로 여러 데이터베이스를 마이그레이션할 수 있습니다. 여러 데이터베이스를 마이그레이션하려면 다음 단계에 따릅니다.

1. MongoDB 소스 엔드포인트를 생성할 때 다음 중 하나를 수행합니다.
 - DMS 콘솔의 엔드포인트 생성 페이지에서 엔드포인트 구성 아래 데이터베이스 이름을 비워 둡니다.
 - 명령을 사용하여 AWS CLI CreateEndpoint의 DatabaseName 파라미터에 빈 문자열 값을 할당합니다MongoDBSettings.
2. MongoDB 소스에서 마이그레이션하려는 각 데이터베이스의 데이터베이스 이름을 작업 테이블 매핑의 스키마 이름으로 지정합니다. 콘솔의 안내 입력을 사용하거나 JSON에서 직접 이 작업을 수행할 수 있습니다. 안내 입력에 대한 자세한 내용은 [콘솔에서 테이블 선택 및 변환 규칙 지정](#) 섹션을 참조하세요. JSON에 대한 자세한 내용은 [선택 규칙 및 작업](#) 섹션을 참조하세요.

예를 들면 다음과 같이 JSON을 지정하여 MongoDB 데이터베이스 3개를 마이그레이션할 수 있습니다.

Example스키마에서 모든 테이블 마이그레이션

다음 JSON은 소스 엔드포인트에 있는 Customers, Orders, Suppliers 데이터베이스의 모든 테이블을 대상 엔드포인트로 마이그레이션합니다.

```
{
```

```

"rules": [
  {
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "Customers",
      "table-name": "%"
    },
    "rule-action": "include",
    "filters": []
  },
  {
    "rule-type": "selection",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
      "schema-name": "Orders",
      "table-name": "%"
    },
    "rule-action": "include",
    "filters": []
  },
  {
    "rule-type": "selection",
    "rule-id": "3",
    "rule-name": "3",
    "object-locator": {
      "schema-name": "Inventory",
      "table-name": "%"
    },
    "rule-action": "include",
    "filters": []
  }
]
}

```

MongoDB를의 소스로 사용할 때의 제한 사항 AWS DMS

MongoDB를 소스로 사용할 때의 제한 사항은 AWS DMS 다음과 같습니다.

- 테이블 모드에서 모음에 있는 문서의 데이터 형식은 동일한 필드의 값에 사용하는 데이터 형식과 일치해야 합니다. 예를 들어 모음의 문서에 '{ a:{ b:value ... }'가 포함된 경우, 해당 a.b 필

드의 *value*를 참조하는 모음의 모든 문서는 모음의 어디에 표시되든 *value*에 동일한 데이터 형식을 사용해야 합니다.

- `_id` 옵션이 별도의 열로 설정된 경우 ID 문자열이 200자를 초과할 수 없습니다.
- 객체 ID와 배열 형식 키는 테이블 모드에서 `oid` 및 `array` 접두사가 붙은 열로 변환됩니다.

내부적으로 이러한 열은 접두사 이름으로 참조됩니다. 이러한 열을 참조 AWS DMS 하는에서 변환 규칙을 사용하는 경우 접두사가 붙은 열을 지정해야 합니다. 예를 들면 `${oid__id}`를 지정하고 `${_id}`를 지정하지 않거나 `${array__addresses}`를 지정하고 `${_addresses}`는 지정하지 않습니다.

- 모음 이름과 키 이름에는 달러 기호(\$)를 포함할 수 없습니다.
- AWS DMS 는 RDBMS 대상이 있는 테이블 모드에서 대소문자가 다른(상위, 아래) 동일한 필드가 포함된 컬렉션을 지원하지 않습니다. 예를 들어 AWS DMS 는 `Field1` 및 `라`는 두 개의 컬렉션이 있는 것을 지원하지 않습니다 `field1`.
- 테이블 모드와 문서 모드에는 앞서 설명한 제한이 있습니다.
- 자동 분할을 사용한 병렬 마이그레이션에는 앞에서 설명한 제한 사항이 있습니다.
- MongoDB에는 소스 필터가 지원되지 않습니다.
- AWS DMS 는 중첩 수준이 97보다 큰 문서를 지원하지 않습니다.
- AWS DMS 에서는 DocumentDB가 아닌 대상으로 마이그레이션할 때 UTF-8 인코딩 소스 데이터가 필요합니다. non-UTF-8 소스의 경우 마이그레이션하기 전에 UTF-8로 변환하거나 대신 Amazon DocumentDB로 마이그레이션합니다.
- AWS DMS 는 MongoDB 버전 5.0의 다음 기능을 지원하지 않습니다.
 - 라이브 리샤딩
 - 클라이언트 측 필드 수준 암호화(CSFLE)
 - 시계열 모음 마이그레이션

Note

Amazon DocumentDB는 시계열 모음을 지원하지 않기 때문에 전체 로드 단계에서 마이그레이션된 시계열 모음은 Amazon DocumentDB에서 일반 모음으로 변환됩니다.

MongoDB를의 소스로 사용할 때 엔드포인트 구성 설정 AWS DMS

MongoDB 소스 엔드포인트를 설정할 때 AWS DMS 콘솔을 사용하여 여러 엔드포인트 구성 설정을 지정할 수 있습니다.

다음 표에서는 MongoDB 데이터베이스를 AWS DMS 소스로 사용할 때 사용할 수 있는 구성 설정을 설명합니다.

설정(속성)	유효값	기본값과 설명
인증 모드	"none" "password"	값 "password" 는 사용자 이름과 암호를 입력하라는 메시지를 표시합니다. "none"을 지정하면 사용자 이름 및 암호 파라미터가 사용되지 않습니다.
인증 소스	유효한 MongoDB 데이터베이스 이름입니다.	인증을 위한 자격 증명 정보를 검증하는 데 사용할 MongoDB 데이터베이스의 이름입니다. 기본값은 "admin"입니다.
인증 메커니즘	"default" "mongodb_cr" "scram_sha_1"	인증 메커니즘입니다. "default" 값은 "scram_sha_1" 입니다. authType이 "no"로 설정된 경우에는 이 설정이 사용되지 않습니다.
메타데이터 모드	문서 및 테이블	문서 모드 또는 테이블 모드를 선택합니다.
스캔할 문서 수(docsToInvestigate)	0보다 큰 양의 정수입니다.	이 옵션은 테이블 모드에서 대상 테이블 정의를 정의할 때만 사용하세요.
_id를 별도의 열로	상자의 확인 표시	프라이머리 키 역할을 하는 _id라는 두 번째 열을 생성하는 선택적 확인 표시 상자입니다.
ExtractDocID	true false	false - 이 속성은 NestingLevel 이 "none"으로 설정된 경우에 사용합니다. 다중 문서 트랜잭션 을 생성하는 소스와 함께 CDC를 사용하는 경우 ExtractDocId 파라미터를 로 설정해야 합니다true. 이 파라미터를 활성화하지 않으면 다중 문서 트랜잭션이 발생하면 AWS DMS 작업이 실패합니다.
socketTimeoutMS	0보다 크거나 같은 정수입니다.	이 설정은 밀리초 단위이며 MongoDB 클라이언트의 연결 제한 시간을 구성합니다. 값이 0보다 작거나 같으면 MongoDB 클라이언트 기본값이 사용됩니다.

설정(속성)	유효값	기본값과 설명
	추가 연결 속성 (ECA) 전용입니다.	
UseUpdateLookUp	true false	true인 경우 CDC 업데이트 이벤트 중에는 업데이트된 문서 전체를 대상으로 AWS DMS 복사합니다. false로 설정하면 MongoDB 업데이트 명령을 AWS DMS 사용하여 대상의 문서에서 수정된 필드만 업데이트합니다.
ReplicateShardCollections	true false	true인 경우는 데이터를 샤드 컬렉션에 AWS DMS 복제합니다. 대상 엔드포인트가 DocumentDB 탄력적 클러스터인 경우에만 이 설정을 AWS DMS 사용합니다. 이 설정이 true인 경우 다음 사항에 유의하세요. - TargetTablePrepMode 를 nothing으로 설정해야 합니다. - AWS DMS 는 useUpdateLookup 로 자동 설정됩니다 false.
useTransactionVerification	true false	false인 경우는 변경 스트림과 oplog 간의 확인을 비활성화합니다.  Note 기본 DMS 동작은 이러한 시나리오에서 작업에 실패하는 것이므로 변경 스트림과 oplog 항목 간에 불일치가 발생하는 경우 작업을 놓칠 수 있습니다. 기본값: true.
useOplog	true false	true인 경우를 사용하면 DMS 태스크가 변경 스트림을 사용하는 대신 'oplog'에서 직접 읽을 수 있습니다. 기본값: false.

문서를 메타데이터 모드로 선택하면 다른 옵션을 사용할 수 있습니다.

대상 엔드포인트가 DocumentDB인 경우, 문서 모드에서 마이그레이션을 실행해야 합니다. 또한 소스 엔드포인트를 수정하고 `_id`를 별도의 열로 옵션을 선택합니다. 이는 소스 MongoDB 워크로드에 트랜잭션이 포함된 경우에 필수 사전 요구 사항입니다.

MongoDB용 소스 데이터 형식

MongoDB를 소스로 사용하는 데이터 마이그레이션은 대부분의 MongoDB 데이터 유형을 AWS DMS 지원합니다. 다음 표에서는 사용할 때 지원되는 MongoDB 소스 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑을 확인할 수 있습니다. MongoDB 데이터 형식에 대한 자세한 내용은 MongoDB 설명서의 [BSON Types\(BSON 형식\)](#)를 참조하세요.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

MongoDB 데이터 형식	AWS DMS 데이터 유형
부울	부울
바이너리	BLOB
Date	Date
타임스탬프	Date
정수	INT4
Long	INT8
배정밀도 실수	REAL8
문자열(UTF-8)	CLOB
배열	CLOB
OID	문자열
REGEX	CLOB

MongoDB 데이터 형식	AWS DMS 데이터 유형
코드	CLOB

Amazon DocumentDB(MongoDB 호환)를 소스로 사용 AWS DMS

AWS DMS 가 소스로 지원하는 Amazon DocumentDB(MongoDB 호환) 버전에 대한 자세한 내용은 [의 소스 AWS DMS](#) 섹션을 참조하세요.

Amazon DocumentDB를 소스로 사용하여 한 Amazon DocumentDB 클러스터에서 다른 Amazon DocumentDB 클러스터로 데이터를 마이그레이션할 수 있습니다. Amazon DocumentDB 클러스터에서 지원하는 다른 대상 엔드포인트 중 하나로 데이터를 마이그레이션할 수도 있습니다 AWS DMS.

Amazon DocumentDB를 처음 사용하는 경우, Amazon DocumentDB 데이터베이스의 다음 중요 개념을 알고 있어야 합니다.

- Amazon DocumentDB의 레코드는 필드와 값의 쌍으로 구성된 데이터 구조인 문서입니다. 필드의 값에는 다른 문서, 배열 및 문서의 배열이 포함될 수 있습니다. 문서는 대체로 관계형 데이터베이스 테이블의 행에 상응합니다.
- Amazon DocumentDB의 모음은 문서 그룹이며, 대략 관계형 데이터베이스 테이블에 해당합니다.
- Amazon DocumentDB의 데이터베이스는 모음 세트이며, 대략 관계형 데이터베이스의 스키마에 해당합니다.

AWS DMS 는 Amazon DocumentDB를 소스로 사용할 때 문서 모드와 테이블 모드라는 두 가지 마이그레이션 모드를 지원합니다. AWS DMS 콘솔에서 Amazon DocumentDB 소스 엔드포인트를 생성할 때 메타데이터 모드 옵션 또는 추가 연결 속성을 사용하여 마이그레이션 모드를 지정합니다. `nestingLevel`. 마이그레이션 모드 선택이 대상 데이터의 형식에 결과적으로 어떤 영향을 미치는지 아래에서 설명합니다.

문서 모드

문서 모드에서는 JSON 문서가 그대로 마이그레이션됩니다. 즉, 문서 데이터가 두 항목 중 하나로 통합됩니다. 관계형 데이터베이스를 대상으로 사용하는 경우, 데이터는 대상 테이블에서 이름이 `_doc`인 단일 열입니다. 비관계형 데이터베이스를 대상으로 사용하는 경우, 데이터는 단일 JSON 문서입니다. 문서 모드가 기본 모드이며, Amazon DocumentDB 대상으로 마이그레이션할 때 사용하는 것이 좋습니다.

예를 들어 myCollection이라는 Amazon DocumentDB 모음에 있는 다음 문서를 생각해 보세요.

```
db.myCollection.find()
{ "_id" : ObjectId("5a94815f40bd44d1b02bdfe0"), "a" : 1, "b" : 2, "c" : 3 }
{ "_id" : ObjectId("5a94815f40bd44d1b02bdfe1"), "a" : 4, "b" : 5, "c" : 6 }
```

데이터를 문서 모드를 사용하는 관계형 데이터베이스 테이블로 마이그레이션 하면, 데이터의 구조는 다음과 같습니다. 문서의 데이터 필드는 `_doc` 열로 통합됩니다.

oid_id	_doc
5a94815f40bd44d1b02bdfe0	{ "a" : 1, "b" : 2, "c" : 3 }
5a94815f40bd44d1b02bdfe1	{ "a" : 4, "b" : 5, "c" : 6 }

선택적으로 추가 연결 속성 `extractDocID`를 `true`로 설정하여 프라이머리 키 역할을 하는 `"_id"`라는 두 번째 열을 생성할 수 있습니다. 변경 데이터 캡처(CDC)를 사용하려는 경우, Amazon DocumentDB를 대상으로 사용하는 경우를 제외하고 이 파라미터를 `true`로 설정합니다.

[다중 문서 트랜잭션](#)을 생성하는 소스와 함께 CDC를 사용하는 경우 `ExtractDocId` 파라미터를 `true`로 설정해야 합니다. 이 파라미터를 활성화하지 않으면 다중 문서 트랜잭션이 발생하면 AWS DMS 작업이 실패합니다.

Note

소스 데이터베이스에 새 컬렉션을 추가하는 경우는 컬렉션에 대한 새 대상 테이블을 AWS DMS 생성하고 모든 문서를 복제합니다.

테이블 모드

테이블 모드에서 AWS DMS는 Amazon DocumentDB 문서의 각 최상위 수준 필드를 대상 테이블의 열로 변환합니다. 필드가 중첩된 경우 중첩된 값을 단일 열로 AWS DMS 확장합니다. 그런 AWS DMS 다음은 대상 테이블의 열 세트에 키 필드와 데이터 유형을 추가합니다.

각 Amazon DocumentDB 문서에 대해서는 각 키와 유형을 대상 테이블의 열 세트에 AWS DMS 추가합니다. 예를 들어 테이블 모드를 사용하면 이전 예제를 다음 테이블로 AWS DMS 마이그레이션합니다.

oid_id	a	b	c
5a94815f4 0bd44d1b02bdfe0	1	2	3
5a94815f4 0bd44d1b02bdfe1	4	5	6

중첩된 값은 점으로 분리된 키 이름을 포함하는 열에 평면화됩니다. 열의 이름은 마침표로 구분되고 평면화된 필드 이름의 연결을 사용하여 지정됩니다. 예를 들어,와 같은 중첩 값 필드가 있는 JSON 문서를 라는 열{"a" : {"b" : {"c": 1}}}로 AWS DMS 마이그레이션합니다. a.b.c.

대상 열을 생성하기 위해서는 지정된 수의 Amazon DocumentDB 문서를 AWS DMS 스캔하고 모든 필드 및 유형 세트를 생성합니다. AWS DMS 그런 다음이 세트를 사용하여 대상 테이블의 열을 생성합니다. 콘솔을 사용하여 Amazon DocumentDB 소스 엔드포인트를 생성하거나 수정하는 경우, 스캔할 문서 수를 지정할 수 있습니다. 기본값은 1,000 개 문서입니다. 를 사용하는 경우 추가 연결 속성을 사용할 AWS CLI수 있습니다docsToInvestigate.

테이블 모드에서는 다음과 같은 문서 및 컬렉션을 AWS DMS 관리합니다.

- 기존 모음에 문서를 추가할 때 문서가 복제됩니다. 대상에 없는 필드가 있으면 이 필드는 복제되지 않습니다.
- 문서를 업데이트하면 업데이트된 문서가 복제됩니다. 대상에 없는 필드가 있으면 이 필드는 복제되지 않습니다.
- 문서 삭제는 완벽하게 지원됩니다.
- 새 모음을 CDC 작업 중에 추가해도 대상에 테이블이 새로 생성되지 않습니다.
- 변경 데이터 캡처(CDC) 단계에서 AWS DMS 는 컬렉션 이름 변경을 지원하지 않습니다.

주제

- [Amazon DocumentDB를 소스로 사용하기 위한 권한 설정](#)
- [Amazon DocumentDB 클러스터를 위한 CDC 구성](#)
- [TLS를 사용하여 Amazon DocumentDB에 연결](#)
- [Amazon DocumentDB 소스 엔드포인트 생성](#)
- [Amazon DocumentDB 모음을 분할하고 병렬로 마이그레이션](#)
- [Amazon DocumentDB를 소스로 사용할 때 여러 데이터베이스 마이그레이션 AWS DMS](#)

- [Amazon DocumentDB를 소스로 사용할 때의 제한 사항 AWS DMS](#)
- [Amazon DocumentDB를 소스로 하는 엔드포인트 설정 사용](#)
- [Amazon DocumentDB의 소스 데이터 형식](#)

Amazon DocumentDB를 소스로 사용하기 위한 권한 설정

AWS DMS 마이그레이션에 Amazon DocumentDB 소스를 사용하는 경우 루트 권한이 있는 사용자 계정을 생성할 수 있습니다. 또는 마이그레이션할 데이터베이스에 대한 권한만 있는 사용자를 생성할 수도 있습니다.

다음 코드는 루트 계정으로 사용자를 생성합니다.

```
use admin
db.createUser(
  {
    user: "root",
    pwd: "password",
    roles: [ { role: "root", db: "admin" } ]
  })
```

Amazon DocumentDB 3.6의 경우, 다음 코드는 마이그레이션할 데이터베이스에 대한 최소 권한이 있는 사용자를 생성합니다.

```
use db_name
db.createUser(
  {
    user: "dms-user",
    pwd: "password",
    roles: [{ role: "read", db: "db_name" }]
  }
)
```

Amazon DocumentDB 4.0 이상의 경우 배포 전반의 변경 스트림을 AWS DMS 사용합니다. 여기서 다음 코드는 최소 권한을 가진 사용자를 생성합니다.

```
db.createUser(
  {
    user: "dms-user",
```

```
pwd: "password",
roles: [ { role: "readAnyDatabase", db: "admin" } ]
}))
```

Amazon DocumentDB 클러스터를 위한 CDC 구성

Amazon DocumentDB에서 지속적 복제 또는 CDC를 사용하려면 Amazon DocumentDB 클러스터의 변경 스트림에 대한 액세스 권한이 AWS DMS 필요합니다. 클러스터 모음 및 데이터베이스의 시간순 업데이트 이벤트 시퀀스에 대한 설명은 Amazon DocumentDB 개발자 안내서의 [변경 스트림 사용](#)을 참조하세요.

MongoDB 셸을 사용하여 Amazon DocumentDB 클러스터에 인증합니다. 그리고 나서 다음 명령을 실행하여 변경 스트림을 활성화합니다.

```
db.adminCommand({modifyChangeStreams: 1,
  database: "DB_NAME",
  collection: "",
  enable: true});
```

이 방식을 사용하면 데이터베이스의 모든 모음에 변경 스트림을 활성화할 수 있습니다. 변경 스트림이 활성화되면 기존 데이터를 마이그레이션하는 동시에 진행 중인 변경 사항을 복제하는 마이그레이션 작업을 생성할 수 있습니다. 대량 데이터가 로드된 후에도는 AWS DMS 계속해서 변경 사항을 캡처하고 적용합니다. 결국 소스 데이터베이스와 대상 데이터베이스가 동기화되어 마이그레이션의 가동 중지 시간이 최소화됩니다.

Note

AWS DMS 는 작업 로그(oplog)를 사용하여 지속적 복제 중에 변경 사항을 캡처합니다. Amazon DocumentDB가 oplog에서 레코드를 AWS DMS 읽기 전에 풀러시하면 작업이 실패합니다. 24시간 이상 변경 사항이 유지되도록 oplog의 크기를 조정하는 것이 좋습니다.

TLS를 사용하여 Amazon DocumentDB에 연결

기본적으로 새로 생성한 Amazon DocumentDB 클러스터는 Transport Layer Security(TLS)을 사용한 보안 연결만 허용합니다. TLS가 활성화되면 Amazon DocumentDB에 대한 모든 연결에 퍼블릭 키가 필요합니다.

AWS호스팅된 Amazon S3 버킷에서 파일을 다운로드하여 Amazon DocumentDB의 퍼블릭 키를 검색할 수 있습니다. `rds-combined-ca-bundle.pem` Amazon S3 이 파일을 다운로드하는 방법에 관한 자세한 내용은 Amazon DocumentDB 개발자 안내서의 [TLS를 사용한 연결 암호화](#)를 참조하세요.

`rds-combined-ca-bundle.pem` 파일을 다운로드한 후 포함된 퍼블릭 키를 가져올 수 있습니다 AWS DMS. 다음 단계에서 그 방법을 설명합니다.

AWS DMS 콘솔을 사용하여 퍼블릭 키를 가져오려면

1. 에 로그인 AWS Management Console 하고를 선택합니다 AWS DMS.
2. 탐색 창에서 인증서를 선택합니다.
3. 인증서 가져오기를 선택합니다. 새 CA 인증서 가져오기 페이지가 나타납니다.
4. 인증서 구성 섹션에서 다음 중 하나를 수행합니다.
 - 인증서 식별자에 해당 인증서의 고유 이름(예: `docdb-cert`)을 입력합니다.
 - 파일 선택을 선택하고 `rds-combined-ca-bundle.pem` 파일을 저장한 위치로 이동한 다음 파일을 선택합니다.
5. 새 CA 인증서 추가를 선택합니다.

AWS CLI 다음 예제에서는 명령을 사용하여 AWS DMS `import-certificate` 퍼블릭 키 `rds-combined-ca-bundle.pem` 파일을 가져옵니다.

```
aws dms import-certificate \  
  --certificate-identifier docdb-cert \  
  --certificate-pem file:///./rds-combined-ca-bundle.pem
```

Amazon DocumentDB 소스 엔드포인트 생성

콘솔 또는 AWS CLI를 사용하여 Amazon DocumentDB 소스 엔드포인트를 생성할 수 있습니다. 콘솔에서는 다음 절차를 사용하세요.

AWS DMS 콘솔을 사용하여 Amazon DocumentDB 소스 엔드포인트를 구성하려면

1. 에 로그인 AWS Management Console 하고를 선택합니다 AWS DMS.
2. 탐색 창에서 엔드포인트를 선택하고 엔드포인트 생성을 선택합니다.
3. 엔드포인트 식별자에 쉽게 식별할 수 있는 이름(예: `docdb-source`)을 제공합니다.

4. 소스 엔진에서 Amazon DocumentDB(MongoDB 호환)를 선택합니다.
5. 서버 이름에 Amazon DocumentDB 데이터베이스 엔드포인트가 있는 서버의 이름을 입력합니다. 예를 들어 Amazon EC2 인스턴스의 퍼블릭 DNS 이름(예: democluster.cluster-cj6q8nxfefi.us-east-2.docdb.amazonaws.com)을 입력할 수 있습니다.
6. 포트에 27017을 입력합니다.
7. SSL 모드에서 verify-full을 선택합니다. Amazon DocumentDB 클러스터에서 SSL을 비활성화한 경우, 이 단계를 건너뛰어도 됩니다.
8. CA 인증서에서 Amazon DocumentDB 인증서인 rds-combined-ca-bundle.pem을 선택합니다. 이 인증서를 추가하는 방법에 대한 지침은 [TLS를 사용하여 Amazon DocumentDB에 연결](#) 섹션을 참조하세요.
9. 데이터베이스 이름에 마이그레이션할 데이터베이스의 이름을 입력합니다.

CLI에서는 다음 절차를 사용하세요.

를 사용하여 Amazon DocumentDB 소스 엔드포인트를 구성하려면 AWS CLI

- 다음 AWS DMS create-endpoint 명령을 실행하여 Amazon DocumentDB 소스 엔드포인트를 구성하고 자리 표시자를 고유한 값으로 바꿉니다.

```
aws dms create-endpoint \
    --endpoint-identifier a_memorable_name \
    --endpoint-type source \
    --engine-name docdb \
    --username value \
    --password value \
    --server-name servername_where_database_endpoint_resides \
    --port 27017 \
    --database-name name_of_endpoint_database
```

Amazon DocumentDB 모음을 분할하고 병렬로 마이그레이션

마이그레이션 작업의 성능을 개선하기 위해 Amazon DocumentDB 소스 엔드포인트는 테이블 매핑에서 병렬 전체 로드 기능 옵션 두 가지를 지원합니다. 다시 말해 JSON 설정에서 병렬 전체 로드를 위해 테이블 매핑에서 자동 분할 옵션이나 범위 분할 옵션을 사용하여 모음을 병렬로 마이그레이션할 수 있습니다. 자동 분할 옵션을 사용하면 각 스레드에서 마이그레이션을 위해 소스를 자동으로 분할 AWS DMS 하는 기준을 지정할 수 있습니다. 범위 분할 옵션을 사용하면 DMS AWS DMS 가 각 스레드에서

마이그레이션할 각 세그먼트의 특정 범위를 지정할 수 있습니다. 이러한 설정에 대한 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 섹션을 참조하세요.

자동 분할 범위를 사용하여 Amazon DocumentDB 데이터베이스를 병렬로 마이그레이션

AWS DMS 가 각 스레드의 데이터를 자동으로 파티셔닝(분할)하는 기준, 특히 스레드당 마이그레이션할 문서 수를 지정하여 문서를 병렬로 마이그레이션할 수 있습니다. AWS DMS 는 이 방식을 사용하여 스레드당 성능을 극대화하는 세그먼트 경계 최적화를 시도합니다.

테이블 매핑에서 다음 테이블 설정 옵션을 사용하여 분할 기준을 지정할 수 있습니다.

테이블 설정 옵션	설명
"type"	(필수) 소스로 사용하는 Amazon DocumentDB의 경우, "partitions-auto" 로 설정합니다.
"number-of-partitions"	(선택 사항) 마이그레이션에 사용되는 총 파티션 (세그먼트) 수. 기본값은 16입니다.
"collection-count-from-meta data"	(선택 사항) 로 설정하면 예상 컬렉션 수를 true AWS DMS 사용하여 파티션 수를 결정합니다. 로 설정하면 실제 컬렉션 수를 false AWS DMS 사용합니다. 기본값은 true입니다.
"max-records-skip-per-page"	(선택 사항) 각 파티션의 경계를 결정할 때 한 번에 건너뛴 레코드 수입니다.는 페이지가 매겨진 건너뛰기 접근 방식을 AWS DMS 사용하여 파티션의 최소 경계를 결정합니다. 기본값은 10,000입니다. 상대적으로 큰 값을 설정하면 커서 시간이 초과되고 작업이 실패할 수 있습니다. 상대적으로 작은 값을 설정하면 페이지당 작업 수가 늘어나고 전체 로드 속도가 느려집니다.
"batch-size"	(선택 사항) 배치 하나에서 반환되는 문서 수를 제한합니다. 배치마다 서버를 왕복해야 합니다. 배치 크기가 0인 경우, 커서는 서버에서 정의한 최대 배치 크기를 사용합니다. 기본값은 0입니다.

다음 예제는 자동 분할을 위한 테이블 매핑을 보여 줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "admin",
        "table-name": "departments"
      },
      "rule-action": "include",
      "filters": []
    },
    {
      "rule-type": "table-settings",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "admin",
        "table-name": "departments"
      },
      "parallel-load": {
        "type": "partitions-auto",
        "number-of-partitions": 5,
        "collection-count-from-metadata": "true",
        "max-records-skip-per-page": 1000000,
        "batch-size": 50000
      }
    }
  ]
}
```

자동 분할에는 다음과 같은 제한이 있습니다. 각 세그먼트의 마이그레이션은 모음 개수와 모음의 최소 `_id`를 개별적으로 가져옵니다. 그런 다음 페이지 매김 건너뛰기를 사용하여 해당 세그먼트의 최소 경계를 계산합니다. 따라서 모음의 모든 세그먼트 경계를 계산할 때까지 각 모음의 최소 `_id` 값이 일정하게 유지되어야 합니다. 세그먼트 경계를 계산하는 동안 모음의 최소 `_id` 값을 변경하면 데이터가 손실되거나 행 중복 오류가 발생할 수 있습니다.

특정 세그먼트 범위를 사용하여 Amazon DocumentDB 데이터베이스를 병렬로 마이그레이션

다음 예제에 나온 Amazon DocumentDB 모음에는 항목 7개와 프라이머리 키 `_id`가 있습니다.

Key	Value	Type
▼ (1) ObjectId("5f805c74873173399a278d78")	{ 3 fields }	Object
_id	ObjectId("5f805c74873173399a278d78")	ObjectId
num	1	Int32
name	a	String
▼ (2) ObjectId("5f805c97873173399a278d79")	{ 3 fields }	Object
_id	ObjectId("5f805c97873173399a278d79")	ObjectId
num	2	Int32
name	b	String
▼ (3) ObjectId("5f805cb0873173399a278d7a")	{ 3 fields }	Object
_id	ObjectId("5f805cb0873173399a278d7a")	ObjectId
num	3	Int32
name	c	String
▼ (4) ObjectId("5f805cbb873173399a278d7b")	{ 3 fields }	Object
_id	ObjectId("5f805cbb873173399a278d7b")	ObjectId
num	4	Int32
name	d	String
▼ (5) ObjectId("5f805cc5873173399a278d7c")	{ 3 fields }	Object
_id	ObjectId("5f805cc5873173399a278d7c")	ObjectId
num	5	Int32
name	e	String
▼ (6) ObjectId("5f805cd0873173399a278d7d")	{ 3 fields }	Object
_id	ObjectId("5f805cd0873173399a278d7d")	ObjectId
num	6	Int32
name	f	String
▼ (7) ObjectId("5f805cdd873173399a278d7e")	{ 3 fields }	Object
_id	ObjectId("5f805cdd873173399a278d7e")	ObjectId
num	7	Int32
name	g	String

모음을 세 개의 세그먼트로 분할하고 병렬로 마이그레이션하려면 다음 JSON 예제와 같이 마이그레이션 작업에 테이블 매핑 규칙을 추가하면 됩니다.

```
{ // Task table mappings:
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "testdatabase",
        "table-name": "testtable"
      },
      "rule-action": "include"
    }
  ]
}
```



```

    }, // "selection" : "rule-type"
    {
      "rule-type": "table-settings",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "testdatabase",
        "table-name": "testtable"
      },
      "parallel-load": {
        "type": "ranges",
        "columns": [
          "_id",
          "num"
        ],
        "boundaries": [
          // First segment selects documents with _id less-than-or-equal-to
          5f805c97873173399a278d79
          // and num less-than-or-equal-to 2.
          [
            "5f805c97873173399a278d79",
            "2"
          ],
          // Second segment selects documents with _id > 5f805c97873173399a278d79 and
          // _id less-than-or-equal-to 5f805cc5873173399a278d7c and
          // num > 2 and num less-than-or-equal-to 5.
          [
            "5f805cc5873173399a278d7c",
            "5"
          ]
          // Third segment is implied and selects documents with _id >
          5f805cc5873173399a278d7c.
        ] // : "boundaries"
      } // : "parallel-load"
    } // "table-settings" : "rule-type"
  ] // : "rules"
} // : Task table mappings

```

이 테이블 매핑 정의는 소스 모음을 세 개의 세그먼트로 분할하고 병렬로 마이그레이션합니다. 분할 경계는 다음과 같습니다.

Data with `_id` less-than-or-equal-to "5f805c97873173399a278d79" and num less-than-or-equal-to 2 (2 records)

Data with `_id` less-than-or-equal-to "5f805cc5873173399a278d7c" and num less-than-or-equal-to 5 and not in (`_id` less-than-or-equal-to "5f805c97873173399a278d79" and num less-than-or-equal-to 2) (3 records)

Data not in (`_id` less-than-or-equal-to "5f805cc5873173399a278d7c" and num less-than-or-equal-to 5) (2 records)

마이그레이션 작업이 완료되면 다음 예제와 같이 작업 로그에서 테이블이 병렬로 로드된 것을 확인할 수 있습니다. 소스 테이블에서 각 세그먼트를 언로드하는 데 사용된 Amazon DocumentDB find 절도 확인할 수 있습니다.

```
[TASK_MANAGER    ] I: Start loading segment #1 of 3 of table
'testdatabase'. 'testtable' (Id = 1) by subtask 1. Start load timestamp
0005B191D638FE86 (replicationtask_util.c:752)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is initialized.
(mongodb_unload.c:157)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is: { "_id" :
{ "$lte" : { "$oid" : "5f805c97873173399a278d79" } }, "num" : { "$lte" :
{ "$numberInt" : "2" } } } (mongodb_unload.c:328)

[SOURCE_UNLOAD   ] I: Unload finished for segment #1 of segmented table
'testdatabase'. 'testtable' (Id = 1). 2 rows sent.

[TASK_MANAGER    ] I: Start loading segment #1 of 3 of table
'testdatabase'. 'testtable' (Id = 1) by subtask 1. Start load timestamp
0005B191D638FE86 (replicationtask_util.c:752)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is initialized.
(mongodb_unload.c:157)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is: { "_id" : { "$lte" :
{ "$oid" : "5f805c97873173399a278d79" } }, "num" : { "$lte" : { "$numberInt" :
"2" } } } (mongodb_unload.c:328)

[SOURCE_UNLOAD   ] I: Unload finished for segment #1 of segmented table
'testdatabase'. 'testtable' (Id = 1). 2 rows sent.
```

```
[TARGET_LOAD      ] I: Load finished for segment #1 of segmented table
'testdatabase'.'testtable' (Id = 1). 1 rows received. 0 rows skipped. Volume
transferred 480.
```

```
[TASK_MANAGER     ] I: Load finished for segment #1 of table
'testdatabase'.'testtable' (Id = 1) by subtask 1. 2 records transferred.
```

현재는 다음 Amazon DocumentDB 데이터 형식을 세그먼트 키 열로 AWS DMS 지원합니다.

- 배정밀도 실수
- 문자열
- ObjectId
- 32비트 정수
- 64비트 정수

Amazon DocumentDB를 소스로 사용할 때 여러 데이터베이스 마이그레이션 AWS DMS

AWS DMS 버전 3.4.5 이상에서는 Amazon DocumentDB 버전 4.0 이상에서만 단일 작업으로 여러 데이터베이스를 마이그레이션할 수 있습니다. 여러 데이터베이스를 마이그레이션하려면 다음을 수행하세요.

1. Amazon DocumentDB 소스 엔드포인트를 생성할 때:

- AWS Management Console 의 엔드포인트 생성 페이지의 엔드포인트 구성에서 데이터베이스 이름을 비워 AWS DMS입니다.
- AWS Command Line Interface (AWS CLI)에서 CreateEndpoint 작업에 지정한 DocumentDBSettings의 DatabaseName 파라미터에 빈 문자열 값을 할당합니다.

2. 이 Amazon DocumentDB 소스 엔드포인트에서 마이그레이션하려는 각 데이터베이스마다 콘솔의 안내 입력을 사용하거나 JSON에서 직접 각 데이터베이스의 이름을 작업 테이블 매핑의 스키마 이름으로 지정합니다. 안내 입력에 대한 자세한 내용은 [콘솔에서 테이블 선택 및 변환 규칙 지정](#)의 설명을 참조하세요. JSON에 대한 자세한 내용은 [선택 규칙 및 작업](#) 섹션을 참조하세요.

예를 들면 다음과 같이 JSON을 지정하여 Amazon DocumentDB 데이터베이스 3개를 마이그레이션할 수 있습니다.

Example스키마에서 모든 테이블 마이그레이션

다음 JSON은 소스 엔드포인트에 있는 Customers, Orders, Suppliers 데이터베이스의 모든 테이블을 대상 엔드포인트로 마이그레이션합니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "Customers",
        "table-name": "%"
      },
      "object-locator": {
        "schema-name": "Orders",
        "table-name": "%"
      },
      "object-locator": {
        "schema-name": "Inventory",
        "table-name": "%"
      },
      "rule-action": "include"
    }
  ]
}
```

Amazon DocumentDB를 소스로 사용할 때의 제한 사항 AWS DMS

다음은 Amazon DocumentDB를 소스로 사용할 때의 제한 사항입니다 AWS DMS.

- `_id` 옵션이 별도의 열로 설정된 경우 ID 문자열이 200자를 초과할 수 없습니다.
- 객체 ID와 배열 형식 키는 테이블 모드에서 oid 및 array 접두사가 붙은 열로 변환됩니다.

내부적으로 이러한 열은 접두사 이름으로 참조됩니다. 이러한 열을 참조 AWS DMS 하는에서 변환 규칙을 사용하는 경우 접두사가 붙은 열을 지정해야 합니다. 예를 들면 `${_id}`가 아니라 `${oid__id}`, 또는 `${_addresses}`가 아니라 `${array__addresses}`를 지정합니다.

- 모음 이름과 키 이름에는 달러 기호(\$)를 포함할 수 없습니다.
- 테이블 모드와 문서 모드에는 앞서 설명한 제한이 있습니다.
- 자동 분할을 사용한 병렬 마이그레이션에는 앞에서 설명한 제한 사항이 있습니다.

- Amazon DocumentDB(MongoDB 호환) 소스는 변경 데이터 캡처(CDC) 시작 위치로 특정 타임스탬프를 사용하는 것을 지원하지 않습니다. 지속적 복제 작업은 타임스탬프에 상관없이 변경 내용을 캡처하기 시작합니다.
- AWS DMS 는 3.5.2보다 낮은 AWS DMS 버전의 경우 중첩 수준이 97보다 큰 문서를 지원하지 않습니다.
- DocumentDB에는 소스 필터가 지원되지 않습니다.
- AWS DMS 는 탄력적 클러스터 모드에서 DocumentDB에 대한 CDC(변경 데이터 캡처) 복제를 소스로 지원하지 않습니다.

Amazon DocumentDB를 소스로 하는 엔드포인트 설정 사용

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Amazon DocumentDB 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 `--doc-db-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 [AWS CLI](#) 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

Amazon DocumentDB를 소스로 하여 사용할 수 있는 엔드포인트 설정이 다음 표에 나와 있습니다.

속성 이름	유효값	기본값과 설명
NestingLevel	"none" "one"	"none" - 문서 모드를 사용하려면 "none"을 지정합니다. 테이블 모드를 사용하려면 "one"을 지정합니다.
ExtractDocID	true false	false - 이 속성은 NestingLevel 이 "none"으로 설정된 경우에 사용합니다. 다중 문서 트랜잭션 을 생성하는 소스와 함께 CDC를 사용하는 경우 ExtractDocId 파라미터를 로 설정해야 합니다true. 이 파라미터를 활성화하지 않으면 다중 문서 트랜잭션이 발생하면 AWS DMS 작업이 실패합니다.
DocsToInvestigate	0보다 큰 양의 정수입니다.	1000 - 이 속성은 NestingLevel 이 "one"으로 설정된 경우에 사용합니다.
ReplicateShardCollections	true false	true인 경우는 데이터를 샤드 컬렉션에 AWS DMS 복제합니다. 대상 엔드포인트가 DocumentDB 탄력적 클러스터인 경우에만이 설정을 AWS DMS 사용합니다.

속성 이름	유효값	기본값과 설명
		이 설정이 true인 경우 다음 사항에 유의하세요. - TargetTablePrepMode 를 nothing으로 설정해야 합니다. - AWS DMS 는 useUpdateLookup 로 자동 설정됩니다 false.

Amazon DocumentDB의 소스 데이터 형식

다음 표에서 AWS DMS사용 시 지원되는 Amazon DocumentDB 소스 데이터 형식을 찾을 수 있습니다. 이 표의 AWS DMS 데이터 형식에서 기본 매핑을 찾을 수도 있습니다. 데이터 형식에 대한 자세한 내용은 MongoDB 설명서의 [BSON Types](#)를 참조하세요.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

Amazon DocumentDB 데이터 형식	AWS DMS 데이터 형식
부울	부울
바이너리	BLOB
Date	Date
타임스탬프	Date
정수	INT4
Long	INT8
배정밀도 실수	REAL8
문자열(UTF-8)	CLOB
배열	CLOB

Amazon DocumentDB 데이터 형식	AWS DMS 데이터 형식
OID	문자열

Amazon S3를의 소스로 사용 AWS DMS

를 사용하여 Amazon S3 버킷에서 데이터를 마이그레이션할 수 있습니다 AWS DMS. 이렇게 하려면 하나 이상의 데이터 파일을 포함하는 Amazon S3 버킷에 액세스합니다. 해당 S3 버킷에서, 데이터와 데이터의 데이터베이스 테이블 간 매핑을 설명하는 JSON 파일을 이러한 파일 안에 포함합니다.

전체 로드 시작되기 전에 Amazon S3 버킷에 소스 데이터 파일이 있어야 합니다. bucketName 파라미터를 사용하여 버킷 이름을 지정합니다.

소스 데이터 파일은 다음 형식일 수 있습니다.

- 쉼표로 구분된 값(.csv)
- Parquet(DMS 버전 3.5.3 이상). Parquet 형식 파일에 대한 자세한 내용은 [Amazon S3의 Parquet 형식 파일의 소스로 사용 AWS DMS](#) 섹션을 참조하세요.

쉼표로 구분된 값(.csv) 형식의 소스 데이터 파일의 경우 다음 이름 지정 규칙을 사용하여 이름을 지정합니다. 이 규칙에서 *schemaName*은 원본 스키마이고 *tableName*은 해당 스키마 내 테이블 이름입니다.

```
/schemaName/tableName/LOAD001.csv
/schemaName/tableName/LOAD002.csv
/schemaName/tableName/LOAD003.csv
...
```

예를 들면 다음 Amazon S3 경로에서 amzn-s3-demo-bucket에 데이터 파일이 있다고 가정합니다.

```
s3://amzn-s3-demo-bucket/hr/employee
```

로드 시 소스 스키마 이름이 이고 소스 테이블 이름이 hr라고 AWS DMS 가정합니다employee.

bucketName (필수) 외에도 선택적으로 bucketFolder 파라미터를 제공하여가 Amazon S3 버킷에서 데이터 파일을 찾을 위치를 지정할 수 AWS DMS 있습니다. 이전 예제를 계속 진행하면서 bucketFolder로 설정하면가 다음 경로에서 데이터 파일을 sourcedata AWS DMS 읽습니다.

```
s3://amzn-s3-demo-bucket/sourcedata/hr/employee
```

추가 연결 속성을 사용하여 열 구분 기호, 행 구분 기호, null 값 지표 등 여러 파라미터를 지정할 수 있습니다. 자세한 내용은 [Amazon S3를 소스로 사용하기 위한 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

다음과 같이 ExpectedBucketOwner Amazon S3 엔드포인트 설정을 사용하여 버킷 소유자를 지정하고 스나이핑을 방지할 수 있습니다. 그런 다음, 연결을 테스트하거나 마이그레이션을 수행하도록 요청하면 S3는 지정된 파라미터와 비교하여 버킷 소유자의 계정 ID를 확인합니다.

```
--s3-settings='{ "ExpectedBucketOwner": "AWS_Account_ID" }'
```

주제

- [Amazon S3용 외부 테이블의 소스로 정의 AWS DMS](#)
- [Amazon S3와 함께 CDC를의 소스로 사용 AWS DMS](#)
- [Amazon S3를의 소스로 사용할 때의 사전 조건 AWS DMS](#)
- [Amazon S3를의 소스로 사용할 때의 제한 사항 AWS DMS](#)
- [Amazon S3를 소스로 사용하기 위한 엔드포인트 설정 AWS DMS](#)
- [Amazon S3의 소스 데이터 형식](#)
- [Amazon S3의 Parquet 형식 파일의 소스로 사용 AWS DMS](#)

Amazon S3용 외부 테이블의 소스로 정의 AWS DMS

데이터 파일 외에, 외부 테이블 정의도 제공해야 합니다. 외부 테이블 정의는 Amazon S3의 데이터를 해석하는 방법을 설명하는 JSON 문서 AWS DMS입니다. 이 문서의 최대 크기는 2 MB입니다. AWS DMS Management Console을 사용하여 소스 엔드포인트를 생성하는 경우 테이블 매핑 상자에 JSON을 직접 입력할 수 있습니다. AWS Command Line Interface (AWS CLI) 또는 AWS DMS API를 사용하여 마이그레이션을 수행하는 경우 JSON 파일을 생성하여 외부 테이블 정의를 지정할 수 있습니다.

다음은 포함하는 데이터 파일이 있다고 가정해 보겠습니다.

```
101,Smith,Bob,2014-06-04,New York
102,Smith,Bob,2015-10-08,Los Angeles
103,Smith,Bob,2017-03-13,Dallas
```



```
104,Smith,Bob,2017-03-13,Dallas
```

다음은 이 데이터에 대한 외부 테이블 정의 예제입니다.

```
{
  "TableCount": "1",
  "Tables": [
    {
      "TableName": "employee",
      "TablePath": "hr/employee/",
      "TableOwner": "hr",
      "TableColumns": [
        {
          "ColumnName": "Id",
          "ColumnType": "INT8",
          "ColumnNullable": "false",
          "ColumnIsPk": "true"
        },
        {
          "ColumnName": "LastName",
          "ColumnType": "STRING",
          "ColumnLength": "20"
        },
        {
          "ColumnName": "FirstName",
          "ColumnType": "STRING",
          "ColumnLength": "30"
        },
        {
          "ColumnName": "HireDate",
          "ColumnType": "DATETIME"
        },
        {
          "ColumnName": "OfficeLocation",
          "ColumnType": "STRING",
          "ColumnLength": "20"
        }
      ],
      "TableColumnsTotal": "5"
    }
  ]
}
```

이 JSON 문서의 요소는 다음과 같습니다.

TableCount - 소스 테이블 수. 이 예제에는 테이블이 하나뿐입니다.

Tables - 소스 테이블당 하나의 JSON 맵으로 구성된 배열. 이 예제에는 맵이 하나뿐입니다. 각 맵은 다음 요소로 구성됩니다.

- **TableName** - 소스 테이블의 이름.
- **TablePath** - AWS DMS 가 전체 데이터 로드 파일을 찾을 수 있는 Amazon S3 버킷의 경로. **bucketFolder** 값이 지정된 경우, 해당 값이 경로에 추가됩니다.
- **TableOwner** - 이 테이블의 스키마 이름.
- **TableColumns** - 하나 이상의 맵으로 이루어진 배열. 각 맵은 소스 테이블의 열을 설명합니다.
 - **ColumnName** - 소스 테이블의 열 이름.
 - **ColumnType** - 열의 데이터 형식. 유효한 데이터 형식은 [Amazon S3의 소스 데이터 형식](#) 섹션을 참조하세요.
 - **ColumnLength** - 이 열의 바이트 수. S3 소스가 전체 LOB 모드를 지원하지 않으므로 최대 열 길이는 2,147,483,647바이트(2,047메가바이트)로 제한됩니다. **ColumnLength**가 유효한 데이터 형식은 다음과 같습니다.
 - BYTE
 - STRING
 - **ColumnNullable** - 이 열에 NULL 값을 포함할 수 있는 경우 true인 부울 값(기본값=false).
 - **ColumnIsPk** - 이 열이 프라이머리 키의 일부인 경우 true인 부울 값(기본값=false).
 - **ColumnDateFormat** - DATE, TIME, DATETIME 형식이 있는 열의 입력 날짜 형식으로, 데이터 문자열을 날짜 객체로 파싱하는 데 사용됩니다. 가능한 값은 다음과 같습니다.

```
- YYYY-MM-dd HH:mm:ss
- YYYY-MM-dd HH:mm:ss.F
- YYYY/MM/dd HH:mm:ss
- YYYY/MM/dd HH:mm:ss.F
- MM/dd/YYYY HH:mm:ss
- MM/dd/YYYY HH:mm:ss.F
- YYYYMMdd HH:mm:ss
- YYYYMMdd HH:mm:ss.F
```

- **TableColumnsTotal** - 총 열 수. 이 숫자는 **TableColumns** 배열의 요소 수와 일치해야 합니다.

달리 지정하지 않으면 **ColumnLength**가 0이라고 AWS DMS 가정합니다.

Note

지원되는 버전에서는 AWS DMS S3 소스 데이터에 선택적 작업 열이 열 TableName 값 앞에 첫 번째 열로 포함될 수도 있습니다. 이 작업 열은 전체 로드 중에 데이터를 S3 대상 엔드포인트로 마이그레이션하는 데 사용되는 작업(INSERT)을 식별합니다.

있는 경우 이 열의 값은 INSERT 작업 키워드(I)의 초기 문자열입니다. 지정된 경우 일반적으로 이 열은 이전 마이그레이션 중에 DMS에서 S3 대상으로 생성한 S3 소스입니다.

3.4.2 이전 DMS 버전에서 이 열은 이전 DMS 전체 로드에서 생성한 S3 소스 데이터에 없었습니다. S3 대상 데이터에 열을 추가하면 모든 행이 데이터의 전체 로드 중에 기록되는지 또는 CDC 로드 중에 기록되는지와 상관없이 S3 대상에 기록된 모든 행의 형식을 일관적으로 유지할 수 있습니다. S3 대상 데이터 서식 지정에 대한 자세한 내용은 [마이그레이션된 S3 데이터에 소스 DB 작업 표시](#) 섹션을 참조하세요.

NUMERIC 유형 열의 경우, 정밀도와 배율을 지정합니다. 정밀도는 숫자의 총 자릿수이며, 배율은 소수점 오른쪽의 자릿수입니다. 다음과 같이, 이를 위해 ColumnPrecision 및 ColumnScale 요소를 사용할 수 있습니다.

```
...
{
    "ColumnName": "HourlyRate",
    "ColumnType": "NUMERIC",
    "ColumnPrecision": "5"
    "ColumnScale": "2"
}
...
```

소수점 이하 초가 포함된 데이터가 있는 DATETIME 유형 열의 경우, 스케일을 지정합니다. 스케일은 소수점 이하 초의 자릿수이며, 범위는 0에서 9까지입니다. 다음과 같이 이를 위해 ColumnScale 요소를 사용할 수 있습니다.

```
...
{
    "ColumnName": "HireDate",
    "ColumnType": "DATETIME",
    "ColumnScale": "3"
}
...
```

달리 지정하지 않으면 ColumnScale가 0이고 소수 초를 잘라낸 AWS DMS 라고 가정합니다.

Amazon S3와 함께 CDC를의 소스로 사용 AWS DMS

가 전체 데이터 로드를 AWS DMS 수행한 후 선택적으로 데이터 변경 사항을 대상 엔드포인트에 복제할 수 있습니다. 이렇게 하려면 변경 데이터 캡처 파일(CDC 파일)을 Amazon S3 버킷에 업로드합니다. 는 업로드할 때 이러한 CDC 파일을 AWS DMS 읽은 다음 대상 엔드포인트에 변경 사항을 적용합니다.

CDC 파일은 다음과 같이 지정됩니다.

```
CDC00001.csv
CDC00002.csv
CDC00003.csv
...
```

Note

변경 데이터 폴더에서 CDC 파일을 복제하려면 어휘(순차적) 순으로 CDC 파일을 성공적으로 업로드합니다. 예를 들어 CDC00002.csv 파일을 CDC00003.csv 파일 앞에 업로드합니다. 그렇지 않으면 CDC00002.csv를 건너뛰고 CDC00003.csv 이후에 로드해도 CDC00002.csv가 복제되지 않습니다. CDC00003.csv 이후에 CDC00004.csv를 로드하면 CDC00004.csv가 성공적으로 복제됩니다.

에서 파일을 찾을 AWS DMS 수 있는 위치를 표시하려면 cdcPath 파라미터를 지정합니다. 이전 예제를 계속하여, cdcPath를 *changedata*로 설정하는 경우 AWS DMS 는 다음 경로에서 CDC 파일을 읽습니다.

```
s3://amzn-s3-demo-bucket/changedata
```

cdcPath를 *changedata*로, bucketFolder를 *myFolder*로 설정하면 AWS DMS 는 다음 경로에서 CDC 파일을 읽습니다.

```
s3://amzn-s3-demo-bucket/myFolder/changedata
```

다음과 같이 CDC 파일의 레코드에는 형식 지정되어 있습니다.

- 작업 - INSERT 또는 I, UPDATE 또는 U, DELETE 또는 D와 같은 수행할 변경 작업. 이러한 키워드 및 문자 값은 대소문자를 구분합니다.

Note

지원되는 AWS DMS 버전에서는 두 가지 방법으로 각 로드 레코드에 대해 수행할 작업을 식별할 AWS DMS 수 있습니다. AWS DMS 는 레코드의 키워드 값(예: INSERT) 또는 키워드 초기 문자(예:)에서 작업을 수행할 수 있습니다. 이전 버전에서는 전체 키워드 값에서만 로드 작업을 AWS DMS 인식했습니다.

이전 버전에서는 CDC 데이터를 로깅하기 위해 AWS DMS 전체 키워드 값이 작성되었습니다. 또한 이전 버전에서는 키워드 첫 문자만 사용하여 작업 값을 S3 대상에 기록했습니다. 두 형식을 모두 인식하면 작업 열이 S3 소스 데이터를 생성하기 위해 작성되는 방식에 관계없이 작업을 AWS DMS 처리할 수 있습니다. 이 접근 방식은 이후 마이그레이션에 대한 소스로 S3 대상 데이터 사용을 지원합니다. 이 접근 방식을 사용하면 이후 S3 소스의 작업 열에 나타나는 키워드 초기 값의 형식을 변경할 필요가 없습니다.

- 테이블 이름 - 소스 테이블의 이름.
- 스키마 이름 - 소스 스키마의 이름.
- 데이터 - 변경할 데이터를 나타내는 하나 이상의 열.

다음은 employee 이름의 테이블에 대한 예제 CDC 파일입니다.

```
INSERT,employee,hr,101,Smith,Bob,2014-06-04,New York
UPDATE,employee,hr,101,Smith,Bob,2015-10-08,Los Angeles
UPDATE,employee,hr,101,Smith,Bob,2017-03-13,Dallas
DELETE,employee,hr,101,Smith,Bob,2017-03-13,Dallas
```

Amazon S3를 소스로 사용할 때의 사전 조건 AWS DMS

Amazon S3를 소스로 사용하려면 AWS DMS 소스 S3 버킷이 데이터를 마이그레이션하는 DMS 복제 인스턴스와 동일한 AWS 리전에 있어야 합니다. 또한, 마이그레이션에 사용하는 AWS 계정에 소스 버킷에 대한 읽기 액세스 권한이 있어야 합니다. AWS DMS 버전 3.4.7 이상의 경우 DMS는 VPC 엔드포인트 또는 퍼블릭 경로를 통해 소스 버킷에 액세스해야 합니다. VPC 엔드포인트에 대한 자세한 내용은 [에 대한 VPC 엔드포인트 구성 AWS DMS](#) 섹션을 참조하세요.

마이그레이션 작업을 생성하는 데 사용되는 사용자 계정에 할당된 AWS Identity and Access Management (IAM) 역할에는 다음과 같은 권한 집합이 있어야 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/*/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket*"
      ]
    }
  ]
}
```

Amazon S3 버킷에서 버전 관리가 활성화된 경우 마이그레이션 작업을 생성하는 데 사용되는 사용자 계정에 할당된 AWS Identity and Access Management (IAM) 역할에는 다음과 같은 권한 집합이 있어야 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
```

```

        "s3:GetObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket*/**"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket*"
      ]
    }
  ]
}

```

Amazon S3를의 소스로 사용할 때의 제한 사항 AWS DMS

Amazon S3를 소스로 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- S3의 버전 관리를 활성화하지 마세요. S3 버전 관리가 필요한 경우, 수명 주기 정책을 사용하여 이전 버전을 적극적으로 삭제하세요. 그렇지 않으면 S3 list-object 직접 호출 시간 초과로 인해 엔드 포인트 테스트 연결 실패가 발생할 수 있습니다. S3 버킷의 수명 주기 정책을 생성하려면 [스토리지 수명 주기 관리](#)를 참조하세요. S3 객체의 버전을 삭제하려면 [버전 관리 활성화 버킷에서 객체 버전 삭제](#)를 참조하세요.
- VPC 지원(게이트웨이 VPC) S3 버킷은 버전 3.4.7 이상 버전에서 지원됩니다.
- MySQL은 time 데이터 유형을 string으로 변환합니다. MySQL에서 time 데이터 유형 값을 보려면 대상 테이블의 열을 string으로 정의하고 작업의 대상 테이블 준비 모드 설정을 잘라내기로 설정합니다.
- AWS DMS 는 BYTE 및 BYTE 데이터 형식의 데이터에 대해 내부적으로 BYTES 데이터 형식을 사용합니다.
- S3 소스 엔드포인트는 DMS 테이블 다시 로드 기능을 지원하지 않습니다.
- AWS DMS 는 Amazon S3를 소스로 사용하는 전체 LOB 모드를 지원하지 않습니다.

소스로 사용하는 Amazon S3에서 Parquet 형식 파일을 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- MMYYYYDD 또는 DDMMYYYY의 날짜는 S3 Parquet 소스 날짜 파티셔닝 기능에서 지원되지 않습니다.

Amazon S3를 소스로 사용하기 위한 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Amazon S3 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 `--s3-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 [AWS CLI](#) 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

Note

AWS DMS 는 SSL 모드 또는 인증서를 지정할 필요 없이 Amazon S3 엔드포인트에 대한 보안 연결을 기본값으로 설정합니다.

Amazon S3를 소스로 하여 사용할 수 있는 엔드포인트 설정이 다음 표에 나와 있습니다.

옵션	설명
BucketFolder	(선택 사항) S3 버킷의 폴더 이름. 이 속성이 지정된 경우, 소스 데이터 파일과 CDC 파일을 경로 <code>s3://<i>amzn-s3-demo-bucket</i> /<i>bucketFolder</i> /<i>schemaName</i> /<i>tableName</i></code> /과 <code>s3://<i>amzn-s3-demo-bucket</i> /<i>bucketFolder</i></code> /에서 각각 읽습니다. 이 속성이 지정되지 않은 경우, 사용되는 경로는 <code><i>schemaName</i> /<i>tableName</i></code> /입니다. '{"BucketFolder": " <i>sourceData</i> "}'
BucketName	S3 버킷의 이름. '{"BucketName": " <i>amzn-s3-demo-bucket</i> "}'
CdcPath	CDC 파일의 위치. 작업에서 변경 데이터를 캡처하는 경우에는 이 속성이 필수이며, 그렇지 않은 경우에는 선택 사항입니다. 이 있는 경우 CdcPath는이 경로에서 CDC 파일을 AWS DMS 읽고 대상 엔드포인트에 데이터 변경 사항을 복제합니다. 자세한 내용은 Amazon S3와 함께 CDC를의 소스로 사용 AWS DMS 단원을 참조하십시오.

옵션	설명
	<code>'{"CdcPath": " <i>changeData</i> "}'</code>
CsvDelimiter	원본 파일에서 열을 구분하는 데 사용되는 구분 기호입니다. 기본값은 쉼표입니다. 예를 들면 다음과 같습니다. <code>'{"CsvDelimiter": ","}'</code>
CsvNullValue	소스에서 읽을 때 null로 AWS DMS 취급되는 사용자 정의 문자열입니다. 기본값은 빈 문자열입니다. 이 파라미터를 설정하지 않으면 빈 문자열을 null 값으로 AWS DMS 처리합니다. 이 파라미터를 "\N"과 같은 문자열로 설정하면 이 문자열을 null 값으로 AWS DMS 취급하고 빈 문자열을 빈 문자열 값으로 취급합니다.
CsvRowDelimiter	원본 파일에서 행을 구분하는 데 사용되는 구분 기호입니다. 기본값은 줄바꿈(\n)입니다. <code>'{"CsvRowDelimiter": "\n"}'</code>
DataFormat	Parquet 형식으로 데이터를 읽으려면 이 값을 Parquet으로 설정합니다. <code>'{"DataFormat": "Parquet"}'</code>
IgnoreHeaderRows	이 값을 1로 설정하면 .csv 파일의 첫 번째 행 헤더를 AWS DMS 무시합니다. 1 값은 이 기능을 활성화하고, 0 값은 비활성화합니다. 기본값은 0입니다. <code>'{"IgnoreHeaderRows": 1}'</code>

옵션	설명
Rfc4180	이 값을 true 또는 y로 설정하면 여는 큰따옴표 뒤에 닫는 큰따옴표가 와야 합니다. 이 형식은 RFC 4180을 준수합니다. 이 값을 false 또는 n으로 설정하면 문자열 리터럴이 그대로 대상에 복사됩니다. 이 경우 구분 기호(행 또는 열)는 필드의 끝을 알립니다. 이렇게 구분 기호는 값의 끝을 알리는 것이므로 문자열의 일부로 사용할 수 없습니다. 기본값은 true입니다. 유효값: true, false, y, n '{"Rfc4180": false}'

Amazon S3의 소스 데이터 형식

Amazon S3를 소스로 사용하는 데이터 마이그레이션은 Amazon S3의 데이터를 AWS DMS 데이터 형식으로 매핑 AWS DMS 해야 합니다. 자세한 내용은 [Amazon S3용 외부 테이블의 소스로 정의의 AWS DMS](#) 단원을 참조하십시오.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

다음 AWS DMS 데이터 형식은 Amazon S3를 소스로 사용할 때 사용됩니다.

- BYTE - ColumnLength가 필요합니다. 자세한 내용은 [Amazon S3용 외부 테이블의 소스로 정의의 AWS DMS](#) 단원을 참조하십시오.
- DATE
- TIME
- DATETIME - 자세한 내용과 예제는 [Amazon S3용 외부 테이블의 소스로 정의의 AWS DMS](#)에서 DATETIME 형식 예제를 참조하세요.
- INT1
- INT2
- INT4

- INT8
- NUMERIC - ColumnPrecision 및 ColumnScale.는 다음 최대값을 AWS DMS 지원합니다.
 - ColumnPrecision: 38
 - ColumnScale: 31

자세한 내용과 예제는 [Amazon S3용 외부 테이블을의 소스로 정의 AWS DMS](#)에서 NUMERIC 형식 예제를 참조하세요.

- REAL4
- REAL8
- STRING - ColumnLength가 필요합니다. 자세한 내용은 [Amazon S3용 외부 테이블을의 소스로 정의 AWS DMS](#) 단원을 참조하십시오.
- UINT1
- UINT2
- UINT4
- UINT8
- BLOB
- CLOB
- BOOLEAN

Amazon S3의 Parquet 형식 파일의 소스로 사용 AWS DMS

AWS DMS 버전 3.5.3 이상에서는 S3 버킷의 Parquet 형식 파일을 전체 로드 또는 CDC 복제의 소스로 사용할 수 있습니다.

DMS는 데이터를 S3 대상 엔드포인트로 마이그레이션하여 DMS가 생성하는 소스로 Parquet 형식 파일만 지원합니다. 파일 이름은 지원되는 형식이어야 합니다. 그렇지 않으면 DMS는 마이그레이션에 이를 포함하지 않습니다.

Parquet 형식의 소스 데이터 파일의 경우 다음 폴더 및 명명 규칙을 따라야 합니다.

```
schema/table1/LOAD00001.parquet
schema/table2/LOAD00002.parquet
schema/table2/LOAD00003.parquet
```

Parquet 형식의 CDC 데이터에 대한 소스 데이터 파일의 경우 다음 폴더 및 명명 규칙을 사용하여 이름을 지정하고 저장합니다.

```
schema/table/20230405-094615814.parquet
schema/table/20230405-094615853.parquet
schema/table/20230405-094615922.parquet
```

Parquet 형식의 파일에 액세스하려면 다음 엔드포인트 설정을 지정합니다.

- DataFormat를 Parquet으로 설정합니다.
- cdcPath 설정을 지정하지 마세요. 지정된 스키마/테이블 폴더에 Parquet 형식 파일을 생성해야 합니다.

S3 엔드포인트 설정에 대한 자세한 내용은 AWS Database Migration Service API 참조의 [S3Settings](#)를 참조하세요.

Parquet 형식 파일에 지원되는 데이터 유형

AWS DMS 는 Parquet 형식 파일에서 데이터를 마이그레이션할 때 다음과 같은 소스 및 대상 데이터 형식을 지원합니다. 마이그레이션하기 전에 대상 테이블에 올바른 데이터 유형의 열이 있는지 확인합니다.

소스 데이터 유형	대상 데이터 유형
BYTE	BINARY
DATE	DATE32
TIME	TIME32
DATETIME	TIMESTAMP
INT1	INT8
INT2	INT16
INT4	INT32
INT8	INT64

소스 데이터 유형	대상 데이터 유형
NUMERIC	DECIMAL
REAL4	FLOAT
REAL8	DOUBLE
STRING	STRING
UINT1	UINT8
UINT2	UINT16
UINT4	UINT32
UINT8	UINT
WSTRING	STRING
BLOB	BINARY
NCLOB	STRING
CLOB	STRING
BOOLEAN	BOOL

Linux, Unix, Windows 및 Amazon RDS 데이터베이스(Db2 LUW)용 IBM Db2 를의 소스로 사용 AWS DMS

AWS Database Migration Service ()를 사용하여 Linux, Unix, Windows 및 Amazon RDS용 IBM Db2(Db2 LUW) 데이터베이스에서 지원되는 모든 대상 데이터베이스로 데이터를 마이그레이션할 수 있습니다AWS DMS.

가 소스로 AWS DMS 지원하는 Linux, Unix, Windows 및 RDS의 Db2 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

Secure Sockets Layer(SSL)를 사용하여 Db2 LUW 엔드포인트와 복제 인스턴스 간 연결을 암호화할 수 있습니다. Db2 LUW 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 [에서 SSL 사용 AWS Database Migration Service](#) 섹션을 참조하세요.

가 IBM Db2 소스 데이터베이스에서 데이터를 AWS DMS 읽을 때 Db2 버전 9.7 이상의 기본 격리 수준 CURSOR STABILITY(CS)을 사용합니다. 자세한 내용은 [IBM DB2 for Linux, UNIX 및 Windows](#) 설명서를 참조하세요.

Db2 LUW를의 소스로 사용할 때의 사전 조건 AWS DMS

Db2 LUW 데이터베이스를 소스로 사용하기 전에 다음 사전 조건이 요구됩니다.

변경 데이터 캡처(CDC)라고도 하는 지속적 복제를 활성화하기 위해서는 다음을 수행하세요.

- 데이터베이스를 복구 가능으로 설정하여가 변경 사항을 캡처 AWS DMS 해야 합니다. 데이터베이스 구성 파라미터인 LOGARCHMETH1과 LOGARCHMETH2 중 하나, 또는 둘 모두가 ON으로 설정되어 있어야 데이터베이스를 복구할 수 있습니다.

데이터베이스를 복구할 수 있는 경우는 필요한 ARCHIVE LOG 경우 Db2에 액세스할 AWS DMS 수 있습니다.

- DB2 트랜잭션 로그를 사용할 수 있고 처리할 보존 기간이 충분한지 확인합니다 AWS DMS.
- DB2에는 트랜잭션 로그 레코드를 추출하기 위한 SYSADM 또는 DBADM 권한 부여가 필요합니다. 사용자 계정에 다음 권한을 부여합니다.
 - SYSADM 또는 DBADM
 - DATAACCESS

Note

전체 로드 전용 작업의 경우, DMS 사용자 계정에 DATAACCESS 권한이 필요합니다.

- IBM DB2 for LUW 버전 9.7을 소스로 사용하는 경우, 추가 연결 속성(ECA) CurrentLsn을 다음과 같이 설정합니다.

CurrentLsn=**LSN**, 여기서 **LSN**은 복제가 시작될 로그 시퀀스 번호(LSN)를 지정합니다. 또는 CurrentLsn=**scan**.

- Amazon RDS for Db2 LUW를 소스로 사용하는 경우 아카이브 로그를 사용할 수 있는지 확인합니다 AWS DMS. AWS관리형 Db2 데이터베이스는 아카이브 로그를 최대한 빨리 제거하므로 로그를 계속 사용할 수 있는 시간을 늘려야 합니다. 예를 들어, 로그 보존 시간을 24시간으로 늘리려면 다음 명령을 실행합니다.

```
db2 "call rdsadmin.set_archive_log_retention( ?, 'TESTDB', '24' )"
```

Amazon RDS for Db2 LUW 절차에 대한 자세한 내용은 [Amazon Relational Database Service 사용 설명서](#)의 Amazon RDS for Db2 저장 프로시저 참조를 참조하세요.

- DB2별 마이그레이션 전 평가를 사용하는 경우 다음 권한을 부여합니다.

```
GRANT CONNECT ON DATABASE TO USER <DMS_USER>;
GRANT SELECT ON SYSIBM.SYSDUMMY1 TO USER <DMS_USER>;
GRANT SELECT ON SYSIBMADM.ENV_INST_INFO TO USER <DMS_USER>;
GRANT SELECT ON SYSIBMADM.DBCFG TO USER <DMS_USER>;
GRANT SELECT ON SYSCAT.SCHEMATA TO USER <DMS_USER>;
GRANT SELECT ON SYSCAT.COLUMNS TO USER <DMS_USER>;
GRANT SELECT ON SYSCAT.TABLES TO USER <DMS_USER>;
GRANT EXECUTE ON FUNCTION SYSPROC.AUTH_LIST_AUTHORITIES_FOR_AUTHID TO <DMS_USER>;
GRANT EXECUTE ON PACKAGE NULLID.SYSSH200 TO USER <DMS_USER>;
```

Db2 LUW를 소스로 사용할 때의 제한 사항 AWS DMS

AWS DMS 는 클러스터링된 데이터베이스를 지원하지 않습니다. 하지만 클러스터의 각 엔드포인트에 별도의 Db2 LUW를 정의할 수 있습니다. 예를 들어 클러스터의 노드 중 하나를 사용하여 전체 로드 마이그레이션 작업을 생성한 다음 각 노드에서 별도의 작업을 생성할 수 있습니다.

AWS DMS 는 소스 Db2 LUW 데이터베이스의 BOOLEAN 데이터 형식을 지원하지 않습니다.

지속적 복제(CDC)를 사용할 때 다음 제한 사항이 적용됩니다.

- 파티션이 여러 개인 테이블이 잘리면 AWS DMS 콘솔에 표시된 DDL 이벤트 수가 파티션 수와 같습니다. Db2 LUW가 각 파티션에 별개의 DDL을 기록하기 때문입니다.
- 분할된 테이블에서는 다음 DDL 작업이 지원되지 않습니다.
 - ALTER TABLE ADD PARTITION
 - ALTER TABLE DETACH PARTITION
 - ALTER TABLE ATTACH PARTITION
- AWS DMS 는 DB2 고가용성 재해 복구(HADR) 대기 인스턴스에서 지속적인 복제 마이그레이션을 지원하지 않습니다. 대기 인스턴스에 액세스할 수 없습니다.
- DECFLOAT 데이터 형식은 지원되지 않습니다. 따라서 지속적 복제 진행 중 DECFLOAT 열들의 변경 내용이 무시됩니다.

- RENAME COLUMN 문은 지원되지 않습니다.
- 다중 차원 클러스터링(MDC) 테이블에 대한 업데이트를 수행할 때 각 업데이트는 AWS DMS 콘솔에 INSERT + DELETE로 표시됩니다.
- 작업 설정 복제에 LOB 열 포함이 활성화되지 않은 경우, LOB 열이 있는 테이블은 지속적 복제 도중 일시 중지됩니다.
- Db2 LUW 버전 10.5 이상의 경우, 데이터가 행 외부에 저장된 가변 길이 문자열 열은 무시됩니다. 이 제한은 VARCHAR 및 VARGRAPHIC과 같은 데이터 형식을 가진 열의 확장된 행 크기로 생성된 테이블에만 적용됩니다. 이 제한을 해결하려면 페이지 크기가 더 큰 테이블 공간으로 테이블을 이동하세요. 자세한 내용은 [What can I do if I want to change the pagesize of DB2 tablespaces](#)를 참조하세요.
- 지속적 복제의 경우, DMS는 DB2 LOAD 유틸리티에 의해 페이지 수준에서 로드된 데이터의 마이그레이션을 지원하지 않습니다. 대신 SQL 삽입을 사용하는 IMPORT 유틸리티를 사용하세요. 자세한 내용은 [differences between the import and load utilities](#)를 참조하세요.
- 복제 작업이 실행되는 동안 DMS는 테이블이 DATA CAPTURE CHANGE 속성으로 생성된 경우에만 CREATE TABLE DDL을 캡처합니다.
- DMS는 Db2 데이터베이스 파티션 기능(DPF)을 사용할 때 다음과 같은 제한 사항이 있습니다.
 - DMS는 DPF 환경의 Db2 노드 간 트랜잭션을 조정할 수 없습니다. 이는 IBM DB2READLOG API 인터페이스 내의 제약으로 인한 것입니다. DPF에서 트랜잭션은 DB2가 데이터를 분할하는 방식에 따라 여러 Db2 노드에 걸쳐 있을 수 있습니다. 따라서 DMS 솔루션은 각 Db2 노드에서 독립적으로 트랜잭션을 캡처해야 합니다.
 - DMS는 여러 DMS 소스 엔드포인트에서 connectNode를 1로 설정하여 DPF 클러스터의 각 Db2 노드에서 로컬 트랜잭션을 캡처할 수 있습니다. 이 구성은 DB2 서버 구성 파일 db2nodes.cfg에 정의된 논리적 노드 번호에 해당합니다.
 - 개별 Db2 노드의 로컬 트랜잭션은 대규모 글로벌 트랜잭션의 일부일 수 있습니다. DMS는 다른 Db2 노드의 트랜잭션과 조정하지 않고 대상에 각 로컬 트랜잭션을 독립적으로 적용합니다. 이러한 독립적인 처리는 특히 파티션 간에 행을 이동할 때 복잡성을 초래할 수 있습니다.
 - DMS는 각 Db2 노드에 독립적으로 작업을 적용하므로 DMS가 여러 Db2 노드에서 복제할 때 대상에 대한 올바른 작업 순서를 보장할 수 없습니다. 각 Db2 노드와 독립적으로 로컬 트랜잭션을 캡처하는 것이 특정 사용 사례에 적합한지 확인해야 합니다.
 - DPF 환경에서 마이그레이션할 때는 먼저 캐시된 이벤트 없이 전체 로드 작업을 실행한 다음 CDC 전용 태스크를 실행하는 것이 좋습니다. StartFromContext 엔드포인트 추가 연결 속성을 사용하여 설정한 전체 로드 시작 타임스탬프 또는 LRI(로그 레코드 식별자)부터 시작하여 Db2 노드당 하나의 작업을 실행하는 것이 좋습니다. 복제 시작점을 결정하는 방법에 대한 자세한 내용은 IBM 지원 설명서의 [복제 시작을 위한 LSN 또는 LRI 값 찾기](#)를 참조하세요.

- 지속적 복제(CDC)의 경우 특정 타임스탬프에서 복제를 시작하려면 StartFromContext 추가 연결 속성을 필수 타임스탬프로 설정해야 합니다.
- 현재 DMS는 데이터베이스 솔루션을 확장하는 데 사용할 수 있는 DB2 LUW의 확장인 Db2 PureScale 기능을 지원하지 않습니다.
- DATA CAPTURE CHANGES 테이블 옵션은 DB2 데이터 복제 프로세스의 중요한 사전 조건입니다. 테이블을 생성할 때 이 옵션을 활성화하지 않으면 데이터가 누락될 수 있습니다. 특히 CDC(Change Data Capture)의 경우 이전 시작점에서 시작된 복제 작업만 해당됩니다. AWS DMS는 CDC 또는 FULL+CDC 작업을 다시 시작할 때 기본적으로 이 속성을 활성화합니다. 그러나 작업 재시작 전에 소스 데이터베이스에서 변경한 내용은 누락될 수 있습니다.

```
ALTER TABLE TABLE_SCHEMA.TABLE_NAME DATA CAPTURE CHANGES INCLUDE LONGVAR COLUMNS;
```

Db2 LUW를의 소스로 사용할 때 엔드포인트 설정 AWS DMS

AWS DMS 콘솔을 사용하거나에서 create-endpoint 명령을 사용하여 소스 엔드포인트를 생성할 때 설정을 지정할 수 [AWS CLI](#) 있습니다.

```
--ibm-db2-settings '{"EndpointSetting1": "value1", "EndpointSetting2": "value2"}'
```

JSON 구문.

Db2 LUW를 소스로 하여 사용할 수 있는 엔드포인트 설정은 다음 표에 나와 있습니다.

설정 이름	설명
CurrentLsn	지속적 복제(CDC)의 경우, CurrentLsn 을 사용하여 복제가 시작될 로그 시퀀스 번호(LSN)를 지정합니다.
MaxKBytesPerRead	읽기 당 바이트의 최대 수(NUMBER 값). 기본값은 64KB입니다.
SetDataCaptureChanges	부울 값으로 지속적 복제(CDC)를 활성화합니다. 기본값은 true입니다.

Db2 LUW를의 소스로 사용할 때 추가 연결 속성(ECAs) AWS DMS

AWS DMS 콘솔을 사용하거나에서 `create-endpoint` 명령을 사용하여 소스 엔드포인트를 생성할 때 추가 연결 속성(ECAs)을와 [AWS CLI](#) 함께 지정할 수 있습니다.

```
--extra-connection-attributes 'ECAname1=value1;ECAname2=value2;'
```

다음 표에는 Db2 LUW를 소스로 사용할 수 있는 ECAs가 나와 있습니다.

속성 이름	설명
ConnectionTimeout	이 ECA를 사용하여 Db2 LUW 엔드포인트의 엔드포인트 연결 제한 시간을 초 단위로 설정합니다. 기본값은 10초입니다. 예시: ConnectionTimeout=30;
executeTimeout	DB2 LUW 엔드포인트에 대한 문(쿼리) 제한 시간을 초 단위로 설정하는 추가 연결 속성입니다. 기본값은 60초입니다. 예시: executeTimeout=120;
StartFromContext	지속적 복제(CDC)의 경우, StartFromContext 를 사용하여 복제가 시작될 로그 하한선을 지정합니다. StartFromContext 는 다양한 형식의 값을 허용합니다. 유효한 값으로는 다음이 포함됩니다. - timestamp (UTC). 예시: StartFromContext=timestamp: 2026-02-23T13:00:00; - NOW IBM DB2 LUW 버전 10.5 이상의 경우 이제 '{"CurrentLsn": "scan"}' 엔드포인트 설정과 함께가 최신 로그 시퀀스 오프셋(LSO)에서 작업을 시작합니다. 예시: StartFromContext=NOW;

속성 이름	설명
	특정 로그 레코드 식별자(LRI). 예시: StartFromContext=0100000000 000022CC000000000004FB13; 로그 파일의 LRI/LSN 범위를 확인하려면 다음 예제와 같이 db2f1sn 명령을 실행합니다. <pre>db2f1sn -db SAMPLE -lrrange 2</pre> 예제의 출력은 다음과 비슷합니다. <pre>S0000002.LOG: has LRI range 000000000000000100 0000000000225400000000004F9A6 to 0000000000000001000000000022CC0000000004 FB13</pre> 이 출력에서 로그 파일은 S0000002.LOG이고 StartFrom Context LRI 값은 범위 끝에 있는 34바이트입니다. <pre>0100000000000022CC00000000004FB13</pre>

IBM DB2 LUW용 소스 데이터 형식

Db2 LUW를 소스로 사용하는 데이터 마이그레이션은 대부분의 Db2 LUW 데이터 유형을 AWS DMS 지원합니다. 다음 표에는 사용 시 지원되는 Db2 LUW 소스 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다. Db2 LUW 데이터 형식에 대한 자세한 내용은 [Db2 LUW 설명서](#)를 참조하세요.

대상에서 매핑된 데이터 형식을 보는 방법에 대한 자세한 내용은 사용 중인 대상 엔드포인트 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

Db2 LUW 데이터 형식	AWS DMS 데이터 형식
INTEGER	INT4
SMALLINT	INT2
BIGINT	INT8
DECIMAL (p,s)	NUMERIC(p,s)
FLOAT	REAL8
DOUBLE	REAL8
REAL	REAL4
DECFLOAT (p)	정밀도가 16이면 REAL8, 정밀도가 34이면 STRING
GRAPHIC (n)	길이가 0보다 크고 127 이하인 2바이트 문자 고정 길이 그래픽 문자열의 경우, WSTRING
VARGRAPHIC (n)	길이가 0보다 크고 16,352 이하인 2바이트 문자 가변 길이 그래픽 문자열의 경우, WSTRING
LONG VARGRAPHIC (n)	길이가 0보다 크고 16,352 이하인 2바이트 문자 가변 길이 그래픽 문자열의 경우, CLOB
CHARACTER (n)	길이가 0보다 크고 255 이하인 2바이트 문자 고정 길이 문자열의 경우, STRING
VARCHAR (n)	길이가 0보다 크고 32,704 이하인 2바이트 문자 가변 길이 문자열의 경우, STRING
LONG VARCHAR (n)	길이가 0보다 크고 32,704 이하인 2바이트 문자 가변 길이 문자열의 경우, CLOB
CHAR (n) FOR BIT DATA	BYTES
VARCHAR (n) FOR BIT DATA	BYTES

Db2 LUW 데이터 형식	AWS DMS 데이터 형식
LONG VARCHAR FOR BIT DATA	BYTES
DATE	DATE
TIME	TIME
TIMESTAMP	DATETIME
BLOB (n)	BLOB 최대 길이는 2,147,483,647바이트
CLOB (n)	CLOB 최대 길이는 2,147,483,647바이트
DBCLOB (n)	CLOB 최대 길이는 1,073,741,824 2바이트 문자
XML	CLOB

IBM Db2 for z/OS 데이터베이스를의 소스로 사용 AWS DMS

AWS Database Migration Service (AWS DMS)를 사용하여 IBM for z/OS 데이터베이스에서 데이터를 마이그레이션할 수 있습니다.

가 소스로 AWS DMS 지원하는 z/OS용 Db2 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 소스 AWS DMS](#).

z/OS용 Db2를의 소스로 사용할 때의 사전 조건 AWS DMS

IBM Db2 for z/OS 데이터베이스를 소스로 사용하려면 소스 엔드포인트 연결 설정에 지정된 Db2 for z/OS 사용자에게 다음 권한을 AWS DMS부여합니다.

```
GRANT SELECT ON SYSIBM.SYSTABLES TO Db2USER;
GRANT SELECT ON SYSIBM.SYSTABLESPACE TO Db2USER;
GRANT SELECT ON SYSIBM.SYSTABLEPART TO Db2USER;
GRANT SELECT ON SYSIBM.SYSCOLUMNS TO Db2USER;
```

```
GRANT SELECT ON SYSIBM.SYSDATABASE TO Db2USER;  
GRANT SELECT ON SYSIBM.SYSDUMMY1 TO Db2USER
```

user defined 소스 테이블에 대한 SELECT 권한도 부여하세요.

AWS DMS IBM Db2 for z/OS 소스 엔드포인트는 IBM Data Server Driver for ODBC를 사용하여 데이터에 액세스합니다. DMS가 이 엔드포인트에 연결하려면 데이터베이스 서버에 유효한 IBM ODBC Connect 라이선스가 있어야 합니다.

z/OS용 Db2를 소스로 사용할 때의 제한 사항 AWS DMS

IBM DB2 for z/OS를 AWS DMS소스로 사용 시 다음 제한 사항이 적용됩니다.

- 전체 로드 복제 작업만 지원됩니다. 변경 데이터 캡처(CDC)는 지원되지 않습니다.
- 병렬 로드는 지원되지 않습니다.
- 뷰의 데이터 검증은 지원되지 않습니다.
- 열/테이블 수준 변환 및 행 수준 선택 필터를 위한 테이블 매핑에서 스키마, 테이블, 열 이름을 대문자로 지정해야 합니다.

IBM DB2 for z/OS 소스 데이터 형식

z/OS용 Db2를 소스로 사용하는 데이터 마이그레이션은 z/OS용 대부분의 Db2 데이터 형식을 AWS DMS 지원합니다. 다음 표에는 사용 시 지원되는 z/OS 소스 데이터 형식의 Db2 AWS DMS와 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

Db2 for z/OS 데이터 형식에 대한 자세한 내용은 [IBM DB2 for z/OS 설명서](#)를 참조하세요.

대상에서 매핑된 데이터 형식을 보는 방법에 대한 자세한 내용은 사용 중인 대상 엔드포인트 섹션을 참조하세요.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

Db2 for z/OS 데이터 형식	AWS DMS 데이터 형식
INTEGER	INT4
SMALLINT	INT2

Db2 for z/OS 데이터 형식	AWS DMS 데이터 형식
BIGINT	INT8
DECIMAL (p,s)	NUMERIC(p,s) DB2 구성에서 소수점이 쉼표(,)로 설정된 경우, DB2 설정을 지원하도록 Replicate를 구성하세요.
FLOAT	REAL8
DOUBLE	REAL8
REAL	REAL4
DECFLOAT (p)	정밀도가 16이면 REAL8, 정밀도가 34이면 STRING
GRAPHIC (n)	n>=127이면 길이가 0보다 크고 127 이하인 2 바이트 문자 고정 길이 그래픽 문자열의 경우, WSTRING
VARGRAPHIC (n)	길이가 0보다 크고 16,352 이하인 2바이트 문자 가변 길이 그래픽 문자열의 경우, WSTRING
LONG VARGRAPHIC (n)	길이가 0보다 크고 16,352 이하인 2바이트 문자 가변 길이 그래픽 문자열의 경우, CLOB
CHARACTER (n)	길이가 0보다 크고 255 이하인 2바이트 문자 고정 길이 문자열의 경우, STRING
VARCHAR (n)	길이가 0보다 크고 32,704 이하인 2바이트 문자 가변 길이 문자열의 경우, STRING
LONG VARCHAR (n)	길이가 0보다 크고 32,704 이하인 2바이트 문자 가변 길이 문자열의 경우, CLOB
CHAR (n) FOR BIT DATA	BYTES

Db2 for z/OS 데이터 형식	AWS DMS 데이터 형식
VARCHAR (n) FOR BIT DATA	BYTES
LONG VARCHAR FOR BIT DATA	BYTES
DATE	DATE
TIME	TIME
TIMESTAMP	DATETIME
BLOB (n)	BLOB 최대 길이는 2,147,483,647바이트
CLOB (n)	CLOB 최대 길이는 2,147,483,647바이트
DBCLOB (n)	CLOB 최대 길이는 1,073,741,824 2바이트 문자
XML	CLOB
BINARY	BYTES
VARBINARY	BYTES
ROWID	BYTES ROWID 작업에 대한 자세한 내용은 다음을 참조하세요.
TIMESTAMP(시간대 사용)	지원되지 않음.

작업의 대상 테이블 준비 모드가 DROP_AND_CREATE(기본값)로 설정된 경우, ROWID 열이 기본적으로 마이그레이션됩니다. 특정 데이터베이스 및 테이블 외부에서는 행이 무의미하므로 데이터 검증은 이러한 열을 무시합니다. 이러한 열의 마이그레이션을 끄려면 다음 준비 단계 중 하나를 수행하면 됩니다.

- 이러한 열이 없는 대상 테이블을 미리 생성합니다. 그런 다음 작업의 대상 테이블 준비 모드를 DO_NOTHING 또는 TRUNCATE_BEFORE_LOAD로 설정합니다. AWS Schema Conversion Tool (AWS SCT)를 사용하여 열 없이 대상 테이블을 미리 생성할 수 있습니다.
- 이러한 열을 필터링하여 무시되도록 하는 테이블 매핑 규칙을 작업에 추가합니다. 자세한 내용은 [변환 규칙 및 작업](#) 단원을 참조하십시오.

PostgreSQL for AWS Mainframe Modernization 서비스의 EBCDIC 데이터 정렬

AWS 메인프레임 현대화 프로그램은 메인프레임 애플리케이션을 AWS 관리형 런타임 환경으로 현대화하는 데 도움이 됩니다. 이 서비스는 마이그레이션 및 현대화 프로젝트를 계획하고 구현하는 데 도움이 되는 도구와 리소스를 제공합니다. 메인프레임 현대화 및 마이그레이션에 대한 자세한 내용은 [를 사용한 메인프레임 현대화 AWS](#)를 참조하세요.

일부 IBM DB2 for z/OS 데이터세트는 EBCDIC(Extended Binary Coded Decimal Interchange) 문자 집합으로 인코딩됩니다. 이 문자 집합은 ASCII(American Standard Code for Information Interchange)가 일반적으로 사용되기 전에 개발되었습니다. 코드 페이지는 텍스트의 각 문자를 문자 집합의 문자에 매핑합니다. 기존 코드 페이지에는 코드 포인트와 문자 ID 간의 매핑 정보가 들어 있습니다. 문자 ID는 8 바이트 문자 데이터 문자열입니다. 코드 포인트는 문자를 나타내는 8비트 2진수입니다. 코드 포인트는 일반적으로 해당 2진수 값의 16진수 표현으로 표시됩니다.

현재 Mainframe Modernization 서비스의 Micro Focus 또는 BluAge 구성 요소를 사용하는 경우 특정 코드 포인트를 이동(번역) AWS DMS 하도록에 지시해야 합니다. AWS DMS 작업 설정을 사용하여 교대 근무를 수행할 수 있습니다. 다음 예제에서는 작업을 사용하여 AWS DMS CharacterSetSettings DMS 작업 설정의 교대를 매핑하는 방법을 보여줍니다.

```
"CharacterSetSettings": {
    "CharacterSetSupport": null,
    "CharacterReplacements": [
        {"SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180"},
        {"SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0160"},
        {"SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "0161"},
        {"SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017D"},
        {"SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "017E"},
        {"SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0152"},
        {"SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0153"},
        {"SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0178"}
    ]
}
```

필요한 시프팅을 이해하는 EBCDIC 데이터 정렬이 PostgreSQL에 이미 일부 있습니다. 여러 가지 코드 페이지가 지원됩니다. 다음 섹션에서는 지원되는 모든 코드 페이지에서 시프트해야 하는 항목의 JSON 샘플을 제공합니다. DMS 작업에 필요한 필수 JSON을 간단히 복사하여 붙여넣을 수 있습니다.

Micro Focus 관련 EBCDIC 데이터 정렬

Micro Focus의 경우, 다음 데이터 정렬에 필요한 문자 하위 집합을 시프트합니다.

```
da-DK-cp1142m-x-icu
de-DE-cp1141m-x-icu
en-GB-cp1146m-x-icu
en-US-cp1140m-x-icu
es-ES-cp1145m-x-icu
fi-FI-cp1143m-x-icu
fr-FR-cp1147m-x-icu
it-IT-cp1144m-x-icu
nl-BE-cp1148m-x-icu
```

Example데이터 정렬에 따른 Micro Focus 데이터 시프트:

en_us_cp1140m

코드 시프트:

```
0000    0180
00A6    0160
00B8    0161
00BC    017D
00BD    017E
00BE    0152
00A8    0153
00B4    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0160"
}, {
  "SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0161"
}, {
  "SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "017D"
}, {
  "SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017E"
}
```

```
, {"SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "0152"}
, {"SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0153"}
, {"SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0178"}
```

en_us_cp1141m

코드 시프트:

0000	0180
00B8	0160
00BC	0161
00BD	017D
00BE	017E
00A8	0152
00B4	0153
00A6	0178

AWS DMS 작업에 해당하는 입력 매핑:

```
{ "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180" }
, { "SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0160" }
, { "SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "0161" }
, { "SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017D" }
, { "SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "017E" }
, { "SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0152" }
, { "SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0153" }
, { "SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0178" }
```

en_us_cp1142m

코드 시프트:

0000	0180
00A6	0160
00B8	0161
00BC	017D
00BD	017E
00BE	0152
00A8	0153
00B4	0178

AWS DMS 작업에 해당하는 입력 매핑:

```
{ "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180" }
, { "SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0160" }
, { "SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0161" }
, { "SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "017D" }
, { "SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017E" }
, { "SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "0152" }
, { "SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0153" }
, { "SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0178" }
```

en_us_cp1143m

코드 시프트:

```
0000    0180
00B8    0160
00BC    0161
00BD    017D
00BE    017E
00A8    0152
00B4    0153
00A6    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
{ "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180" }
, { "SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0160" }
, { "SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "0161" }
, { "SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017D" }
, { "SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "017E" }
, { "SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0152" }
, { "SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0153" }
, { "SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0178" }
```

en_us_cp1144m

코드 시프트:

0000	0180
00B8	0160
00BC	0161
00BD	017D
00BE	017E
00A8	0152
00B4	0153
00A6	0178

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0160"
}, {
  "SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "0161"
}, {
  "SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017D"
}, {
  "SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "017E"
}, {
  "SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0152"
}, {
  "SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0153"
}, {
  "SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0178"
}
```

en_us_cp1145m

코드 시프트:

0000	0180
00A6	0160
00B8	0161
00A8	017D
00BC	017E
00BD	0152
00BE	0153
00B4	0178

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180"
}
```

```
, {"SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0160"}
, {"SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0161"}
, {"SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "017D"}
, {"SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "017E"}
, {"SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "0152"}
, {"SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "0153"}
, {"SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0178"}
```

en_us_cp1146m

코드 시프트:

0000	0180
00A6	0160
00B8	0161
00BC	017D
00BD	017E
00BE	0152
00A8	0153
00B4	0178

AWS DMS 작업에 해당하는 입력 매핑:

```
{ "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180" }
, { "SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0160" }
, { "SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0161" }
, { "SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "017D" }
, { "SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017E" }
, { "SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "0152" }
, { "SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0153" }
, { "SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0178" }
```

en_us_cp1147m

코드 시프트:

0000	0180
00B8	0160
00A8	0161

00BC	017D
00BD	017E
00BE	0152
00B4	0153
00A6	0178

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0160"
}, {
  "SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0161"
}, {
  "SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "017D"
}, {
  "SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017E"
}, {
  "SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "0152"
}, {
  "SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0153"
}, {
  "SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0178"
}
```

en_us_cp1148m

코드 시프트:

0000	0180
00A6	0160
00B8	0161
00BC	017D
00BD	017E
00BE	0152
00A8	0153
00B4	0178

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0000", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "00A6", "TargetCharacterCodePoint": "0160"
}, {
  "SourceCharacterCodePoint": "00B8", "TargetCharacterCodePoint": "0161"
}, {
  "SourceCharacterCodePoint": "00BC", "TargetCharacterCodePoint": "017D"
}, {
  "SourceCharacterCodePoint": "00BD", "TargetCharacterCodePoint": "017E"
}, {
  "SourceCharacterCodePoint": "00BE", "TargetCharacterCodePoint": "0152"
}, {
  "SourceCharacterCodePoint": "00A8", "TargetCharacterCodePoint": "0153"
}, {
  "SourceCharacterCodePoint": "00B4", "TargetCharacterCodePoint": "0178"
}
```

BluAge 관련 EBCDIC 데이터 정렬

BluAge의 경우, 필요에 따라 다음의 낮은 값과 높은 값을 모두 시프트합니다. 이러한 데이터 정렬은 Mainframe Migration BluAge 서비스를 지원하는 용도로만 사용해야 합니다.

```
da-DK-cp1142b-x-icu
da-DK-cp277b-x-icu
de-DE-cp1141b-x-icu
de-DE-cp273b-x-icu
en-GB-cp1146b-x-icu
en-GB-cp285b-x-icu
en-US-cp037b-x-icu
en-US-cp1140b-x-icu
es-ES-cp1145b-x-icu
es-ES-cp284b-x-icu
fi-FI-cp1143b-x-icu
fi-FI-cp278b-x-icu
fr-FR-cp1147b-x-icu
fr-FR-cp297b-x-icu
it-IT-cp1144b-x-icu
it-IT-cp280b-x-icu
nl-BE-cp1148b-x-icu
nl-BE-cp500b-x-icu
```

Example BluAge 데이터 시프트:

da-DK-cp277b 및 da-DK-cp1142b

코드 시프트:

0180	0180
0001	0181
0002	0182
0003	0183
009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189

008E	018A
000B	018B
000C	018C
000D	018D
000E	018E
000F	018F
0010	0190
0011	0191
0012	0192
0013	0193
009D	0194
0085	0195
0008	0196
0087	0197
0018	0198
0019	0199
0092	019A
008F	019B
001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0
0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5
0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA
008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF
0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5

0096	01B6
0004	01B7
0098	01B8
0099	01B9
009A	01BA
009B	01BB
0014	01BC
0015	01BD
009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181"
}, {
  "SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"
}, {
  "SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"
}, {
  "SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"
}, {
  "SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"
}, {
  "SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"
}, {
  "SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"
}, {
  "SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"
}, {
  "SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"
}, {
  "SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"
}, {
  "SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"
}, {
  "SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"
}, {
  "SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"
}, {
  "SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"
}, {
  "SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"
}, {
  "SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"
}, {
  "SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"
}, {
  "SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"
}, {
  "SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"
}, {
  "SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"
}, {
  "SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"
}, {
  "SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"
}, {
  "SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"
}, {
  "SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"
}, {
  "SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"
}, {
  "SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"
}, {
  "SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"
}
```

```
, {"SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"}
, {"SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"}
, {"SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"}
, {"SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"}
, {"SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"}
, {"SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"}
, {"SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"}
, {"SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"}
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

de-DE-273b 및 de-DE-1141b

코드 시프트:

0180	0180
0001	0181
0002	0182
0003	0183
009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189
008E	018A
000B	018B
000C	018C
000D	018D
000E	018E
000F	018F
0010	0190
0011	0191
0012	0192
0013	0193
009D	0194
0085	0195
0008	0196
0087	0197
0018	0198
0019	0199
0092	019A
008F	019B
001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0
0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5
0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA

008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF
0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5
0096	01B6
0004	01B7
0098	01B8
0099	01B9
009A	01BA
009B	01BB
0014	01BC
0015	01BD
009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181"
}, {
  "SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"
}, {
  "SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"
}, {
  "SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"
}, {
  "SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"
}, {
  "SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"
}, {
  "SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"
}, {
  "SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"
}, {
  "SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"
}, {
  "SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"
}, {
  "SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"
}, {
  "SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"
}, {
  "SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"
}, {
  "SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"
}, {
  "SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"
}, {
  "SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"
}
```

```
, {"SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"}
, {"SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"}
, {"SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"}
, {"SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"}
, {"SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"}
, {"SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"}
, {"SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"}
, {"SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"}
, {"SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"}
, {"SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"}
, {"SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"}
, {"SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"}
, {"SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"}
, {"SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"}
, {"SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"}
, {"SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"}
, {"SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"}
, {"SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"}
, {"SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"}
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
```

```
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

en-GB-285b 및 en-GB-1146b

코드 시프트:

0180	0180
0001	0181
0002	0182
0003	0183
009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189
008E	018A
000B	018B
000C	018C
000D	018D
000E	018E
000F	018F
0010	0190
0011	0191
0012	0192
0013	0193
009D	0194
0085	0195
0008	0196
0087	0197
0018	0198
0019	0199
0092	019A
008F	019B
001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0

0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5
0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA
008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF
0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5
0096	01B6
0004	01B7
0098	01B8
0099	01B9
009A	01BA
009B	01BB
0014	01BC
0015	01BD
009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181"
}, {
  "SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"
}, {
  "SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"
}, {
  "SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"
}, {
  "SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"
}, {
  "SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"
}
```



```
, {"SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"}
, {"SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"}
, {"SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"}
, {"SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"}
, {"SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"}
, {"SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"}
, {"SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"}
, {"SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"}
, {"SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"}
, {"SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"}
, {"SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"}
, {"SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"}
, {"SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"}
, {"SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"}
, {"SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"}
, {"SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"}
, {"SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"}
, {"SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"}
, {"SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"}
, {"SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"}
, {"SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"}
, {"SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"}
, {"SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"}
, {"SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"}
, {"SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"}
, {"SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"}
, {"SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"}
, {"SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"}
, {"SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"}
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
```

```
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

en-us-037b 및 en-us-1140b

코드 시프트:

0180	0180
0001	0181
0002	0182
0003	0183
009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189
008E	018A
000B	018B
000C	018C
000D	018D
000E	018E
000F	018F
0010	0190
0011	0191
0012	0192
0013	0193
009D	0194
0085	0195
0008	0196

0087	0197
0018	0198
0019	0199
0092	019A
008F	019B
001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0
0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5
0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA
008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF
0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5
0096	01B6
0004	01B7
0098	01B8
0099	01B9
009A	01BA
009B	01BB
0014	01BC
0015	01BD
009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```

{"SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180"}
, {"SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181"}
, {"SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"}
, {"SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"}
, {"SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"}
, {"SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"}
, {"SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"}
, {"SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"}
, {"SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"}
, {"SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"}
, {"SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"}
, {"SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"}
, {"SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"}
, {"SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"}
, {"SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"}
, {"SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"}
, {"SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"}
, {"SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"}
, {"SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"}
, {"SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"}
, {"SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"}
, {"SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"}
, {"SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"}
, {"SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"}
, {"SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"}
, {"SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"}
, {"SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"}
, {"SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"}
, {"SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"}
, {"SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"}
, {"SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"}
, {"SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"}
, {"SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"}
, {"SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"}
, {"SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"}
, {"SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"}
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}

```

```
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

es-ES-284b 및 es-ES-1145b

코드 시프트:

0180	0180
0001	0181
0002	0182
0003	0183
009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189
008E	018A
000B	018B
000C	018C
000D	018D

000E	018E
000F	018F
0010	0190
0011	0191
0012	0192
0013	0193
009D	0194
0085	0195
0008	0196
0087	0197
0018	0198
0019	0199
0092	019A
008F	019B
001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0
0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5
0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA
008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF
0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5
0096	01B6
0004	01B7
0098	01B8
0099	01B9

009A	01BA
009B	01BB
0014	01BC
0015	01BD
009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181"
}, {
  "SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"
}, {
  "SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"
}, {
  "SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"
}, {
  "SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"
}, {
  "SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"
}, {
  "SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"
}, {
  "SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"
}, {
  "SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"
}, {
  "SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"
}, {
  "SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"
}, {
  "SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"
}, {
  "SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"
}, {
  "SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"
}, {
  "SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"
}, {
  "SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"
}, {
  "SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"
}, {
  "SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"
}, {
  "SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"
}, {
  "SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"
}, {
  "SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"
}, {
  "SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"
}, {
  "SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"
}, {
  "SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"
}, {
  "SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"
}, {
  "SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"
}, {
  "SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"
}, {
  "SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"
}, {
  "SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"
}, {
  "SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"
}, {
  "SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"
}
```

```
, {"SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"}
, {"SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"}
, {"SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"}
, {"SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"}
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

fi_FI-278b 및 fi-FI-1143b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
```


009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189
008E	018A
000B	018B
000C	018C
000D	018D
000E	018E
000F	018F
0010	0190
0011	0191
0012	0192
0013	0193
009D	0194
0085	0195
0008	0196
0087	0197
0018	0198
0019	0199
0092	019A
008F	019B
001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0
0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5
0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA
008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF

0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5
0096	01B6
0004	01B7
0098	01B8
0099	01B9
009A	01BA
009B	01BB
0014	01BC
0015	01BD
009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181"
}, {
  "SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"
}, {
  "SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"
}, {
  "SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"
}, {
  "SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"
}, {
  "SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"
}, {
  "SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"
}, {
  "SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"
}, {
  "SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"
}, {
  "SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"
}, {
  "SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"
}, {
  "SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"
}, {
  "SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"
}, {
  "SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"
}, {
  "SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"
}, {
  "SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"
}, {
  "SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"
}, {
  "SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"
}, {
  "SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"
}, {
  "SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"
}, {
  "SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"
}
```

```
, {"SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"}
, {"SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"}
, {"SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"}
, {"SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"}
, {"SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"}
, {"SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"}
, {"SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"}
, {"SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"}
, {"SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"}
, {"SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"}
, {"SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"}
, {"SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"}
, {"SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"}
, {"SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"}
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

fr-FR-297b 및 fr-FR-1147b

코드 시프트:

0180	0180
0001	0181
0002	0182
0003	0183
009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189
008E	018A
000B	018B
000C	018C
000D	018D
000E	018E
000F	018F
0010	0190
0011	0191
0012	0192
0013	0193
009D	0194
0085	0195
0008	0196
0087	0197
0018	0198
0019	0199
0092	019A
008F	019B
001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0
0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5

0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA
008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF
0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5
0096	01B6
0004	01B7
0098	01B8
0099	01B9
009A	01BA
009B	01BB
0014	01BC
0015	01BD
009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181"
}, {
  "SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"
}, {
  "SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"
}, {
  "SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"
}, {
  "SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"
}, {
  "SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"
}, {
  "SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"
}, {
  "SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"
}, {
  "SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"
}, {
  "SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"
}, {
  "SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"
}
```

```
, {"SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"}
, {"SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"}
, {"SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"}
, {"SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"}
, {"SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"}
, {"SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"}
, {"SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"}
, {"SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"}
, {"SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"}
, {"SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"}
, {"SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"}
, {"SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"}
, {"SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"}
, {"SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"}
, {"SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"}
, {"SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"}
, {"SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"}
, {"SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"}
, {"SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"}
, {"SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"}
, {"SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"}
, {"SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"}
, {"SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"}
, {"SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"}
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
```

```
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

it-IT-280b 및 it-IT-1144b

코드 시프트:

0180	0180
0001	0181
0002	0182
0003	0183
009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189
008E	018A
000B	018B
000C	018C
000D	018D
000E	018E
000F	018F
0010	0190
0011	0191
0012	0192
0013	0193
009D	0194
0085	0195
0008	0196
0087	0197
0018	0198
0019	0199
0092	019A
008F	019B

001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0
0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5
0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA
008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF
0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5
0096	01B6
0004	01B7
0098	01B8
0099	01B9
009A	01BA
009B	01BB
0014	01BC
0015	01BD
009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```
{ "SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180" }
, { "SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181" }
```



```
, {"SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"}
, {"SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"}
, {"SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"}
, {"SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"}
, {"SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"}
, {"SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"}
, {"SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"}
, {"SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"}
, {"SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"}
, {"SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"}
, {"SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"}
, {"SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"}
, {"SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"}
, {"SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"}
, {"SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"}
, {"SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"}
, {"SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"}
, {"SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"}
, {"SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"}
, {"SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"}
, {"SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"}
, {"SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"}
, {"SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"}
, {"SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"}
, {"SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"}
, {"SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"}
, {"SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"}
, {"SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"}
, {"SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"}
, {"SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"}
, {"SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"}
, {"SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"}
, {"SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"}
, {"SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"}
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
```

```
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

nl-BE-500b 및 nl-BE-1148b

코드 시프트:

0180	0180
0001	0181
0002	0182
0003	0183
009C	0184
0009	0185
0086	0186
007F	0187
0097	0188
008D	0189
008E	018A
000B	018B
000C	018C
000D	018D
000E	018E
000F	018F
0010	0190
0011	0191

0012	0192
0013	0193
009D	0194
0085	0195
0008	0196
0087	0197
0018	0198
0019	0199
0092	019A
008F	019B
001C	019C
001D	019D
001E	019E
001F	019F
0080	01A0
0081	01A1
0082	01A2
0083	01A3
0084	01A4
000A	01A5
0017	01A6
001B	01A7
0088	01A8
0089	01A9
008A	01AA
008B	01AB
008C	01AC
0005	01AD
0006	01AE
0007	01AF
0090	01B0
0091	01B1
0016	01B2
0093	01B3
0094	01B4
0095	01B5
0096	01B6
0004	01B7
0098	01B8
0099	01B9
009A	01BA
009B	01BB
0014	01BC
0015	01BD

009E	01BE
001A	01BF
009F	027F

AWS DMS 작업에 해당하는 입력 매핑:

```
{
  "SourceCharacterCodePoint": "0180", "TargetCharacterCodePoint": "0180"
}, {
  "SourceCharacterCodePoint": "0001", "TargetCharacterCodePoint": "0181"
}, {
  "SourceCharacterCodePoint": "0002", "TargetCharacterCodePoint": "0182"
}, {
  "SourceCharacterCodePoint": "0003", "TargetCharacterCodePoint": "0183"
}, {
  "SourceCharacterCodePoint": "009C", "TargetCharacterCodePoint": "0184"
}, {
  "SourceCharacterCodePoint": "0009", "TargetCharacterCodePoint": "0185"
}, {
  "SourceCharacterCodePoint": "0086", "TargetCharacterCodePoint": "0186"
}, {
  "SourceCharacterCodePoint": "007F", "TargetCharacterCodePoint": "0187"
}, {
  "SourceCharacterCodePoint": "0097", "TargetCharacterCodePoint": "0188"
}, {
  "SourceCharacterCodePoint": "008D", "TargetCharacterCodePoint": "0189"
}, {
  "SourceCharacterCodePoint": "008E", "TargetCharacterCodePoint": "018A"
}, {
  "SourceCharacterCodePoint": "000B", "TargetCharacterCodePoint": "018B"
}, {
  "SourceCharacterCodePoint": "000C", "TargetCharacterCodePoint": "018C"
}, {
  "SourceCharacterCodePoint": "000D", "TargetCharacterCodePoint": "018D"
}, {
  "SourceCharacterCodePoint": "000E", "TargetCharacterCodePoint": "018E"
}, {
  "SourceCharacterCodePoint": "000F", "TargetCharacterCodePoint": "018F"
}, {
  "SourceCharacterCodePoint": "0010", "TargetCharacterCodePoint": "0190"
}, {
  "SourceCharacterCodePoint": "0011", "TargetCharacterCodePoint": "0191"
}, {
  "SourceCharacterCodePoint": "0012", "TargetCharacterCodePoint": "0192"
}, {
  "SourceCharacterCodePoint": "0013", "TargetCharacterCodePoint": "0193"
}, {
  "SourceCharacterCodePoint": "009D", "TargetCharacterCodePoint": "0194"
}, {
  "SourceCharacterCodePoint": "0085", "TargetCharacterCodePoint": "0195"
}, {
  "SourceCharacterCodePoint": "0008", "TargetCharacterCodePoint": "0196"
}, {
  "SourceCharacterCodePoint": "0087", "TargetCharacterCodePoint": "0197"
}, {
  "SourceCharacterCodePoint": "0018", "TargetCharacterCodePoint": "0198"
}, {
  "SourceCharacterCodePoint": "0019", "TargetCharacterCodePoint": "0199"
}, {
  "SourceCharacterCodePoint": "0092", "TargetCharacterCodePoint": "019A"
}, {
  "SourceCharacterCodePoint": "008F", "TargetCharacterCodePoint": "019B"
}, {
  "SourceCharacterCodePoint": "001C", "TargetCharacterCodePoint": "019C"
}, {
  "SourceCharacterCodePoint": "001D", "TargetCharacterCodePoint": "019D"
}, {
  "SourceCharacterCodePoint": "001E", "TargetCharacterCodePoint": "019E"
}, {
  "SourceCharacterCodePoint": "001F", "TargetCharacterCodePoint": "019F"
}, {
  "SourceCharacterCodePoint": "0080", "TargetCharacterCodePoint": "01A0"
}, {
  "SourceCharacterCodePoint": "0081", "TargetCharacterCodePoint": "01A1"
}, {
  "SourceCharacterCodePoint": "0082", "TargetCharacterCodePoint": "01A2"
}, {
  "SourceCharacterCodePoint": "0083", "TargetCharacterCodePoint": "01A3"
}
```

```
, {"SourceCharacterCodePoint": "0084", "TargetCharacterCodePoint": "01A4"}
, {"SourceCharacterCodePoint": "000A", "TargetCharacterCodePoint": "01A5"}
, {"SourceCharacterCodePoint": "0017", "TargetCharacterCodePoint": "01A6"}
, {"SourceCharacterCodePoint": "001B", "TargetCharacterCodePoint": "01A7"}
, {"SourceCharacterCodePoint": "0088", "TargetCharacterCodePoint": "01A8"}
, {"SourceCharacterCodePoint": "0089", "TargetCharacterCodePoint": "01A9"}
, {"SourceCharacterCodePoint": "008A", "TargetCharacterCodePoint": "01AA"}
, {"SourceCharacterCodePoint": "008B", "TargetCharacterCodePoint": "01AB"}
, {"SourceCharacterCodePoint": "008C", "TargetCharacterCodePoint": "01AC"}
, {"SourceCharacterCodePoint": "0005", "TargetCharacterCodePoint": "01AD"}
, {"SourceCharacterCodePoint": "0006", "TargetCharacterCodePoint": "01AE"}
, {"SourceCharacterCodePoint": "0007", "TargetCharacterCodePoint": "01AF"}
, {"SourceCharacterCodePoint": "0090", "TargetCharacterCodePoint": "01B0"}
, {"SourceCharacterCodePoint": "0091", "TargetCharacterCodePoint": "01B1"}
, {"SourceCharacterCodePoint": "0016", "TargetCharacterCodePoint": "01B2"}
, {"SourceCharacterCodePoint": "0093", "TargetCharacterCodePoint": "01B3"}
, {"SourceCharacterCodePoint": "0094", "TargetCharacterCodePoint": "01B4"}
, {"SourceCharacterCodePoint": "0095", "TargetCharacterCodePoint": "01B5"}
, {"SourceCharacterCodePoint": "0096", "TargetCharacterCodePoint": "01B6"}
, {"SourceCharacterCodePoint": "0004", "TargetCharacterCodePoint": "01B7"}
, {"SourceCharacterCodePoint": "0098", "TargetCharacterCodePoint": "01B8"}
, {"SourceCharacterCodePoint": "0099", "TargetCharacterCodePoint": "01B9"}
, {"SourceCharacterCodePoint": "009A", "TargetCharacterCodePoint": "01BA"}
, {"SourceCharacterCodePoint": "009B", "TargetCharacterCodePoint": "01BB"}
, {"SourceCharacterCodePoint": "0014", "TargetCharacterCodePoint": "01BC"}
, {"SourceCharacterCodePoint": "0015", "TargetCharacterCodePoint": "01BD"}
, {"SourceCharacterCodePoint": "009E", "TargetCharacterCodePoint": "01BE"}
, {"SourceCharacterCodePoint": "001A", "TargetCharacterCodePoint": "01BF"}
, {"SourceCharacterCodePoint": "009F", "TargetCharacterCodePoint": "027F"}
```

마이그레이션에 적합한 대상

AWS Database Migration Service (AWS DMS)는 가장 많이 사용되는 데이터베이스를 데이터 복제의 대상으로 사용할 수 있습니다. 대상은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, Amazon Relational Database Service(RDS) 인스턴스 또는 온프레미스 데이터베이스에 있을 수 있습니다.

유효한 대상의 전체 목록은 [AWS DMS대상](#) 섹션을 참조하세요.

Note

AWS DMS 는 다음 대상 엔드포인트 유형에 대해 AWS 리전 간 마이그레이션을 지원하지 않습니다.

- Amazon DynamoDB
- Amazon OpenSearch Service
- Amazon Kinesis Data Streams

Amazon Aurora PostgreSQL Limitless는 AWS Database Migration Service ()의 대상으로 사용할 수 있습니다. 자세한 내용은 [PostgreSQL 데이터베이스를의 대상으로 사용을 참조하세요 AWS Database Migration Service](#).

주제

- [Oracle 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#)
- [Microsoft SQL Server 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#)
- [PostgreSQL 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#)
- [MySQL 호환 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#)
- [Amazon Redshift 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#)
- [SAP ASE 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#)
- [Amazon S3를의 대상으로 사용 AWS Database Migration Service](#)
- [Amazon DynamoDB 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#)
- [Amazon Kinesis Data Streams를의 대상으로 사용 AWS Database Migration Service](#)
- [Apache Kafka를의 대상으로 사용 AWS Database Migration Service](#)
- [Amazon OpenSearch Service 클러스터를의 대상으로 사용 AWS Database Migration Service](#)
- [Amazon DocumentDB를 AWS Database Migration Service의 대상으로 사용](#)
- [Amazon Neptune을의 대상으로 사용 AWS Database Migration Service](#)
- [Redis OSS를의 대상으로 사용 AWS Database Migration Service](#)
- [Babelfish를의 대상으로 사용 AWS Database Migration Service](#)
- [Amazon Timestream을의 대상으로 사용 AWS Database Migration Service](#)
- [Amazon RDS for Db2 및 IBM Db2 LUW를의 대상으로 사용 AWS DMS](#)

Oracle 데이터베이스를의 대상으로 사용 AWS Database Migration Service

다른 Oracle 데이터베이스 AWS DMS 또는 지원되는 다른 데이터베이스 중 하나를 사용하여 Oracle 데이터베이스 대상으로 데이터를 마이그레이션할 수 있습니다. Secure Sockets Layer(SSL)을 사용하여 Oracle 엔드포인트와 복제 인스턴스 연결을 암호화할 수 있습니다. Oracle 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 섹션을 참조하세요 [에서 SSL 사용 AWS Database Migration Service](#). AWS DMS Oracle TDE는 데이터베이스에 쓰는 데 암호화 키나 암호가 필요하지 않기 때문에 대상 데이터베이스의 저장 데이터를 암호화하기 위해 Oracle 투명 데이터 암호화(TDE) 사용을 지원합니다.

가 대상으로 AWS DMS 지원하는 Oracle 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 대상 AWS DMS](#).

Oracle을 대상으로 사용할 때는 데이터를 대상 연결에 사용되는 스키마 또는 사용자로 마이그레이션해야 한다고 가정합니다. 데이터를 다른 스키마로 마이그레이션하려면 스키마 변환을 사용해야 합니다. 예를 들어, 대상 엔드포인트가 사용자 RDSMASTER에 연결되고 사용자 PERFDATA1에서 PERFDATA2로 마이그레이션하려 한다고 가정합니다. 이 경우, 다음과 같이 변환을 생성합니다.

```
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "rename",
  "rule-target": "schema",
  "object-locator": {
    "schema-name": "PERFDATA1"
  },
  "value": "PERFDATA2"
}
```

Oracle을 대상으로 사용하는 경우는 모든 테이블과 인덱스를 대상의 기본 테이블 및 인덱스 테이블스페이스로 AWS DMS 마이그레이션합니다. 테이블과 인덱스를 다른 테이블 및 인덱스 테이블스페이스로 마이그레이션하려면 테이블스페이스 변환을 사용해야 합니다. 예를 들어, Oracle 소스의 일부 테이블스페이스에 할당된 INVENTORY 스키마에 테이블 집합이 있다고 가정해 보겠습니다. 마이그레이션을 위해 이러한 모든 테이블을 대상의 단일 INVENTORYSPACE 테이블스페이스에 할당하려고 합니다. 이 경우, 다음과 같이 변환을 생성합니다.

```
{
```

```

"rule-type": "transformation",
"rule-id": "3",
"rule-name": "3",
"rule-action": "rename",
"rule-target": "table-tablespace",
"object-locator": {
  "schema-name": "INVENTORY",
  "table-name": "%",
  "table-tablespace-name": "%"
},
"value": "INVENTORYSPACE"
}

```

변환에 관한 자세한 내용은 [JSON을 사용하여 테이블 선택 및 변환 지정](#) 섹션을 참조하세요.

Oracle이 소스와 대상인 경우, Oracle 소스 추가 접속 속성인 `enableHomogenousTablespace=true`를 설정하여 기존 테이블 또는 인덱스 테이블스페이스 할당을 유지할 수 있습니다. 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요.

Oracle 데이터베이스를 대상으로 사용하는 방법에 대한 자세한 내용은 다음 섹션을 AWS DMS참조하세요.

주제

- [의 대상으로서 Oracle에 대한 제한 사항 AWS Database Migration Service](#)
- [Oracle을 대상으로 사용할 때 필요한 사용자 계정 권한](#)
- [Oracle 데이터베이스를의 대상으로 구성 AWS Database Migration Service](#)
- [Oracle을의 대상으로 사용할 때 엔드포인트 설정 AWS DMS](#)
- [Oracle용 대상 데이터 형식](#)

의 대상으로서 Oracle에 대한 제한 사항 AWS Database Migration Service

데이터 마이그레이션에서 Oracle 데이터베이스를 대상으로 사용할 때 적용되는 제한 사항에는 다음이 있습니다.

- AWS DMS 는 대상 Oracle 데이터베이스에 스키마를 생성하지 않습니다. 대상 Oracle 데이터베이스에서 필요한 스키마를 생성해야 합니다. Oracle 대상에서 스키마 이름이 이미 있어야 합니다. 소스 스키마의 테이블은 AWS DMS 사용하여 대상 인스턴스에 연결하는 사용자 또는 스키마로 가져옵니다. 여러 스키마를 마이그레이션하려는 경우, 다중 복제 작업을 생성하면 됩니다. 데이터를 대상의

다른 스키마로 마이그레이션할 수도 있습니다. 이렇게 하려면 AWS DMS 테이블 매핑에 스키마 변환 규칙을 사용해야 합니다.

- AWS DMS 는 INDEXTYPE CONTEXT가 있는 테이블에 대한 Use direct path full load 옵션을 지원하지 않습니다. 차선택으로서 어레이 로드를 사용할 수 있습니다.
- 배치 작업에 최적화된 적용 옵션을 사용하면 넷 변경 테이블에 로드할 때 XML 유형을 지원하지 않는 직접 경로가 사용됩니다. 차선택으로서 트랜잭션 적용 모드를 사용할 수 있습니다.
- 소스 데이터베이스에서 마이그레이션된 빈 문자열은 Oracle 대상에서 다르게 처리될 수 있습니다 (예: 하나의 공백 문자열로 변환). 이로 인해 AWS DMS 검증에서 불일치가 보고될 수 있습니다.
- 다음 공식을 사용하여 Batch 최적화 적용 모드에서 지원되는 테이블당 총 열 수를 표현할 수 있습니다.

$$2 * \text{columns\_in\_original\_table} + \text{columns\_in\_primary\_key} \leq 999$$

예를 들어 원본 테이블에 25개의 열이 있고 프라이머리 키가 5개의 열로 구성된 경우, 총 열 수는 55 개입니다. 테이블이 지원되는 열 수를 초과하는 경우, 모든 변경 내용은 1대1 모드로 적용됩니다.

- AWS DMS 는 Oracle 클라우드 인프라(OCI)의 자율 DB를 지원하지 않습니다.
- 트랜잭션 적용 모드에서 Oracle 대상은 최대 32KB 크기의 DML 문을 처리할 수 있습니다. 이 제한은 많은 사용 사례에 충분하지만 32KB를 초과하는 DML 문은 "ORA-01460: 구현되지 않았거나 불합리한 변환이 요청됨"이라는 오류와 함께 실패합니다. 이 문제를 해결하려면 BatchApplyEnabled 작업 설정을 true로 설정하여 배치 적용 기능을 활성화해야 합니다. 배치 적용은 전체 문 크기를 줄여 32KB 제한을 우회할 수 있습니다. 자세한 내용은 [the section called “대상 메타데이터 작업 설정”](#) 단원을 참조하십시오.
- AWS DMS LOB 테이블에 대한 직접 경로 전체 로드는 LOB 데이터에 대한 특수 처리 요구 사항으로 인해 ORA-39777 오류와 함께 실패할 수 있습니다. 이 오류는 직접 경로 로드 프로세스 중에 발생하며 LOB 열과 관련된 마이그레이션 작업을 방해할 수 있습니다. 해결하려면 대상 엔드포인트에서 useDirectPathFullLoad 설정을 비활성화하고 로드 작업을 다시 시도합니다.

Oracle을 대상으로 사용할 때 필요한 사용자 계정 권한

AWS Database Migration Service 작업에서 Oracle 대상을 사용하려면 Oracle 데이터베이스에서 다음 권한을 부여합니다. 이러한 권한은 AWS DMS의 Oracle 데이터베이스 정의에서 지정된 사용자 계정에 부여합니다.

- SELECT ANY TRANSACTION
- SELECT on V\$NLS_PARAMETERS

- SELECT on V\$TIMEZONE_NAMES
- SELECT on ALL_INDEXES
- SELECT on ALL_OBJECTS
- SELECT on DBA_OBJECTS
- SELECT on ALL_TABLES
- SELECT on ALL_USERS
- SELECT on ALL_CATALOG
- SELECT on ALL_CONSTRAINTS
- SELECT on ALL_CONS_COLUMNS
- SELECT on ALL_TAB_COLS
- SELECT on ALL_IND_COLUMNS
- DROP ANY TABLE
- SELECT ANY TABLE
- INSERT ANY TABLE
- UPDATE ANY TABLE
- CREATE ANY VIEW
- DROP ANY VIEW
- CREATE ANY PROCEDURE
- ALTER ANY PROCEDURE
- DROP ANY PROCEDURE
- CREATE ANY SEQUENCE
- ALTER ANY SEQUENCE
- DROP ANY SEQUENCE
- DELETE ANY TABLE

다음과 같은 요구 사항에서 이러한 추가 권한을 부여합니다.

- 특정 테이블 목록을 사용하려면 SELECT를 복제된 테이블에 부여하고 ALTER도 복제된 테이블에 부여합니다.
- 사용자가 기본 테이블스페이스에서 테이블을 생성하도록 허용하려면 권한 GRANT UNLIMITED TABLESPACE를 부여합니다.

- 로그인된 경우, 권한 CREATE SESSION을 부여합니다.
- 직접 경로(전체 로드 기본값)를 사용하는 경우 GRANT LOCK ANY TABLE to *dms_user*;을 실행합니다.
- “DROP and CREATE” 테이블 준비 모드를 사용할 때 스키마가 다른 경우, GRANT CREATE ANY INDEX to *dms_user*;를 실행합니다.
- 일부 전체 로드 시나리오의 경우, 대상 테이블 스키마가 DMS 사용자의 스키마와 다른 “테이블 DROP 및 CREATE” 또는 “로딩 전 TRUNCATE” 옵션을 선택할 수 있습니다. 이 경우, DROP ANY TABLE을 부여합니다.
- 대상 테이블 스키마가 DMS 사용자의 테이블 스키마와 다를 때 변경 테이블이나 감사 테이블에 변경 사항을 저장하려면 CREATE ANY TABLE 및 CREATE ANY INDEX를 부여합니다.
- 검증 기능으로 LOB 열을 검증하려면 SYS.DBMS_CRYPTO DMS 사용자에게 EXECUTE 권한을 부여합니다.

대상 데이터베이스의 AWS Database Migration Service 에 필요한 읽기 권한

다음 DBA 테이블에 대한 읽기 권한이 AWS DMS 사용자 계정에 부여되어야 합니다.

- SELECT on DBA_USERS
- SELECT on DBA_TAB_PRIVS
- SELECT on DBA_OBJECTS
- SELECT on DBA_SYNONYMS
- SELECT on DBA_SEQUENCES
- SELECT on DBA_TYPES
- SELECT on DBA_INDEXES
- SELECT on DBA_TABLES
- SELECT on DBA_TRIGGERS
- SELECT on SYS.DBA_REGISTRY

필요한 권한을 V\$xxx에 부여할 수 없는 경우, V_\$xxx에 부여합니다.

마이그레이션 전 평가

Oracle을 대상으로 [Oracle 평가](#)에 나열된 마이그레이션 전 평가를 사용하려면 Oracle 데이터베이스 대상 엔드포인트에 명시된 사용자 계정에 다음 권한을 추가해야 합니다.

```
GRANT SELECT ON V_$INSTANCE TO dms_user;
GRANT EXECUTE ON SYS.DBMS_XMLGEN TO dms_user;
```

Oracle 데이터베이스를 대상으로 구성 AWS Database Migration Service

Oracle 데이터베이스를 데이터 마이그레이션 대상으로 사용하기 전에 Oracle 사용자 계정을 제공해야 합니다. AWS DMS 사용자 계정에는 [Oracle을 대상으로 사용할 때 필요한 사용자 계정 권한](#)에 지정된 바와 같이 Oracle 데이터베이스에서 읽기/쓰기 권한이 있어야 합니다.

Oracle을 대상으로 사용할 때 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Oracle 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 `--oracle-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Oracle을 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

이름	설명
UseDirectPathFullLoad	로 설정하면 직접 경로 전체 로드 Y AWS DMS 사용합니다. Oracle 호출 인터페이스(OCI)에서 직접 경로 프로토콜을 활성화하려면 이 값을 지정합니다. 이 OCI 프로토콜을 사용하면 전체 로드 중에 Oracle 대상 테이블을 대량 로드할 수 있습니다. 기본값: true 유효값: true/false 예시: <code>--oracle-settings '{"UseDirectPathFullLoad": false}'</code>
DirectPathParallelLoad	true로 설정하면 UseDirectPathFullLoad 가 Y로 설정된 경우 이 속성이 병렬 로드를 지정합니다. 이 속성은 AWS DMS 병렬 로드 기능을 사용하는 경우에만 적용됩니다. 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업의 parallel-load 작업에 관한 설명을 참조하세요.

이름	설명
	이 병렬 로드 설정을 지정할 때 제한 사항은 대상 테이블에 제약 조건이나 인덱스를 가질 수 없다는 것입니다. 이 제한 사항에 관한 자세한 내용은 병렬 직접 경로 로드 후 제약 조건 활성화를 참조하세요. 제약 조건 또는 인덱스가 활성화된 경우, 이 속성을 true로 설정해도 아무 효과가 없습니다. 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"DirectPathParallelLoad": true}'</code>
DirectPathNoLog	이 속성을 true로 설정하면 테이블에 직접 쓰고 데이터베이스 로그에 추적을 쓰지 않고 Oracle 대상 데이터베이스의 커밋 비율을 높일 수 있습니다. 자세한 내용은 Direct-Load INSERT를 참조하세요. 이 속성은 UseDirectPathFullLoad 를 Y로 설정한 경우에만 적용됩니다. 기본값: false 유효값: true/false 예시: <code>--oracle-settings '{"DirectPathNoLog": true}'</code>
CharLengthSemantics	문자 열의 길이가 바이트 단위인지 또는 문자 단위인지 지정합니다. 문자 열 길이가 문자 단위임을 나타내려면 이 속성을 CHAR로 설정합니다. 그렇지 않으면 문자 열 길이는 바이트 단위입니다. 기본값: CHAR로 설정되지 않음 유효값: CHAR 예시: <code>--oracle-settings '{"CharLengthSemantics": "CHAR"}'</code>

이름	설명
AlwaysReplaceEmptyString	AWS DMS 는 Oracle 대상으로 마이그레이션할 때 빈 문자열을 복제하기 위해 추가 공간을 추가합니다. 일반적으로 Oracle에는 빈 문자열에 대한 표기법이 없습니다. varchar2에 빈 문자열을 삽입하면 빈 문자열이 NULL로 로드됩니다. Oracle에서 데이터를 NULL로 삽입하려면 이 속성을 FALSE로 설정하세요. 기본값: true 유효값: true/false 예시: --oracle-settings '{"AlwaysReplaceEmptyString": false}'

Oracle용 대상 데이터 형식

와 함께 사용되는 대상 Oracle 데이터베이스는 대부분의 Oracle 데이터 유형을 AWS DMS 지원합니다. 다음 표에는를 사용할 때 지원되는 Oracle 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다. 소스에서 매핑된 데이터 형식을 확인하는 방법에 관한 자세한 내용은 사용 중인 소스에 대한 섹션을 참조하세요.

AWS DMS 데이터 유형	Oracle 데이터 형식
BOOLEAN	NUMBER (1)
BYTES	RAW(길이)
DATE	DATETIME
TIME	TIMESTAMP (0)
DATETIME	TIMESTAMP(크기)
INT1	NUMBER (3)
INT2	NUMBER (5)

AWS DMS 데이터 유형	Oracle 데이터 형식
INT4	NUMBER (10)
INT8	NUMBER (19)
NUMERIC	NUMBER (p,s)
REAL4	FLOAT
REAL8	FLOAT
STRING	날짜 표시: DATE 포함 시간 표시: TIMESTAMP 포함 타임스탬프 표시: TIMESTAMP 포함 timestamp_with_timezone 표시: TIMESTAMP WITH TIMEZONE 포함 timestamp_with_local_timezone 표시: TIMESTAMP WITH LOCAL TIMEZONE 포함 interval_year_to_month 표시: INTERVAL YEAR TO MONTH 포함 interval_day_to_second 표시: INTERVAL DAY TO SECOND 포함 길이 > 4000인 경우: CLOB 다른 모든 경우: VARCHAR2(길이)
UINT1	NUMBER (3)
UINT2	NUMBER (5)
UINT4	NUMBER (10)
UINT8	NUMBER (19)
WSTRING	길이가 > 2000: NCLOB인 경우 다른 모든 경우: NVARCHAR2(길이)

AWS DMS 데이터 유형	Oracle 데이터 형식
BLOB	BLOB 이 데이터 형식과 함께 사용하려면 특정 작업에 대해 BLOBs 사용을 활성화 AWS DMS해야 합니다. BLOB 데이터 형식은 프라이머리 키를 포함하는 테이블에서만 지원됩니다.
CLOB	CLOB 이 데이터 형식과 함께 사용하려면 특정 작업에 대해 CLOBs 사용을 활성화 AWS DMS해야 합니다. 변경 데이터 캡처(CDC) 중에 프라이머리 키를 포함하는 테이블에서만 CLOB 데이터 형식이 지원됩니다. STRING 선언된 크기가 4,000바이트를 초과하는 소스의 Oracle VARCHAR2 데이터 형식은 AWS DMS CLOB를 통해 Oracle 대상의 STRING에 매핑됩니다.
NCLOB	NCLOB 이 데이터 형식과 함께 사용하려면 특정 작업에 대해 NCLOBs 사용을 활성화 AWS DMS해야 합니다. CDC 중에 NCLOB 데이터 형식은 프라이머리 키를 포함하는 테이블에서만 지원됩니다. WSTRING 선언된 크기가 4,000바이트보다 큰 소스의 Oracle VARCHAR2 데이터 형식은 AWS DMS NCLOB를 통해 Oracle 대상의 WSTRING으로 매핑됩니다.
XMLTYPE	XMLTYPE 대상 데이터 형식은 Oracle 간 복제 작업에서만 관련이 있습니다. 소스 데이터베이스가 Oracle이면 소스 데이터 형식은 Oracle 대상에 '그대로' 복제됩니다. 예를 들어, 원본의 XMLTYPE 데이터 형식은 대상에서 XMLTYPE 데이터 형식으로 생성됩니다.

Microsoft SQL Server 데이터베이스를의 대상으로 사용 AWS Database Migration Service

를 사용하여 Microsoft SQL Server 데이터베이스로 데이터를 마이그레이션할 수 있습니다 AWS DMS. SQL Server 데이터베이스를 대상으로 사용할 때에는 다른 SQL Server 데이터베이스나 지원되는 다른 데이터베이스 중 하나에서 가져온 데이터베이스를 마이그레이션할 수 있습니다.

가 대상으로 AWS DMS 지원하는 SQL Server 버전에 대한 자세한 내용은 [섹션을 참조하세요](#) [의 대상 AWS DMS](#).

AWS DMS 는 Enterprise, Standard, Workgroup 및 Developer의 온프레미스 및 Amazon RDS 에디션 을 지원합니다.

AWS DMS 및 SQL Server 대상 데이터베이스 작업에 대한 자세한 내용은 다음을 참조하세요.

주제

- [SQL Server를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service](#)
- [SQL Server를의 대상으로 사용할 때의 보안 요구 사항 AWS Database Migration Service](#)
- [SQL Server를의 대상으로 사용할 때 엔드포인트 설정 AWS DMS](#)
- [Microsoft SQL Server용 대상 데이터 형식](#)

SQL Server를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

다음 제한 사항은 SQL Server 데이터베이스를 AWS DMS의 대상으로 사용 시 적용됩니다.

- 계산된 열에서 SQL Server 대상 테이블을 수동으로 생성하면, BCP 대량 복사 유틸리티를 사용할 때 전체 로드 복제가 지원되지 않습니다. 전체 로드 복제를 사용하려면 엔드포인트에서 추가 연결 속성 (ECA) 'useBCPFullLoad=false' 를 설정하여 BCP 로드를 비활성화하세요. 엔드포인트에서 ECA 설정에 관한 자세한 내용은 [소스 및 대상 엔드포인트 생성](#) 섹션을 참조하세요. BCP 사용에 관한 자세한 내용은 [Microsoft SQL Server 설명서](#)를 참조하세요.
- SQL Server 공간 데이터 형식(GEOMETRY 및 GEOGRAPHY)으로 테이블을 복제할 때는 기본 SRID로 삽입했을 수 있는 공간 참조 식별자(SRID)를 AWS DMS 대체합니다. GEOMETRY에서 기본 SRID는 0이고 GEOGRAPHY에서는 4326입니다.
- 임시 테이블은 지원되지 않습니다. 임시 테이블 마이그레이션은 이 테이블이 대상에서 수동으로 생성되지 않는 경우 Transactional Apply 모드에서 복제 전용 작업을 통해 수행할 수도 있습니다.
- 현재 PostgreSQL 소스의 boolean 데이터 형식은 값이 일치하지 않는 bit 데이터 형식으로 SQLServer 대상에 마이그레이션됩니다.

차선책으로 다음을 수행합니다.

- 열에 대한 VARCHAR(1) 데이터 유형으로 테이블을 미리 생성합니다(또는 테이블을 AWS DMS 생성하도록 합니다). 그런 다음, 다운스트림 처리에서 'F'를 False로 처리하고 'T'를 True로 처리합니다.
- 다운스트림 처리를 변경할 필요가 없도록 작업에 변환 규칙을 추가하여 'F' 값을 '0'으로 변경하고 'T' 값을 1로 변경한 다음 이를 SQL 서버 비트 데이터 형식으로 저장하세요.
- AWS DMS 는 열 null성을 설정하기 위한 변경 처리를 지원하지 않습니다(ALTER TABLE문과 함께 ALTER COLUMN [SET|DROP] NOT NULL 절 사용).
- Windows 인증은 지원되지 않습니다.

SQL Server를 대상으로 사용할 때의 보안 요구 사항 AWS Database Migration Service

다음은 Microsoft SQL Server 대상과 AWS DMS 함께를 사용하기 위한 보안 요구 사항을 설명합니다.

- AWS DMS 사용자 계정에는 연결하려는 SQL Server 데이터베이스의 db_owner 사용자 역할이 있어야 합니다.
- SQL Server 시스템 관리자는 모든 AWS DMS 사용자 계정에 이 권한을 부여해야 합니다.

SQL Server를 대상으로 사용할 때 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 SQL Server 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --microsoft-sql-server-settings '{"EndpointSetting": "value", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

SQL Server를 대상으로 할 때 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

이름	설명
ControlTablesFileGroup	AWS DMS 내부 테이블의 파일 그룹을 지정합니다. 복제 작업이 시작되면 모든 내부 AWS DMS 제어 테이블(awsdms_apply_exception, awsdms_apply, awsdms_changes)이 지정된 파일 그룹에 생성됩니다. 기본값: 해당 사항 없음

이름	설명
	유효값: 문자열 예시: --microsoft-sql-server-settings '{"ControlTablesFileGroup": "filegroup1"}' 다음은 파일 그룹을 생성한 명령의 예입니다. <pre>ALTER DATABASE replicate ADD FILEGROUP Test1FG1; GO ALTER DATABASE replicate ADD FILE (NAME = test1dat5, FILENAME = 'C:\temp\DATA\t1dat5.ndf', SIZE = 5MB, MAXSIZE = 100MB, FILEGROWTH = 5MB) TO FILEGROUP Test1FG1; GO</pre>
ConnectionTimeout	이 추가 연결 속성(ECA)을 사용하여 SQL Server 인스턴스의 엔드포인트 연결 제한 시간을 초 단위로 설정합니다. 기본값은 10초입니다. ECA 예제: ConnectionTimeout= 30 .
ExecuteTimeout	이 추가 연결 속성(ECA)을 사용하여 SQL Server 인스턴스의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값은 60초입니다. 예시: '{"ExecuteTimeout": 100}'

이름	설명
UseBCPFullLoad	이 속성을 사용하여 BCP를 사용한 전체 로드 작업에서 데이터를 전송합니다. 소스 테이블에 없는 자격 증명 열이 대상 테이블에 포함되어 있다면 use BCP for loading table(테이블 로드 시 BCP 사용) 옵션을 비활성화해야 합니다. 기본값: true 유효값: true/false 예시: --microsoft-sql-server-settings '{"UseBCPFullLoad": false}'

Microsoft SQL Server용 대상 데이터 형식

다음 표에는를 사용할 때 지원되는 Microsoft SQL Server 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다. AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

AWS DMS 데이터 유형	SQL Server 데이터 형식
BOOLEAN	TINYINT
BYTES	VARBINARY(길이)
DATE	SQL Server 2008 및 이후 버전에서는 DATE를 사용합니다. 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에는 VARCHAR(37)을 사용합니다.
TIME	SQL Server 2008 및 이후 버전에서는 DATETIME2(%d)를 사용합니다. 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에는 VARCHAR(37)을 사용합니다.

AWS DMS 데이터 유형	SQL Server 데이터 형식
DATETIME	SQL Server 2008 및 이후 버전에서는 DATETIME2(크기)를 사용합니다. 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에는 VARCHAR(37)을 사용합니다.
INT1	SMALLINT
INT2	SMALLINT
INT4	INT
INT8	BIGINT
NUMERIC	NUMERIC(p,s)
REAL4	REAL
REAL8	FLOAT
STRING	열이 날짜 또는 시간 열인 경우, 다음 작업을 수행합니다. • SQL Server 2008 이상에서는 DATETIME2를 사용합니다. • 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에는 VARCHAR(37)을 사용합니다. 열이 날짜 또는 시간 열이 아닌 경우, VARCHAR(길이)를 사용합니다.
UINT1	TINYINT
UINT2	SMALLINT
UINT4	INT
UINT8	BIGINT
WSTRING	NVARCHAR(길이)

AWS DMS 데이터 유형	SQL Server 데이터 형식
BLOB	VARBINARY(최대) IMAGE 이 데이터 형식들과 함께 사용하려면 특정 작업에 대해 BLOBs 사용을 활성화 AWS DMS해야 합니다.는 기본 키가 포함된 테이블에서만 BLOB 데이터 형식을 AWS DMS 지원합니다.
CLOB	VARCHAR(최대) 이 데이터 형식들과 함께 사용하려면 특정 작업에 대해 CLOBs 사용을 활성화 AWS DMS해야 합니다. 변경 데이터 캡처(CDC) 중에 AWS DMS 는 프라이머리 키를 포함하는 데이터에서만 CLOB 데이터 형식을 지원합니다.
NCLOB	NVARCHAR(최대) 이 데이터 형식들과 함께 사용하려면 특정 작업에 대해 NCLOBs 사용을 활성화 AWS DMS해야 합니다. CDC 중에는 기본 키가 포함된 테이블에서만 NCLOB 데이터 형식을 AWS DMS 지원합니다.

PostgreSQL 데이터베이스를 대상으로 사용 AWS Database Migration Service

다른 PostgreSQL 데이터베이스 AWS DMS또는 지원되는 다른 데이터베이스 중 하나를 사용하여 PostgreSQL 데이터베이스로 데이터를 마이그레이션할 수 있습니다.

가 대상으로 AWS DMS 지원하는 PostgreSQL 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 대상 AWS DMS](#).

Note

- Amazon Aurora Serverless는 PostgreSQL과 호환되는 Amazon Aurora의 대상으로 사용할 수 있습니다. Amazon Aurora Serverless에 관한 자세한 내용을 알아보려면 Amazon Aurora 사용 설명서의 [Amazon Aurora Serverless v2 사용](#)을 참조하세요.

- Aurora Serverless DB 클러스터는 Amazon VPC에서만 액세스할 수 있으며 [퍼블릭 IP 주소](#)를 사용할 수 없습니다. 따라서 Aurora PostgreSQL Serverless가 아닌 다른 지역에 복제 인스턴스를 사용하려는 경우, [vpc 피어링](#)을 구성해야 합니다. 그렇지 않으면 Aurora PostgreSQL 서버리스 [지역](#)의 가용성을 확인하고 Aurora PostgreSQL 서버리스와 복제 인스턴스 모두에 해당 지역 중 하나를 사용하기로 결정하세요.
- Babelfish 기능은 Amazon Aurora에 내장되어 있으며 추가 비용이 들지 않습니다. 자세한 내용은 [AWS Database Migration Service의 대상으로 Babelfish for Aurora PostgreSQL 사용](#)을 참조하세요.

AWS DMS 는 전체 로드 단계에서 소스에서 대상으로 데이터를 마이그레이션할 때 table-by-table 접근 방식을 사용합니다. 전체 로드 단계 시 테이블 순서를 보장할 수 없습니다. 일반적으로 전체 로드 단계가 수행되는 동안과 개별 테이블의 캐시된 트랜잭션이 적용되는 동안 테이블은 동기화되지 않습니다. 결과적으로 활성 참조 무결성 제약으로 인해 전체 로드 단계 중에 작업 실패가 발생합니다.

PostgreSQL에서는 트리거를 사용하여 외래 키(참조 무결성 제약)가 구현됩니다. 전체 로드 단계에서는 각 테이블을 한 번에 하나씩 AWS DMS 로드합니다. 다음 방법 중 하나를 사용하여 전체 로드 동안 외래 키 제약을 비활성화하는 것이 좋습니다.

- 인스턴스에서 모든 트리거를 임시로 비활성화한 후 전체 로드를 완료합니다.
- PostgreSQL에서 `session_replication_role` 파라미터를 사용합니다.

해당 시점에 트리거는 `origin`, `replica`, `always` 또는 `disabled` 상태 중 하나일 수 있습니다. `session_replication_role` 파라미터가 `replica`로 설정되는 경우, `replica` 상태의 트리거만 활성 상태이며 호출될 때 트리거됩니다. 그렇지 않으면, 트리거가 비활성 상태로 유지됩니다.

PostgreSQL에는 `session_replication_role`이 설정되었더라도 테이블 잠금을 방지하기 위한 안전 메커니즘이 있습니다. 전체 로드 실행이 완료될 수 있도록 이 메커니즘을 대안으로 사용하여 트리거를 비활성화할 수 있습니다. 이렇게 하려면 대상 테이블 준비 모드를 `DO_NOTHING`으로 설정합니다. 그렇지 않으면 외래 키 제약이 있는 경우 `DROP` 및 `TRUNCATE` 작업이 실패합니다.

Amazon RDS에서는 파라미터 그룹을 사용하여 이 파라미터 설정을 제어할 수 있습니다. Amazon EC2에서 실행 중인 PostgreSQL 인스턴스의 경우, 파라미터를 직접 설정할 수 있습니다.

PostgreSQL 데이터베이스를 대상으로 사용하는 방법에 대한 자세한 내용은 다음 섹션을 AWS DMS참조하세요.

주제

- [PostgreSQL을의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service](#)
- [Amazon Aurora PostgreSQL Limitless를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service](#)
- [PostgreSQL 데이터베이스를의 대상으로 사용할 때의 보안 요구 사항 AWS Database Migration Service](#)
- [PostgreSQL을의 대상으로 사용할 때 엔드포인트 설정 및 추가 연결 속성\(ECAs\) AWS DMS](#)
- [PostgreSQL용 대상 데이터 형식](#)
- [Babelfish for Aurora PostgreSQL을의 대상으로 사용 AWS Database Migration Service](#)

PostgreSQL을의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

다음 제한 사항은 AWS DMS에서 PostgreSQL 데이터베이스를 대상으로 사용할 때 적용됩니다.

- 이기종 마이그레이션의 경우 내부적으로 JSON 데이터 형식이 네이티브 CLOB 데이터 형식으로 변환됩니다.
- Oracle에서 PostgreSQL로 마이그레이션할 때 Oracle의 열에 NULL 문자(16진수 값 U+0000)가 포함된 경우는 NULL 문자를 공백(16진수 값 U+0020)으로 AWS DMS 변환합니다. 이는 PostgreSQL 제한 사항 때문입니다.
- AWS DMS 는 coalesce 함수로 생성된 고유 인덱스가 있는 테이블에 대한 복제를 지원하지 않습니다.
- 테이블이 시퀀스를 사용하는 경우 소스 데이터베이스에서 복제를 중지한 후 대상 데이터베이스의 각 시퀀스에 NEXTVAL 대한 값을 업데이트합니다.는 소스 데이터베이스의 데이터를 AWS DMS 복사하지만, 지속적 복제 중에는 시퀀스를 대상으로 마이그레이션하지 않습니다.

Amazon Aurora PostgreSQL Limitless를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

Amazon Aurora PostgreSQL Limitless를 대상으로 사용할 때는 AWS DMS다음과 같은 제한 사항이 적용됩니다.

- AWS DMS 데이터 검증은 Amazon Aurora PostgreSQL Limitless를 지원하지 않습니다.
- AWS DMS 는 소스 테이블을 분산되지 않는 표준 테이블로 마이그레이션합니다. 마이그레이션 후 공식 변환 가이드에 따라 이러한 표준 테이블을 Limitless 테이블로 변환할 수 있습니다.

PostgreSQL 데이터베이스를의 대상으로 사용할 때의 보안 요구 사항 AWS Database Migration Service

보안상의 목적으로 데이터 마이그레이션에 사용되는 사용자 계정은 대상으로 사용하는 모든 PostgreSQL 데이터베이스에서 등록된 사용자여야 합니다.

AWS DMS 마이그레이션을 실행하려면 PostgreSQL 대상 엔드포인트에 최소 사용자 권한이 필요합니다. 다음 예제를 참조하세요.

```
CREATE USER newuser WITH PASSWORD 'your-password';
ALTER SCHEMA schema_name OWNER TO newuser;
```

또는

```
GRANT USAGE ON SCHEMA schema_name TO myuser;
GRANT CONNECT ON DATABASE postgres TO myuser;
GRANT CREATE ON DATABASE postgres TO myuser;
GRANT CREATE ON SCHEMA schema_name TO myuser;
GRANT UPDATE, INSERT, SELECT, DELETE, TRUNCATE ON ALL TABLES IN SCHEMA schema_name
TO myuser;
GRANT TRUNCATE ON schema_name."BasicFeed" TO myuser;
```

PostgreSQL을의 대상으로 사용할 때 엔드포인트 설정 및 추가 연결 속성(ECAs) AWS DMS

엔드포인트 설정 및 추가 연결 속성(ECA)을 사용하여 PostgreSQL 대상 데이터베이스를 구성할 수 있습니다.

AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --postgre-sql-settings '{"*EndpointSetting*': "*value*", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

엔드포인트의 ExtraConnectionAttributes 파라미터를 사용하여 ECA를 지정합니다.

PostgreSQL을 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

이름	설명
ConnectionTimeout	이 추가 연결 속성(ECA)을 사용하여 PostgreSQL 인스턴스의 엔드포인트 연결 제한 시간을 초 단위로 설정합니다. 기본값은 10초입니다. ECA 예제: ConnectionTimeout=30 .
MaxFileSize	데이터를 PostgreSQL로 전송할 때 사용되는 .csv 파일의 최대 크기(단위: KB)를 지정합니다. 기본값: 32,768KB(32MB) 유효값: 1~1,048,576KB(최대 1.1GB) 예시: --postgre-sql-settings '{"MaxFileSize": 512}'
ExecuteTimeout	PostgreSQL 인스턴스의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값은 60초입니다. 예시: --postgre-sql-settings '{"ExecuteTimeout": 100}'
AfterConnectScript= SET session_replication_role = replica	이 속성에는 외부 키 및 사용자 트리거를 AWS DMS 우회하여 데이터를 대량 로드하는 데 걸리는 시간을 줄입니다.
MapUnboundedNumericAsString	이 파라미터는 숫자 값의 정밀도를 유지하면서 성공적으로 마이그레이션할 수 있도록 제한이 없는 숫자 데이터 형식의 열을 문자열로 취급합니다. 이 파라미터는 PostgreSQL 소스에서 PostgreSQL 대상 또는 PostgreSQL과 호환되는 데이터베이스로 복제하는 경우에만 사용하세요. 기본값: false 유효값: false/true 예시: --postgre-sql-settings '{"MapUnboundedNumericAsString": "true"}

이름	설명
	이 파라미터를 사용하면 숫자에서 문자열로, 다시 숫자로 변환되므로 복제 성능이 일부 저하될 수 있습니다. 이 파라미터는 DMS 버전 3.4.4 이상에서 사용할 수 있습니다.  Note 단지 MapUnboundedNumericAsString 을 PostgreSQL 소스 및 대상 엔드포인트에서 함께 사용해야 합니다. 소스 PostgreSQL 엔드포인트에서 MapUnboundedNumericAsString 을 사용하면 CDC 중에 정밀도가 28로 제한됩니다. 대상 엔드포인트에서 MapUnboundedNumericAsString 을 사용하면 정밀도 28, 스케일 6으로 데이터를 마이그레이션할 수 있습니다. PostgreSQL이 아닌 대상에 MapUnboundedNumericAsString 을 사용하지 마세요.
loadUsingCSV	이 추가 연결 속성(ECA)을 사용하여 \COPY 명령을 사용한 전체 로드 작업에서 데이터를 전송합니다. 기본값: true 유효값: true/false ECA 예제: loadUsingCSV=true; 참고: 이 ECA를 false로 설정하면 INSERT가 직접 실행되기 때문에 복제 성능이 일부 저하될 수 있습니다.

이름	설명
DatabaseMode	이 속성을 사용하면 Babelfish 엔드포인트와 같이 일부 추가 구성이 필요한 PostgreSQL 호환 엔드포인트에 대한 복제 처리의 기본 동작을 변경할 수 있습니다. 기본값: DEFAULT 유효값: DEFAULT, BABELFISH 예시: DatabaseMode=default;
BabelfishDatabaseName	이 속성을 사용하여 마이그레이션할 대상 Babelfish T-SQL 데이터베이스의 이름을 지정할 수 있습니다. DatabaseMode 가 Babelfish 로 설정된 경우 이 데이터베이스는 필수입니다. 이는 예약된 babelfish_db 데이터베이스가 아닙니다. 예시: BabelfishDatabaseName=TargetDb;

PostgreSQL용 대상 데이터 형식

용 PostgreSQL 데이터베이스 엔드포인트는 대부분의 PostgreSQL 데이터베이스 데이터 형식을 AWS DMS 지원합니다. 다음 표에는 사용 시 지원되는 PostgreSQL 데이터베이스 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

AWS DMS 데이터 유형	PostgreSQL 데이터 형식
BOOLEAN	BOOLEAN
BLOB	BYTEA
BYTES	BYTEA
DATE	DATE

AWS DMS 데이터 유형	PostgreSQL 데이터 형식
TIME	TIME
DATETIME	크기 범위가 0~6인 경우, TIMESTAMP를 사용합니다. 크기 범위가 7~9인 경우, VARCHAR(37)을 사용합니다.
INT1	SMALLINT
INT2	SMALLINT
INT4	INTEGER
INT8	BIGINT
NUMERIC	DECIMAL (P,S)
REAL4	FLOAT4
REAL8	FLOAT8
STRING	길이가 1~21,845인 경우, VARCHAR(바이트 단위의 길이)을 사용합니다. 길이가 21,846~2,147,483,647인 경우, VARCHAR(65535)을 사용합니다.
UINT1	SMALLINT
UINT2	INTEGER
UINT4	BIGINT
UINT8	BIGINT
WSTRING	길이가 1~21,845인 경우, VARCHAR(바이트 단위의 길이)을 사용합니다. 길이가 21,846~2,147,483,647인 경우, VARCHAR(65535)을 사용합니다.

AWS DMS 데이터 유형	PostgreSQL 데이터 형식
NCLOB	TEXT
CLOB	TEXT

Note

PostgreSQL 소스에서 복제할 때는 사용자 정의 데이터 형식이 있는 열을 제외하고 모든 열에 대해 동일한 데이터 형식으로 대상 테이블을 AWS DMS 생성합니다. 그러한 경우에 데이터 형식은 대상에서 'character varying'으로 생성됩니다.

Babelfish for Aurora PostgreSQL을의 대상으로 사용 AWS Database Migration Service

AWS Database Migration Service을 사용하여 SQL Server 소스 테이블을 Babelfish for Amazon Aurora PostgreSQL 대상으로 마이그레이션할 수 있습니다. Aurora PostgreSQL은 Babelfish를 사용하여 Microsoft SQL Server의 전용 SQL 방언인 T-SQL을 이해하며 동일한 통신 프로토콜을 지원합니다. 따라서 이제 SQL Server용으로 작성된 애플리케이션은 코드 변경 횟수를 줄이면서 Aurora와 연동할 수 있습니다. Babelfish 기능은 Amazon Aurora에 내장되어 있으며 추가 비용이 들지 않습니다. Amazon RDS 콘솔에서 Amazon Aurora 클러스터의 Babelfish를 활성화할 수 있습니다.

AWS DMS 콘솔, API 또는 CLI 명령을 사용하여 AWS DMS 대상 엔드포인트를 생성할 때 대상 엔진을 Amazon Aurora PostgreSQL로 지정하고 데이터베이스 이름을 babelfish_db로 지정합니다. 엔드포인트 설정 섹션에서 DatabaseMode 설정을 Babelfish에 추가하고 대상 Babelfish T-SQL 데이터베이스의 이름에 BabelfishDatabaseName을 추가합니다.

마이그레이션 작업에 변환 규칙 추가

Babelfish 대상에 대한 마이그레이션 작업을 정의할 때는 DMS가 대상 데이터베이스의 사전 생성된 T-SQL Babelfish 테이블을 사용하도록 하는 변환 규칙을 포함해야 합니다.

먼저 모든 테이블 이름을 소문자로 만드는 변환 규칙을 마이그레이션 작업에 추가하세요. Babelfish는 T-SQL을 사용하여 생성한 테이블 이름을 PostgreSQL pg_class 카탈로그에 소문자로 저장합니다. 하지만 SQL Server 테이블에 대소문자가 혼합된 이름이 있는 경우, DMS는 T-SQL 호환 데이터 형식 대신 PostgreSQL 네이티브 데이터 형식을 사용하여 테이블을 생성합니다. 따라서 모든 테이블 이름을 소문자로 만드는 변환 규칙을 추가해야 합니다. 단, 열 이름은 소문자로 변환해서는 안 됩니다.

다음으로 클러스터를 정의할 때 다중 데이터베이스 마이그레이션 모드를 사용한 경우, 원본 SQL Server 스키마의 이름을 바꾸는 변환 규칙을 추가하세요. T-SQL 데이터베이스의 이름을 포함하도록 SQL Server 스키마의 이름을 바꿔야 합니다. 예를 들어 원래 SQL Server 스키마 이름이 dbo이고 T-SQL 데이터베이스 이름이 mydb인 경우, 변환 규칙을 사용하여 스키마 이름을 mydb_dbo로 바꾸세요.

Note

Babelfish for Aurora PostgreSQL 16 이상을 사용하는 경우 기본 마이그레이션 모드는 "mutidatabase"입니다. DMS 마이그레이션 작업을 실행할 때는 마이그레이션 모드 파라미터를 검토하고 필요한 경우 변환 규칙을 업데이트해야 합니다.

단일 데이터베이스 모드를 사용하는 경우, 스키마 이름을 바꾸는 변환 규칙은 필요하지 않습니다. 스키마 이름은 Babelfish의 대상 T-SQL 데이터베이스와 일대일 매핑됩니다.

다음 샘플 변환 규칙은 모든 테이블 이름을 소문자로 만들고 원래 SQL Server 스키마 이름을 dbo에서 mydb_dbo로 바꿉니다.

```
{
  "rules": [
    {
      "rule-type": "transformation",
      "rule-id": "566251737",
      "rule-name": "566251737",
      "rule-target": "schema",
      "object-locator": {
        "schema-name": "dbo"
      },
      "rule-action": "rename",
      "value": "mydb_dbo",
      "old-value": null
    },
    {
      "rule-type": "transformation",
      "rule-id": "566139410",
      "rule-name": "566139410",
      "rule-target": "table",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
    },
  ],
}
```

```

    "rule-action": "convert-lowercase",
    "value": null,
    "old-value": null
  },
  {
    "rule-type": "selection",
    "rule-id": "566111704",
    "rule-name": "566111704",
    "object-locator": {
      "schema-name": "dbo",
      "table-name": "%"
    },
    "rule-action": "include",
    "filters": []
  }
]
}

```

PostgreSQL 대상 엔드포인트를 Babelfish 테이블과 함께 사용할 때 적용되는 제한 사항

PostgreSQL 대상 엔드포인트를 Babelfish 테이블과 함께 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- 대상 테이블 준비 모드인 경우, 아무것도 안 함 또는 잘라내기 모드만 사용하세요. 대상에서 테이블 삭제 모드를 사용하지 마세요. 이 모드에서 DMS는 T-SQL이 인식하지 못할 수도 있는 PostgreSQL 테이블로 테이블을 생성합니다.
- AWS DMS 는 sql_variant 데이터 형식을 지원하지 않습니다.
- Postgres 엔드포인트의 Babelfish는 HEIRARCHYID, GEOMETRY (3.5.4 이전) 및 GEOGRAPHY (3.5.4 이전) 데이터 형식을 지원하지 않습니다. 이러한 데이터 형식을 마이그레이션하려면 변환 규칙을 추가하여 데이터 형식을 wstring(250)으로 변환할 수 있습니다.
- Babelfish는 해당 BYTEA 데이터 형식을 사용하여 BINARY, VARBINARY 및 IMAGE 데이터 형식의 마이그레이션만 지원합니다. Aurora PostgreSQL의 이전 버전에서는 DMS를 사용하여 이러한 테이블을 [Babelfish 대상 엔드포인트](#)로 마이그레이션할 수 있습니다. 다음 예제와 같이 BYTEA 데이터 형식에 길이를 지정할 필요가 없습니다.

```
[Picture] [VARBINARY](max) NULL
```

이전 T-SQL 데이터 형식을 T-SQL 지원 BYTEA 데이터 형식으로 변경합니다.

[Picture] BYTEA NULL

- 이전 버전의 Aurora PostgreSQL Babelfish에서 PostgreSQL 대상 엔드포인트를 사용하여 SQL Server에서 Babelfish로 지속적인 복제를 위한 마이그레이션 작업을 생성하는 경우, IDENTITY 열을 사용하는 모든 테이블에 SERIAL 데이터 형식을 할당해야 합니다. Aurora PostgreSQL(버전 15.3/14.8 및 이후 버전)과 Babelfish(버전 3.2.0 및 이후 버전)부터 ID 열이 지원되므로 시리얼 데이터 형식을 더 이상 할당할 필요가 없습니다. 자세한 내용은 SQL Server에서 Aurora PostgreSQL로의 마이그레이션 플레이북의 시퀀스 및 ID 섹션에서 [시리얼 사용](#)을 참조하세요. 그런 다음, Babelfish에서 테이블을 생성할 때 다음의 열 정의를 변경하세요.

[IDCol] [INT] IDENTITY(1,1) NOT NULL PRIMARY KEY

이전 정의를 다음과 같이 변경하세요.

[IDCol] SERIAL PRIMARY KEY

Babelfish와 호환되는 Aurora PostgreSQL은 기본 구성을 사용하여 하나의 시퀀스를 생성하고 해당 열에 NOT NULL 제약 조건을 추가합니다. 새로 만든 시퀀스는 (1씩 증가하는) 일반 시퀀스처럼 동작하며 복합 SERIAL 옵션이 없습니다.

- IDENTITY 열 또는 SERIAL 데이터 형식을 사용하는 테이블로 데이터를 마이그레이션한 후, 해당 열의 최대값을 기준으로 PostgreSQL 기반 시퀀스 객체를 재설정합니다. 테이블을 완전히 로드한 후 다음 T-SQL 쿼리를 사용하여 연결된 시퀀스 객체를 시드하는 문을 생성합니다.

```
DECLARE @schema_prefix NVARCHAR(200) = ''

IF current_setting('babelfishpg_tsql.migration_mode') = 'multi-db'
    SET @schema_prefix = db_name() + '_'

SELECT 'SELECT setval(pg_get_serial_sequence('' + @schema_prefix +
    schema_name.tables.schema_id) + '.' + tables.name + '', '' + columns.name + '')
    ,(select max(' + columns.name + ') from ' +
    schema_name.tables.schema_id) + '.' + tables.name + ');'
FROM sys.tables tables
JOIN sys.columns columns ON tables.object_id = columns.object_id
WHERE columns.is_identity = 1

UNION ALL
```

```
SELECT 'SELECT setval(pg_get_serial_sequence('' + @schema_prefix + table_schema +
    '.' + table_name + ''',
    '' + column_name + '''),(select max(' + column_name + ') from ' + table_schema + '.' +
    table_name + '));'
FROM information_schema.columns
WHERE column_default LIKE 'nextval(%)';
```

쿼리는 최대 IDENTITY 및 SERIAL 값을 업데이트하기 위해 실행하는 일련의 SELECT 문을 생성합니다.

- Babelfish 3.2 이전 버전의 경우 전체 LOB 모드에서 테이블 오류가 발생할 수 있습니다. 이 경우 로드에 실패한 테이블에 대해 별도의 작업을 생성하세요. 그런 다음 제한된 LOB 모드를 사용하여 최대 LOB 크기(KB)에 적합한 값을 지정합니다. 또 다른 옵션은 SQL Server 엔드포인트 연결 속성 설정을 ForceFullLob=True로 설정하는 것입니다.
- Babelfish 3.2 이전 버전의 경우, 정수 기반 프라이머리 키를 사용하지 않는 Babelfish 테이블로 데이터 유효성 검사를 수행하면 적절한 고유 키를 찾을 수 없다는 메시지가 생성됩니다. Aurora PostgreSQL(버전 15.3/14.8 및 이후 버전) 및 Babelfish(버전 3.2.0 및 이후 버전)를 시작으로 정수가 아닌 프라이머리 키에 대한 데이터 검증이 지원됩니다.
- 초 단위 소수점 자릿수의 정밀도 차이 때문에 DMS는 DATETIME 데이터 형식을 사용하는 Babelfish 테이블에 대한 데이터 검증 실패를 보고합니다. 이러한 실패를 억제하기 위해 DATETIME 데이터 형식에 대해 다음과 같은 유효성 검사 규칙 유형을 추가할 수 있습니다.

```
{
    "rule-type": "validation",
    "rule-id": "3",
    "rule-name": "3",
    "rule-target": "column",
    "object-locator": {
        "schema-name": "dbo",
        "table-name": "%",
        "column-name": "%",
        "data-type": "datetime"
    },
    "rule-action": "override-validation-function",
    "source-function": "case when ${column-name} is NULL then NULL else 0 end",
    "target-function": "case when ${column-name} is NULL then NULL else 0 end"
}
```

MySQL 호환 데이터베이스를 대상으로 사용 AWS Database Migration Service

가 AWS DMS 지원하는 소스 데이터 엔진 AWS DMS에서 사용하여 모든 MySQL 호환 데이터베이스로 데이터를 마이그레이션할 수 있습니다. 온프레미스 MySQL 호환 데이터베이스로 마이그레이션하는 경우 AWS DMS에서는 소스 엔진이 AWS 에코시스템 내에 있어야 합니다. 엔진은 Amazon RDS, Amazon Aurora 또는 Amazon S3와 같은 AWS관리형 서비스에 있을 수 있습니다. 또는 Amazon EC2의 자체 관리형 데이터베이스에 있을 수도 있습니다.

SSL을 사용하여 MySQL 호환 엔드포인트와 복제 인스턴스 간의 연결을 암호화할 수 있습니다. MySQL 호환 엔드포인트에서 SSL 사용에 관한 자세한 내용은 [에서 SSL 사용 AWS Database Migration Service](#) 섹션을 참조하세요.

가 대상으로 AWS DMS 지원하는 MySQL 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 대상 AWS DMS](#).

다음 MySQL 호환 데이터베이스를 대상으로 사용할 수 있습니다 AWS DMS.

- MySQL Community Edition
- MySQL Standard Edition
- MySQL Enterprise Edition
- MySQL Cluster Carrier Grade Edition
- MariaDB Community Edition
- MariaDB Enterprise Edition
- MariaDB Column Store
- Amazon Aurora MySQL

Note

소스 스토리지 엔진(MyISAM, MEMORY 등)에 상관없이 AWS DMS 는 기본적으로 MySQL 호환 대상 테이블을 InnoDB 테이블로서 생성합니다.

InnoDB 이외에 스토리지 엔진에 테이블이 있어야 하는 경우, MySQL 호환 대상에서 테이블을 수동으로 생성하고 아무 작업 안 함 옵션을 사용하여 테이블을 마이그레이션할 수 있습니다. 자세한 내용은 [전체 로드 작업 설정](#) 단원을 참조하십시오.

MySQL 호환 데이터베이스를 AWS DMS의 대상으로 사용하여 작업하는 방법에 관한 자세한 내용은 다음 섹션을 참조하세요.

주제

- [MySQL 호환 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#)
- [MySQL 호환 데이터베이스를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service](#)
- [MySQL 호환 데이터베이스를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#)
- [MySQL용 대상 데이터 형식](#)

MySQL 호환 데이터베이스를의 대상으로 사용 AWS Database Migration Service

AWS DMS에서 MySQL 호환 데이터베이스를 대상으로 사용하기 전에 다음 사전 조건이 완료되어 있는지 확인해야 합니다.

- MySQL 호환 데이터베이스에 대한 읽기/쓰기 권한이 AWS DMS 있는 사용자 계정에 제공됩니다. 필요한 권한을 생성하려면 다음 명령을 실행합니다.

```
CREATE USER '<user acct>'@'%' IDENTIFIED BY '<user password>';
GRANT ALTER, CREATE, DROP, INDEX, INSERT, UPDATE, DELETE, SELECT, CREATE TEMPORARY
TABLES ON <schema>.* TO
'<user acct>'@'%;
GRANT ALL PRIVILEGES ON awsdms_control.* TO '<user acct>'@'%';
```

- 전체 로드 마이그레이션 단계 중에 대상 테이블에 대해 외래 키를 비활성화해야 합니다. 전체 로드 중에 MySQL 호환 데이터베이스에서 외래 키 검사를 비활성화하려면 대상 엔드포인트에 대한 AWS DMS 콘솔의 추가 연결 속성 섹션에 다음 명령을 추가할 수 있습니다.

```
Initstmt=SET FOREIGN_KEY_CHECKS=0;
```

- AWS DMS 가 데이터를 대상 데이터베이스로 로드할 수 있도록 데이터베이스 파라미터 `local_infile = 1`을 설정합니다.
- MySQL별 마이그레이션 전 평가를 사용하는 경우 다음 권한을 부여합니다.

```
grant select on mysql.user to <dms_user>;
grant select on mysql.db to <dms_user>;
grant select on mysql.tables_priv to <dms_user>;
```

```
grant select on mysql.role_edges to <dms_user> #only for MySQL version 8.0.11 and higher
```

MySQL 호환 데이터베이스를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

MySQL 데이터베이스를 대상으로 사용하는 경우 AWS DMS 는 다음을 지원하지 않습니다.

- 데이터 정의 언어(DDL) 문은 TRUNCATE PARTITION, DROP TABLE 및 CREATE TABLE 명령을 수행합니다.
- ALTER TABLE *table_name* ADD COLUMN *column_name* 문을 사용하여 테이블의 시작이나 중간에 열을 추가합니다.
- 전체 로드 작업에서 MySQL 호환 대상에 데이터를 로드할 때 AWS DMS 는 작업 로그의 제약으로 인해 발생한 오류를 보고하지 않습니다. 이로 인해 중복 키 오류가 발생하거나 레코드 수와 일치하지 않을 수 있습니다. 이러한 문제는 MySQL이 LOAD DATA 명령으로 로컬 데이터를 처리하는 방식 때문에 발생합니다. 전체 로드 단계에서 다음을 꼭 수행해야 합니다.
 - d제약 조건 비활성화
 - AWS DMS 검증을 사용하여 데이터가 일치하는지 확인합니다.
- 열의 값을 기존 값으로 업데이트하면 MySQL 호환 데이터베이스는 0 rows affected 경고를 반환합니다. 이 동작은 사실 오류는 아니지만 다른 데이터베이스 엔진이 이러한 상황을 처리하는 방식과는 다릅니다. 예를 들어, Oracle은 행 하나를 업데이트합니다. MySQL 호환 데이터베이스의 경우 awsdms_apply_exceptions 제어 테이블에 항목을 AWS DMS 생성하고 다음 경고를 기록합니다.

Some changes from the source database had no impact when applied to the target database. See awsdms_apply_exceptions table for details.

- MySQL 버전 5.7과 호환되는 Amazon Aurora 버전 2의 대상으로 Aurora Serverless를 사용할 수 있습니다. (MySQL 버전 5.7과 호환되는 Aurora Serverless를 사용할 수 있도록 Aurora MySQL 버전 2.07.1을 선택합니다.) Aurora Serverless에 관한 자세한 내용을 알아보려면 Amazon Aurora 사용 설명서의 [Aurora Serverless v2 사용](#)을 참조하세요.
- AWS DMS 는 인스턴스가 쓰기 가능 모드, 즉 read_only 및 innodb_read_only 파라미터가 0 또는 1로 설정되어 있지 않은 한 Aurora 또는 Amazon RDS에 대한 리더 엔드포인트 사용을 지원하지 않

습니다. OFF. Amazon RDS 및 Aurora를 대상으로 사용하는 방법에 관한 자세한 내용은 다음을 참조하세요.

- [연결되어 있는 DB 인스턴스 확인](#)
- [MySQL을 사용한 읽기 전용 복제본 업데이트](#)
- TIME 데이터 유형을 복제할 때 시간 값의 소수 부분은 복제되지 않습니다.
- 추가 연결 속성을 사용하여 TIME 데이터 유형을 복제할 때 loadUsingCSV=false 시간 값은 범위 [00:00:00, 23:59:59]로 제한됩니다.

MySQL 호환 데이터베이스를의 대상으로 사용 시 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 MySQL 호환 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --my-sql-settings '{"EndpointSetting": "value", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

MySQL을 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

이름	설명
ConnectionTimeout	이 추가 연결 속성(ECA)을 사용하여 MySQL 인스턴스의 엔드포인트 연결 제한 시간을 초 단위로 설정합니다. 기본 값은 10초입니다. ECA 예제: ConnectionTimeout=30 .
TargetDbType	대상에 있는 소스 테이블을 단일 데이터베이스 또는 다중 데이터베이스로 마이그레이션할 위치를 지정합니다. 를 지정하는 경우 AWS CLI 또는를 사용할 때 데이터베이스 이름을 지정SPECIFIC_DATABASE 해야 합니다 AWS Management Console. 기본값: MULTIPLE_DATABASES 유효값: {SPECIFIC_DATABASE , MULTIPLE_DATABASES } 예시: --my-sql-settings '{"TargetDbType": "MULTIPLE_DATABASES"}'

이름	설명
ParallelLoadThreads	데이터를 MySQL 호환 대상 데이터베이스로 로드할 때의 성능을 개선합니다. 데이터를 MySQL 호환 대상 데이터베이스에 로드할 때 사용할 스레드 개수를 지정합니다. 각 스레드마다 별도의 연결이 필요하기 때문에 대량의 스레드를 설정하면 데이터베이스 성능에 부정적인 영향을 줄 수 있습니다. 기본값: 1 유효값: 1~5 예시: <code>--my-sql-settings '{"ParallelLoadThreads": 1}'</code>
AfterConnectScript	AWS DMS 가 엔드포인트에 연결된 즉시 실행할 스크립트를 지정합니다. 예를 들면, MySQL 호환 대상이 받은 명령문을 latin1 문자 세트로 변환해야 함을 지정할 수 있습니다. 이 문자 세트는 데이터베이스의 기본 컴파일 입력 문자 세트입니다. 이 파라미터는 UTF8 클라이언트로부터 변환될 때 일반적으로 성능을 개선합니다. 예시: <code>--my-sql-settings '{"AfterConnectScript": "SET character_set_connection='latin1'"}</code>
MaxFileSize	데이터를 MySQL 호환 데이터베이스로 전송할 때 사용되는 .csv 파일의 최대 크기(단위: KB)를 지정합니다. 기본값: 32,768KB(32MB) 유효값: 1~1,048,576 <code>--my-sql-settings '{"MaxFileSize": 512}'</code>

또한 추가 연결 속성을 사용하여 MySQL 호환 대상을 구성할 수도 있습니다.

다음 표에는 MySQL을 대상으로 하여 사용할 수 있는 추가 연결 속성이 나와 있습니다.

이름	설명
Initstmt=SET FOREIGN_KEY_CHECKS=0;	외래 키 점검을 비활성화합니다. 예시: --extra-connection-attributes "Initstmt=SET FOREIGN_KEY_CHECKS=0;"
Initstmt=SET time_zone	대상 MySQL 호환 데이터베이스의 시간대를 지정합니다. 기본값: UTC 유효값: 대상 MySQL 데이터베이스에서 사용할 수 있는 시간대 이름. 예시: --extra-connection-attributes "Initstmt=SET time_zone= <i>US/Pacific</i> ;"

또는 --my-sql-settings 명령의 AfterConnectScript 파라미터를 사용하여 외래 키 점검을 비활성화하고 데이터베이스의 시간대를 지정할 수 있습니다.

MySQL용 대상 데이터 형식

다음 표에는를 사용할 때 지원되는 MySQL 데이터베이스 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

AWS DMS 데이터 형식	MySQL 데이터 형식
BOOLEAN	BOOLEAN
BYTES	길이가 1~65,535인 경우, VARBINARY(길이)를 사용합니다. 길이가 65,536~2,147,483,647인 경우, LONGLOB를 사용합니다.

AWS DMS 데이터 형식	MySQL 데이터 형식
DATE	DATE
TIME	TIME
TIMESTAMP	"크기가 0 이상이고 6 이하인 경우: DATETIME(크기) 크기가 7 이상이고 9 이하인 경우: VARCHAR(37)"
INT1	TINYINT
INT2	SMALLINT
INT4	INTEGER
INT8	BIGINT
NUMERIC	DECIMAL (p,s)
REAL4	FLOAT
REAL8	DOUBLE PRECISION
STRING	길이가 1~21,845인 경우, VARCHAR(길이)를 사용합니다. 길이가 21,846~2,147,483,647인 경우, LONGTEXT를 사용합니다.
UINT1	UNSIGNED TINYINT
UINT2	UNSIGNED SMALLINT
UINT4	UNSIGNED INTEGER
UINT8	UNSIGNED BIGINT

AWS DMS 데이터 형식	MySQL 데이터 형식
WSTRING	길이가 1~32,767인 경우, VARCHAR(길이)를 사용합니다. 길이가 32,768~2,147,483,647인 경우, LONGTEXT를 사용합니다.
BLOB	길이가 1~65,535인 경우, BLOB를 사용합니다. 길이가 65,536~2,147,483,647인 경우, LONGBLOB를 사용합니다. 길이가 0인 경우, LONGBLOB(전체 LOB 지원)를 사용합니다.
NCLOB	길이가 1~65,535인 경우, TEXT를 사용합니다. 길이가 65,536~2,147,483,647인 경우, CHARACTER SET용 ucs2와 함께 LONGTEXT를 사용합니다. 길이가 0인 경우, CHARACTER SET용 ucs2와 함께 LONGTEXT(전체 LOB 지원)를 사용합니다.
CLOB	길이가 1~65,535인 경우, TEXT를 사용합니다. 길이가 65,536~2,147,483,647인 경우, LONGTEXT를 사용합니다. 길이가 0인 경우, LONGTEXT(전체 LOB 지원)를 사용합니다.

Amazon Redshift 데이터베이스를 대상으로 사용 AWS Database Migration Service

를 사용하여 Amazon Redshift 데이터베이스로 데이터를 마이그레이션할 수 있습니다 AWS Database Migration Service. Amazon Redshift는 클라우드에서 완전히 관리되는 페타바이트급 데이터 웨어하우

스 서비스입니다. Amazon Redshift 데이터베이스를 대상으로 사용할 때에는 지원되는 다른 소스 데이터베이스에서 가져온 데이터를 마이그레이션할 수 있습니다.

Amazon Redshift Serverless를 대상으로 사용할 수 있습니다 AWS DMS. 자세한 내용은 [AWS DMS Amazon Redshift Serverless에서 대상으로 사용](#) 섹션을 참조하세요.

Amazon Redshift 클러스터는 복제 인스턴스와 동일한 AWS 계정 및 동일한 AWS 리전에 있어야 합니다.

Amazon Redshift로 데이터베이스를 마이그레이션하는 동안 AWS DMS 먼저 데이터를 Amazon S3 버킷으로 이동합니다. 파일이 Amazon S3 버킷에 있는 경우 Amazon Redshift 데이터 웨어하우스의 적절한 테이블로 전송합니다. AWS DMS 그러면 Amazon Redshift 데이터베이스와 동일한 AWS 리전에 S3 버킷이 AWS DMS 생성됩니다. AWS DMS 복제 인스턴스는 동일한 AWS 리전에 있어야 합니다.

AWS CLI 또는 DMS API를 사용하여 데이터를 Amazon Redshift로 마이그레이션하는 경우 S3 액세스를 허용하도록 AWS Identity and Access Management (IAM) 역할을 설정합니다. 이 IAM 역할 생성에 관한 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.

Amazon Redshift 엔드포인트가 전체 자동화를 지원하는 경우는 다음과 같습니다.

- 스키마 생성 및 데이터 형식 매핑
- 소스 데이터베이스 테이블 전체 로드
- 원본 테이블에 적용된 변경 사항을 증분식으로 로드
- 원본 테이블에 적용된 데이터 정의 언어(DDL)에서 스키마 변경 사항 적용
- 전체 로드와 변경 데이터 캡처(CDC) 프로세스 동기화.

AWS Database Migration Service 는 전체 로드 및 변경 처리 작업을 모두 지원합니다.는 소스 데이터베이스에서 데이터를 AWS DMS 읽고 일련의 쉼표로 구분된 값(.csv) 파일을 생성합니다. 전체 로드 작업의 경우는 각 테이블의 파일을 AWS DMS 생성합니다. AWS DMS 그런 다음은 각 테이블의 테이블 파일을 Amazon S3의 별도 폴더에 복사합니다. 파일이 Amazon S3에 업로드되면 복사 명령을 AWS DMS 전송하고 파일의 데이터가 Amazon Redshift에 복사됩니다. 변경 처리 작업의 경우는 .csv 파일에 대한 순 변경 사항을 AWS DMS 복사합니다. AWS DMS 그런 다음은 Amazon S3에 순 변경 파일을 업로드하고 Amazon Redshift에 데이터를 복사합니다.

Amazon Redshift를 대상으로 사용하는 방법에 대한 자세한 내용은 다음 섹션을 AWS DMS참조하세요.

주제

- [Amazon Redshift 데이터베이스를의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service](#)
- [Redshift를 대상으로 사용하는 데 필요한 권한](#)
- [Amazon Redshift를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service](#)
- [Amazon Redshift 데이터베이스를의 대상으로 구성 AWS Database Migration Service](#)
- [Amazon Redshift에서의 대상으로 향상된 VPC 라우팅 사용 AWS Database Migration Service](#)
- [AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화](#)
- [Amazon Redshift를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#)
- [데이터 암호화 키와 Amazon S3 버킷을 중간 스토리지로 사용](#)
- [Amazon Redshift에 대한 멀티스레드 작업 설정](#)
- [Amazon Redshift에 대한 대상 데이터 형식](#)
- [AWS DMS Amazon Redshift Serverless에서 대상으로 사용](#)

Amazon Redshift 데이터베이스를의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service

다음 목록에는 데이터 마이그레이션에서 Amazon Redshift를 대상으로 사용할 때 필요한 사전 조건에 대한 설명이 나와 있습니다.

- AWS Management Console을 사용하여 Amazon Redshift 클러스터를 시작합니다. 암호, 사용자 이름, 데이터베이스 이름 등 AWS 계정 및 Amazon Redshift 클러스터에 대한 기본 정보를 기록해 둡니다. Amazon Redshift 대상 엔드포인트를 생성할 때 이 값이 필요합니다.
- Amazon Redshift 클러스터는 복제 인스턴스와 동일한 AWS 계정 및 동일한 AWS 리전에 있어야 합니다.
- AWS DMS 복제 인스턴스에는 클러스터에서 사용하는 Amazon Redshift 엔드포인트(호스트 이름 및 포트)에 대한 네트워크 연결이 필요합니다.
- AWS DMS 는 Amazon S3 버킷을 사용하여 Amazon Redshift 데이터베이스로 데이터를 전송합니다. AWS DMS 에서 버킷을 생성하는 경우, 콘솔은 IAM 역할인 `dms-access-for-endpoint`를 사용합니다. AWS CLI 또는 DMS API를 사용하여 Amazon Redshift를 대상 데이터베이스로 사용하여 데이터베이스 마이그레이션을 생성하는 경우 IAM 역할을 생성해야 합니다. 이 역할 생성에 관한 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.
- AWS DMS 는 BLOBs, CLOBs 및 NCLOBs 대상 Amazon Redshift 인스턴스의 VARCHAR로 변환합니다. Amazon Redshift는 64KB보다 큰 VARCHAR 데이터 형식을 지원하지 않으므로 Amazon Redshift에는 기존 LOB를 저장할 수 없습니다.

- CDC 중에 Amazon Redshift 대상 테이블에 대한 변경 사항을 처리하려면 AWS DMS 에 대한 대상 메타데이터 작업 설정 [BatchApplyEnabled](#)를 true로 설정합니다. 소스와 대상 테이블에 모두 프라이머리 키가 필요합니다. 프라이머리 키가 없으면 변경 사항이 문별로 적용됩니다. 또한 대상 지연 시간이 발생하고 클러스터 커밋 대기열에 영향을 주므로 CDC 중에 작업 성능이 저하될 수 있습니다.
- Redshift의 테이블에서 행 수준 보안이 활성화된 경우 모든 DMS 사용자에게 적절한 권한을 부여해야 합니다.

Redshift를 대상으로 사용하는 데 필요한 권한

GRANT 명령을 사용하여 사용자 또는 사용자 그룹의 액세스 권한을 정의합니다. 권한은 테이블과 뷰에서 데이터를 읽거나 쓰고 테이블을 만들 수 있는 것과 같은 액세스 옵션을 포함합니다. Amazon Redshift에서 GRANT를 사용하는 방법에 관한 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서의 [GRANT](#)를 참조하세요.

다음은 테이블, 데이터베이스, 스키마, 함수, 프로시저 또는 Amazon Redshift 테이블 및 뷰의 언어 수준 권한에 대한 특정 권한을 부여하는 구문입니다.

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | REFERENCES } [,...] | ALL
[ PRIVILEGES ] }
    ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]

GRANT { { CREATE | TEMPORARY | TEMP } [,...] | ALL [ PRIVILEGES ] }
    ON DATABASE db_name [, ...]
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]

GRANT { { CREATE | USAGE } [,...] | ALL [ PRIVILEGES ] }
    ON SCHEMA schema_name [, ...]
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { FUNCTION function_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
    FUNCTIONS IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { PROCEDURE procedure_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
    PROCEDURES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]
```

```
GRANT USAGE
    ON LANGUAGE language_name [, ...]
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]
```

다음은 Amazon Redshift 테이블 및 뷰에서 열 수준 권한에 대한 구문입니다.

```
GRANT { { SELECT | UPDATE } ( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ]
    ( column_name [, ...] ) }
    ON { [ TABLE ] table_name [, ...] }
    TO { username | GROUP group_name | PUBLIC } [, ...]
```

다음은 지정된 역할을 가진 사용자 및 그룹에 부여된 ASSUMEROLE 권한에 대한 구문입니다.

```
GRANT ASSUMEROLE
    ON { 'iam_role' [, ...] | ALL }
    TO { username | GROUP group_name | PUBLIC } [, ...]
    FOR { ALL | COPY | UNLOAD } [, ...]
```

Amazon Redshift를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

다음 제한 사항은 Amazon Redshift 데이터베이스를 대상으로 사용 시 적용됩니다.

- Amazon Redshift 대상의 중간 스토리지로 사용하는 S3 버킷의 버전 관리를 활성화하지 마세요. S3 버전 관리가 필요한 경우, 수명 주기 정책을 사용하여 이전 버전을 적극적으로 삭제하세요. 그렇지 않으면 S3 list-object 직접 호출 시간 초과로 인해 엔드포인트 테스트 연결 실패가 발생할 수 있습니다. S3 버킷의 수명 주기 정책을 생성하려면 [스토리지 수명 주기 관리](#)를 참조하세요. S3 객체의 버전을 삭제하려면 [버전 관리 활성화 버킷에서 객체 버전 삭제](#)를 참조하세요.
- 다음 DDL은 지원되지 않습니다.

```
ALTER TABLE table name MODIFY COLUMN column name data type;
```

- AWS DMS 는 이름이 밑줄(_)로 시작하는 스키마에 대한 변경 사항을 마이그레이션하거나 복제할 수 없습니다. 밑줄로 시작하는 이름의 스키마가 있다면 매핑 변환을 사용하여 대상에 있는 스키마의 이름을 바꿉니다.
- Amazon Redshift는 64KB를 초과하는 VARCHAR를 지원하지 않습니다. 기존 데이터베이스의 LOB 는 Amazon Redshift에 저장할 수 없습니다.

- 프라이머리 키 열 이름 중 하나에서 예약어를 사용하는 경우 다중 열 프라이머리 키가 있는 테이블에 DELETE 문을 적용하는 것은 지원되지 않습니다. Amazon Redshift 예약어 목록을 보려면 [여기](#)로 이동합니다.
- 소스 시스템이 소스 테이블의 프라이머리 키에 대해 UPDATE 작업을 수행하는 경우 성능 문제가 발생할 수 있습니다. 이러한 성능 문제는 대상에 변경 내용을 적용할 때 발생합니다. 이는 UPDATE(및 DELETE) 작업이 대상 행을 식별하는 프라이머리 키 값에 의존하기 때문입니다. 소스 테이블의 프라이머리 키를 업데이트하면 작업 로그에 다음과 같은 메시지가 포함됩니다.

Update on table 1 changes PK to a PK that was previously updated in the same bulk update.

- DMS는 Redshift 클러스터의 엔드포인트를 구성할 때 사용자 지정 DNS 이름을 지원하지 않으므로 Amazon에서 제공한 DNS 이름을 사용해야 합니다. Amazon Redshift 클러스터는 복제 인스턴스와 동일한 AWS 계정 및 리전에 있어야 하므로 사용자 지정 DNS 엔드포인트를 사용하는 경우 검증이 실패합니다.
- Amazon Redshift에는 기본적으로 4시간의 유휴 세션 제한 시간이 있습니다. DMS 복제 작업 내에 활동이 없는 경우, Redshift는 4시간 후에 세션 연결을 끊습니다. DMS를 연결할 수 없고 다시 시작해야 할 수도 있기 때문에 오류가 발생할 수 있습니다. 해결 방법으로 DMS 복제 사용자의 세션 제한 시간을 4시간 이상으로 설정하세요. 또는 Amazon Redshift 데이터베이스 개발자 안내서의 [ALTER USER](#)에 관한 설명을 참조하세요.
- 가 기본 또는 고유 키 없이 소스 테이블 데이터를 AWS DMS 복제하는 경우 CDC 지연 시간이 높아져 허용할 수 없는 수준의 성능이 발생할 수 있습니다.
- Oracle 소스에서 Redshift 대상으로 CDC 복제 중에는 파티션 잘라내기가 지원되지 않습니다.
- Amazon Redshift는 기본 키를 적용하지 않고 작업이 재개될 때 AWS DMS가 CDC를 재생할 수 있기 때문에 대상 테이블에 중복 레코드가 나타날 수 있습니다. 중복을 방지하려면 `ApplyErrorInsertPolicy=INSERT_RECORD` 설정을 사용합니다. 자세한 내용은 [오류 처리 작업 설정](#) 단원을 참조하십시오. 또는 애플리케이션 수준의 중복 탐지 및 마이그레이션 후 정리 절차를 구현할 수 있습니다.

Amazon Redshift 데이터베이스를 대상으로 구성 AWS Database Migration Service

AWS Database Migration Service Amazon Redshift 인스턴스에서 작동하도록 구성해야 합니다. 다음 표에는 Amazon Redshift 엔드포인트에서 사용할 수 있는 구성 속성에 관한 설명이 나와 있습니다.

속성	설명
서버	사용 중인 Amazon Redshift 클러스터의 이름입니다.
포트	Amazon Redshift에 대한 포트 번호입니다. 기본값은 5439입니다.
사용자 이름	등록된 사용자의 Amazon Redshift 사용자 이름입니다.
암호	사용자 이름 속성에 명명된 사용자의 암호입니다.
데이터베이스	사용 중인 Amazon Redshift 데이터 웨어하우스(서비스)의 이름입니다.

Amazon Redshift 엔드포인트에 추가 연결 문자열 속성을 추가하려면 `maxFileSize` 및 `fileTransferUploadStreams` 속성을 지정합니다. 이 속성에 관한 자세한 내용은 [Amazon Redshift를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요.

Amazon Redshift에서의 대상으로 향상된 VPC 라우팅 사용 AWS Database Migration Service

Amazon Redshift 대상에 Enhanced VPC Routing을 사용하는 경우, Amazon Redshift 클러스터와 데이터 리포지토리 간 모든 COPY 트래픽은 VPC를 통과합니다. Enhanced VPC Routing은 Amazon Redshift가 다른 리소스에 액세스하는 방식에 영향을 주기 때문에 VPC를 올바르게 구성하지 않았다면 COPY 명령이 실패할 수 있습니다.

AWS DMS 는 COPY 명령을 사용하여 S3의 데이터를 Amazon Redshift 클러스터로 이동하기 때문에 이 동작의 영향을 받을 수 있습니다.

다음은 Amazon Redshift 대상으로 데이터를 로드하기 위해 AWS DMS 수행하는 단계입니다.

1. AWS DMS 는 소스의 데이터를 복제 서버의 .csv 파일로 복사합니다.
2. AWS DMS 는 AWS SDK를 사용하여 .csv 파일을 계정의 S3 버킷에 복사합니다.
3. AWS DMS 그런 다음은 Amazon Redshift의 COPY 명령을 사용하여 S3의 .csv 파일에서 Amazon Redshift의 적절한 테이블로 데이터를 복사합니다.

Enhanced VPC Routing이 활성화되지 않은 경우 Amazon Redshift는 트래픽을 포함하여 인터넷을 통해 AWS 네트워크 내의 다른 서비스로 트래픽을 라우팅합니다. 이 기능이 활성화되지 않을 경우, 네트워크 경로를 구성하지 않아도 됩니다. 이 기능이 활성화되면 특히 클러스터의 VPC와 사용자의 데이터

리소스 사이의 네트워크 경로를 생성해야 합니다. 필요한 구성에 관한 자세한 내용은 Amazon Redshift 설명서의 [Enhanced VPC Routing](#)을 참조하세요.

AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화

Amazon Redshift에 복사되기 전에 Amazon S3로 푸시되는 대상 데이터를 암호화할 수 있습니다. 이렇게 하려면 사용자 지정 AWS KMS 키를 생성하고 사용할 수 있습니다. 생성한 키를 사용하여 Amazon Redshift 대상 엔드포인트를 생성할 때 다음 메커니즘 중 하나를 사용하여 대상 데이터를 암호화할 수 있습니다.

- AWS CLI를 사용하여 create-endpoint 명령을 실행할 때 다음 옵션 중 하나를 사용합니다.

```
--redshift-settings '{"EncryptionMode": "SSE_KMS", "ServerSideEncryptionKmsKeyId":  
  "your-kms-key-ARN"}
```

여기서 *your-kms-key-ARN*은 KMS 키의 Amazon 리소스 이름(ARN)입니다. 자세한 내용은 [데이터 암호화 키와 Amazon S3 버킷을 중간 스토리지로 사용](#) 단원을 참조하십시오.

- 추가 연결 속성 encryptionMode를 값 SSE_KMS로 설정하고, 추가 연결 속성 serverSideEncryptionKmsKeyId는 KMS 키의 ARN으로 설정합니다. 자세한 내용은 [Amazon Redshift를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

KMS 키를 사용하여 Amazon Redshift 대상 데이터를 암호화하려면 Amazon Redshift 데이터에 액세스할 수 있는 권한이 있는 AWS Identity and Access Management (IAM) 역할이 필요합니다. 그러면 생성한 암호화 키에 연결된 정책(키 정책)에서 이 IAM 역할에 액세스합니다. 이 작업은 IAM 콘솔에서 다음을 생성하여 수행할 수 있습니다.

- AWS관리형 정책이 있는 IAM 역할입니다.
- 이 역할을 참조하는 키 정책이 있는 KMS 키.

다음 절차에서 그 방법을 설명합니다.

AWS필수 관리형 정책을 사용하여 IAM 역할을 생성하려면

1. IAM 콘솔(<https://console.aws.amazon.com/iam/>)을 엽니다.
2. 탐색 창에서 역할을 선택합니다. 역할 페이지가 열립니다.
3. 역할 생성을 선택합니다. 역할 생성 페이지가 열립니다.

4. AWS 서비스를 신뢰할 수 있는 엔터티 유형으로 선택한 상태에서 해당 역할을 사용할 서비스로 DMS를 선택합니다.
5. 다음: 권한을 선택합니다. 권한 정책 연결 페이지가 나타납니다.
6. AmazonDMSRedshiftS3Role 정책을 찾아 선택합니다.
7. 다음: 태그를 선택합니다. 태그 추가 페이지가 나타납니다. 여기서 원하는 태그를 추가할 수 있습니다.
8. 다음: 검토를 선택하고 결과를 검토합니다.
9. 필요한 설정이 되었다면 역할의 이름(예: DMS-Redshift-endpoint-access-role)과 추가 설명을 입력한 다음 역할 생성을 선택합니다. 역할이 생성되었다는 메시지와 함께 역할 페이지가 열립니다.

이제 암호화를 위해 Amazon Redshift 리소스에 액세스할 수 있는 지정된 이름의 새 역할이 생성되었습니다(예: DMS-Redshift-endpoint-access-role).

IAM 역할을 참조하는 키 정책을 사용하여 AWS KMS 암호화 키를 생성하려면

Note

에서 AWS KMS 암호화 키를 AWS DMS 사용하는 방법에 대한 자세한 내용은 섹션을 참조하세요. [암호화 키 설정 및 AWS KMS 권한 지정](#).

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/kms> AWS Key Management Service (AWS KMS) 콘솔을 엽니다.
2. 를 변경하려면 페이지 오른쪽 상단에 있는 리전 선택기를 AWS 리전사용합니다.
3. 탐색 창에서 고객 관리형 키를 선택합니다.
4. 키 생성을 선택합니다. 키 구성 페이지가 열립니다.
5. 키 유형에 대해 대칭을 선택합니다.

Note

이 키를 생성할 때 Amazon Redshift와 같은 모든 AWS 서비스는 대칭 암호화 키로만 작동하므로 대칭 키만 생성할 수 있습니다.

6. 고급 옵션을 선택합니다. 키 구성 요소 오리진에 KMS가 선택되어 있는지 확인하고 다음을 선택합니다. 레이블 추가 페이지가 열립니다.

7. 별칭 및 설명 생성에 키의 별칭(예: DMS-Redshift-endpoint-encryption-key)과 추가 설명을 입력합니다.
8. 태그에서 키 식별과 키 사용 추적에 도움이 되는 태그를 추가한 후 다음을 선택합니다. 키 관리 권한 정의 페이지가 열리고 선택할 수 있는 사용자 및 역할 목록이 표시됩니다.
9. 키를 관리할 사용자와 역할을 추가합니다. 키를 관리하는 데 필요한 권한이 이러한 사용자와 역할에 있는지 확인합니다.
10. 키 삭제에서 키 관리자가 키를 삭제할 수 있는지 여부를 선택한 후 다음을 선택합니다. 키 사용 권한 정의 페이지가 열리고 선택할 수 있는 추가 사용자 및 역할 목록이 표시됩니다.
11. 이 계정의 Amazon Redshift 대상에 대해 암호화 작업을 수행할 사용자를 선택합니다. 또한 이전에 역할에서 생성한 역할을 선택하여 Amazon Redshift 대상 객체 암호화를 위한 액세스를 활성화합니다(예: DMS-Redshift-endpoint-access-role).
12. 이 동일한 액세스 권한을 갖도록 목록에 없는 다른 계정을 추가하려면 다른 AWS 계정에서 다른 AWS 계정 추가를 선택한 후 다음을 선택합니다. 키 정책 검토 및 편집 페이지가 열리고 기존 JSON을 입력하여 검토하고 편집할 수 있는 키 정책 JSON이 표시됩니다. 여기서 키 정책이 이전 단계에서 선택한 역할 및 사용자(예: Admin 및 User1)를 참조하는 위치를 확인할 수 있습니다. 다음 예에 나온 것처럼 다양한 위탁자(사용자 및 역할)에 허용되는 다양한 키 작업도 볼 수 있습니다.

JSON

```
{
  "Id": "key-consolepolicy-3",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
        ]
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow access for Key Administrators",
      "Effect": "Allow",
      "Principal": {
```

```

        "AWS": [
            "arn:aws:iam::111122223333:role/Admin"
        ]
    },
    "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",
        "kms>Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
    ],
    "Resource": "*"
},
{
    "Sid": "Allow use of the key",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::111122223333:role/DMS-Redshift-endpoint-
access-role",
            "arn:aws:iam::111122223333:role/Admin",
            "arn:aws:iam::111122223333:role/User1"
        ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
},
{
    "Sid": "Allow attachment of persistent resources",

```

```

        "Effect": "Allow",
        "Principal": {
            "AWS": [
                "arn:aws:iam::111122223333:role/DMS-Redshift-endpoint-
access-role",
                "arn:aws:iam::111122223333:role/Admin",
                "arn:aws:iam::111122223333:role/User1"
            ]
        },
        "Action": [
            "kms:CreateGrant",
            "kms:ListGrants",
            "kms:RevokeGrant"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {
                "kms:GrantIsForAWSResource": true
            }
        }
    }
}

```

13. 마침을 클릭합니다. 가 생성되었음을 나타내는 메시지와 함께 암호화 키 페이지 AWS KMS key 가 열립니다.

이제 지정된 별칭의 새 KMS 키가 생성되었습니다(예: DMS-Redshift-endpoint-encryption-key). 이 키를 사용하면 Amazon Redshift 대상 데이터를 암호화 AWS DMS 할 수 있습니다.

Amazon Redshift를의 대상으로 사용 시 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 마찬가지로 엔드포인트 설정을 사용하여 Amazon Redshift 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --redshift-settings '{"EndpointSetting": "value", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Amazon Redshift를 대상으로 하여 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

이름	설명
MaxFileSize	데이터를 Amazon Redshift로 전송할 때 사용되는 .csv 파일의 최대 크기(단위: KB)를 지정합니다. 기본값: 32768KB(32MB) 유효한 값: 1~1048576 예시: <code>--redshift-settings '{"MaxFileSize": 512}'</code>
FileTransferUploadStreams	단일 파일을 업로드할 때 사용되는 스레드 수를 지정합니다. 기본값: 10 유효값: 1~64 예시: <code>--redshift-settings '{"FileTransferUploadStreams": 20}'</code>
Acceptanydate	0000-00-00과 같이 잘못된 날짜 형식을 포함하여 모든 날짜 형식을 수락할지 여부를 지정합니다. 부울 값입니다. 기본값: false 유효한 값: true false(부울) 예시: <code>--redshift-settings '{"Acceptanydate": true}'</code>
Dateformat	날짜 형식을 지정합니다. 문자열 입력이며 기본적으로 비어 있습니다. 기본 형식은 YYYY-MM-DD이지만 DD-MM-YYYY 등으로 변경할 수 있습니다. 날짜 또는 시간 값에 다른 형식을 사용할 경우 auto 인수와 함께 Dateformat 파라미터를 사용하세요. auto 인수는 Dateformat 문자열을 사용할 때 지원되지 않는 몇 가지 형식을 인식합니다. auto 키워드는 대/소문자를 구별합니다.

이름	설명
	기본값: 비어 있음 유효값: <i>'dateformat_string'</i> 또는 auto 예제: <code>--redshift-settings '{"Dateformat": "auto"}'</code>
Timeformat	시간 형식을 지정합니다. 문자열 입력이며 기본적으로 비어 있습니다. auto 인수는 Timeformat 문자열을 사용할 때 지원되지 않는 몇 가지 형식을 인식합니다. 날짜 및 시간 값에 사용되는 형식이 서로 다를 경우 auto 인수와 함께 Timeformat 파라미터를 사용하세요. 기본값: 10 유효값: <i>'Timeformat_string'</i> 'auto' 'epochsecs' 'epochmillisecs' 예제: <code>--redshift-settings '{"Timeformat": "auto"}'</code>
Emptyasnull	가 빈 CHAR 및 VARCHAR 필드를 null로 마이그레이션 AWS DMS 할지 여부를 지정합니다. true 값은 빈 CHAR 및 VARCHAR 필드를 null로 설정합니다. 기본값: false 유효값: true/false 예시: <code>--redshift-settings '{"Emptyasnull": true}'</code>

이름	설명
TruncateColumns	열의 데이터를 열 명세에 따라 적합한 수의 문자로 자릅니다. VARCHAR 또는 CHAR 데이터 형식의 열에만 적용되며, 행의 크기는 4MB 이하입니다. 기본값: false 유효값: true/false 예시: <code>--redshift-settings '{"TruncateColumns": true}'</code>
RemoveQuotes	입력 데이터의 문자열에서 묶고 있는 인용 부호를 제거합니다. 인용 부호 안의 문자는 구분자를 포함하여 모두 유지됩니다. Amazon Redshift 대상의 따옴표 제거에 관한 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서를 참조하세요. 기본값: false 유효값: true/false 예시: <code>--redshift-settings '{"RemoveQuotes": true}'</code>
TrimBlanks	VARCHAR 문자열에서 후행 공백 문자를 제거합니다. 이 파라미터는 VARCHAR 데이터 형식의 열에만 적용됩니다. 기본값: false 유효값: true/false 예시: <code>--redshift-settings '{"TrimBlanks": true}'</code>

이름	설명
EncryptionMode	Amazon Redshift로 복사되기 전에 데이터를 S3로 푸시하는 데 사용할 서버 측 암호화 모드를 지정합니다. 유효 값은 SSE_S3(S3 서버 측 암호화) 또는 SSE_KMS(KMS 키 암호화)입니다. SSE_KMS를 선택하는 경우, ServerSideEncryptionKmsKeyId 파라미터를 암호화에 사용할 KMS 키의 Amazon 리소스 이름(ARN)으로 설정합니다.  Note 또한 CLI modify-endpoint 명령을 사용하여 기존 엔드포인트의 EncryptionMode 설정 값을 SSE_KMS에서 SSE_S3로 변경할 수 있습니다. 그러나 EncryptionMode 값을 SSE_S3에서 SSE_KMS로 변경할 수 없습니다. 기본값: SSE_S3 유효값: SSE_S3 또는 SSE_KMS 예시: <code>--redshift-settings '{"EncryptionMode": "SSE_S3"}'</code>
ServerSideEncryptionKmsKeyId	EncryptionMode 를 SSE_KMS로 설정하는 경우, 이 파라미터를 KMS 키의 ARN으로 설정합니다. 계정에 대해 생성된 키 목록에서 AWS KMS 키 별칭을 선택하여 이 ARN을 찾을 수 있습니다. 키를 생성할 때 특정 정책과 역할을 키에 연결해야 합니다. 자세한 내용은 AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화 단원을 참조하십시오. 예시: <code>--redshift-settings '{"ServerSideEncryptionKmsKeyId":"arn:aws:kms:us-east-1:111122223333:key/11a1a1a1-aaaa-9999-abab-2bbbbbb222a2"}'</code>

이름	설명
EnableParallelBatchInMemoryCSVFiles	이 EnableParallelBatchInMemoryCSVFiles 설정은 DMS가 메모리 대신 디스크에 쓰도록 함으로써 대규모 다중 스레드 전체 로드 작업의 성능을 향상시킵니다. 기본값은 false입니다.
CompressCsvFiles	이 속성을 사용하면 마이그레이션 중에 Amazon Redshift 대상으로 전송되는 데이터를 압축할 수 있습니다. 기본값은 true이며 압축은 기본적으로 활성화되어 있습니다.
CaseSensitiveNames	Redshift 대상 엔드포인트로 작업할 때 식별자(예: 테이블 및 열 이름)의 대/소문자 구분 처리를 활성화할지 여부를 지정합니다. 이 설정은 true 파라미터가 기본적으로 enable_case_sensitive_identifier 로 설정된 제로 ETL Redshift 데이터베이스로 작업할 때 특히 중요합니다. CaseSensitiveNames 이 true로 설정되면 DMS 는 마이그레이션 중에 식별자의 대/소문자 구분을 유지합니다. 기본값: false 유효한 값: true false(부울) 예시: --redshift-settings '{"CaseSensitiveNames": true}'

데이터 암호화 키와 Amazon S3 버킷을 중간 스토리지로 사용

Amazon Redshift 대상 엔드포인트 설정을 사용하여 다음을 구성할 수 있습니다.

- 사용자 지정 AWS KMS 데이터 암호화 키입니다. 그런 다음, 이 키를 사용하여 Amazon Redshift에 복사되기 전에 Amazon S3로 푸시되는 데이터를 암호화할 수 있습니다.
- Amazon Redshift로 마이그레이션한 데이터의 중간 스토리지인 사용자 지정 S3 버킷.
- PostgreSQL 소스에서 boolean을 부울로 매핑합니다. 기본적으로 부울(BOOLEAN) 형식은 varchar(1)로 마이그레이션됩니다. 다음 예와 같이 Redshift 대상이 부울 형식을 부울로 마이그레이션하도록 MapBooleanAsBoolean을 지정할 수 있습니다.

```
--redshift-settings '{"MapBooleanAsBoolean": true}'
```

단, 이 설정이 적용되려면 소스 엔드포인트와 대상 엔드포인트 모두에서 이 설정을 지정해야 합니다.

데이터 암호화를 위한 KMS 키 설정

다음 예는 S3로 푸시되는 데이터를 암호화하도록 사용자 지정 KMS 키를 구성하는 방법을 보여 줍니다. 먼저 AWS CLI에서 다음 `create-endpoint` 호출을 할 수 있습니다.

```
aws dms create-endpoint --endpoint-identifier redshift-target-endpoint --endpoint-type
target
--engine-name redshift --username your-username --password your-password
--server-name your-server-name --port 5439 --database-name your-db-name
--redshift-settings '{"EncryptionMode": "SSE_KMS",
"ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-east-1:111122223333:key/24c3c5a1-
f34a-4519-a85b-2debbef226d1"}'
```

여기서 `--redshift-settings` 옵션에 의해 지정된 JSON 객체는 다음 두 가지 파라미터를 정의합니다. 하나는 값이 `SSE_KMS`인 `EncryptionMode` 파라미터입니다. 다른 하나는 값이 `arn:aws:kms:us-east-1:111122223333:key/24c3c5a1-f34a-4519-a85b-2debbef226d1`인 `ServerSideEncryptionKmsKeyId` 파라미터입니다. 이 값은 사용자 지정 KMS 키의 Amazon 리소스 이름(ARN)입니다.

기본적으로 S3 데이터 암호화는 S3 서버 측 암호화를 사용하여 수행됩니다. 이전 예의 Amazon Redshift 대상의 경우, 다음 예에서처럼 이것은 엔드포인트 설정을 지정하는 것에 해당합니다.

```
aws dms create-endpoint --endpoint-identifier redshift-target-endpoint --endpoint-type
target
--engine-name redshift --username your-username --password your-password
--server-name your-server-name --port 5439 --database-name your-db-name
--redshift-settings '{"EncryptionMode": "SSE_S3"}'
```

S3 서버측 암호화 사용에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 [서버 측 암호화를 사용하여 데이터 보호](#)를 참조하세요.

Note

또한 CLI `modify-endpoint` 명령을 사용하여 기존 엔드포인트의 `EncryptionMode` 파라미터 값을 `SSE_KMS`에서 `SSE_S3`로 변경할 수 있습니다. 그러나 `EncryptionMode` 값을 `SSE_S3`에서 `SSE_KMS`로 변경할 수 없습니다.

Amazon S3 버킷 설정

Amazon Redshift 대상 엔드포인트로 데이터를 마이그레이션할 때는 마이그레이션된 데이터를 Amazon Redshift로 복사하기 전에 기본 Amazon S3 버킷을 중간 작업 스토리지로 AWS DMS 사용합니다. 예를 들어 AWS KMS 데이터 암호화 키가 있는 Amazon Redshift 대상 엔드포인트를 생성하는 예제에서는 이 기본 S3 버킷을 사용합니다([데이터 암호화를 위한 KMS 키 설정](#) 참조).

대신 `create-endpoint` 명령의 `--redshift-settings` 옵션 값에 다음 파라미터를 포함하여이 중간 스토리지에 대한 사용자 지정 S3 버킷을 AWS CLI 지정할 수 있습니다.

- `BucketName` – S3 버킷 스토리지의 이름으로 지정하는 문자열. 서비스 액세스 역할이 `AmazonDMSRedshiftS3Role` 정책을 기반으로 하는 경우, 이 값 앞에는 `dms-` 접두사(예를 들면 `dms-my-bucket-name`)가 붙어야 합니다.
- `BucketFolder` – (선택 사항) 지정된 S3 버킷의 스토리지 폴더 이름으로 지정할 수 있는 문자열.
- `ServiceAccessRoleArn` – S3 버킷에 대한 관리 액세스를 허용하는 IAM 역할의 ARN. 일반적으로 `AmazonDMSRedshiftS3Role` 정책에 따라 이 역할을 만듭니다. 예를 들어 [AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화](#)에서 필수 AWS관리형 정책을 사용하여 IAM 역할을 생성하는 절차를 참조하세요.

Note

`create-endpoint` 명령의 `--service-access-role-arn` 옵션을 사용하여 다른 IAM 역할의 ARN을 지정하면 이 IAM 역할 옵션이 우선 적용됩니다.

다음 예에서는 AWS CLI를 사용하는 다음 `create-endpoint` 호출에서 이러한 파라미터를 사용하여 사용자 지정 Amazon S3 버킷을 지정하는 방법을 보여줍니다.

```
aws dms create-endpoint --endpoint-identifier redshift-target-endpoint --endpoint-type
target
--engine-name redshift --username your-username --password your-password
```

```
--server-name your-server-name --port 5439 --database-name your-db-name
--redshift-settings '{"ServiceAccessRoleArn": "your-service-access-ARN",
"BucketName": "your-bucket-name", "BucketFolder": "your-bucket-folder-name"}'
```

Amazon Redshift에 대한 멀티스레드 작업 설정

다중 스레드 작업 설정을 사용하여 Amazon Redshift 대상 엔드포인트에 대한 전체 로드 및 변경 데이터 캡처(CDC) 작업의 성능을 개선할 수 있습니다. 이를 통해 버퍼에 저장할 동시 스레드 수와 레코드 수를 지정할 수 있습니다.

Amazon Redshift에 대한 멀티스레드 전체 로드 작업 설정

전체 로드 성능을 높일 수 있도록 다음과 같은 ParallelLoad* 작업 설정을 사용할 수 있습니다.

- **ParallelLoadThreads** – 데이터 레코드를 Amazon Redshift 대상 엔드포인트로 푸시하기 위해 전체 로드 중에 DMS가 사용하는 동시 스레드의 수를 지정합니다. 기본값은 0이고 최대값은 32입니다. 자세한 내용은 [전체 로드 작업 설정](#) 단원을 참조하십시오.

ParallelLoadThreads 작업 설정을 사용할 때 false로 설정된

enableParallelBatchInMemoryCSVFiles 속성을 사용할 수 있습니다. 이 속성은 DMS가 메모리 대신 디스크에 쓰기를 수행함으로써 대규모 멀티스레드 전체 로드 작업의 성능을 향상시킵니다. 기본값은 true입니다.

- **ParallelLoadBufferSize** – Redshift 대상과 함께 병렬 로드 스레드를 사용하는 동안 최대 데이터 레코드 요청을 지정합니다. 기본값은 100이고 최대값은 1,000입니다. ParallelLoadThreads > 1(1보다 큼)인 경우 이 옵션을 사용하는 것이 좋습니다.

Note

FULL LOAD에서 Amazon Redshift 대상 엔드포인트로의 ParallelLoad* 작업 설정 사용에 대한 지원은 AWS DMS 버전 3.4.5 이상에서 사용할 수 있습니다.

ReplaceInvalidChars Redshift 엔드포인트 설정은 변경 데이터 캡처(CDC) 중이거나 또는 병렬 로드가 활성화된 전체 로드(FULL LOAD) 마이그레이션 작업 중에는 지원되지 않습니다. 병렬 로드가 활성화되지 않은 경우, 이 설정은 전체 로드(FULL LOAD) 마이그레이션에 대해 지원됩니다. 자세한 내용은 AWS Database Migration Service API 참조의 [RedshiftSettings](#)를 참조하세요.

Amazon Redshift에 대한 멀티스레드 CDC 작업 설정

CDC 성능을 높일 수 있도록 다음과 같은 ParallelApply* 작업 설정을 사용할 수 있습니다.

- **ParallelApplyThreads** - CDC 로드 중에가 데이터 레코드를 Amazon Redshift 대상 엔드포인트로 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다. 최소 권장값은 클러스터 내 조각 수와 같습니다.
- **ParallelApplyBufferSize** - Redshift 대상과 함께 병렬 적용 스레드를 사용하는 동안 최대 데이터 레코드 요청을 지정합니다. 기본값은 100이고 최대값은 1,000입니다. ParallelApplyThreads > 1(1보다 큼)인 경우 이 옵션을 사용하는 것이 좋습니다.

대상으로서 Redshift의 이점을 최대한 활용하려면 ParallelApplyBufferSize의 값을 ParallelApplyThreads 값의 두 배 이상으로 설정하는 것이 좋습니다.

Note

CDC에서 Amazon Redshift 대상 엔드포인트로의 ParallelApply* 작업 설정 사용에 대한 지원은 AWS DMS 버전 3.4.3 이상에서 사용할 수 있습니다.

적용되는 병렬화 수준은 전체 배치 크기와 데이터 전송에 사용된 최대 파일 크기 간의 상관 관계에 따라 달라집니다. 최대 파일 크기에 비해 배치 크기가 클 때 Redshift 대상과 함께 멀티스레드 CDC 작업 설정을 사용하면 이점이 있습니다. 예를 들어, 다음과 같은 엔드포인트 및 작업 설정 조합을 사용하여 최적의 성능을 발휘하도록 조정할 수 있습니다.

```
// Redshift endpoint setting

MaxFileSize=250000;

// Task settings

BatchApplyEnabled=true;
BatchSplitSize =8000;
BatchApplyTimeoutMax =1800;
BatchApplyTimeoutMin =1800;
ParallelApplyThreads=32;
ParallelApplyBufferSize=100;
```

이전 예제의 설정을 사용하면 트랜잭션 워크로드가 많은 고객은 8000개의 레코드를 포함하는 배치 버퍼를 1800초 내에 채우고 최대 파일 크기가 250MB인 병렬 스레드 32개를 활용하여 이점을 얻을 수 있습니다.

자세한 내용은 [변경 처리 튜닝 설정](#) 단원을 참조하십시오.

Note

Redshift 클러스터로 계속 복제하는 동안 실행되는 DMS 쿼리는 실행 중인 다른 애플리케이션 쿼리와 동일한 WLM(워크로드 관리) 대기열을 공유할 수 있습니다. 따라서 Redshift 대상으로 계속 복제하는 동안 성능에 영향을 미치도록 WLM 속성을 적절하게 구성하는 것이 좋습니다. 예를 들어, 다른 병렬 ETL 쿼리가 실행 중인 경우 DMS는 더 느리게 실행되며 성능 향상은 사라집니다.

Amazon Redshift에 대한 대상 데이터 형식

용 Amazon Redshift 엔드포인트는 대부분의 Amazon Redshift 데이터 유형을 AWS DMS 지원합니다. 다음 표에는 사용 시 지원되는 Amazon Redshift 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

AWS DMS 데이터 유형	Amazon Redshift 데이터 형식
BOOLEAN	BOOL
BYTES	VARCHAR(길이)
DATE	DATE
TIME	VARCHAR(20)
DATETIME	Redshift 대상 열 유형에 따라 크기가 0 이상이고 6 이하라면 다음 중 하나가 적용됩니다. TIMESTAMP (s)

AWS DMS 데이터 유형	Amazon Redshift 데이터 형식
	TIMESTAMPTZ (s) — 소스 타임스탬프에 (예를 들면 SQL Server 또는 Oracle 내) 영역 오프셋이 포함된 경우, 삽입/업데이트 시 UTC로 변환됩니다. 오프셋이 없으면 시간은 이미 UTC로 간주됩니다. 크기가 7 이상이고 9 이하인 경우: VARCHAR(37)
INT1	INT2
INT2	INT2
INT4	INT4
INT8	INT8
NUMERIC	크기가 0 이상이고 37 이하인 경우: NUMERIC(p,s) 크기가 38 이상이고 127 이하인 경우: VARCHAR(길이)
REAL4	FLOAT4
REAL8	FLOAT8
STRING	길이가 1~65,535인 경우, VARCHAR(바이트 단위의 길이)를 사용합니다. 길이가 65,536~2,147,483,647인 경우, VARCHAR(65535)를 사용합니다.
UINT1	INT2
UINT2	INT2

AWS DMS 데이터 유형	Amazon Redshift 데이터 형식
UINT4	INT4
UINT8	NUMERIC(20,0)
WSTRING	길이가 1~65,535인 경우, NVARCHAR(바이트 단위의 길이)를 사용합니다. 길이가 65,536~2,147,483,647인 경우, NVARCHAR(65535)를 사용합니다.
BLOB	VARCHAR(최대 LOB 크기 *2) 최대 LOB 크기는 31KB를 초과할 수 없습니다. Amazon Redshift는 64KB를 초과하는 VARCHAR를 지원하지 않습니다.
NCLOB	NVARCHAR(최대 LOB 크기) 최대 LOB 크기는 63KB를 초과할 수 없습니다. Amazon Redshift는 64KB를 초과하는 VARCHAR를 지원하지 않습니다.
CLOB	VARCHAR(최대 LOB 크기) 최대 LOB 크기는 63KB를 초과할 수 없습니다. Amazon Redshift는 64KB를 초과하는 VARCHAR를 지원하지 않습니다.

AWS DMS Amazon Redshift Serverless에서 대상으로 사용

AWS DMS 는 Amazon Redshift Serverless를 대상 엔드포인트로 사용할 수 있도록 지원합니다. Amazon Redshift Serverless 사용에 관한 자세한 내용은 [Amazon Redshift 관리 안내서](#)의 [Amazon Redshift Serverless](#)를 참조하세요.

이 주제에서는 Amazon Redshift Serverless 엔드포인트를와 함께 사용하는 방법을 설명합니다 AWS DMS.

Note

Amazon Redshift Serverless 엔드포인트를 생성할 때는 [RedshiftSettings](#) 엔드포인트 구성의 DatabaseName 필드에 Amazon Redshift 데이터 웨어하우스의 이름 또는 워크그룹 엔드포인트의 이름을 사용합니다. ServerName 필드에는 서버리스 클러스터의 Workgroup 페이지에 표시된 Endpoint 값(예: default-workgroup.093291321484.us-east-1.redshift-serverless.amazonaws.com)을 사용합니다. 엔드포인트 생성에 관한 자세한 내용은 [소스 및 대상 엔드포인트 생성](#) 섹션을 참조하세요. 워크그룹 엔드포인트에 관한 자세한 내용은 [Amazon Redshift Serverless에 연결](#)을 참조하세요.

대상으로서 Amazon Redshift Serverless를 사용하는 신뢰 정책

Amazon Redshift Serverless를 대상 엔드포인트로 사용하는 경우, 다음과 같이 강조 표시된 섹션을 신뢰 정책에 추가해야 합니다. 이 신뢰 정책은 dms-access-for-endpoint 역할에 연결됩니다.

에서 신뢰 정책을 사용하는 방법에 대한 자세한 내용은 섹션을 [AWS DMS참조하세요와 함께 사용할 IAM 역할 생성 AWS DMS](#).

대상으로서 Amazon Redshift Serverless 사용 시 제한 사항

Redshift Serverless를 대상으로 사용할 경우 다음과 같은 제한 사항이 있습니다.

- AWS DMS 는 Amazon Redshift Serverless를 지원하는 리전에서만 Amazon Redshift Serverless를 엔드포인트로 지원합니다. Amazon Redshift Serverless를 지원하는 리전에 관한 자세한 내용은 [AWS 일반 참조](#)의 [Amazon Redshift 엔드포인트 및 할당량](#) 항목에 있는 Redshift Serverless API를 참조하세요.
- Enhanced VPC Routing을 사용할 때는 Redshift Serverless 또는 Redshift Provisioned 클러스터와 동일한 VPC에 Amazon S3 엔드포인트를 생성해야 합니다. 자세한 내용은 [Amazon Redshift에서의 대상으로 향상된 VPC 라우팅 사용 AWS Database Migration Service](#) 단원을 참조하십시오.
- AWS DMS 는 Amazon Redshift Serverless에 대한 향상된 처리량을 대상으로 지원하지 않습니다. 자세한 내용은 [Oracle에서 Amazon Redshift 및 Amazon S3로의 전체 로드 마이그레이션을 위한 향상된 처리량](#) 단원을 참조하십시오.
- AWS DMS 는 SSL 모드가 로 설정된 경우 Amazon Redshift Redshift Serverless에 대한 연결을 지원하지 않습니다. verify-full. Amazon Redshift Serverless 대상에 대한 SSL 확인이 필요한 연결의 경우 require 또는와 같은 대체 SSL 모드를 사용합니다. verify-ca.

SAP ASE 데이터베이스를의 대상으로 사용 AWS Database Migration Service

지원되는 데이터베이스 소스 중 AWS DMS 하나를 사용하여 이전에는 Sybase-databases라고 하는 SAP Adaptive Server Enterprise(ASE)로 데이터를 마이그레이션할 수 있습니다.

가 대상으로 AWS DMS 지원하는 SAP ASE 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 대상 AWS DMS](#).

SAP ASE 데이터베이스를의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service

SAP ASE 데이터베이스를 대상으로 사용하기 전에 다음 사전 요구 사항이 있는지 AWS DMS 확인합니다.

- AWS DMS 사용자에게 SAP ASE 계정 액세스 권한을 제공합니다. 이 사용자는 SAP ASE 데이터베이스에서 읽기/쓰기 권한이 있어야 합니다.
- 경우에 따라 라틴 문자 외 다른 문자(예: 중국어)로 구성된 Amazon EC2 인스턴스에 설치된 SAP ASE 버전 15.7로 복제할 수 있습니다. 이러한 경우 AWS DMS에서는 대상 SAP ASE 시스템에 SAP ASE 15.7 SP121을 설치해야 합니다.

SAP ASE 데이터베이스를의 대상으로 사용할 때의 제한 사항 AWS DMS

SAP ASE 데이터베이스를 AWS DMS의 대상으로 사용 시 다음과 같은 제한 사항이 적용됩니다.

- AWS DMS 는 다음 데이터 형식의 필드가 포함된 테이블을 지원하지 않습니다. 이 데이터 형식을 사용하는 복제된 열은 null로 표시됩니다.
 - 사용자 정의 유형(UDT)

SAP ASE를의 대상으로 사용할 때 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 SAP ASE 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --sybase-settings '{"**EndpointSetting**": "**value**", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

SAP ASE를 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

이름	설명
Driver	ASE 15.7 및 이후 버전에 TLS를 사용하려면 이 속성을 설정하세요. 기본값: Adaptive Server Enterprise 예시: driver=Adaptive Server Enterprise 16.03.06; 유효값: Adaptive Server Enterprise 16.03.06
AdditionalConnectionProperties	지정할 추가 ODBC 연결 파라미터입니다.

SAP ASE용 대상 데이터 형식

다음 표에는를 사용할 때 지원되는 SAP ASE 데이터베이스 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

AWS DMS 데이터 형식	SAP ASE 데이터 형식
BOOLEAN	BIT
BYTES	VARBINARY(길이)
DATE	DATE
TIME	TIME
TIMESTAMP	크기가 0 이상이고 6 이하인 경우: BIGDATETIME 크기가 7 이상이고 9 이하인 경우: VARCHAR(37)

AWS DMS 데이터 형식	SAP ASE 데이터 형식
INT1	TINYINT
INT2	SMALLINT
INT4	INTEGER
INT8	BIGINT
NUMERIC	NUMERIC(p,s)
REAL4	REAL
REAL8	DOUBLE PRECISION
STRING	VARCHAR(길이)
UINT1	TINYINT
UINT2	UNSIGNED SMALLINT
UINT4	UNSIGNED INTEGER
UINT8	UNSIGNED BIGINT
WSTRING	VARCHAR(길이)
BLOB	IMAGE
CLOB	UNITEXT
NCLOB	TEXT

Amazon S3를 대상으로 사용 AWS Database Migration Service

지원되는 데이터베이스 소스에서 AWS DMS 를 사용하여 Amazon S3로 데이터를 마이그레이션할 수 있습니다. Amazon S3를 AWS DMS 작업의 대상으로 사용하는 경우 전체 로드 및 변경 데이터 캡처 (CDC) 데이터는 모두 기본적으로 쉼표로 구분된 값(.csv) 형식으로 기록됩니다. 보다 간소화된 스토리지 및 보다 빠른 쿼리 옵션을 위해 데이터를 Apache Parquet(.parquet) 형식에 작성할 수도 있습니다.

AWS DMS 는 LOAD00001.csv, LOAD00002..., LOAD00009, LOAD0000A 등과 같은 중분 16진수 카운터를 사용하여 전체 로드 중에 생성된 파일의 이름을 지정합니다.는 20141029-1134010000.csv와 같은 타임스탬프를 사용하여 CDC 파일의 AWS DMS 이름을 지정합니다. 레코드가 포함된 각 소스 테이블에 대해서는 지정된 대상 폴더 아래에 폴더를 AWS DMS 생성합니다(소스 테이블이 비어 있지 않은 경우).는 모든 전체 로드 및 CDC 파일을 지정된 Amazon S3 버킷에 AWS DMS 씁니다. [MaxFileSize](#) 엔드포인트 설정을 사용하여가 AWS DMS 생성하는 파일의 크기를 제어할 수 있습니다.

파라미터 bucketFolder에는 S3 버킷에 업로드되기 전에 .csv 또는 .parquet 파일이 저장되는 위치가 포함되어 있습니다. .csv 파일의 경우, 테이블 데이터는 S3 버킷에 다음 형식으로 저장되며 전체 로드 파일과 함께 표시됩니다.

```
database_schema_name/table_name/LOAD000000001.csv
database_schema_name/table_name/LOAD000000002.csv
...
database_schema_name/table_name/LOAD000000009.csv
database_schema_name/table_name/LOAD00000000A.csv
database_schema_name/table_name/LOAD00000000B.csv
...database_schema_name/table_name/LOAD00000000F.csv
database_schema_name/table_name/LOAD000000010.csv
...
```

추가 연결 속성을 사용하여 열 구분 기호, 행 구분 기호 등 여러 파라미터를 지정할 수 있습니다. 추가 연결 속성에 관한 자세한 내용은 이 섹션의 마지막에 나오는 [Amazon S3를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#)을 참조하세요.

스푸핑을 방지하기 위해는 작업을 수행하기 전에 버킷 소유권을 AWS DMS 확인합니다. 기본적으로 ExpectedBucketOwner Amazon S3 엔드포인트 설정이 지정되지 않은 경우는 AWS DMS 서비스 역할을 소유한 AWS 계정 ID를 예상 버킷 소유자로 AWS DMS 사용합니다.

다른 AWS 계정이 소유한 S3 버킷으로 데이터를 마이그레이션하려면 다음과 같이 ExpectedBucketOwner Amazon S3 엔드포인트 설정에서 실제 버킷 소유자를 명시적으로 지정해야 합니다. 그렇지 않으면 교차 계정 복제 작업이 실패합니다.

```
--s3-settings '{"ExpectedBucketOwner": "AWS_Account_ID"}'
```

AWS DMS 를 사용하여 CDC 작업을 사용하여 데이터 변경 사항을 복제하는 경우 .csv 또는 .parquet 출력 파일의 첫 번째 열은 다음 .csv 파일에 표시된 대로 행 데이터가 어떻게 변경되었는지를 나타냅니다.

```
I,101,Smith,Bob,4-Jun-14,New York
U,101,Smith,Bob,8-Oct-15,Los Angeles
U,101,Smith,Bob,13-Mar-17,Dallas
D,101,Smith,Bob,13-Mar-17,Dallas
```

이 예제에서는 소스 데이터베이스에 EMPLOYEE 테이블이 있다고 가정합니다.는 다음 이벤트에 대한 응답으로 .csv 또는 .parquet 파일에 데이터를 AWS DMS 씁니다.

- 새 직원(Bob Smith, 직원 ID 101)이 2014년 6월 4일에 뉴욕 지사에 입사했습니다. .csv 또는 .parquet 파일에서 첫 번째 열의 I는 새 행이 소스 데이터베이스의 EMPLOYEE 테이블에 삽입(INSERT)되었음을 나타냅니다.
- 2015년 10월 8일에 Bob은 LA 지사로 전근갑니다. .csv 또는 .parquet 파일에서 U는 Bob의 새 지사 위치를 반영하여 EMPLOYEE 테이블의 해당 행이 업데이트(UPDATE)되었음을 나타냅니다. 이 줄 뒷부분에는 UPDATE 이후에 EMPLOYEE 테이블에서 이 행이 어떻게 표시되는지가 반영됩니다.
- 2017년 3월 13일에 Bob은 Dallas 지사로 다시 전근갑니다. .csv 또는 .parquet 파일에서 U는 이 행이 다시 업데이트(UPDATE)되었음을 나타냅니다. 이 줄 뒷부분에는 UPDATE 이후에 EMPLOYEE 테이블에서 이 행이 어떻게 표시되는지가 반영됩니다.
- Bob은 Dallas에서 얼마간 근무한 후 퇴사합니다. .csv 또는 .parquet 파일에서 D는 소스 테이블에서 이 행이 삭제(DELETE)되었음을 나타냅니다. 이 줄의 나머지는 삭제 전에 EMPLOYEE 테이블에서 이 행이 어떻게 표시되었는지가 반영됩니다.

CDC의 경우 기본적으로는 트랜잭션 순서에 관계없이 각 데이터베이스 테이블의 행 변경 사항을 AWS DMS 저장합니다. 트랜잭션 순서에 따라 행 변경 내용을 CDC 파일에 저장하려면 S3 엔드포인트 설정을 사용하여 이 경로는 물론, CDC 트랜잭션 파일을 S3 대상에 저장할 폴더 경로도 지정해야 합니다. 자세한 내용은 [S3 대상의 트랜잭션 순서를 포함한 변경 데이터 캡처\(CDC\)](#) 단원을 참조하십시오.

데이터 복제 작업 중에 Amazon S3 대상에 대한 쓰기 빈도를 제어하기 위해 cdcMaxBatchInterval 및 cdcMinFileSize 추가 연결 속성을 구성할 수 있습니다. 따라서 추가 오버헤드 작업 없이 데이터를 분석할 때 성능이 향상될 수 있습니다. 자세한 내용은 [Amazon S3를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요.

주제

- [Amazon S3를 대상으로 사용하기 위한 사전 조건](#)
- [Amazon S3를 대상으로 사용할 때 적용되는 제한 사항](#)
- [보안](#)
- [Apache Parquet를 사용하여 Amazon S3 객체 저장](#)

- [Amazon S3 객체 태그 지정](#)
- [Amazon S3 대상 객체를 암호화하는 AWS KMS 키 생성](#)
- [날짜 기반 폴더 파티셔닝 사용](#)
- [Amazon S3를의 대상으로 사용할 때 분할된 소스의 병렬 로드 AWS DMS](#)
- [Amazon S3를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#)
- [예 Amazon S3 대상과 AWS Glue Data Catalog 함께 사용 AWS DMS](#)
- [Amazon S3 대상에서 데이터 암호화, parquet 파일 및 CDC 사용](#)
- [마이그레이션된 S3 데이터에 소스 DB 작업 표시](#)
- [S3 Parquet의 대상 데이터 유형](#)

Amazon S3를 대상으로 사용하기 위한 사전 조건

Amazon S3를 대상으로 사용하기 전에 다음 사항이 참인지 확인하세요.

- 대상으로 사용하는 S3 버킷은 데이터를 마이그레이션하는 데 사용하는 DMS 복제 인스턴스와 동일한 AWS 리전에 있습니다.
- 마이그레이션에 사용하는 AWS 계정에는 대상으로 사용 중인 S3 버킷에 대한 쓰기 및 삭제 액세스 권한이 있는 IAM 역할이 있습니다.
- 이 역할에는 태그 지정 액세스 권한이 있으므로 대상 버킷에 작성되는 S3 객체에 태그를 지정할 수 있습니다.
- IAM 역할에는 DMS(dms.amazonaws.com)가 신뢰할 수 있는 엔터티로 추가되었습니다.
- AWS DMS 버전 3.4.7 이상의 경우 DMS는 VPC 엔드포인트 또는 퍼블릭 경로를 통해 소스 버킷에 액세스해야 합니다. VPC 엔드포인트에 대한 자세한 내용은 [예 대한 VPC 엔드포인트 구성 AWS DMS](#) 섹션을 참조하세요.

이 계정의 액세스 권한을 설정하려면 마이그레이션 작업 생성에 사용되는 사용자 계정에 할당된 역할에 다음 권한 집합이 있어야 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```



```

        "Action": [
            "s3:PutObject",
            "s3:DeleteObject",
            "s3:PutObjectTagging"
        ],
        "Resource": [
            "arn:aws:s3:::buckettest2/*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:ListBucket"
        ],
        "Resource": [
            "arn:aws:s3:::buckettest2"
        ]
    }
]
}

```

S3를 대상으로 하는 검증을 사용하기 위한 사전 요구 사항은 [S3 대상 검증 사전 조건](#)를 참조하세요.

Amazon S3를 대상으로 사용할 때 적용되는 제한 사항

Amazon S3를 대상으로 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- S3의 버전 관리를 활성화하지 마세요. S3 버전 관리가 필요한 경우, 수명 주기 정책을 사용하여 이전 버전을 적극적으로 삭제하세요. 그렇지 않으면 S3 list-object 직접 호출 시간 초과로 인해 엔드 포인트 테스트 연결 실패가 발생할 수 있습니다. S3 버킷의 수명 주기 정책을 생성하려면 [스토리지 수명 주기 관리](#)를 참조하세요. S3 객체의 버전을 삭제하려면 [버전 관리 활성화 버킷에서 객체 버전 삭제](#)를 참조하세요.
- VPC 지원(게이트웨이 VPC) S3 버킷은 버전 3.4.7 및 이후 버전에서 지원됩니다.
- 변경 데이터 캡처(CDC)에는 테이블 자르기(Truncate Table), 테이블 삭제(Drop Table), 테이블 생성(Create Table), 테이블 이름 바꾸기(Rename Table), 열 추가(Add Column), 열 삭제(Drop Column), 열 이름 바꾸기(Rename Column), 열 데이터 유형 변경(Change Column Data Type)과 같은 데이터 정의 언어(DDL) 명령이 지원됩니다. 소스 데이터베이스에 열이 추가, 삭제 또는 이름 변경되면 대상 S3 버킷에 ALTER 문이 기록되지 않으며 이전에 생성된 레코드가 새 구조와 일치하도록 변경되지 AWS DMS 않습니다. 변경 후는 새 테이블 구조를 사용하여 새 레코드를 AWS DMS 생성합니다.

Note

DDL 잘라내기(truncate DDL) 작업을 하면 S3 버킷에서 모든 파일 및 해당 테이블 폴더가 제거됩니다. 작업 설정을 사용하여 해당 동작을 비활성화하고 변경 데이터 캡처(CDC) 중에 DMS가 DDL 동작을 처리하는 방식을 구성할 수 있습니다. 자세한 내용은 [변경 처리 DDL을 다루기 위한 작업 설정](#) 단원을 참조하십시오.

- 전체 LOB 모드는 지원되지 않습니다.
- 전체 로드 중에 소스 테이블의 구조 변경은 지원되지 않습니다. 전체 로드 중에 데이터 변경은 지원됩니다.
- 동일한 소스 테이블에서 동일한 대상 S3 엔드포인트 버킷으로 데이터를 복제하는 작업을 여러 번 수행하면 동일한 파일에 해당 작업을 쓰게 됩니다. 동일한 테이블의 데이터 소스인 경우 서로 다른 대상 엔드포인트(버킷)를 지정하는 것이 좋습니다.
- BatchApply는 S3 엔드포인트에는 지원되지 않습니다. S3 대상에 대해 Batch Apply(예: BatchApplyEnabled 대상 메타데이터 작업 설정)를 사용하면 데이터 손실이 발생할 수 있습니다.
- DatePartitionEnabled 또는 addColumnName를 PreserveTransactions 또는 CdcPath와 함께 사용할 수 없습니다.
- AWS DMS 는 변환 규칙을 사용하여 여러 소스 테이블을 동일한 대상 폴더로 이름을 바꾸는 것을 지원하지 않습니다.
- 전체 로드 단계에서 소스 테이블에 많은 양의 쓰기가 있는 경우 DMS는 S3 버킷이나 캐시된 변경 사항에 중복 레코드를 쓸 수 있습니다.
- TargetTablePrepMode가 DO_NOTHING인 작업을 구성하면 전체 로드 단계에서 작업이 갑자기 중지되었다가 재개되는 경우 DMS가 S3 버킷에 중복 레코드를 쓸 수 있습니다.
- PreserveTransactions를 true로 설정하여 대상 엔드포인트를 구성하는 경우 테이블을 다시 로드해도 이전에 생성된 CDC 파일은 지워지지 않습니다. 자세한 내용은 [S3 대상의 트랜잭션 순서를 포함한 변경 데이터 캡처\(CDC\)](#) 단원을 참조하십시오.

S3를 대상으로 하는 검증을 사용하기 위한 사전 요구 사항은 [S3 대상 검증 사용에 대한 제한 사항](#)을 참조하세요.

보안

Amazon S3를 대상으로 사용하려면 마이그레이션에 사용되는 계정에는 대상으로 사용되는 Amazon S3 버킷에 대한 쓰기 및 삭제 액세스 권한이 있어야 합니다. Amazon S3에 액세스하는 데 필요한 권한을 가진 IAM 역할의 Amazon 리소스 이름(ARN)을 지정합니다.

AWS DMS 는 미리 준비된 ACLs(액세스 제어 목록)이라고 하는 Amazon S3에 대한 사전 정의된 권한 부여 세트를 지원합니다. 각각의 미리 제공된 ACL에는 Amazon S3 버킷에 대한 권한을 설정하는데 사용할 수 있는 피부여자 및 권한 집합이 있습니다. S3 대상 엔드포인트에 대한 연결 문자열 속성에서 `cannedAclForObjects`를 사용하여 미리 제공된 ACL을 지정할 수 있습니다. 추가 연결 속성 `cannedAclForObjects` 사용에 관한 자세한 내용은 [Amazon S3를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요. Amazon S3에서 미리 제공된 ACL에 관한 자세한 내용은 [미리 제공된 ACL](#) 섹션을 참조하세요.

마이그레이션에 사용하는 IAM 역할이 `s3:PutObjectAcl` API 작업을 수행할 수 있어야 합니다.

Apache Parquet를 사용하여 Amazon S3 객체 저장

쉼표로 구분된 값(.csv) 형식이 Amazon S3 대상 객체의 기본 스토리지 형식입니다. 보다 간소화된 스토리지와 보다 빠른 쿼리를 위해 Apache Parquet(.parquet)를 스토리지 형식으로 대신 사용할 수 있습니다.

Apache Parquet는 원래 하둡용으로 개발된 오픈 소스 파일 스토리지 형식입니다. Apache Parquet에 관한 자세한 내용은 <https://parquet.apache.org/>를 참조하세요.

.parquet를 마이그레이션된 S3 대상 객체의 스토리지 형식으로 설정하기 위해 다음 메커니즘을 사용할 수 있습니다.

- AWS CLI 또는 API for AWS DMS를 사용하여 엔드포인트를 생성할 때 JSON 객체의 파라미터로 제공하는 엔드포인트 설정. 자세한 내용은 [Amazon S3 대상에서 데이터 암호화, parquet 파일 및 CDC 사용](#) 단원을 참조하십시오.
- 엔드포인트를 생성할 때 세미콜론으로 구분된 목록으로 제공하는 추가 연결 속성. 자세한 내용은 [Amazon S3를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

Amazon S3 객체 태그 지정

작업-테이블 매핑 규칙의 일환으로 적절한 JSON 객체를 지정하여 복제 인스턴스가 생성하는 Amazon S3 객체에 태그를 지정할 수 있습니다. 유효한 태그 이름을 포함하여 S3 객체 태그 지정의 요구 사항 및 옵션에 관한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 [객체 태그 지정](#)을 참조하세요. JSON을 사용한 테이블 매핑에 관한 자세한 내용은 [JSON을 사용하여 테이블 선택 및 변환 지정](#) 섹션을 참조하세요.

selection 규칙 유형의 JSON 객체를 하나 이상 사용하여 지정된 테이블 및 스키마를 위해 생성된 S3 객체에 태그를 지정합니다. 그런 다음, add-tag 작업을 사용하여 post-processing 규칙 유형의 JSON 객체 중 하나 이상에서 이 selection 객체(또는 객체들)를 따릅니다. 이러한 사후 처리 규칙

은 태그 지정하려는 S3 객체를 식별하고 이러한 S3 객체에 추가하려는 태그의 이름과 값을 지정합니다.

다음 테이블에서 post-processing 규칙 유형의 JSON 객체에 지정할 파라미터를 찾을 수 있습니다.

파라미터	가능한 값	설명
rule-type	post-processing	생성된 대상 객체에 사후 처리 작업을 적용하는 값입니다. 하나 이상의 사후 처리 규칙을 지정하여 선택한 S3 객체에 태그를 지정할 수 있습니다.
rule-id	숫자 값.	규칙을 식별하기 위한 고유한 숫자 값입니다.
rule-name	영숫자 값입니다.	규칙을 식별하기 위한 고유한 이름입니다.
rule-action	add-tag	S3 객체에 적용할 사후 처리 작업입니다. add-tag 작업의 단일 JSON 사후 처리 객체를 사용하여 하나 이상의 태그를 추가할 수 있습니다.
object-locator	schema-name – 테이블 스키마의 이름입니다. table-name – 테이블의 이름입니다.	규칙이 적용되는 각 스키마와 테이블의 이름입니다. 각 object-locator 파라미터 값의 전부 또는 일부에 '%' (퍼센트 기호)를 와일드카드로 사용할 수 있습니다. 따라서 다음 항목들을 일치시킬 수 있습니다. • 단일 스키마의 단일 테이블 • 일부 또는 전체 스키마의 단일 테이블 • 단일 스키마의 일부 또는 전체 테이블 • 일부 또는 전체 스키마의 일부 또는 전체 테이블

파라미터	가능한 값	설명
tag-set	key – 단일 태그의 유효한 이름입니다. value – 이 태그의 유효한 JSON 값입니다.	지정된 object-locator 와 일치하는 생성된 각 S3 객체에 설정하려는 하나 이상의 태그 이름과 값입니다. 하나의 tag-set 파라미터 객체에서 최대 10개의 키-값 페어를 지정할 수 있습니다. S3 객체 태그 지정에 관한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 객체 태그 지정을 참조하세요. <code>\${dyn-value }</code>를 사용하여 태그의 key 및 value 파라미터 값 전부 또는 일부에 동적 값을 지정할 수도 있습니다. 여기서 <code>\${dyn-value }</code>는 <code>\${schema-name}</code> 또는 <code>\${table-name}</code> 일 수 있습니다. 따라서 현재 선택된 스키마 또는 테이블의 이름을 파라미터 값의 전체 또는 일부로 삽입할 수 있습니다.  Note  Important key 파라미터의 동적 값을 삽입하는 경우, 사용 방법에 따라 S3 객체의 중복된 이름이 있는 태그를 생성할 수 있습니다. 이 경우, 중복 태그 설정 중 하나만 객체에 추가됩니다.

여러 post-processing 규칙 유형을 지정하여 일련의 선택된 S3 객체에 태그를 지정하는 경우, 각 S3 객체에는 하나의 사후 처리 규칙으로부터 단 하나의 tag-set 객체만 사용하여 태그가 지정됩니다. 주어진 S3 객체에 태그를 지정하는 데 사용되는 특정 태그 세트는 연결된 객체 로케이터가 해당 S3 객체와 가장 잘 일치하는 사후 처리 규칙의 태그 세트입니다.

예를 들어, 2개의 사후 처리 규칙이 동일한 S3 객체를 식별한다고 가정해 보겠습니다. 또한 한 규칙의 객체 로케이터가 와일드카드를 사용하며 다른 한 규칙의 객체 로케이터는 정확한 일치를 사용하여 S3 객체를 (와일드카드 없이) 식별하는 경우를 가정해 보겠습니다. 이 경우, 정확한 일치를 사용하는 사후 처리 규칙에 연결된 태그 세트가 S3 객체의 태그 지정에 사용됩니다. 여러 사후 처리 규칙이 주어진 S3 객체를 똑같이 잘 일치시키는 경우, 이러한 첫 번째 사후 처리 규칙에 연결된 태그 세트가 객체 태그 지정에 사용됩니다.

Example 단일 테이블 및 스키마를 위해 생성된 S3 객체에 정적 태그 추가

다음 선택 및 사후 처리 규칙은 3개의 태그(tag_1, tag_2, tag_3 및 해당 정적 값 value_1, value_2, value_3)를 생성된 S3 객체에 추가합니다. 이 S3 객체는 스키마 이름이 aat2인 STOCK이라는 이름의 소스에 있는 단일 테이블에 해당됩니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "5",
      "rule-name": "5",
      "object-locator": {
        "schema-name": "aat2",
        "table-name": "STOCK"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "post-processing",
      "rule-id": "41",
      "rule-name": "41",
      "rule-action": "add-tag",
      "object-locator": {
        "schema-name": "aat2",
        "table-name": "STOCK"
      },
      "tag-set": [
        {
          "key": "tag_1",
```

```

        "value": "value_1"
      },
      {
        "key": "tag_2",
        "value": "value_2"
      },
      {
        "key": "tag_3",
        "value": "value_3"
      }
    ]
  }
}

```

Example 여러 테이블 및 스키마를 위해 생성된 S3 객체에 정적 및 동적 태그 추가

다음 예에는 1개의 선택 규칙과 2개의 사후 처리 규칙이 있습니다. 여기서 소스의 입력에는 모든 테이블과 테이블의 모든 스키마가 포함됩니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "post-processing",
      "rule-id": "21",
      "rule-name": "21",
      "rule-action": "add-tag",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
      "tag-set": [
        {

```

```

        "key": "dw-schema-name",
        "value": "${schema-name}"
    },
    {
        "key": "dw-schema-table",
        "value": "my_prefix_${table-name}"
    }
]
},
{
    "rule-type": "post-processing",
    "rule-id": "41",
    "rule-name": "41",
    "rule-action": "add-tag",
    "object-locator": {
        "schema-name": "aat",
        "table-name": "ITEM",
    },
    "tag-set": [
        {
            "key": "tag_1",
            "value": "value_1"
        },
        {
            "key": "tag_2",
            "value": "value_2"
        }
    ]
}
]
}

```

첫 번째 사후 처리 규칙은 2개의 태그(dw-schema-name과 dw-schema-table) 및 해당 동적 값 (\${schema-name}과 my_prefix_\${table-name})을 대상에 생성된 거의 모든 S3 객체에 추가합니다. 예외는 두 번째 사후 처리 규칙으로 식별되고 태그 지정된 S3 객체입니다. 따라서 와일드카드 객체 로케이터에 의해 식별된 각각의 S3 객체는 소스에서 상응하는 스키마와 테이블을 식별하는 태그를 사용하여 생성됩니다.

두 번째 사후 처리 규칙은 tag_1 및 tag_2와 해당 정적 값 value_1 및 value_2를 정확한 일치 객체 로케이터에 의해 식별되는 생성된 S3 객체에 추가합니다. 생성된 이 S3 객체는 따라서 스키마 이름이 aat인 ITEM이라는 이름의 소스에 있는 단일 테이블에 상응합니다. 정확한 일치로 인해 이러한 태그는 와일드카드만으로 S3 객체를 일치시키는 첫 번째 사후 처리 규칙에서 추가된 이 객체의 모든 태그를 대체합니다.

Example S3 객체에 동적 태그 이름과 값을 모두 추가

다음 예에는 2개의 선택 규칙과 1개의 사후 처리 규칙이 있습니다. 여기서 소스의 입력에는 retail 또는 wholesale 스키마의 ITEM 테이블만 포함됩니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "retail",
        "table-name": "ITEM"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "wholesale",
        "table-name": "ITEM"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "post-processing",
      "rule-id": "21",
      "rule-name": "21",
      "rule-action": "add-tag",
      "object-locator": {
        "schema-name": "%",
        "table-name": "ITEM",
      },
      "tag-set": [
        {
          "key": "dw-schema-name",
          "value": "${schema-name}"
        },
        {
          "key": "dw-schema-table",
          "value": "my_prefix_ITEM"
        }
      ]
    }
  ]
}
```

```

    },
    {
      "key": "${schema-name}_ITEM_tag_1",
      "value": "value_1"
    },
    {
      "key": "${schema-name}_ITEM_tag_2",
      "value": "value_2"
    }
  ]
}

```

사후 처리 규칙의 태그 세트는 2개의 태그(dw-schema-name 및 dw-schema-table)를 대상의 ITEM 테이블을 위해 생성된 모든 S3 객체에 추가합니다. 첫 번째 태그는 동적 값 "\${schema-name}"를 갖고 두 번째 태그는 정적 값 "my_prefix_ITEM"을 갖습니다. 따라서 각각의 대상 S3 객체는 소스에서 상응하는 스키마와 테이블을 식별하는 태그를 사용하여 생성됩니다.

또한 태그 세트는 동적 이름이 있는 2개의 태그("\${schema-name}_ITEM_tag_1" 및 "\${schema-name}_ITEM_tag_2")를 추가합니다. 이러한 태그는 상응하는 정적 값 value_1과 value_2를 갖습니다. 따라서 이러한 태그는 각각 현재의 스키마 retail 또는 wholesale에 대해 이름이 지정됩니다. 각 객체가 단일한 고유 스키마 이름을 위해 생성되기 때문에 이 객체에는 중복된 동적 태그 이름을 생성할 수 없습니다. 이 스키마 이름은 다른 방식으로 고유한 태그 이름을 생성하는 데 사용됩니다.

Amazon S3 대상 객체를 암호화하는 AWS KMS 키 생성

사용자 지정 AWS KMS 키를 생성하고 사용하여 Amazon S3 대상 객체를 암호화할 수 있습니다. KMS 키를 생성한 후 S3 대상 엔드포인트를 만들 때 다음 방법 중 하나로 이 키를 사용하여 객체를 암호화할 수 있습니다.

- AWS CLI를 사용하여 create-endpoint 명령을 실행할 때 (기본 .csv 파일 스토리지 형식의) S3 대상 객체에 다음 옵션을 사용하세요.

```

--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN",
"CsvRowDelimiter": "\n", "CsvDelimiter": ",", "BucketFolder": "your-bucket-folder",
"BucketName": "your-bucket-name", "EncryptionMode": "SSE_KMS",
"ServerSideEncryptionKmsKeyId": "your-KMS-key-ARN"}'

```

여기서 *your-your-KMS-key-ARN*은 KMS 키의 Amazon 리소스 이름(ARN)이며 IAM 역할에 액세스 권한이 있어야 합니다. [Amazon S3 대상에서 데이터 암호화, parquet 파일 및 CDC 사용](#) 섹션을 참조하세요.

- 추가 연결 속성 encryptionMode를 값 SSE_KMS로 설정하고, 추가 연결 속성 serverSideEncryptionKmsKeyId는 KMS 키의 ARN으로 설정합니다. 자세한 내용은 [Amazon S3를 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

KMS 키를 사용하여 Amazon S3 대상 객체를 암호화하려면 S3 버킷에 액세스할 권한이 있는 IAM 역할이 필요합니다. 그러면 생성한 암호화 키에 연결된 정책(키 정책)에서 이 IAM 역할에 액세스합니다. 이 작업은 IAM 콘솔에서 다음을 생성하여 수행할 수 있습니다.

- Amazon S3 버킷에 액세스할 권한이 있는 정책.
- 이 정책이 있는 IAM 역할.
- 이 역할을 참조하는 키 정책이 있는 KMS 키 암호화 키.

다음 절차에서 그 방법을 설명합니다.

Amazon S3 버킷에 액세스할 권한이 있는 IAM 정책을 생성하려면

1. IAM 콘솔(<https://console.aws.amazon.com/iam/>)을 엽니다.
2. 탐색 창에서 정책을 선택합니다. 정책 페이지가 열립니다.
3. 정책 생성을 선택합니다. 정책 생성 페이지가 열립니다.
4. 서비스를 선택하고 S3를 선택합니다. 작업 권한 목록이 나타납니다.
5. 모두 확장을 선택하여 목록을 확장하고 최소한 다음 권한을 선택합니다.
 - ListBucket
 - PutObject
 - DeleteObject

필요한 그 밖의 권한을 선택한 다음 모두 축소를 선택하여 목록을 축소합니다.

6. 리소스를 선택하여 액세스하려는 리소스를 지정합니다. 최소한 모든 리소스를 선택하여 일반적인 S3 리소스 액세스 권한을 제공합니다.
7. 필요한 그 밖의 조건 또는 권한을 추가한 다음 정책 검토를 선택합니다. 정책 검토 페이지에서 결과를 확인합니다.
8. 필요한 설정이 되었다면 정책의 이름(예: DMS-S3-endpoint-access)과 추가 설명을 입력한 다음 정책 생성을 선택합니다. 정책이 생성되었다는 메시지와 함께 정책 페이지가 열립니다.

9. 정책 목록에서 정책 이름을 검색하고 선택합니다. 다음과 비슷한 정책의 JSON이 표시되는 요약 페이지가 나타납니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:ListBucket",
        "s3:DeleteObject"
      ],
      "Resource": "*"
    }
  ]
}
```

이제 암호화를 위해 Amazon S3 리소스에 액세스할 수 있는 지정된 이름의 새 정책이 생성되었습니다 (예: DMS-S3-endpoint-access).

이 정책을 사용하여 IAM 역할을 생성하려면

1. IAM 콘솔의 탐색 창에서 역할을 선택합니다. 역할 세부 정보 페이지가 열립니다.
2. 역할 생성을 선택합니다. 역할 생성 페이지가 열립니다.
3. AWS 서비스를 신뢰할 수 있는 엔터티로 선택한 상태에서 IAM 역할을 사용할 서비스로 DMS를 선택합니다.
4. 다음: 권한을 선택합니다. 권한 정책 연결 뷰는 역할 생성 페이지에 표시됩니다.
5. 이전 절차(DMS-S3-endpoint-access)에서 생성한 IAM 역할에 대한 IAM 정책을 찾아 선택합니다.
6. 다음: 태그를 선택합니다. 태그 추가 뷰는 역할 생성 페이지에 나타납니다. 여기서 원하는 태그를 추가할 수 있습니다.
7. 다음: 검토를 선택합니다. 검토 뷰는 역할 생성 페이지에 나타납니다. 여기서 결과를 확인할 수 있습니다.

- 필요한 설정이 되었다면 역할의 이름(필수, 예를 들어 DMS-S3-endpoint-access-role)과 추가 설명을 입력한 다음, 역할 생성을 선택합니다. 역할이 생성되었다는 메시지와 함께 역할 세부 정보 페이지가 열립니다.

이제 암호화를 위해 Amazon S3 리소스에 액세스할 수 있는 지정된 이름의 새 역할이 생성되었습니다(예: DMS-S3-endpoint-access-role).

IAM 역할을 참조하는 키 정책이 있는 KMS 키 암호화 키를 생성하려면

 Note

에서 AWS KMS 암호화 키를 AWS DMS 사용하는 방법에 대한 자세한 내용은 섹션을 참조하세요. [암호화 키 설정 및 AWS KMS 권한 지정](#).

- 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/kms> AWS Key Management Service (AWS KMS) 콘솔을 엽니다.
- 를 변경하려면 페이지 오른쪽 상단에 있는 리전 선택기를 AWS 리전사용합니다.
- 탐색 창에서 고객 관리형 키를 선택합니다.
- 키 생성을 선택합니다. 키 구성 페이지가 열립니다.
- 키 유형에 대해 대칭을 선택합니다.

 Note

이 키를 생성할 때 Amazon S3와 같은 모든 AWS 서비스는 대칭 암호화 키로만 작동하므로 대칭 키만 생성할 수 있습니다.

- 고급 옵션을 선택합니다. 키 구성 요소 오리진에 KMS가 선택되어 있는지 확인하고 다음을 선택합니다. 레이블 추가 페이지가 열립니다.
- 별칭 및 설명 생성에 키의 별칭(예: DMS-S3-endpoint-encryption-key)과 추가 설명을 입력합니다.
- 태그에서 키 식별과 키 사용 추적에 도움이 되는 태그를 추가한 후 다음을 선택합니다. 키 관리 권한 정의 페이지가 열리고 선택할 수 있는 사용자 및 역할 목록이 표시됩니다.
- 키를 관리할 사용자와 역할을 추가합니다. 키를 관리하는 데 필요한 권한이 이러한 사용자와 역할에 있는지 확인합니다.

10. 키 삭제에서 키 관리자가 키를 삭제할 수 있는지 여부를 선택한 후 다음을 선택합니다. 키 사용 권한 정의 페이지가 열리고 선택할 수 있는 추가 사용자 및 역할 목록이 표시됩니다.
11. 이 계정의 Amazon S3 대상에 대해 암호화 작업을 수행할 사용자를 선택합니다. 또한 이전에 역할에서 생성한 역할을 선택하여 Amazon S3 대상 객체 암호화를 위한 액세스를 활성화합니다(예: DMS-S3-endpoint-access-role).
12. 이 동일한 액세스 권한을 갖도록 목록에 없는 다른 계정을 추가하려면 다른 AWS 계정에서 다른 AWS 계정 추가를 선택한 후 다음을 선택합니다. 키 정책 검토 및 편집 페이지가 열리고 기존 JSON을 입력하여 검토하고 편집할 수 있는 키 정책 JSON이 표시됩니다. 여기서 키 정책이 이전 단계에서 선택한 역할 및 사용자(예: Admin 및 User1)를 참조하는 위치를 확인할 수 있습니다. 다음 예제에 나온 것처럼 다양한 위탁자(사용자 및 역할)에 허용되는 다양한 키 작업도 볼 수 있습니다.

JSON

```
{
  "Id": "key-consolepolicy-3",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
        ]
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow access for Key Administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:role/Admin"
        ]
      },
      "Action": [
        "kms:Create*",
        "kms:Describe*",

```

```

        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",
        "kms:Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
    ],
    "Resource": "*"
},
{
    "Sid": "Allow use of the key",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::111122223333:role/DMS-S3-endpoint-access-
role",
            "arn:aws:iam::111122223333:role/Admin",
            "arn:aws:iam::111122223333:role/User1"
        ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
},
{
    "Sid": "Allow attachment of persistent resources",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::111122223333:role/DMS-S3-endpoint-access-
role",
            "arn:aws:iam::111122223333:role/Admin",
            "arn:aws:iam::111122223333:role/User1"

```

```

    ],
    },
    "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
    ],
    "Resource": "*",
    "Condition": {
        "Bool": {
            "kms:GrantIsForAWSResource": true
        }
    }
}
]
}

```

13. 마침을 클릭합니다. KMS 키가 생성되었다는 메시지와 함께 암호화 키 페이지가 열립니다.

이제 지정된 별칭의 새 KMS 키가 생성되었습니다(예: DMS-S3-endpoint-encryption-key). 이 키를 사용하면 Amazon S3 대상 객체를 암호화 AWS DMS 할 수 있습니다.

날짜 기반 폴더 파티셔닝 사용

AWS DMS 는 Amazon S3를 대상 엔드포인트로 사용할 때 트랜잭션 커밋 날짜를 기준으로 S3 폴더 파티셔닝을 지원합니다. Amazon S3 날짜 기반 폴더 파티셔닝을 사용하면 단일 소스 테이블의 데이터를 S3 버킷의 시간 계층 폴더 구조에 쓸 수 있습니다. S3 대상 엔드포인트를 생성할 때 폴더를 파티셔닝하면 다음을 수행할 수 있습니다.

- S3 객체를 더 잘 관리할 수 있습니다.
- 각 S3 폴더의 크기를 제한하세요.
- 데이터 레이크 쿼리 또는 기타 후속 작업을 최적화합니다.

S3 대상 엔드포인트를 생성할 때 날짜 기반 폴더 파티셔닝을 활성화할 수 있습니다. 기존 데이터를 마이그레이션하고 진행 중인 변경 사항을 복제(전체 로드+CDC)하거나 데이터 변경 사항만 복제(CDC만 해당)할 때 활성화할 수 있습니다. 기존 데이터를 마이그레이션하고 진행 중인 변경 사항을 복제하면 진행 중인 변경 사항만 분할됩니다. 다음과 같은 대상 엔드포인트 설정을 사용합니다.

- DatePartitionEnabled – 날짜를 기준으로 파티셔닝을 지정합니다. 이 부울 옵션을 true로 설정하여 트랜잭션 커밋 날짜를 기준으로 S3 버킷 폴더를 파티셔닝합니다.

이 설정은 PreserveTransactions 또는 CdcPath와 함께 사용할 수 없습니다.

기본값은 false입니다.

- **DatePartitionSequence** – 폴더 파티셔닝 중 사용할 날짜 형식의 순서를 식별합니다. 이 ENUM 옵션을 YYYYMMDD, YYYYMMDDHH, YYYYMM, MMYYYYDD 또는 DDDMMYYYY로 설정합니다. 기본값은 YYYYMMDD입니다. DatePartitionEnabled가 true로 설정된 경우 이 설정을 사용합니다.
- **DatePartitionDelimiter** – 폴더 파티셔닝 중 사용할 날짜 구분 기호를 지정합니다. 이 ENUM 옵션을 SLASH, DASH, UNDERSCORE 또는 NONE로 설정합니다. 기본값은 SLASH입니다. DatePartitionEnabled가 true로 설정된 경우 이 설정을 사용합니다.
- **DatePartitionTimezone** – S3 대상 엔드포인트를 생성할 때 DatePartitionTimezone을 설정하여 현재 UTC 시간을 지정된 시간대로 변환합니다. 날짜 파티션 폴더가 생성되고 CDC 파일 이름이 생성될 때 변환이 수행됩니다. 시간대 형식은 지역/위치입니다. 다음 예제와 같이 DatePartitionEnabled가 true로 설정된 경우 이 파라미터를 사용합니다.

```
s3-settings='{ "DatePartitionEnabled": true, "DatePartitionSequence": "YYYYMMDDHH",
  "DatePartitionDelimiter": "SLASH", "DatePartitionTimezone": "Asia/Seoul",
  "BucketName": "dms-nattarat-test" }'
```

다음 예제는 데이터 파티션 시퀀스의 기본값과 구분 기호를 사용하여 날짜 기반 폴더 파티셔닝을 활성화하는 방법을 보여줍니다. 이 명령은 AWS CLI create-endpoint 명령의 --s3-settings '*json-settings*' 옵션을 사용합니다.

```
--s3-settings '{ "DatePartitionEnabled": true, "DatePartitionSequence":
  "YYYYMMDD", "DatePartitionDelimiter": "SLASH" }'
```

Amazon S3를 대상으로 사용할 때 분할된 소스의 병렬 로드 AWS DMS

분할된 데이터 소스를 Amazon S3 대상에 병렬로 전체 로드하도록 구성할 수 있습니다. 이 접근 방식은 지원되는 소스 데이터베이스 엔진에서 S3 대상으로 분할된 데이터를 마이그레이션할 때 로드 시간을 개선합니다. 분할된 소스 데이터의 로드 시간을 개선하려면 소스 데이터베이스에 있는 모든 테이블의 파티션에 매핑된 S3 대상 하위 폴더를 생성합니다. 이러한 파티션 바운드 하위 폴더를 사용하면 AWS DMS가 병렬 프로세스를 실행하여 대상의 각 하위 폴더를 채울 수 있습니다.

S3 대상의 병렬 전체 로드를 구성하기 위해 S3는 테이블 매핑의 table-settings 규칙에 대해 세 가지 parallel-load 규칙 유형을 지원합니다.

- partitions-auto
- partitions-list
- ranges

이러한 병렬 로드 규칙 유형에 관한 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 섹션을 참조하세요.

partitions-auto 및 partitions-list 규칙 유형의 경우, AWS DMS 는 다음과 같이 소스 엔드 포인트의 각 파티션 이름을 사용하여 대상 하위 폴더 구조를 식별합니다.

```
bucket_name/bucket_folder/database_schema_name/table_name/partition_name/
LOADseq_num.csv
```

여기서 데이터가 S3 대상으로 마이그레이션되어 저장되는 하위 폴더 경로에는 이름이 같은 소스 파티션에 해당하는 추가 *partition_name* 하위 폴더가 포함됩니다. 그러면 이 *partition_name* 하위 폴더는 지정된 소스 파티션에서 마이그레이션된 데이터를 포함하는 하나 이상의 *LOADseq_num.csv* 파일을 저장합니다. 여기서 *seq_num*은 .csv 파일 이름의 시퀀스 번호 접미사입니다(예: .csv 파일 *LOAD00000001.csv*의 이름에 포함된 *00000001*).

하지만 MongoDB, DocumentDB 같은 일부 데이터베이스 엔진에는 파티션이라는 개념이 없습니다. 이러한 데이터베이스 엔진의 경우 다음과 같이 실행 중인 소스 세그먼트 인덱스를 대상 .csv 파일 이름에 접두사로 AWS DMS 추가합니다.

```
.../database_schema_name/table_name/SEGMENT1_LOAD00000001.csv
.../database_schema_name/table_name/SEGMENT1_LOAD00000002.csv
...
.../database_schema_name/table_name/SEGMENT2_LOAD00000009.csv
.../database_schema_name/table_name/SEGMENT3_LOAD0000000A.csv
```

여기서 파일 *SEGMENT1_LOAD00000001.csv* 및 *SEGMENT1_LOAD00000002.csv*는 실행 중인 소스 세그먼트 인덱스 접두사 *SEGMENT1*과 동일한 접두사를 포함하여 이름이 지정됩니다. 이 두 개의 .csv 파일에 대해 마이그레이션된 소스 데이터는 실행 중인 동일한 소스 세그먼트 인덱스와 연결되어 있기 때문에 그와 같은 이름이 지정됩니다. 반면, 대상 *SEGMENT2_LOAD00000009.csv* 및 *SEGMENT3_LOAD0000000A.csv* 파일에 각각 저장된 마이그레이션된 데이터는 서로 다른 실행 소스 세그먼트 인덱스와 연결됩니다. 각 파일의 파일 이름 앞에는 실행 중인 세그먼트 인덱스의 이름 *SEGMENT2* 및 *SEGMENT3*이 접두사로 붙습니다.

ranges 병렬 로드 유형의 경우, table-settings 규칙의 columns 및 boundaries 설정을 사용하여 열 이름과 열 값을 정의합니다. 이러한 규칙을 사용하여 다음과 같이 세그먼트 이름에 해당하는 파티션을 지정할 수 있습니다.

```
"parallel-load": {
  "type": "ranges",
  "columns": [
    "region",
    "sale"
  ],
  "boundaries": [
    [
      "NORTH",
      "1000"
    ],
    [
      "WEST",
      "3000"
    ]
  ],
  "segment-names": [
    "custom_segment1",
    "custom_segment2",
    "custom_segment3"
  ]
}
```

여기서 segment-names 설정은 S3 대상에서 데이터를 병렬로 마이그레이션하기 위한 세 개의 파티션 이름을 정의합니다. 마이그레이션된 데이터는 다음과 같이 병렬로 로드되어 파티션 하위 폴더 아래의 .csv 파일에 순서대로 저장됩니다.

```
.../database_schema_name/table_name/custom_segment1/LOAD[00000001...].csv
.../database_schema_name/table_name/custom_segment2/LOAD[00000001...].csv
.../database_schema_name/table_name/custom_segment3/LOAD[00000001...].csv
```

여기서는 세 개의 파티션 하위 폴더 각각에 일련의 .csv 파일을 AWS DMS 저장합니다. 각 파티션 하위 폴더에 있는 일련의 .csv 파일은 모든 데이터가 마이그레이션될 때까지 LOAD00000001.csv를 시작으로 단계적으로 이름이 지정됩니다.

경우에 따라서는 segment-names 설정을 사용하여 ranges 병렬 로드 유형에 대한 파티션 하위 폴더의 이름을 명시적으로 지정하지 않을 수 있습니다. 이 경우는 *table_name* 하위 폴더 아래에 각 .csv

파일 시리즈를 생성하는 기본값을 AWS DMS 적용합니다. 여기서 AWS DMS 는 다음과 같이 일련의 .csv 파일의 s파일 이름 앞에 실행 중인 소스 세그먼트 인덱스의 이름을 접두사로 붙입니다.

```
.../database_schema_name/table_name/SEGMENT1_LOAD[00000001...].csv
.../database_schema_name/table_name/SEGMENT2_LOAD[00000001...].csv
.../database_schema_name/table_name/SEGMENT3_LOAD[00000001...].csv
...
.../database_schema_name/table_name/SEGMENTZ_LOAD[00000001...].csv
```

Amazon S3를 대상으로 사용 시 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 마찬가지로 엔드포인트 설정을 사용하여 Amazon S3 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --s3-settings '{"EndpointSetting": "value", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Note

DMS는 소스 데이터베이스의 커밋 순서에 따라 Parquet 파일에 변경 사항을 작성하지만 여러 테이블을 마이그레이션할 때 테이블 수준 파티셔닝으로 인해 원래 트랜잭션 순서가 유지되지 않습니다. 트랜잭션 시퀀스 정보를 유지하려면 각 행에 대한 소스 커밋 타임스탬프를 포함하도록 TimestampColumnName 엔드포인트 설정을 구성합니다. 그러면 다운스트림 처리에서 이를 사용하여 원래 트랜잭션 시퀀스를 재구성할 수 있습니다. PreserveTransactions 설정을 제공하는 CSV 형식과 달리 Parquet 파일은 열 기반 스토리지 구조로 인해 트랜잭션을 다르게 처리하지만, 이 접근 방식을 사용하면 소스 커밋 시간을 정확하게 추적할 수 있고, 마이그레이션 후 트랜잭션 순서 재구성을 지원하고, 데이터 일관성을 유지하면서 효율적인 데이터 처리를 수행할 수 있습니다.

Amazon S3를 대상으로 하여 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

옵션	설명
CsvNullValue	가 null 값을 AWS DMS 처리하는 방법을 지정하는 선택적 파라미터입니다. null 값을 처리하는 동안 이 파라미터를 사용하여 대상에 쓸 때 사용자 정의 문자열을 null로 전달할 수 있습니다. 예를 들어, 대상 열이 null을 허용하는 경우, 이 옵션을 사용하여 빈 문자열 값과 null 값을 구분할 수 있습니다.

옵션	설명
	기본값: "" 유효값: 모든 유효한 문자열 예시: <code>--s3-settings '{"CsvNullValue": "NULL"}'</code> 소스 데이터베이스 열 값이 null인 경우 S3 CSV 파일에서 열 값은 "" 문자열 NULL 대신입니다.
AddColumnName	true 또는 y로 설정할 때 .csv 출력 파일에 열 이름 정보를 추가하는 데 사용할 수 있는 선택적 파라미터입니다. 이 파라미터는 PreserveTransactions 또는 CdcPath와 함께 사용할 수 없습니다. 기본값: false 유효값: true, false, y, n 예시: <code>--s3-settings '{"AddColumnName": true}'</code>
AddTrailingPaddingCharacter	S3 대상 엔드포인트 설정 AddTrailingPaddingCharacter 를 사용하여 문자열 데이터에 패딩을 추가합니다. 기본값은 false입니다. 유형: Boolean 예시: <code>--s3-settings '{"AddTrailingPaddingCharacter": true}'</code>
BucketFolder	S3 버킷의 폴더 이름을 설정하는 선택적 파라미터입니다. 제공되는 경우, 대상 객체는 경로 <i>BucketFolder</i> /<i>schema_name</i> /<i>table_name</i> /에서 .csv 또는 .parquet 파일로 생성됩니다. 이 파라미터가 지정되지 않은 경우, 사용되는 경로는 <i>schema_name</i> /<i>table_name</i> /입니다. 예시: <code>--s3-settings '{"BucketFolder": "testFolder"}'</code>

옵션	설명
BucketName	S3 대상 객체가 .csv 또는 .parquet 파일로 생성되는 S3 버킷의 이름입니다. 예시: <code>--s3-settings '{"BucketName": "buckettest"}'</code>
CannedAclForObjects	가 S3 버킷에서 생성된 객체에 대해 사전 정의된(캔된) 액세스 제어 목록을 .csv 또는 .parquet 파일로 AWS DMS 지정할 수 있도록 하는 값입니다. Amazon S3에서 미리 제공된 ACL에 관한 자세한 내용은 Amazon S3 개발자 안내서의 미리 제공된 ACL을 참조하세요. 기본값: 없음 이 속성의 유효값은 NONE, PRIVATE, PUBLIC_READ, PUBLIC_READ_WRITE, AUTHENTICATED_READ, AWS_EXEC_READ, BUCKET_OWNER_READ, BUCKET_OWNER_FULL_CONTROL입니다. 예시: <code>--s3-settings '{"CannedAclForObjects": "PUBLIC_READ"}'</code>

옵션	설명
CdcInsertsOnly	변경 데이터 캡처(CDC) 로드 중 심표로 구분된 값(.csv) 또는 컬럼 방식 스토리지(.parquet) 출력 파일에 INSERT 작업만을 작성하는 선택적 파라미터입니다. 기본적으로(false 설정), .csv 또는 .parquet 레코드의 첫 번째 필드에는 문자 I(INSERT), U(UPDATE) 또는 D(DELETE)가 포함됩니다. 이 문자는 대상에 대한 CDC 로드를 위해 소스 데이터베이스에 행이 삽입, 업데이트 또는 삭제되었는지 여부를 나타냅니다. cdcInsertsOnly 가 true 또는 y로 설정되면 소스 데이터베이스의 INSERT만 .csv 또는 .parquet 파일로 마이그레이션됩니다. .csv 형식의 경우에만 IncludeOpForFullLoad 의 값에 따라 이 INSERT가 기록되는 방식이 달라집니다. IncludeOpForFullLoad 가 true로 설정되면 모든 CDC 레코드의 첫 번째 필드는 소스에서 INSERT 작업을 나타내기 위해 I로 설정됩니다. IncludeOpForFullLoad 가 false로 설정되면 모든 CDC 레코드는 소스에서 INSERT 작업을 나타내는 첫 번째 필드 없이 작성됩니다. 이러한 파라미터가 상호 작용하는 방식에 관한 자세한 내용은 마이그레이션된 S3 데이터에 소스 DB 작업 표시 섹션을 참조하세요. 기본값: false 유효값: true, false, y, n 예시: --s3-settings '{"CdcInsertsOnly": true}'

옵션	설명
CdcInsertsAndUpdates	변경 데이터 캡처(CDC) 로드를 사용하여 INSERT 및 UPDATE 작업을 .csv 또는 .parquet(열 기반 스토리지) 출력 파일에 기록할 수 있습니다. 기본 설정은 false지만 cdcInsertsAndUpdates 를 true 또는 y로 설정하면 소스 데이터베이스의 INSERT 및 UPDATE가 .csv 또는 .parquet 파일로 마이그레이션됩니다. .csv 파일 형식의 경우에만 이러한 INSERT 및 UPDATE가 기록되는 방법이 includeOpForFullLoad 파라미터 값에 따라 달라집니다. includeOpForFullLoad 를 true로 설정하면 모든 CDC 레코드의 첫 번째 필드가 소스에서 INSERT 및 UPDATE 작업을 나타내도록 I 또는 U로 설정됩니다. 그러나 includeOpForFullLoad 를 false로 설정하면 소스에서 INSERT 또는 UPDATE 작업 표시 없이 CDC 레코드가 기록됩니다. 이러한 파라미터가 상호 작용하는 방식에 관한 자세한 내용은 마이그레이션된 S3 데이터에 소스 DB 작업 표시 섹션을 참조하세요.  Note CdcInsertsOnly 및 cdcInsertsAndUpdates 는 동일한 엔드 포인트에서 둘 다 true로 설정할 수 없습니다. 동일한 엔드포인트에 대해 cdcInsertsOnly 또는 cdcInsertsAndUpdates 를 true로 설정합니다. 단, 둘 다 설정하지는 않습니다. 기본값: false 유효값: true, false, y, n 예시: --s3-settings '{"CdcInsertsAndUpdates": true}'

옵션	설명
CdcPath	CDC 파일의 폴더 경로를 지정합니다. S3 소스의 경우, 태스크가 변경 데이터를 캡처하는 경우 이 설정이 필요합니다. 그렇지 않으면 선택 사항입니다. CdcPath가 설정된 경우 DMC는 이 경로에서 CDC 파일을 읽고 데이터 변경 사항을 대상 엔드포인트로 복제합니다. S3 대상의 경우 PreserveTransactions 를 true로 설정하면 DMS는 이 파라미터를 S3 대상의 폴더 경로로 설정했는지 확인합니다. 폴더 경로에서는 DMS가 CDC 로드와 대한 트랜잭션 순서를 저장할 수 있습니다. DMS는 이 CDC 폴더 경로를 S3 대상 작업 디렉터리 또는 BucketFolder 및 BucketName 으로 지정된 S3 대상 위치에 생성합니다. 이 파라미터는 DatePartitionEnabled 또는 AddColumnName 과 함께 사용할 수 없습니다. 유형: 문자열 예를 들어, CdcPath를 MyChangedData 로 지정하고 BucketName 을 MyTargetBucket 으로 지정하지만 BucketFolder 를 지정하지 않으면 DMS는 CDC 폴더 경로(MyTargetBucket/MyChangedData)를 생성합니다. 동일한 CdcPath를 지정하고 BucketName 을 MyTargetBucket 으로, BucketFolder 를 MyTargetData 로 지정하면 DMS는 CDC 폴더 경로(MyTargetBucket/MyTargetData/MyChangedData)를 생성합니다.  Note 이 설정은 AWS DMS 버전 3.4.2 이상에서 지원됩니다. 트랜잭션 순서대로 데이터 변경을 캡처할 때 DMS는 대상의 DataFormat S3 설정 값에 관계없이 항상 행 변경 내용을 .csv 파일에 저장합니다.

옵션	설명
CdcMaxBatchInterval	파일을 Amazon S3에 출력할 때까지의 최대 간격(초)입니다. 기본값: 60초 CdcMaxBatchInterval 이 지정되고 CdcMinFileSize 가 지정되면 파일 쓰기는 가장 먼저 충족되는 파라미터 조건에 의해 트리거됩니다.  Note AWS DMS 버전 3.5.3부터 PostgreSQL 또는 Aurora PostgreSQL을 소스로 사용하고 Amazon S3를 Parquet을 대상으로 사용하는 경우 <code>confirmed_flush_lsn</code> 업데이트 빈도는 대상 엔드포인트가 메모리에 보관하도록 구성된 데이터의 양에 따라 달라집니다. 메모리의 데이터가 Amazon S3에 기록된 후에만 소스로 <code>confirmed_flush_lsn</code> 다시 AWS DMS 보냅니다. CdcMaxBatchInterval 파라미터를 더 높은 값으로 구성하면 소스 데이터베이스에서 복제 슬롯 사용량이 증가하는 것을 관찰할 수 있습니다.
CdcMinFileSize	Amazon S3에 파일을 출력하기 위한 최소 파일 크기 조건(단위: 킬로바이트[KB]) 기본값: 32000 KB CdcMinFileSize 가 지정되고 CdcMaxBatchInterval 이 지정되면 파일 쓰기는 가장 먼저 충족되는 파라미터 조건에 의해 트리거됩니다.

옵션	설명
PreserveTransactions	true로 설정된 경우 DMS는 CdcPath로 지정된 Amazon S3 대상의 변경 데이터 캡처(CDC)에 대한 트랜잭션 순서를 저장합니다. 이 파라미터는 DatePartitionEnabled 또는 AddColumnName 과 함께 사용할 수 없습니다. 유형: Boolean 트랜잭션 순서대로 데이터 변경을 캡처할 때 DMS는 대상의 DataFormat S3 설정 값에 관계없이 항상 행 변경 내용을 .csv 파일에 저장합니다.  Note 이 설정은 AWS DMS 버전 3.4.2 이상에서 지원됩니다.

옵션	설명
IncludeOpForFullLoad	전체 로드 중 쉼표로 구분된 값(.csv) 출력 파일에만 INSERT 작업을 쓰기 위한 선택적 파라미터입니다. 전체 로드의 경우, 레코드는 삽입만 가능합니다. 기본적으로(false 설정) 전체 로드에는 이 출력 파일에는 행이 소스 데이터베이스에 삽입되었음을 나타내는 레코드가 기록되어 있지 않습니다. IncludeOpForFullLoad 가 true 또는 y로 설정되면 .csv 파일의 첫 번째 필드에 INSERT가 1 주석으로 기록됩니다.  Note 이 파라미터는 CdcInsertsOnly 또는 CdcInsertsAndUpdates 와 함께 .csv 파일의 출력에만 적용됩니다. 이러한 파라미터가 상호 작용하는 방식에 관한 자세한 내용은 마이그레이션된 S3 데이터에 소스 DB 작업 표시 섹션을 참조하세요. 기본값: false 유효값: true, false, y, n 예시: --s3-settings '{"IncludeOpForFullLoad": true}'
CompressionType	GZIP으로 설정하면 GZIP을 사용하여 대상 .csv 파일을 압축하는 선택적 파라미터입니다. 이 파라미터를 기본값으로 설정하면 파일을 압축되지 않은 상태로 둡니다. 기본값: NONE 유효값: GZIP 또는 NONE 예시: --s3-settings '{"CompressionType": "GZIP}"
CsvDelimiter	.csv 소스 파일에서 열을 구분하는 데 사용되는 구분 기호입니다. 기본값은 쉼표(,)입니다. 예시: --s3-settings '{"CsvDelimiter": ","}'

옵션	설명
CsvRowDelimiter	.csv 소스 파일에서 행을 구분하는 데 사용되는 구분 기호입니다. 기본값은 줄 바꿈(\n)입니다. 예시: <code>--s3-settings '{"CsvRowDelimiter": "\n"}'</code>
MaxFileSize	전체 로드 중에 S3 대상으로 마이그레이션하는 동안 생성할 .csv 파일의 최대 크기(KB 단위)를 지정하는 값입니다. 기본값: 1,048,576KB(1GB) 유효값: 1~1,048,576 예시: <code>--s3-settings '{"MaxFileSize": 512}'</code>
Rfc4180	.csv 파일 형식만 사용하여 Amazon S3로 마이그레이션되는 데이터의 RFC 규정 준수 동작을 설정하는 데 사용되는 선택적 파라미터입니다. 이 값을 true 대상으로 또는 Amazon S3를 y 사용하여 로 설정하면 데이터에 따옴표, 쉼표 또는 줄 바꿈 문자가 있는 경우는 전체 열을 추가 큰따옴표 쌍(")으로 AWS DMS 묶습니다. 데이터 내 모든 따옴표는 두 번 반복됩니다. 이 형식은 RFC 4180을 준수합니다. 기본값: true 유효값: true, false, y, n 예시: <code>--s3-settings '{"Rfc4180": false}'</code>

옵션	설명
EncryptionMode	S3로 복사되는 .csv 또는 .parquet 객체 파일을 암호화할 서버 측 암호화 모드입니다. 유효값은 SSE_S3(S3 서버 측 암호화) 또는 SSE_KMS(KMS 키 암호화)입니다. SSE_KMS를 선택하는 경우, ServerSideEncryptionKmsKeyId 파라미터를 암호화에 사용할 KMS 키의 Amazon 리소스 이름(ARN)으로 설정합니다.  Note 또한 CLI modify-endpoint 명령을 사용하여 기존 엔드포인트의 EncryptionMode 속성 값을 SSE_KMS에서 SSE_S3로 변경할 수 있습니다. 그러나 EncryptionMode 값을 SSE_S3에서 SSE_KMS로 변경할 수 없습니다. 기본값: SSE_S3 유효값: SSE_S3 또는 SSE_KMS 예시: <code>--s3-settings '{"EncryptionMode": SSE_S3}'</code>
ServerSideEncryptionKmsKeyId	EncryptionMode 를 SSE_KMS로 설정하는 경우, 이 파라미터를 KMS 키의 Amazon 리소스 이름(ARN)으로 설정합니다. 계정에 대해 생성된 키 목록에서 AWS KMS 키 별칭을 선택하여 이 ARN을 찾을 수 있습니다. 키를 생성할 때 특정 정책과 역할을 이 KMS 키와 연결해야 합니다. 자세한 내용은 Amazon S3 대상 객체를 암호화하는 AWS KMS 키 생성 단원을 참조하십시오. 예시: <code>--s3-settings '{"ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-east-1:111122223333:key/11a1a1a1-aaaa-9999-abab-2bbbbbb222a2"}'</code>

옵션	설명
DataFormat	가 S3 객체를 생성하는 데 AWS DMS 사용하는 파일의 출력 형식입니다. Amazon S3 대상의 경우는 .csv 또는 .parquet 파일을 AWS DMS 지원합니다. .parquet 파일은 효율적인 압축 옵션과 보다 빠른 쿼리 성능을 갖춘 이진 열 방식 스토리지 형식을 갖고 있습니다. .parquet 파일에 관한 자세한 내용은 https://parquet.apache.org/를 참조하세요. 기본값: csv 유효값: csv 또는 parquet 예시: <code>--s3-settings '{"DataFormat": "parquet"}'</code>
EncodingType	Parquet 인코딩 유형입니다. 인코딩 유형 옵션에는 다음이 포함됩니다. • rle-dictionary – 이 사전식 인코딩은 반복되는 값을 보다 효율적으로 저장하기 위해 비트 압축과 실행 길이 인코딩의 조합을 사용합니다. • plain – 인코딩 없음. • plain-dictionary – 이 사전식 인코딩은 주어진 열에서 발생하는 값의 사전을 빌드합니다. 사전은 각 열 청크의 사전 페이지에 저장됩니다. 기본값: rle-dictionary 유효값: rle-dictionary , plain 또는 plain-dictionary 예시: <code>--s3-settings '{"EncodingType": "plain-dictionary"}'</code>

옵션	설명
DictPageSizeLimit	.parquet 파일에서 허용되는 사전 페이지의 최대 크기(단위: 바이트)입니다. 사전 페이지가 이 값을 초과하면 해당 페이지는 일반 인코딩을 사용합니다. 기본값: 1,024,000(1MB) 유효값: 유효한 정수 값 예시: <code>--s3-settings '{"DictPageSizeLimit": 2,048,000}'</code>
RowGroupLength	.parquet 파일의 한 행 그룹에 있는 행 수입니다. 기본값: 10,024(10KB) 유효값: 유효한 정수 값 예시: <code>--s3-settings '{"RowGroupLength": 20,048}'</code>
DataPageSize	.parquet 파일에서 허용되는 데이터 페이지의 최대 크기(단위: 바이트)입니다. 기본값: 1,024,000(1MB) 유효값: 유효한 정수 값 예시: <code>--s3-settings '{"DataPageSize": 2,048,000}'</code>
ParquetVersion	.parquet 파일 형식의 버전입니다. 기본값: PARQUET_1_0 유효값: PARQUET_1_0 또는 PARQUET_2_0 예시: <code>--s3-settings '{"ParquetVersion": "PARQUET_2_0"}'</code>

옵션	설명
EnableStatistics	true 또는 y로 설정하여 .parquet 파일 페이지와 행 그룹에 관한 통계를 활성화합니다. 기본값: true 유효값: true, false, y, n 예시: <code>--s3-settings '{"EnableStatistics": false}'</code>
TimestampColumnName	S3 대상 엔드포인트 데이터에 타임스탬프 열을 포함시키기 위한 선택적 파라미터입니다. AWS DMS 를 비어 TimestampColumnName 값으로 설정하면는 마이그레이션된 데이터의 .csv 또는 .parquet 객체 파일에 추가 STRING 열을 포함합니다. 전체 로드와 CDC 로드의 경우, 타임스탬프 열의 각 행에는 DMS에 의해 소스에서 대상으로 데이터가 전송된 시점에 대한 타임스탬프가 포함됩니다. CDC 로드의 경우, 타임스탬프 열의 각 행에는 소스 데이터베이스의 해당 행 커밋에 대한 타임스탬프가 있습니다. 이 타임스탬프 열 값의 문자열 형식은 yyyy-MM-dd HH:mm:ss.SSSSSS 입니다. 기본적으로 이 값의 정밀도는 마이크로초입니다. CDC 로드의 경우, 정밀도의 라운딩은 소스 데이터베이스에 대해 DMS가 지원하는 커밋 타임스탬프에 따라 다릅니다. AddColumnName 파라미터가 true로 설정되면 DMS는 비어 있지 않은 TimestampColumnName 값으로 설정한 타임스탬프 열의 이름도 포함합니다. 예시: <code>--s3-settings '{"TimestampColumnName": "TIMESTAMP"}'</code>

옵션	설명
UseTaskStartTimeForFullLoadTimestamp	true로 설정하면 이 파라미터는 데이터를 대상에 쓰는 시간 대신 태스크 시작 시간을 타임스탬프 열 값으로 사용합니다. 전체 로드の場合, UseTaskStartTimeForFullLoadTimestamp 가 true로 설정되면 타임스탬프 열의 각 행에 태스크 시작 시간이 포함됩니다. CDC 로드の場合 타임스탬프 열의 각 행에는 트랜잭션 커밋 시간이 포함됩니다. UseTaskStartTimeForFullLoadTimestamp 가 false로 설정되면 타임스탬프 열의 전체 로드 타임스탬프는 데이터가 대상에 도착하는 시간과 함께 증가합니다. 기본값: false 유효값: true, false 예시: <code>--s3-settings '{"UseTaskStartTimeForFullLoadTimestamp": true}'</code> UseTaskStartTimeForFullLoadTimestamp: true 는 전체 로드에 대한 S3 대상 TimestampColumnName 을 CDC 로드 시 TimestampColumnName 과 함께 정렬할 수 있도록 도움을 줍니다.

옵션	설명
ParquetTimestampInMillisecond	.parquet 형식의 S3 객체 파일에 기록된 모든 TIMESTAMP 열 값의 정밀도를 지정하는 선택적 파라미터입니다. 이 속성을 true 또는 로 설정하면는 .parquet 형식 파일의 모든 TIMESTAMP 열을 밀리초 정밀도로 y AWS DMS 씁니다. 그렇지 않으면 DMS는 해당 열을 마이크로초 정밀도로 기록합니다. 현재 Amazon Athena 및는 TIMESTAMP 값에 대해 밀리초 정밀도만 처리할 AWS Glue 수 있습니다. Athena 또는 AWS Glue를 사용하여 데이터를 쿼리하거나 처리할 계획인 경우에만 .parquet 형식의 S3 엔드포인트 객체 파일에 대해 이 속성을 true로 설정합니다.  Note • AWS DMS 는 S3 파일에 기록된 모든 TIMESTAMP 열 값을 마이크로초 정밀도로 .csv 형식으로 씁니다. • 이 속성의 설정은 TimestampColumnName 속성을 설정하여 삽입된 타임스탬프 열 값의 문자열 형식에는 영향을 미치지 않습니다. 기본값: false 유효값: true, false, y, n 예시: <code>--s3-settings '{"ParquetTimestampInMillisecond": true}'</code>

옵션	설명
GlueCatalogGeneration	를 생성하려면이 엔드포인트 설정을 로 AWS Glue Data Catalog 설정합니다true. 기본값: false 유효값: true, false. 예시: --s3-settings '{"GlueCatalogGeneration": true}' 참고: GlueCatalogGeneration 을 PreserveTransactions 및 CdcPath와 함께 사용하지 마세요.

에 Amazon S3 대상과 AWS Glue Data Catalog 함께 사용 AWS DMS

AWS Glue 는 데이터를 분류하는 간단한 방법을 제공하는 서비스로, 라고 하는 메타데이터 리포지토리로 구성됩니다 AWS Glue Data Catalog. Amazon S3 대상 엔드포인트 AWS Glue Data Catalog 와를 통합하고 Amazon Athena와 같은 다른 AWS 서비스를 통해 Amazon S3 데이터를 쿼리할 수 있습니다. Amazon Athena Amazon Redshift는에서 작동 AWS Glue 하지만 사전 구축된 옵션으로 이를 지원하지 AWS DMS 않습니다.

데이터 카탈로그를 생성하려면 다음 AWS CLI 예제와 true같이 GlueCatalogGeneration 엔드포인트 설정을 로 설정합니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint
                        --engine-name s3 --endpoint-type target--s3-settings
                        '{"ServiceAccessRoleArn":
                          "your-service-access-ARN", "BucketFolder": "your-bucket-folder",
                          "BucketName":
                          "your-bucket-name", "DataFormat": "parquet", "GlueCatalogGeneration":
                          true}'
```

csv 형식 데이터가 포함된 전체 부하 복제 작업의 경우, IncludeOpForFullLoad를 true로 설정합니다.

참고: GlueCatalogGeneration을 PreserveTransactions 및 CdcPath와 함께 사용하지 마세요. AWS Glue 크롤러는 지정된에 저장된 파일의 서로 다른 스키마를 조정할 수 없습니다CdcPath.

Amazon Athena가 Amazon S3 데이터를 인덱싱하고 Amazon Athena를 통해 표준 SQL 쿼리를 사용하여 데이터를 쿼리하려면 엔드포인트에 연결된 IAM 역할에 다음 정책이 있어야 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads",
        "s3:ListMultipartUploadParts",
        "s3:AbortMultipartUpload"
      ],
      "Resource": [
        "arn:aws:s3:::bucket123",
        "arn:aws:s3:::bucket123/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "glue:CreateDatabase",
        "glue:GetDatabase",
        "glue:CreateTable",
        "glue:DeleteTable",
        "glue:UpdateTable",
        "glue:GetTable",
        "glue:BatchCreatePartition",
        "glue:CreatePartition",
        "glue:UpdatePartition",
        "glue:GetPartition",
        "glue:GetPartitions",
        "glue:BatchGetPartition"
      ],
      "Resource": [
        "arn:aws:glue*:111122223333:catalog",
        "arn:aws:glue*:111122223333:database/*",
        "arn:aws:glue*:111122223333:table/*"
      ]
    }
  ]
}
```

```

    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "athena:StartQueryExecution",
      "athena:GetQueryExecution",
      "athena:CreateWorkGroup"
    ],
    "Resource": "arn:aws:athena:*:111122223333:workgroup/
glue_catalog_generation_for_task_*"
  }
]
}

```

참조

- 에 대한 자세한 내용은 AWS Glue 개발자 안내서의 [개념](#)을 AWS Glue참조하세요.
- 에 대한 자세한 내용은 AWS Glue 개발자 안내서의 [구성 요소를](#) AWS Glue Data Catalog 참조하세요.

Amazon S3 대상에서 데이터 암호화, parquet 파일 및 CDC 사용

S3 대상 엔드포인트 설정을 사용하여 다음을 구성할 수 있습니다.

- S3 대상 객체를 암호화하기 위한 사용자 지정 KMS 키.
- S3 대상 객체의 스토리지 형식인 parquet 파일
- S3 대상의 트랜잭션 순서를 포함한 변경 데이터 캡처(CDC).
- Amazon S3 대상 엔드포인트 AWS Glue Data Catalog 와 통합하고 Amazon Athena와 같은 다른 서비스를 통해 Amazon Amazon S3 데이터를 쿼리합니다.

AWS KMS 데이터 암호화를 위한 키 설정

다음 예는 S3 대상 객체를 암호화하도록 사용자 지정 KMS 키를 구성하는 방법을 보여 줍니다. 시작하려면 다음 create-endpoint CLI 명령을 실행합니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint --engine-name s3 --
endpoint-type target
```

```
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN", "CsvRowDelimiter":
"\n",
"CsvDelimiter": ",", "BucketFolder": "your-bucket-folder",
"BucketName": "your-bucket-name",
"EncryptionMode": "SSE_KMS",
"ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/72abb6fb-1e49-4ac1-9aed-c803dfcc0480"}'
```

여기서 `--s3-settings` 옵션에 의해 지정된 JSON 객체는 다음 두 가지 파라미터를 정의합니다. 하나는 값이 `SSE_KMS`인 `EncryptionMode` 파라미터입니다. 다른 하나는 값이 `arn:aws:kms:us-east-1:111122223333:key/72abb6fb-1e49-4ac1-9aed-c803dfcc0480`인 `ServerSideEncryptionKmsKeyId` 파라미터입니다. 이 값은 사용자 지정 KMS 키의 Amazon 리소스 이름(ARN)입니다. S3 대상의 경우, 추가 설정도 지정할 수 있습니다. 이러한 설정은 서버 액세스 역할을 식별하고 기본 CSV 객체 스토리지 형식의 구분 기호를 제공하며 S3 대상 객체를 저장할 버킷 위치와 이름을 제공합니다.

기본적으로 S3 데이터 암호화는 S3 서버 측 암호화를 사용하여 수행됩니다. 이전 예의 S3 대상의 경우, 다음 예에서처럼 이것은 엔드포인트 설정을 지정하는 것에 해당합니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint --engine-name s3 --
endpoint-type target
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN", "CsvRowDelimiter":
"\n",
"CsvDelimiter": ",", "BucketFolder": "your-bucket-folder",
"BucketName": "your-bucket-name",
"EncryptionMode": "SSE_S3"}'
```

S3 서버 측 암호화 작업에 관한 자세한 내용은 [서버 측 암호화를 사용하여 데이터 보호](#) 섹션을 참조하세요.

Note

또한 CLI `modify-endpoint` 명령을 사용하여 기존 엔드포인트의 `EncryptionMode` 파라미터 값을 `SSE_KMS`에서 `SSE_S3`로 변경할 수 있습니다. 그러나 `EncryptionMode` 값을 `SSE_S3`에서 `SSE_KMS`로 변경할 수 없습니다.

.parquet 파일을 사용하여 S3 대상 객체를 저장하기 위한 설정

S3 대상 객체 생성의 기본 형식은 .csv 파일입니다. 다음 예는 S3 대상 객체 생성을 위한 형식으로 .parquet 파일을 지정하기 위한 몇 가지 엔드포인트 설정을 보여 줍니다. 다음 예에서처럼 .parquet 파일 형식을 모두 기본값으로 지정할 수 있습니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint --engine-name s3 --
endpoint-type target
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN", "DataFormat":
"parquet"}'
```

여기서는 모든 S3 기본값으로 형식을 활성화하기 위해 DataFormat 파라미터가 parquet로 설정됩니다. 이러한 기본값에는 반복되는 값을 보다 효율적으로 저장하기 위해 비트 압축과 실행 길이 인코딩의 조합을 사용하는 사전식 인코딩("EncodingType: "rle-dictionary")이 포함됩니다.

다음 예에서처럼 기본값이 아닌 옵션의 추가 설정을 추가할 수 있습니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint --engine-name s3 --
endpoint-type target
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN", "BucketFolder":
"your-bucket-folder",
"BucketName": "your-bucket-name", "DataFormat": "parquet", "EncodingType": "plain-
dictionary", "DictPageSizeLimit": 3,072,000,
"EnableStatistics": false }'
```

여기서 몇 가지 표준 S3 버킷 옵션의 파라미터와 DataFormat 파라미터 외에 다음의 추가 .parquet 파일 파라미터가 설정됩니다.

- EncodingType – 사전 페이지의 열당 청크의 각 열에서 발생하는 값을 저장하는 사전식 인코딩(plain-dictionary)으로 설정됩니다.
- DictPageSizeLimit – 사전 페이지 최대 크기인 3MB로 설정됩니다.
- EnableStatistics – Parquet 파일 페이지와 행 그룹에 관한 통계 수집을 활성화하는 기본값을 비활성화합니다.

S3 대상의 트랜잭션 순서를 포함한 변경 데이터 캡처(CDC).

기본적으로는 CDC 작업을 AWS DMS 실행할 때 소스 데이터베이스(또는 데이터베이스)에 기록된 모든 행 변경 사항을 각 테이블에 대한 하나 이상의 파일에 저장합니다. 동일한 테이블에 대한 변경 사항이 포함된 각 파일 세트는 해당 테이블과 연결된 단일 대상 디렉터리에 있습니다.는 Amazon S3 대

상 엔드포인트로 마이그레이션된 데이터베이스 테이블만큼 많은 대상 디렉토리를 AWS DMS 생성합니다. 파일은 트랜잭션 순서에 관계없이 S3 대상의 이러한 디렉토리에 저장됩니다. 파일 명명 규칙, 데이터 내용 및 형식에 관한 자세한 내용은 [Amazon S3를의 대상으로 사용 AWS Database Migration Service](#)을 참조하세요.

트랜잭션 순서도 캡처하는 방식으로 소스 데이터베이스 변경 사항을 캡처하려면 트랜잭션 크기에 따라 생성된 하나 이상의 .csv 파일에 모든 데이터베이스 테이블에 대한 행 변경 사항을 저장 AWS DMS 하도록 지시하는 S3 엔드포인트 설정을 지정할 수 있습니다. 이러한 .csv 트랜잭션 파일에는 각 트랜잭션과 관련된 모든 테이블에 대해 트랜잭션 순서대로 나열된 모든 행 변경 내용이 포함되어 있습니다. 이러한 트랜잭션 파일은 사용자가 S3 대상에서도 지정하는 단일 트랜잭션 디렉토리에 함께 있습니다. 각 트랜잭션 파일에서 트랜잭션 작업과 각 행 변경에 대한 데이터베이스 및 소스 테이블의 ID는 다음과 같이 행 데이터의 일부로 저장됩니다.

```
operation,table_name,database_schema_name,field_value,...
```

여기서 *operation*은 변경된 행의 트랜잭션 작업이고 *table_name*은 행이 변경된 데이터베이스 테이블의 이름이며 *database_schema_name*은 테이블이 있는 데이터베이스 스키마의 이름이고 *field_value*는 행의 데이터를 지정하는 하나 이상의 필드 값 중 첫 번째 필드 값입니다.

다음 트랜잭션 파일 예제는 두 테이블을 포함하는 하나 이상의 트랜잭션에 대해 변경된 행을 보여줍니다.

```
I,Names_03cdcad11a,rdsTempsdb,13,Daniel
U,Names_03cdcad11a,rdsTempsdb,23,Kathy
D,Names_03cdcad11a,rdsTempsdb,13,Cathy
I,Names_6d152ce62d,rdsTempsdb,15,Jane
I,Names_6d152ce62d,rdsTempsdb,24,Chris
I,Names_03cdcad11a,rdsTempsdb,16,Mike
```

여기서 각 행의 트랜잭션 작업은 첫 번째 열에 I(삽입), U(업데이트) 또는 D(삭제)로 표시됩니다. 테이블 이름은 두 번째 열 값입니다(예:Names_03cdcad11a). 데이터베이스 스키마의 이름은 세 번째 열의 값입니다(예:rdsTempsdb). 그리고 나머지 열은 사용자 고유의 행 데이터(예:13,Daniel)로 채워집니다.

또한 다음 AWS DMS 명명 규칙에 따라 타임스탬프를 사용하여 Amazon S3 대상에 생성하는 트랜잭션 파일의 이름을 지정합니다.

```
CDC_TXN-timestamp.csv
```

여기서 *timestamp*는 다음 예와 같이 트랜잭션 파일이 생성된 시간입니다.

```
CDC_TXN-20201117153046033.csv
```

파일 이름의 이 타임스탬프를 사용하면 트랜잭션 디렉터리에 트랜잭션 파일을 나열할 때 트랜잭션 파일이 트랜잭션 순서대로 생성되고 나열됩니다.

Note

트랜잭션 순서로 데이터 변경 사항을 캡처할 때는 AWS DMS 항상 대상의 DataFormat S3 설정 값에 관계없이 행 변경 사항을 .csv 파일에 저장합니다.

데이터 복제 작업 중에 Amazon S3 대상에 대한 쓰기 빈도를 제어하기 위해 CdcMaxBatchInterval 및 CdcMinFileSize 설정을 구성할 수 있습니다. 따라서 추가 오버헤드 작업 없이 데이터를 분석할 때 성능이 향상될 수 있습니다. 자세한 내용은 [Amazon S3를 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요.

AWS DMS 에 모든 행 변경 사항을 트랜잭션 순서로 저장하도록 지시하려면

1. 대상의 PreserveTransactions S3 설정을 true로 설정합니다.
2. 대상의 CdcPath S3 설정을 .csv 트랜잭션 파일을 저장할 상대 폴더 경로 AWS DMS 로 설정합니다.

AWS DMS 는 기본 S3 대상 버킷 및 작업 디렉터리 아래 또는 대상의 BucketName 및 BucketFolder S3 설정을 사용하여 지정하는 버킷 및 버킷 폴더 아래에이 경로를 생성합니다.

마이그레이션된 S3 데이터에 소스 DB 작업 표시

가 레코드를 S3 대상으로 AWS DMS 마이그레이션할 때 마이그레이션된 각 레코드에 추가 필드를 생성할 수 있습니다. 이 추가 필드는 소스 데이터베이스의 레코드에 적용된 작업을 나타냅니다. 가이 첫 번째 필드를 AWS DMS 생성하고 설정하는 방법은 includeOpForFullLoad, 및의 마이그레이션 작업 유형 및 설정에 따라 달라집니다cdcInsertsOnlycdcInsertsAndUpdates.

includeOpForFullLoad가 true인 전체 로드의 경우, AWS DMS 는 항상 각 .csv 레코드에 첫 번째 추가 필드를 생성합니다. 이 필드에는 행이 소스 데이터베이스에 삽입되었음을 나타내는 문자 I(INSERT)가 들어 있습니다. 이 false (기본값)cdcInsertsOnly인 경우 CDC 로드의 경우 AWS DMS 는 항상 각 .csv 또는 .parquet 레코드에 첫 번째 필드를 추가로 생성합니다. 이 필드에는

소스 데이터베이스에서 행이 삽입, 업데이트 또는 삭제되었는지 여부를 나타내는 문자 I(INSERT), U(UPDATE) 또는 D(DELETE)가 포함됩니다.

다음 표에서 `includeOpForFullLoad` 및 `cdcInsertsOnly` 속성의 설정들이 작용하여 마이그레이션된 레코드의 설정에 영향을 주는 것을 볼 수 있습니다.

이 파라미터 설정을 사용할 경우		DMS는 .csv 및 .parquet 출력에 다음과 같이 대상 레코드를 설정합니다.	
<code>includeOpForFullLoad</code>	<code>cdcInsertsOnly</code>	전체 로드	CDC 로드
true	true	추가된 첫 번째 필드 값은 I로 설정	추가된 첫 번째 필드 값은 I로 설정
false	false	추가된 필드 없음	추가된 첫 번째 필드 값은 I, U 또는 D로 설정
false	true	추가된 필드 없음	추가된 필드 없음
true	false	추가된 첫 번째 필드 값은 I로 설정	추가된 첫 번째 필드 값은 I, U 또는 D로 설정

`includeOpForFullLoad` 및 `cdcInsertsOnly`가 같은 값으로 설정되면 대상 레코드는 현재 마이그레이션 유형에 대한 레코드 설정을 제어하는 속성에 따라 설정됩니다. 이 속성은 전체 로드의 경우 `includeOpForFullLoad`이고 CDC 로드의 경우 `cdcInsertsOnly`입니다.

`includeOpForFullLoad` 및 `cdcInsertsOnly`가 서로 다른 값으로 설정된 경우는 CDC 및 전체 로드 모두에 대해 대상 레코드 설정을 일관되게 AWS DMS 만듭니다. 이를 위해 `includeOpForFullLoad`에서 이전에 지정한 전체 로드와 관련된 레코드 설정에 부합하도록 CDC 로드와 관련된 레코드 설정을 지정합니다.

즉, 삽입된 레코드를 나타내는 첫 번째 필드를 추가하도록 전체 로드가 설정되었다고 가정하세요. 이 경우 다음 CDC 로드는 삽입, 업데이트 또는 삭제된 레코드가 소스에 적절함을 나타내는 첫 번째 필드를 추가하도록 설정됩니다. 이와 달리, 삽입된 레코드를 나타내는 첫 번째 필드를 추가하지 않도록 전체 로드가 설정되었다고 가정하세요. 이 경우 CDC 로드 역시, 소스에서의 해당 레코드 작업에 관계없이 각 레코드에 첫 번째 필드를 추가하지 않도록 설정됩니다.

마찬가지로 DMS가 첫 번째 추가 필드를 생성하고 설정하는 방법은 `includeOpForFullLoad` 및 `cdcInsertsAndUpdates`의 설정에 따라 다릅니다. 다음 표에서 `includeOpForFullLoad` 및 `cdcInsertsAndUpdates` 속성의 설정들이 작용하여 이 형식으로 마이그레이션된 레코드의 설정에 영향을 주는 것을 볼 수 있습니다.

이 파라미터 설정을 사용할 경우		DMS는 .csv 출력에 다음과 같이 대상 레코드를 설정합니다.	
<code>includeOpForFullLoad</code>	<code>cdcInsertsAndUpdates</code>	전체 로드	CDC 로드
true	true	추가된 첫 번째 필드 값은 I로 설정	추가된 첫 번째 필드 값은 I 또는 U로 설정
false	false	추가된 필드 없음	추가된 첫 번째 필드 값은 I, U 또는 D로 설정
false	true	추가된 필드 없음	추가된 첫 번째 필드 값은 I 또는 U로 설정
true	false	추가된 첫 번째 필드 값은 I로 설정	추가된 첫 번째 필드 값은 I, U 또는 D로 설정

S3 Parquet의 대상 데이터 유형

다음 표에는 사용 시 지원되는 Parquet 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

AWS DMS 데이터 유형	S3 Parquet 데이터 형식
BYTES	BINARY
DATE	DATE32

AWS DMS 데이터 유형	S3 Parquet 데이터 형식
TIME	TIME32
DATETIME	TIMESTAMP
INT1	INT8
INT2	INT16
INT4	INT32
INT8	INT64
NUMERIC	DECIMAL
REAL4	FLOAT
REAL8	DOUBLE
STRING	STRING
UINT1	UINT8
UINT2	UINT16
UINT4	UINT32
UINT8	UINT64
WSTRING	STRING
BLOB	BINARY
NCLOB	STRING
CLOB	STRING
BOOLEAN	BOOL

Amazon DynamoDB 데이터베이스를의 대상으로 사용 AWS Database Migration Service

AWS DMS 를 사용하여 데이터를 Amazon DynamoDB 테이블로 마이그레이션할 수 있습니다. Amazon DynamoDB는 원활한 확장성과 함께 빠르고 예측 가능한 성능을 제공하는 완전 관리형 NoSQL 데이터베이스 서비스입니다.는 관계형 데이터베이스 또는 MongoDB를 소스로 사용할 수 있도록 AWS DMS 지원합니다.

DynamoDB에서 테이블, 항목 및 속성은 작업 시 사용하는 핵심 구성 요소입니다. 테이블은 항목의 컬렉션이고 각 항목은 속성의 컬렉션입니다. DynamoDB는 파티션 키라는 프라이머리 키를 사용하여 테이블의 각 항목을 고유하게 식별합니다. 또한 키와 보조 인덱스를 사용할 수 있어 쿼리 유연성이 향상됩니다.

객체 매핑을 사용하여 소스 데이터베이스에서 대상 DynamoDB 테이블로 데이터를 마이그레이션할 수 있습니다. 객체 매핑을 통해 소스 데이터가 대상에 배치되는 위치를 결정할 수 있습니다.

가 DynamoDB 대상 엔드포인트에 테이블을 AWS DMS 생성할 때 소스 데이터베이스 엔드포인트에 있는 만큼 테이블을 생성합니다. AWS DMS 또한는 여러 DynamoDB 파라미터 값을 설정합니다. 테이블 생성 비용은 마이그레이션할 테이블 수와 데이터 양에 따라 달라집니다.

Note

AWS DMS 콘솔 또는 API의 SSL 모드 옵션은 Kinesis 및 DynamoDB와 같은 일부 데이터 스트리밍 및 NoSQL 서비스에는 적용되지 않습니다. 기본적으로 안전하므로 SSL 모드 설정이 없음(SSL 모드=없음)과 같음을 AWS DMS 보여줍니다. 엔드포인트에서 SSL을 사용하기 위한 추가 구성을 제공할 필요는 없습니다. 예를 들어 DynamoDB를 대상 엔드포인트로 사용하는 경우 기본적으로 안전합니다. DynamoDB에 대한 모든 API 호출은 SSL을 사용하므로 AWS DMS 엔드포인트에 추가 SSL 옵션이 필요하지 않습니다. DynamoDB 데이터베이스에 연결할 때 기본적으로 AWS DMS 가 사용하는 HTTPS 프로토콜을 사용하여 SSL 엔드포인트를 통해 안전하게 데이터를 넣고 검색할 수 있습니다.

전송 속도를 높이기 위해서는 DynamoDB 대상 인스턴스에 대한 멀티스레드 전체 로드를 AWS DMS 지원합니다. DMS는 다음과 같은 작업 설정을 통해 이 멀티스레딩을 지원합니다.

- `MaxFullLoadSubTasks` – 병렬로 로드할 최대 소스 테이블 수를 표시하려면 이 옵션을 사용합니다. DMS는 전용 하위 작업을 사용하여 각 테이블을 해당 DynamoDB 대상 테이블에 로드합니다. 기본값은 8입니다. 최대값은 49입니다.

- `ParallelLoadThreads` -이 옵션을 사용하여 각 테이블을 DynamoDB 대상 테이블에 로드하는데 AWS DMS 사용하는 스레드 수를 지정합니다. 기본값은 0입니다(단일 스레드). 최대값은 200입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.

 Note

DNMS는 로딩을 위해 테이블의 각 세그먼트를 고유의 스레드에 할당합니다. 따라서 `ParallelLoadThreads`를 소스의 테이블에 지정하는 최대 세그먼트 수로 설정합니다.

- `ParallelLoadBufferSize` - 병렬 로드 스레드에서 데이터를 DynamoDB 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 `ParallelLoadThreads`와 함께 사용하세요. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다.
- 개별 테이블에 대한 테이블 매핑 설정 - `table-settings` 규칙을 사용하여 소스에서 병렬로 로드할 개별 테이블을 식별합니다. 또한 멀티스레드 로딩을 위해 각 테이블의 행을 세그먼트화하는 방법도 이 규칙을 사용하여 지정합니다. 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 단원을 참조하십시오.

 Note

가 마이그레이션 작업에 DynamoDB 파라미터 값을 AWS DMS 설정하면 기본 읽기 용량 단위(RCU) 파라미터 값이 200으로 설정됩니다.

쓰기 용량 단위(WCU) 파라미터 값도 설정되지만 이 파라미터의 값은 다른 여러 설정에 따라 달라집니다.

- WCU 파라미터의 기본값은 200입니다.
- `ParallelLoadThreads` 작업 설정이 1(기본값은 0)보다 크게 설정되면 WCU 파라미터는 `ParallelLoadThreads` 값의 200배로 설정됩니다.
- 사용하는 리소스에는 표준 AWS DMS 사용 요금이 적용됩니다.

관계형 데이터베이스에서 DynamoDB 테이블로 마이그레이션

AWS DMS 는 DynamoDB 스칼라 데이터 형식으로 데이터 마이그레이션을 지원합니다. Oracle 또는 MySQL 같은 관계형 데이터베이스에서 DynamoDB로 마이그레이션할 때에는 이 데이터를 저장하는 방법을 다시 구성하는 것이 좋습니다.

현재 AWS DMS 는 DynamoDB 스칼라 유형 속성으로 단일 테이블 재구성을 지원합니다. 관계형 데이터베이스 테이블에서 데이터를 DynamoDB로 마이그레이션하려는 경우, 테이블에서 데이터를 가져와서 DynamoDB 스칼라 데이터 형식 속성으로 재지정합니다. 이 속성은 여러 열에서 데이터를 수락하여 열을 속성에 직접 매핑할 수 있습니다.

AWS DMS 는 다음과 같은 DynamoDB 스칼라 데이터 형식을 지원합니다.

- 문자열
- 숫자
- 부울

 Note

소스의 NULL 데이터는 대상에서 무시됩니다.

DynamoDB를의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service

DynamoDB 데이터베이스를 대상으로 사용하기 전에 IAM 역할을 생성해야 AWS DMS합니다. 이 IAM 역할은 AWS DMS 가 마이그레이션 중인 DynamoDB 테이블에 대한 액세스 권한을 수임하고 부여할 수 있도록 허용해야 합니다. 최소 액세스 권한 집합이 다음 IAM 정책에 나와 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```


DynamoDB로의 마이그레이션에 사용하는 역할에는 다음 권한이 있어야 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:PutItem",
        "dynamodb:CreateTable",
        "dynamodb:DescribeTable",
        "dynamodb>DeleteTable",
        "dynamodb>DeleteItem",
        "dynamodb:UpdateItem"
      ],
      "Resource": [
        "arn:aws:dynamodb:us-west-2:111122223333:table/name1",
        "arn:aws:dynamodb:us-west-2:111122223333:table/OtherName*",
        "arn:aws:dynamodb:us-west-2:111122223333:table/awsdms_apply_exceptions",
        "arn:aws:dynamodb:us-west-2:111122223333:table/awsdms_full_load_exceptions"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:ListTables"
      ],
      "Resource": "*"
    }
  ]
}
```

DynamoDB를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

DynamoDB를 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- DynamoDB는 숫자 데이터 형식의 정밀도를 38자리로 제한합니다. 정밀도가 더 높은 모든 데이터 형식은 문자열로 저장합니다. 객체 매핑 기능을 사용하여 이것을 명시적으로 지정해야 합니다.
- DynamoDB에는 날짜 데이터 형식이 없기 때문에 날짜 데이터 형식을 사용한 데이터는 문자열로 변환됩니다.
- DynamoDB에서는 프라이머리 키 속성 업데이트를 허용하지 않습니다. 대상에 원치 않는 데이터가 발생할 수 있으므로 변경 데이터 캡처(CDC)와 함께 지속적인 복제 사용 시 이 제한이 중요합니다. 객체 매핑 형태에 따라 프라이머리 키를 업데이트하는 CDC 작업은 두 가지 중 하나를 수행할 수 있습니다. 즉, 실패하거나 업데이트된 키와 불완전한 데이터로 새 항목을 삽입할 수 있습니다.
- AWS DMS 는 비복합 기본 키가 있는 테이블의 복제만 지원합니다. 예외는 사용자 지정 파티션 키나 정렬 키 또는 두 키를 모두 사용하여 대상 테이블의 객체 매핑을 지정하는 경우입니다.
- AWS DMS 는 CLOB가 아닌 한 LOB 데이터를 지원하지 않습니다. 는 데이터를 마이그레이션할 때 CLOB 데이터를 DynamoDB 문자열로 AWS DMS 변환합니다.
- DynamoDB를 대상으로 사용하는 경우 예외 적용 제어 테이블(dmslogs.aws_dms_apply_exceptions)만 지원됩니다. 제어 테이블에 대한 자세한 내용은 [제어 테이블 작업 설정](#) 섹션을 참조하세요.
- AWS DMS 는 DynamoDB에 TargetTablePrepMode=TRUNCATE_BEFORE_LOAD 대한 작업 설정을 대상으로 지원하지 않습니다.
- AWS DMS 는 DynamoDB에 TaskRecoveryTableEnabled 대한 작업 설정을 대상으로 지원하지 않습니다.
- BatchApply는 DynamoDB 엔드포인트에는 지원되지 않습니다.
- AWS DMS 는 이름이 DynamoDB의 예약어와 일치하는 속성을 마이그레이션할 수 없습니다. 자세한 내용은 Amazon DynamoDB 개발자 안내서의 [DynamoDB의 예약어](#)를 참조하세요.

객체 매핑을 사용하여 데이터를 DynamoDB로 마이그레이션

AWS DMS 는 테이블 매핑 규칙을 사용하여 소스의 데이터를 대상 DynamoDB 테이블로 매핑합니다. 데이터를 DynamoDB 대상에 매핑하려면 객체 매핑이라는 테이블 매핑 규칙 유형을 사용합니다. 객체 매핑을 통해 마이그레이션할 속성 이름과 데이터를 정의할 수 있습니다. 객체 매핑을 사용할 때는 선택 규칙이 있어야 합니다.

DynamoDB에는 파티션 키와 정렬 키(선택 사항) 외에 사전 설정 구조가 없습니다. 비복합 기본 키가 있는 경우 AWS DMS 에서 사용합니다. 복합 프라이머리 키가 있거나 정렬 키를 사용하려는 경우, 대상 DynamoDB 테이블에서 이러한 키와 기타 속성을 정의하세요.

객체 매핑 규칙을 만들려면 rule-type을 object-mapping으로 지정합니다. 이 규칙은 사용할 객체 매핑의 유형을 지정합니다.

이 규칙의 구조는 다음과 같습니다.

```
{ "rules": [
  {
    "rule-type": "object-mapping",
    "rule-id": "<id>",
    "rule-name": "<name>",
    "rule-action": "<valid object-mapping rule action>",
    "object-locator": {
      "schema-name": "<case-sensitive schema name>",
      "table-name": ""
    },
    "target-table-name": "<table_name>"
  }
]
```

AWS DMS 는 현재 rule-action 파라미터 map-record-to-document에 유효한 유일한 값으로 map-record-to-record 및를 지원합니다. 이러한 값은 exclude-columns 속성 목록의 일부로 제외되지 않은 레코드에 대해 기본적으로 AWS DMS 가 수행하는 작업을 지정합니다. 이러한 값은 어떤 식으로든 속성 매핑에 영향을 미치지 않습니다.

- 관계형 데이터베이스에서 DynamoDB로 마이그레이션할 때 map-record-to-record를 사용할 수 있습니다. 이것은 DynamoDB에서 파티션 키로서 관계형 데이터베이스의 프라이머리 키를 사용하고 소스 데이터베이스의 각 열마다 속성을 생성합니다. 를 사용할 때 exclude-columns 속성 목록에 나열되지 않은 소스 테이블의 모든 열에 map-record-to-record대해는 대상 DynamoDB 인스턴스에 해당 속성을 AWS DMS 생성합니다. 이 속성은 소스 열이 속성 매핑에 사용되는지 여부와 상관없이 생성됩니다.
- 속성 이름 '_doc'를 사용하여 대상에 있는 단일 플랫폼 DynamoDB 맵에 소스 열을 배치하려면 map-record-to-document를 사용합니다. 를 사용하는 경우 map-record-to-document는 데이터를 소스의 단일 플랫폼 DynamoDB 맵 속성에 AWS DMS 배치합니다. 이 속성을 '_doc'라고 합니다. 이 배치는 exclude-columns 속성 목록에 나열되지 않은 소스 테이블의 모든 열에 적용됩니다.

rule-action 파라미터 map-record-to-record와 map-record-to-document의 차이를 이해하는 한 가지 방법은 두 파라미터를 실행하는 것입니다. 이 예에서는 다음 구조와 데이터를 사용하여 관계형 데이터베이스 테이블 행에서 시작한다고 가정합니다.

FirstName	LastName	NickName	WorkAddress	WorkPhone	HomeAddress	HomePhone	Income
▶ Daniel	Sheridan	Dan	101 Main St Cambridge, MA	800-867-5309	100 Secret St, Unknownville, MA	123-456-7890	12345678

이 정보를 DynamoDB로 마이그레이션하려면 데이터를 DynamoDB 테이블 항목으로 매핑하는 규칙을 생성해야 합니다. `exclude-columns` 파라미터에 대해 나열된 열을 기록합니다. 이러한 열은 대상에 직접 매핑되지 않습니다. 그 대신 속성 매핑을 사용하여 데이터를 새 항목으로 결합합니다. 예를 들어 `FirstName`과 `LastName`이 함께 그룹화되어 DynamoDB 대상에서 `CustomerName`이 됩니다. `NickName`과 `income`은 제외되지 않습니다.

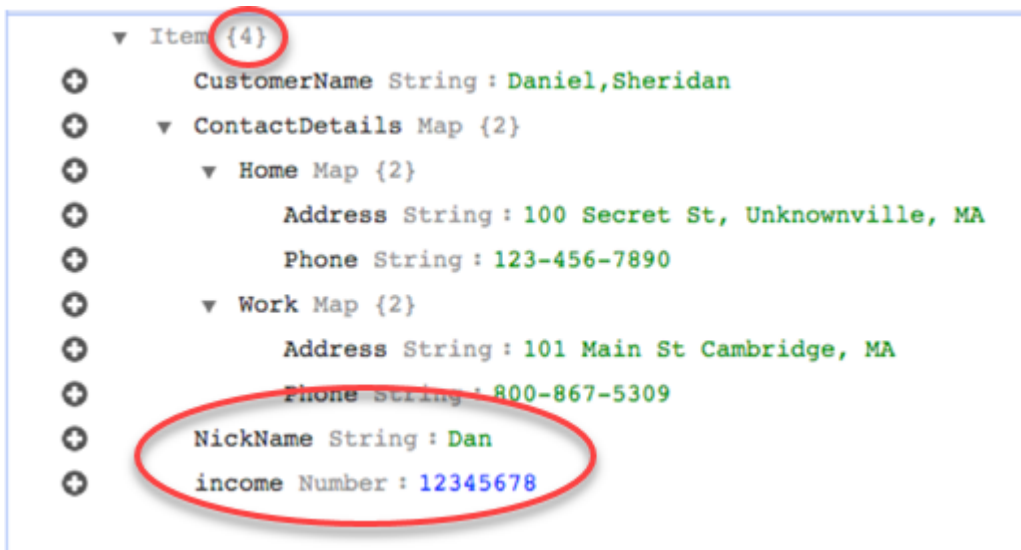
```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "test",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "2",
      "rule-name": "TransformToDDB",
      "rule-action": "map-record-to-record",
      "object-locator": {
        "schema-name": "test",
        "table-name": "customer"
      },
      "target-table-name": "customer_t",
      "mapping-parameters": {
        "partition-key-name": "CustomerName",
        "exclude-columns": [
          "FirstName",
          "LastName",
          "HomeAddress",
          "HomePhone",
          "WorkAddress",
          "WorkPhone"
        ],
        "attribute-mappings": [
          {
            "target-attribute-name": "CustomerName",
            "attribute-type": "scalar",
```

```

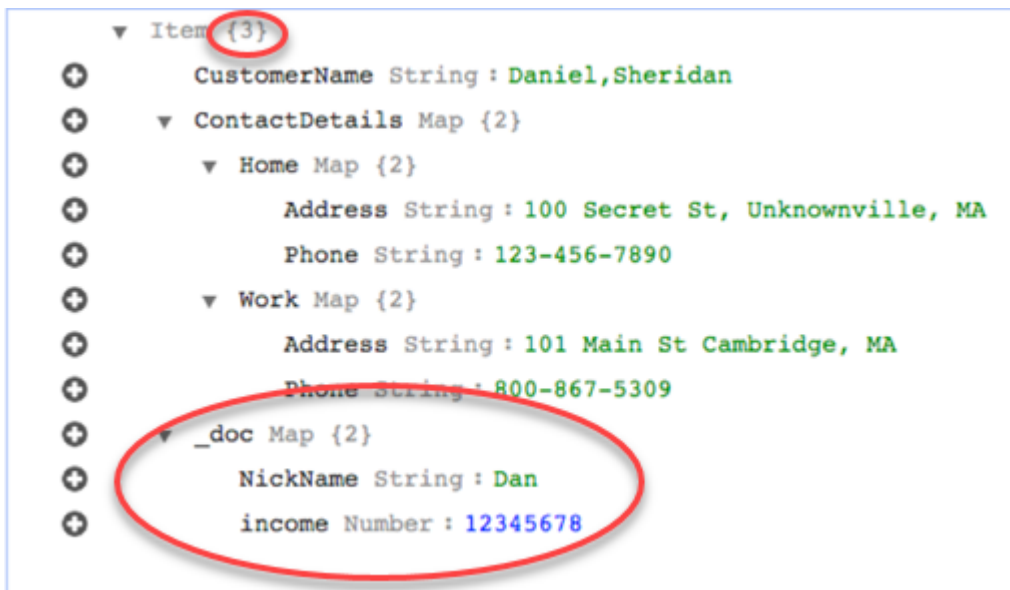
        "attribute-sub-type": "string",
        "value": "${FirstName},${LastName}"
    },
    {
        "target-attribute-name": "ContactDetails",
        "attribute-type": "document",
        "attribute-sub-type": "dynamodb-map",
        "value": {
            "M": {
                "Home": {
                    "M": {
                        "Address": {
                            "S": "${HomeAddress}"
                        },
                        "Phone": {
                            "S": "${HomePhone}"
                        }
                    }
                },
                "Work": {
                    "M": {
                        "Address": {
                            "S": "${WorkAddress}"
                        },
                        "Phone": {
                            "S": "${WorkPhone}"
                        }
                    }
                }
            }
        }
    }
]
}

```

rule-action 파라미터 map-record-to-record를 사용하여 NickName 및 income의 데이터가 DynamoDB 대상에 있는 동일한 이름의 항목에 매핑됩니다.



단, 동일한 규칙을 사용하되 rule-action 파라미터를 map-record-to-document로 변경한다고 가정합니다. 이 경우, exclude-columns 파라미터에 나열되어 있지 않은 열인 NickName 및 income은 _doc 항목으로 매핑됩니다.



객체 매핑과 함께 사용자 지정 조건식 사용

조건식이라는 DynamoDB의 기능을 사용하여 DynamoDB 테이블에 작성되고 있는 데이터를 조작할 수 있습니다. DynamoDB의 조건식에 대한 자세한 내용은 [조건식 섹션](#)을 참조하세요.

조건식 멤버의 구성 요소는 다음과 같습니다.

- 표현식(필수)

- 표현식 속성 값(필수). 속성 값의 DynamoDB json 구조를 지정합니다. 이는 속성을 런타임까지 모를 수 있는 DynamoDB의 값과 비교하는 데 유용합니다. 표현식 속성 값을 실제 값에 대한 자리 표시자로 정의할 수 있습니다.
- 표현식 속성 이름(필수). 이렇게 하면 DynamoDB 예약어, 특수 문자가 포함된 속성 이름 등과의 잠재적 충돌을 방지할 수 있습니다.
- 조건식을 사용할 때 수반되는 옵션(선택 사항). 기본값은 `apply-during-cdc = false`와 `apply-during-full-load = true`입니다.

이 규칙의 구조는 다음과 같습니다.

```
"target-table-name": "customer_t",
  "mapping-parameters": {
    "partition-key-name": "CustomerName",
    "condition-expression": {
      "expression": "<conditional expression>",
      "expression-attribute-values": [
        {
          "name": "<attribute name>",
          "value": <attribute value>
        }
      ],
      "apply-during-cdc": <optional Boolean value>,
      "apply-during-full-load": <optional Boolean value>
    }
  }
```

다음 샘플은 조건식에 사용되는 섹션을 강조한 것입니다.

```

{
  "rules": [
    {
      "rule-type": "object-mapping",
      "rule-id": "1",
      "rule-name": "TransformToDDB",
      "rule-action": "map-record-to-record",
      "object-locator": {
        "schema-name": "test",
        "table-name": "customer",
      },
      "target-table-name": "customer_t",
      "mapping-parameters": {
        "partition-key-name": "CustomerName",
        "condition-expression": {
          "expression": "attribute_not_exists(version) or version <= :record_version",
          "expression-attribute-values": [
            {
              "name": ":record_version",
              "value": {"N": "${version}"}
            }
          ],
          "apply-during-cdc": true,
          "apply-during-full-load": true
        },
        "attribute-mappings": [
          {
            "target-attribute-name": "CustomerName",
            "attribute-type": "scalar",
            "attribute-sub-type": "string",
            "value": "${FirstName},${LastName}"
          }
        ]
      }
    }
  ]
}

```

Object mapping section defines name, rule-action, and object locator information

Condition expression

Options

객체 매핑과 함께 속성 매핑 사용

속성 매핑을 통해 소스 열 이름을 사용한 템플릿 문자열을 지정하여 대상에서 데이터를 재구성할 수 있습니다. 사용자가 템플릿에서 지정한 항목 외에 완료한 서식 변경이 없습니다.

다음 예는 소스 데이터베이스의 구조와 필요한 DynamoDB 대상 구조를 보여 줍니다. 첫 번째는 소스 (이 경우, Oracle 데이터베이스)의 구조를, 그 다음은 필요한 DynamoDB 데이터 구조를 나타냅니다. 이 예는 필요한 대상 구조를 생성하는 데 사용되는 JSON으로 끝납니다.

Oracle 데이터의 구조는 다음과 같습니다.

First	Las	St	Home/	HomeF	WorkAddr	Work	DateOfBirth
			SS		SS		
프라이				해당 사항 없음			
머리 키							

First	Last	Street	HomeAddress	HomePhone	WorkAddress	WorkPhone	DateOfBirth
Randy	Marshall	5 Baker Street	221B	1234567890	31 Spooner Street, Quahog	9876541230	1988/02/29

DynamoDB 데이터의 구조는 다음과 같습니다.

CustomerName	StoreId	ContactDetails	DateOfBirth
파티션 키	정렬 키	해당 사항 없음	
Randy Marshall	5	<pre>{ "Name": "Randy", "Home": { "Address": "221B Baker Street", "Phone": 1234567890 }, "Work": { "Address": "31 Spooner Street, Quahog", "Phone": 9876541230 } }</pre>	02/29/1988

다음 JSON은 DynamoDB 구조를 달성하는 데 사용되는 객체 매핑과 열 매핑을 보여 줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
```

```
"rule-id": "1",
"rule-name": "1",
"object-locator": {
  "schema-name": "test",
  "table-name": "%"
},
"rule-action": "include"
},
{
  "rule-type": "object-mapping",
  "rule-id": "2",
  "rule-name": "TransformToDDB",
  "rule-action": "map-record-to-record",
  "object-locator": {
    "schema-name": "test",
    "table-name": "customer"
  },
  "target-table-name": "customer_t",
  "mapping-parameters": {
    "partition-key-name": "CustomerName",
    "sort-key-name": "StoreId",
    "exclude-columns": [
      "FirstName",
      "LastName",
      "HomeAddress",
      "HomePhone",
      "WorkAddress",
      "WorkPhone"
    ],
    "attribute-mappings": [
      {
        "target-attribute-name": "CustomerName",
        "attribute-type": "scalar",
        "attribute-sub-type": "string",
        "value": "${FirstName},${LastName}"
      },
      {
        "target-attribute-name": "StoreId",
        "attribute-type": "scalar",
        "attribute-sub-type": "string",
        "value": "${StoreId}"
      },
      {
        "target-attribute-name": "ContactDetails",
```

```

        "attribute-type": "scalar",
        "attribute-sub-type": "string",
        "value": "{\"Name\": \"${FirstName}\", \"Home\": {\"Address
\": \"${HomeAddress}\", \"Phone\": \"${HomePhone}\"}, \"Work\": {\"Address\":
\": \"${WorkAddress}\", \"Phone\": \"${WorkPhone}\"}}"}
    }
}
}
}
}
}
}
}

```

열 매핑을 사용하는 또 다른 방법은 DynamoDB 형식을 문서 형식으로 사용하는 것입니다. 다음 코드 예제에서는 dynamodb-map을 속성 매핑을 위한 attribute-sub-type으로 사용합니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "test",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "2",
      "rule-name": "TransformToDDB",
      "rule-action": "map-record-to-record",
      "object-locator": {
        "schema-name": "test",
        "table-name": "customer"
      },
      "target-table-name": "customer_t",
      "mapping-parameters": {
        "partition-key-name": "CustomerName",
        "sort-key-name": "StoreId",
        "exclude-columns": [

```

```
        "FirstName",
        "LastName",
        "HomeAddress",
        "HomePhone",
        "WorkAddress",
        "WorkPhone"
    ],
    "attribute-mappings": [
        {
            "target-attribute-name": "CustomerName",
            "attribute-type": "scalar",
            "attribute-sub-type": "string",
            "value": "${FirstName},${LastName}"
        },
        {
            "target-attribute-name": "StoreId",
            "attribute-type": "scalar",
            "attribute-sub-type": "string",
            "value": "${StoreId}"
        },
        {
            "target-attribute-name": "ContactDetails",
            "attribute-type": "document",
            "attribute-sub-type": "dynamodb-map",
            "value": {
                "M": {
                    "Name": {
                        "S": "${FirstName}"
                    },
                    "Home": {
                        "M": {
                            "Address": {
                                "S": "${HomeAddress}"
                            },
                            "Phone": {
                                "S": "${HomePhone}"
                            }
                        }
                    },
                    "Work": {
                        "M": {
                            "Address": {
                                "S": "${WorkAddress}"
                            }
                        }
                    }
                }
            }
        }
    ]
}
```

```

    "Phone": {
      "S": "${WorkPhone}"
    }
  }
}

```

dynamodb-map 대신 다음 예제와 같이 속성 매핑을 위한 속성-하위 유형으로 dynamodb-list를 사용할 수 있습니다.

```

{
  "target-attribute-name": "ContactDetailsList",
  "attribute-type": "document",
  "attribute-sub-type": "dynamodb-list",
  "value": {
    "L": [
      {
        "N": "${FirstName}"
      },
      {
        "N": "${HomeAddress}"
      },
      {
        "N": "${HomePhone}"
      },
      {
        "N": "${WorkAddress}"
      },
      {
        "N": "${WorkPhone}"
      }
    ]
  }
}

```

예제 1: 객체 매핑과 함께 속성 매핑 사용

다음 예제는 두 MySQL 데이터베이스 테이블인 nfl_data 및 sport_team에서 NFLTeams와 SportTeam이라는 두 개의 DynamoDB 테이블로 데이터를 마이그레이션합니다. 이 테이블의 구조와 MySQL 데이터베이스 테이블의 데이터를 DynamoDB 테이블로 매핑하는 데 사용된 JSON은 다음과 같습니다.

MySQL 데이터베이스 테이블 nfl_data의 구조는 다음과 같습니다.

```
mysql> desc nfl_data;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Position       | varchar(5)    | YES  |     | NULL    |       |
| player_number  | smallint(6)   | YES  |     | NULL    |       |
| Name           | varchar(40)   | YES  |     | NULL    |       |
| status         | varchar(10)   | YES  |     | NULL    |       |
| stat1          | varchar(10)   | YES  |     | NULL    |       |
| stat1_val      | varchar(10)   | YES  |     | NULL    |       |
| stat2          | varchar(10)   | YES  |     | NULL    |       |
| stat2_val      | varchar(10)   | YES  |     | NULL    |       |
| stat3          | varchar(10)   | YES  |     | NULL    |       |
| stat3_val      | varchar(10)   | YES  |     | NULL    |       |
| stat4          | varchar(10)   | YES  |     | NULL    |       |
| stat4_val      | varchar(10)   | YES  |     | NULL    |       |
| team           | varchar(10)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
```

MySQL 데이터베이스 테이블 sport_team의 구조는 다음과 같습니다.

```
mysql> desc sport_team;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id             | mediumint(9)  | NO   | PRI | NULL    | auto_increment |
| name           | varchar(30)   | NO   |     | NULL    |       |
| abbreviated_name | varchar(10)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
```

home_field_id	smallint(6)	YES	MUL	NULL		
sport_type_name	varchar(15)	NO	MUL	NULL		
sport_league_short_name	varchar(10)	NO		NULL		
sport_division_short_name	varchar(10)	YES		NULL		

테이블 2개를 DynamoDB 테이블 2개에 매핑하는 데 사용된 테이블 매핑 규칙은 다음과 같습니다.

```
{
  "rules":[
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "dms_sample",
        "table-name": "nfl_data"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "selection",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "dms_sample",
        "table-name": "sport_team"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "3",
      "rule-name": "MapNFLData",
      "rule-action": "map-record-to-record",
      "object-locator": {
        "schema-name": "dms_sample",
        "table-name": "nfl_data"
      },
      "target-table-name": "NFLTeams",
      "mapping-parameters": {
        "partition-key-name": "Team",
```

```

    "sort-key-name": "PlayerName",
    "exclude-columns": [
        "player_number", "team", "name"
    ],
    "attribute-mappings": [
        {
            "target-attribute-name": "Team",
            "attribute-type": "scalar",
            "attribute-sub-type": "string",
            "value": "${team}"
        },
        {
            "target-attribute-name": "PlayerName",
            "attribute-type": "scalar",
            "attribute-sub-type": "string",
            "value": "${name}"
        },
        {
            "target-attribute-name": "PlayerInfo",
            "attribute-type": "scalar",
            "attribute-sub-type": "string",
            "value": "{\"Number\": \"${player_number}\", \"Position\": \"${Position}\",
\"Status\": \"${status}\", \"Stats\": {\"Stat1\": \"${stat1}:${stat1_val}\", \"Stat2\":
\"${stat2}:${stat2_val}\", \"Stat3\": \"${stat3}:${stat3_val}\", \"Stat4\": \"${stat4}:${stat4_val}\"}"}"
        }
    ]
},
{
    "rule-type": "object-mapping",
    "rule-id": "4",
    "rule-name": "MapSportTeam",
    "rule-action": "map-record-to-record",
    "object-locator": {
        "schema-name": "dms_sample",
        "table-name": "sport_team"
    },
    "target-table-name": "SportTeams",
    "mapping-parameters": {
        "partition-key-name": "TeamName",
        "exclude-columns": [
            "name", "id"
        ]
    }
}

```



```

    "attribute-mappings":[
      {
        "target-attribute-name":"TeamName",
        "attribute-type":"scalar",
        "attribute-sub-type":"string",
        "value":"${name}"
      },
      {
        "target-attribute-name":"TeamInfo",
        "attribute-type":"scalar",
        "attribute-sub-type":"string",
        "value":"{\"League\": \"${sport_league_short_name}\",\"Division\": \"${sport_division_short_name}\"}"
      }
    ]
  }
}
]
}

```

NFLTeams DynamoDB 테이블의 샘플 출력은 다음과 같습니다.

```

  "PlayerInfo": "{\"Number\": \"6\", \"Position\": \"P\", \"Status\": \"ACT\", \"Stats\": {\"Stat1\": \"PUNTS:73\", \"Stat2\": \"AVG:46\", \"Stat3\": \"LNG:67\", \"Stat4\": \"IN 20:31\"}}",
  "PlayerName": "Allen, Ryan",
  "Position": "P",
  "stat1": "PUNTS",
  "stat1_val": "73",
  "stat2": "AVG",
  "stat2_val": "46",
  "stat3": "LNG",
  "stat3_val": "67",
  "stat4": "IN 20",
  "stat4_val": "31",
  "status": "ACT",
  "Team": "NE"
}

```

SportsTeams DynamoDB 테이블의 샘플 출력은 다음과 같습니다.

```
{
  "abbreviated_name": "IND",
  "home_field_id": 53,
  "sport_division_short_name": "AFC South",
  "sport_league_short_name": "NFL",
  "sport_type_name": "football",
  "TeamInfo": "{\"League\": \"NFL\", \"Division\": \"AFC South\"}",
  "TeamName": "Indianapolis Colts"
}
```

DynamoDB에 대한 대상 데이터 형식

용 DynamoDB 엔드포인트는 대부분의 DynamoDB 데이터 형식을 AWS DMS 지원합니다. 다음 표에 는를 사용할 때 지원되는 Amazon AWS DMS 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식 의 기본 매핑이 나와 있습니다.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

가 이기종 데이터베이스에서 데이터를 AWS DMS 마이그레이션하면 소스 데이터베이스의 데이터 형 식을 AWS DMS 데이터 유형이라고 하는 중간 데이터 유형으로 매핑합니다. 그런 다음, 중간 데이터 형 식을 대상 데이터 형식에 매핑합니다. 다음 표에는 DynamoDB에서 매핑되는 각 AWS DMS 데이터 유 형과 데이터 유형이 나와 있습니다.

AWS DMS 데이터 유형	DynamoDB 데이터 형식
문자열	문자열
WString	문자열
부울	부울
날짜	문자열
DateTime	문자열
INT1	숫자
INT2	숫자

AWS DMS 데이터 유형	DynamoDB 데이터 형식
INT4	숫자
INT8	숫자
Numeric	숫자
Real4	숫자
Real8	숫자
UINT1	숫자
UINT2	숫자
UINT4	숫자
UINT8	숫자
CLOB	문자열

Amazon Kinesis Data Streams를 대상으로 사용 AWS Database Migration Service

AWS DMS 를 사용하여 데이터를 Amazon Kinesis 데이터 스트림으로 마이그레이션할 수 있습니다. Amazon Kinesis Data Streams는 Amazon Kinesis Data Streams 서비스의 일부입니다. Kinesis Data Streams를 사용하여 대규모 데이터 레코드 스트림을 실시간으로 수집하고 처리할 수 있습니다.

Kinesis Data Streams는 샤드로 구성됩니다. 샤드는 스트림에서 고유하게 식별되는 데이터 레코드 시퀀스입니다. Amazon Kinesis Data Streams 내 샤드에 관한 자세한 내용은 Amazon Kinesis Data Streams 개발자 안내서의 [샤드](#)를 참조하세요.

AWS Database Migration Service 는 JSON을 사용하여 Kinesis 데이터 스트림에 레코드를 게시합니다. 변환 중 AWS DMS 는 각 레코드를 소스 데이터베이스로부터 JSON 형식 또는 JSON_UNFORMATTED 메시지 형식의 속성-값 페어로 직렬화합니다. JSON_UNFORMATTED 메시지 형식은 줄 바꿈 구분 기호가 있는 한 줄 JSON 문자열입니다. 이 형식을 사용하면 Amazon Data Firehose가 Amazon S3 대상에 Kinesis 데이터를 전송한 다음 Amazon Athena를 비롯한 다양한 쿼리 엔진을 사용하여 데이터를 쿼리할 수 있습니다.

객체 매핑을 사용하여 데이터를 지원되는 데이터 소스에서 대상 스트림으로 마이그레이션합니다. 객체 매핑을 통해 데이터 레코드를 스트림에 구조화하는 방법을 결정합니다. 또한 각 테이블에 대한 파티션 키를 정의합니다. Kinesis Data Streams는 이러한 키를 사용해 데이터를 샤드로 그룹화합니다.

AWS DMS 는 여러 Kinesis Data Streams 파라미터 값도 설정합니다. 테이블 생성 비용은 마이그레이션할 테이블 수와 데이터 양에 따라 달라집니다.

Note

AWS DMS 콘솔 또는 API의 SSL 모드 옵션은 Kinesis 및 DynamoDB와 같은 일부 데이터 스트리밍 및 NoSQL 서비스에는 적용되지 않습니다. 기본적으로 안전하므로 SSL 모드 설정이 없음(SSL 모드=없음)과 같음을 AWS DMS 보여줍니다. 엔드포인트에서 SSL을 사용하기 위한 추가 구성을 제공할 필요는 없습니다. 예를 들어 Kinesis를 대상 엔드포인트로 사용하는 경우, 기본적으로 안전합니다. Kinesis에 대한 모든 API 호출은 SSL을 사용하므로 AWS DMS 엔드포인트에 추가 SSL 옵션이 필요하지 않습니다. Kinesis 데이터 스트림에 연결할 때 기본적으로 AWS DMS 가 사용하는 HTTPS 프로토콜을 사용하여 SSL 엔드포인트를 통해 안전하게 데이터를 넣고 검색할 수 있습니다.

Kinesis Data Streams 엔드포인트 설정

Kinesis Data Streams 대상 엔드포인트를 사용하는 경우 AWS DMS API의 KinesisSettings 옵션을 사용하여 트랜잭션 및 제어 세부 정보를 가져올 수 있습니다.

연결을 설정하는 방법은 다음과 같습니다.

- AWS DMS 콘솔에서 엔드포인트 설정을 사용합니다.
- CLI에서 [CreateEndpoint](#) 명령의 kinesis-settings 옵션을 사용합니다.

CLI에서 다음 kinesis-settings 옵션의 요청 파라미터를 사용합니다.

Note

IncludeNullAndEmpty 엔드포인트 설정에 대한 지원은 AWS DMS 버전 3.4.1 및 이후 버전에서 제공됩니다. 그러나 Kinesis Data Streams 대상에 대한 다음 다른 엔드포인트 설정에 대한 지원은에서 사용할 수 있습니다 AWS DMS.

- **MessageFormat** – 엔드포인트에 생성된 레코드의 출력 형식입니다. 메시지 형식은 JSON(기본값) 또는 JSON_UNFORMATTED(탭이 없는 한 줄)입니다.
- **IncludeControlDetails** – Kinesis 메시지 출력에서 테이블 정의, 열 정의, 테이블 및 열 변경 사항에 대한 자세한 제어 정보를 표시합니다. 기본값은 false입니다.
- **IncludeNullAndEmpty** – NULL 및 빈 열을 대상에 포함합니다. 기본값은 false입니다.
- **IncludePartitionValue** – 파티션 유형이 schema-table-type이 아닌 경우, Kinesis 메시지 출력 내에 파티션 값을 표시합니다. 기본값은 false입니다.
- **IncludeTableAlterOperations** – 제어 데이터의 테이블을 변경하는 데이터 정의 언어(DDL) 작업(예: rename-table, drop-table, add-column, drop-column 및 rename-column)을 포함합니다. 기본값은 false입니다.
- **IncludeTransactionDetails** – 소스 데이터베이스의 자세한 트랜잭션 정보를 제공합니다. 이 정보에는 커밋 타임스탬프, 로그 위치 및 transaction_id, previous_transaction_id, transaction_record_id (트랜잭션 내의 레코드 오프셋)에 대한 값이 포함됩니다. 기본값은 false입니다.
- **PartitionIncludeSchemaTable** – 파티션 유형이 primary-key-type인 경우 스키마 및 테이블 이름을 파티션 값에 접두사로 지정합니다. 이렇게 하면 Kinesis 샤드 간의 데이터 분산이 증가합니다. 예를 들어 SysBench 스키마에 수천 개의 테이블이 있고 각 테이블에 프라이머리 키의 범위가 제한되어 있다고 가정하겠습니다. 이 경우 동일한 프라이머리 키가 수천 개의 테이블에서 동일한 샤드로 전송되어 스로틀링이 발생합니다. 기본값은 false입니다.
- **UseLargeIntegerValue** - 3.5.4 AWS DMS 버전에서 사용할 수 있는 두 배로 묶는 대신 최대 18 자리 int를 사용합니다. 기본값은 false입니다.

다음 예제는 AWS CLI를 사용하여 실행한 예제 create-endpoint 명령과 함께 사용 중인 kinesis-settings 옵션을 보여줍니다.

```
aws dms \
  create-endpoint \
    --region <aws-region> \
    --endpoint-identifier <user-endpoint-identifier> \
    --endpoint-type target \
    --engine-name kinesis \
    --kinesis-settings ServiceAccessRoleArn=arn:aws:iam::<account-id>:role/<kinesis-
role-name>,StreamArn=arn:aws:kinesis:<aws-region>:<account-id>:stream/<stream-
name>,MessageFormat=json-unformatted,
IncludeControlDetails=true,IncludeTransactionDetails=true,IncludePartitionValue=true,PartitionI
IncludeTableAlterOperations=true
```

멀티스레드 전체 로드 작업 설정

전송 속도를 높이기 위해서는 Kinesis Data Streams 대상 인스턴스에 대한 멀티스레드 전체 로드를 AWS DMS 지원합니다. DMS는 다음과 같은 작업 설정을 통해 이 멀티스레딩을 지원합니다.

- **MaxFullLoadSubTasks** – 병렬로 로드할 최대 소스 테이블 수를 표시하려면 이 옵션을 사용합니다. DMS는 전용 하위 태스크를 사용하여 각 테이블을 해당 Kinesis 대상 테이블에 로드합니다. 기본값은 8이며 최대값은 49입니다.
- **ParallelLoadThreads** -이 옵션을 사용하여 각 테이블을 Kinesis 대상 테이블에 로드하는 데 AWS DMS 사용하는 스레드 수를 지정합니다. Kinesis Data Streams 대상의 최대값은 32입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.
- **ParallelLoadBufferSize** – 이 옵션을 사용하여 병렬 로드 스레드에서 데이터를 Kinesis 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 **ParallelLoadThreads**와 함께 사용하세요. **ParallelLoadBufferSize**는 둘 이상의 스레드가 있는 경우에만 유효합니다.
- **ParallelLoadQueuesPerThread** – 각 동시 스레드가 액세스하는 대기열 수를 지정하여 대기열에서 데이터 레코드를 가져오고 대상에 대한 배치 로드를 생성하려면 이 옵션을 사용합니다. 기본값은 1입니다. 그러나 다양한 페이로드 크기의 Kinesis 대상에서 스레드당 대기열 수의 유효 범위는 5~512입니다.

멀티스레드 CDC 로드 작업 설정

PutRecords API 직접 호출의 동작을 수정하는 데 작업 설정을 사용하여 Kinesis와 같은 실시간 데이터 스트리밍 대상 엔드포인트에 대한 변경 데이터 캡처(CDC)의 성능을 향상시킬 수 있습니다. 이렇게 하려면 **ParallelApply*** 작업 설정을 사용하여 동시 스레드 수, 스레드당 대기열 및 버퍼에 저장할 레코드 수를 지정해야 합니다. 예를 들어 CDC 로드를 수행하고 128개의 스레드를 병렬로 적용한다고 가정하겠습니다. 또한 버퍼당 50개 레코드가 저장된 스레드당 64개 대기열에 액세스하려고 합니다.

CDC 성능을 높이기 위해서는 다음 작업 설정을 AWS DMS 지원합니다.

- **ParallelApplyThreads** - CDC 로드 중에는 데이터 레코드를 Kinesis 대상 엔드포인트로 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- **ParallelApplyBufferSize** – CDC 로드 중에 Kinesis 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대값은 1,000입니다. **ParallelApplyThreads**가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.

- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Kinesis 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다. 기본값은 1이고 최대값은 512입니다.

`ParallelApply*` 작업 설정을 사용할 때 `partition-key-type` 기본값은 테이블의 `schema-name.table-name`가 아니라 `primary-key`입니다.

Kinesis 데이터 스트림 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기

Kinesis 같은 데이터 스트리밍 대상에 CDC 업데이트를 작성하는 경우 업데이트를 통한 변경 전에 소스 데이터베이스 행의 원래 값을 볼 수 있습니다. 이를 가능하게 하기 위해서는 소스 데이터베이스 엔진에서 제공하는 데이터를 기반으로 업데이트 이벤트의 이전 이미지를 AWS DMS 채웁니다.

소스 데이터베이스 엔진에 따라 이전 이미지에 대해 서로 다른 양의 정보를 제공합니다.

- Oracle은 열이 변경되는 경우에만 열에 대한 업데이트를 제공합니다.
- PostgreSQL은 프라이머리 키의 일부인 열에 대한 데이터(변경 여부에 관계 없음)만 제공합니다. 모든 열에 대한 데이터를 제공(변경 여부에 관계 없음)하려면 `REPLICA_IDENTITY`를 `DEFAULT` 대신 `FULL`로 설정해야 합니다. 참고로 각 테이블의 `REPLICA_IDENTITY` 설정을 신중하게 선택해야 합니다. `REPLICA_IDENTITY`를 `FULL`로 설정하면 모든 열 값이 계속해서 미리 쓰기 로깅(WAL)에 기록됩니다. 이로 인해 자주 업데이트되는 테이블에 성능 또는 리소스 문제가 발생할 수 있습니다.
- MySQL은 일반적으로 BLOB 및 CLOB 데이터 형식을 제외한 모든 열에 대한 데이터(변경 여부에 관계 없음)를 제공합니다.

이전 이미징을 활성화하여 소스 데이터베이스의 원래 값을 AWS DMS 출력에 추가하려면 `BeforeImageSettings` 태스크 설정 또는 `add-before-image-columns` 파라미터를 사용합니다. 이 파라미터는 열 변환 규칙을 적용합니다.

`BeforeImageSettings`는 다음과 같이 소스 데이터베이스 시스템에서 수집된 값을 사용하여 모든 업데이트 작업에 새 JSON 속성을 추가합니다.

```
"BeforeImageSettings": {
  "EnableBeforeImage": boolean,
  "FieldName": string,
  "ColumnFilter": pk-only (default) / non-lob / all (but only one)
}
```

Note

전체 로드 및 CDC BeforeImageSettings AWS DMS 작업(기존 데이터를 마이그레이션하고 진행 중인 변경 사항을 복제)과 같은 CDC 구성 요소가 포함된 작업 또는 CDC 전용 작업(데이터 변경 사항만 복제)에만 적용됩니다. 전체 로드 전용 작업에는 BeforeImageSettings를 적용하지 마세요.

BeforeImageSettings 옵션의 경우, 다음이 적용됩니다.

- 이전 이미징을 활성화하려면 EnableBeforeImage 옵션을 true로 설정합니다. 기본값은 false입니다.
- FieldName 옵션을 사용하여 새 JSON 속성에 이름을 지정합니다. EnableBeforeImage가 true인 경우 FieldName은 필수이며 비워 둘 수 없습니다.
- ColumnFilter 옵션은 이전 이미징을 사용하여 추가할 열을 지정합니다. 테이블 프라이머리 키의 일부에 속하는 열만 추가하려면 기본값 pk-only를 사용합니다. 이전 이미지 값이 있는 모든 열을 추가하려면 all을 사용합니다. 이전 이미지에는 CLOB 또는 BLOB와 같은 LOB 데이터 형식의 열이 포함되지 않는다는 점에 유의하세요.

```
"BeforeImageSettings": {
  "EnableBeforeImage": true,
  "FieldName": "before-image",
  "ColumnFilter": "pk-only"
}
```

Note

Amazon S3 대상은 BeforeImageSettings를 지원하지 않습니다. S3 대상의 경우 add-before-image-columns 변환 규칙만 사용하여 CDC 중에 이전 이미징을 수행합니다.

이전 이미지 변환 규칙 사용

작업 설정 대신 열 변환 규칙을 적용하는 add-before-image-columns 파라미터를 사용할 수 있습니다. 이 파라미터를 사용하면 Kinesis 같은 데이터 스트리밍 대상에서 CDC 중에 이전 이미징을 활성화할 수 있습니다.

변환 규칙에 `add-before-image-columns`를 사용하면 이전 이미지 결과를 보다 세밀하게 제어할 수 있습니다. 변환 규칙을 통해 객체 로케이터를 사용하여 규칙에 대해 선택한 테이블을 제어할 수 있습니다. 또한 변환 규칙을 함께 연결하여 테이블마다 서로 다른 규칙을 적용할 수 있습니다. 그런 다음, 다른 규칙을 사용하여 생성된 열을 조작할 수 있습니다.

Note

동일한 작업 내에서 `BeforeImageSettings` 작업 설정과 함께 `add-before-image-columns` 파라미터를 사용해서는 안 됩니다. 단일 작업에는 이 파라미터 또는 설정 중 하나만 사용하고 둘 다 사용하지는 마세요.

해당 열에 대해 `add-before-image-columns` 파라미터가 있는 `transformation` 규칙 유형이 `before-image-def` 섹션을 제공해야 합니다. 다음은 그 한 예입니다.

```
{
  "rule-type": "transformation",
  ...
  "rule-target": "column",
  "rule-action": "add-before-image-columns",
  "before-image-def":{
    "column-filter": one-of (pk-only / non-lob / all),
    "column-prefix": string,
    "column-suffix": string,
  }
}
```

`column-prefix`의 값은 열 이름 앞에 추가되며, `column-prefix`의 기본값은 `BI_`입니다. `column-suffix`의 값은 열 이름 뒤에 추가되며, 기본값은 비어 있습니다. `column-prefix`와 `column-suffix`를 모두 빈 문자열로 설정하지 마세요.

`column-filter`에 대해 하나의 값을 선택합니다. 테이블 프라이머리 키의 일부인 열만 추가하려면 `pk-only`를 선택합니다. LOB 유형이 아닌 열만 추가하려면 `non-lob`를 선택합니다. 이전 이미지 값이 있는 모든 열을 추가하려면 `all`을 선택합니다.

이전 이미지 변환 규칙의 예

다음 예의 변환 규칙은 대상에서 `BI_emp_no`라는 새 열을 추가합니다. `UPDATE employees SET emp_no = 3 WHERE emp_no = 1;` 같은 문은 `BI_emp_no` 필드를 1로 채웁니다. Amazon S3 대상에 CDC 업데이트를 작성할 때 `BI_emp_no` 열을 통해 업데이트된 원래 행을 알 수 있습니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-target": "column",
      "object-locator": {
        "schema-name": "%",
        "table-name": "employees"
      },
      "rule-action": "add-before-image-columns",
      "before-image-def": {
        "column-prefix": "BI_",
        "column-suffix": "",
        "column-filter": "pk-only"
      }
    }
  ]
}
```

add-before-image-columns 규칙 작업 사용에 관한 자세한 내용은 [변환 규칙 및 작업](#) 섹션을 참조하세요.

Kinesis 데이터 스트림을의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service

Kinesis 데이터 스트림을의 대상으로 사용하기 위한 IAM 역할 AWS Database Migration Service

Kinesis 데이터 스트림을 대상으로 설정하기 전에 IAM 역할을 생성해야 AWS DMS합니다. 이 역할은 AWS DMS 가 마이그레이션 중인 Kinesis 데이터 스트림에 대한 액세스 권한을 수임하고 부여하도록 허용해야 합니다. 최소 액세스 권한 집합이 다음 IAM 정책에 나와 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Kinesis 데이터 스트림으로 마이그레이션할 때 사용하는 역할에는 다음 권한이 있어야 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kinesis:DescribeStream",
        "kinesis:PutRecord",
        "kinesis:PutRecords"
      ],
      "Resource": "*"
    }
  ]
}
```

의 대상으로 Kinesis 데이터 스트림에 액세스 AWS Database Migration Service

AWS DMS 버전 3.4.7 이상에서 Kinesis 엔드포인트에 연결하려면 다음 중 하나를 수행해야 합니다.

- VPC 엔드포인트를 사용하도록 DMS를 구성합니다. VPC 엔드포인트를 사용하도록 DMS를 구성하는 방법은 [에 대한 VPC 엔드포인트 구성 AWS DMS](#) 섹션을 참조하세요.
- 퍼블릭 경로를 사용하도록 DMS를 구성합니다. 즉, 복제 인스턴스를 퍼블릭으로 설정합니다. 퍼블릭 복제 인스턴스에 대한 자세한 내용은 [퍼블릭 및 프라이빗 복제 인스턴스](#) 섹션을 참조하세요.

Kinesis Data Streams를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

Kinesis Data Streams를 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- AWS DMS 는 트랜잭션에 관계없이 각 업데이트를 소스 데이터베이스의 단일 레코드에 지정된 Kinesis 데이터 스트림의 하나의 데이터 레코드로 게시합니다. 그러나 KinesisSettings API의 관련 파라미터를 사용하여 각 데이터 레코드에 대한 트랜잭션 세부 정보를 포함할 수 있습니다.
- 전체 LOB 모드는 지원되지 않습니다.
- 지원되는 최대 LOB 크기는 1MB입니다.
- Kinesis 데이터 스트림은 중복 제거를 지원하지 않습니다. 스트림의 데이터를 소비하는 애플리케이션은 중복 레코드를 처리해야 합니다. 자세한 내용은 Amazon Kinesis Data Streams 개발자 안내서의 [중복 레코드 처리](#)를 참조하세요.
- AWS DMS 는 파티션 키에 대해 다음 네 가지 양식을 지원합니다.
 - SchemaName.TableName: 스키마와 테이블 이름의 조합입니다.
 - \${AttributeName}: JSON 내 한 필드의 값 또는 소스 데이터베이스 내 테이블의 프라이머리 키입니다.
 - transaction-id: CDC 트랜잭션 ID입니다. 동일한 트랜잭션 내의 모든 레코드는 동일한 파티션으로 이동합니다.
 - constant: 테이블 또는 데이터에 관계없이 모든 레코드에 대해 고정된 리터럴 값입니다. 모든 레코드는 동일한 파티션 키 값 "상수"로 전송되어 모든 테이블에서 엄격한 전역 순서를 제공합니다.

```
{
  "rule-type": "object-mapping",
  "rule-id": "2",
  "rule-name": "PartitionKeyTypeExample",
  "rule-action": "map-record-to-document",
```

```

    "object-locator": {
      "schema-name": "onprem",
      "table-name": "it_system"
    },
    "mapping-parameters": {
      "partition-key-type": "transaction-id | constant | attribute-name | schema-
table"
    }
  }
}

```

- Kinesis Data Streams 내 저장 데이터를 암호화하는 방법에 관한 자세한 내용은 AWS Key Management Service 개발자 안내서의 [Kinesis Data Streams의 데이터 보호](#)를 참조하세요.
- IncludeTransactionDetails 엔드포인트 설정은 소스 엔드포인트가 Oracle, SQL Server, PostgreSQL 또는 MySQL인 경우에만 지원됩니다. 다른 소스 엔드포인트 유형의 경우 트랜잭션 세부 정보가 포함되지 않습니다.
- BatchApply는 Kinesis 엔드포인트에는 지원되지 않습니다. Kinesis 대상에 대해 배치 적용(예: BatchApplyEnabled 대상 메타데이터 작업 설정)를 사용하면 작업 실패와 데이터 손실이 발생할 수 있습니다. Kinesis를 대상 엔드포인트로 사용할 때는 BatchApply를 활성화하지 마세요.
- Kinesis 대상은 복제 인스턴스와 AWS 리전 동일한 AWS 계정의 Kinesis 데이터 스트림에만 지원됩니다.
- MySQL 소스에서 마이그레이션하는 경우 이전 이미지 데이터에는 CLOB 및 BLOB 데이터 형식이 포함되지 않습니다. 자세한 내용은 [Kinesis 데이터 스트림 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기](#) 단원을 참조하십시오.
- AWS DMS 는 16자리를 초과하는 BigInt 데이터 형식의 값 마이그레이션을 지원하지 않습니다. 이 제한을 해결하기 위해 다음 변환 규칙을 사용하여 BigInt 열을 문자열로 변환할 수 있습니다. 변환 규칙에 대한 자세한 내용은 [변환 규칙 및 작업](#) 섹션을 참조하세요.

```

{
  "rule-type": "transformation",
  "rule-id": "id",
  "rule-name": "name",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "valid object-mapping rule action",
    "table-name": "",
    "column-name": ""
  },
  "rule-action": "change-data-type",
  "data-type": {

```

```

        "type": "string",
        "length": 20
    }
}

```

- 단일 트랜잭션 내의 여러 DML 작업이 소스 데이터베이스의 대형 객체(LOB) 열을 수정하면 대상 데이터베이스는 해당 트랜잭션의 마지막 작업에서 최종 LOB 값만 유지합니다. 동일한 트랜잭션의 이전 작업에서 설정한 중간 LOB 값을 덮어쓰면 잠재적인 데이터 손실 또는 불일치가 발생할 수 있습니다. 이 동작은 복제 중에 LOB 데이터가 처리되는 방식으로 인해 발생합니다.
- AWS DMS 는 Kinesis를 대상 엔드포인트로 사용할 때 임베디드 '\0' 문자가 포함된 소스 데이터를 지원하지 않습니다. 포함된 '\0' 문자가 포함된 데이터는 첫 번째 '\0' 문자에서 잘립니다.

객체 매핑을 사용하여 데이터를 Kinesis 데이터 스트림으로 마이그레이션

AWS DMS 는 테이블 매핑 규칙을 사용하여 소스의 데이터를 대상 Kinesis 데이터 스트림으로 매핑합니다. 데이터를 대상 스트림에 매핑하기 위해 객체 매핑이라는 테이블 매핑 규칙 유형을 사용합니다. 객체 매핑을 사용하여 소스의 데이터 레코드를 Kinesis 데이터 스트림에 게시된 데이터 레코드로 매핑하는 방법을 지정합니다.

Kinesis 데이터 스트림에는 파티션 키 외에 별다른 사전 설정 구조가 없습니다. 객체 매핑 규칙에서 데이터 레코드의 `partition-key-type`에 사용할 수 있는 값은 `schema-table`, `transaction-id`, `primary-key`, `constant` 및 `attribute-name`입니다.

객체 매핑 규칙을 만들려면 `rule-type`을 `object-mapping`으로 지정합니다. 이 규칙은 사용할 객체 매핑의 유형을 지정합니다.

이 규칙의 구조는 다음과 같습니다.

```

{
  "rules": [
    {
      "rule-type": "object-mapping",
      "rule-id": "id",
      "rule-name": "name",
      "rule-action": "valid object-mapping rule action",
      "object-locator": {
        "schema-name": "case-sensitive schema name",
        "table-name": ""
      }
    }
  ]
}

```

}

AWS DMS 는 현재 rule-action 파라미터 `map-record-to-document`에 유효한 유일한 값으로 `map-record-to-record` 및를 지원합니다. 이러한 설정은 `exclude-columns` 속성 목록의 일부로 제외되지 않은 값에 영향을 줍니다. `map-record-to-record` 및 `map-record-to-document` 값은 기본적으로 이러한 레코드를 AWS DMS 처리하는 방법을 지정합니다. 이러한 값은 어떤 식으로든 속성 매핑에 영향을 미치지 않습니다.

관계형 데이터베이스에서 Kinesis 데이터 스트림으로 마이그레이션할 때 `map-record-to-record`를 사용합니다. 이러한 규칙 유형은 관계형 데이터베이스의 `taskResourceId.schemaName.tableName` 값을 Kinesis 데이터 스트림의 파티션 키로 사용하며 소스 데이터베이스의 각 열마다 속성을 하나씩 생성합니다.

`map-record-to-record`를 사용하는 경우 다음 사항에 유의하세요.

- 이 설정은 `exclude-columns` 목록에서 제외된 열에만 영향을 줍니다.
- 이러한 모든 열에 대해서는 대상 주제에 해당 속성을 AWS DMS 생성합니다.
- AWS DMS 는 소스 열이 속성 매핑에 사용되는지 여부에 관계없이 해당 속성을 생성합니다.

속성 이름 “_doc”를 사용하여 소스 열을 적절한 대상 스트림의 단일 플랫폼 문서에 넣는 데 `map-record-to-document`를 사용합니다. AWS DMS 는 “_doc”라는 소스의 단일 플랫폼 맵에 데이터를 배치합니다. 이 배치는 `exclude-columns` 속성 목록에 나열되지 않은 소스 테이블의 모든 열에 적용됩니다.

`map-record-to-record`를 이해하는 한 가지 방법은 작업 중일 때 관찰하는 것입니다. 이 예에서는 다음 구조와 데이터를 사용하여 관계형 데이터베이스 테이블 행에서 시작한다고 가정합니다.

FirstName	LastName	StoreId	HomeAddress	HomePhone	WorkAddress	WorkPhone	DateofBirth
Randy	Marsh	5	221B Baker Street	1234567890	31 Spooner Street, Quahog	9876543210	1988/02/29

이 정보를 Test라는 스키마에서 Kinesis 데이터 스트림으로 마이그레이션하려면 데이터를 대상 스트림으로 매핑하는 규칙을 생성해야 합니다. 다음 규칙은 그 매핑 과정을 보여 줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      }
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "2",
      "rule-name": "DefaultMapToKinesis",
      "rule-action": "map-record-to-record",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Customers"
      }
    }
  ]
}
```

다음은 Kinesis 데이터 스트림에 생성되는 레코드 형식을 예시합니다.

- StreamName: XXX
- PartitionKey: Test.Customers //schmaName.tableName
- 날짜: //다음의 JSON 메시지

```
{
  "FirstName": "Randy",
  "LastName": "Marsh",
  "StoreId": "5",
  "HomeAddress": "221B Baker Street",
  "HomePhone": "1234567890",
  "WorkAddress": "31 Spooner Street, Quahog",
  "WorkPhone": "9876543210",
  "DateOfBirth": "02/29/1988"
}
```


동일한 규칙을 사용하지만 rule-action 파라미터를 map-record-to-document로 변경하거나 특정 열을 제외한다고 가정해 보겠습니다. 다음 규칙은 그 매핑 과정을 보여 줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      }
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "2",
      "rule-name": "DefaultMapToKinesis",
      "rule-action": "map-record-to-document",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Customers"
      },
      "mapping-parameters": {
        "exclude-columns": [
          "homeaddress",
          "homephone",
          "workaddress",
          "workphone"
        ]
      }
    }
  ]
}
```

이 경우, exclude-columns 파라미터에 나열되어 있지 않은 열인 FirstName, LastName, StoreId 및 DateOfBirth는 _doc로 매핑됩니다. 다음은 생성되는 레코드 형식을 보여줍니다.

```
{
  "data":{
    "_doc":{
      "FirstName": "Randy",
      "LastName": "Marsh",
      "StoreId": "5",
      "DateOfBirth": "02/29/1988"
    }
  }
}
```

속성 매핑으로 날짜 재구성

속성 맵을 사용하여 날짜를 Kinesis 데이터 스트림으로 마이그레이션하는 동안 날짜를 재구성할 수 있습니다. 예를 들어 소스의 필드 몇 개를 대상의 단일 필드로 묶어야 하는 경우도 있을 것입니다. 다음의 속성 맵은 날짜를 재구성하는 방법을 보여줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      }
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "2",
      "rule-name": "TransformToKinesis",
      "rule-action": "map-record-to-record",
      "target-table-name": "CustomerData",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Customers"
      },
      "mapping-parameters": {
        "partition-key-type": "attribute-name",
```

```

    "partition-key-name": "CustomerName",
    "exclude-columns": [
        "firstname",
        "lastname",
        "homeaddress",
        "homephone",
        "workaddress",
        "workphone"
    ],
    "attribute-mappings": [
        {
            "target-attribute-name": "CustomerName",
            "attribute-type": "scalar",
            "attribute-sub-type": "string",
            "value": "${lastname}, ${firstname}"
        },
        {
            "target-attribute-name": "ContactDetails",
            "attribute-type": "document",
            "attribute-sub-type": "json",
            "value": {
                "Home": {
                    "Address": "${homeaddress}",
                    "Phone": "${homephone}"
                },
                "Work": {
                    "Address": "${workaddress}",
                    "Phone": "${workphone}"
                }
            }
        }
    ]
}

```

에 상수 값을 설정하려면 `partition-key` 지정하고 파티션 값을 로 `"partition-key-type: constant"` 설정합니다. 예를 들어 이 방법을 통해 모든 데이터를 단일 샤드에 저장할 수 있습니다. 다음 매핑은 이러한 접근 방식을 보여줍니다.

```

{
  "rules": [

```

```
{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
    "schema-name": "Test",
    "table-name": "%"
  },
  "rule-action": "include"
},
{
  "rule-type": "object-mapping",
  "rule-id": "2",
  "rule-name": "TransformToKinesis",
  "rule-action": "map-record-to-document",
  "object-locator": {
    "schema-name": "Test",
    "table-name": "Customer"
  },
  "mapping-parameters": {
    "partition-key-type": "constant",
    "exclude-columns": [
      "FirstName",
      "LastName",
      "HomeAddress",
      "HomePhone",
      "WorkAddress",
      "WorkPhone"
    ],
    "attribute-mappings": [
      {
        "target-attribute-name": "CustomerName",
        "attribute-type": "scalar",
        "attribute-sub-type": "string",
        "value": "${FirstName},${LastName}"
      },
      {
        "target-attribute-name": "ContactDetails",
        "attribute-type": "scalar",
        "attribute-sub-type": "string",
        "value": {
          "Home": {
            "Address": "${HomeAddress}"
          }
        }
      }
    ]
  }
}
```

```

        "Phone": "${HomePhone}"
    },
    "Work": {
        "Address": "${WorkAddress}",
        "Phone": "${WorkPhone}"
    }
},
{
    "target-attribute-name": "DateOfBirth",
    "attribute-type": "scalar",
    "attribute-sub-type": "string",
    "value": "${DateOfBirth}"
}
]
}
]
}
}

```

Note

특정 테이블을 위한 제어 레코드에 대한 partition-key 값은 TaskId.SchemaName.TableName입니다. 특정 작업을 위한 제어 레코드에 대한 partition-key 값은 해당 레코드의 TaskId입니다. partition-key 값을 객체 매핑에 지정해도 제어 레코드에 대한 partition-key에는 영향이 없습니다.

테이블 매핑 규칙 attribute-name에서 이 로 partition-key-type 설정된 경우 소스 테이블의 열 또는 매핑에 정의된 사용자 지정 열을 참조 partition-key-name 해야 하는를 지정해야 합니다. 또한 소스 열이 대상 Kinesis Stream에 매핑되는 방식을 정의하려면 제공해야 attribute-mappings 합니다.

Kinesis Data Streams에 대한 메시지 형식

JSON 출력은 단지 키-값 페어의 목록입니다. JSON_UNFORMATTED 메시지 형식은 줄 바꿈 구분 기호가 있는 한 줄 JSON 문자열입니다.

AWS DMS 는 Kinesis Data Streams의 데이터를 더 쉽게 사용할 수 있도록 다음과 같은 예약 필드를 제공합니다.

RecordType

레코드 유형은 데이터나 제어일 수 있습니다. 데이터 레코드는 소스 내의 실제 행을 나타냅니다. 제어 레코드는 스트림 내의 중요 이벤트를 나타냅니다(예: 작업의 재시작).

연산

데이터 레코드의 경우, 작업은 load, insert, update 또는 delete일 수 있습니다.

데이터 레코드의 경우, 작업은 create-table, rename-table, drop-table, change-columns, add-column, drop-column, rename-column 또는 column-type-change일 수 있습니다.

SchemaName

레코드에 대한 원본 스키마입니다. 제어 레코드의 경우, 이 필드는 비워둘 수 있습니다.

TableName

레코드에 대한 소스 테이블입니다. 제어 레코드의 경우, 이 필드는 비워둘 수 있습니다.

타임스탬프

JSON 메시지가 구성된 경우의 타임스탬프입니다. 이 필드는 ISO 8601 형식으로 구성되었습니다.

Apache Kafka를 대상으로 사용 AWS Database Migration Service

AWS DMS 를 사용하여 데이터를 Apache Kafka 클러스터로 마이그레이션할 수 있습니다. Apache Kafka는 분산 스트리밍 플랫폼입니다. Apache Kafka를 사용하여 스트리밍 데이터의 실시간 수집 및 처리를 수행할 수 있습니다.

AWS 는 AWS DMS 대상으로 사용할 Amazon Managed Streaming for Apache Kafka(Amazon MSK)도 제공합니다. Amazon MSK는 Apache Kafka 인스턴스의 구현 및 관리를 단순화하는 완전 관리형 Apache Kafka 스트리밍 서비스입니다. 오픈 소스 Apache Kafka 버전에서 작동하며 모든 Apache Kafka 인스턴스와 동일한 AWS DMS 대상으로 Amazon MSK 인스턴스에 액세스합니다. 자세한 내용은 Amazon Managed Streaming for Apache Kafka 개발자 안내서의 [Amazon MSK란 무엇인가요?](#)를 참조하세요.

Kafka 클러스터는 여러 파티션으로 구분된 주제라는 범주에 레코드 스트림을 저장합니다. 파티션은 주제에서 고유하게 식별되는 데이터 레코드(메시지) 시퀀스입니다. 파티션은 주제 레코드의 병렬 처리가 가능하도록 한 클러스터의 여러 브로커로 분산될 수 있습니다. Apache Kafka의 주제 및 파티션과 분산에 관한 자세한 내용은 [주제 및 로그](#)와 [분산](#)을 참조하세요.

Kafka 클러스터는 Amazon MSK 인스턴스, Amazon EC2 인스턴스에서 실행되는 클러스터 또는 온프레미스 클러스터일 수 있습니다. Amazon MSK 인스턴스 또는 Amazon EC2 인스턴스의 클러스터는 동일한 VPC에 있을 수도 있고 다른 VPC에 있을 수도 있습니다. 클러스터가 온프레미스인 경우 복제 인스턴스용 자체 온프레미스 이름 서버를 사용하여 클러스터의 호스트 이름을 확인할 수 있습니다. 복제 인스턴스에서 이름 서버를 설정하는 방법에 관한 자세한 내용은 [자체 온프레미스 이름 서버 사용](#) 섹션을 참조하세요. 네트워크 설정에 관한 자세한 내용은 [복제 인스턴스용으로 네트워크 설정](#) 섹션을 참조하세요.

Amazon MSK 클러스터를 사용할 때는 해당 보안 그룹이 복제 인스턴스에서의 액세스를 허용하는지 확인하세요. Amazon MSK 클러스터의 보안 그룹 변경에 관한 자세한 내용은 [Amazon MSK 클러스터의 보안 그룹 변경](#)을 참조하세요.

AWS Database Migration Service 는 JSON을 사용하여 Kafka 주제에 레코드를 게시합니다. 변환 중 AWS DMS 는 각 레코드를 소스 데이터베이스로부터 JSON 형식의 속성-값 페어로 직렬화합니다.

지원되는 데이터 소스에서 대상 Kafka 클러스터로 데이터를 마이그레이션하려면 객체 매핑을 사용합니다. 객체 매핑을 통해 데이터 레코드를 대상 주제에 구조화하는 방법을 결정합니다. 또한 각 테이블에 대한 파티션 키를 정의합니다. Apache Kafka가 이러한 테이블을 사용해 데이터를 파티션으로 그룹화합니다.

현재는 작업당 단일 주제를 AWS DMS 지원합니다. 테이블이 여러 개 있는 단일 작업의 경우 모든 메시지가 단일 주제로 이동합니다. 각 메시지에는 대상 스키마와 테이블을 식별하는 메타데이터 섹션이 포함되어 있습니다. AWS DMS 버전 3.4.6 이상은 객체 매핑을 사용한 다중 주제 복제를 지원합니다. 자세한 내용은 [객체 매핑을 사용한 멀티주제 복제](#) 단원을 참조하십시오.

Apache Kafka 엔드포인트 설정

AWS DMS 콘솔의 엔드포인트 설정 또는 CLI의 --kafka-settings 옵션을 통해 연결 세부 정보를 지정할 수 있습니다. 각 설정의 요구 사항은 다음과 같습니다.

- **Broker** – Kafka 클러스터에 있는 하나 이상의 브로커 위치를 쉼표로 구분된 각 **broker-hostname:port**의 목록 형식으로 지정합니다. 예를 들면, "ec2-12-345-678-901.compute-1.amazonaws.com:2345,ec2-10-987-654-321.compute-1.amazonaws.com:2345"입니다. 이 설정은 클러스터 내 일부 또는 모든 브로커의 위치를 지정할 수 있습니다. 모든 클러스터 브로커는 주제로 마이그레이션된 데이터 레코드의 분할을 처리하기 위해 통신합니다.
- **Topic** – (선택 사항) 최대 길이 255개 문자 및 기호로 주제 이름을 지정합니다. 마침표(.), 밑줄(_) 및 빼기(-)를 사용할 수 있습니다. 마침표(.) 또는 밑줄(_)이 있는 주제 이름이 내부 데이터 구조에서 충돌할 수 있습니다. 주제 이름에 이러한 기호 중 하나만 사용하고 둘 다 사용하지는 마세요. 주제 이름을 지정하지 않으면 마이그레이션 주제 "kafka-default-topic"로 AWS DMS 사용합니다.

Note

지정한 마이그레이션 주제 또는 기본 주제를 AWS DMS 생성하려면 Kafka 클러스터 구성의 `auto.create.topics.enable = true` 일부로 설정합니다. 자세한 내용은 [Apache Kafka를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service](#) 섹션을 참조하세요.

- `MessageFormat` – 엔드포인트에 생성된 레코드의 출력 형식입니다. 메시지 형식은 JSON(기본값) 또는 JSON_UNFORMATTED(탭이 없는 한 줄)입니다.
- `MessageMaxBytes` – 엔드포인트에 생성된 레코드의 최대 크기(바이트)입니다. 기본값은 1,000,000입니다.

Note

AWS CLI/SDK를 사용하여 기본값이 아닌 값으로 `MessageMaxBytes`만 변경할 수 있습니다. 예를 들어 기존 Kafka 엔드포인트를 수정하고 `MessageMaxBytes`를 변경하려면 다음 명령을 사용합니다.

```
aws dms modify-endpoint --endpoint-arn your-endpoint
--kafka-settings Broker="broker1-server:broker1-port,broker2-server:broker2-port,...",
Topic=topic-name,MessageMaxBytes=integer-of-max-message-size-in-bytes
```

- `IncludeTransactionDetails` – 소스 데이터베이스의 자세한 트랜잭션 정보를 제공합니다. 이 정보에는 커밋 타임스탬프, 로그 위치 및 `transaction_id`, `previous_transaction_id`, `transaction_record_id`(트랜잭션 내의 레코드 오프셋)에 대한 값이 포함됩니다. 기본값은 `false`입니다.
- `IncludePartitionValue` – 파티션 유형이 `schema-table-type`이 아닌 경우 Kafka 메시지 출력에 파티션 값을 표시합니다. 기본값은 `false`입니다.
- `PartitionIncludeSchemaTable` – 파티션 유형이 `primary-key-type`인 경우 스키마 및 테이블 이름을 파티션 값에 접두사로 지정합니다. 이렇게 하면 Kafka 파티션 간의 데이터 분산이 증가합니다. 예를 들어 SysBench 스키마에 수천 개의 테이블이 있고 각 테이블에 프라이머리 키의 범위가 제한되어 있다고 가정하겠습니다. 이 경우 동일한 프라이머리 키가 수천 개의 테이블에서 동일한 파티션으로 전송되어 스로틀링이 발생합니다. 기본값은 `false`입니다.

- **IncludeTableAlterOperations** – 제어 데이터의 테이블을 변경하는 데이터 정의 언어(DDL) 작업(예: rename-table, drop-table, add-column, drop-column 및 rename-column)을 포함합니다. 기본값은 false입니다.
- **IncludeControlDetails** – Kafka 메시지 출력에서 테이블 정의, 열 정의, 테이블 및 열 변경 사항에 관한 자세한 제어 정보를 표시합니다. 기본값은 false입니다.
- **IncludeNullAndEmpty** – NULL 및 빈 열을 대상에 포함합니다. 기본값은 false입니다.
- **SecurityProtocol** – Transport Layer Security(TLS)을 사용하여 Kafka 대상 엔드포인트에 대한 보안 연결을 설정합니다. 옵션에는 ssl-authentication, ssl-encryption 및 sasl-ssl이 포함됩니다. sasl-ssl을 사용하려면 SaslUsername과 SaslPassword가 필요합니다.
- **SslEndpointIdentificationAlgorithm** – 인증서에 대한 호스트 이름 확인을 설정합니다. 이 설정은 AWS DMS 버전 3.5.1 이상에서 지원됩니다. 옵션에는 다음 사항이 포함됩니다.
 - NONE: 클라이언트 연결에서 브로커의 호스트 이름 확인을 비활성화합니다.
 - HTTPS: 클라이언트 연결에서 브로커의 호스트 이름 확인을 활성화합니다.
- **useLargeIntegerValue** - 3.5.4 AWS DMS 버전에서 사용할 수 있는 두 배로 묶는 대신 최대 18 자리 int를 사용합니다. 기본값은 false입니다.

설정을 사용하여 전송 속도를 높일 수 있습니다. 이를 위해 AWS DMS 는 Apache Kafka 대상 클러스터에 대한 멀티스레드 전체 로드를 지원합니다. AWS DMS 는 다음을 포함하는 작업 설정으로 이 멀티스레딩을 지원합니다.

- **MaxFullLoadSubTasks** -이 옵션을 사용하여 병렬로 로드할 최대 소스 테이블 수를 나타냅니다. 전용 하위 작업을 사용하여 각 테이블을 해당 Kafka 대상 테이블에 AWS DMS 로드합니다. 기본값은 8이며 최대값은 49입니다.
- **ParallelLoadThreads** -이 옵션을 사용하여 각 테이블을 Kafka 대상 테이블에 로드하는 데 AWS DMS 사용하는 스레드 수를 지정합니다. Apache Kafka 대상의 최대값은 32입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.
- **ParallelLoadBufferSize** – 이 옵션을 사용하여 병렬 로드 스레드에서 데이터를 Kafka 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 ParallelLoadThreads와 함께 사용하세요. ParallelLoadBufferSize는 둘 이상의 스레드가 있는 경우에만 유효합니다.
- **ParallelLoadQueuesPerThread** – 각 동시 스레드가 액세스하는 대기열 수를 지정하여 대기열에서 데이터 레코드를 가져오고 대상에 대한 배치 로드를 생성하려면 이 옵션을 사용합니다. 기본값은 1입니다. 최대 한도는 512입니다.

병렬 스레드 및 대량 작업에 대한 작업 설정을 조정하여 Kafka 엔드포인트의 변경 데이터 캡처(CDC) 성능을 개선할 수 있습니다. 이렇게 하려면 `ParallelApply*` 작업 설정을 사용하여 동시 스레드 수, 스레드당 대기열 및 버퍼에 저장할 레코드 수를 지정해야 합니다. 예를 들어 CDC 로드를 수행하고 128개의 스레드를 병렬로 적용한다고 가정하겠습니다. 또한 버퍼당 50개 레코드가 저장된 스레드당 64개 대기열에 액세스하려고 합니다.

CDC 성능을 높이기 위해서는 다음 작업 설정을 AWS DMS 지원합니다.

- `ParallelApplyThreads` - CDC 로드 중에가 데이터 레코드를 Kafka 대상 엔드포인트로 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- `ParallelApplyBufferSize` - CDC 로드 중에 Kafka 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.
- `ParallelApplyQueuesPerThread` - 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Kafka 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다. 기본값은 1입니다. 최대 한도는 512입니다.

`ParallelApply*` 작업 설정을 사용할 때 `partition-key-type` 기본값은 테이블의 `schema-name.table-name`가 아니라 `primary-key`입니다.

Transport Layer Security(TLS)을 사용하여 Kafka에 연결

Kafka 클러스터는 Transport Layer Security(TLS)을 사용한 보안 연결만 허용합니다. DMS를 사용하면 다음 세 가지 보안 프로토콜 옵션 중 하나를 사용하여 Kafka 엔드포인트 연결을 보호할 수 있습니다.

SSL 암호화(**server-encryption**)

클라이언트는 서버의 인증서를 통해 서버 ID를 확인합니다. 그러면 서버와 클라이언트 간에 암호화된 연결이 이루어집니다.

SSL 인증(**mutual-authentication**)

서버와 클라이언트는 자체 인증서를 통해 서로 ID를 확인합니다. 그러면 서버와 클라이언트 간에 암호화된 연결이 이루어집니다.

SASL-SSL(**mutual-authentication**)

SASL(Simple Authentication and Security Layer) 메서드는 클라이언트 인증서를 사용자 이름 및 암호로 대체하여 클라이언트 ID를 검증합니다. 특히 서버가 클라이언트의 ID를 검증할 수 있도록 서

버가 등록된 사용자 이름과 암호를 제공합니다. 그러면 서버와 클라이언트 간에 암호화된 연결이 이루어집니다.

Important

Apache Kafka와 Amazon MSK는 확인된 인증서를 수락합니다. 이는 Kafka와 Amazon MSK에서 해결해야 하는 알려진 제한 사항입니다. 자세한 내용은 [Apache Kafka 문제, KAFKA-3700](#) 섹션을 참조하세요.

Amazon MSK를 사용하는 경우 이 알려진 제한에 대한 해결 방법으로 액세스 제어 목록(ACL)을 사용하는 것이 좋습니다. ACL 사용에 관한 자세한 내용은 Amazon Managed Streaming for Apache Kafka 개발자 안내서의 [Apache Kafka ACL](#) 섹션을 참조하세요.

자체 관리형 Kafka 클러스터를 사용하는 경우, 클러스터 구성에 관한 자세한 내용은 [2018년 10월 21일자 설명](#)을 참조하세요.

Amazon MSK 또는 자체 관리형 Kafka 클러스터에서 SSL 암호화 사용

Amazon MSK 또는 자체 관리형 Kafka 클러스터에 대한 엔드포인트 연결을 보호하는 데 SSL 암호화를 사용할 수 있습니다. SSL 암호화 인증 방법을 사용하면 클라이언트가 서버의 인증서를 통해 서버의 ID를 검증합니다. 그러면 서버와 클라이언트 간에 암호화된 연결이 이루어집니다.

SSL 암호화를 사용하여 Amazon MSK에 연결하려면

- 대상 Kafka 엔드포인트를 생성할 때 `ssl-encryption` 옵션을 사용하여 보안 프로토콜 엔드포인트 설정(`SecurityProtocol`)을 설정합니다.

다음 JSON 예제는 보안 프로토콜을 SSL 암호화로 설정합니다.

```
"KafkaSettings": {
  "SecurityProtocol": "ssl-encryption",
}
```

자체 관리형 Kafka 클러스터에 SSL 암호화를 사용하려면

- 온프레미스 Kafka 클러스터에서 프라이빗 인증 기관(CA)를 사용하는 경우 프라이빗 CA 인증서를 업로드하고 Amazon 리소스 이름(ARN)을 받으세요.

2. 대상 Kafka 엔드포인트를 생성할 때 `ssl-encryption` 옵션을 사용하여 보안 프로토콜 엔드포인트 설정(`SecurityProtocol`)을 설정합니다. 다음 JSON 예제는 보안 프로토콜을 `ssl-encryption`으로 설정합니다.

```
"KafkaSettings": {
  "SecurityProtocol": "ssl-encryption",
}
```

3. 프라이빗 CA를 사용하는 경우 위의 첫 번째 단계에서 얻은 ARN에 `SslCaCertificateArn`을 설정하세요.

SSL 인증 사용

Amazon MSK 또는 자체 관리형 Kafka 클러스터에 대한 엔드포인트 연결을 보호하는 데 SSL 인증을 사용할 수 있습니다.

Amazon MSK에 연결할 때 SSL 인증을 사용한 클라이언트 인증 및 암호화를 활성화하려면 다음을 수행하세요.

- Kafka용 프라이빗 키와 퍼블릭 인증서를 준비합니다.
- 인증서를 DMS Certificate Manager에 업로드합니다.
- Kafka 엔드포인트 설정에 지정된 해당 인증서 ARN을 사용하여 Kafka 대상 엔드포인트를 생성합니다.

Amazon MSK용 프라이빗 키와 퍼블릭 인증서를 준비하려면

1. Amazon Managed Streaming for Apache Kafka 개발자 안내서의 [클라이언트 인증](#) 섹션에 있는 1~9단계에 설명된 대로 EC2 인스턴스를 생성하고 클라이언트가 인증을 사용하도록 설정합니다.

이러한 단계를 완료하면 인증서-ARN(ACM에 저장된 공개 인증서 ARN)과 `kafka.client.keystore.jks` 파일에 포함된 프라이빗 키를 갖게 됩니다.

2. 다음 명령을 사용하여 퍼블릭 인증서를 가져오고 인증서를 `signed-certificate-from-acm.pem` 파일에 복사합니다.

```
aws acm-pca get-certificate --certificate-authority-arn Private_CA_ARN --
certificate-arn Certificate_ARN
```

이 명령은 다음 예와 유사한 정보를 반환합니다.

```
{"Certificate": "123", "CertificateChain": "456"}
```

그런 다음, "123"과 동일한 데이터를 signed-certificate-from-acm.pem 파일에 복사합니다.

- 다음 예제와 같이 kafka.client.keystore.jks to keystore.p12에서 msk-rsa 키를 가져와서 프라이빗 키를 받습니다.

```
keytool -importkeystore \
-srckeystore kafka.client.keystore.jks \
-destkeystore keystore.p12 \
-deststoretype PKCS12 \
-srcalias msk-rsa-client \
-deststorepass test1234 \
-destkeypass test1234
```

- 다음 명령을 사용하여 keystore.p12를 .pem 형식으로 내보냅니다.

```
Openssl pkcs12 -in keystore.p12 -out encrypted-private-client-key.pem -nocerts
```

Enter PEM 암호문 메시지가 나타나고 인증서를 암호화하는 데 적용되는 키를 식별합니다.

- .pem 파일에서 백 속성 및 키 속성을 제거하여 첫 번째 줄이 다음 문자열로 시작되도록 합니다.

```
---BEGIN ENCRYPTED PRIVATE KEY---
```

퍼블릭 인증서와 프라이빗 키를 DMS 인증서 관리자에 업로드하고 Amazon MSK에 대한 연결을 테스트하려면

- 다음 명령을 사용하여 DMS 인증서 관리자에 업로드합니다.

```
aws dms import-certificate --certificate-identifier signed-cert --certificate-pem
file://path to signed cert
aws dms import-certificate --certificate-identifier private-key --certificate-pem
file://path to private key
```

2. Amazon MSK 대상 엔드포인트를 생성하고 연결을 테스트하여 TLS 인증이 작동하는지 확인합니다.

```
aws dms create-endpoint --endpoint-identifier $endpoint-identifier --engine-name
kafka --endpoint-type target --kafka-settings
'{"Broker": "b-0.kafka260.aaaaa1.a99.kafka.us-east-1.amazonaws.com:0000",
"SecurityProtocol":"ssl-authentication",
"SslClientCertificateArn": "arn:aws:dms:us-east-1:012346789012:cert:",
"SslClientKeyArn": "arn:aws:dms:us-
east-1:0123456789012:cert:", "SslClientKeyPassword":"test1234"}'
aws dms test-connection -replication-instance-arn=$rep_inst_arn --endpoint-arn=
$kafka_tar_arn_msk
```

Important

자체 관리형 Kafka 클러스터에 대한 연결을 보호하는 데 SSL 인증을 사용할 수 있습니다. 경우에 따라 온프레미스 Kafka 클러스터에서 프라이빗 인증 기관(CA)을 사용할 수 있습니다. 그렇다면 CA 체인, 퍼블릭 인증서, 프라이빗 키를 DMS 인증서 관리자에게 업로드하세요. 그런 다음 온프레미스 Kafka 대상 엔드포인트를 생성할 때 엔드포인트 설정에서 해당 Amazon 리소스 이름(ARN)을 사용합니다.

자체 관리형 Kafka 클러스터에 사용할 프라이빗 키와 서명된 인증서를 준비하려면

1. 다음 예제에서 보는 것처럼 키 페어를 생성합니다.

```
keytool -genkey -keystore kafka.server.keystore.jks -validity 300 -storepass your-keystore-password
-keypass your-key-passphrase -dname "CN=your-cn-name"
-alias alias-of-key-pair -storetype pkcs12 -keyalg RSA
```

2. 인증서 서명 요청(CSR)을 생성합니다.

```
keytool -keystore kafka.server.keystore.jks -certreq -file server-cert-sign-
request-rsa -alias on-premise-rsa -storepass your-key-store-password
-keypass your-key-password
```

- 클러스터 트러스트 스토어의 CA를 사용하여 CSR에 서명합니다. CA가 없는 경우 자체 프라이빗 CA를 생성할 수 있습니다.

```
openssl req -new -x509 -keyout ca-key -out ca-cert -days validate-days
```

- ca-cert를 서버 트러스트 스토어 및 키 스토어로 가져옵니다. 신뢰 스토어가 없는 경우에는 다음 명령을 사용하여 트러스트 스토어를 생성하고 ca-cert 를 해당 스토어로 가져옵니다.

```
keytool -keystore kafka.server.truststore.jks -alias CARoot -import -file ca-cert
keytool -keystore kafka.server.keystore.jks -alias CARoot -import -file ca-cert
```

- 인증서에 서명합니다.

```
openssl x509 -req -CA ca-cert -CAkey ca-key -in server-cert-sign-request-rsa -out
signed-server-certificate.pem
-days validate-days -CAcreateserial -passin pass:ca-password
```

- 서명된 인증서를 키 스토어로 가져옵니다.

```
keytool -keystore kafka.server.keystore.jks -import -file signed-certificate.pem -
alias on-premise-rsa -storepass your-keystore-password
-keypass your-key-password
```

- 다음 명령을 사용하여 kafka.server.keystore.jks에서 keystore.p12으로 on-premise-rsa 키를 가져옵니다.

```
keytool -importkeystore \
-srckeystore kafka.server.keystore.jks \
-destkeystore keystore.p12 \
-deststoretype PKCS12 \
-srcalias on-premise-rsa \
-deststorepass your-truststore-password \
-destkeypass your-key-password
```

- 다음 명령을 사용하여 keystore.p12를 .pem 형식으로 내보냅니다.

```
openssl pkcs12 -in keystore.p12 -out encrypted-private-server-key.pem -nocerts
```

9. encrypted-private-server-key.pem, signed-certificate.pem 및 ca-cert를 DMS 인증서 관리자에 업로드합니다.
10. 반환된 ARN을 사용하여 엔드포인트를 생성합니다.

```
aws dms create-endpoint --endpoint-identifier $endpoint-identifier --engine-name
kafka --endpoint-type target --kafka-settings
'{"Broker": "b-0.kafka260.aaaaa1.a99.kafka.us-east-1.amazonaws.com:9092",
  "SecurityProtocol": "ssl-authentication",
  "SslClientCertificateArn": "your-client-cert-arn", "SslClientKeyArn": "your-client-key-arn", "SslClientKeyPassword": "your-client-key-password",
  "SslCaCertificateArn": "your-ca-certificate-arn"}'
```

```
aws dms test-connection -replication-instance-arn=$rep_inst_arn --endpoint-arn=
$kafka_tar_arn_msk
```

SASL-SSL 인증을 사용하여 Amazon MSK에 연결

SASL(Simple Authentication and Security Layer) 메서드는 사용자 이름과 암호를 사용하여 클라이언트 ID를 검증하고 서버와 클라이언트 간에 암호화된 연결을 만듭니다.

SASL을 사용하려면 Amazon MSK 클러스터를 설정할 때 먼저 안전한 사용자 이름과 암호를 생성해야 합니다. Amazon MSK 클러스터의 보안 사용자 이름과 암호를 설정하는 방법에 관한 설명은 [Amazon Managed Streaming for Apache Kafka 개발자 안내서의 Amazon MSK 클러스터용 SASL/SCRAM 인증 설정](#)을 참조하세요.

그런 다음, Kafka 대상 엔드포인트를 생성할 때 sasl-ssl 옵션을 사용하여 보안 프로토콜 엔드포인트 설정(SecurityProtocol)을 설정합니다. SaslUsername 및 SaslPassword 옵션도 설정합니다. Amazon MSK 클러스터를 처음 설정할 때 다음 JSON 예제에서 보는 것처럼 이 옵션 설정이 보안 사용자 이름 및 암호와 일치하는지 확인하세요.

```
"KafkaSettings": {
  "SecurityProtocol": "sasl-ssl",
  "SaslUsername": "Amazon MSK cluster secure user name",
  "SaslPassword": "Amazon MSK cluster secure password"
```


}

Note

- 현재는 퍼블릭 CA 지원 SASL-SSL만 AWS DMS 지원합니다. DMS는 프라이빗 CA가 지원하는 자체 관리형 Kafka와 함께 사용할 수 있는 SASL-SSL을 지원하지 않습니다.
- SASL-SSL 인증의 경우는 기본적으로 SCRAM-SHA-512 메커니즘을 AWS DMS 지원합니다. AWS DMS 버전 3.5.0 이상은 일반 메커니즘도 지원합니다. Plain 메커니즘을 지원하려면 KafkaSettings API 데이터 형식의 SaslMechanism 파라미터를 PLAIN으로 설정하세요. 데이터 유형은 Kafka에서 PLAIN 지원되지만 Amazon Kafka(MSK)에서는 지원되지 않습니다.

Apache Kafka 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기

Kafka 같은 데이터 스트리밍 대상에 CDC 업데이트를 작성하는 경우 업데이트를 통한 변경 전에 소스 데이터베이스 행의 원래 값을 볼 수 있습니다. 이를 가능하게 하기 위해서는 소스 데이터베이스 엔진에서 제공하는 데이터를 기반으로 업데이트 이벤트의 이전 이미지를 AWS DMS 채웁니다.

소스 데이터베이스 엔진에 따라 이전 이미지에 대해 서로 다른 양의 정보를 제공합니다.

- Oracle은 열이 변경되는 경우에만 열에 대한 업데이트를 제공합니다.
- PostgreSQL은 프라이머리 키의 일부인 열에 대한 데이터(변경 여부에 관계 없음)만 제공합니다. 논리적 복제를 사용 중이고 원본 테이블에 대해 REPLICA IDENTITY FULL이 설정된 경우 WAL에 기록된 행의 전체 이전 정보 및 이후 정보를 얻을 수 있으며 여기에서 확인할 수 있습니다.
- MySQL은 일반적으로 모든 열에 대한 데이터(변경 여부에 관계 없음)를 제공합니다.

이전 이미징을 활성화하여 소스 데이터베이스의 원래 값을 AWS DMS 출력에 추가하려면 BeforeImageSettings 태스크 설정 또는 add-before-image-columns 파라미터를 사용합니다. 이 파라미터는 열 변환 규칙을 적용합니다.

BeforeImageSettings는 다음과 같이 소스 데이터베이스 시스템에서 수집된 값을 사용하여 모든 업데이트 작업에 새 JSON 속성을 추가합니다.

```
"BeforeImageSettings": {
  "EnableBeforeImage": boolean,
```

```

    "FieldName": string,
    "ColumnFilter": pk-only (default) / non-lob / all (but only one)
}

```

Note

전체 로드와 CDC 작업(기존 데이터 마이그레이션 및 지속적인 변경 사항 복제) 또는 CDC 전용 작업(데이터 변경 사항만 복제)에 BeforeImageSettings를 적용합니다. 전체 로드 전용 작업에는 BeforeImageSettings를 적용하지 마세요.

BeforeImageSettings 옵션의 경우, 다음이 적용됩니다.

- 이전 이미징을 활성화하려면 EnableBeforeImage 옵션을 true로 설정합니다. 기본값은 false입니다.
- FieldName 옵션을 사용하여 새 JSON 속성에 이름을 지정합니다. EnableBeforeImage가 true인 경우 FieldName은 필수이며 비워 둘 수 없습니다.
- ColumnFilter 옵션은 이전 이미징을 사용하여 추가할 열을 지정합니다. 테이블 프라이머리 키의 일부에 속하는 열만 추가하려면 기본값 pk-only를 사용합니다. LOB 유형이 아닌 열만 추가하려면 non-lob를 사용합니다. 이전 이미지 값이 있는 모든 열을 추가하려면 all을 사용합니다.

```

"BeforeImageSettings": {
  "EnableBeforeImage": true,
  "FieldName": "before-image",
  "ColumnFilter": "pk-only"
}

```

이전 이미지 변환 규칙 사용

작업 설정 대신 열 변환 규칙을 적용하는 add-before-image-columns 파라미터를 사용할 수 있습니다. 이 파라미터를 사용하면 Kafka 같은 데이터 스트리밍 대상에서 CDC 중에 이전 이미징을 활성화할 수 있습니다.

변환 규칙에 add-before-image-columns를 사용하면 이전 이미지 결과를 보다 세밀하게 제어할 수 있습니다. 변환 규칙을 통해 객체 로케이터를 사용하여 규칙에 대해 선택한 테이블을 제어할 수 있습니다. 또한 변환 규칙을 함께 연결하여 테이블마다 서로 다른 규칙을 적용할 수 있습니다. 그런 다음, 다른 규칙을 사용하여 생성된 열을 조작할 수 있습니다.

Note

동일한 작업 내에서 BeforeImageSettings 작업 설정과 함께 add-before-image-columns 파라미터를 사용해서는 안 됩니다. 단일 작업에는 이 파라미터 또는 설정 중 하나만 사용하고 둘 다 사용하지는 마세요.

해당 열에 대해 add-before-image-columns 파라미터가 있는 transformation 규칙 유형이 before-image-def 섹션을 제공해야 합니다. 다음은 그 한 예입니다.

```
{
  "rule-type": "transformation",
  ...
  "rule-target": "column",
  "rule-action": "add-before-image-columns",
  "before-image-def": {
    "column-filter": one-of (pk-only / non-lob / all),
    "column-prefix": string,
    "column-suffix": string,
  }
}
```

column-prefix의 값은 열 이름 앞에 추가되며, column-prefix의 기본값은 BI_입니다. column-suffix의 값은 열 이름 뒤에 추가되며, 기본값은 비어 있습니다. column-prefix와 column-suffix를 모두 빈 문자열로 설정하지 마세요.

column-filter에 대해 하나의 값을 선택합니다. 테이블 프라이머리 키의 일부인 열만 추가하려면 pk-only를 선택합니다. LOB 유형이 아닌 열만 추가하려면 non-lob를 선택합니다. 이전 이미지 값이 있는 모든 열을 추가하려면 all을 선택합니다.

이전 이미지 변환 규칙의 예

다음 예의 변환 규칙은 대상에서 BI_emp_no라는 새 열을 추가합니다. UPDATE employees SET emp_no = 3 WHERE emp_no = 1; 같은 문은 BI_emp_no 필드를 1로 채웁니다. Amazon S3 대상에 CDC 업데이트를 작성할 때 BI_emp_no 열을 통해 업데이트된 원래 행을 알 수 있습니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
```

```

    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-target": "column",
    "object-locator": {
      "schema-name": "%",
      "table-name": "employees"
    },
    "rule-action": "add-before-image-columns",
    "before-image-def": {
      "column-prefix": "BI_",
      "column-suffix": "",
      "column-filter": "pk-only"
    }
  }
]
}

```

add-before-image-columns 규칙 작업 사용에 관한 자세한 내용은 [변환 규칙 및 작업](#) 섹션을 참조하세요.

Apache Kafka를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

Apache Kafka를 대상으로 사용할 때 다음 제한 사항이 적용됩니다.

- AWS DMS Kafka 대상 엔드포인트는 Amazon Managed Streaming for Apache Kafka(Amazon MSK)에 대한 IAM 액세스 제어를 지원하지 않습니다.
- 전체 LOB 모드는 지원되지 않습니다.
- 예서 새 주제를 AWS DMS 자동으로 생성할 수 있는 속성을 사용하여 클러스터에 대한 Kafka 구성 파일을 지정합니다. `auto.create.topics.enable = true` 설정을 포함합니다. Amazon MSK 설정을 사용하는 경우 Kafka 클러스터를 만들 때 기본 구성을 지정한 다음 `auto.create.topics.enable` 설정을 `true`로 변경할 수 있습니다. 기본 구성 설정에 관한 자세한 내용은 Amazon Managed Streaming for Apache Kafka 개발자 안내서의 [기본 Amazon MSK 구](#)

성을 참조하세요. Amazon MSK를 사용하여 생성된 기존 Kafka 클러스터를 수정해야 하는 경우 다음 예제와 같이 AWS CLI 명령을 실행 `aws kafka create-configuration`하여 Kafka 구성을 업데이트합니다.

```
14:38:41 $ aws kafka create-configuration --name "kafka-configuration" --kafka-versions "2.2.1" --server-properties file://~/kafka_configuration
{
  "LatestRevision": {
    "Revision": 1,
    "CreationTime": "2019-09-06T14:39:37.708Z"
  },
  "CreationTime": "2019-09-06T14:39:37.708Z",
  "Name": "kafka-configuration",
  "Arn": "arn:aws:kafka:us-east-1:111122223333:configuration/kafka-configuration/7e008070-6a08-445f-9fe5-36ccf630ecfd-3"
}
```

여기서, `file://~/kafka_configuration`은 필요한 속성 설정으로 만든 구성 파일입니다.

Amazon EC2에 설치된 자체 Kafka 인스턴스를 사용하는 경우 인스턴스와 함께 제공된 옵션을 사용하여 새 주제를 AWS DMS 자동으로 생성할 수 있도록 `auto.create.topics.enable = true` 설정으로 Kafka 클러스터 구성을 수정합니다.

- AWS DMS 는 트랜잭션에 관계없이 각 업데이트를 소스 데이터베이스의 단일 레코드에 지정된 Kafka 주제의 하나의 데이터 레코드(메시지)로 게시합니다.
- AWS DMS 는 파티션 키에 대해 다음 네 가지 양식을 지원합니다.
 - `SchemaName.TableName`: 스키마와 테이블 이름의 조합입니다.
 - `${AttributeName}`: JSON 내 한 필드의 값 또는 소스 데이터베이스 내 테이블의 프라이머리 키입니다.
 - `transaction-id`: CDC 트랜잭션 ID입니다. 동일한 트랜잭션 내의 모든 레코드는 동일한 파티션으로 이동합니다.
 - `constant`: 테이블 또는 데이터에 관계없이 모든 레코드에 대해 고정된 리터럴 값입니다. 모든 레코드는 동일한 파티션 키 값 "상수"로 전송되어 모든 테이블에서 엄격한 전역 순서를 제공합니다.

```
{
  "rule-type": "object-mapping",
  "rule-id": "2",
  "rule-name": "TransactionIdPartitionKey",
  "rule-action": "map-record-to-document",
```

```

    "object-locator": {
      "schema-name": "onprem",
      "table-name": "it_system"
    },
    "mapping-parameters": {
      "partition-key-type": "transaction-id | constant | attribute-name | schema-
table"
    }
  }
}

```

- IncludeTransactionDetails 엔드포인트 설정은 소스 엔드포인트가 Oracle, SQL Server, PostgreSQL 또는 MySQL인 경우에만 지원됩니다. 다른 소스 엔드포인트 유형의 경우 트랜잭션 세부 정보가 포함되지 않습니다.
- BatchApply는 Kafka 엔드포인트에는 지원되지 않습니다. Kafka 대상에 대해 Batch Apply(예: BatchApplyEnabled 대상 메타데이터 작업 설정)를 사용하면 데이터 손실이 발생할 수 있습니다.
- AWS DMS 는 16자리를 초과하는 BigInt 데이터 형식의 값 마이그레이션을 지원하지 않습니다. 이 제한을 해결하기 위해 다음 변환 규칙을 사용하여 BigInt 열을 문자열로 변환할 수 있습니다. 변환 규칙에 대한 자세한 내용은 [변환 규칙 및 작업](#) 섹션을 참조하세요.

```

{
  "rule-type": "transformation",
  "rule-id": "id",
  "rule-name": "name",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "valid object-mapping rule action",
    "table-name": "",
    "column-name": ""
  },
  "rule-action": "change-data-type",
  "data-type": {
    "type": "string",
    "length": 20
  }
}

```

- AWS DMS Kafka 대상 엔드포인트는 Amazon MSK servless를 지원하지 않습니다.
- 매핑 규칙을 정의할 때 객체 매핑 규칙과 변환 규칙이 모두 있으면 지원되지 않습니다. 규칙을 하나만 설정해야 합니다.

- AWS DMS 는 최대 3.8의 Apache Kafka 버전에 대한 SASL 인증을 지원합니다. Kafka 4.0 이상을 사용하는 경우 SASL 인증 없이만 연결할 수 있습니다.
- AWS DMS 는 Kafka를 대상 엔드포인트로 사용할 때 임베디드 '\0' 문자가 포함된 소스 데이터를 지원하지 않습니다. 포함된 '\0' 문자가 포함된 데이터는 첫 번째 '\0' 문자에서 잘립니다.

객체 매핑을 사용하여 데이터를 Kafka 주제로 마이그레이션

AWS DMS 는 테이블 매핑 규칙을 사용하여 소스의 데이터를 대상 Kafka 주제로 매핑합니다. 데이터를 대상 주제에 매핑하려면 객체 매핑이라는 테이블 매핑 규칙 유형을 사용합니다. 객체 매핑을 사용하여 원본의 데이터 레코드를 Kafka 주제에 게시된 데이터 레코드로 매핑하는 방법을 지정합니다.

Kafka 주제에는 파티션 키 외에 별다른 사전 설정 구조가 없습니다.

Note

객체 매핑을 사용할 필요는 없습니다. 일반 테이블 매핑을 다양한 변환에 사용할 수 있습니다. 하지만 파티션 키 유형은 다음과 같은 기본 동작을 따릅니다.

- 프라이머리 키는 전체 로드의 파티션 키로 사용됩니다.
- parallel-apply 작업 설정을 사용하지 않는 경우, CDC의 파티션 키로 schema.table이 사용됩니다.
- parallel-apply 작업 설정을 사용하지 않는 경우, CDC의 파티션 키로 프라이머리 키가 사용됩니다.

object-mapping 규칙을 만들려면 rule-type을 object-mapping으로 지정합니다. 이 규칙은 사용할 객체 매핑의 유형을 지정합니다.

이 규칙의 구조는 다음과 같습니다.

```
{
  "rules": [
    {
      "rule-type": "object-mapping",
      "rule-id": "id",
      "rule-name": "name",
      "rule-action": "valid object-mapping rule action",
      "object-locator": {
        "schema-name": "case-sensitive schema name",
```

```

        "table-name": ""
    }
}
]
}

```

AWS DMS 는 현재 rule-action 파라미터 map-record-to-document에 유효한 유일한 값으로 map-record-to-record 및를 지원합니다. 이러한 설정은 exclude-columns 속성 목록의 일부로 제외되지 않은 값에 영향을 줍니다. map-record-to-record 및 map-record-to-document 값은 기본적으로 이러한 레코드를 AWS DMS 처리하는 방법을 지정합니다. 이러한 값은 어떤 식으로든 속성 매핑에 영향을 미치지 않습니다.

관계형 데이터베이스에서 Kafka 주제로 마이그레이션할 때 map-record-to-record를 사용합니다. 이러한 규칙 유형은 Kafka 주제의 파티션 키로 관계형 데이터베이스의 taskResourceId.schemaName.tableName 값을 사용하며 소스 데이터베이스의 각 열마다 속성을 하나씩 생성합니다.

map-record-to-record를 사용하는 경우 다음 사항에 유의하세요.

- 이 설정은 exclude-columns 목록에서 제외된 열에만 영향을 줍니다.
- 이러한 모든 열에 대해서는 대상 주제에 해당 속성을 AWS DMS 생성합니다.
- AWS DMS 는 소스 열이 속성 매핑에 사용되는지 여부에 관계없이 해당 속성을 생성합니다.

map-record-to-record를 이해하는 한 가지 방법은 작업 중일 때 관찰하는 것입니다. 이 예에서는 다음 구조와 데이터를 사용하여 관계형 데이터베이스 테이블 행에서 시작한다고 가정합니다.

FirstName	LastName	StoreId	HomeAddress	HomePhone	WorkAddress	WorkPhone	DateofBirth
Randy	Marsh	5	221B Baker Street	1234567890	31 Spooner Street, Quahog	9876543210	1988/02/29

이 정보를 Test라는 스키마에서 Kafka 주제로 마이그레이션하려면 데이터를 대상 주제로 매핑하는 규칙을 생성합니다. 다음 규칙은 그 매핑 과정을 보여 줍니다.

```

{

```



```

"rules": [
  {
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "rule-action": "include",
    "object-locator": {
      "schema-name": "Test",
      "table-name": "%"
    }
  },
  {
    "rule-type": "object-mapping",
    "rule-id": "2",
    "rule-name": "DefaultMapToKafka",
    "rule-action": "map-record-to-record",
    "object-locator": {
      "schema-name": "Test",
      "table-name": "Customers"
    }
  }
]
}

```

다음은 지정된 Kafka 주제와 파티션 키(이 경우 `taskResourceId.schemaName.tableName`)로 Kafka 대상 주제의 샘플 데이터를 사용하여 결과 레코드 형식을 보여줍니다.

```

{
  "FirstName": "Randy",
  "LastName": "Marsh",
  "StoreId": "5",
  "HomeAddress": "221B Baker Street",
  "HomePhone": "1234567890",
  "WorkAddress": "31 Spooner Street, Quahog",
  "WorkPhone": "9876543210",
  "DateOfBirth": "02/29/1988"
}

```

주제

- [속성 매핑으로 날짜 재구성](#)
- [객체 매핑을 사용한 멀티주제 복제](#)

- [Apache Kafka 메시지 형식](#)

속성 매핑으로 날짜 재구성

속성 맵을 사용하여 데이터를 Kafka 주제로 마이그레이션하는 동안 데이터를 재구성할 수 있습니다. 예를 들어 소스의 필드 몇 개를 대상의 단일 필드로 묶어야 하는 경우도 있을 것입니다. 다음의 속성 맵은 날짜를 재구성하는 방법을 보여줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      }
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "2",
      "rule-name": "TransformToKafka",
      "rule-action": "map-record-to-record",
      "target-table-name": "CustomerData",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Customers"
      },
      "mapping-parameters": {
        "partition-key-type": "attribute-name",
        "partition-key-name": "CustomerName",
        "exclude-columns": [
          "firstname",
          "lastname",
          "homeaddress",
          "homephone",
          "workaddress",
          "workphone"
        ],
        "attribute-mappings": [
          {
```

```
        "target-attribute-name": "CustomerName",
        "attribute-type": "scalar",
        "attribute-sub-type": "string",
        "value": "${lastname}, ${firstname}"
    },
    {
        "target-attribute-name": "ContactDetails",
        "attribute-type": "document",
        "attribute-sub-type": "json",
        "value": {
            "Home": {
                "Address": "${homeaddress}",
                "Phone": "${homephone}"
            },
            "Work": {
                "Address": "${workaddress}",
                "Phone": "${workphone}"
            }
        }
    }
]
}
]
```

에 상수 값을 설정하려면 partition-key 지정하고 파티션 값을 로 "partition-key-type: "constant" 설정합니다 constant. 예를 들어 이 방법을 통해 모든 데이터를 단일 파티션에 저장할 수 있습니다. 다음 매핑은 이러한 접근 방식을 보여줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {

```

```

    "rule-type": "object-mapping",
    "rule-id": "1",
    "rule-name": "TransformToKafka",
    "rule-action": "map-record-to-document",
    "object-locator": {
      "schema-name": "Test",
      "table-name": "Customer"
    },
    "mapping-parameters": {
      "partition-key-type": "constant",
      "exclude-columns": [
        "FirstName",
        "LastName",
        "HomeAddress",
        "HomePhone",
        "WorkAddress",
        "WorkPhone"
      ],
      "attribute-mappings": [
        {
          "attribute-name": "CustomerName",
          "value": "${FirstName},${LastName}"
        },
        {
          "attribute-name": "ContactDetails",
          "value": {
            "Home": {
              "Address": "${HomeAddress}",
              "Phone": "${HomePhone}"
            },
            "Work": {
              "Address": "${WorkAddress}",
              "Phone": "${WorkPhone}"
            }
          }
        },
        {
          "attribute-name": "DateOfBirth",
          "value": "${DateOfBirth}"
        }
      ]
    }
  }
]

```

}

Note

특정 테이블을 위한 제어 레코드에 대한 partition-key 값은 `TaskId.SchemaName.TableName`입니다. 특정 작업을 위한 제어 레코드에 대한 partition-key 값은 해당 레코드의 `TaskId`입니다. partition-key 값을 객체 매핑에 지정해도 제어 레코드에 대한 partition-key에는 영향이 없습니다.

테이블 매핑 규칙 `attribute-name`에서 `i`로 partition-key-type 설정된 경우 소스 테이블의 열 또는 매핑에 정의된 사용자 지정 열을 참조 partition-key-name 해야 하는를 지정해야 합니다. 또한 소스 열이 대상 Kafka 주제에 매핑되는 방식을 정의하려면 제공해야 attribute-mappings 합니다.

객체 매핑을 사용한 멀티주제 복제

기본적으로 AWS DMS 작업은 모든 소스 데이터를 다음 Kafka 주제 중 하나로 마이그레이션합니다.

- AWS DMS 대상 엔드포인트의 주제 필드에 지정된 대로.
- 대상 엔드포인트의 Topic 필드가 채워지지 않고 Kafka `auto.create.topics.enable` 설정이 `true`로 설정된 경우에는 `kafka-default-topic`에서 지정한 대로 마이그레이션합니다.

AWS DMS 엔진 버전 3.4.6 이상에서는 `kafka-target-topic` 속성을 사용하여 마이그레이션된 각 소스 테이블을 별도의 주제에 매핑할 수 있습니다. 예를 들어, 다음 객체 매핑 규칙은 원본 테이블 `Customer`과 `Address`를 Kafka 주제 `customer_topic` 및 `address_topic`에 각각 마이그레이션합니다. 동시에 `Test`는 스키마의 테이블을 포함한 다른 모든 소스 `Bills` 테이블을 대상 엔드포인트에 지정된 주제로 AWS DMS 마이그레이션합니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      }
    }
  ]
}
```

```

    },
    {
      "rule-type": "object-mapping",
      "rule-id": "2",
      "rule-name": "MapToKafka1",
      "rule-action": "map-record-to-record",
      "kafka-target-topic": "customer_topic",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Customer"
      },
      "partition-key-type": "constant"
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "3",
      "rule-name": "MapToKafka2",
      "rule-action": "map-record-to-record",
      "kafka-target-topic": "address_topic",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Address"
      },
      "partition-key-type": "constant"
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "4",
      "rule-name": "DefaultMapToKafka",
      "rule-action": "map-record-to-record",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Bills"
      }
    }
  ]
}

```

Kafka 멀티주제 복제를 사용하면 단일 복제 작업을 사용하여 원본 테이블을 그룹화하여 별도의 Kafka 주제로 마이그레이션할 수 있습니다.

Apache Kafka 메시지 형식

JSON 출력은 단지 키-값 페어의 목록입니다.

RecordType

레코드 유형은 데이터나 제어일 수 있습니다. 데이터 레코드는 소스 내의 실제 행을 나타냅니다. 제어 레코드는 스트림 내의 중요 이벤트를 나타냅니다(예: 작업의 재시작).

연산

데이터 레코드의 경우, 작업은 load, insert, update 또는 delete일 수 있습니다.

데이터 레코드의 경우, 작업은 create-table, rename-table, drop-table, change-columns, add-column, drop-column, rename-column 또는 column-type-change일 수 있습니다.

SchemaName

레코드에 대한 원본 스키마입니다. 제어 레코드의 경우, 이 필드는 비워둘 수 있습니다.

TableName

레코드에 대한 소스 테이블입니다. 제어 레코드의 경우, 이 필드는 비워둘 수 있습니다.

타임스탬프

JSON 메시지가 구성된 경우의 타임스탬프입니다. 이 필드는 ISO 8601 형식으로 구성되었습니다.

다음 JSON 메시지 예제는 모든 추가 메타데이터가 포함된 데이터 유형 메시지를 보여줍니다.

```
{
  "data":{
    "id":100000161,
    "fname":"val61s",
    "lname":"val61s",
    "REGION":"val61s"
  },
  "metadata":{
    "timestamp":"2019-10-31T22:53:59.721201Z",
    "record-type":"data",
    "operation":"insert",
    "partition-key-type":"primary-key",
    "partition-key-value":"sbtest.sbtest_x.100000161",
    "schema-name":"sbtest",
    "table-name":"sbtest_x",
    "transaction-id":9324410911751,
    "transaction-record-id":1,
    "prev-transaction-id":9324410910341,
```

```

    "prev-transaction-record-id":10,
    "commit-timestamp":"2019-10-31T22:53:55.000000Z",
    "stream-position":"mysql-bin-
changelog.002171:36912271:0:36912333:9324410911751:mysql-bin-changelog.002171:36912209"
  }
}

```

다음 JSON 메시지 예제는 컨트롤 유형 메시지를 보여줍니다.

```

{
  "control":{
    "table-def":{
      "columns":{
        "id":{
          "type":"WSTRING",
          "length":512,
          "nullable":false
        },
        "fname":{
          "type":"WSTRING",
          "length":255,
          "nullable":true
        },
        "lname":{
          "type":"WSTRING",
          "length":255,
          "nullable":true
        },
        "REGION":{
          "type":"WSTRING",
          "length":1000,
          "nullable":true
        }
      },
      "primary-key":[
        "id"
      ],
      "collation-name":"latin1_swedish_ci"
    },
    "metadata":{
      "timestamp":"2019-11-21T19:14:22.223792Z",

```



```

    "record-type": "control",
    "operation": "create-table",
    "partition-key-type": "task-id",
    "schema-name": "sbtest",
    "table-name": "sbtest_t1"
  }
}

```

Amazon OpenSearch Service 클러스터로의 대상으로 사용 AWS Database Migration Service

AWS DMS 를 사용하여 Amazon OpenSearch Service(OpenSearch Service)로 데이터를 마이그레이션할 수 있습니다. OpenSearch Service는 OpenSearch Service 클러스터를 손쉽게 배포, 운영 및 확장할 수 있도록 해주는 관리형 서비스입니다.

OpenSearch Service에서는 인덱스와 문서로 작업할 수 있습니다. 인덱스는 문서 모음이고 문서는 스칼라 값, 배열 및 기타 객체를 포함하는 JSON 객체입니다. JSON 기반 쿼리 언어를 OpenSearch 제공하므로 인덱스의 데이터를 쿼리하고 해당 문서를 검색할 수 있습니다.

AWS DMS 가 OpenSearch Service의 대상 엔드포인트에 대한 인덱스를 생성할 때 소스 엔드포인트에서 각 테이블에 대해 하나의 인덱스를 생성합니다. OpenSearch Service 인덱스를 생성하기 위한 비용은 다음과 같은 다양한 요인에 따라 달라집니다. 생성된 인덱스 수, 이러한 인덱스의 총 데이터 양, 각 문서에 대해 OpenSearch 저장하는 소량의 메타데이터입니다.

마이그레이션 범위에 적합한 컴퓨팅 및 스토리지 리소스를 갖춘 OpenSearch Service 클러스터를 구성합니다. 사용할 복제 작업에 따라 다음 요소를 고려하는 것이 좋습니다.

- 전체 데이터 로드와 전송의 경우, 마이그레이션할 데이터의 총량과 전송 속도를 고려합니다.
- 지속적인 변경 사항을 복제하는 경우, 업데이트 빈도와 중단 간 지연 시간 요건을 고려합니다.

또한 문서 개수에 주의하며 OpenSearch 클러스터의 인덱스 설정을 구성합니다.

멀티스레드 전체 로드 작업 설정

전송 속도를 높이기 위해 OpenSearch Service 대상 클러스터에 대한 멀티스레드 전체 로드를 AWS DMS 지원합니다. 다음을 포함하는 작업 설정으로 멀티스레딩을 AWS DMS 지원합니다.

- `MaxFullLoadSubTasks` – 병렬로 로드할 최대 소스 테이블 수를 표시하려면 이 옵션을 사용합니다. DMS는 전용 하위 작업을 사용하여 각 테이블을 해당 OpenSearch Service 대상 인덱스에 로드합니다. 기본값은 8이며 최대값은 49입니다.

- `ParallelLoadThreads` - 이 옵션을 사용하여 각 테이블을 OpenSearch Service 대상 인덱스로 로드하는 데 AWS DMS 사용하는 스레드 수를 지정합니다. OpenSearch Service 대상의 최대값은 32입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.

Note

`ParallelLoadThreads`를 기본값(0)에서 변경하지 않으면 AWS DMS 는 한 번에 하나의 레코드를 전송합니다. 이 접근 방식은 OpenSearch Service 클러스터에서 고유의 로드를 생성합니다. 이 옵션을 1 이상으로 설정해야 합니다.

- `ParallelLoadBufferSize` - 병렬 로드 스레드에서 데이터를 OpenSearch Service 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 `ParallelLoadThreads`와 함께 사용하세요. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다.

DMS가 멀티스레딩을 사용하여 OpenSearch Service 클러스터를 로드하는 방법에 대한 자세한 내용은 AWS 블로그 게시물 [마이그레이션을 위한 AWS Database Migration Service Amazon OpenSearch Service 규모 조정을 참조하세요](#).

멀티스레드 CDC 로드 작업 설정

작업 설정을 사용하여 `PutRecords` API 직접 호출의 동작을 수정하여 OpenSearch Service 대상 클러스터의 변경 데이터 캡처(CDC) 성능을 개선할 수 있습니다. 이렇게 하려면 `ParallelApply*` 작업 설정을 사용하여 동시 스레드 수, 스레드당 대기열 및 버퍼에 저장할 레코드 수를 지정해야 합니다. 예를 들어 CDC 로드를 수행하고 32개의 스레드를 병렬로 적용한다고 가정하겠습니다. 또한 버퍼당 50개 레코드가 저장된 스레드당 64개 대기열에 액세스하려고 합니다.

Note

CDC에서 Amazon OpenSearch Service 대상 엔드포인트로의 `ParallelApply*` 작업 설정 사용에 대한 지원은 AWS DMS 버전 3.4.0 이상에서 사용할 수 있습니다.

CDC 성능을 높이기 위해서는 다음 작업 설정을 AWS DMS 지원합니다.

- `ParallelApplyThreads` - CDC 로드 중에가 데이터 레코드를 OpenSearch Service 대상 엔드포인트로 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.

- `ParallelApplyBufferSize` – CDC 로드 중에 OpenSearch Service 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대 값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.
- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 OpenSearch Service 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다.

`ParallelApply*` 작업 설정을 사용할 때 `partition-key-type` 기본값은 테이블의 `schema-name.table-name`가 아니라 `primary-key`입니다.

관계형 데이터베이스 테이블에서 OpenSearch Service 인덱스로 마이그레이션

AWS DMS 는 OpenSearch Service의 스칼라 데이터 유형으로 데이터 마이그레이션을 지원합니다. Oracle 또는 MySQL 같은 관계형 데이터베이스에서 OpenSearch Service로 마이그레이션할 때 필요하다면 이 데이터를 저장하는 방법을 다시 구성할 수 있습니다.

AWS DMS 는 다음과 같은 OpenSearch Service 스칼라 데이터 형식을 지원합니다.

- 부울
- Date
- Float
- 정수
- 문자열

AWS DMS 는 날짜 유형의 데이터를 문자열 유형으로 변환합니다. 사용자 지정 매핑을 지정하여 이러한 날짜를 해석할 수 있습니다.

AWS DMS 는 LOB 데이터 형식의 마이그레이션을 지원하지 않습니다.

Amazon OpenSearch Service를의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service

OpenSearch Service 데이터베이스를 대상으로 사용하기 전에 AWS Identity and Access Management (IAM) 역할을 생성해야 AWS DMS합니다. 이 역할은가 대상 엔드포인트의 OpenSearch Service 인덱스에 AWS DMS 액세스하도록 허용해야 합니다. 최소 액세스 권한 집합이 다음 IAM 정책에 나와 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

OpenSearch Service로의 마이그레이션에 사용하는 역할에는 다음 권한이 있어야 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "es:ESHttpDelete",
        "es:ESHttpGet",
        "es:ESHttpHead",
        "es:ESHttpPost",
        "es:ESHttpPut"
      ],
      "Resource": "*"
    }
  ]
}
```

앞의 예에서 **region**를 AWS 리전 식별자로, **account-id**로, **domain-name**를 Amazon OpenSearch Service 도메인 이름으로 바꿉니다. `arn:aws:es:us-west-2:123456789012:domain/my-es-domain`를 예로 들 수 있음

OpenSearch Service를 대상으로 사용 시 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 OpenSearch Service 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 --`elasticsearch-settings` '{"**EndpointSetting**': "**value**", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

OpenSearch Service를 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

속성 이름	유효값	기본값과 설명
FullLoadErrorPercentage	0보다 크고 100 이하인 양의 정수입니다.	10 - 전체 로드 작업의 경우 이 속성이 작업 실패 전에 허용되는 오류의 임계값을 결정합니다. 예를 들어 소스 엔드포인트에 1,500개 행이 있으며 이 파라미터가 10으로 설정되어 있다고 가정합니다. 그러면에서 대상 엔드포인트에 쓸 때 150개 이상의 오류(행 수의 10%)가 AWS DMS 발생하면 작업이 실패합니다.
ErrorRetryDuration	0보다 큰 양의 정수입니다.	300 - 대상 엔드포인트에서 오류가 발생하면 몇 초 동안 AWS DMS 재시도합니다. 그렇지 않으면 작업이 실패한 것입니다.
UseNewMappingType	true 또는 false	false, 하지만 opensearch v2.x를 사용하여 작업하려면 true로 설정해야 합니다.

Amazon OpenSearch Service를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

Amazon OpenSearch Service를 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- OpenSearch Service는 동적 매핑(자동 추정)을 사용하여 마이그레이션된 데이터에 사용할 데이터 형식을 결정합니다.
- OpenSearch Service는 각 문서를 고유한 ID로 저장합니다. 다음은 ID의 예입니다.

```
"_id": "D359F8B537F1888BC71FE20B3D79EAE6674BE7ACA9B645B0279C7015F6FF19FD"
```

각 문서 ID의 길이는 스토리지 요구 사항에 부합하도록 64바이트입니다. 예를 들어 AWS DMS 소스에서 100,000개의 행을 마이그레이션하는 경우 결과 OpenSearch Service 인덱스에는 추가 6,400,000바이트에 대한 스토리지가 필요합니다.

- OpenSearch Service에서는 프라이머리 키 속성 업데이트를 허용하지 않습니다. 대상에 원치 않는 데이터가 발생할 수 있으므로 변경 데이터 캡처(CDC)와 함께 지속적인 복제 사용 시 이 제한이 중요합니다. CDC 모드에서 프라이머리 키는 32바이트 길이의 SHA256 값으로 매핑됩니다. 이는 사람이 읽을 수 있는 64바이트 문자열로 변환되며 OpenSearch Service 문서 ID로 사용됩니다.
- 에서 마이그레이션할 수 없는 항목이 AWS DMS 발견되면 Amazon CloudWatch Logs에 오류 메시지를 기록합니다. 이 동작은 예외 테이블에 오류를 쓰는 다른 AWS DMS 대상 엔드포인트의 동작과 다릅니다.
- AWS DMS 는 마스터 사용자 및 암호로 세분화된 액세스 제어가 활성화된 Amazon ES 클러스터에 대한 연결을 지원하지 않습니다.
- AWS DMS 는 OpenSearch Service 서버리스를 지원하지 않습니다.
- OpenSearch Service는 기존 인덱스에 데이터를 쓰는 것을 지원하지 않습니다.
- 복제 작업 설정 TargetTablePrepMode:TRUNCATE_BEFORE_LOAD는 OpenSearch 대상 엔드포인트와 함께 사용할 수 없습니다.
- 를 사용하여 Amazon Elasticsearch로 데이터를 마이그레이션할 때 AWS DMS소스 데이터에 기본 키 또는 고유 식별자 열이 있어야 합니다. 소스 데이터에 프라이머리 키 또는 고유 식별자가 없는 경우 define-primary-key 변환 규칙을 사용하여 정의해야 합니다.

Amazon OpenSearch Service의 대상 데이터 유형

가 이기종 데이터베이스에서 데이터를 AWS DMS 마이그레이션하면 서비스는 소스 데이터베이스의 데이터 형식을 AWS DMS 데이터 형식이라는 중간 데이터 형식으로 매핑합니다. 그런 다음, 중간 데이터 형식을 대상 데이터 형식에 매핑합니다. 다음 표에는 OpenSearch Service에서 매핑되는 각 AWS DMS 데이터 유형과 데이터 유형이 나와 있습니다.

AWS DMS 데이터 유형	OpenSearch Service 데이터 형식
부울	부울
날짜	문자열

AWS DMS 데이터 유형	OpenSearch Service 데이터 형식
시간	날짜
타임스탬프	날짜
INT4	정수
Real4	실수
UINT4	정수

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

Amazon DocumentDB를 AWS Database Migration Service의 대상으로 사용

AWS DMS에서 지원하는 Amazon DocumentDB 버전(MongoDB 호환)에 대한 자세한 내용은 섹션을 참조하세요 [의 대상 AWS DMS](#). AWS DMS를 사용하면 AWS DMS가 지원하는 소스 데이터 엔진에서 Amazon DocumentDB(MongoDB 호환)로 데이터를 마이그레이션할 수 있습니다. 소스 엔진은 Amazon RDS, Aurora 또는 Amazon S3와 같은 AWS 관리형 서비스에 있을 수 있습니다. 또는 Amazon EC2 또는 온프레미스에서 실행되는 MongoDB와 같이 자체 관리형 데이터베이스를 기반으로 할 수 있습니다.

AWS DMS를 사용하여 소스 데이터를 Amazon DocumentDB 데이터베이스, 컬렉션 또는 문서에 복제할 수 있습니다.

Note

소스 엔드포인트가 MongoDB 또는 Amazon DocumentDB인 경우 문서 모드에서 마이그레이션을 실행하세요.

MongoDB는 데이터를 이진 JSON 형식(BSON)으로 저장합니다. Amazon DocumentDB에서 지원하는 모든 BSON 데이터 형식을 AWS DMS 지원합니다. 이러한 데이터 형식의 목록은 Amazon DocumentDB 개발자 안내서의 [지원되는 MongoDB API, 작업 및 데이터 형식](#)을 참조하세요.

소스 엔드포인트가 관계형 데이터베이스인 경우는 다음과 같이 데이터베이스 객체를 Amazon DocumentDB에 AWS DMS 매핑합니다.

- 관계형 데이터베이스 또는 데이터베이스 스키마는 Amazon DocumentDB 데이터베이스에 매핑됩니다.
- 관계형 데이터베이스 내 테이블은 Amazon DocumentDB의 컬렉션에 매핑됩니다.
- 관계형 테이블의 레코드는 Amazon DocumentDB의 문서에 매핑됩니다. 각 문서는 소스 레코드의 데이터로 구성됩니다.

소스 엔드포인트가 Amazon S3이면 그 결과로 만들어지는 Amazon DocumentDB 객체는 Amazon S3을 위한 AWS DMS 매핑 규칙에 해당합니다. 예를 들어 다음 URI를 고려해 보세요.

```
s3://amzn-s3-demo-bucket/hr/employee
```

이 경우는 다음과 같이의 객체amzn-s3-demo-bucket를 Amazon DocumentDB에 AWS DMS 매핑합니다.

- URI의 최상위 부분(hr)은 Amazon DocumentDB 데이터베이스로 매핑됩니다.
- 다음 URI 부분(employee)은 Amazon DocumentDB 컬렉션으로 매핑됩니다.
- employee의 각 객체는 Amazon DocumentDB의 문서에 매핑됩니다.

Amazon S3의 매핑 규칙에 관한 자세한 내용은 [Amazon S3를의 소스로 사용 AWS DMS](#) 섹션을 참조하세요.

Amazon DocumentDB 엔드포인트 설정

AWS DMS 버전 3.5.0 이상에서는 병렬 스레드 및 대량 작업에 대한 작업 설정을 조정하여 Amazon DocumentDB 엔드포인트의 변경 데이터 캡처(CDC) 성능을 개선할 수 있습니다. 이렇게 하려면 ParallelApply* 작업 설정을 사용하여 동시 스레드 수, 스레드당 대기열 및 버퍼에 저장할 레코드 수를 지정해야 합니다. 예를 들어 CDC 로드를 수행하고 128개의 스레드를 병렬로 적용한다고 가정하겠습니다. 또한 버퍼당 50개 레코드가 저장된 스레드당 64개 대기열에 액세스하려고 합니다.

CDC 성능을 높이기 위해서는 다음 작업 설정을 AWS DMS 지원합니다.

- ParallelApplyThreads - CDC 로드 중에가 데이터 레코드를 Amazon DocumentDB 대상 엔드포인트로 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- ParallelApplyBufferSize - CDC 로드 중에 Amazon DocumentDB 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대

값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.

- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Amazon DocumentDB 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다. 기본값은 1입니다. 최대 한도는 512입니다.

Note

Amazon DocumentDB 대상의 경우 병렬 CDC 적용으로 인해 보조 고유 인덱스를 사용하거나 엄격한 변경 순서가 필요한 워크로드에 중복 키 오류가 발생하거나 CDC가 중단될 수 있습니다. 이러한 워크로드에 대해 기본 단일 스레드 CDC 적용 구성을 사용합니다.

Amazon DocumentDB를 대상으로 사용하는 방법에 대한 자세한 내용은 다음 섹션을 AWS DMS참조하세요.

주제

- [소스에서 Amazon DocumentDB 대상으로 데이터 매핑](#)
- [Amazon DocumentDB Elastic Clusters에 대상으로 연결](#)
- [Amazon DocumentDB를 대상으로 지속적 복제](#)
- [Amazon DocumentDB를 대상으로 사용할 때 적용되는 제한 사항](#)
- [Amazon DocumentDB를 대상으로 하는 엔드포인트 설정 사용](#)
- [Amazon DocumentDB의 대상 데이터 형식](#)

Note

마이그레이션 프로세스에 대한 step-by-step 연습은 AWS Database Migration Service Step-by-Step 마이그레이션 안내서의 [MongoDB에서 Amazon DocumentDB로](#) 마이그레이션을 참조하세요.

소스에서 Amazon DocumentDB 대상으로 데이터 매핑

AWS DMS 는 소스 엔드포인트에서 레코드를 읽고 읽는 데이터를 기반으로 JSON 문서를 구성합니다. 각 JSON 문서에 대해 고유 식별자 역할을 할 `_id` 필드를 결정해야 AWS DMS 합니다. 그런 다음, `_id` 필드를 프라이머리 키로 사용하여 JSON 문서를 Amazon DocumentDB 컬렉션에 기록합니다.

단일 열인 소스 데이터

단일 열로 구성된 소스 데이터는 문자열 형식으로 간주됩니다. (소스 엔진에 따라 실제 데이터 형식은 VARCHAR, NVARCHAR, TEXT, LOB, CLOB 등일 수 있습니다.)는 데이터가 유효한 JSON 문서라고 AWS DMS 가정하고 데이터를 Amazon DocumentDB에 있는 그대로 복제합니다.

그 결과로 생성된 JSON 문서에 `_id`라는 필드가 포함되어 있는 경우, 해당 필드는 Amazon DocumentDB에서 고유한 `_id`로 사용됩니다.

JSON에 `_id` 필드가 포함되어 있지 않으면 Amazon DocumentDB에서 `_id` 값이 자동으로 생성됩니다.

다중 열인 소스 데이터

소스 데이터가 여러 열로 구성된 경우는 이러한 모든 열에서 JSON 문서를 AWS DMS 구성합니다. 문서의 `_id` 필드를 확인하려면 다음과 같이 AWS DMS 진행합니다.

- 열 중 하나의 이름이 `_id`이면 해당 열의 데이터가 대상 `_id`로 사용됩니다.
- `_id` 열이 없지만 소스 데이터에 기본 키 또는 고유 인덱스가 있는 경우는 해당 키 또는 인덱스 값을 `_id` 값으로 AWS DMS 사용합니다. 또한 프라이머리 키 또는 고유 인덱스에서 나온 데이터는 JSON 문서에 명시적 필드 형태로 나타납니다.
- `_id` 열과 프라이머리 키 또는 고유 인덱스가 없으면 Amazon DocumentDB에서 `_id` 값이 자동으로 생성됩니다.

대상 엔드포인트에서 데이터 형식을 강제로 적용

AWS DMS 는 Amazon DocumentDB 대상 엔드포인트에 쓸 때 데이터 구조를 수정할 수 있습니다. 소스 엔드포인트에서 열과 테이블의 이름을 변경하거나 작업 실행 시 적용되는 변환 규칙을 제공하여 이러한 변경을 요청할 수 있습니다.

중첩 JSON 문서(json_prefix) 사용

데이터 형식을 강제로 적용하려면 직접, 또는 변환 기능을 사용해 소스 열 이름에 json_(즉, json_*columnName*) 접두사를 붙일 수 있습니다. 이 경우, 열은 문자열 필드가 아니라 대상 문서 내에 중첩 JSON 문서 형태로 생성됩니다.

예를 들어 MongoDB 소스 엔드포인트에서 다음 문서를 마이그레이션하고 싶다고 가정해 봅시다.

```
{
  "_id": "1",
  "FirstName": "John",
  "LastName": "Doe",
  "ContactDetails": "{\"Home\": {\"Address\": \"Boston\", \"Phone\": \"1111111\"}, \"Work\": { \"Address\": \"Boston\", \"Phone\": \"222222222\"}}"
```

소스 데이터 형식을 강제로 적용하지 않으면 기본적으로 포함된 ContactDetails 문서가 문자열 형태로 마이그레이션됩니다.

```
{
  "_id": "1",
  "FirstName": "John",
  "LastName": "Doe",
  "ContactDetails": "{\"Home\\\": {\\\"Address\\\": \\\"Boston\\\", \\\"Phone\\\": \\\"1111111\\\"}, \\\"Work\\\": { \\\"Address\\\": \\\"Boston\\\", \\\"Phone\\\": \\\"222222222\\\"}}"
```

하지만 JSON 객체에 ContactDetails를 강제 적용하기 위한 변환 규칙을 추가할 수 있습니다. 예를 들어 원래의 소스 열 이름이 ContactDetails라고 가정해 봅시다. 데이터 형식을 중첩된 JSON으로 강제 변환하려면 소스에 “*json_” 접두사를 수동으로 추가하거나 변환 규칙을 통해 소스 엔드포인트의 열 이름을 “JSON_ContactDetails”로 변경해야 합니다. 예를 들어, 아래의 변환 규칙을 사용할 수 있습니다.

```
{
  "rules": [
    {
      "rule-type": "transformation",
      "rule-id": "1",
      "rule-name": "1",
```

```

    "rule-target": "column",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%",
      "column-name": "ContactDetails"
    },
    "rule-action": "rename",
    "value": "json_ContactDetails",
    "old-value": null
  }
]
}

```

AWS DMS 는 다음과 같이 ContactDetails 필드를 중첩 JSON으로 복제합니다.

```

{
  "_id": "1",
  "FirstName": "John",
  "LastName": "Doe",
  "ContactDetails": {
    "Home": {
      "Address": "Boston",
      "Phone": "1111111111"
    },
    "Work": {
      "Address": "Boston",
      "Phone": "2222222222"
    }
  }
}

```

JSON 배열(array_ prefix) 사용

데이터 형식을 강제로 적용하려면 직접, 또는 변환 기능을 사용해 열 이름에 array_(즉, array_*columnName*) 접두사를 붙일 수 있습니다. 이 경우는 열을 JSON 배열로 AWS DMS 간주하고 대상 문서에 이와 같이 생성합니다.

MongoDB 소스 엔드포인트에서 다음 문서를 마이그레이션하고 싶다고 가정해 봅시다.

```

{
  "_id" : "1",

```

```

    "FirstName": "John",
    "LastName": "Doe",

    "ContactAddresses": ["Boston", "New York"],

    "ContactPhoneNumbers": ["1111111111", "2222222222"]
  }

```

소스 데이터 형식을 강제로 적용하지 않으면 기본적으로 포함된 ContactDetails 문서가 문자열 형태로 마이그레이션됩니다.

```

{
  "_id": "1",
  "FirstName": "John",
  "LastName": "Doe",

  "ContactAddresses": "[\"Boston\", \"New York\"]",

  "ContactPhoneNumbers": "[\"1111111111\", \"2222222222\"]"
}

```

하지만 JSON 어레이에 ContactAddress 및 ContactPhoneNumbers를 강제 적용하기 위한 변환 규칙을 추가할 수 있습니다(아래 표 참조).

원래의 소스 열 이름	이름이 변경된 소스 열
ContactAddress	array_ContactAddress
ContactPhoneNumbers	array_ContactPhoneNumbers

AWS DMS 는 다음과 ContactPhoneNumbers 같이 ContactAddress 및를 복제합니다.

```

{
  "_id": "1",
  "FirstName": "John",
  "LastName": "Doe",
  "ContactAddresses": [
    "Boston",
    "New York"
  ]
}

```

```

    ],
    "ContactPhoneNumbers": [
        "1111111111",
        "2222222222"
    ]
}

```

TLS를 사용하여 Amazon DocumentDB에 연결

기본적으로 새로 생성한 Amazon DocumentDB 클러스터는 Transport Layer Security(TLS)을 사용한 보안 연결만 허용합니다. TLS가 활성화되면 Amazon DocumentDB에 대한 모든 연결에 퍼블릭 키가 필요합니다.

AWS 호스팅된 Amazon S3 버킷에서 파일을 다운로드하여 Amazon DocumentDB의 퍼블릭 키를 검색할 수 있습니다. `rds-combined-ca-bundle.pem` Amazon S3 이 파일을 다운로드하는 방법에 관한 자세한 내용은 Amazon DocumentDB 개발자 안내서의 [TLS를 사용한 연결 암호화](#)를 참조하세요.

이 .pem 파일을 다운로드한 후 다음 설명과 AWS DMS 같이 포함된 퍼블릭 키를 로 가져올 수 있습니다.

AWS Management Console

퍼블릭 키(.pem) 파일을 가져오려면

1. <https://console.aws.amazon.com/dms> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 인증서를 선택합니다.
3. 인증서 가져오기를 선택하고 다음과 같이 합니다.
 - 인증서 식별자에 해당 인증서의 고유 이름(예: docdb-cert)을 입력합니다.
 - 파일 가져오기에서 .pem 파일이 저장된 위치로 이동합니다.

원하는 대로 설정이 되었으면 새 CA 인증서 추가를 선택합니다.

AWS CLI

다음 예와 같이 `aws dms import-certificate` 명령을 사용하세요.

```

aws dms import-certificate \
    --certificate-identifier docdb-cert \

```

```
--certificate-pem file:///./rds-combined-ca-bundle.pem
```

AWS DMS 대상 엔드포인트를 생성할 때 인증서 식별자(예: docdb-cert)를 제공합니다. 또한 SSL 모드 파라미터를 `verify-full`로 설정합니다.

Amazon DocumentDB Elastic Clusters에 대상으로 연결

AWS DMS 버전 3.4.7 이상에서는 Amazon DocumentDB 대상 엔드포인트를 탄력적 클러스터로 생성할 수 있습니다. 대상 엔드포인트를 Elastic 클러스터로 생성하는 경우 기존 SSL 인증서는 작동하지 않으므로 Amazon DocumentDB Elastic 클러스터 엔드포인트에 새 SSL 인증서를 연결해야 합니다.

Amazon DocumentDB Elastic 클러스터 엔드포인트에 새 SSL 인증서를 연결하려면

1. 브라우저에서 <https://www.amazontrust.com/repository/SFSRootCAG2.pem>을 열고 콘텐츠를 고유한 파일 이름(예: SFSRootCAG2.pem)을 가진 .pem 파일에 저장합니다. 이 파일은 다음 단계에서 가져와야 하는 인증서 파일입니다.
2. Elastic 클러스터 엔드포인트를 생성한 후 다음 옵션을 설정합니다.
 - a. 엔드포인트 구성에서 새 CA 인증서 추가를 선택합니다.
 - b. 인증서 식별자에 **SFSRootCAG2.pem**을 입력합니다.
 - c. 인증서 파일 가져오기에서 파일 선택을 선택하고 이전에 다운로드한 SFSRootCAG2.pem 파일로 이동합니다.
 - d. SFSRootCAG2.pem 파일 다운로드를 선택하고 엽니다.
 - e. 인증서 가져오기를 선택합니다.
 - f. 인증서 선택 드롭다운에서 SFSRootCag2.pem을 선택합니다.

다운로드한 SFSRootCAG2.pem 파일의 새 SSL 인증서가 이제 Amazon DocumentDB Elastic 클러스터 엔드포인트에 연결됩니다.

Amazon DocumentDB를 대상으로 지속적 복제

Amazon DocumentDB를 대상으로 지속적인 복제(변경 데이터 캡처, CDC)를 활성화한 경우 AWS DMS 버전 3.5.0 이상에서는 이전 릴리스보다 20배 향상된 성능 향상을 제공합니다. 가 초당 최대 250개의 레코드를 AWS DMS 처리하는 이전 릴리스에서는 AWS DMS 이제가 초당 5,000개 이상의 레코드를 효율적으로 처리합니다. AWS DMS 또한 Amazon DocumentDB의 문서가 소스와 동기화되도록 합니다. 소스 레코드를 생성하거나 업데이트할 때 먼저 다음을 수행하여 영향을 받는 Amazon DocumentDB 레코드를 결정해야 AWS DMS 합니다.

- 소스 레코드에 `_id`라는 값이 있으면 해당 열의 값이 Amazon DocumentDB 컬렉션에서 해당되는 `_id`를 확인합니다.
- `_id` 열이 없지만 소스 데이터에 기본 키 또는 고유 인덱스가 있는 경우는 해당 키 또는 인덱스 값을 Amazon DocumentDB 컬렉션의 `_id`로 AWS DMS 사용합니다.
- 소스 레코드에 `_id` 열, 기본 키 또는 고유 인덱스가 없는 경우는 모든 소스 열을 Amazon DocumentDB 컬렉션의 해당 필드와 AWS DMS 일치시킵니다.

새 소스 레코드가 생성되면 Amazon DocumentDB에 해당 문서를 AWS DMS 작성합니다. 기존 소스 레코드가 업데이트되면 Amazon DocumentDB의 대상 문서에서 해당 필드를 AWS DMS 업데이트합니다. 대상 문서에는 존재하지만 소스 레코드에 존재하지 않는 필드는 그대로 남아 있습니다.

소스 레코드가 삭제되면 Amazon DocumentDB에서 해당 문서를 AWS DMS 삭제합니다.

소스에서 구조 변경(DDL)

지속적 복제가 이루어지면서 소스 데이터 구조(테이블, 열 등)의 변경 사항이 Amazon DocumentDB의 구조에 전파됩니다. 관계형 데이터베이스에서 이러한 변경 사항은 데이터 정의 언어(DDL) 명령문을 사용해 시작됩니다. 다음 표에서 AWS DMS가 이러한 변경 사항을 Amazon DocumentDB에 전파하는 방법을 확인할 수 있습니다.

소스의 DDL	Amazon DocumentDB 타겟에서의 효과
CREATE TABLE	빈 컬렉션을 생성합니다.
테이블 이름을 변경하는 명령문(RENAME TABLE, ALTER TABLE...RENAME 등)	컬렉션의 이름을 바꿉니다.
TRUNCATE TABLE	해당 컬렉션에서 모든 문서를 제거합니다(단, <code>HandleSourceTableTruncated</code> 가 <code>true</code> 인 경우에만). 자세한 내용은 변경 처리 DDL을 다루기 위한 작업 설정 섹션을 참조하세요.
DROP TABLE	해당 컬렉션을 삭제합니다(단, <code>HandleSourceTableDropped</code> 가 <code>true</code> 인 경우에만). 자세한 내용은 변경 처리 DDL을 다루기 위한 작업 설정 섹션을 참조하세요.

소스의 DDL	Amazon DocumentDB 타겟에서의 효과
테이블에 열을 추가하는 명령문(ALTER TABLE...ADD 등)	DDL 명령문이 무시되면서 경고가 발행됩니다. 첫 번째 INSERT가 소스에서 수행될 때 대상 문서에 새 필드가 추가됩니다.
ALTER TABLE...RENAME COLUMN	DDL 명령문이 무시되면서 경고가 발행됩니다. 첫 번째 INSERT가 소스에서 수행될 때 대상 문서에 새로운 이름의 필드가 추가됩니다.
ALTER TABLE...DROP COLUMN	DDL 명령문이 무시되면서 경고가 발행됩니다.
열 데이터 형식을 변경하는 명령문(ALTER COLUMN...MODIFY 등)	DDL 명령문이 무시되면서 경고가 발행됩니다. 첫 번째 INSERT가 소스에서 새 데이터 형식으로 수행될 때 이러한 새 데이터 형식을 가진 필드와 함께 대상 문서가 생성됩니다.

Amazon DocumentDB를 대상으로 사용할 때 적용되는 제한 사항

Amazon DocumentDB를 대상으로 사용할 때는 AWS DMS 다음과 같은 제한 사항이 적용됩니다.

- Amazon DocumentDB에서 모음 이름에는 달러 기호(\$)가 포함될 수 없습니다. 또한 데이터베이스 이름에는 어떤 Unicode 문자도 포함될 수 없습니다.
- AWS DMS 는 여러 소스 테이블을 단일 Amazon DocumentDB 컬렉션으로 병합하는 것을 지원하지 않습니다.
- 가 프라이머리 키가 없는 소스 테이블에서 변경 사항을 AWS DMS 처리하면 해당 테이블의 모든 LOB 열은 무시됩니다.
- 변경 테이블 옵션이 활성화되고 AWS DMS 에 "_id"라는 소스 열이 나타나면 해당 열은 변경 테이블에 "_id"(2개의 밑줄)로 표시됩니다.
- 소스 엔드포인트로 Oracle을 선택하면 Oracle 소스에서 전체 보충 로깅을 활성화해야 합니다. 그렇지 않으면 변경되지 않은 소스의 열이 있을 경우에 Amazon DocumentDB에 데이터가 null 값으로 로드됩니다.
- 복제 작업 설정 TargetTablePrepMode:TRUNCATE_BEFORE_LOAD는 DocumentDB 대상 엔드포인트와 함께 사용할 수 없습니다.
- MongoDB 제한 컬렉션은 Amazon DocumentDB에서 지원되지 않습니다. 그러나 대상 DocumentDB에서 캡핑되지 않은 컬렉션과 같은 객체를 AWS DMS 자동으로 마이그레이션합니다.

- Amazon DocumentDB 대상에 병렬 CDC가 적용되면 중복 키 오류가 발생하거나 보조 고유 인덱스를 사용하거나 엄격한 변경 순서가 필요한 워크로드에 CDC가 중단될 수 있습니다. 이러한 워크로드의 경우 기본 단일 스레드 CDC 적용 구성을 사용합니다.

Amazon DocumentDB를 대상으로 하는 엔드포인트 설정 사용

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Amazon DocumentDB 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 `--doc-db-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Amazon DocumentDB를 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

속성 이름	유효값	기본값과 설명
replicate ShardColl ections	부울 true false	값이 true일 때 이 엔드포인트 설정은 다음과 같은 영향을 미치며 다음과 같은 제한이 적용됩니다. • AWS DMS 는 대상 샤드 컬렉션에 데이터를 복제할 수 있습니다. 이 설정은 대상 DocumentDB 엔드포인트가 Elastic 클러스터인 경우에만 적용됩니다. • TargetTablePrepMode 를 DO_NOTHING 으로 설정해야 합니다. • AWS DMS 는 마이그레이션 false 중에 자동으로 useUpdateLookup 를 로 설정합니다.

Amazon DocumentDB의 대상 데이터 형식

다음 표에서는 AWS DMS 사용 시 지원되는 Amazon DocumentDB 대상 데이터 형식과 AWS DMS 데이터 형식의 기본 매핑을 확인할 수 있습니다. AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

AWS DMS 데이터 형식	Amazon DocumentDB 데이터 형식
BOOLEAN	부울
BYTES	이진 데이터

AWS DMS 데이터 형식	Amazon DocumentDB 데이터 형식
DATE	Date
TIME	문자열(UTF8)
DATETIME	Date
INT1	32비트 정수
INT2	32비트 정수
INT4	32비트 정수
INT8	64비트 정수
NUMERIC	문자열(UTF8)
REAL4	배정밀도 실수
REAL8	배정밀도 실수
STRING	데이터가 JSON으로 인식되면는 데이터를 문서로 Amazon DocumentDB로 AWS DMS 마이그레이션합니다. 그렇지 않으면 데이터가 문자열(UTF8)로 매핑됩니다.
UINT1	32비트 정수
UINT2	32비트 정수
UINT4	64비트 정수
UINT8	문자열(UTF8)
WSTRING	데이터가 JSON으로 인식되면는 데이터를 문서로 Amazon DocumentDB로 AWS DMS 마이그레이션합니다. 그렇지 않으면 데이터가 문자열(UTF8)로 매핑됩니다.
BLOB	바이너리

AWS DMS 데이터 형식	Amazon DocumentDB 데이터 형식
CLOB	데이터가 JSON으로 인식되면는 데이터를 문서로 Amazon DocumentDB로 AWS DMS 마이그레이션합니다. 그렇지 않으면 데이터가 문자열(UTF8)로 매핑됩니다.
NCLOB	데이터가 JSON으로 인식되면는 데이터를 문서로 Amazon DocumentDB로 AWS DMS 마이그레이션합니다. 그렇지 않으면 데이터가 문자열(UTF8)로 매핑됩니다.

Amazon Neptune을의 대상으로 사용 AWS Database Migration Service

Amazon Neptune은 빠르고 안정적인 종합 관리형 그래프 데이터베이스 서비스로, 고도로 연결된 데이터셋을 사용하는 애플리케이션을 쉽게 빌드하고 실행할 수 있습니다. Neptune의 핵심은 특별한 용도의 고성능 그래프 데이터베이스 엔진입니다. 이 엔진은 수십억 개의 관계를 저장하고 몇 밀리초의 지연 시간으로 그래프를 쿼리하도록 최적화되었습니다. Neptune은 인기 있는 그래프 쿼리 언어인 Apache TinkerPop Gremlin과 W3C의 SPARQL을 지원합니다. Amazon Neptune에 관한 자세한 내용은 Amazon Neptune 사용 설명서의 [Amazon Neptune이란 무엇인가?](#) 섹션을 참조하세요.

Neptune과 같은 그래프 데이터베이스가 없더라도 관계형 데이터베이스에서 고도로 연결된 데이터를 모델링할 수 있습니다. 데이터에 동적 연결이 있을 가능성이 있기 때문에 이러한 데이터 소스를 사용하는 애플리케이션은 연결된 데이터 쿼리를 SQL로 모델링해야 합니다. 이 방법을 사용하려면 그래프 쿼리를 SQL로 변환하기 위해 추가 계층을 작성해야 합니다. 또한 관계형 데이터베이스에는 스키마 엄격성이 있습니다. 변경되는 연결을 모델링하기 위해 스키마를 변경하면 가동 중지 시간이 발생하고 새 스키마를 지원하기 위해 쿼리 변환을 추가로 유지 관리해야 합니다. 쿼리 성능은 애플리케이션을 설계하는 동안 고려해야 할 또 하나의 큰 제약 조건입니다.

그래프 데이터베이스는 이러한 상황을 크게 단순화할 수 있습니다. 스키마 없이 서식 있는 그래프 쿼리 계층(Gremlin 또는 SPARQL) 및 그래프 쿼리에 최적화된 인덱스는 유연성과 성능을 향상시킵니다. Amazon Neptune 그래프 데이터베이스에는 유틸리티 암호화, 보안 승인 계층, 기본 백업, 다중 AZ 지원, 읽기 전용 복제 지원 등의 엔터프라이즈 기능도 있습니다.

를 사용하면 고도로 연결된 그래프를 모델링하는 관계형 데이터를 지원되는 SQL 데이터베이스의 DMS 소스 엔드포인트에서 Neptune 대상 엔드포인트로 마이그레이션할 AWS DMS 수 있습니다.

자세한 내용은 다음을 참조하세요.

주제

- [대상으로서 Amazon Neptune으로 마이그레이션 개요](#)
- [대상으로서 Amazon Neptune에 대한 엔드포인트 설정 지정](#)
- [Amazon Neptune에 대상으로 액세스하기 위한 IAM 서비스 역할 생성](#)
- [Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정](#)
- [대상으로서 Amazon Neptune으로 Gremlin 및 R2RML을 마이그레이션하기 위한 데이터 형식](#)
- [Amazon Neptune을 대상으로 사용 시 제한 사항](#)

대상으로서 Amazon Neptune으로 마이그레이션 개요

Neptune 대상으로 마이그레이션을 시작하기 전에 AWS 계정에 다음 리소스를 생성합니다.

- 대상 엔드포인트에 대한 Neptune 클러스터.
- 소스 엔드포인트에 AWS DMS 대해에서 지원하는 SQL 관계형 데이터베이스입니다.
- 대상 엔드포인트용 Amazon S3 버킷. Neptune 클러스터와 동일한 AWS 리전에이 S3 버킷을 생성합니다. 이 S3 버킷을 Neptune 데이터베이스에 대량 로드하는 대상 데이터의 중간 파일 스토리지로 AWS DMS 사용합니다. Amazon S3 버킷 생성에 관한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 [버킷 생성](#) 섹션을 참조하세요.
- Neptune 클러스터와 동일한 VPC에 있는 S3에 대한 가상 프라이빗 클라우드(VPC) 엔드포인트.
- IAM 정책을 포함하는 AWS Identity and Access Management (IAM) 역할입니다. 이 정책은 대상 엔드포인트에 대한 S3 버킷에 GetObject, PutObject, DeleteObject 및 ListObject 권한을 지정해야 합니다. 이 역할은 대상 S3 버킷 AWS DMS 과 Neptune 데이터베이스 모두에 대한 IAM 액세스 권한이 있는 및 Neptune 모두에서 수입합니다. 자세한 내용은 [Amazon Neptune에 대상으로 액세스하기 위한 IAM 서비스 역할 생성](#) 단원을 참조하십시오.

이러한 리소스를 확보한 후 Neptune 대상으로 마이그레이션을 설정하고 시작하는 방법은 콘솔 또는 DMS API를 사용하는 전체 로드 마이그레이션과 유사합니다. 그러나 Neptune 대상으로 마이그레이션하려면 몇 가지 고유한 단계가 필요합니다.

AWS DMS 관계형 데이터베이스를 Neptune으로 마이그레이션하려면

1. [복제 인스턴스 생성](#)에 설명된 대로 복제 인스턴스를 생성합니다.
2. 소스 엔드포인트에 AWS DMS 대해에서 지원하는 SQL 관계형 데이터베이스를 생성하고 테스트합니다.
3. Neptune 데이터베이스의 대상 엔드포인트를 만들고 테스트합니다.

대상 엔드포인트를 Neptune 데이터베이스에 연결하려면 Neptune 클러스터 엔드포인트 또는 Neptune 라이터 인스턴스 엔드포인트에 대한 서버 이름을 지정합니다. 또한에 대한 S3 버킷 폴더를 지정 AWS DMS 하여 Neptune 데이터베이스에 대량 로드할 중간 파일을 저장합니다.

마이그레이션 중에는 마이그레이션된 모든 대상 데이터를 사용자가 지정한 최대 파일 크기까지이 S3 버킷 폴더에 AWS DMS 저장합니다. 이 파일 스토리지가이 최대 크기에 도달하면 AWS DMS 대량은 저장된 S3 데이터를 대상 데이터베이스로 로드합니다. 그러면 폴더를 지워서 이후에 대상 데이터베이스에 로드할 추가 대상 데이터를 저장할 수 있습니다. 이러한 설정 지정에 관한 자세한 내용은 [대상으로서 Amazon Neptune에 대한 엔드포인트 설정 지정](#) 섹션을 참조하세요.

4. 1~3단계에서 만든 리소스를 사용하여 전체 로드 복제 작업을 생성한 후 다음을 수행합니다.
 - a. 평소와 같이 작업 테이블 매핑을 사용하여 적절한 선택 및 변환 규칙을 사용하여 관계형 데이터베이스에서 마이그레이션할 특정 소스 스키마, 테이블 및 뷰를 식별합니다. 자세한 내용은 [작업 설정을 지정하기 위한 테이블 매핑 사용](#) 섹션을 참조하세요.
 - b. 다음 중 하나를 선택하여 소스 테이블 및 뷰에서 대상 데이터베이스 그래프로의 매핑 규칙을 지정하여 Neptune 대상 매핑을 지정합니다.
 - Gremlin JSON — Gremlin JSON을 사용하여 Neptune 데이터베이스를 로드하는 방법에 관한 자세한 내용은 Amazon Neptune 사용 설명서의 [Gremlin 로드 데이터 형식](#)을 참조하세요.
 - SPARQL RDB에서 R2RML(Resource Description Framework Mapping Language)로 – SPARQL R2RML 사용에 관한 자세한 내용은 W3C 사양 [R2RML: RDB-RDF 매핑 언어](#)를 참조하세요.
 - c. 다음 중 하나를 수행하세요.
 - AWS DMS 콘솔을 사용하여 데이터베이스 마이그레이션 작업 생성 페이지에서 그래프 매핑 규칙을 사용하여 그래프 매핑 옵션을 지정합니다.
 - AWS DMS API를 사용하여 CreateReplicationTask API 호출의 TaskData 요청 파라미터를 사용하여 이러한 옵션을 지정합니다.

Gremlin JSON 및 SPARQL R2RML을 사용하여 그래프 매핑 규칙을 지정하는 방법에 관한 자세한 내용과 예제는 [Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정](#) 섹션을 참조하세요.

5. 마이그레이션 작업에 대한 복제를 시작합니다.

대상으로서 Amazon Neptune에 대한 엔드포인트 설정 지정

대상 엔드포인트를 만들거나 수정하려면 콘솔이나 CreateEndpoint 또는 ModifyEndpoint API 작업을 사용할 수 있습니다.

AWS DMS 콘솔의 Neptune 대상의 경우 엔드포인트 생성 또는 엔드포인트 수정 콘솔 페이지에서 엔드포인트별 설정을 지정합니다. CreateEndpoint, ModifyEndpoint의 경우 NeptuneSettings 옵션에 대한 요청 파라미터를 지정합니다. 다음 예에서는 CLI를 사용하여 이 작업을 수행하는 방법을 보여줍니다.

```
dms create-endpoint --endpoint-identifier my-neptune-target-endpoint
--endpoint-type target --engine-name neptune
--server-name my-neptune-db.cluster-cspckvklbvgf.us-east-1.neptune.amazonaws.com
--port 8192
--neptune-settings
  '{"ServiceAccessRoleArn":"arn:aws:iam::123456789012:role/myNeptuneRole",
    "S3BucketName":"amzn-s3-demo-bucket",
    "S3BucketFolder":"amzn-s3-demo-bucket-folder",
    "ErrorRetryDuration":57,
    "MaxFileSize":100,
    "MaxRetryCount": 10,
    "IAMAuthEnabled":false}'
```

여기서 CLI --server-name 옵션은 Neptune 클러스터 라이터 엔드포인트의 서버 이름을 지정합니다. 또는 Neptune 라이터 인스턴스 엔드포인트에 대한 서버 이름을 지정할 수 있습니다.

--neptune-settings 옵션 요청 파라미터는 다음과 같습니다.

- **ServiceAccessRoleArn** – (필수) Neptune 대상 엔드포인트에 대해 생성한 서비스 역할의 Amazon 리소스 이름(ARN)입니다. 자세한 내용은 [Amazon Neptune에 대상으로 액세스하기 위한 IAM 서비스 역할 생성](#) 섹션을 참조하세요.
- **S3BucketName** – (필수) DMS가 마이그레이션된 그래프 데이터를 Neptune 대상 데이터베이스에 대량으로 로드하기 전에 .csv 파일에 임시로 저장할 수 있는 S3 버킷의 이름입니다. DMS는 이 .csv 파일에 저장하기 전에 SQL 소스 데이터를 그래프 데이터에 매핑합니다.
- **S3BucketFolder** – (필수) DMS가 S3BucketName에 의해 지정된 S3 버킷에 마이그레이션된 그래프 데이터를 저장할 때 사용할 폴더 경로입니다.
- **ErrorRetryDuration** – (옵션) DMS는 이 시간(밀리초) 동안 기다렸다가 마이그레이션된 그래프 데이터를 Neptune 대상 데이터베이스에 대량 로드하고자 재시도한 다음 오류를 발생시킵니다. 기본 값은 250입니다.

- **MaxFileSize** – (옵션) DMS가 Neptune 대상 데이터베이스에 데이터를 대량으로 로드하기 전에 .csv 파일에 저장되어 있던 마이그레이션된 그래프 데이터의 최대 크기(KB)입니다. 기본값은 1,048,576KB(1GB)입니다. 성공하면 DMS는 버킷을 지우고 마이그레이션된 그래프 데이터의 다음 배치를 저장할 준비가 됩니다.
- **MaxRetryCount** – (옵션) DMS는 이 횟수만큼 Neptune 대상 데이터베이스로 마이그레이션된 그래프 데이터의 대량 로드를 재시도한 뒤에 오류를 발생시킵니다. 기본값은 5입니다.
- **IAMAuthEnabled** – (옵션) 이 엔드포인트에 대해 IAM 권한 부여를 활성화하려면 이 파라미터를 true로 설정하고 ServiceAccessRoleArn에서 지정한 서비스 역할에 적절한 IAM 정책 문서를 연결합니다. 기본값은 false입니다.

Amazon Neptune에 대상으로 액세스하기 위한 IAM 서비스 역할 생성

Neptune에 대상으로 액세스하려면 IAM을 사용하여 서비스 역할을 만듭니다. Neptune 엔드포인트 구성에 따라 이 역할에 IAM 정책의 일부 또는 전부를 연결하고 다음에 설명된 문서를 신뢰합니다. Neptune 엔드포인트를 생성할 때 이 서비스 역할의 ARN을 제공합니다. 이렇게 하면 AWS DMS 및 Amazon Neptune이 Neptune과 연결된 Amazon S3 버킷 모두에 액세스할 수 있는 권한을 받을 수 있습니다.

Neptune 엔드포인트 구성에서 NeptuneSettings의 IAMAuthEnabled 파라미터를 true로 설정한 경우, 다음과 같은 IAM 정책을 서비스 역할에 연결합니다. IAMAuthEnabled를 false로 설정하면 이 정책을 무시할 수 있습니다.

```
// Policy to access Neptune

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": "neptune-db:*",
      "Resource": "arn:aws:neptune-db:us-east-1:123456789012:cluster-CLG7H7FHK54AZGHEH6MNS55JKM/*"
    }
  ]
}
```

앞의 IAM 정책은 Resource에서 지정한 Neptune 대상 클러스터에 대한 전체 액세스를 허용합니다.

다음과 같은 IAM 정책을 서비스 역할에 연결합니다. 이 정책을 통해 DMS는 Neptune 대상 데이터베이스에 대량 로드하기 위해 생성한 S3 버킷에 마이그레이션된 그래프 데이터를 임시로 저장할 수 있습니다.

```
//Policy to access S3 bucket

{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "ListObjectsInBucket0",
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket"
    ]
  },
  {
    "Sid": "AllObjectActions",
    "Effect": "Allow",
    "Action": ["s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket/"
    ]
  },
  {
    "Sid": "ListObjectsInBucket1",
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": [
      "arn:aws:s3:::amzn-s3-demo-bucket",
      "arn:aws:s3:::amzn-s3-demo-bucket/"
    ]
  }
  ]
}
```

앞의 IAM 정책을 사용하면 계정에서 Neptune 대상에 대해 생성된 S3 버킷(arn:aws:s3:::amzn-s3-demo-bucket)의 콘텐츠를 쿼리할 수 있습니다. 또한 계정에서 모든 버킷 파일 및 폴더(arn:aws:s3:::amzn-s3-demo-bucket/)의 콘텐츠에 대해 완벽하게 작동할 수 있습니다.

신뢰 관계를 편집하고 다음 IAM 역할을 서비스 역할에 연결하여 AWS DMS 및 Amazon Neptune 데이터베이스 서비스가 역할을 수임할 수 있도록 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    },
    {
      "Sid": "neptune",
      "Effect": "Allow",
      "Principal": {
        "Service": "rds.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Neptune 대상 엔드포인트에 대해 이 서비스 역할을 지정하는 방법에 관한 자세한 내용은 [대상으로서 Amazon Neptune에 대한 엔드포인트 설정 지정](#) 섹션을 참조하세요.

Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정

생성하는 그래프 매핑 규칙은 SQL 관계형 데이터베이스 소스에서 추출된 데이터를 Neptune 데이터베이스 클러스터 대상으로 로드하는 방법을 지정합니다. 이러한 매핑 규칙의 형식은 규칙이 Apache TinkerPop Gremlin을 사용하여 속성-그래프 데이터를 로드하는지 또는 R2RML을 사용하여

RDF(Resource Description Framework) 데이터를 로드하는지 여부에 따라 다릅니다. 다음에서는 이러한 형식에 대한 정보와 확인할 위치를 찾을 수 있습니다.

콘솔 또는 DMS API를 사용하여 마이그레이션 작업을 생성할 때 이러한 매핑 규칙을 지정할 수 있습니다.

콘솔을 사용하면서 데이터베이스 마이그레이션 작업 생성 페이지의 그래프 매핑 규칙을 사용하여 이러한 그래프 매핑 규칙을 지정합니다. 그래프 매핑 규칙에서 제공된 편집기를 사용하여 매핑 규칙을 직접 입력하고 편집할 수 있습니다. 또는 적절한 그래프 매핑 형식으로 매핑 규칙이 포함된 파일을 찾아볼 수 있습니다.

API를 사용하여 CreateReplicationTask API 직접 호출의 TaskData 요청 파라미터를 사용하여 이러한 옵션을 지정합니다. 적절한 그래프 매핑 형식의 매핑 규칙이 포함된 파일의 경로로 TaskData를 설정합니다.

Gremlin을 사용하여 속성-그래프 데이터를 생성하기 위한 그래프 매핑 규칙

Gremlin을 사용하여 속성-그래프 데이터를 생성하고 소스 데이터에서 생성할 각 그래프 엔터티에 대한 매핑 규칙이 있는 JSON 객체를 지정합니다. 이 JSON의 형식은 특별히 Amazon Neptune 대량 로드를 위해 정의됩니다. 다음 템플릿에서는 이 객체의 각 규칙이 어떻게 표시되는지 보여줍니다.

```
{
  "rules": [
    {
      "rule_id": "(an identifier for this rule)",
      "rule_name": "(a name for this rule)",
      "table_name": "(the name of the table or view being loaded)",
      "vertex_definitions": [
        {
          "vertex_id_template": "{col1}",
          "vertex_label": "(the vertex to create)",
          "vertex_definition_id": "(an identifier for this vertex)",
          "vertex_properties": [
            {
              "property_name": "(name of the property)",
              "property_value_template": "{col2} or text",
              "property_value_type": "(data type of the property)"
            }
          ]
        }
      ]
    }
  ],
}
```

```

{
  "rule_id": "(an identifier for this rule)",
  "rule_name": "(a name for this rule)",
  "table_name": "(the name of the table or view being loaded)",
  "edge_definitions": [
    {
      "from_vertex": {
        "vertex_id_template": "{col1}",
        "vertex_definition_id": "(an identifier for the vertex
referenced above)"
      },
      "to_vertex": {
        "vertex_id_template": "{col3}",
        "vertex_definition_id": "(an identifier for the vertex
referenced above)"
      },
      "edge_id_template": {
        "label": "(the edge label to add)",
        "template": "{col1}_{col3}"
      },
      "edge_properties": [
        {
          "property_name": "(the property to add)",
          "property_value_template": "{col4} or text",
          "property_value_type": "(data type like String, int,
double)"
        }
      ]
    }
  ]
}

```

버텍스 레이블이 있으면 여기에 버텍스가 만들어지고 있음을 의미합니다. 없는 경우, 버텍스가 다른 소스에 의해 생성된다는 것을 의미하며 이 정의는 버텍스 속성만 추가하는 것입니다. 전체 관계형 데이터베이스 소스에 대한 매핑을 지정하는 데 필요한 만큼 버텍스 및 엣지 정의를 지정합니다.

employee 테이블에 대한 샘플 규칙은 다음과 같습니다.

```

{

```

```
"rules": [
  {
    "rule_id": "1",
    "rule_name": "vertex_mapping_rule_from_nodes",
    "table_name": "nodes",
    "vertex_definitions": [
      {
        "vertex_id_template": "{emp_id}",
        "vertex_label": "employee",
        "vertex_definition_id": "1",
        "vertex_properties": [
          {
            "property_name": "name",
            "property_value_template": "{emp_name}",
            "property_value_type": "String"
          }
        ]
      }
    ]
  },
  {
    "rule_id": "2",
    "rule_name": "edge_mapping_rule_from_emp",
    "table_name": "nodes",
    "edge_definitions": [
      {
        "from_vertex": {
          "vertex_id_template": "{emp_id}",
          "vertex_definition_id": "1"
        },
        "to_vertex": {
          "vertex_id_template": "{mgr_id}",
          "vertex_definition_id": "1"
        },
        "edge_id_template": {
          "label": "reportsTo",
          "template": "{emp_id}_{mgr_id}"
        },
        "edge_properties": [
          {
            "property_name": "team",
            "property_value_template": "{team}",
            "property_value_type": "String"
          }
        ]
      }
    ]
  }
]
```

```

    ]
  }
]
}

```

여기서, 버텍스 및 엣지 정의는 직원 ID(EmpID)의 employee 노드와 관리자 ID(managerId)의 employee 노드에서 보고 관계를 매핑합니다.

Gremlin JSON을 사용하여 그래프 매핑 규칙을 생성하는 방법에 관한 자세한 내용은 Amazon Neptune 사용 설명서의 [Gremlin 로드 데이터 형식](#)을 참조하세요.

RDF/SPARQL 데이터 생성을 위한 그래프 매핑 규칙

SPARQL을 사용하여 쿼리할 RDF 데이터를 로드하는 경우 R2RML로 그래프 매핑 규칙을 작성합니다. R2RML은 RDF에 관계형 데이터를 매핑하기 위한 표준 W3C 언어입니다. R2RML 파일에서 triples 맵(예: 다음 <#TriplesMap1>)은 논리 테이블의 각 행을 0개 이상의 RDF triples로 변환하는 규칙을 지정합니다. 주제 맵(예: 다음 rr:subjectMap)은 트리플 맵에 의해 생성된 RDF triples의 주제를 생성하기 위한 규칙을 지정합니다. 조건자-객체 맵(예: 다음 rr:predicateObjectMap)은 논리 테이블의 각 논리 테이블 행에 대해 하나 이상의 조건자-객체 쌍을 만드는 함수입니다.

nodes 테이블에 대한 간단한 예는 다음과 같습니다.

```

@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ex: <http://example.com/ns#>.

<#TriplesMap1>
  rr:logicalTable [ rr:tableName "nodes" ];
  rr:subjectMap [
    rr:template "http://data.example.com/employee/{id}";
    rr:class ex:Employee;
  ];
  rr:predicateObjectMap [
    rr:predicate ex:name;
    rr:objectMap [ rr:column "label" ];
  ]

```

앞의 예에서 매핑은 직원 테이블에서 매핑된 그래프 노드를 정의합니다.

Student 테이블에 대한 또 다른 간단한 예는 다음과 같습니다.

```

@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ex: <http://example.com/#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

<#TriplesMap2>
  rr:logicalTable [ rr:tableName "Student" ];
  rr:subjectMap [ rr:template "http://example.com/{ID}{Name}";
                  rr:class foaf:Person ];
  rr:predicateObjectMap [
    rr:predicate ex:id ;
    rr:objectMap [ rr:column "ID";
                   rr:datatype xsd:integer ]
  ];
  rr:predicateObjectMap [
    rr:predicate foaf:name ;
    rr:objectMap [ rr:column "Name" ]
  ].

```

앞의 예에서 매핑은 Student 테이블의 사람 사이의 친구 관계를 매핑하는 그래프 노드를 정의합니다.

SPARQL R2RML을 사용하여 그래프 매핑 규칙을 생성하는 방법에 관한 자세한 내용은 W3C 사양 [R2RML: RDB-RDF 매핑 언어](#)를 참조하세요.

대상으로서 Amazon Neptune으로 Gremlin 및 R2RML을 마이그레이션하기 위한 데이터 형식

AWS DMS 는 두 가지 방법 중 하나로 SQL 소스 엔드포인트에서 Neptune 대상으로 데이터 유형 매핑을 수행합니다. 사용하는 방법은 Neptune 데이터베이스를 로드하는 데 사용하는 그래프 매핑 형식에 따라 다릅니다.

- Apache TinkerPop Gremlin, 마이그레이션 데이터의 JSON 표현을 사용.
- W3C의 SPARQL, 마이그레이션 데이터의 R2RML 표현을 사용.

이러한 두 그래프 매핑 형식에 관한 자세한 내용은 [Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정](#) 섹션을 참조하세요.

다음에서는 각 형식에 대한 데이터 형식 매핑에 대한 설명을 찾을 수 있습니다.

SQL 소스를 Gremlin 대상 데이터 형식 매핑으로

다음 표에서는 SQL 소스에서 Gremlin 형식 대상으로의 데이터 형식 매핑을 보여줍니다.

AWS DMS 는 나열되지 않은 SQL 소스 데이터 유형을 Gremlin에 매핑합니다String.

SQL 소스 데이터 형식	Gremlin 대상 데이터 형식
NUMERIC(및 변형)	Double
DECIMAL	
TINYINT	Byte
SMALLINT	Short
INT, INTEGER	Int
BIGINT	Long
FLOAT	Float
DOUBLE PRECISION	
REAL	Double
BIT	Boolean
BOOLEAN	
DATE	Date
TIME	
TIMESTAMP	
CHARACTER (및 변형)	String

Neptune을 로드하기 위한 Gremlin 데이터 형식에 관한 자세한 내용은 Neptune 사용 설명서의 [Gremlin 데이터 형식](#)을 참조하세요.

SQL 소스를 R2RML(RDF) 대상 데이터 형식 매핑으로

다음 표에서는 SQL 소스에서 R2RML 형식 대상으로의 데이터 형식 매핑을 보여줍니다.

나열된 모든 RDF 데이터 형식은 대/소문자를 구분합니다. 단, RDF literal.는 나열되지 않은 SQL 소스 데이터 형식을 RDF 리터럴에 AWS DMS 매핑합니다.

RDF 리터럴은 다양한 리터럴 어휘 형태와 데이터 형식 중 하나입니다. 자세한 내용은 W3C 사양 리소스 설명 프레임워크 (RDF): 개념 및 추상 구문의 [RDF 리터럴](#)을 참조하세요.

SQL 소스 데이터 형식	R2RML(RDF) 대상 데이터 형식
BINARY(및 변형)	xsd:hexBinary
NUMERIC(및 변형)	xsd:decimal
DECIMAL	
TINYINT	
SMALLINT	
INT, INTEGER	
BIGINT	
FLOAT	xsd:double
DOUBLE PRECISION	
REAL	
BIT	xsd:boolean
BOOLEAN	
DATE	xsd:date
TIME	xsd:time
TIMESTAMP	xsd:dateTime

SQL 소스 데이터 형식	R2RML(RDF) 대상 데이터 형식
CHARACTER (및 변형)	RDF 리터럴

Neptune을 로드하기 위한 RDF 데이터 형식 및 SQL 소스 데이터 형식으로의 매핑에 관한 자세한 내용은 W3C 사양 R2RML: RDB에서 RDF로 매핑 언어의 [데이터 형식 변환](#)을 참조하세요.

Amazon Neptune을 대상으로 사용 시 제한 사항

Neptune을 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- AWS DMS 는 현재 Neptune 대상으로 마이그레이션하는 경우에만 전체 로드 작업을 지원합니다. Neptune 대상으로 CDC(데이터 캡처 변경) 마이그레이션은 지원되지 않습니다.
- 다음 예와 같이 마이그레이션 작업을 시작하기 전에 대상 Neptune 데이터베이스에서 모든 데이터를 수동으로 지워야 합니다.

그래프 내의 모든 데이터(버텍스 및 엣지)를 삭제하려면 다음 Gremlin 명령을 실행합니다.

```
gremlin> g.V().drop().iterate()
```

'customer' 레이블이 있는 버텍스를 삭제하려면 다음 Gremlin 명령을 실행합니다.

```
gremlin> g.V().hasLabel('customer').drop()
```

Note

대용량 데이터셋을 삭제하려면 약간의 시간이 걸릴 수 있습니다. 필요하다면 제한을 두어 `drop()`을 반복할 수도 있습니다(예: `limit(1000)`).

'rated' 레이블이 있는 엣지를 삭제하려면 다음 Gremlin 명령을 실행합니다.

```
gremlin> g.E().hasLabel('rated').drop()
```

Note

대용량 데이터셋을 삭제하려면 약간의 시간이 걸릴 수 있습니다. 필요하다면 제한을 두어 `drop()`을 반복할 수도 있습니다(예: `limit(1000)`).

- DMS API 연산 `DescribeTableStatistics`는 Neptune 그래프 데이터 구조의 속성으로 인해 주어진 테이블에 관한 부정확한 결과를 반환할 수 있습니다.

마이그레이션 중에는 각 소스 테이블을 AWS DMS 스캔하고 그래프 매핑을 사용하여 소스 데이터를 Neptune 그래프로 변환합니다. 변환된 데이터는 먼저 대상 엔드포인트에 지정된 S3 버킷 폴더에 저장됩니다. 소스가 검색되고 이 중간 S3 데이터가 성공적으로 생성되면 `DescribeTableStatistics`는 데이터가 Neptune 대상 데이터베이스에 성공적으로 로드되었다고 가정합니다. 그러나 이것이 항상 사실인 것은 아닙니다. 지정된 테이블에 대해 데이터가 올바르게 로드되었는지 확인하려면 해당 테이블에 대한 마이그레이션의 양측에서 `count()` 반환 값을 비교합니다.

다음 예제에서는 대상 Neptune 데이터베이스 그래프 'customer'에 레이블이 할당된 소스 데이터베이스에서 customer 테이블을 로드 AWS DMS 했습니다. 그러면 이 레이블이 대상 데이터베이스에 확실하게 기록됩니다. 이렇게 하려면 소스 데이터베이스에서 사용할 수 있는 customer 행 수를 작업이 완료된 후 Neptune 대상 데이터베이스에 로드된 'customer' 레이블이 지정된 행 수와 비교합니다.

SQL을 사용하여 소스 데이터베이스에서 사용 가능한 고객 행 수를 가져오려면 다음을 실행합니다.

```
select count(*) from customer;
```

Gremlin을 사용하여 대상 데이터베이스 그래프에 로드된 'customer' 레이블이 지정된 행 수를 얻으려면 다음을 실행하세요.

```
gremlin> g.V().hasLabel('customer').count()
```

- 현재 단일 테이블이 로드되지 않으면 전체 작업이 실패합니다. 관계형 데이터베이스 대상과는 달리 Neptune의 데이터는 고도로 연결되어 있으므로 대부분의 경우 작업을 다시 시작할 수 없습니다. 이러한 유형의 데이터 로드 실패로 인해 작업을 성공적으로 재개할 수 없는 경우, 로드하지 못한 테이블을 로드하는 새 작업을 생성합니다. 이 새 작업을 실행하기 전에 Neptune 대상에서 부분적으로 로드된 테이블을 수동으로 지웁니다.

Note

장애를 복구할 수 있는 경우(예: 네트워크 전송 오류) Neptune 대상으로의 마이그레이션에 실패한 작업을 재개할 수 있습니다.

- AWS DMS 는 R2RML에 대한 대부분의 표준을 지원합니다. 그러나 AWS DMS 는 역 표현식, 조인 및 뷰를 포함한 특정 R2RML 표준을 지원하지 않습니다. R2RML 뷰의 차선책은 소스 데이터베이스에 해당 사용자 지정 SQL 뷰를 만드는 것입니다. 마이그레이션 작업에서 테이블 매핑을 사용하여 뷰를 입력으로 선택합니다. 그런 다음, 테이블에 뷰를 매핑하면 R2RML이 이를 사용하여 그래프 데이터를 생성합니다.
- 지원되지 않는 SQL 데이터 형식을 사용하여 소스 데이터를 마이그레이션하면 결과 대상 데이터의 정밀도가 손실될 수 있습니다. 자세한 내용은 [대상으로서 Amazon Neptune으로 Gremlin 및 R2RML을 마이그레이션하기 위한 데이터 형식](#) 단원을 참조하십시오.
- AWS DMS 는 LOB 데이터를 Neptune 대상으로 마이그레이션하는 기능을 지원하지 않습니다.

Redis OSS를 대상으로 사용 AWS Database Migration Service

Redis OSS는 데이터베이스, 캐시 및 메시지 브로커로 사용되는 오픈 소스 인메모리 데이터 구조 저장소입니다. 데이터를 인메모리로 관리하면 읽기 또는 쓰기 작업의 소요 시간이 밀리초 미만으로 단축되며 초당 수억 건의 작업을 수행할 수 있습니다. Redis OSS는 인메모리 데이터 저장소로서 밀리초 미만의 응답 시간을 요구하는 가장 까다로운 애플리케이션을 지원합니다.

를 사용하면 가동 중지 시간을 최소화하면서 지원되는 소스 데이터베이스의 데이터를 대상 Redis OSS 데이터 스토어로 마이그레이션할 AWS DMS 수 있습니다. Redis OSS에 관한 추가 정보는 [Redis OSS 설명서](#)를 참조하세요.

온프레미스 Redis OSS 외에도 다음을 AWS Database Migration Service 지원합니다.

- 대상 데이터 저장소로 사용되는 [Amazon ElastiCache\(Redis OSS\)](#). ElastiCache(Redis OSS)는 Redis OSS 클라이언트와 연동하며 개방형 Redis OSS 데이터 유형을 사용하여 데이터를 저장합니다.
- 대상 데이터 저장소로 사용되는 [Amazon MemoryDB](#). MemoryDB는 Redis OSS와 호환되며 현재 사용 중인 모든 Redis OSS 데이터 구조, API 및 명령을 사용하여 애플리케이션을 구축할 수 있습니다.

Redis OSS를 대상으로 사용하는 방법에 대한 자세한 내용은 다음 섹션을 AWS DMS참조하세요.

주제

- [Redis OSS 클러스터의 대상으로 사용하기 위한 사전 조건 AWS DMS](#)
- [Redis를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service](#)
- [관계형 또는 비관계형 데이터베이스에서 Redis OSS 대상으로 데이터 마이그레이션](#)
- [대상으로서 Redis OSS에 대한 엔드포인트 설정 지정](#)

Redis OSS 클러스터의 대상으로 사용하기 위한 사전 조건 AWS DMS

DMS는 독립형 구성의 온프레미스 Redis OSS 대상을 지원하며 또는 데이터가 여러 노드에 걸쳐 자동으로 분할(샤딩)되는 Redis OSS 클러스터로 지원합니다. 샤딩은 데이터를 여러 서버 또는 노드에 분산된 '샤드'라는 작은 청크로 분할하는 프로세스입니다. 실제로 하나의 샤드는 전체 데이터세트의 하위 집합을 포함하며 전체 워크로드의 일부를 차지하는 데이터 파티션입니다.

Redis OSS는 키-값 NoSQL 데이터 저장소이므로 소스가 관계형 데이터베이스일 때 사용하는 Redis OSS 키 명명 규칙은 schema-name.table-name.primary key입니다. Redis OSS에서는 키와 값에 특수 문자 %가 포함되어서는 안 됩니다. 그렇지 않으면 DMS는 레코드를 건너뜁니다.

Note

ElastiCache(Redis OSS)를 대상으로 사용하는 경우, DMS는 클러스터 모드가 활성화된 구성만 지원합니다. ElastiCache(Redis OSS) 버전 6.x 또는 이후 버전을 사용하여 클러스터 모드가 활성화된 대상 데이터 저장소를 생성하는 방법에 관한 자세한 내용은 Amazon ElastiCache(Redis OSS) 사용 설명서의 [시작하기](#)를 참조하세요.

데이터베이스 마이그레이션을 시작하기 전에 다음 기준에 따라 Redis OSS 클러스터를 시작하세요.

- 클러스터에 샤드가 하나 이상 있습니다.
- ElastiCache(Redis OSS) 대상을 사용하는 경우, 클러스터에서 IAM 역할 기반 액세스 제어를 사용하지 않도록 하세요. 그 대신 Redis OSS 인증을 사용하여 사용자를 인증하세요.
- 다중 AZ(가용 영역)를 활성화합니다.
- 데이터베이스에서 마이그레이션할 데이터에 맞는 충분한 메모리가 클러스터에 있는지 확인하세요.
- 초기 마이그레이션 작업을 시작하기 전에 대상 Redis OSS 클러스터에서 모든 데이터가 삭제되었는지 확인하세요.

클러스터 구성을 생성하기 전에 데이터 마이그레이션에 대한 보안 요구 사항을 결정해야 합니다. DMS 는 암호화 구성과 상관없이 대상 복제 그룹으로의 마이그레이션을 지원합니다. 하지만 클러스터 구성 을 생성할 때만 암호화를 활성화하거나 비활성화할 수 있습니다.

Redis를의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

Redis OSS를 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- Redis OSS는 키-값 NoSQL 데이터 저장소이므로 소스가 관계형 데이터베이스일 때 사용하는 Redis OSS 키 명명 규칙은 `schema-name.table-name.primary-key`입니다.
- Redis OSS에서는 키-값에 특수 문자 %가 포함되어서는 안 됩니다. 그렇지 않으면 DMS는 레코드를 건너뛵니다.
- DMS는 % 문자가 포함된 행을 마이그레이션하지 않습니다.
- DMS는 필드 이름에 % 문자가 포함된 필드를 마이그레이션하지 않습니다.
- 전체 LOB 모드는 지원되지 않습니다.
- ElastiCache(Redis OSS)를 대상으로 사용할 때는 프라이빗 인증 기관(CA)이 지원되지 않습니다.
- AWS DMS 는 Redis를 대상 엔드포인트로 사용할 때 임베디드 '\0' 문자가 포함된 소스 데이터를 지원하지 않습니다. 포함된 '\0' 문자가 포함된 데이터는 첫 번째 '\0' 문자에서 잘립니다.

관계형 또는 비관계형 데이터베이스에서 Redis OSS 대상으로 데이터 마이그레이션

모든 소스 SQL 또는 NoSQL 데이터 저장소의 데이터를 Redis OSS 대상으로 직접 마이그레이션할 수 있습니다. Redis OSS 대상으로 마이그레이션을 설정하고 시작하는 방법은 DMS 콘솔 또는 API를 사용하는 전체 로드 및 데이터 캡처 변경(CDC) 마이그레이션과 유사합니다. Redis OSS 대상으로 데이터베이스 마이그레이션을 수행하려면 다음과 같이 하세요.

- 마이그레이션을 위한 모든 프로세스를 수행하는 복제 인스턴스를 생성합니다. 자세한 내용은 [복제 인스턴스 생성](#) 섹션을 참조하세요.
- 소스 엔드포인트를 지정합니다. 자세한 내용은 [소스 및 대상 엔드포인트 생성](#) 섹션을 참조하세요.
- 클러스터의 DNS 이름과 포트 번호를 찾습니다.
- SSL 연결을 확인하는 데 사용할 수 있는 인증서 번들을 다운로드합니다.
- 아래 설명에 따라 대상 엔드포인트를 지정합니다.
- 하나의 작업이나 작업 집합을 생성하여 사용할 테이블과 복제 프로세스를 정의합니다. 자세한 내용은 [작업 생성](#) 섹션을 참조하세요.
- 소스 데이터베이스의 데이터를 대상 클러스터로 마이그레이션합니다.

데이터베이스 마이그레이션은 다음의 두 가지 방법으로 시작할 수 있습니다.

1. AWS DMS 콘솔을 선택하고 각 단계를 수행할 수 있습니다.
2. AWS Command Line Interface ()를 사용할 수 있습니다AWS CLI. 에서 CLI를 사용하는 방법에 대한 자세한 내용은의 섹션을 AWS DMS참조하세요. [AWS CLI](#)[AWS DMS](#)

클러스터의 DNS 이름과 포트 번호를 찾으려면

- 다음 AWS CLI 명령을 사용하여 replication-group-id에 복제 그룹의 이름을 제공합니다.

```
aws elasticache describe-replication-groups --replication-group-id myreplgroup
```

여기서 출력에는 Address 속성에 있는 DNS 이름과 클러스터의 프라이머리 노드 Port 속성에 있는 포트 번호가 표시됩니다.

```
...
"ReadEndpoint": {
  "Port": 6379,
  "Address": "myreplgroup-
111.1abc1d.1111.uuu1.cache.example.com"
}
...
```

MemoryDB를 대상으로 사용하는 경우 다음 AWS CLI 명령을 사용하여 Redis OSS 클러스터에 엔드포인트 주소를 제공하세요.

```
aws memorydb describe-clusters --clusterid clusterid
```

SSL 연결을 확인하는 데 사용할 인증서 번들을 다운로드합니다.

- 명령줄에 다음 wget 명령을 입력합니다. Wget은 인터넷에서 파일을 다운로드하는 데 사용되는 무료 GNU 명령줄 유틸리티 도구입니다.

```
wget https://s3.aws-api-domain/rds-downloads/rds-combined-ca-bundle.pem
```

여기서는 지정된 S3 버킷과 해당 버킷이 제공하는 rds-combined-ca-bundle.pem 파일에 액세스하는 데 필요한 리전의 Amazon S3 도메인을 *aws-api-domain* 완료합니다. AWS rds-combined-ca-bundle

AWS DMS 콘솔을 사용하여 대상 엔드포인트를 생성하려면

이 엔드포인트는 이미 실행 중인 Redis OSS 대상에 사용됩니다.

- 콘솔의 탐색 창에서 엔드포인트를 선택하고 엔드포인트 생성을 선택합니다. 다음 표는 설정에 대한 설명입니다.

옵션	조치
[엔드포인트 유형]	대상 엔드포인트 유형을 선택합니다.
[엔드포인트 식별자]	엔드포인트 이름을 입력합니다. 예를 들어, 이름에 엔드포인트 유형(예: my-redis-target)을 포함하세요.
[Target Engine]	이 엔드포인트에서 연결하려는 데이터베이스 엔진 유형으로 Redis OSS를 선택합니다.
클러스터 이름	Redis OSS 클러스터의 DNS 이름을 입력합니다.
포트	포트에 Redis OSS 클러스터의 포트 번호를 입력합니다.
SSL 보안 프로토콜	일반 텍스트 또는 SSL 암호화를 선택합니다. 일반 텍스트 - 이 옵션은 엔드포인트와 데이터베이스 간의 트래픽에 대해 Transport Layer Security(TLS) 암호화를 제공하지 않습니다.

옵션	조치
	SSL 암호화 - 이 옵션을 선택하는 경우, SSL 인증 기관 (CA) 인증서 ARN을 입력하여 서버의 인증서를 확인하고 암호화된 연결을 설정합니다. 온프레미스 Redis OSS의 경우, DMS는 퍼블릭 및 프라이빗 인증 기관(CA)을 모두 지원합니다. ElastiCache(Redis OSS)의 경우, DMS는 퍼블릭 CA만 지원합니다.
인증 유형	Redis OSS 대상에 연결할 때 수행할 인증 유형을 선택합니다. 옵션에는 없음, 인증 역할 및 인증 토큰이 포함됩니다. 인증 역할을 선택하는 경우, 인증 사용자 이름과 인증 암호를 제공하세요. 인증 토큰을 선택하는 경우, 인증 암호만 제공하세요.
복제 인스턴스	[선택 사항] 연결을 테스트하려는 경우에만 이전에 복제 인스턴스 생성 페이지에 입력한 복제 인스턴스의 이름을 선택합니다.

엔드포인트에 대한 모든 정보를 모두 제공했다면 AWS DMS 는 데이터베이스 마이그레이션 중에 사용할 Redis OSS 대상 엔드포인트를 생성합니다.

마이그레이션 작업 생성 및 데이터베이스 마이그레이션 시작에 관한 자세한 내용은 [작업 생성](#)을 참조하세요.

대상으로서 Redis OSS에 대한 엔드포인트 설정 지정

대상 엔드포인트를 만들거나 수정하려면 콘솔이나 CreateEndpoint 또는 ModifyEndpoint API 작업을 사용할 수 있습니다.

AWS DMS 콘솔의 Redis OSS 대상의 경우 엔드포인트 생성 또는 엔드포인트 수정 콘솔 페이지에서 엔드포인트별 설정을 지정합니다.

CreateEndpoint 및 ModifyEndpoint API 작업을 사용할 때는 RedisSettings 옵션에 대한 요청 파라미터를 지정하세요. 다음 예에서는 AWS CLI를 사용하여 이 작업을 수행하는 방법을 보여줍니다.

```
aws dms create-endpoint --endpoint-identifier my-redis-target
--endpoint-type target --engine-name redis --redis-settings
'{"ServerName": "sample-test-sample.zz012zz.cluster.eee1.cache.bbbxxx.com", "Port": 6379, "AuthType": "auth-token",
  "SslSecurityProtocol": "ssl-encryption", "AuthPassword": "notanactualpassword"}'

{
  "Endpoint": {
    "EndpointIdentifier": "my-redis-target",
    "EndpointType": "TARGET",
    "EngineName": "redis",
    "EngineDisplayName": "Redis",
    "TransferFiles": false,
    "ReceiveTransferredFiles": false,
    "Status": "active",
    "KmsKeyId": "arn:aws:kms:us-east-1:999999999999:key/x-b188188x",
    "EndpointArn": "arn:aws:dms:us-east-1:555555555555:endpoint:ABCDEFGHIJKLMONOPQRSTUVWXYZ",
    "SslMode": "none",
    "RedisSettings": {
      "ServerName": "sample-test-sample.zz012zz.cluster.eee1.cache.bbbxxx.com",
      "Port": 6379,
      "SslSecurityProtocol": "ssl-encryption",
      "AuthType": "auth-token"
    }
  }
}
```

--redis-settings 파라미터는 다음과 같습니다.

- ServerName – (필수) 유형 string에서 데이터가 마이그레이션되고 동일한 VPC에 있는 Redis OSS 클러스터를 지정합니다.
- Port – (필수) number 유형에서 엔드포인트에 액세스하는 데 사용되는 포트 값입니다.
- SslSecurityProtocol - (선택) 유효값에는 plaintext 및 ssl-encryption이 포함됩니다. 기본값은 ssl-encryption입니다.

plaintext 옵션은 엔드포인트와 데이터베이스 간의 트래픽에 대해 Transport Layer Security(TLS) 암호화를 제공하지 않습니다.

암호화된 연결을 만드는 데 ssl-encryption을 사용합니다. ssl-encryption은 서버의 인증서를 확인하는 데 SSL 인증 기관(CA) ARN이 필요하지 않지만 SslCaCertificateArn 설정을 사용하여 선택적으로 식별할 수 있습니다. 인증 기관(CA) ARN이 없다면 DMS는 Amazon 루트 CA를 사용합니다.

온프레미스 Redis OSS 대상을 사용하는 경우, 퍼블릭 또는 프라이빗 인증 기관(CA)을 DMS로 가져오고 서버 인증 시 해당 ARN을 제공하는 데 SslCaCertificateArn을 사용할 수 있습니다. ElastiCache(Redis OSS)를 대상으로 사용할 때는 프라이빗 CA가 지원되지 않습니다.

- AuthType -(필수) Redis OSS에 연결할 때 수행할 인증 유형을 나타냅니다. 유효값에는 none, auth-token 및 auth-role이 있습니다.

auth-token 옵션에는 'AuthPassword'를 입력해야 하는 반면, auth-role 옵션에는 'AuthUserName'과 'AuthPassword'를 입력해야 합니다.

Babelfish를 대상으로 사용 AWS Database Migration Service

를 사용하여 Microsoft SQL Server 소스 데이터베이스의 데이터를 Babelfish 대상으로 마이그레이션할 수 있습니다 AWS Database Migration Service.

Babelfish for Aurora PostgreSQL은 Microsoft SQL Server 클라이언트에서 데이터베이스 연결을 수락할 수 있는 기능을 사용하여 Amazon Aurora PostgreSQL 호환 버전 데이터베이스를 확장합니다. 이렇게 하면 원래 SQL Server용으로 구축된 애플리케이션이 기존 마이그레이션에 비해 거의 코드 변경 없이, 또 데이터베이스 드라이버 변경 없이 Aurora PostgreSQL에서 직접 작동할 수 있습니다.

가 대상으로 AWS DMS 지원하는 Babelfish 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 대상 AWS DMS](#). Aurora PostgreSQL의 이전 Babelfish 버전은 Babelfish 엔드포인트를 사용하기 전에 업그레이드가 필요합니다.

Note

Aurora PostgreSQL 대상 엔드포인트는 데이터를 Babelfish로 마이그레이션하는 데 선호되는 방법입니다. 자세한 내용은 [대상으로 Babelfish for Aurora PostgreSQL 사용](#) 섹션을 참조하세요.

Babelfish를 데이터베이스 엔드포인트로 사용하는 방법에 대한 자세한 내용은 Aurora용 Amazon Aurora 사용 설명서에서 [Babelfish for Aurora PostgreSQL](#)을 참조하세요.

Babelfish를 대상으로 사용하기 위한 사전 조건 AWS DMS

가 올바른 데이터 형식과 테이블 메타데이터를 AWS DMS 사용하도록 하려면 데이터를 마이그레이션하기 전에 테이블을 생성해야 합니다. 마이그레이션을 실행하기 전에 대상에 테이블을 생성하지 않으면 잘못된 데이터 유형 및 권한으로 테이블을 생성할 AWS DMS 수 있습니다. 예를 들어,는 타임스탬프 열을 이진(8)으로 대신 AWS DMS 생성하고 예상 타임스탬프/행 버전 기능을 제공하지 않습니다.

마이그레이션 전에 테이블을 준비하고 생성하려면

1. 고유한 제약 조건, 프라이머리 키 또는 기본 제약 조건을 포함하는 create table DDL 문을 실행합니다.

뷰, 저장 프로시저, 함수 또는 트리거 같은 객체에 대한 DDL 문이나 외래 키 제약 조건을 포함하지 마세요. 이들은 소스 데이터베이스를 마이그레이션한 후에 적용할 수 있습니다.
2. 테이블의 ID 열, 계산된 열 또는 행 버전이나 타임스탬프 데이터 형식이 포함된 열을 식별합니다. 그런 다음 마이그레이션 작업을 실행할 때 알려진 문제를 처리하는 데 필요한 변환 규칙을 생성합니다. 자세한 내용은 [변환 규칙 및 작업](#) 섹션을 참조하세요.
3. Babelfish가 지원하지 않는 데이터 형식의 열을 식별합니다. 그런 다음 지원되는 데이터 형식을 사용하도록 대상 테이블의 영향을 받는 열을 변경하거나, 마이그레이션 작업 중에 해당 열을 제거하는 변환 규칙을 생성합니다. 자세한 내용은 [변환 규칙 및 작업](#) 섹션을 참조하세요.

다음 표에는 Babelfish에서 지원하지 않는 소스 데이터 형식과 사용할 해당 권장 대상 데이터 형식이 나열되어 있습니다.

소스 데이터 유형	권장 Babelfish 데이터 형식
HEIRARCHYID	NVARCHAR(250)
GEOMETRY	VARCHAR(최대)
GEOGRAPHY	VARCHAR(최대)

Aurora PostgreSQL Serverless V2 소스 데이터베이스의 Aurora 용량 단위(ACU) 수준을 설정하려면 최소 ACU 값을 설정하여 AWS DMS 마이그레이션 작업을 실행하기 전에 성능을 개선할 수 있습니다.

- Serverless v2 용량 설정 창에서 최소 ACU를 2 또는 Aurora DB 클러스터에 적합한 수준으로 설정합니다.

Aurora 용량 단위 설정에 대한 자세한 내용은 Amazon Aurora 사용 설명서의 [Aurora 클러스터의 Aurora Serverless v2 용량 범위 선택](#)을 참조하세요.

AWS DMS 마이그레이션 작업을 실행한 후 ACUs의 최소값을 Aurora PostgreSQL Serverless V2 소스 데이터베이스에 적합한 수준으로 재설정할 수 있습니다.

Babelfish를의 대상으로 사용할 때의 보안 요구 사항 AWS Database Migration Service

다음은 Babelfish 대상과 AWS DMS 함께를 사용하기 위한 보안 요구 사항을 설명합니다.

- 데이터베이스를 생성하는 데 사용된 관리자 사용자 이름(관리자 사용자).
- 충분한 SELECT, INSERT, UPDATE, REFERENCES, REFERENCES 권한을 가진 PSQL 로그인 및 사용자.

Babelfish를의 대상으로 사용하기 위한 사용자 권한 AWS DMS

Important

보안을 위해 데이터 마이그레이션에 사용되는 사용자 계정은 대상으로 사용하는 모든 Babelfish 데이터베이스에서 등록된 사용자여야 합니다.

Babelfish 대상 엔드포인트에는 AWS DMS 마이그레이션을 실행하기 위한 최소 사용자 권한이 필요합니다.

로그인 및 권한이 낮은 Transact-SQL(T-SQL) 사용자를 생성하려면

1. 서버에 연결할 때 사용할 로그인과 암호를 생성합니다.

```
CREATE LOGIN dms_user WITH PASSWORD = 'password';
GO
```

2. Babelfish 클러스터의 가상 데이터베이스를 생성합니다.

```
CREATE DATABASE my_database;
GO
```

3. 대상 데이터베이스의 T-SQL 사용자를 생성합니다.

```
USE my_database
GO
CREATE USER dms_user FOR LOGIN dms_user;
GO
```

4. Babelfish 데이터베이스의 각 테이블에 권한을 부여합니다.

```
GRANT SELECT, DELETE, INSERT, REFERENCES, UPDATE ON [dbo].[Categories] TO dms_user;
```

Babelfish를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

다음 제한 사항은 Babelfish 데이터베이스를 AWS DMS 대상으로 사용할 때 적용됩니다.

- 테이블 준비 모드 “Do Nothing”만 지원됩니다.
- ROWVERSION 데이터 형식에는 마이그레이션 작업 중에 테이블에서 열 이름을 제거하는 테이블 매핑 규칙이 필요합니다.
- sql_variant 데이터 형식은 지원되지 않습니다.
- 전체 LOB 모드가 지원됩니다. SQL Server를 소스 엔드포인트로 사용하려면 SQL Server 엔드포인트 연결 속성 설정 ForceFullLob=True을 설정해야 LOB를 대상 엔드포인트로 마이그레이션할 수 있습니다.
- 복제 작업 설정에는 다음과 같은 제한 사항이 있습니다.

```
{
  "FullLoadSettings": {
    "TargetTablePrepMode": "DO_NOTHING",
    "CreatePkAfterFullLoad": false,
  },
}
```

- Babelfish의 TIME(7), DATETIME2(7), DATETIMEOFFSET(7) 데이터 형식은 시간의 초 부분 정밀도를 6자리로 제한합니다. 이러한 데이터 형식을 사용할 때는 대상 테이블에 정밀도 값 6을 사용하는 것이 좋습니다. Babelfish 버전 2.2.0 이상에서 TIME(7) 및 DATETIME2(7)를 사용할 경우, 정밀도의 일곱 번째 자리는 항상 0입니다.

- DO_NOTHING 모드에서 DMS는 테이블이 이미 존재하는지 확인합니다. 대상 스키마에 테이블이 없으면 DMS는 소스 테이블 정의를 기반으로 테이블을 생성하고, 사용자 정의 데이터 형식을 기본 데이터 형식에 매핑합니다.
- Babelfish 대상으로의 AWS DMS 마이그레이션 작업은 ROWVERSION 또는 TIMESTAMP 데이터 형식을 사용하는 열이 있는 테이블을 지원하지 않습니다. 전송 프로세스 중에 테이블에서 열 이름을 제거하는 테이블 매핑 규칙을 사용할 수 있습니다. 다음 변환 규칙 예제에서 소스에 있는 Actor라는 테이블이 변환되어 대상의 Actor 테이블에서 col 문자로 시작하는 모든 열을 제거합니다.

```
{
  "rules": [{
    "rule-type": "selection",is
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "%"
    },
    "rule-action": "include"
  }, {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-action": "remove-column",
    "rule-target": "column",
    "object-locator": {
      "schema-name": "test",
      "table-name": "Actor",
      "column-name": "col%"
    }
  }
]}
```

- 대상 테이블이 Categories처럼 대소문자가 혼용된 이름을 사용하는 ID 열 또는 계산된 열이 있는 테이블의 경우, DMS 작업을 위해 테이블 이름을 소문자로 변환하는 변환 규칙 작업을 생성해야 합니다. 다음 예제에서는 AWS DMS 콘솔을 사용하여 변환 규칙 작업인 소문자 만들기를 생성하는 방법을 보여줍니다. 자세한 내용은 [변환 규칙 및 작업](#) 단원을 참조하십시오.

▼ Transformation rules

You can use transformation rules to change or transform schema, table or column names of some or all of the selected objects. [Info](#)

Add transformation rule

▼ where **schema name** is like 'dbo' and **table name** is like '%', convert-lowercase

Rule target

Table ▼

Source name

Enter a schema ▼

Source name

Use the % character as a wildcard

dbo

Table name

Use the % character as a wildcard

%

Action

Make lowercase ▼

- Babelfish 버전 2.2.0 이전에 DMS는 Babelfish 대상 엔드포인트에 복제할 수 있는 열 수를 20개로 제한했습니다. Babelfish 2.2.0에서는 이 한도가 100개로 늘어났습니다. 하지만 Babelfish 버전 2.4.0 이상에서는 복제할 수 있는 열 수가 다시 늘어납니다. SQL Server 데이터베이스에 대해 다음 코드 샘플을 실행하여 어떤 테이블이 너무 긴지 확인할 수 있습니다.

```
USE myDB;
GO
DECLARE @Babelfish_version_string_limit INT = 8000; -- Use 380 for Babelfish versions
before 2.2.0
WITH bfindpoint
AS (
SELECT
  [TABLE_SCHEMA]
    , [TABLE_NAME]
    , COUNT( [COLUMN_NAME] ) AS NumberColumns
    , ( SUM( LEN( [COLUMN_NAME] ) + 3 )
      + SUM( LEN( FORMAT(ORDINAL_POSITION, 'N0') ) + 3 )
```



```

    + LEN( TABLE_SCHEMA ) + 3
+ 12 -- INSERT INTO string
+ 12) AS InsertIntoCommandLength -- values string
    , CASE WHEN ( SUM( LEN( [COLUMN_NAME] ) + 3 )
+ SUM( LEN( FORMAT(ORDINAL_POSITION, 'N0') ) + 3 )
    + LEN( TABLE_SCHEMA ) + 3
+ 12 -- INSERT INTO string
+ 12) -- values string
    >= @Babelfish_version_string_limit
    THEN 1
    ELSE 0
    END AS IsTooLong
FROM [INFORMATION_SCHEMA].[COLUMNS]
GROUP BY [TABLE_SCHEMA], [TABLE_NAME]
)
SELECT *
FROM bfbendpoint
WHERE IsTooLong = 1
ORDER BY TABLE_SCHEMA, InsertIntoCommandLength DESC, TABLE_NAME
;

```

Babelfish의 대상 데이터 형식

다음 표에는를 사용할 때 지원되는 Babelfish 대상 데이터 형식 AWS DMS 과 AWS DMS 데이터 형식의 기본 매핑이 나와 있습니다.

AWS DMS 데이터 형식에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service에서 사용되는 데이터 형식](#).

AWS DMS 데이터 유형	Babelfish 데이터 형식
BOOLEAN	TINYINT
BYTES	VARBINARY(길이)
DATE	DATE
TIME	TIME
INT1	SMALLINT

AWS DMS 데이터 유형	Babelfish 데이터 형식
INT2	SMALLINT
INT4	INT
INT8	BIGINT
NUMERIC	NUMERIC(p,s)
REAL4	REAL
REAL8	FLOAT
STRING	열이 날짜 또는 시간 열인 경우, 다음 작업을 수행합니다. - SQL Server 2008 이상에서는 DATETIME2를 사용합니다. - 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에는 VARCHAR(37)을 사용합니다. 열이 날짜 또는 시간 열이 아닌 경우, VARCHAR(길이)를 사용합니다.
UINT1	TINYINT
UINT2	SMALLINT
UINT4	INT
UINT8	BIGINT
WSTRING	NVARCHAR(길이)

AWS DMS 데이터 유형	Babelfish 데이터 형식
BLOB	VARBINARY(최대) DMS에 이 데이터 형식을 사용하려면 특정 작업에서 BLOB 사용을 활성화해야 합니다. DMS는 프라이머리 키를 포함하는 테이블에서만 BLOB 데이터 형식을 지원합니다.
CLOB	VARCHAR(최대) DMS에 이 데이터 형식을 사용하려면 특정 작업에서 CLOB 사용을 활성화해야 합니다.
NCLOB	NVARCHAR(최대) DMS에 이 데이터 형식을 사용하려면 특정 작업에서 NCLOB 사용을 활성화해야 합니다. CDC 중에 DMS는 프라이머리 키를 포함하는 테이블에서만 NCLOB 데이터 형식을 지원합니다.

Amazon Timestream을의 대상으로 사용 AWS Database Migration Service

AWS Database Migration Service 를 사용하여 전체 로드 및 CDC 데이터 마이그레이션을 지원하여 소스 데이터베이스에서 Amazon Timestream 대상 엔드포인트로 데이터를 마이그레이션할 수 있습니다.

Amazon Timestream은 대량 데이터 수집을 위해 설계된 빠르고 확장 가능한 서버리스 시계열 데이터베이스 서비스입니다. 시계열 데이터는 일정 기간 동안 수집되는 일련의 데이터 포인트로, 시간이 지남에 따라 변화하는 이벤트를 측정하는 데 사용됩니다. 이를 사용하여 IoT 애플리케이션, DevOps 애플리케이션 및 분석 애플리케이션에서 지표를 수집, 저장, 분석할 수 있습니다. Timestream에 데이터를 저장하면 데이터의 추세와 패턴을 거의 실시간으로 시각화하고 식별할 수 있습니다. Amazon Timestream에 대한 자세한 내용은 Amazon Timestream 개발자 가이드에서 [Amazon Timestream이란 무엇인가요?](#)를 참조하세요.

주제

- [Amazon Timestream을의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service](#)
- [멀티스레드 전체 로드 작업 설정](#)
- [멀티스레드 CDC 로드 작업 설정](#)

- [Timestream을의 대상으로 사용할 때 엔드포인트 설정 AWS DMS](#)
- [Amazon Timestream 대상 엔드포인트 생성 및 수정](#)
- [객체 매핑을 사용하여 데이터를 Timestream 주제로 마이그레이션](#)
- [Amazon Timestream을의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service](#)

Amazon Timestream을의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service

Amazon Timestream을 대상으로 설정하기 전에 IAM 역할을 생성해야 AWS DMS합니다. 이 역할은 AWS DMS 가 Amazon Timestream으로 마이그레이션되는 데이터에 액세스할 수 있도록 허용해야 합니다. Timestream으로 마이그레이션하는 데 사용하는 역할에 대한 최소 액세스 권한 세트가 다음 IAM 정책에 나와 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDescribeEndpoints",
      "Effect": "Allow",
      "Action": [
        "timestream:DescribeEndpoints"
      ],
      "Resource": "*"
    },
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "timestream:ListTables",
        "timestream:DescribeDatabase"
      ],
      "Resource": "arn:aws:timestream:us-east-1:123456789012:database/DATABASE_NAME"
    },
    {
      "Sid": "VisualEditor1",
      "Effect": "Allow",
```

```

    "Action": [
      "timestream:DeleteTable",
      "timestream:WriteRecords",
      "timestream:UpdateTable",
      "timestream:CreateTable"
    ],
    "Resource": "arn:aws:timestream:us-east-1:123456789012:database/
DATABASE_NAME/table/TABLE_NAME"
  }
]
}

```

모든 테이블을 마이그레이션하려는 경우 위 예시에서 **TABLE_NAME**에 *를 사용합니다.

Timestream을 대상으로 사용할 때는 다음 지침을 참고하세요.

- 타임스탬프가 1년 전을 초과하는 기록 데이터를 수집하려는 경우 AWS DMS 를 사용하여 Amazon S3에 쉼표로 구분된 값(csv) 형식으로 데이터를 쓰는 것이 좋습니다. 그런 다음 Timestream의 배치 로드를 사용하여 데이터를 Timestream으로 수집합니다. 자세한 내용은 [Amazon Timestream 개발자 안내서](#)의 [Using batch load in Timestream](#)을 참조하세요.
- 1년 전 미만 데이터의 전체 로드 데이터 마이그레이션인 경우 Timestream 테이블의 메모리 스토어 보존 기간을 가장 오래된 타임스탬프보다 크거나 같게 설정하는 것이 좋습니다. 그런 다음 마이그레이션이 완료된 후 테이블의 메모리 스토어 보존 기간을 원하는 값으로 편집합니다. 예를 들어 가장 오래된 타임스탬프가 2개월 전인 데이터를 마이그레이션하려면 다음과 같이 합니다.
 - Timestream 대상 테이블의 메모리 스토어 보존 기간을 2개월로 설정합니다.
 - 를 사용하여 데이터 마이그레이션을 시작합니다 AWS DMS.
 - 데이터 마이그레이션이 완료되면 대상 Timestream 테이블의 보존 기간을 원하는 값으로 변경합니다.

마이그레이션 전에 다음 페이지의 정보를 사용하여 메모리 스토어 비용을 추정하는 것이 좋습니다.

- [Amazon Timestream 요금](#)
- [AWS 요금 계산기](#)
- CDC 데이터 마이그레이션인 경우 수집된 데이터가 메모리 스토어 보존 한도 내에 포함되도록 대상 테이블의 메모리 스토어 보존 기간을 설정하는 것이 좋습니다. 자세한 내용은 [Amazon Timestream 개발자 안내서](#)의 [Writes Best Practices](#)를 참조하세요.

멀티스레드 전체 로드 작업 설정

데이터 전송 속도를 높이기 위해서는 다음 작업 설정을 사용하여 Timestream 대상 엔드포인트로 멀티스레드 전체 로드 마이그레이션 작업을 AWS DMS 지원합니다.

- `MaxFullLoadSubTasks` - 병렬로 로드할 최대 소스 테이블 수를 표시하려면 이 옵션을 사용합니다. DMS는 전용 하위 작업을 사용하여 각 테이블을 해당 Amazon Timestream 대상 테이블에 로드합니다. 기본값은 8이며 최대값은 49입니다.
- `ParallelLoadThreads` - 이 옵션을 사용하여 각 테이블을 Amazon Timestream 대상 테이블에 로드하는 데 AWS DMS 사용하는 스레드 수를 지정합니다. Timestream 대상의 최대값은 32입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.
- `ParallelLoadBufferSize` - 이 옵션을 사용하여 병렬 로드 스레드에서 데이터를 Amazon Timestream 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 `ParallelLoadThreads`와 함께 사용하세요. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다.
- `ParallelLoadQueuesPerThread` - 각 동시 스레드가 액세스하는 대기열 수를 지정하여 대기열에서 데이터 레코드를 가져오고 대상에 대한 배치 로드를 생성하려면 이 옵션을 사용합니다. 기본값은 1입니다. 그러나 다양한 페이로드 크기의 Amazon Timestream 대상에서 스레드당 대기열 수의 유효 범위는 5~512입니다.

멀티스레드 CDC 로드 작업 설정

CDC 성능을 높이기 위해서는 다음 작업 설정을 AWS DMS 지원합니다.

- `ParallelApplyThreads` - CDC 로드 중에 데이터 레코드를 Timestream 대상 엔드포인트로 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- `ParallelApplyBufferSize` - CDC 로드 중에 Timestream 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.
- `ParallelApplyQueuesPerThread` - 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Timestream 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다. 기본값은 1이고 최대값은 512입니다.

Timestream을의 대상으로 사용할 때 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 마찬가지로 엔드포인트 설정을 사용하여 Timestream 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 `--timestream-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Timestream을 대상으로 하여 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

이름	설명
MemoryDuration	Timestream의 메모리 스토어에 마이그레이션된 데이터를 저장할 보존 한도를 지정하려면 이 속성을 설정합니다. 시간은 시간 단위로 측정됩니다. Timestream의 메모리 스토어는 높은 수집 처리량과 빠른 액세스에 최적화되어 있습니다. 기본값: 24(시간) 유효한 값: 1~8,736(시간 단위로 측정된 1시간~12개월) 예시: <code>--timestream-settings '{"MemoryDuration": 20}'</code>
DatabaseName	대상 Timestream 데이터베이스 이름을 지정하려면 이 속성을 설정합니다. 유형: 문자열 예시: <code>--timestream-settings '{"DatabaseName": "db_name"}</code>
TableName	대상 Timestream 테이블 이름을 지정하려면 이 속성을 설정합니다. 유형: 문자열 예시: <code>--timestream-settings '{"TableName": "table_name"}</code>

이름	설명
MagneticDuration	Timestream 테이블에 적용되는 마그네틱 스토어 저장 기간을 일 단위로 지정하려면 이 속성을 설정합니다. 수집된 데이터의 보존 한도입니다. Timestream은 보존 한도를 초과하는 모든 타임스탬프를 삭제합니다. 자세한 내용은 Amazon Timestream 개발자 안내서의 Storage를 참조하세요. 예시: <code>--timestream-settings '{"MagneticDuration": "3"}'</code>
CdcInsertsAndUpdates	이 속성을 로 설정 <code>true</code> 하여가 삽입 및 업데이트 AWS DMS 만 적용하고 삭제는 적용하지 않도록 지정합니다. Timestream은 레코드 삭제를 허용하지 않으므로 이 값이 이면는 레코드를 삭제하는 대신 Timestream 데이터베이스에서 해당 레코드를 <code>false</code> AWS DMS Null아웃합니다. 자세한 내용은 제한 사항 섹션을 참조하세요. 기본값: <code>false</code> 예시: <code>--timestream-settings '{"CdcInsertsAndUpdates": "true"}'</code>
EnableMagneticStoreWrites	마그네틱 스토어 쓰기를 활성화하려면 이 속성을 <code>true</code>로 설정합니다. 이 값이 인 경우 Timestream은 기본적으로 마그네틱 스토어 <code>false</code> AWS DMS 쓰기를 허용하지 않으므로 타임스탬프가 대상 테이블의 메모리 스토어 보존 기간보다 오래된 레코드를 쓰지 않습니다. 자세한 내용은 Amazon Timestream 개발자 안내서의 Writes Best Practices를 참조하세요. 기본값: <code>false</code> 예시: <code>--timestream-settings '{"EnableMagneticStoreWrites": "true"}'</code>

Amazon Timestream 대상 엔드포인트 생성 및 수정

IAM 역할을 생성하고 최소 액세스 권한 세트를 설정한 후에는 AWS DMS 콘솔을 사용하거나의 `create-endpoint` 명령을 사용하여 `--timestream-settings '{"EndpointSetting": "value", ...}'` JSON 구문 [AWS CLI](#)으로 Amazon Timestream 대상 엔드포인트를 생성할 수 있습니다.

다음 예시에서는 AWS CLI를 사용하여 Timestream 대상 엔드포인트를 생성하고 수정하는 방법을 보여줍니다.

Timestream 대상 엔드포인트 생성 명령

```
aws dms create-endpoint --endpoint-identifier timestream-target-demo
--endpoint-type target --engine-name timestream
--service-access-role-arn arn:aws:iam::123456789012:role/my-role
--timestream-settings
{
  "MemoryDuration": 20,
  "DatabaseName": "db_name",
  "MagneticDuration": 3,
  "CdcInsertsAndUpdates": true,
  "EnableMagneticStoreWrites": true,
}
```

Timestream 대상 엔드포인트 수정 명령

```
aws dms modify-endpoint --endpoint-identifier timestream-target-demo
--endpoint-type target --engine-name timestream
--service-access-role-arn arn:aws:iam::123456789012:role/my-role
--timestream-settings
{
  "MemoryDuration": 20,
  "MagneticDuration": 3,
}
```

객체 매핑을 사용하여 데이터를 Timestream 주제로 마이그레이션

AWS DMS 는 테이블 매핑 규칙을 사용하여 소스의 데이터를 대상 Timestream 주제로 매핑합니다. 데이터를 대상 주제에 매핑하려면 객체 매핑이라는 테이블 매핑 규칙 유형을 사용합니다. 객체 매핑을 사용하여 원본의 데이터 레코드를 Timestream 주제에 게시된 데이터 레코드로 매핑하는 방법을 지정합니다.

Timestream 주제에는 파티션 키 외에 별다른 사전 설정 구조가 없습니다.

Note

객체 매핑을 사용할 필요는 없습니다. 일반 테이블 매핑을 다양한 변환에 사용할 수 있습니다. 하지만 파티션 키 유형은 다음과 같은 기본 동작을 따릅니다.

- 프라이머리 키는 전체 로드의 파티션 키로 사용됩니다.
- parallel-apply 작업 설정을 사용하지 않는 경우, CDC의 파티션 키로 `schema.table`이 사용됩니다.
- parallel-apply 작업 설정을 사용하지 않는 경우, CDC의 파티션 키로 프라이머리 키가 사용됩니다.

object-mapping 규칙을 만들려면 rule-type을 object-mapping으로 지정합니다. 이 규칙은 사용할 객체 매핑의 유형을 지정합니다. 이 규칙의 구조는 다음과 같습니다.

```
{
  "rules": [
    {
      "rule-type": "object-mapping",
      "rule-id": "id",
      "rule-name": "name",
      "rule-action": "valid object-mapping rule action",
      "object-locator": {
        "schema-name": "case-sensitive schema name",
        "table-name": ""
      }
    }
  ]
}
```

```
{
  "rules": [
    {
      "rule-type": "object-mapping",
      "rule-id": "1",
      "rule-name": "timestream-map",
      "rule-action": "map-record-to-record",

```

```

        "target-table-name": "tablename",
        "object-locator": {
            "schema-name": "",
            "table-name": ""
        },
        "mapping-parameters": {
            "timestream-dimensions": [
                "column_name1",
                "column_name2"
            ],
            "timestream-timestamp-name": "time_column_name",
            "timestream-multi-measure-name": "column_name1or2",
            "timestream-hash-measure-name": true or false,
            "timestream-memory-duration": x,
            "timestream-magnetic-duration": y
        }
    }
}

```

AWS DMS 는 현재 rule-action 파라미터 `map-record-to-document`에 유효한 유일한 값으로 `map-record-to-record` 및 `map-record-to-document` 값을 `exclude-columns` 속성 목록의 일부로 제외되지 않은 레코드에 대해 기본적으로 AWS DMS 수행하는 작업을 지정합니다. 이러한 값은 어떤 식으로든 속성 매핑에 영향을 미치지 않습니다.

관계형 데이터베이스에서 Timestream 주제로 마이그레이션할 때 `map-record-to-record`를 사용합니다. 이러한 규칙 유형은 Timestream 주제의 파티션 키로 관계형 데이터베이스의 `taskResourceId.schemaName.tableName` 값을 사용하며 소스 데이터베이스의 각 열마다 속성을 하나씩 생성합니다. 를 사용할 때 `exclude-columns` 속성 목록에 나열되지 않은 소스 테이블의 열에 `map-record-to-record`대해서는 대상 주제에 해당 속성을 AWS DMS 생성합니다. 이러한 해당 속성은 원본 열이 속성 매핑에 사용되는지 여부와 상관없이 생성됩니다.

`map-record-to-record`를 이해하는 한 가지 방법은 작업 중일 때 관찰하는 것입니다. 이 예에서는 다음 구조와 데이터를 사용하여 관계형 데이터베이스 테이블 행에서 시작한다고 가정합니다.

FirstName	LastName	StoreId	HomeAddress	HomePhone	WorkAddress	WorkPhone	DateofBirth
Randy	Marsh	5	221B Baker Street	123456789 0	31 Spooner Street, Quahog	987654321 0	1988/02/2 9

이 정보를 Test라는 스키마에서 Timestream 주제로 마이그레이션하려면 데이터를 대상 주제로 매핑하는 규칙을 생성합니다. 다음 규칙은 그 매핑 과정을 보여 줍니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "rule-action": "include",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      }
    },
    {
      "rule-type": "object-mapping",
      "rule-id": "2",
      "rule-name": "DefaultMapToTimestream",
      "rule-action": "map-record-to-record",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Customers"
      }
    }
  ]
}
```

다음은 지정된 Timestream 주제와 파티션 키(이 경우 `taskResourceId.schemaName.tableName`)로 Timestream 대상 주제의 샘플 데이터를 사용하여 결과 레코드 형식을 보여줍니다.

```
{
  "FirstName": "Randy",
  "LastName": "Marsh",
  "StoreId": "5",
  "HomeAddress": "221B Baker Street",
  "HomePhone": "1234567890",
  "WorkAddress": "31 Spooner Street, Quahog",
  "WorkPhone": "9876543210",
  "DateOfBirth": "02/29/1988"
}
```

Amazon Timestream을의 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

Amazon Timestream을 대상으로 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- **차원 및 타임스탬프:** Timestream은 소스 데이터의 차원 및 타임스탬프를 복합 프라이머리 키처럼 사용하며 이러한 값을 업서트하는 것도 허용하지 않습니다. 즉, 소스 데이터베이스에서 레코드의 타임스탬프 또는 차원을 변경하면 Timestream 데이터베이스가 새 레코드를 생성하려고 합니다. 따라서 다른 기존 레코드의 차원 또는 타임스탬프와 일치하도록 레코드의 차원 또는 타임스탬프를 변경하면가 새 레코드를 생성하거나 이전 해당 레코드를 AWS DMS 업데이트하는 대신 다른 레코드의 값을 업데이트할 수 있습니다.
- **DDL 명령:**의 현재 릴리스는 CREATE TABLE 및 DROP TABLE DDL 명령 AWS DMS 만 지원합니다.
- **레코드 제한:** Timestream에는 레코드 크기 및 측정값 크기와 같은 레코드에 대한 제한이 있습니다. 자세한 내용은 [Amazon Timestream 개발자 안내서](#)의 [할당량](#)을 참조하세요.
- **레코드 및 Null 값 삭제:** Timestream은 레코드 삭제를 지원하지 않습니다. 소스에서 삭제된 레코드 마이그레이션을 지원하기 위해 Timestream 대상 데이터베이스의 레코드에서 해당 필드를 AWS DMS 지웁니다. 숫자 필드의 경우 0, 텍스트 필드의 경우 null, 부울 필드의 경우 false로 해당 대상 레코드의 필드 값을 AWS DMS 변경합니다.
- **대상으로 사용 시** Timestream은 관계형 데이터베이스(RDBMS)가 아닌 소스를 지원하지 않습니다.
- **AWS DMS 는 다음 리전에서만** Timestream을 대상으로 지원합니다.
 - 미국 동부(버지니아 북부)
 - 미국 동부(오하이오)
 - 미국 서부(오리곤)
 - 유럽(아일랜드)
 - 유럽(프랑크푸르트)

- 아시아 태평양(시드니)
- 아시아 태평양(도쿄)
- 대상으로 사용 시 Timestream은 TargetTablePrepMode를 TRUNCATE_BEFORE_LOAD로 설정하는 것을 지원하지 않습니다. 이 설정에는 DROP_AND_CREATE를 사용하는 것이 좋습니다.

Amazon RDS for Db2 및 IBM Db2 LUW를의 대상으로 사용 AWS DMS

AWS Database Migration Service ()를 사용하여 Db2 LUW 데이터베이스에서 Amazon RDS for Db2 또는 온프레미스 Db2 데이터베이스로 데이터를 마이그레이션할 수 있습니다AWS DMS.

가 대상으로 AWS DMS 지원하는 Db2 LUW 버전에 대한 자세한 내용은 섹션을 참조하세요 [의 대상 AWS DMS](#).

Secure Sockets Layer(SSL)를 사용하여 Db2 LUW 엔드포인트와 복제 인스턴스 간 연결을 암호화할 수 있습니다. Db2 LUW 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 [에서 SSL 사용 AWS Database Migration Service](#) 섹션을 참조하세요.

Db2 LUW를의 대상으로 사용할 때의 제한 사항 AWS DMS

Db2 LUW 데이터베이스를 대상으로 사용할 때는 다음과 같은 제한 사항이 적용됩니다 AWS DMS. Db2 LUW를 소스로 사용 시 적용되는 제한 사항은 [Db2 LUW를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요.

- AWS DMS 는 소스가 z/OS용 Db2 LUW 또는 Db2인 경우에만 Db2 LUW를 대상으로 지원합니다.
- Db2 LUW를 대상으로 사용하는 경우 전체 LOB 모드에서의 복제가 지원되지 않습니다.
- Db2 LUW를 대상으로 사용하는 경우 전체 로드 단계에서 XML 데이터 유형이 지원되지 않습니다. 이는 IBM dbload 유틸리티의 제한 사항입니다. 자세한 내용은 IBM Informix 서버 설명서의 [dbload 유틸리티](#)를 참조하세요.
- AWS DMS 는 큰따옴표 문자(")에 해당하는 값으로 BLOB 필드를 잘라냅니다. 이는 IBM dbload 유틸리티의 제한 사항입니다.
- AWS DMS 는 DMS 버전 3.5.3에서 Db2 LUW 대상으로 마이그레이션할 때 병렬 전체 로드 옵션을 지원하지 않습니다. 이 옵션은 DMS 버전 3.5.4 이상에서만 사용할 수 있습니다.

Db2 LUW를의 대상으로 사용 시 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Db2 LUW 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나의 create-endpoint 명령을 --ibm-db2-

settings '{"EndpointSetting": "value", ...}' JSON 구문과 [AWS CLI](#) 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Db2 LUW를 대상으로 하여 사용할 수 있는 엔드포인트 설정은 다음 표에 나와 있습니다.

이름	설명
KeepCsvFiles	true인 경우 데이터를 복제하는 데 사용된 Db2 LUW 대상에 .csv 파일을 AWS DMS 저장합니다. DMS는 이러한 파일을 분석 및 문제 해결에 사용합니다.
LoadTimeout	Db2 대상에서 DMS가 수행한 AWS DMS 시간 초과 작업 이전의 시간(밀리초)입니다. 기본값은 1,200(20분)입니다.
MaxFileSize	데이터를 Db2 LUW로 전송할 때 사용되는 .csv 파일의 최대 크기(KB 단위)를 지정합니다.
WriteBufferSize	DMS 복제 인스턴스의 로컬 디스크에 .csv 파일을 생성할 때 사용되는 메모리 내 파일 쓰기 버퍼의 크기(KB 단위)입니다. 기본값은 1,024(1MB)입니다.

에 대한 VPC 엔드포인트 구성 AWS DMS

AWS DMS 는 Amazon Virtual Private Cloud(VPC) 엔드포인트를 소스 및 대상으로 지원합니다. AWS DMS 는 AWS 이러한 소스 및 대상 데이터베이스에 대한 명시적으로 정의된 경로가 VPC에 정의되어 있는 한 Amazon VPC 엔드포인트를 사용하여 모든 소스 또는 대상 데이터베이스에 연결할 수 있습니다 AWS DMS .

Amazon VPC 엔드포인트 AWS DMS 를 지원하면 추가 네트워킹 구성 및 설정 없이 모든 복제 작업에 대해 end-to-end 네트워크 보안을 더 쉽게 유지할 수 있습니다. 모든 소스 및 대상 엔드포인트에 VPC 엔드포인트를 사용하면 모든 트래픽을 VPC 내에서 유지하고 제어할 수 있습니다.

프라이빗 서브넷 또는 AWS DMS 서버리스 AWS DMS 복제에서 생성된 복제 인스턴스의 경우 AWS 관리형 데이터베이스에 연결하려면 Amazon VPC 엔드포인트를 설정해야 합니다.

- Amazon S3
- Amazon DynamoDB
- Amazon Kinesis

- Amazon Redshift
- Amazon OpenSearch Service

AWS Secrets Manager를 사용하여 DMS가 사용할 연결 자격 증명을 저장하는 경우 VPC 엔드포인트도 설정해야 합니다.

AWS DMS 버전 3.4.7부터 프라이빗 네트워크를 사용할 때 DMS 복제 인스턴스 또는 서버리스 복제와 위의 Amazon 서비스 간에 연결을 설정하려면 VPC 엔드포인트가 필요합니다.

일반적인 AWS DMS 사전 조건

VPC 엔드포인트를 구성하기 전에 다음 사전 조건을 충족해야 합니다.

- AWS DMS 복제 인스턴스 또는 서버리스 복제에 사용할 VPC를 찾거나 AWS DMS 생성합니다. 이 정보를 제공하지 않으면 DMS는 설정된 리전에서 기본 VPC를 사용하려고 시도합니다.
- VPC 엔드포인트를 생성할 수 있는 IAM 권한이 없습니다. Amazon S3 및 Amazon DynamoDB에 연결하려면, 게이트웨이 VPC 엔드포인트를 사용하면 VPC용 인터넷 게이트웨이 또는 NAT 디바이스가 없어도 안정적으로 연결할 수 있습니다. 게이트웨이 엔드포인트는 다른 유형의 VPC 엔드포인트와 달리 AWS PrivateLink를 사용하지 않습니다. 자세한 내용은 AWS PrivateLink 사용 설명서의 [게이트웨이 엔드포인트](#)를 참조하세요.
- IAM 사용자 권한을 구성하여 DMS 사용:
 - dms-vpc-role 역할을 구성합니다. 자세한 내용은 [AWS 관리형 정책: AmazonDMSVPCManagementRole](#)를 참조하세요.
 - dms-cloudwatch-logs-role 역할을 구성합니다. 자세한 내용은 [AWS 관리형 정책: AmazonDMSCloudWatchLogsRole](#)를 참조하세요.
 - AWS DMS Serverless를 사용하려면 계정에 서비스 연결 역할(SLR)이 있어야 합니다. 이 역할의 생성 및 사용을 AWS DMS 관리합니다. 필요한 SLR이 있는지 확인하는 방법에 대한 자세한 내용은 [AWS DMS 서버리스에 대한 서비스 링크 역할](#) 섹션을 참조하세요. 복제를 생성할 때 AWS DMS 서버리스는 프로그래밍 방식으로 서버리스 서비스 연결 역할을 생성합니다. 이 역할은 IAM 콘솔에서 볼 수 있습니다.

AWS Secrets Manager를 사용하여 Amazon VPC 엔드포인트 설정

AWS Secrets Manager에서 사용할 Amazon VPC 엔드포인트를 설정할 수 있습니다 AWS DMS. 이 엔드포인트를 생성하면 프라이빗 서브넷에서 AWS DMS 복제 인스턴스 또는 서버리스 복제 구성을 활성화

화하여 퍼블릭 인터넷 액세스 없이 Secrets Manager에 저장된 데이터베이스 자격 증명에 안전하게 액세스할 수 있습니다.

사전 조건

AWS Secrets Manager를 사용하여 VPC 엔드포인트를 구성 AWS DMS하기 전에 다음 사전 조건을 충족해야 합니다.

- 모든 [the section called “일반적인 AWS DMS 사전 조건”](#)를 구성해야 합니다.
- 연결하려는 [소스](#) 또는 [대상](#) 데이터베이스를 생성하고 구성합니다.
- 보안 인증 정보를 사용하여 AWS Secrets Manager에서 보안 암호를 생성하여 소스 및 대상 데이터베이스에 액세스합니다. 보안 암호는 AWS DMS 복제 인스턴스 또는 AWS DMS 서버리스 복제와 동일한 리전에 있어야 합니다. 데이터베이스 유형에 따라 보안 암호의 스키마가 다를 수 있습니다. 자세한 내용은 [AWS DMS 엔드포인트 작업을 참조하세요](#).

Important

AWS DMS 복제 인스턴스 및 AWS DMS 서버리스 복제는 Amazon RDS에서 관리하는 보안 암호에서 작동하지 않습니다. 이러한 자격 증명에는에서 연결을 설정하는 AWS DMS 데 필요한 호스트 및 포트 정보가 포함되지 않습니다.

- Amazon S3, Amazon Kinesis, Amazon DynamoDB, Amazon Redshift, Amazon OpenSearch Service, Amazon Neptune 및 Amazon Timestream과 같은 일부 데이터베이스에는 DMS 엔드포인트를 관리하기 위한 IAM 권한을 구성해야 합니다. 자세한 내용은 [AWS DMS 엔드포인트 작업을 참조하세요](#).

AWS Secrets Manager용 VPC 엔드포인트 생성

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/vpc/> Amazon VPC 콘솔을 엽니다.
2. VPC 콘솔 메뉴 모음에서 복제 인스턴스 AWS 리전 와 동일한 AWS DMS 를 선택합니다.
3. VPC 탐색 창에서 엔드포인트를 선택합니다.
4. 엔드포인트에서 엔드포인트 생성을 선택합니다.
5. 다음과 같이 VPC 엔드포인트를 구성합니다.
 - a. AWS 서비스로 유형을 선택합니다.

- b. 서비스 이름 텍스트 상자에서 **secretsmanager** 검색 후 `com.amazonaws.[region].secretsmanager`를 선택합니다. 선택한 서비스의 유형이 인터페이스인지 확인합니다.
- c. 네트워크 설정에서 DMS 복제 인스턴스와 동일한 리전에서 실행 중이거나 서버리스 복제를 생성한 VPC를 선택합니다.
- d. 서브넷 섹션에서 DMS를 작동할 서브넷을 선택합니다. 프라이빗 서브넷만 선택해야 합니다. 서브넷 ID로 프라이빗 서브넷을 식별할 수 있습니다. 예를 들어 `vpc-xxxxxx-subnet-private1-us-west-2a`입니다.

퍼블릭 액세스 없이 DMS 복제 인스턴스를 생성하는 경우 복제 인스턴스가 있는 프라이빗 서브넷과 연결된 라우팅 테이블을 선택해야 합니다.

Note

프라이빗 서브넷은 DMS 복제 서브넷 그룹을 생성할 때 제공해야 하므로 기록해 두어야 합니다. VPC 엔드포인트를 사용하여 DMS를 AWS Secrets Manager와 연결하려면 VPC 엔드포인트에 지정된 서브넷이 DMS 복제 서브넷 그룹의 서브넷과 동일해야 합니다.

- e. 원하는 보안 그룹을 선택합니다. 보안 그룹 규칙은 VPC의 리소스로부터 엔드포인트 네트워크 인터페이스로의 트래픽을 제어합니다. 보안 그룹을 지정하지 않은 경우 기본 설정된 보안 그룹이 사용됩니다.
6. 정책에서 전체 액세스를 선택합니다. 사용자 지정 정책을 사용하여 자체 액세스 제어를 지정하려면 사용자 지정을 선택합니다. `dms-vpc-role` JSON 정책 문서를 준수하는 신뢰 정책을 사용합니다. 자세한 내용은 [AWS DMS사용을 위해 IAM 역할 생성](#)을 참조하세요.
7. 엔드포인트 생성을 선택합니다.

상태가 Available가 될 때까지 기다려야 합니다. 이제 VPC 엔드포인트에 `vpce-xxxx`로 시작하는 ID가 있습니다.

이제 VPC 엔드포인트를 성공적으로 생성했습니다. AWS DMS 엔드포인트, DMS 서브넷 그룹을 구성해야 합니다. 선택한 마이그레이션 옵션에 따라 DMS 복제 인스턴스 또는 서버리스 복제를 구성합니다.

Amazon S3를 사용하여 Amazon VPC 엔드포인트 설정

Amazon S3에서 사용할 Amazon VPC 엔드포인트를 설정할 수 있습니다 AWS DMS. 이 엔드포인트를 생성하면 퍼블릭 인터넷 액세스 없이 프라이빗 서브넷의 AWS DMS 복제 인스턴스 또는 서버리스 복제 구성을 활성화하여 S3 버킷에 저장된 데이터베이스 자격 증명에 안전하게 액세스할 수 있습니다.

사전 조건

에서 Amazon S3로 VPC 엔드포인트를 구성 AWS DMS하기 전에 다음 사전 조건을 충족해야 합니다.

- 모든 [the section called “일반적인 AWS DMS 사전 조건”](#)를 구성해야 합니다.
- [소스](#) 또는 [대상](#) 데이터베이스로 사용할 Amazon S3 버킷을 생성합니다 AWS DMS. S3의 버전 관리를 활성화하지 마세요. S3 버전 관리가 필요한 경우, 수명 주기 정책을 사용하여 이전 버전을 적극적으로 삭제하세요. 그렇지 않으면 S3 list-object 직접 호출 시간 초과로 인해 엔드포인트 테스트 연결 실패가 발생할 수 있습니다.
- DMS Amazon S3 엔드포인트를 관리하도록 IAM 권한을 구성합니다. AWS DMS 콘솔을 사용하는 경우 IAM 역할을 생성할 권한이 있는 경우 필요한 권한이 있는 IAM 역할을 생성할 수 있습니다.

Amazon S3용 VPC 엔드포인트 생성

- 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/vpc/> Amazon VPC 콘솔을 엽니다.
- VPC 콘솔 메뉴 모음에서 복제 인스턴스 AWS 리전 와 동일한 AWS DMS 를 선택합니다.
- VPC 탐색 창에서 엔드포인트를 선택합니다.
- 엔드포인트에서 엔드포인트 생성을 선택합니다.
- 다음과 같이 VPC 엔드포인트를 구성합니다.
 - AWS 서비스로 유형을 선택합니다.
 - 서비스 이름 텍스트 상자에서 **s3** 검색 후 com.amazonaws.[region].s3를 선택합니다. 선택한 서비스의 유형이 게이트웨이인지 확인합니다. Amazon S3 및 DynamoDB에 연결할 때 게이트웨이 VPC 엔드포인트를 생성할 수 있습니다. 게이트웨이 엔드포인트는 다른 유형의 VPC 엔드포인트와 달리 AWS PrivateLink를 사용하지 않습니다.
 - 네트워크 설정에서 DMS 복제 인스턴스와 동일한 리전에서 실행 중이거나 서버리스 복제를 생성한 VPC를 선택합니다.

- d. 서브넷 섹션에서 DMS를 작동할 서브넷을 선택합니다. 프라이빗 서브넷만 선택해야 합니다. 서브넷 ID로 프라이빗 서브넷을 식별할 수 있습니다. 예를 들어 `vpc-xxxxxx-subnet-private1-us-west-2a`입니다.

 Note

퍼블릭 액세스 없이 DMS 복제 인스턴스를 생성한 경우 DMS 인스턴스와 동일한 리전에 있는 프라이빗 서브넷과 연결된 라우팅 테이블을 선택해야 합니다. 프라이빗 서브넷은 DMS 복제 서브넷 그룹을 생성할 때 제공해야 하므로 기록해 두어야 합니다. VPC 엔드포인트를 사용하여 DMS를 Amazon S3와 연결하려면 VPC 엔드포인트에 지정된 서브넷이 DMS 복제 서브넷 그룹의 서브넷과 동일해야 합니다.

6. 정책에서 전체 액세스를 선택합니다. 사용자 지정 정책을 사용하여 자체 액세스 제어를 지정하려면 사용자 지정을 선택합니다. `dms-vpc-role` JSON 정책 문서를 준수하는 신뢰 정책을 사용합니다. 자세한 내용은 [AWS DMS사용을 위해 IAM 역할 생성](#)을 참조하세요.
7. 엔드포인트 생성을 선택합니다.

상태가 Available가 될 때까지 기다려야 합니다. 이제 VPC 엔드포인트에 `vpce-xxxx`로 시작하는 ID가 있습니다.

이제 VPC 엔드포인트를 성공적으로 생성했습니다. AWS DMS 엔드포인트, DMS 서브넷 그룹을 구성해야 합니다. 선택한 마이그레이션 옵션에 따라 DMS 복제 인스턴스 또는 서버리스 복제를 구성합니다.

Amazon DynamoDB용 Amazon VPC 엔드포인트 설정

프라이빗 서브넷 또는 AWS DMS 서버리스 AWS DMS 복제에서 복제 인스턴스를 사용하는 경우 VPC 엔드포인트를 생성하여 Amazon DynamoDB와의 보안 연결을 설정해야 합니다. VPC 엔드포인트 구성 없으면 연결 오류가 AWS DMS 발생합니다.

VPC 엔드포인트를 생성할 때 DMS 콘솔에서 엔드포인트 유형을 게이트웨이 또는 인터페이스로 선택해야 합니다. 자세한 내용은 다음을 참조하세요.

- [the section called “일반적인 AWS DMS 사전 조건”](#)
- [Amazon DynamoDB용 게이트웨이 엔드포인트](#)
- [게이트웨이 Amazon VPC 엔드포인트의 연결 문제를 해결하려면 어떻게 해야 하나?](#)

Amazon Kinesis용 Amazon VPC 엔드포인트 설정

프라이빗 서브넷 또는 AWS DMS 서버리스 AWS DMS 복제에서 복제 인스턴스를 사용하는 경우 VPC 엔드포인트를 생성하여 Amazon Kinesis와의 보안 연결을 설정해야 합니다. VPC 엔드포인트 구성이 없으면 연결 오류가 AWS DMS 발생합니다. 자세한 내용은 다음을 참조하세요.

- [the section called “일반적인 AWS DMS 사전 조건”](#)
- [Amazon Kinesis Data Streams를의 대상으로 사용 AWS Database Migration Service](#)

Amazon Redshift용 Amazon VPC 엔드포인트 설정

프라이빗 서브넷 또는 AWS DMS 서버리스 AWS DMS 복제에서 복제 인스턴스를 사용하는 경우 VPC 엔드포인트를 생성하여 Amazon Redshift와의 보안 연결을 설정해야 합니다. VPC 엔드포인트 구성이 없으면 연결 오류가 AWS DMS 발생합니다. 자세한 내용은 다음을 참조하세요.

- [the section called “일반적인 AWS DMS 사전 조건”](#)
- [Redshift 관리형 VPC 엔드포인트](#)

Amazon OpenSearch Service용 Amazon VPC 엔드포인트 설정

프라이빗 서브넷 또는 AWS DMS 서버리스 AWS DMS 복제에서 복제 인스턴스를 사용하는 경우 VPC 엔드포인트를 생성하여 Amazon OpenSearch Service와의 보안 연결을 설정해야 합니다. VPC 엔드포인트 구성이 없으면 연결 오류가 AWS DMS 발생합니다. 자세한 내용은 다음을 참조하세요.

- [the section called “일반적인 AWS DMS 사전 조건”](#)
- [Amazon OpenSearch Ingestion 파이프라인에 대한 VPC 액세스 구성](#)

복제 인스턴스, DMS 서브넷 그룹 및 DMS 엔드포인트 설정

VPC 엔드포인트를 생성한 후 AWS DMS 복제 리소스를 구성해야 합니다. 네트워크 격리를 위한 복제 서브넷 그룹, 처리를 위한 복제 인스턴스 또는 서버리스 복제, 소스 및 대상 데이터베이스에 연결하기 위한 엔드포인트를 설정하여 VPC 내에서 안전한 데이터베이스 마이그레이션을 활성화할 수 있습니다.

AWS DMS 복제 인스턴스 설정

AWS DMS 프로비저닝된 복제 인스턴스를 구성하려면 DMS 복제 서브넷 그룹을 설정해야 합니다.

DMS 복제 서브넷 그룹 생성

1. 에 로그인 AWS Management Console 하고 DMS 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 서브넷 그룹을 열고 서브넷 그룹 생성을 선택합니다.
3. 이름 및 설명을 입력합니다.
4. VPC 드롭다운 메뉴에서 DMS 복제 인스턴스를 생성할 리전과 동일한 리전에서 실행 중인 VPC를 선택합니다.
5. 서브넷 추가 드롭다운 메뉴에서 VPC 엔드포인트를 생성할 때 지정한 프라이빗 서브넷을 추가합니다. 서브넷 ID로 프라이빗 서브넷을 식별할 수 있습니다. 예를 들어 vpc-xxxxxx-subnet-private1-us-west-2a입니다.
6. 서브넷 그룹 생성을 클릭합니다.

DMS 복제 인스턴스 생성(프로비저닝됨)

1. AWS Management Console 로 이동하여 복제 인스턴스를 생성합니다. 자세한 정보는 [복제 인스턴스 생성](#) 섹션을 참조하세요. 복제 인스턴스 선택, 크기 조정 및 구성을 자세히 알아보려면 [AWS DMS 복제 인스턴스 작업을 참조하세요](#).
2. 연결 및 보안 섹션에서 AWS DMS 복제 인스턴스를 생성하려는 IPv4용 가상 프라이빗 클라우드 (VPC) 또는 듀얼 스택 모드에서 VPC를 선택합니다. 자세한 내용은 [복제 인스턴스용 네트워크 설정](#)을 참조하세요.
3. 복제 서브넷 그룹 드롭다운 메뉴에서 복제 인스턴스에 대해 생성한 서브넷 그룹을 선택합니다.

Note

VPC 엔드포인트에 지정된 서브넷이 DMS 복제 인스턴스 서브넷 그룹의 서브넷과 동일한지 확인합니다. VPC 엔드포인트와 연결되지 않은 서브넷 그룹에서 서브넷을 모두 제거해야 합니다.

4. 퍼블릭 액세스를 비활성화하려면 퍼블릭 액세스 가능 확인란의 선택을 취소합니다.
5. 고급 설정 섹션의 VPC 보안 그룹 드롭다운 메뉴에서 복제 인스턴스와 연결된 모든 VPC 서브넷 그룹을 선택합니다. 이러한 그룹에는 VPC 엔드포인트를 생성할 때 지정한 서브넷이 포함된 서브넷 그룹이 포함되어야 합니다.

서브넷 그룹을 지정하지 않으면 DMS가 기본 복제 서브넷 그룹을 선택하거나 없는 경우 생성합니다. 자세한 내용은 [AWS DMS에 대한 보안 그룹 구성](#) 섹션을 참조하세요.

6. 복제 인스턴스 구성을 완료하고 복제 인스턴스 생성을 선택합니다.

상태가 Available가 될 때까지 기다려야 합니다.

AWS DMS 소스 및 대상 엔드포인트 생성

1. DMS 콘솔에 로그인합니다.
2. AWS DMS 엔드포인트로 이동하여 엔드포인트 생성을 선택합니다.
3. [소스](#) 및 [대상](#) 엔드포인트를 생성 및 구성합니다.
4. DMS 콘솔에서 기존 IAM 역할을 선택하거나 새 IAM 역할을 생성하여 AWS Secrets Manager에 저장된 데이터베이스 자격 증명에 액세스할 수 있습니다.
5. 테스트 실행을 클릭하여 DMS 복제 인스턴스에서 엔드포인트 연결을 테스트합니다. 복제 인스턴스를 사용하여 테스트를 실행하려면 복제 인스턴스의 Available 상태가 여야 합니다.
6. 엔드포인트 생성을 선택합니다.

AWS DMS 서버리스 복제 설정

AWS DMS 서버리스 복제를 구성하려면 DMS 복제 서브넷 그룹을 설정해야 합니다.

AWS DMS 소스 및 대상 엔드포인트 생성

1. DMS 콘솔에 로그인합니다.
2. AWS DMS 엔드포인트로 이동하여 엔드포인트 생성을 선택합니다.
3. [소스](#) 및 [대상](#) 엔드포인트를 생성 및 구성합니다.
4. DMS 콘솔에서 기존 IAM 역할을 선택하거나 새 IAM 역할을 생성하여 AWS Secrets Manager에 저장된 데이터베이스 자격 증명에 액세스할 수 있습니다.

Note

AWS DMS 서버리스 복제의 경우 DMS 엔드포인트에 대한 연결을 테스트하거나 TestConnection API를 사용할 수 없습니다. 연결 테스트는 DMS 인스턴스와 소스/대상 데이터베이스 간의 서버리스 복제 시작 중에 수행됩니다. 자세한 내용은 [AWS DMS 서버리스 구성 요소](#) 섹션을 참조하세요.

5. 엔드포인트 생성을 선택합니다.

DMS 복제 서브넷 그룹 생성

1. 에 로그인 AWS Management Console 하고 DMS 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 서브넷 그룹을 열고 서브넷 그룹 생성을 선택합니다.
3. 이름 및 설명을 입력합니다.
4. VPC 드롭다운 메뉴에서 DMS 서버리스 인스턴스와 동일한 리전에서 실행 중인 VPC를 선택합니다.
5. 서브넷 추가 드롭다운 메뉴에서 VPC 엔드포인트를 생성할 때 지정한 프라이빗 서브넷을 추가합니다. 서브넷 ID로 프라이빗 서브넷을 식별할 수 있습니다. 예를 들어 `vpc-xxxxxx-subnet-private1-us-west-2a`입니다.
6. 서브넷 그룹 생성을 클릭합니다.

DMS 서버리스 애플리케이션 생성

1. DMS 콘솔로 이동하여 서버리스 인스턴스를 생성합니다. 자세한 내용은 [서버리스 복제 생성](#)을 참조하세요. 서버리스 인스턴스의 선택, 크기 조정 및 구성을 자세히 알아보려면 [AWS DMS 서버리스 작업을](#) 참조하세요.
2. 연결 및 보안 섹션의 Virtual Private Cloud(VPC) 드롭다운 메뉴에서 AWS DMS 서버리스 인스턴스를 생성할 VPC를 선택합니다. 자세한 내용은 [복제 인스턴스용 네트워크 설정](#)을 참조하세요.
3. 서브넷 그룹 드롭다운 메뉴에서 서버리스 인스턴스에 대해 생성한 서브넷 그룹을 선택합니다.

Note

VPC 엔드포인트에 지정된 서브넷이 DMS 서버리스 인스턴스 서브넷 그룹의 서브넷과 동일한지 확인합니다. 서버리스 인스턴스와 연결되지 않은 서브넷 그룹에서 서브넷을 모두 제거해야 합니다.

4. 가용 영역을 선택합니다.
5. 최대 DMS 용량 단위(DCU) 드롭다운 메뉴에서 원하는 DCU 용량을 선택합니다.
6. 작업생성을 선택합니다. 그러면 작업 목록에 상태 `Start required`로 표시되는 DMS 서버리스 복제 구성이 생성됩니다.
7. 서버리스 복제를 시작하려면 작업을 선택하고 작업 메뉴에서 시작을 선택합니다.

AWS DMS 버전 3.4.7 이상으로 마이그레이션할 때 영향을 받는 사람은 누구입니까?

앞서 나열한 AWS DMS 엔드포인트 중 하나 이상을 사용 중이고 이러한 엔드포인트가 공개적으로 라우팅 불가능하거나 VPC 엔드포인트가 기존에 연결되어 있지 않은 경우, 사용자가 영향을 받습니다.

AWS DMS 버전 3.4.7 이상으로 마이그레이션할 때 영향을 받지 않는 사람은 누구입니까?

다음과 같은 경우에는 영향을 받지 않습니다.

- 이전에 나열된 AWS DMS 엔드포인트 중 하나 이상을 사용하고 있지 않습니다.
- 앞서 나열한 엔드포인트 중 하나를 사용하고 있으며, 해당 엔드포인트가 공개적으로 라우팅 가능한 경우.
- 앞서 나열한 엔드포인트 중 하나를 사용하고 있으며, 해당 엔드포인트에 VPC 엔드포인트가 연결된 경우.

AWS DMS 버전 3.4.7 이상으로 마이그레이션 준비

앞서 설명한 엔드포인트를 사용할 때 AWS DMS 작업 실패를 방지하려면 AWS DMS를 버전 3.4.7 이상으로 업그레이드하기 전에 다음 단계 중 하나를 수행합니다.

- 영향을 받는 AWS DMS 엔드포인트를 공개적으로 라우팅 가능하도록 설정합니다. 예를 들어 AWS DMS 복제 인스턴스에서 이미 사용하고 있는 VPC에 Internet Gateway(IGW) 경로를 추가하여 모든 소스 및 대상 엔드포인트를 공개적으로 라우팅할 수 있도록 합니다.
- 아래에 설명된 대로, AWS DMS에서 사용하는 모든 소스 및 대상 엔드포인트에 액세스할 수 있는 VPC 엔드포인트를 생성합니다.

AWS DMS 소스 및 대상 엔드포인트에 사용하는 기존 VPC 엔드포인트의 경우 XML 정책 문서를 준수하는 신뢰 정책을 사용해야 합니다 `dms-vpc-role`. 이 XML 정책 문서에 대한 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.

또는 복제 인스턴스가 포함된 VPC에 VPC 엔드포인트를 추가하여 복제 인스턴스를 VPC 엔드포인트로 구성합니다. 퍼블릭 엔드포인트 없이 복제 인스턴스를 구성한 경우, 공개적으로 액세스 가능한 VPC 엔드포인트를 복제 인스턴스가 포함된 VPC에 추가하면 공개적으로 액세스가 가능해집니다. 복제 인스턴스를 VPC 엔드포인트와 명확히 연결하기 위해 추가 작업을 수행하지 않아도 됩니다.

Note

서비스마다 고유한 VPC 엔드포인트 구성이 존재할 수 있습니다. 예를 들어, AWS Secrets Manager를 사용할 경우 일반적으로 라우팅 테이블을 조정할 필요가 없습니다. 항상 각 서비스의 특정한 요구 사항을 확인하세요.

AWS DMS 복제 인스턴스에 대한 VPC 엔드포인트 구성에 대한 자세한 내용은 [섹션을 참조하세요](#) [데이터베이스 마이그레이션을 위한 네트워크 구성](#). 일반적으로 AWS 서비스에 액세스하기 위한 인터페이스 VPC 엔드포인트를 생성하는 방법에 대한 자세한 내용은 AWS PrivateLink 가이드의 [인터페이스 VPC 엔드포인트를 사용하여 AWS 서비스 액세스](#)를 참조하세요. VPC 엔드포인트의 AWS DMS 리전 가용성에 대한 자세한 내용은 [AWS 리전 표](#)를 참조하세요.

에서 지원하는 DDL 문 AWS DMS

데이터 마이그레이션 프로세스 중에 원본 데이터베이스에서 데이터 정의 언어(DDL) 문을 실행할 수 있습니다. 이 문은 복제 서버를 통해 대상 데이터베이스로 복제됩니다.

지원되는 DDL 문은 다음과 같습니다.

- 테이블 생성
- 테이블 삭제
- 테이블 이름 바꾸기
- 테이블 자르기
- 열 추가
- 열 삭제
- 열 이름 바꾸기
- 열 데이터 유형 변경

DMS는 일부 소스 엔진 유형에서 지원되는 모든 DDL 문을 캡처하지 않습니다. 또한 DMS는 DDL 문을 특정 대상 엔진에 적용할 때 DDL 문을 다르게 처리합니다. 특정 소스에 지원되는 DDL 문 및 이러한 문이 대상에 어떻게 적용되는지에 대한 자세한 내용은 해당 소스 및 대상 엔드포인트에 대한 설명서의 관련 항목을 참조하세요.

태스크 설정을 사용하여 변경 데이터 캡처(CDC) 중에 DMS가 DDL 동작을 처리하는 방식을 구성할 수 있습니다. 자세한 내용은 [변경 처리 DDL을 다루기 위한 작업 설정](#) 섹션을 참조하세요.

제한 사항 및 고려 사항

소스 데이터베이스의 빠른 DDL 작업 시퀀스(예: DDL>DML>DDL)로 인해 AWS DMS 가 로그를 잘못 구문 분석하여 데이터 손실 또는 예기치 않은 동작으로 이어질 수 있습니다. 데이터 일관성을 유지하려면 후속 작업을 수행하기 전에 AWS DMS 가 대상에 각 변경 사항을 적용할 때까지 기다립니다.

예를 들어 변경 데이터 캡처(CDC) 중에 소스 테이블에서 여러 개의 빠른 테이블 이름 변경 작업이 오류를 트리거할 수 있습니다. 테이블 이름을 바꾼 다음 다시 원래 이름으로 빠르게 바꾸는 경우 테이블이 대상 데이터베이스에 이미 있다고 보고할 수 AWS DMS 있습니다.

고급 엔드포인트 구성

AWS Database Migration Service (AWS DMS)에서 엔드포인트에 대한 고급 설정을 구성하여 마이그레이션 프로세스 중에 소스 및 대상 엔드포인트가 작동하는 방식을 제어할 수 있습니다. 고급 설정의 일부로 AWS DMS VPC 피어링을 구성하여 VPCs, 인바운드 및 아웃바운드 트래픽을 제어하는 DMS 보안 그룹, 추가 보안 계층으로서의 Newtwork 액세스 제어 목록(NACLs) 및 AWS Secrets Manager용 VPC 엔드포인트 간의 보안 통신을 활성화할 수 있습니다.

엔드포인트 생성 중에 이러한 구성을 설정하거나 나중에 AWS DMS 콘솔 또는 API를 통해 수정하여 특정 데이터베이스 엔진 요구 사항 및 성능 요구 사항에 따라 마이그레이션 프로세스를 미세 조정할 수 있습니다.

아래에서 고급 엔드포인트 구성에 대한 자세한 정보를 확인할 수 있습니다.

주제

- [에 대한 VPC 피어링 구성입니다 AWS DMS.](#)
- [에 대한 보안 그룹 구성 AWS DMS](#)
- [에 대한 네트워크 액세스 제어 목록\(NACL\) 구성 AWS DMS](#)
- [AWS DMS 보안 암호 관리자 VPC 엔드포인트 구성](#)
- [추가 고려 사항](#)

에 대한 VPC 피어링 구성입니다 AWS DMS.

VPC 피어링을 사용하면 두 VPC 간의 프라이빗 네트워크 연결이 가능하므로 AWS DMS 복제 인스턴스와 데이터베이스 엔드포인트가 동일한 네트워크에 있는 것처럼 서로 다른 VPC 간에 통신할 수 있습니다. 이는 소스 또는 대상 데이터베이스가 별도의 VPC에 있는 동안 DMS 복제 인스턴스가 하나의

VPC에 상주할 때 중요하므로 퍼블릭 인터넷을 통과하지 않고도 직접 보안 데이터 마이그레이션이 가능합니다.

Amazon RDS를 사용하는 경우 인스턴스가 다른 VPC에 있는 경우 DMS와 RDS 간에 VPC 피어링을 구성해야 합니다.

다음 절차를 수행해야 합니다.

VPC 피어링 연결 생성

1. [Amazon VPC 콘솔](#)로 이동하세요.
 2. 탐색 창의 가상 프라이빗 클라우드에서 피어링 연결을 선택합니다.
 3. 피어링 연결 생성을 클릭합니다.
 4. 피어링 연결을 구성합니다.
 - 이름 태그(선택 사항): 피어링 연결의 이름을 입력합니다(예: DMS-RDS-Peering).
- VPC 요청자: DMS 인스턴스가 포함된 VPC를 선택합니다.
- VPC 수락자: RDS 인스턴스 인스턴스가 포함된 VPC를 선택합니다.

Note

수락자 VPC가 다른 AWS 계정과 연결된 경우 해당 계정의 계정 ID와 VPC ID가 있어야 합니다.

5. 피어링 연결 생성을 클릭합니다.

VPC 피어링 연결 허용

1. 피어링 연결 목록에서 수락 보류 중 상태의 새 피어링 연결을 찾습니다.
2. 적절한 피어링 연결을 선택하고 작업을 클릭한 다음 요청 수락을 선택합니다.

피어링 연결의 상태가 활성화로 변경됩니다.

라우팅 테이블 업데이트

VPC 간의 트래픽을 활성화하려면 두 VPC 모두에서 라우팅 테이블을 업데이트해야 합니다. DMS VPC에서 라우팅 테이블을 업데이트하려면

1. RDS VPC의 CIDR 블록을 식별합니다.
 - a. VPC로 이동하여 RDS VPC를 선택합니다.
 - b. CIDR 탭에서 IPv4 CIDR 값을 복사합니다.
2. 리소스 맵을 사용하여 관련 DMS 라우팅 테이블을 식별합니다.
 - a. VPC로 이동하여 DMS VPC를 선택합니다.
 - b. 리소스 맵 탭을 클릭하고 DMS 인스턴스가 있는 서브넷과 연결된 라우팅 테이블을 기록해 둡니다.
3. DMS VPC의 모든 라우팅 테이블을 업데이트합니다.
 - a. [Amazon VPC 콘솔](#)에서 라우팅 테이블로 이동합니다.
 - b. DMS VPC에 대해 식별되는 라우팅 테이블을 선택합니다. VPC의 리소스 맵 탭에서 열 수 있습니다.
 - c. 경로 편집을 클릭합니다.
 - d. 라우팅 추가를 클릭하고 다음 정보를 입력합니다.
 - 대상: RDS VPC의 IPv4 CIDR 블록을 입력합니다(예: 10.1.0.0/16).
 - 대상: 피어링 구성 ID를 선택합니다(예: pcx-1234567890abcdef).
 - e. 경로 저장을 클릭합니다.

VPC 경로는 DMS VPC에 대해 저장됩니다. RDS VPC에 대해 동일한 단계를 수행합니다.

보안 그룹 업데이트

1. DMS 인스턴스 보안 그룹을 확인합니다.
 - 아웃바운드 규칙이 RDS 인스턴스로의 트래픽을 허용하는지 확인해야 합니다.
 - 유형: 사용자 지정 TCP 또는 특정 데이터베이스 포트(예: 3306 for MySQL).
 - 대상: RDS VPC의 CIDR 블록 또는 RDS 인스턴스의 보안 그룹입니다.
2. RDS 인스턴스 보안 그룹을 확인합니다.
 - 인바운드 규칙이 DMS 인스턴스의 트래픽을 허용하는지 확인해야 합니다.
 - 유형: 특정 데이터베이스 포트입니다.
 - 소스: DMS VPC의 CIDR 블록 또는 RDS 인스턴스의 보안 그룹입니다.

Note

다음 작업도 수행해야 합니다.

- 활성 피어링 연결: 계속하기 전에 VPC 피어링 연결이 활성 상태인지 확인합니다.
- 리소스 맵: [Amazon VPC 콘솔](#)의 리소스 맵 탭을 사용하여 업데이트가 필요한 라우팅 테이블을 식별합니다.
- 겹치는 CIDR 블록 없음: VPC에는 겹치지 않는 CIDR 블록이 있어야 합니다.
- 보안 모범 사례: 필요한 포트 및 소스에 대한 보안 그룹 규칙을 삭제합니다.

자세한 내용은 Amazon Virtual Private Cloud 사용 설명서의 [VPC 피어링 연결](#)을 참조하세요.

에 대한 보안 그룹 구성 AWS DMS

의 보안 그룹은 적절한 데이터베이스 포트에서 복제 인스턴스에 대한 인바운드 및 아웃바운드 연결을 허용 AWS DMS 해야 합니다. Amazon RDS를 사용하는 경우 인스턴스에 대해 DMS와 RDS 간에 보안 그룹을 구성해야 합니다.

다음 절차를 수행해야 합니다.

RDS 인스턴스 보안 그룹 구성

1. [Amazon VPC 콘솔](#)로 이동하세요.
2. 보안 아래 왼쪽 탐색 창에서 보안 그룹을 선택합니다.
3. RDS 인스턴스에 연결된 RDS 보안 그룹 중 하나를 선택합니다.
4. 인바운드 규칙 편집:
 - a. 작업을 클릭하고 인바운드 규칙 편집을 선택합니다.
 - b. 규칙 추가를 클릭하여 새 규칙을 생성합니다.
 - c. 다음과 같이 규칙을 구성하세요.
 - 유형: 데이터베이스 유형을 선택합니다(예: 포트 3306의 경우 MySQL/Aurora, 포트 5432의 경우 PostgreSQL).
 - 프로토콜: 데이터베이스 유형에 따라 자동으로 채워집니다.
 - 포트 범위: 데이터베이스 유형에 따라 자동으로 채워집니다.

- 소스: 사용자 지정을 선택하고 DMS 인스턴스와 연결된 보안 그룹 ID를 붙여 넣습니다. 이렇게 하면 해당 보안 그룹 내 모든 리소스의 트래픽이 허용됩니다. DMS 인스턴스의 IP 범위(CIDR 블록)를 지정할 수도 있습니다.

d. 규칙 저장을 클릭합니다.

DMS 복제 인스턴스 보안 그룹 구성

1. [Amazon VPC 콘솔](#)로 이동하세요.
2. 보안 아래 왼쪽 탐색 창에서 보안 그룹을 선택합니다.
3. 보안 그룹 목록에서 DMS 복제 인스턴스와 연결된 보안 그룹을 찾아 선택합니다.
4. 아웃바운드 규칙을 편집합니다.
 - a. 작업을 클릭하고 아웃바운드 규칙 편집을 선택합니다.
 - b. 규칙 추가를 클릭하여 새 규칙을 생성합니다.
 - c. 다음과 같이 규칙을 구성하세요.
 - 유형: 데이터베이스 유형을 선택합니다(예: MySQL/Aurora, PostgreSQL).
 - 프로토콜: 데이터베이스 유형에 따라 자동으로 채워집니다.
 - 포트 범위: 데이터베이스 유형에 따라 자동으로 채워집니다.
 - 소스: 사용자 지정을 선택하고 RDS 인스턴스와 연결된 보안 그룹 ID를 붙여 넣습니다. 이렇게 하면 해당 보안 그룹 내 모든 리소스의 트래픽이 허용됩니다. RDS 인스턴스의 IP 범위(CIDR 블록)를 지정할 수도 있습니다.
 - d. 규칙 저장을 클릭합니다.

추가 고려 사항

다음과 같은 추가 구성 정보를 고려해야 합니다.

- 보안 그룹 참조 사용: 소스 또는 페기 인스턴스에서 보안 그룹을 참조하면 동적 관리가 가능하며 그룹 내의 모든 리소스가 자동으로 포함되므로 IP 주소를 사용하는 것보다 안전합니다.
- 데이터베이스 포트: 데이터베이스에 올바른 포트를 사용하고 있는지 확인합니다.
- 보안 모범 사례: 보안 위험을 최소화하기 위해 필요한 포트만 엽니다. 또한 보안 그룹 규칙을 정기적으로 검토하여 보안 표준 및 요구 사항을 충족하는지 확인해야 합니다.

에 대한 네트워크 액세스 제어 목록(NACL) 구성 AWS DMS

Amazon RDS를 복제 소스로 사용하는 경우 DMS 및 RDS 인스턴스의 네트워크 액세스 제어 목록(NACL)을 업데이트해야 합니다. NACL이 이러한 인스턴스가 있는 서브넷과 연결되어 있는지 확인합니다. 이렇게 하면 특정 데이터베이스 포트에서 인바운드 및 아웃바운드 트래픽이 허용됩니다.

네트워크 액세스 제어 목록을 업데이트하려면 다음 단계를 수행해야 합니다.

Note

DMS 및 RDS 인스턴스가 동일한 서브넷에 있는 경우 해당 서브넷의 NACL만 업데이트하면 됩니다.

관련 NACL 식별

1. [Amazon VPC 콘솔](#)로 이동하세요.
2. 보안 아래 왼쪽 탐색 메뉴 섹션에서 네트워크 ACL을 선택합니다.
3. DMS 및 RDS 인스턴스가 있는 서브넷과 연결된 관련 NACL을 선택합니다.

DMS 인스턴스 서브넷의 NACL 업데이트

1. DMS 인스턴스의 서브넷과 연결된 NACL을 식별합니다. 이렇게 하려면 [Amazon VPC 콘솔](#)에서 서브넷을 탐색하고 DMS 서브넷을 찾은 다음 연결된 NACL ID를 기록해 둡니다.
2. 인바운드 규칙 편집:
 - a. 선택한 NACL에 대한 인바운드 규칙 탭을 클릭합니다.
 - b. 인바운드 규칙 편집을 선택합니다.
 - c. 새 규칙 추가:
 - 규칙 #: 고유 번호를 선택합니다(예: 100).
 - 유형: 사용자 지정 TCP 규칙을 선택합니다.
 - 프로토콜: TCP
 - 포트 범위: 데이터베이스 포트를 입력합니다(예: MySQL의 경우 3306).
 - 소스: RDS 서브넷의 CIDR 블록을 입력합니다(예: 10.1.0.0/16).
 - 허용/거부: 허용을 선택합니다.

3. 아웃바운드 규칙을 편집합니다.
 - a. 선택한 NACL에 대한 아웃바운드 규칙 탭을 클릭합니다.
 - b. 아웃바운드 규칙 편집을 클릭합니다.
 - c. 새 규칙 추가:
 - 규칙 #: 인바운드 규칙에 사용된 것과 동일한 번호를 사용합니다.
 - 유형:: 모든 트래픽.
 - 대상 주소: 0.0.0.0/0
 - 허용/거부: 허용을 선택합니다.
4. 변경 사항 저장을 클릭합니다.
5. 동일한 단계를 수행하여 RDS 인스턴스의 서브넷과 연결된 NACL을 업데이트합니다.

NACL 규칙 확인

NACL 규칙과 관련하여에 대한 다음 기준을 확인해야 합니다.

- 규칙 순서: NACL 규칙 번호를 기준으로 규칙을 오름차순으로 처리합니다. 트래픽을 차단할 수 있으므로 "허용"으로 설정된 모든 규칙의 규칙 번호가 "거부"로 설정된 모든 규칙보다 낮은지 확인합니다.
- 상태 비저장 특성: NACL은 상태 비저장입니다. 인바운드 트래픽과 아웃바운드 트래픽을 모두 명시적으로 허용해야 합니다.
- CIDR 블록: 사용하는 CIDR 블록이 DMS 및 RDS 인스턴스의 서브넷을 정확하게 나타내는지 확인해야 합니다.

AWS DMS 보안 암호 관리자 VPC 엔드포인트 구성

프라이빗 서브넷의 복제 인스턴스에서 AWS Secrets Manager에 액세스하려면 VPC 엔드포인트를 생성해야 합니다. 이렇게 하면 복제 인스턴스가 퍼블릭 인터넷을 통해 트래픽을 전송하지 않고 프라이빗 네트워크를 통해 직접 Secrets Manager에 액세스할 수 있습니다.

구성하려면 다음 단계를 따르세요.

VPC 엔드포인트에 대한 보안 그룹을 생성합니다.

1. [Amazon VPC 콘솔](#)로 이동하세요.

2. 왼쪽 탐색 창에서 보안 그룹과 보안 그룹 생성을 차례로 선택합니다.
3. 보안 그룹 세부정보 구성:
 - 보안 그룹 이름: 예시: SecretsManagerEndpointSG
 - 설명: 적절한 설명을 입력합니다. (예: 보안 암호 관리자 VPC 엔드포인트의 보안 그룹).
 - VPC: 복제 인스턴스와 엔드포인트가 있는 VPC를 선택합니다.
4. 규칙 추가를 클릭하여 인바운드 규칙을 설정하고 다음을 구성합니다.
 - 유형: HTTPS(보안 암호 관리자가 포트 443에서 HTTPS를 사용함).
 - 소스: 사용자 지정을 선택하고 복제 인스턴스의 보안 그룹 ID를 입력합니다. 이렇게 하면 해당 보안 그룹과 연결된 모든 인스턴스가 VPC 엔드포인트에 액세스할 수 있습니다.
5. 변경 사항을 검토하고 보안 그룹 생성을 클릭합니다.

Secrets Manager용 VPC 엔드포인트 생성

Note

Amazon Virtual Private Cloud 사용 설명서의 [인터페이스 엔드포인트 생성 설명서](#)에 설명된 대로 인터페이스 VPC 엔드포인트를 생성합니다. 이 절차를 따를 때는 다음을 확인하세요.

- 서비스 범주에서 AWS 서비스를 선택합니다.
- 서비스 이름에서 `seretsmanager` 검색 후 보안 암호 관리자 서비스를 선택합니다.

1. VPC 및 서브넷을 선택하고 다음을 구성합니다.
 - VPC: 복제 인스턴스와 동일한 VPC인지 확인합니다.
 - 서브넷: 복제 인스턴스가 있는 서브넷을 선택합니다.
2. 추가 설정에서 인터페이스 엔드포인트에 대해 DNS 이름 활성화가 기본적으로 활성화되어 있는지 확인합니다.
3. 보안 그룹에서 적절한 보안 그룹 이름을 선택합니다. 예: SecretsManagerEndpointSG, 앞서 생성한 대로).
4. 모든 설정을 검토하고 엔드포인트 생성을 클릭합니다.

VPC 엔드포인트 DNS 이름 검색

1. VPC 엔드포인트 세부 정보에 액세스합니다.
 - a. [Amazon VPC 콘솔](#)로 이동하여 엔드포인트를 선택합니다.
 - b. 생성한 적절한 엔드포인트를 선택합니다.
2. DNS 이름을 복사합니다.
 - a. 세부 정보 탭에서 DNS 이름 섹션으로 이동합니다.
 - b. 나열된 첫 번째 DNS 이름을 복사합니다. (예: `vpce-0abc123def456789g-secretsmanager.us-east-1.vpce.amazonaws.com`). 이는 지역 DNS 이름입니다.

DMS 엔드포인트 업데이트

1. [AWS DMS](#) 콘솔로 이동합니다.
2. DMS 엔드포인트를 수정합니다.
 - a. 왼쪽 탐색 창에서 엔드포인트를 선택합니다.
 - b. 구성하고자 하는 적절한 엔드포인트를 선택합니다.
 - c. 작업을 클릭하고 수정을 선택합니다.
3. 엔드포인트 서비스 구성:
 - a. 엔드포인트 설정으로 이동하여 엔드포인트 연결 속성 사용 확인란을 선택합니다.
 - b. 연결 속성 필드에 `secretsManagerEndpointOverride=<copied DNS name>`를 추가합니다.

Note

연결 속성이 여러 개인 경우 세미콜론 ";"로 구분할 수 있습니다. 예:

```
datePartitionEnabled=false;secretsManagerEndpointOverride=vpce-0abc123def456789g-secretsmanager.us-east-1.vpce.amazonaws.com
```

4. 엔드포인트 수정을 클릭하여 변경 사항을 저장합니다.

추가 고려 사항

다음과 같은 추가 구성 정보를 고려해야 합니다.

복제 인스턴스 보안 그룹:

- 복제 인스턴스와 연결된 보안 그룹이 포트 443(HTTPS)에서 VPC 엔드포인트로의 아웃바운드 트래픽을 허용하는지 확인합니다.

VPC DNS 설정:

- VPC에서 DNS 확인 및 DNS 핫 이름이 활성화되어 있는지 확인합니다. 이렇게 하면 인스턴스가 VPC 엔드포인트 DNS 이름을 확인할 수 있습니다. [Amazon VPC 콘솔](#)에서 VPC로 이동하여 VPC를 선택하여 DNS 확인 및 DNS 핫 이름이 "예"로 설정되어 있는지 확인할 수 있습니다.

연결 테스트:

- 복제 인스턴스에서 DNS 조회를 수행하여 nslookup secretsmanager.<region>amazonaws.com VPC 엔드포인트인을 확인할 수 있습니다. VPC 엔드포인트와 연결된 IP 주소를 반환해야 합니다

AWS DMS 작업 작업

AWS Database Migration Service(AWS DMS) 작업은 모든 작업이 수행되는 곳입니다. 마이그레이션 및 특수 처리(로깅 요구 사항, 제어 테이블 데이터 및 오류 처리 등)에 사용할 테이블(또는 뷰)과 스키마를 지정합니다.

작업은 세 가지 주요 단계로 구성되어 있습니다.

- 기존 데이터 마이그레이션(전체 로드)
- 캐시된 변경 사항 적용
- 지속적 복제(변경 데이터 캡처)

AWS DMS 마이그레이션 작업이 데이터를 마이그레이션하는 방법에 대한 자세한 내용과 개요는 섹션을 참조하세요. [의 상위 수준 보기 AWS DMS](#)

마이그레이션 작업 생성 시 몇 가지 사항에 대해 알고 있어야 합니다.

- 작업을 생성하려면 먼저 소스 엔드포인트, 대상 엔드포인트, 복제 인스턴스를 생성해야 합니다.
- 많은 작업 설정을 지정하여 마이그레이션 작업을 조정할 수 있습니다. AWS Management Console, AWS Command Line Interface (AWS CLI) 또는 AWS DMS API를 사용하여 이를 설정할 수 있습니다. 이러한 설정에는 마이그레이션 오류의 처리 방식 지정, 오류 로깅 및 제어 테이블 정보가 포함됩니다. 태스크 구성 파일을 사용하여 태스크 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#) 섹션을 참조하세요.
- 작업을 생성하면 즉시 실행할 수 있습니다. 필수 메타데이터가 정의된 대상 테이블은 자동으로 생성되어 로드되고, 복제 프로세스가 지속됨을 지정할 수 있습니다.
- 기본적으로는 작업을 생성하는 즉시 작업을 AWS DMS 시작합니다. 그렇지만 경우에 따라 작업 시작을 연기하기도 합니다. 예를 들어를 사용할 때 작업을 생성하는 프로세스와 일부 트리거 이벤트에 따라 작업을 시작하는 다른 프로세스가 AWS CLI있을 수 있습니다. 필요에 따라 작업 시작을 연기할 수 있습니다.
- 콘솔 또는 API를 사용하여 작업을 모니터링 AWS CLI, 중지 또는 AWS DMS 다시 시작할 수 있습니다. AWS DMS API를 사용하여 작업을 중지하는 방법에 대한 자세한 내용은 API 참조의 [StopReplicationTask](#)를 [AWS DMS 참조](#)하세요.

다음은 AWS DMS 작업으로 작업할 때 수행할 수 있는 작업입니다.

Task	관련 설명서
태스크 생성 작업 생성 시 마이그레이션 설정과 함께 소스, 대상 및 복제 인스턴스를 지정합니다.	작업 생성
지속적 복제 태스크 생성 원본과 대상 간에 지속적인 복제를 수행하는 작업을 설정할 수 있습니다.	를 사용하여 지속적 복제를 위한 작업 생성 AWS DMS
태스크 설정 적용 각 작업에는 데이터베이스 마이그레이션의 요구 사항에 따라 구성할 수 있는 설정이 있습니다. JSON 파일로 이 설정을 생성하거나 AWS DMS 콘솔을 사용하여 일부 설정과 함께 이 설정을 지정할 수 있습니다. 태스크 구성 파일을 사용하여 태스크 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제 섹션을 참조하세요.	AWS Database Migration Service 작업에 대한 작업 설정 지정
테이블 매핑 사용 테이블 매핑은 여러 가지 유형의 규칙을 사용하여 테이블에 대한 추가 태스크 설정을 지정합니다. 이러한 규칙을 사용하여 데이터 소스, 소스 스키마, 테이블 및 보기, 데이터, 태스크 중에 발생하는 테이블 변환 및 데이터 변환, 이러한 테이블과 열을 소스에서 대상으로 마이그레이션하는 방법에 대한 설정을 지정할 수 있습니다.	선택 규칙 선택 규칙 및 작업 변환 규칙 변환 규칙 및 작업 테이블 설정 규칙 테이블 및 컬렉션 설정 규칙과 작업
마이그레이션 전 태스크 평가 실행	작업에 대한 마이그레이션 전 평가 활성화 및 활용

Task	관련 설명서
마이그레이션 도중 문제를 일으킬 수 있는 소스 및 대상 데이터베이스와 관련된 문제를 보여주는 마이그레이션 전 태스크 평가를 활성화하고 실행할 수 있습니다. 여기에는 지원되지 않는 데이터 형식, 일치하지 않는 인덱스와 프라이머리 키, 기타 태스크 설정 충돌 등을 비롯한 문제가 포함될 수 있습니다. 태스크 실행 전에 이러한 마이그레이션 전 평가를 실행하여 마이그레이션 도중 문제가 발생하기 전에 잠재적 문제를 파악합니다.	
데이터 유효성 검사 데이터 검증은 AWS DMS 가 원본 데이터 저장소의 데이터와 대상 데이터 저장소의 데이터 비교를 수행하도록 하는 데 사용할 수 있는 작업 설정입니다.	AWS DMS 데이터 검증.
태스크 수정 작업이 중지되면 작업 설정을 수정할 수 있습니다.	작업 수정
태스크 이동 태스크가 중지되면 태스크를 다른 복제 인스턴스로 이동할 수 있습니다.	태스크 이동
태스크 도중 테이블 다시 로드 작업 중에 오류가 발생하면 작업 중에 테이블을 다시 로드할 수 있습니다.	작업 중 테이블 다시 로드

Task	관련 설명서
필터 적용 소스 필터를 사용하여 소스에서 대상으로 전송되는 레코드의 수와 유형을 제한할 수 있습니다. 예를 들어, 본사 사업장에서 근무하는 직원만이 대상 데이터베이스로 이동하도록 지정할 수 있습니다. 데이터 열에서 필터를 적용합니다.	소스 필터 사용
태스크 모니터링 작업에서 사용하는 테이블과 작업 성능에 관한 정보를 가져오는 방법에는 여러 가지가 있습니다.	AWS DMS 작업 모니터링
태스크 로그 관리 AWS DMS API 또는를 사용하여 작업 로그를 보고 삭제할 수 있습니다 AWS CLI.	AWS DMS 작업 로그 보기 및 관리

주제

- [작업 생성](#)
- [를 사용하여 지속적 복제를 위한 작업 생성 AWS DMS](#)
- [작업 수정](#)
- [태스크 이동](#)
- [작업 중 테이블 다시 로드](#)
- [작업 설정을 지정하기 위한 테이블 매핑 사용](#)
- [소스 필터 사용](#)
- [작업에 대한 마이그레이션 전 평가 활성화 및 활용](#)
- [작업 설정에 대한 보충 데이터 지정](#)

작업 생성

AWS DMS 마이그레이션 작업을 생성하려면 다음을 수행합니다.

- 마이그레이션 작업을 생성하기 전에 먼저 소스 엔드포인트, 대상 엔드포인트 및 복제 인스턴스를 생성합니다.
- 마이그레이션 방법을 선택합니다.
 - 데이터를 대상 데이터베이스로 마이그레이션 – 이 프로세스는 대상 데이터베이스에 파일 또는 테이블을 생성하며 대상에서 필요한 메타데이터를 자동으로 정의합니다. 또한 소스의 데이터로 테이블을 채웁니다. 테이블의 데이터가 동시에 로드되므로 효율성이 향상됩니다. 이 프로세스는의 기존 데이터 마이그레이션 옵션 AWS Management Console 이며 API Full Load에서 호출됩니다.
 - 마이그레이션 중 변경 사항 캡처 – 이 프로세스는 데이터가 소스에서 대상으로 마이그레이션되는 동안 발생하는 소스 데이터베이스의 변경 사항을 캡처합니다. 원래 요청된 데이터의 마이그레이션이 완료되면 변경 데이터 캡처(CDC) 프로세스에서는 캡처된 변경 사항을 대상 데이터베이스에 적용합니다. 변경 사항이 커밋된 트랜잭션의 단위로서 캡처 및 적용되며 다양한 대상 테이블을 하나의 소스 커밋으로서 업데이트할 수 있습니다. 이 접근 방식은 대상 데이터베이스에서 트랜잭션 무결성을 보장합니다. 이 프로세스는 콘솔의 기존 데이터 마이그레이션 및 지속적인 변경 사항 복제 옵션이며 API에서 full-load-and-cdc라고 합니다.
 - 소스 데이터베이스에서 유일한 데이터 변경 사항 복제 – 이 프로세스는 원본 DBMS(데이터베이스 관리 시스템)의 복구 로그 파일을 읽고 트랜잭션별로 항목을 그룹화합니다. 경우에 따라서는 적절한 시간 내에 대상에 변경 사항을 적용할 AWS DMS 수 없습니다(예: 대상에 액세스할 수 없는 경우). 이러한 경우는 필요한 기간 동안 복제 서버의 변경 사항을 AWS DMS 버퍼합니다. 원본 DBMS 로그를 다시 읽지 않으므로 오랜 시간이 소요될 수 있습니다. 이 프로세스는 AWS DMS 콘솔의 데이터 변경 내용 복제 전용 옵션입니다.
- 작업에서 소스에 대해 대용량 이진 객체(LOB)를 처리하는 방식을 결정합니다. 자세한 내용은 [AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정](#) 단원을 참조하십시오.
- 마이그레이션 작업 설정을 지정합니다. 여기에는 로깅 설정, 마이그레이션 제어 테이블에 기록될 데이터 지정, 오류 처리 방식 및 기타 설정이 포함됩니다. 작업 설정에 관한 자세한 내용은 [AWS Database Migration Service 작업에 대한 작업 설정 지정](#) 섹션을 참조하세요.
- 테이블 매핑을 설정하여 마이그레이션 중인 데이터를 선택하고 필터링하기 위한 규칙을 정의합니다. 테이블 매핑에 관한 자세한 내용은 [작업 설정을 지정하기 위한 테이블 매핑 사용](#) 섹션을 참조하세요. 매핑을 지정하기 전에 소스 및 대상 데이터베이스의 데이터 형식 매핑에 대한 관련 설명서 섹션을 참조하세요.

- 작업을 실행하기 전에 마이그레이션 전 작업 평가를 활성화하고 실행하세요. 마이그레이션 전 평가에 관한 자세한 내용은 [작업에 대한 마이그레이션 전 평가 활성화 및 활용](#) 섹션을 참조하세요.
- 데이터를 마이그레이션하는 작업에 필요한 보충 데이터를 지정합니다. 자세한 내용은 [작업 설정에 대한 보충 데이터 지정](#) 단원을 참조하십시오.

작업 생성 페이지에서 해당 작업에 관한 정보를 모두 지정하면 바로 작업을 시작할 수 있습니다. 또는 나중에 대시보드 페이지에서 작업을 시작할 수도 있습니다.

다음 절차에서는 복제 인스턴스 정보와 엔드포인트를 이미 지정했다고 가정합니다. 엔드포인트 설정에 관한 자세한 내용은 [소스 및 대상 엔드포인트 생성](#) 섹션을 참조하세요.

마이그레이션 작업을 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

AWS Identity and Access Management (IAM) 사용자로 로그인한 경우에 액세스할 수 있는 적절한 권한이 있는지 확인합니다 AWS DMS. 필요한 권한에 관한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

2. 탐색 창에서 데이터베이스 마이그레이션 작업을 선택한 후 작업 생성을 선택합니다.
3. 데이터베이스 마이그레이션 작업 생성 페이지의 작업 구성 섹션에서 작업 옵션을 지정합니다. 다음 표는 설정에 대한 설명입니다.

Create database migration task

Task configuration

Task identifier

Type a unique identifier for the task

Descriptive Amazon Resource Name (ARN) - *optional*

A friendly name to override the default DMS ARN. You cannot modify it after creation.

Friendly-ARN-name

Replication instance

Choose a replication instance

Source database endpoint

Choose a source database endpoint

Target database endpoint

Choose a target database endpoint

Migration type [Info](#)

Migrate existing data

옵션

작업 식별자

조치

작업의 이름을 입력합니다.

옵션	조치
설명이 포함된 Amazon 리소스 이름 (ARN) - 선택 사항	기본 AWS DMS ARN을 재정의할 표시 이름입니다. 작업을 생성한 후에는 이 이름을 변경할 수 없습니다.
복제 인스턴스	사용할 복제 인스턴스를 표시합니다.
소스 데이터베이스 엔드포인트	사용할 소스 엔드포인트를 표시합니다.
대상 데이터베이스 엔드포인트	사용할 대상 엔드포인트를 표시합니다.
[Migration type]	사용할 마이그레이션 방법을 선택합니다. 기존 데이터가 대상 데이터베이스에 마이그레이션되도록 선택하거나 마이그레이션된 데이터 외에 지속적 변경 사항이 대상 데이터베이스에 전송되도록 선택할 수 있습니다.

4. 작업 설정 섹션에서 작업 편집, 대상 테이블 준비 모드, 작업 중지, LOB 설정, 검증 및 로깅을 위한 값을 지정합니다.

옵션	조치
[Editing mode]	작업 설정을 지정하는 데 마법사를 사용할지 아니면 JSON 편집기를 사용할지 선택합니다. 마법사를 선택하면 다음 옵션이 표시됩니다.
소스 트랜잭션을 위한 CDC 시작 모드	이 설정은 이전 섹션의 마이그레이션 유형으로만 데이터 변경 복제를 선택한 경우에만 표시됩니다. 사용자 지정 CDC 시작 모드 비활성화 - 이 옵션을 선택하면 다음에 나오는 생성 시 자동으로 옵션을 사용하여 작업을 자동으로 시작하거나 콘솔을 사용하여 수동으로 작업을 시작할 수 있습니다. 사용자 지정 CDC 시작 모드 활성화 - 이 옵션을 선택하면 변경 처리를 시작할 사용자 지정 UTC 시작 시간을 지정할 수 있습니다.
대상 테이블 준비 모드	이 설정은 이전 섹션의 마이그레이션 유형에 대해 기존 데이터 마이그레이션 또는 기존 데이터를 마이그레이

옵션	조치
	선하고 진행 중인 변경 사항을 복제를 선택하는 경우에만 표시됩니다. 아무 작업 안 함 - 아무 작업 안 함 모드에서 대상 테이블에 대상 테이블이 미리 생성되었다고 AWS DMS가 정합니다. 테이블이 비어 있지 않으면 데이터 마이그레이션 중에 충돌이 발생하여 DMS 작업 오류가 발생할 수 있습니다. 대상 테이블이 없는 경우 DMS에서 테이블을 생성합니다. 테이블 구조는 그대로 유지되고 기존 데이터는 테이블에 남습니다. 아무 작업 안 함 모드는 소스에서 대상 테이블이 채워져 있는 경우 CDC 전용 작업에 적합하고 진행 중 복제는 소스 및 대상 간 동기화를 유지합니다. 테이블을 미리 생성하려면 AWS Schema Conversion Tool (AWS SCT)를 사용하면 됩니다. 자세한 내용은 설치를 AWS SCT 참조하세요. 대상에서 테이블 삭제 - 대상에서 테이블 삭제 모드에서는 AWS DMS가 마이그레이션 시작 전에 대상 테이블을 삭제하고 다시 만듭니다. 이 접근 방식은 마이그레이션이 시작될 때 대상 테이블이 비어 있도록 합니다.는 테이블, 기본 키 및 경우에 따라 고유 인덱스와 같이 데이터를 효율적으로 마이그레이션하는 데 필요한 객체만 AWS DMS 생성합니다. AWS DMS는 보조 인덱스, 비기본 키 제약 조건 또는 열 데이터 기본값을 생성하지 않습니다. 전체 로드와 CDC 또는 CDC 전용 작업을 수행하는 경우, 이 시점에서 마이그레이션을 일시 중지하는 것이 좋습니다. 그런 다음, update 및 delete 문에 대한 필터링을 지원하는 보조 인덱스를 생성합니다. 대상에서 테이블 삭제 모드를 사용할 경우 대상 데이터베이스에서 몇 가지 구성을 실행해야 합니다. 예를 들어 Oracle 대상의 경우는 보안상의 이유로 스키마(데이터베이스 사용자)를 생성할 AWS DMS 수 없습니다. 이 경우 마이그레이션이 시작될 때가 테이블을 생성할 AWS DMS 수 있도록 스키마 사용자를 미리 생성합니다. 대부분의 다른 대상 유형의 경우는 적절한 구성 파라미

옵션	조치
	터블 사용하여 스키마와 연결된 모든 테이블을 AWS DMS 생성합니다. 잘라내기 - 잘라내기 모드에서는 마이그레이션이 시작되기 전에 모든 대상 테이블을 AWS DMS 잘라냅니다. 대상 테이블이 없는 경우 DMS에서 테이블을 생성합니다. 테이블 구조는 그대로 유지되지만 대상에서 테이블이 잘립니다. 자르기 모드는 마이그레이션이 시작되기 전에 대상 스키마가 미리 생성된 전체 로드 또는 전체 로드와 CDC 마이그레이션에 적합합니다. 테이블을 미리 생성하려면 AWS SCT를 사용하면 됩니다. 자세한 내용은 설치를 AWS SCT 참조하세요.  Note 대상이 MongoDB인 경우 자르기 모드에서는 대상의 테이블이 잘리지 않습니다. 그 대신 컬렉션이 삭제되고 모든 인덱스가 손실됩니다. 대상이 MongoDB인 경우 자르기 모드를 사용하지 마세요.

옵션	조치
전체 로드 완료 후 작업 중지	이 설정은 이전 섹션의 마이그레이션 유형에 대해 기존 데이터를 마이그레이션하고 진행 중인 변경 사항을 복제를 선택하는 경우에만 표시됩니다. 멈추지 않음 - 작업을 중지하지 않고, 캐시된 변경 사항을 즉시 적용하고 계속 진행합니다. 캐시된 변경 사항 적용 전 멈춤 - 캐시된 변경 사항을 적용하기 전에 작업을 중지합니다. 이 방법을 사용하면 보조 인덱스를 추가할 수 있으며 이에 따라 변경 사항을 적용하는 속도가 향상될 수 있습니다. [Stop after applying cached changes] - 캐시된 변경 사항 적용 후 작업을 중지합니다. 이 방법을 사용하면 트랜잭션 적용을 사용하는 경우 외래 키를 추가할 수 있습니다.
복제에 LOB 컬럼 포함	LOB 열 포함 안 함 - LOB 열이 마이그레이션에서 제외됩니다. 전체 LOB 모드 - size. AWS DMS migrates에 LOBs 관계없이 전체 LOBs 청크 크기 파라미터로 제어되는 청크로 구분하여 마이그레이션합니다. 이 모드는 제한적 LOB 모드를 사용하는 것보다 더 느립니다. 제한적 LOB 모드 - LOB를 최대 LOB 크기 파라미터 값으로 자릅니다. 이 모드는 전체 LOB 모드를 사용하는 것보다 더 빠릅니다.
최대 LOB 크기(kb)	제한적 LOB 모드에서는 최대 LOB 크기의 설정을 초과하는 LOB 열이 지정된 최대 LOB 크기 값으로 잘립니다.
검증 활성화	원본에서 대상으로 데이터가 정확히 마이그레이션되는지 확인하기 위해 데이터 검증을 활성화합니다. 자세한 내용은 AWS DMS 데이터 검증 단원을 참조하십시오.

옵션	조치
CloudWatch 로그 활성화	Amazon CloudWatch를 통해 로깅을 활성화합니다.

5. 마이그레이션 전 평가 섹션에서 마이그레이션 전 평가 실행 여부를 선택합니다. 마이그레이션 전 평가를 통해 데이터베이스 마이그레이션 작업을 시작하기 전에 잠재적인 마이그레이션 문제를 경고할 수 있습니다. 자세한 내용은 [마이그레이션 전 평가 활성화 및 활용](#) 단원을 참조하십시오.
6. 마이그레이션 작업 시작 구성 섹션에서 작업 생성 후 자동으로 시작할지 여부를 지정합니다.
7. 태그 섹션에서 작업을 정리하는 데 필요한 태그를 지정합니다. 태그를 사용하여 IAM 역할 및 정책을 관리하고 DMS 비용을 추적할 수 있습니다. 자세한 내용은 [리소스에 태그 지정](#) 단원을 참조하십시오.
8. 작업 설정을 완료했다면 작업 생성을 선택합니다.

AWS Database Migration Service 작업에 대한 작업 설정 지정

각 작업에는 데이터베이스 마이그레이션의 요구 사항에 따라 구성할 수 있는 설정이 있습니다. JSON 파일에서 이러한 설정을 생성하거나 일부 설정에서는 AWS DMS 콘솔을 사용하여 설정을 지정할 수 있습니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

기본적인 작업 설정 유형은 다음과 같습니다.

주제

- [작업 설정 예제](#)
- [대상 메타데이터 작업 설정](#)
- [전체 로드 작업 설정](#)
- [Time Travel 작업 설정](#)
- [작업 설정 로깅](#)
- [제어 테이블 작업 설정](#)
- [스트림 버퍼 작업 설정](#)
- [변경 처리 튜닝 설정](#)
- [데이터 검증 작업 설정](#)
- [데이터 재동기 설정](#)
- [변경 처리 DDL을 다루기 위한 작업 설정](#)

- [문자 대체 작업 설정](#)
- [이전 이미지 작업 설정](#)
- [오류 처리 작업 설정](#)
- [작업 설정 저장](#)

작업 설정	관련 설명서
작업 평가 보고서 생성 마이그레이션 중에 문제를 유발할 수 있는 지원되지 않는 데이터 형식을 보여 주는 작업 평가 보고서를 생성할 수 있습니다. 작업을 실행하기 전에 잠재적인 문제를 알아보기 위해 작업에 대해 이 보고서를 실행할 수 있습니다.	작업에 대한 마이그레이션 전 평가 활성화 및 활용
작업 생성 작업 생성 시 마이그레이션 설정과 함께 소스, 대상 및 복제 인스턴스를 지정합니다.	작업 생성
지속적 복제 태스크 생성 원본과 대상 간에 지속적인 복제를 수행하는 작업을 설정할 수 있습니다.	를 사용하여 지속적 복제를 위한 작업 생성 AWS DMS
태스크 설정 적용 각 작업에는 데이터베이스 마이그레이션의 요구 사항에 따라 구성할 수 있는 설정이 있습니다. JSON 파일에서 이러한 설정을 생성하거나 일부 설정에서는 AWS DMS 콘솔을 사용하여 설정을 지정할 수 있습니다.	AWS Database Migration Service 작업에 대한 작업 설정 지정
데이터 유효성 검사	AWS DMS 데이터 검증

작업 설정	관련 설명서
데이터 검증을 사용하여 대상 데이터 스토어의 데이터를 소스 데이터 스토어의 데이터와 AWS DMS 비교합니다.	
작업 수정 작업이 중지되면 작업 설정을 수정할 수 있습니다.	작업 수정
작업 중 테이블 다시 로드 작업 중에 오류가 발생하면 작업 중에 테이블을 다시 로드할 수 있습니다.	작업 중 테이블 다시 로드
테이블 매핑 사용 테이블 매핑에서는 여러 유형의 규칙을 사용하여 데이터 소스, 원본 스키마, 데이터, 작업 중 발생하는 모든 변환 등에 대한 작업 설정을 지정합니다.	선택 규칙 선택 규칙 및 작업 변환 규칙 변환 규칙 및 작업
필터 적용 소스 필터를 사용하여 소스에서 대상으로 전송되는 레코드의 수와 유형을 제한할 수 있습니다. 예를 들어, 본사 사업장에서 근무하는 직원만이 대상 데이터베이스로 이동하도록 지정할 수 있습니다. 데이터 열에서 필터를 적용합니다.	소스 필터 사용
태스크 모니터링 작업에서 사용하는 테이블과 작업 성능에 관한 정보를 가져오는 방법에는 여러 가지가 있습니다.	AWS DMS 작업 모니터링

작업 설정	관련 설명서
태스크 로그 관리 AWS DMS API 또는를 사용하여 작업 로그를 보고 삭제할 수 있습니다 AWS CLI.	AWS DMS 작업 로그 보기 및 관리

작업 설정 예제

AWS Management Console 또는를 사용하여 복제 작업을 AWS CLI 생성할 수 있습니다. 를 사용하는 경우 JSON 파일을 생성한 다음 JSON 파일의 file:// URI를 [CreateReplicationTask](#) [작업의 ReplicationTaskSettings](#) 파라미터로 지정하여 작업 설정을 AWS CLI 지정합니다.

다음 예제에서는를 사용하여 CreateReplicationTask 작업을 호출 AWS CLI 하는 방법을 보여줍니다.

```
aws dms create-replication-task \
--replication-task-identifier MyTask \
--source-endpoint-arn arn:aws:dms:us-
west-2:123456789012:endpoint:ABCDEFGHIJKLMN0PQRSTUVWXYZ1234567890ABC \
--target-endpoint-arn arn:aws:dms:us-
west-2:123456789012:endpoint:ABCDEFGHIJKLMN0PQRSTUVWXYZ1234567890ABC \
--replication-instance-arn arn:aws:dms:us-
west-2:123456789012:rep:ABCDEFGHIJKLMN0PQRSTUVWXYZ1234567890ABC \
--migration-type cdc \
--table-mappings file://tablemappings.json \
--replication-task-settings file://settings.json
```

위 예제에서는 tablemappings.json이라는 테이블 매핑 파일을 사용합니다. 테이블 매핑 예는 [작업 설정을 지정하기 위한 테이블 매핑 사용](#) 섹션을 참조하세요.

작업 설정 JSON 파일은 다음과 같습니다.

```
{
  "TargetMetadata": {
    "TargetSchema": "",
    "SupportLobs": true,
```

```

    "FullLobMode": false,
    "LobChunkSize": 64,
    "LimitedSizeLobMode": true,
    "LobMaxSize": 32,
    "InlineLobMaxSize": 0,
    "LoadMaxFileSize": 0,
    "ParallelLoadThreads": 0,
    "ParallelLoadBufferSize": 0,
    "ParallelLoadQueuesPerThread": 1,
    "ParallelApplyThreads": 0,
    "ParallelApplyBufferSize": 100,
    "ParallelApplyQueuesPerThread": 1,
    "BatchApplyEnabled": false,
    "TaskRecoveryTableEnabled": false
  },
  "FullLoadSettings": {
    "TargetTablePrepMode": "DO_NOTHING",
    "CreatePkAfterFullLoad": false,
    "StopTaskCachedChangesApplied": false,
    "StopTaskCachedChangesNotApplied": false,
    "MaxFullLoadSubTasks": 8,
    "TransactionConsistencyTimeout": 600,
    "CommitRate": 10000
  },
  "TTSettings" : {
    "EnableTT" : true,
    "TTS3Settings": {
      "EncryptionMode": "SSE_KMS",
      "ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-west-2:112233445566:key/
myKMSKey",
      "ServiceAccessRoleArn": "arn:aws:iam::112233445566:role/dms-tt-s3-access-role",
      "BucketName": "myttbucket",
      "BucketFolder": "myttfolder",
      "EnableDeletingFromS3OnTaskDelete": false
    },
    "TTRecordSettings": {
      "EnableRawData" : true,
      "OperationsToLog": "DELETE,UPDATE",
      "MaxRecordSize": 64
    }
  },
  "Logging": {
    "EnableLogging": false
  },

```

```
"ControlTablesSettings": {
  "ControlSchema": "",
  "HistoryTimeslotInMinutes": 5,
  "HistoryTableEnabled": false,
  "SuspendedTablesTableEnabled": false,
  "StatusTableEnabled": false
},
"StreamBufferSettings": {
  "StreamBufferCount": 3,
  "StreamBufferSizeInMB": 8
},
"ChangeProcessingTuning": {
  "BatchApplyPreserveTransaction": true,
  "BatchApplyTimeoutMin": 1,
  "BatchApplyTimeoutMax": 30,
  "BatchApplyMemoryLimit": 500,
  "BatchSplitSize": 0,
  "MinTransactionSize": 1000,
  "CommitTimeout": 1,
  "MemoryLimitTotal": 1024,
  "MemoryKeepTime": 60,
  "StatementCacheSize": 50
},
"ChangeProcessingDdlHandlingPolicy": {
  "HandleSourceTableDropped": true,
  "HandleSourceTableTruncated": true,
  "HandleSourceTableAltered": true
},
"LoopbackPreventionSettings": {
  "EnableLoopbackPrevention": true,
  "SourceSchema": "LOOP-DATA",
  "TargetSchema": "loop-data"
},
"CharacterSetSettings": {
  "CharacterReplacements": [ {
    "SourceCharacterCodePoint": 35,
    "TargetCharacterCodePoint": 52
  }, {
    "SourceCharacterCodePoint": 37,
    "TargetCharacterCodePoint": 103
  }
],
  "CharacterSetSupport": {
```

```

    "CharacterSet": "UTF16_PlatformEndian",
    "ReplaceWithCharacterCodePoint": 0
  }
},
"BeforeImageSettings": {
  "EnableBeforeImage": false,
  "FieldName": "",
  "ColumnFilter": "pk-only"
},
"ErrorBehavior": {
  "DataErrorPolicy": "LOG_ERROR",
  "DataTruncationErrorPolicy": "LOG_ERROR",
  "DataMaskingErrorPolicy": "STOP_TASK",
  "DataErrorEscalationPolicy": "SUSPEND_TABLE",
  "DataErrorEscalationCount": 50,
  "TableErrorPolicy": "SUSPEND_TABLE",
  "TableErrorEscalationPolicy": "STOP_TASK",
  "TableErrorEscalationCount": 50,
  "RecoverableErrorCount": 0,
  "RecoverableErrorInterval": 5,
  "RecoverableErrorThrottling": true,
  "RecoverableErrorThrottlingMax": 1800,
  "ApplyErrorDeletePolicy": "IGNORE_RECORD",
  "ApplyErrorInsertPolicy": "LOG_ERROR",
  "ApplyErrorUpdatePolicy": "LOG_ERROR",
  "ApplyErrorEscalationPolicy": "LOG_ERROR",
  "ApplyErrorEscalationCount": 0,
  "FullLoadIgnoreConflicts": true
},
"ValidationSettings": {
  "EnableValidation": false,
  "ValidationMode": "ROW_LEVEL",
  "ThreadCount": 5,
  "PartitionSize": 10000,
  "FailureMaxCount": 1000,
  "RecordFailureDelayInMinutes": 5,
  "RecordSuspendDelayInMinutes": 30,
  "MaxKeyColumnSize": 8096,
  "TableFailureMaxCount": 10000,
  "ValidationOnly": false,
  "HandleCollationDiff": false,
  "RecordFailureDelayLimitInMinutes": 1,
  "SkipLobColumns": false,
  "ValidationPartialLobSize": 0,

```

```

    "ValidationQueryCdcDelaySeconds": 0
  }
}

```

대상 메타데이터 작업 설정

대상 메타데이터 설정에는 다음이 포함됩니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

- **TargetSchema** – 대상 테이블 스키마 이름입니다. 이 메타데이터 옵션이 비어 있는 경우, 원본 테이블의 스키마가 사용됩니다. AWS DMS 는 원본 스키마가 정의되지 않은 경우 대상 데이터베이스의 소유자 접두사를 모든 테이블에 자동으로 추가합니다. 이 옵션은 MySQL 유형 대상 엔드포인트에서 비어두어야 합니다. 데이터 매핑에서 스키마 이름을 바꾸는 것이 이 설정보다 우선합니다.
- **LOB 설정** – 대형 객체(LOB)가 관리되는 방법을 결정하는 설정입니다. SupportLobs=true를 설정하는 경우, 다음 중 하나를 true로 설정해야 합니다.
 - **FullLobMode** – 이 옵션을 true로 설정하는 경우, LobChunkSize 옵션 값을 입력해야 합니다. 데이터를 대상으로 복제할 때 사용할 LOB 청크의 크기(단위: 킬로바이트[KB])를 입력합니다. FullLobMode 옵션은 매우 큰 LOB 크기에서 최고 성능을 발휘하지만 로드가 느려지는 경향이 있습니다. LobChunkSize의 권장값은 64킬로바이트[KB]입니다. LobChunkSize의 값을 64KB 이상으로 늘리면 작업이 실패할 수 있습니다.
 - **InlineLobMaxSize** -이 값은 전체 로드 중에 인라인LOBs AWS DMS 를 결정합니다. 작은 LOB를 전송하면 소스 테이블에서 조회하는 것보다 효율적입니다. 전체 로드 중에는 모든 LOBs AWS DMS 확인하고 보다 작은 LOBs에 대해 인라인 전송을 수행합니다InlineLobMaxSize. 는 InlineLobMaxSize의 보다 큰 모든 LOBs를 AWS DMS 전송합니다FullLobMode. InlineLobMaxSize의 기본값은 0이고 범위는 1~102400킬로바이트(100MB)입니다. 모든 LOB가 InlineLobMaxSize에 지정된 값보다 작으면 InlineLobMaxSize 값만 지정하세요.
 - **LimitedSizeLobMode** – 이 옵션을 true로 설정하는 경우, LobMaxSize 옵션 값을 입력해야 합니다. 개별 LOB에서 최대 크기(KB)를 입력합니다. LobMaxSize의 최대 값은 102400킬로바이트(100MB)입니다.

이러한 작업 LOB 설정을 사용하기 위한 기준에 관한 자세한 내용은 [AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정](#) 섹션을 참조하세요. 개별 테이블에 대한 LOB 관리를 제어할 수도 있습니다. 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 단원을 참조하십시오.

- **BatchApplyEnabled** – 각 트랜잭션이 개별적으로 적용되는지 또는 변경 사항이 배치 단위로 커밋되는지 여부를 결정합니다. 기본값은 false입니다.

BatchApplyEnabled가 true로 설정된 경우, DMS는 소스 테이블에 프라이머리 키(PK) 또는 고유 키(UK)를 필요로 합니다. 소스 테이블에 PK 또는 UK가 없으면 배치 삽입만 적용되며 배치 업데이트 및 삭제는 적용되지 않습니다.

BatchApplyEnabled를 true로 설정할 때 대상 테이블에 고유한 제약 조건과 프라이머리 키가 있는 경우 AWS DMS에서는 오류 메시지가 생성됩니다. BatchApplyEnabled가 true로 설정된 경우, 고유 제약조건과 프라이머리 키가 모두 있는 대상 테이블은 지원되지 않습니다.

BatchApplyEnabled이 true로 설정되고 기본 오류 처리 정책이 있는 테이블에서 데이터 오류가 AWS DMS 발생하면 AWS DMS 작업은 나머지 테이블에서 배치 모드에서 one-by-one 모드로 전환됩니다. 이러한 동작을 변경하려면 작업 설정 JSON 파일의 "ErrorBehavior" 그룹 속성에서 다음 정책에 대한 "SUSPEND_TABLE" 작업을 설정할 수 있습니다.

- DataErrorPolicy
- ApplyErrorDeletePolicy
- ApplyErrorInsertPolicy
- ApplyErrorUpdatePolicy

이 "ErrorBehavior" 그룹 속성에 관한 자세한 내용은 [AWS Database Migration Service 작업에 대한 작업 설정 지정](#)의 예제 작업 설정 JSON 파일을 참조하세요. 이러한 정책을 로 설정한 후 "SUSPEND_TABLE" AWS DMS 작업은 이러한 정책을 발생시키는 테이블에 대한 데이터 오류를 일시 중지하고 모든 테이블에 대해 배치 모드로 계속합니다.

BatchApplyEnabled 파라미터를 BatchApplyPreserveTransaction 파라미터와 함께 사용할 수 있습니다. BatchApplyEnabled가 true로 설정되면 BatchApplyPreserveTransaction 파라미터는 트랜잭션 무결성을 결정합니다.

BatchApplyPreserveTransaction이 true로 설정되면 트랜잭션 무결성이 유지되며 배치는 원본의 트랜잭션 내에 있는 모든 변경 사항을 포함하게 됩니다.

BatchApplyPreserveTransaction이 false로 설정되면 트랜잭션 무결성이 일시적인 시간 경과가 있어 성능이 개선될 수 있습니다.

BatchApplyPreserveTransaction 파라미터는 Oracle 대상 엔드포인트에만 적용되고 BatchApplyEnabled 파라미터가 true로 설정되는 경우에만 관련됩니다.

복제에 LOB 열이 포함되면 BatchApplyEnabled를 제한된 LOB 모드에서만 사용할 수 있습니다.

CDC(Change Data Capture) 로드에서 이 설정을 사용하는 방법에 관한 자세한 내용은 [변경 처리 튜닝 설정](#) 섹션을 참조하세요.

- `MaxFullLoadSubTasks` – 동시에 로드할 최대 테이블 수를 나타냅니다. 기본값은 8이며 최대값은 49입니다.
- `ParallelLoadThreads` -가 각 테이블을 대상 데이터베이스로 로드하는 데 AWS DMS 사용하는 스레드 수를 지정합니다. 이 파라미터는 RDBMS가 아닌 대상에 대한 최대값을 가집니다. DynamoDB 대상의 최대값은 200입니다. Amazon Kinesis Data Streams, Apache Kafka 또는 Amazon OpenSearch Service 대상의 최대값은 32입니다. 이 최대 한도를 늘리도록 요청할 수 있습니다. `ParallelLoadThreads`는 전체 로드 작업에 적용됩니다. 개별 테이블의 병렬 로드 설정에 관한 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 섹션을 참조하세요.

이 설정은 다음 엔드포인트 엔진 유형에 적용됩니다.

- DynamoDB
- Amazon Kinesis Data Streams
- Amazon MSK
- Amazon OpenSearch Service
- Amazon Redshift

AWS DMS는 `MySQLParallelLoadThreads`를 추가 연결 속성으로 지원합니다.

`ParallelLoadThreads`는 MySQL에 태스크 설정으로 적용되지 않습니다.

- `ParallelLoadBufferSize`는 데이터를 대상에 로드하기 위해 병렬 로드 스레드에 사용되는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 현재 이 설정은 DynamoDB, Kinesis, Apache Kafka 또는 OpenSearch가 대상일 때만 유효합니다. 이 파라미터는 `ParallelLoadThreads`와 함께 사용하세요. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다. 개별 테이블의 병렬 로드 설정에 관한 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 섹션을 참조하세요.
- `ParallelLoadQueuesPerThread` – 대기열에서 데이터 레코드를 가져오고 대상에 대한 배치 로드를 생성하기 위해 각 동시 스레드가 액세스하는 대기열의 수를 지정합니다. 기본값은 1입니다. 현재 이 설정은 Kinesis 또는 Apache Kafka가 대상일 때만 유효합니다.
- `ParallelApplyThreads` - CDC 로드 중에 데이터 레코드를 Amazon DocumentDB, Kinesis, Amazon MSK, OpenSearch 또는 Amazon Redshift 대상 엔드포인트로 푸시하기 위해 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0입니다.

이 설정은 CDC 전용입니다. 이 설정은 전체 로드에서 적용되지 않습니다.

이 설정은 다음 엔드포인트 엔진 유형에 적용됩니다.

- Amazon DocumentDB(MongoDB 호환)
- Amazon Kinesis Data Streams
- Amazon Managed Streaming for Apache Kafka
- Amazon OpenSearch Service
- Amazon Redshift
- `ParallelApplyBufferSize` – CDC 로드 중에 동시 스레드가 Amazon DocumentDB, Kinesis, Amazon MSK, OpenSearch 또는 Amazon Redshift 대상 엔드포인트로 푸시할 수 있도록 각 버퍼 대기열에 저장할 레코드의 최대 수를 지정합니다. 기본값은 100입니다. 최대값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.
- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Amazon DocumentDB, Kinesis, Amazon MSK 또는 OpenSearch 엔드포인트에 대한 배치 로드를 생성하기 위해 액세스하는 대기열의 수를 지정합니다. 기본값은 1입니다.

전체 로드 작업 설정

전체 로드 설정에는 다음이 포함됩니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

- 전체 로드 시작 시 대상 로드 처리 방법을 표시하려면, `TargetTablePrepMode` 옵션에서 다음 값 중 하나를 지정합니다.
 - `DO_NOTHING` – 기존 대상 테이블의 데이터와 메타데이터가 영향을 받지 않습니다.
 - `DROP_AND_CREATE` – 기존 테이블은 삭제되고 새 테이블이 그 자리에 생성됩니다.
 - `TRUNCATE_BEFORE_LOAD` – 테이블 메타데이터에 영향을 주지 않고 데이터가 잘립니다.
- 전체 로드가 완료될 때까지 프라이머리 키 또는 고유 인덱스 생성을 지연시키려면 `CreatePkAfterFullLoad` 옵션을 `true`로 설정합니다.
- 전체 로드 및 CDC 활성 작업에서 `Stop task after full load completes`에 대해 다음과 같은 옵션을 설정할 수 있습니다.
 - `StopTaskCachedChangesApplied` – 이 옵션을 `true`로 설정하면 전체 로드가 완료되고 캐시된 변경 사항이 적용된 후 작업이 중지됩니다.
 - `StopTaskCachedChangesNotApplied` – 이 옵션을 `true`로 설정하면 작업을 중지한 후 캐시된 변경 사항이 적용됩니다.

- 동시에 로드할 테이블의 최대 개수를 나타내려면 `MaxFullLoadSubTasks` 옵션을 설정합니다. 기본값은 8이며 최대값은 49입니다.
- 데이터 레코드를 대상 엔드포인트로 푸시하기 위해 전체 로드 프로세스 중에 DMS가 사용할 동시 스레드 수를 나타내도록 `ParallelLoadThreads` 옵션을 설정합니다. 기본값은 0입니다.

Important

`MaxFullLoadSubTasks`는 동시에 로드할 테이블 또는 테이블 세그먼트의 수를 제어합니다. `ParallelLoadThreads`는 마이그레이션 작업에서 로드를 동시에 실행하는 데 사용하는 스레드의 수를 제어합니다. 이러한 설정은 배수로 적용됩니다. 따라서 전체 로드 작업 중에 사용된 스레드의 총 개수는 대략 `ParallelLoadThreads * MaxFullLoadSubTasks`의 값을 곱한 결과(즉, `ParallelLoadThreads * MaxFullLoadSubtasks`))입니다.

전체 로드 하위 작업의 수가 많고 병렬 로드 스레드의 수가 많은 작업을 만들면 작업이 너무 많은 메모리를 소비하여 실패할 수 있습니다.

- 전체 로드 작업을 시작하기 전에가 트랜잭션이 종료될 때까지 AWS DMS 기다리는 초수를 설정할 수 있습니다. 이렇게 하려면 작업을 시작할 때 트랜잭션이 열려 있는 경우 `TransactionConsistencyTimeout` 옵션을 설정합니다. 기본값은 600(10분)입니다. 열려 있는 트랜잭션이 있더라도 제한 시간 값에 도달한 후 전체 로드를 AWS DMS 시작합니다. 전체 로드 전용 작업은 10분 동안 기다리지 않으며 즉시 시작됩니다.
- 함께 전송할 수 있는 레코드의 최대 수를 나타내려면 `CommitRate` 옵션을 설정합니다. 기본값은 10,000이고 최대값은 50,000입니다.

Time Travel 작업 설정

복제 작업을 로깅하고 디버깅하려면 AWS DMS Time Travel을 사용할 수 있습니다. 이 방식에서는 Amazon S3를 사용하여 로그를 저장하고 암호화 키를 사용하여 로그를 암호화합니다. Time Travel S3 버킷에 액세스할 경우에만 날짜-시간 필터를 사용하여 S3 로그를 검색한 다음, 필요에 따라 로그를 보고 다운로드하며 난독화할 수 있습니다. 이렇게 하면 안전하게 '과거로 이동'하여 데이터베이스 활동을 조사할 수 있습니다. Time Travel은 CloudWatch 로깅과는 상관없이 독립적으로 작동합니다. CloudWatch 로깅에 관한 자세한 내용은 [작업 설정 로깅](#)을 참조하세요.

AWS DMS지원되는 Oracle, Microsoft SQL Server 및 PostgreSQL 소스 엔드포인트와 AWS DMS지원되는 PostgreSQL 및 MySQL 대상 엔드포인트가 있는 모든 AWS 리전에서 Time Travel을 사용할 수 있습니다. Time Travel은 전체 로드 및 변경 데이터 캡처(CDC) 작업과 CDC 전용 작업에만 사용할 수 있습니다. Time Travel을 켜거나 기존의 Time Travel 설정을 수정하려면 복제 작업을 중지해야 합니다.

Time Travel 설정에는 다음과 같은 TTSettings 속성이 포함됩니다.

- **EnableTT** – 이 옵션을 `true`로 설정하면 작업에 대한 Time Travel 로깅이 켜집니다. 기본값은 `false`입니다.

유형: 부울

필수 항목 여부: 아니요

- **EncryptionMode** – S3 버킷에서 데이터 및 로그를 저장하는 데 사용되는 서버 측 암호화 유형입니다. "SSE_S3"(기본값) 또는 "SSE_KMS"를 지정할 수 있습니다.

EncryptionMode를 "SSE_KMS"에서 "SSE_S3"으로 변경할 수 있지만 그 반대로 변경할 수는 없습니다.

유형: 문자열

필수 항목 여부: 아니요

- **ServerSideEncryptionKmsKeyId** - "SSE_KMS"에를 지정하는 경우 사용자 지정 관리형 AWS KMS 키의 ID를 EncryptionMode에 제공합니다. 사용하는 키에 AWS Identity and Access Management (IAM) 사용자 권한을 켜고 키 사용을 허용하는 연결된 정책이 있는지 확인합니다.

이 "SSE_KMS" 옵션에서는 사용자 지정 관리형 대칭 KMS 키만 지원됩니다.

유형: 문자열

필수: EncryptionMode이 "SSE_KMS"로 설정된 경우에만 해당됨

- **ServiceAccessRoleArn** – 서비스에서 IAM 역할에 액세스하는 데 사용되는 Amazon 리소스 이름(ARN)입니다. 역할 이름을 `dms-tt-s3-access-role`로 설정합니다. 이는 S3 버킷에서 객체 AWS DMS 를 쓰고 읽을 수 있도록 허용하는 필수 설정입니다.

유형: 문자열

필수: Time Travel이 켜진 경우

다음은 이 역할에 대한 정책 예제입니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```

    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "kms:GenerateDataKey",
        "kms:Decrypt",
        "s3:ListBucket",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::S3bucketName*",
        "arn:aws:kms:us-east-1:112233445566:key/1234a1a1-1m2m-1z2z-d1d2-12dmstt1234"
      ]
    }
  ]
}

```

다음은 이 역할에 대한 신뢰 정책 예제입니다.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "dms.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

- BucketName – Time Travel 로그를 저장할 S3 버킷의 이름입니다. Time Travel 로그를 켜기 전에 이 S3 버킷을 생성해야 합니다.

유형: 문자열

필수: Time Travel이 켜진 경우

- BucketFolder – S3 버킷의 폴더 이름을 설정하는 선택적 파라미터입니다. 이 파라미터를 지정하면 DMS는 경로 `"/BucketName/BucketFolder/taskARN/YYYY/MM/DD/hh"`에 Time Travel 로그를 생성합니다. 이 파라미터를 지정하지 않으면 기본 경로를 로 AWS DMS 생성합니다 `"/BucketName/dms-time-travel-logs/taskARN/YYYY/MM/DD/hh"`.

유형: 문자열

필수 항목 여부: 아니요

- EnableDeletingFromS3OnTaskDelete -이 옵션을 로 설정하면 작업이 true AWS DMS 삭제되면 S3에서 Time Travel 로그를 삭제합니다. 기본 값은 false입니다.

유형: 문자열

필수 항목 여부: 아니요

- EnableRawData – 이 옵션을 true로 설정하면 Time Travel 로그의 DML(데이터 조작 언어) 원시 데이터가 Time Travel 로그의 raw_data 열 아래에 나타납니다. 자세한 내용은 [Time Travel 로그 사용](#)을 참조하세요. 기본값은 false입니다. 이 옵션을 false로 설정하면 DML 유형만 캡처됩니다.

유형: 문자열

필수 항목 여부: 아니요

- RawDataFormat - AWS DMS 버전 3.5.0 이상에서 EnableRawData가 로 설정된 경우 true. 이 속성은 Time Travel 로그의 DML 원시 데이터 형식을 지정하며 다음과 같이 표시할 수 있습니다.
 - "TEXT" – CDC 중에 캡처한 DML 이벤트의 구문 분석된 읽을 수 있는 열 이름과 값을 Raw 필드로 표시합니다.
 - "HEX" – CDC 중에 DML 이벤트에 대해 캡처한 열 이름 및 값의 원래 16진수입니다.

이 속성은 Oracle 및 Microsoft SQL Server 데이터베이스 원본에 적용됩니다.

유형: 문자열

필수 항목 여부: 아니요

- OperationsToLog – Time Travel 로그에 로그인하기 위한 DML 작업의 유형을 지정합니다. 다음 중 한 가지를 지정할 수 있습니다.
 - "INSERT"
 - "UPDATE"

- "DELETE"
- "COMMIT"
- "ROLLBACK"
- "ALL"

기본값은 "ALL"입니다.

유형: 문자열

필수 항목 여부: 아니요

- MaxRecordSize - 각 행에 기록되는 Time Travel 로그 레코드의 최대 크기를 지정합니다. 이 속성을 사용하여 특히 사용량이 많은 테이블의 Time Travel 로그 증가를 제어할 수 있습니다. 기본값은 64KB입니다.

유형: 정수

필수 항목 여부: 아니요

Time Travel 로그를 켜고 사용하는 방법에 관한 자세한 내용은 다음 주제를 참조하세요.

주제

- [작업의 Time Travel 로그 켜기](#)
- [Time Travel 로그 사용](#)
- [가 Time Travel 로그를 S3에 AWS DMS 업로드하는 빈도](#)

작업의 Time Travel 로그 켜기

앞서 설명한 AWS DMS 작업 설정을 사용하여 작업에 대한 Time Travel을 켤 수 있습니다. Time Travel을 켜기 전에 복제 작업이 중지되었는지 확인하세요.

를 사용하여 Time Travel을 켜려면 AWS CLI

1. DMS 작업 구성 JSON 파일을 만들고 다음과 같은 TTSettings 섹션을 추가합니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

```

.
.
.

```

```

    },
    "TTSettings" : {
      "EnableTT" : true,
      "TTS3Settings": {
        "EncryptionMode": "SSE_KMS",
        "ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-west-2:112233445566:key/
myKMSKey",
        "ServiceAccessRoleArn": "arn:aws:iam::112233445566:role/dms-tt-s3-access-
role",
        "BucketName": "myttbucket",
        "BucketFolder": "myttfolder",
        "EnableDeletingFromS3OnTaskDelete": false
      },
      "TTRecordSettings": {
        "EnableRawData" : true,
        "OperationsToLog": "DELETE,UPDATE",
        "MaxRecordSize": 64
      },
    },
    .
    .
    .

```

2. 적절한 작업 동작에서 `--replication-task-settings` 옵션을 사용하여 이 JSON 파일을 지정합니다. 예를 들어, 다음 CLI 코드 조각은 이 Time Travel 설정 파일을 `create-replication-task`의 일부로 지정합니다.

```

aws dms create-replication-task
--target-endpoint-arn arn:aws:dms:us-
east-1:112233445566:endpoint:ELS507YTYV452CAZR2EYBNQGILFHQIFVPWFRQAY \
--source-endpoint-arn arn:aws:dms:us-
east-1:112233445566:endpoint:HNX2BWIIN5ZYFF7F6UFFZVWTDFFSMTNOV2FTXZA \
--replication-instance-arn arn:aws:dms:us-
east-1:112233445566:rep:ERLHG2UA52EEJJKFYNYWRPCG6T7EPUAB5AWBUJQ \
--migration-type full-load-and-cdc --table-mappings 'file:///FilePath/
mappings.json' \
--replication-task-settings 'file:///FilePath/task-settings-tt-enabled.json' \
--replication-task-identifier test-task
.
.
.

```


여기서 이 Time Travel 설정 파일의 이름은 `task-settings-tt-enabled.json`입니다.

마찬가지로 이 파일을 `modify-replication-task` 동작의 일부로 지정할 수 있습니다.

다음과 같은 작업 동작에 대한 Time Travel 로그의 특수 처리에 유의하세요.

- `start-replication-task` – 복제 작업을 실행할 때 Time Travel에 사용되는 S3 버킷에 액세스할 수 없는 경우 작업은 FAILED로 표시됩니다.
- `stop-replication-task` - 작업이 중지되면 복제 인스턴스에 현재 사용 가능한 모든 Time Travel 로그를 Time Travel에 사용되는 S3 버킷으로 AWS DMS 즉시 푸시합니다.

복제 작업이 실행되는 동안 EncryptionMode 값을 "SSE_KMS"에서 "SSE_S3"로 변경할 수 있지만 그 반대로 변경할 수는 없습니다.

진행 중인 작업의 Time Travel 로그 크기가 1GB를 초과하는 경우, DMS는 해당 크기에 도달한 후 5 분 이내에 로그를 S3로 푸시합니다. 작업이 실행된 후 S3 버킷 또는 KMS 키에 액세스할 수 없게 되면 DMS는 이 버킷으로의 로그 푸시를 중단합니다. 로그가 S3 버킷으로 푸시되지 않는 경우 S3 및 AWS KMS 권한을 확인합니다. DMS가 이러한 로그를 S3로 푸시하는 빈도에 관한 자세한 내용은 [가 Time Travel 로그를 S3에 AWS DMS 업로드하는 빈도](#)를 참조하세요.

콘솔에서 기존 작업에 대한 Time Travel을 켜려면 작업 설정에서 JSON 편집기 옵션을 사용하여 TTSettings 섹션을 추가하세요.

Time Travel 로그 사용

Time Travel 로그 파일은 다음과 같은 필드가 있는 쉼표로 구분된 값(CSV) 파일입니다.

```
log_timestamp
component
dms_source_code_location
transaction_id
event_id
event_timestamp
lsn/scn
primary_key
record_type
event_type
schema_name
table_name
```

```
statement
action
result
raw_data
```

S3에서 Time Travel 로그를 사용할 수 있게 되면 Amazon Athena와 같은 도구를 사용하여 이 로그에 직접 액세스하고 쿼리할 수 있습니다. 또는 S3에서 다른 파일처럼 로그를 다운로드할 수 있습니다.

다음 예제는 mytable이라는 테이블의 트랜잭션이 기록되는 Time Travel 로그를 보여줍니다. 가독성을 위해 다음 로그의 줄 끝을 추가했습니다.

```
"log_timestamp ","tt_record_type","dms_source_code_location ","transaction_id",
"event_id","event_timestamp","scn_lsn","primary_key","record_type","event_type",
"schema_name","table_name","statement","action","result","raw_data"
"2021-09-23T01:03:00:778230","SOURCE_CAPTURE","postgres_endpoint_wal_engine.c:00819",
"609284109","565612992","2021-09-23 01:03:00.765321+00","00000E9C/D53AB518","","DML",
"UPDATE (3)","dmstest","mytable","","Migrate","","table dmstest.mytable:
UPDATE: id[bigint]:2244937 phone_number[character varying]:'phone-number-482'
age[integer]:82 gender[character]:'f' isactive[character]:'true '
date_of_travel[timestamp without time zone]:'2021-09-23 01:03:00.76593'
description[text]:'TEST DATA TEST DATA TEST DATA TEST DATA'"
```

가 Time Travel 로그를 S3에 AWS DMS 업로드하는 빈도

복제 인스턴스의 스토리지 사용을 최소화하기 위해서는 주기적으로 Time Travel 로그를 AWS DMS 오프로드합니다.

다음과 같은 경우 Time Travel 로그가 Amazon S3 버킷으로 푸시됩니다.

- 현재 로그 크기가 1GB를 초과하는 경우는 5분 이내에 로그를 S3에 AWS DMS 업로드합니다. 따라서 AWS DMS 는 실행 중인 각 작업에 AWS KMS 대해 S3 및에 대해 시간당 최대 12개의 호출을 수행할 수 있습니다.
- AWS DMS 는 로그 크기에 관계없이 매시간 S3에 로그를 업로드합니다.
- 작업이 중지되면는 시간 이동 로그를 S3에 AWS DMS 즉시 업로드합니다.

작업 설정 로깅

로깅 기능은 마이그레이션 프로세스 중에 Amazon CloudWatch를 사용하여 정보를 로그합니다. 작업 설정 로깅을 사용하면 로깅할 구성 요소 활동과 로그에 작성할 정보량을 지정할 수 있습니다. 작업 설

정 로깅은 JSON 파일에 작성됩니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

CloudWatch 로깅은 여러 가지 방법으로 활성화할 수 있습니다. 마이그레이션 작업을 생성할 AWS Management Console 때에서 EnableLogging 옵션을 선택할 수 있습니다. 또는 AWS DMS API를 사용하여 작업을 생성할 때 true EnableLogging 옵션을 로 설정할 수 있습니다. 작업 설정의 JSON 로깅 섹션에서 "EnableLogging": true를 지정할 수도 있습니다.

를 EnableLogging로 설정하면 다음과 같이 CloudWatch 그룹 이름과 스트림 이름을 true AWS DMS 할당합니다. 이러한 값을 직접 설정할 수 없습니다.

- CloudWatchLogGroup: dms-tasks-<REPLICATION_INSTANCE_IDENTIFIER>
- CloudWatchLogStream: dms-task-<REPLICATION_TASK_EXTERNAL_RESOURCE_ID>

<REPLICATION_INSTANCE_IDENTIFIER>는 복제 인스턴스의 식별자입니다.

<REPLICATION_TASK_EXTERNAL_RESOURCE_ID>는 태스크 ARN의 <resourcename> 섹션의 값입니다. 가 리소스 ARN을 AWS DMS 생성하는 방법에 대한 자세한 내용은 [섹션을 참조하세요](#) [에 대한 Amazon 리소스 이름\(ARN\) 구성 AWS DMS](#). ARNs

CloudWatch는 AWS Identity and Access Management (IAM)과 통합되며 AWS , 계정의 사용자가 수행할 수 있는 CloudWatch 작업을 지정할 수 있습니다. CloudWatch에서 IAM을 사용하는 방법에 관한 자세한 내용은 Amazon CloudWatch 사용 설명서의 [Amazon CloudWatch의 자격 증명 및 액세스 관리](#)와 [Amazon CloudWatch API 직접 호출 로깅](#)을 참조하세요.

작업 로그를 삭제하려면 작업 설정의 로깅 섹션 중 JSON에서 DeleteTaskLogs를 true로 설정하면 됩니다.

다음 유형의 이벤트에 대한 로깅을 지정할 수 있습니다.

- FILE_FACTORY – 파일 팩토리는 배치 적용 및 배치 로드에서 사용되는 파일을 관리하고 Amazon S3 엔드포인트를 관리합니다.
- METADATA_MANAGER – 메타데이터 관리자는 복제 중에 소스 및 대상 메타데이터, 파티셔닝, 테이블 상태를 관리합니다.
- SORTER – SORTER는 SOURCE_CAPTURE 프로세스에서 들어오는 이벤트를 수신합니다. 이벤트는 트랜잭션으로 일괄 처리되어 TARGET_APPLY 서비스 구성 요소로 전달됩니다. SOURCE_CAPTURE 프로세스에서 TARGET_APPLY 구성 요소가 이벤트를 소비하는 속도보다 더 빠르게 이벤트를 생성하는 경우, SORTER 구성 요소는 백로그된 이벤트를 디스크나 스왑 파일에 캐시합니다. 캐시된 이벤트는 복제 인스턴스에서 스토리지 부족 문제를 일으키는 공통적 원인에 속합니다.

SORTER 서비스 구성 요소는 캐시된 이벤트를 관리하고 CDC 통계를 수집하며 작업 지연 시간을 보고합니다.

- SOURCE_CAPTURE – 지속적 복제(CDC) 데이터는 소스 데이터베이스 또는 서비스에서 캡처되어 SORTER 서비스 구성 요소로 전달됩니다.
- SOURCE_UNLOAD – 전체 로드 중에 소스 데이터베이스 또는 서비스에서 데이터가 언로드됩니다.
- TABLES_MANAGER – 테이블 관리자는 캡처한 테이블을 추적하고 테이블 마이그레이션 순서를 관리하며 테이블 통계를 수집합니다.
- TARGET_APPLY – 데이터와 데이터 정의 언어(DDL) 문을 대상 데이터베이스에 적용합니다.
- TARGET_LOAD – 데이터는 대상 데이터베이스로 로드됩니다.
- TASK_MANAGER – 작업 관리자는 실행 중인 작업을 관리하고 병렬 데이터 처리를 위해 작업을 하위 작업으로 분류합니다.
- TRANSFORMATION – 테이블 매핑 변환 이벤트. 자세한 내용은 [작업 설정을 지정하기 위한 테이블 매핑 사용](#) 단원을 참조하십시오.
- VALIDATOR/ VALIDATOR_EXT – VALIDATOR 서비스 구성 요소는 데이터가 소스에서 타겟으로 정확하게 마이그레이션되었는지 확인합니다. 자세한 내용은 [데이터 유효성 검사](#) 단원을 참조하십시오.
- DATA_RESYNC - 데이터 재동기화 흐름을 관리하는 데이터 재동기화 기능의 일반적인 구성 요소입니다. 자세한 내용은 [the section called “AWS DMS 데이터 재동기화”](#) 단원을 참조하십시오.
- RESYNC_UNLOAD – 재동기화 프로세스 중에 소스 데이터베이스 또는 서비스에서 데이터가 언로드됩니다.
- RESYNC_APPLY - 데이터 조작 언어(DML) 문은 재동기화 중에 대상 데이터베이스에 적용됩니다.

다음 로깅 구성 요소는 LOGGER_SEVERITY_DETAILED_DEBUG 로그 심각도 수준을 사용할 때 대량의 로그를 생성합니다.

- COMMON
- ADDONS
- DATA_STRUCTURE
- COMMUNICATION
- FILE_TRANSFER
- FILE_FACTORY

문제 해결 중에 이러한 구성 요소에 대해 DEFAULT 이외의 로깅 수준은 거의 필요하지 않습니다. AWS Support에서 특별히 요청하지 않는 한 DEFAULT 이러한 구성 요소의 로깅 수준을에서 변경하지 않는 것이 좋습니다.

위의 항목 중 하나를 지정한 후 다음 목록과 같이 기록되는 정보의 양을 지정할 수 있습니다.

심각도 수준이 가장 낮은 정보에서 가장 높은 수준의 정보 순으로 표시됩니다. 높은 수준은 항상 낮은 수준의 정보를 포함합니다.

- `LOGGER_SEVERITY_ERROR` – 오류 메시지가 로그에 작성됩니다.
- `LOGGER_SEVERITY_WARNING` – 경고와 오류 메시지가 로그에 작성됩니다.
- `LOGGER_SEVERITY_INFO` – 정보 메시지, 경고 및 오류 메시지가 로그에 작성됩니다.
- `LOGGER_SEVERITY_DEFAULT` – 정보 메시지, 경고 및 오류 메시지가 로그에 작성됩니다.
- `LOGGER_SEVERITY_DEBUG` – 디버그 메시지, 정보 메시지, 경고 및 오류 메시지가 로그에 작성됩니다.
- `LOGGER_SEVERITY_DETAILED_DEBUG` – 모든 정보가 로그에 작성됩니다.

다음 JSON 예제에서는 모든 작업 및 심각도 수준을 로깅하기 위한 작업 설정을 보여줍니다.

```
...
"Logging": {
  "EnableLogging": true,
  "LogComponents": [
    {
      "Id": "FILE_FACTORY",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    }, {
      "Id": "METADATA_MANAGER",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    }, {
      "Id": "SORTER",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    }, {
      "Id": "SOURCE_CAPTURE",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    }, {
      "Id": "SOURCE_UNLOAD",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    }, {
      "Id": "TABLES_MANAGER",
```

```

    "Severity": "LOGGER_SEVERITY_DEFAULT"
  }, {
    "Id": "TARGET_APPLY",
    "Severity": "LOGGER_SEVERITY_DEFAULT"
  }, {
    "Id": "TARGET_LOAD",
    "Severity": "LOGGER_SEVERITY_INFO"
  }, {
    "Id": "TASK_MANAGER",
    "Severity": "LOGGER_SEVERITY_DEBUG"
  }, {
    "Id": "TRANSFORMATION",
    "Severity": "LOGGER_SEVERITY_DEBUG"
  }, {
    "Id": "VALIDATOR",
    "Severity": "LOGGER_SEVERITY_DEFAULT"
  }
],
"CloudWatchLogGroup": null,
"CloudWatchLogStream": null
},
...

```

제어 테이블 작업 설정

제어 테이블은 AWS DMS 작업에 대한 정보를 제공합니다. 또한 현재 마이그레이션 작업과 향후 작업 모두를 계획하고 관리하는 데 사용할 수 있는 유용한 통계가 나와 있습니다. JSON 파일에서 또는 AWS DMS 콘솔의 태스크 생성 페이지에서 고급 설정을 선택하여 이러한 태스크 설정을 적용할 수 있습니다. 예외 적용 테이블(`dmslogs.awsdms_apply_exceptions`)은 데이터베이스 대상에 항상 생성됩니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

AWS DMS 는 전체 로드 + CDC 또는 CDC 전용 작업 중에만 제어 테이블을 생성하고 전체 로드 전용 작업 중에는 생성하지 않습니다.

전체 로드 및 CDC(기존 데이터 마이그레이션 및 진행 중인 변경 사항 복제) 및 CDC 전용(데이터 변경 사항만 복제) 작업의 경우, 다음을 비롯한 추가 테이블을 만들 수도 있습니다.

- 복제 상태(`dmslogs.awsdms_status`) – 이 테이블은 현재 작업에 대한 세부 정보를 제공합니다. 작업 상태, 작업에서 사용하는 메모리 크기, 대상에 아직 적용되지 않은 변경 수 등이 표시됩니다. 또한 이 테이블은 현재 읽 AWS DMS 는 소스 데이터베이스의 위치를 제공합니다. 또한 이것은 작업이 전체 로드 단계이거나 변경 데이터 캡처(CDC)인지 여부를 나타냅니다.

- 일시 중지된 테이블(dmslogs.aws_dms_suspended_tables) – 이 테이블에는 일시 중지된 테이블뿐 아니라 일시 중지된 이유가 나와 있습니다.
- 복제 기록(dmslogs.aws_dms_history) – 이 테이블은 복제 기록에 관한 정보를 제공합니다. 작업 중에 처리된 레코드의 수와 볼륨, CDC 작업 종료 시 지연 시간, 기타 통계 등이 표시됩니다.

예외 적용 테이블(dmslogs.aws_dms_apply_exceptions)에는 다음 파라미터가 포함되어 있습니다.

열	Type	설명
TASK_NAME	nvchar	AWS DMS 작업의 리소스 ID입니다. 리소스 ID는 태스크 ARN에서 찾을 수 있습니다.
TABLE_OWNER	nvchar	테이블 소유자입니다.
TABLE_NAME	nvchar	테이블 이름.
ERROR_TIME	timestamp	예외(오류)가 발생한 시간입니다.
STATEMENT	nvchar	오류가 발생했을 때 실행되고 있었던 문입니다.
오류	nvchar	오류 이름 및 설명입니다.

복제 상태 테이블(dmslogs.aws_dms_status)에는 작업과 대상 데이터베이스의 현재 상태가 포함되어 있습니다. 설정은 다음과 같습니다.

열	Type	설명
SERVER_NAME	nvchar	복제 작업이 실행되고 있는 시스템의 이름입니다.
TASK_NAME	nvchar	AWS DMS 작업의 리소스 ID입니다. 리소스 ID는 태스크 ARN에서 찾을 수 있습니다.
TASK_STATUS	varchar	다음 값 중 하나입니다.

열	Type	설명
		• FULL LOAD • CHANGE PROCESSING(CDC) • NOT RUNNING 전체 로드에서 최소 1개 이상의 테이블이 있는 한 작업 상태가 FULL LOAD로 설정됩니다. 모든 테이블이 로드된 후 CDC가 활성화되어 있으면 작업 상태가 CHANGE PROCESSING으로 바뀝니다. 태스크를 시작하기 전에 또는 태스크가 완료된 후에 태스크가 NOT RUNNING으로 설정됩니다.
STATUS_TIME	timestamp	작업 상태의 타임스탬프.
PENDING_CHANGES	int	소스 데이터베이스에서 커밋되고 복제 인스턴스의 메모리와 디스크에 캐시된 변경 레코드의 수입니다.
DISK_SWAP_SIZE	int	이전 또는 오프로드된 트랜잭션에서 사용하는 디스크 공간 크기입니다.
TASK_MEMORY	int	사용된 현재 메모리(단위: MB).
SOURCE_CURRENT_POSITION	varchar	현재 읽는 소스 데이터베이스의 위치 AWS DMS입니다.
SOURCE_CURRENT_TIMESTAMP	timestamp	현재 읽는 소스 데이터베이스의 타임스탬프 AWS DMS입니다.

열	Type	설명
SOURCE_TAIL _POSITION	varchar	커밋되지 않은 가장 오래된 시작 트랜잭션의 위치입니다. 이 값은 변경 사항을 잃지 않고도 되돌릴 수 있는 가장 최근 위치입니다.
SOURCE_TAIL _TIMESTAMP	timestamp	커밋되지 않은 가장 오래된 시작 트랜잭션의 타임스탬프입니다. 이 값은 변경 사항을 잃지 않고도 되돌릴 수 있는 가장 최근 타임스탬프입니다.
SOURCE_TI MESTAMP _APPLIED	timestamp	마지막 트랜잭션의 타임스탬프가 커밋됩니다. 대량 적용 프로세스에서 이 값은 대량으로 마지막 트랜잭션을 커밋하기 위한 타임스탬프입니다.

일시 중단된 테이블(dmslogs.aws_dms_suspended_tables)에는 다음과 같은 파라미터가 포함되어 있습니다.

열	Type	설명
SERVER_NAME	nvchar	복제 작업이 실행되고 있는 시스템의 이름입니다.
TASK_NAME	nvchar	AWS DMS 작업의 이름입니다.
TABLE_OWNER	nvchar	테이블 소유자입니다.
TABLE_NAME	nvchar	테이블 이름.
SUSPEND_REASON	nvchar	일시 중지 사유.
SUSPEND_T IMESTAMP	timestamp	일시 중단이 발생한 시간입니다.

복제 기록 테이블(dmslogs.awsdms_history)에는 다음 파라미터가 포함되어 있습니다.

열	Type	설명
SERVER_NAME	nvchar	복제 작업이 실행되고 있는 시스템의 이름입니다.
TASK_NAME	nvchar	AWS DMS 작업의 리소스 ID입니다. 리소스 ID는 태스크 ARN에서 찾을 수 있습니다.
TIMESLOT_TYPE	varchar	다음 값 중 하나입니다. - FULL LOAD - CHANGE PROCESSING(CDC) 작업이 전체 로드와 CDC를 실행하고 있는 경우, 두 이력 레코드는 타임 슬롯에 작성됩니다.
TIMESLOT	timestamp	시간 슬롯의 종료 타임스탬프입니다.
TIMESLOT_DURATION	int	시간 슬롯의 지속 시간(분)입니다.
TIMESLOT_LATENCY	int	시간 슬롯이 끝날 때의 대상 지연 시간(초)입니다. 이 값은 CDC 시간 슬롯에만 적용됩니다.
RECORDS	int	시간 슬롯 중에 처리되는 레코드의 수입니다.
TIMESLOT_VOLUME	int	처리되는 데이터의 볼륨(단위: MB)입니다.

검증 실패 테이블(awsdms_validation_failures_v1)에는 작업에 대한 모든 데이터 검증 실패가 포함되어 있습니다. 자세한 내용은 [데이터 검증 문제 해결](#)을 참조하세요.

추가 제어 테이블 설정에는 다음이 포함됩니다.

- **HistoryTimeslotInMinutes** – 이 옵션을 사용하여 복제 이력 테이블에서 각 시간 슬롯의 길이를 나타냅니다. 기본값은 5분입니다.
- **ControlSchema** - AWS DMS 대상에 대한 제어 테이블의 데이터베이스 스키마 이름을 표시하려면 이 옵션을 사용합니다. 이 옵션에 대한 정보를 입력하지 않으면 테이블이 다음과 같이 데이터베이스의 기본 위치로 복사됩니다.
 - PostgreSQL, 퍼블릭
 - Oracle, 대상 스키마
 - Microsoft SQL 서버, 대상 데이터베이스에 dbo
 - MySQL, awsdms_control
 - MariaDB, awsdms_control
 - Amazon Redshift, 퍼블릭
 - DynamoDB, 데이터베이스에 개별 테이블로 생성됨
 - IBM DB2 LUW, awsdms_control

스트림 버퍼 작업 설정

다음에 포함하여 AWS CLI를 사용하여 스트림 버퍼 설정을 지정할 수 있습니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

- **StreamBufferCount** – 이 옵션을 사용하여 마이그레이션 작업을 위한 데이터 스트림 버퍼의 수를 지정합니다. 기본 스트림 버퍼 개수는 3개입니다. 이 설정의 값이 증가하면 데이터 추출 속도가 증가할 수 있습니다. 그렇지만 이 성능 증가는 원본 시스템과 복제 서버의 인스턴스 클래스를 포함한 마이그레이션 환경에 따라 크게 달라집니다. 대부분의 상황에서는 기본값으로 충분합니다.
- **StreamBufferSizeInMB** – 이 옵션을 사용하여 각 데이터 스트림 버퍼의 최대 크기를 나타냅니다. 기본 크기는 8MB입니다. 매우 큰 LOB를 사용하는 경우 이 옵션의 값을 늘려야 할 수 있습니다. 로그 파일에 스트림 버퍼 크기가 부족하다는 메시지가 표시될 경우 이 값을 증가시켜야 할 수도 있습니다. 이 옵션의 크기를 계산하려면 다음 수식을 사용합니다. $[\text{Max LOB size (or LOB chunk size)}] \times [\text{number of LOB columns}] \times [\text{number of stream buffers}] \times [\text{number of tables loading in parallel per task}(\text{MaxFullLoadSubTasks})] \times 3$
- **CtrlStreamBufferSizeInMB** – 이 옵션을 사용하여 제어 스트림 버퍼의 크기를 설정합니다. 값의 단위는 메가바이트이고 범위는 1~8입니다. 기본값은 5입니다. 예를 들어, 수천 개에 이르는 많은 테이블을 사용할 경우 이 값을 증가시켜야 할 수도 있습니다.

변경 처리 튜닝 설정

다음 설정은가 변경 데이터 캡처(CDC) 중에 대상 테이블에 대한 변경 사항을 AWS DMS 처리하는 방법을 결정합니다. 이 설정 중 몇 개는 대상 메타데이터 파라미터 BatchApplyEnabled의 값에 따라 달라집니다. BatchApplyEnabled 파라미터에 관한 자세한 내용은 [대상 메타데이터 작업 설정](#)를 참조하세요. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

변경 처리 튜닝 설정에는 다음이 포함됩니다.

다음 설정은 대상 메타데이터 파라미터 BatchApplyEnabled가 true로 설정되어 있는 경우에만 적용됩니다.

- BatchApplyPreserveTransaction – true로 설정하면 트랜잭션 무결성이 유지되고 배치에서는 원본의 트랜잭션 내에 있는 모든 변경 사항이 포함됩니다. 기본값은 true입니다. 이 설정은 Oracle 대상 엔드포인트에만 적용됩니다.

false로 설정되면 트랜잭션 무결성이 일시적인 시간 경과가 있어 성능이 개선될 수 있습니다. 원본 트랜잭션의 모든 변경 사항이 단일한 배치(batch)의 대상에 적용된다는 보장은 없습니다.

기본적으로는 트랜잭션 모드에서 변경 사항을 AWS DMS 처리하여 트랜잭션 무결성을 유지합니다. 트랜잭션 무결성이 일시적으로 저하될 여지가 있다면 배치 최적화 적용 옵션을 대신 사용할 수 있습니다. 이 옵션은 효율성을 위해 트랜잭션을 효율적으로 그룹화하고 일괄적으로 적용합니다. 배치 최적화 적용 옵션을 사용하면 거의 항상 참조 무결성 제약 조건을 위반합니다. 따라서 마이그레이션 프로세스 중에 이러한 제약 조건을 해제한 다음, 전환 프로세스의 일부로 다시 설정하는 것이 좋습니다.

- BatchApplyTimeoutMin - 배치 변경의 각 적용 간에 AWS DMS 대기하는 최소 시간을 초 단위로 설정합니다. 기본값은 1입니다.
- BatchApplyTimeoutMax - 시간이 초과되기 전에 배치 변경의 각 적용 간에 AWS DMS 대기하는 최대 시간을 초 단위로 설정합니다. 기본값은 30입니다.
- BatchApplyMemoryLimit – 배치 최적화 적용 모드에서 사전 처리를 위해 사용하는 최대 메모리 양(MB)을 설정합니다. 기본값은 500입니다.
- BatchSplitSize – 단일 배치에서 적용되는 변경 사항의 최대 수를 설정합니다. 기본값이 0이면 적용되는 제한이 없음을 뜻합니다.

다음 설정은 대상 메타데이터 파라미터 BatchApplyEnabled가 false로 설정되어 있는 경우에만 적용됩니다.

- **MinTransactionSize** – 각 트랜잭션에 포함할 변경 사항의 최소 수를 설정합니다. 기본값은 1000입니다.
- **CommitTimeout** - 제한 시간을 선언 AWS DMS 하기 전에가 트랜잭션을 배치로 수집하는 최대 시간을 초 단위로 설정합니다. 기본값은 1입니다.

양방향 복제의 경우, 다음 설정은 대상 메타데이터 파라미터 **BatchApplyEnabled**가 **false**로 설정되어 있는 경우에만 적용됩니다.

- **LoopbackPreventionSettings** – 이러한 설정은 양방향 복제와 관련된 작업 쌍으로 된 지속적 복제 작업마다 루프백 방지를 제공합니다. 루프백 방지는 동일한 변경 사항이 양방향 복제의 두 방향 모두에 적용되어 데이터가 손상되는 문제를 방지합니다. 양방향 복제에 관한 자세한 내용은 [양방향 복제 수행](#) 섹션을 참조하세요.

AWS DMS 는 트랜잭션이 소스, 대상 또는 둘 다에 완전히 커밋될 때까지 트랜잭션 데이터를 메모리에 유지하려고 시도합니다. 그렇지만 할당된 메모리보다 크거나 지정된 시간 한도 내에서 커밋되지 않은 트랜잭션은 디스크에 기록됩니다.

다음 설정은 변경 처리 모드에 상관 없이 변경 처리 튜닝에 적용됩니다.

- **MemoryLimitTotal** – 디스크에 기록되기 전에 모든 트랜잭션이 메모리에서 점유할 수 있는 최대 크기(MB 단위)를 설정합니다. 기본값은 1,024입니다.
- **MemoryKeepTime** – 디스크에 기록되기 전에 각 트랜잭션이 메모리에 유지될 수 있는 최대 시간(초)을 설정합니다. 지속 시간은가 트랜잭션 캡처를 AWS DMS 시작한 시간부터 계산됩니다. 기본값은 60입니다.
- **StatementCacheSize** – 변경 사항을 대상에 적용할 때 나중에 실행할 수 있도록 서버에서 저장하는 준비된 문의 최대 수를 설정합니다. 기본값은 50이고 최대값은 200입니다.
- **RecoveryTimeout** - CDC 모드에서 작업을 재개할 때 **RecoveryTimeout**은 작업이 재개 체크포인트에 도달하기 위해 대기하는 시간(분)을 제어합니다. 구성된 기간 내에 체크포인트가 발생하지 않으면 작업이 실패합니다. 기본 동작은 체크포인트 이벤트를 무시한 기다리는 것입니다.

변경 처리 튜닝을 처리하는 작업 설정이 작업 설정 JSON 파일에 표시되는 방식의 한 가지 예:

```
"ChangeProcessingTuning": {
  "BatchApplyPreserveTransaction": true,
  "BatchApplyTimeoutMin": 1,
  "BatchApplyTimeoutMax": 30,
  "BatchApplyMemoryLimit": 500,
```

```

    "BatchSplitSize": 0,
    "MinTransactionSize": 1000,
    "CommitTimeout": 1,
    "MemoryLimitTotal": 1024,
    "MemoryKeepTime": 60,
    "StatementCacheSize": 50
    "RecoveryTimeout": -1
}

```

데이터 복제 작업 중에 Amazon S3 대상에 대한 쓰기 빈도를 제어하기 위해 `cdcMaxBatchInterval` 및 `cdcMinFileSize` 추가 연결 속성을 구성할 수 있습니다. 따라서 추가 오버헤드 작업 없이 데이터를 분석할 때 성능이 향상될 수 있습니다. 자세한 내용은 [Amazon S3를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

데이터 검증 작업 설정

원본에서 대상으로 데이터가 정확히 마이그레이션되었는지 확인할 수 있습니다. 작업에 대한 검증을 활성화하면 테이블에 대한 전체 로드가 수행된 직후 소스와 대상 데이터를 비교하기 AWS DMS 시작합니다. 작업 데이터 검증, 요구 사항, 데이터베이스 지원 범위, 보고 대상 지표에 관한 자세한 내용은 [AWS DMS 데이터 검증](#) 섹션을 참조하세요. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

데이터 검증 설정과 그 값에는 다음 사항이 포함됩니다.

- `EnableValidation` – `true`로 설정한 경우 데이터 검증을 활성화합니다. 그렇지 않으면 작업에 대한 검증은 비활성화됩니다. 기본값은 `false`입니다.
- `ValidationMode` -가 소스 테이블에 대해 대상 테이블의 데이터를 AWS DMS 검증하는 방법을 제어합니다. 복제 엔진 버전 3.5.4부터 DMS는 지원되는 마이그레이션 경로에 대해 이를 `GROUP_LEVEL`로 자동 설정하여 대규모 데이터세트에 대해 향상된 검증 성능과 훨씬 빠른 처리를 제공합니다. 이 개선 사항은 [AWS DMS 데이터 재동기화](#)에 나열된 마이그레이션 경로의 마이그레이션에 적용됩니다. 다른 모든 마이그레이션 경로의 경우 검증 모드의 기본값은 `ROW_LEVEL`입니다.

Note

설정에 관계없이 테이블 AWS DMS 검증을 통해 구성된 모든 행을 검증합니다.

- `FailureMaxCount` – 작업에 대한 검증이 일시 중지되기 전에 검증에 실패할 수 있는 레코드의 최대 수를 지정합니다. 기본값은 10,000입니다. 검증에 실패하는 레코드 수와 상관없이 검증을 계속하려면 이 값을 원본의 레코드 수보다 더 높게 설정합니다.

- **HandleCollationDiff** - 이 옵션을 `true`로 설정하면 검증은 소스 데이터베이스와 대상 데이터베이스 간의 열 데이터 정렬 차이를 고려합니다. 그렇지 않은 경우 열 기준 데이터의 이러한 차이는 검증에서 무시됩니다. 열 기준 데이터 정렬은 데이터 검증에 중요한 행의 순서를 지정할 수 있습니다. **HandleCollationDiff**를 `true`로 설정하면 이러한 데이터 정렬 차이가 자동으로 해결되고 데이터 검증에서 거짓 양성이 방지됩니다. 기본값은 `false`입니다.
- **RecordFailureDelayInMinutes** - 검증 실패 세부 정보를 보고하기 전에 지연 시간을 분 단위로 지정합니다.

전체 DMS 작업 CDC 지연 시간이 **RecordFailureDelayInMinutes**의 값보다 큰 경우 우선합니다. 예를 들어 **RecordFailureDelayInMinutes**가 5이고 CDC 지연 시간이 7분인 경우 DMS는 7분 동안 대기하여 검증 실패 세부 정보를 보고합니다.

- **RecordFailureDelayLimitInMinutes** - 검증 실패 세부 정보를 보고하기 전에 지연을 지정합니다.는 작업 지연 시간을 AWS DMS 사용하여 오탐을 방지하기 위해 대상에 대한 변경 사항의 실제 지연을 인식합니다. 이 설정은 실제 지연 시간과 DMS 작업 CDC 지연 값을 재정의하고 검증 지표를 보고하기 전에 더 긴 지연 시간을 설정할 수 있도록 합니다. 기본값은 0입니다.
- **RecordSuspendDelayInMinutes** - **FailureMaxCount**에 설정된 오류 임계값으로 인해 테이블 검증이 일시 중단될 때까지의 지연 시간(분)을 지정합니다.
- **SkipLobColumns** -이 옵션을 `로` 설정하면 작업 검증의 테이블 부분에 있는 모든 LOB 열에 대한 데이터 검증을 `true` AWS DMS 건너뛸니다. 기본값은 `false`입니다.
- **TableFailureMaxCount** - 테이블에 대한 검증이 일시 중지되기 전에 하나의 테이블에서 검증에 실패할 수 있는 행의 최대 수를 지정합니다. 기본값은 1,000입니다.
- **ThreadCount** - 검증 중에가 AWS DMS 사용하는 실행 스레드 수를 지정합니다. 각 스레드는 비교 및 검증할 원본과 대상에서 아직 검증되지 않은 데이터를 선택합니다. 기본값은 5입니다. 를 더 높은 숫자**ThreadCount**로 설정하면가 검증을 더 빨리 완료할 AWS DMS 수 있습니다. 하지만 AWS DMS는 동시 쿼리를 더 많이 실행하여 소스 및 대상에서 더 많은 리소스를 사용합니다.
- **ValidationOnly** - 이 옵션이 `true`로 설정되면 작업을 실행하는 경우 데이터 마이그레이션이나 복제를 수행하지 않고 데이터 검증을 미리 봅니다. 기본값은 `false`입니다. 작업이 생성된 후에는 이 **ValidationOnly** 설정을 수정할 수 없습니다.

TargetTablePrepMode를 **D0_NOTHING**(검증 전용 작업의 기본값)으로 설정하고 마이그레이션 유형을 다음 중 하나로 설정해야 합니다.

- 전체 로드 - 태스크 마이그레이션 유형을 설정하여 AWS DMS 콘솔의 기존 데이터를 마이그레이션합니다. 또는 AWS DMS API에서 마이그레이션 유형을 **FULL-LOAD**로 설정합니다.
- CDC - AWS DMS 콘솔에서 마이그레이션 유형 작업을 데이터 변경만 복제로 설정합니다. 또는 AWS DMS API에서 마이그레이션 유형을 **CDC**로 설정합니다.

선택한 마이그레이션 유형에 관계없이 검증 전용 작업 중에는 데이터가 실제로 마이그레이션되거나 복제되지 않습니다.

자세한 내용은 [검증 전용 태스크](#) 단원을 참조하십시오.

Important

ValidationOnly 설정은 변경할 수 없습니다. 작업이 생성된 후에는 해당 작업에 대해 수정할 수 없습니다.

- **ValidationPartialLobSize** – 열에 저장된 모든 데이터를 검증하는 대신 LOB 열에 대한 부분 검증을 수행할지 여부를 지정합니다. 이는 전체 LOB 데이터 집합이 아닌 일부 LOB 데이터만 마이그레이션할 때 유용할 수 있습니다. 값의 단위는 KB입니다. 기본값은 0이며, 이는 AWS DMS 가 모든 LOB 열 데이터를 검증함을 의미합니다. 예를 들어는가 소스와 대상 모두에서 열 데이터의 처음 32KB AWS DMS 만 검증함을 "ValidationPartialLobSize": 32 의미합니다.
- **PartitionSize** – 원본과 대상 모두에서 비교하기 위해 읽을 레코드의 배치 크기를 지정합니다. 기본값은 10,000입니다.
- **ValidationQueryCdcDelaySeconds** – 각 CDC 업데이트에 대해 원본과 대상 모두에서 첫 번째 유효성 검사 쿼리가 지연되는 시간입니다. 이렇게 하면 마이그레이션 지연 시간이 길어질 때 리소스 경합을 줄이는 데 도움이 될 수 있습니다. 검증 전용 작업에서는 이 옵션이 180초로 자동 설정됩니다. 기본값은 0입니다.

예를 들어 다음 JSON은 스레드의 기본 수를 두 배로 늘려 데이터 검증을 활성화합니다. 또한 PostgreSQL 엔드포인트에서 열 기준 데이터 정렬 차이로 야기된 레코드 순서의 차이도 고려합니다. 또한 검증 실패를 처리할 추가 시간을 고려하도록 검증 보고 지연을 제공합니다.

```
"ValidationSettings": {
  "EnableValidation": true,
  "ThreadCount": 10,
  "HandleCollationDiff": true,
  "RecordFailureDelayLimitInMinutes": 30
}
```


Note

Oracle 엔드포인트의 경우 DBMS_CRYPTO를 AWS DMS 사용하여 BLOBs. Oracle 엔드포인트에서 BLOB이 사용되는 경우, Oracle 엔드포인트에 액세스하는 사용자 계정에 DBMS_CRYPTO에 대한 execute 권한을 부여해야 합니다. 이 작업을 수행하려면 다음 문을 실행하세요.

```
grant execute on sys.dbms_crypto to dms_endpoint_user;
```

데이터 재동기 설정

데이터 재동기화 기능을 사용하면 데이터 검증 보고서를 기반으로 대상 데이터베이스를 소스 데이터베이스와 재동기화할 수 있습니다. 자세한 내용은 [AWS DMS 데이터 검증](#) 섹션을 참조하세요.

재동기화 프로세스를 구성하는 ReplicationTaskSettings 엔드포인트의 ResyncSettings에 대한 추가 파라미터를 추가할 수 있습니다. 자세한 내용은 [AWS Database Migration Service 작업에 대한 작업 설정 지정](#)의 [작업 설정 예제](#)를 참조하세요.

Note

재동기화 프로세스가 활성화되어 있고 작업에 CDC 구성 요소가 있는 경우 ResyncSchedule 및 MaxResyncTime 파라미터가 필요합니다. 전체 로드 전용 작업에는 유효하지 않습니다.

데이터 재동기화 파라미터 설정 및 값은 다음과 같습니다.

EnableResync

true로 설정하면 데이터 재동기화 기능을 활성화합니다. 기본적으로 데이터 재동기화는 비활성화되어 있습니다.

데이터 형식: 부울

필수 항목 여부: 아니요

기본값: false

검증: TaskSettings에 ResyncSettings 파라미터가 있는 경우 null이 아니어야 합니다.

ResyncSchedule

데이터 재동기화 기능이 적용되는 기간입니다. Cron 형식으로 표시되어야 합니다. 자세한 내용은 [the section called “Cron 표현식 규칙”](#) 단원을 참조하십시오.

데이터 형식: 문자열

필수 항목 여부: 아니요

검증:

- Cron 표현식 형식으로 표시되어야 합니다.
- EnableResync가 true로 설정된 CDC 작업의 경우 null이 아니어야 합니다.
- CDC 구성 요소가 없는 작업에는 설정할 수 없습니다.

MaxResyncTime

데이터 재동기화 기능이 적용되는 최대 시간 제한은 분 단위입니다.

데이터 유형: 정수

필수 항목 여부: 아니요

검증:

- CDC 작업의 경우 null이어서는 안 됩니다.
- CDC가 없는 작업에는 필요하지 않습니다.
- 최솟값: 5 minutes, 최댓값: 14400 minutes(10일).

Validation onlyTaskID

검증 작업의 고유 ID입니다. 검증 전용 작업 ID는 ARN 끝에 추가됩니다. 예제:

- 검증 전용 작업 ARN: arn:aws:dms:us-west-2:123456789012:task:6DG4CLGJ5JSJR67CFD7UDXFY7KV6CYGRICL6KWI
- 검증 전용 작업 ID: 6DG4CLGJ5JSJR67CFD7UDXFY7KV6CYGRICL6KWI

데이터 형식: 문자열

필수 항목 여부: 아니요

검증: 데이터 재동기화 기능이 활성화되고 검증이 비활성화된 작업의 경우 null이 아니어야 합니다.

예시:

```
"ResyncSettings": {
  "EnableResync": true,
  "ResyncSchedule": "30 9 ? * MON-FRI",
  "MaxResyncTime": 400,
  "ValidationTaskId": "JXPP94804DJ0EWIJD9348R3049"
},
```

변경 처리 DDL을 다루기 위한 작업 설정

다음 설정은 변경 데이터 캡처(CDC) 중에 대상 테이블에 대한 데이터 정의 언어(DDL) 변경을 AWS DMS 처리하는 방법을 결정합니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

변경 처리 DDL을 처리하는 작업 설정에는 다음이 포함됩니다.

- `HandleSourceTableDropped` - 이 옵션을 `true`로 설정하면 원본 테이블을 삭제할 때 대상 테이블을 삭제합니다.
- `HandleSourceTableTruncated` - 이 옵션을 `true`로 설정하면 원본 테이블을 자를 때 대상 테이블을 자릅니다.
- `HandleSourceTableAltered` - 원본 테이블을 변경할 때 대상 테이블을 변경하려면 이 옵션을 `true`로 설정합니다.

다음은 변경 처리 DDL을 처리하는 작업 설정이 작업 설정 JSON 파일에 표시되는 방식의 예입니다.

```
"ChangeProcessingDdlHandlingPolicy": {
  "HandleSourceTableDropped": true,
  "HandleSourceTableTruncated": true,
  "HandleSourceTableAltered": true
},
```

Note

특정 엔드포인트에서 지원되는 DDL 문에 관한 자세한 내용은 이 엔드포인트에 대한 설명이 나오는 주제를 참조하세요.

문자 대체 작업 설정

복제 작업이 STRING 또는 WSTRING 데이터 형식의 모든 소스 데이터베이스 열에 대해 대상 데이터베이스에서 문자 대체를 AWS DMS 수행하도록 지정할 수 있습니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

다음 원본 및 대상 데이터베이스에서 엔드포인트가 있는 모든 작업에 대한 문자 대체를 구성할 수 있습니다.

- 소스 데이터베이스:
 - Oracle
 - Microsoft SQL Server
 - MySQL
 - MariaDB
 - PostgreSQL
 - SAP Adaptive Server Enterprise(ASE)
 - IBM DB2 LUW
- 대상 데이터베이스:
 - Oracle
 - Microsoft SQL Server
 - MySQL
 - MariaDB
 - PostgreSQL
 - SAP Adaptive Server Enterprise(ASE)
 - Amazon Redshift

작업 설정에서 CharacterSetSettings 파라미터를 사용하여 문자 대체를 지정할 수 있습니다. 이 문자 대체는 16진수 표기법으로 유니코드 코드 포인트 값을 사용하여 지정된 문자에 대해 발생합니다. 두 단계 모두 지정된 경우 두 단계를 다음 순서로 사용하여 문자 대체를 구현할 수 있습니다.

1. 개별 문자 대체 - 소스에서 선택한 문자의 값을 대상에서 해당 문자의 지정된 대체 값으로 AWS DMS 바꿀 수 있습니다. CharacterSetSettings에서 CharacterReplacements 배열을 사용하여 지정한 유니코드 코드 포인트가 있는 모든 소스 문자를 선택합니다. 또한 이 배열을 사용하여 대상의 해당 문자에 대한 대체 코드 포인트를 지정합니다.

지정된 코드 포인트가 있는 소스의 모든 문자를 선택하려면 CharacterReplacements 배열에서 SourceCharacterCodePoint의 인스턴스를 해당 코드 포인트로 설정합니다. 그런 다음 이 배열에서 TargetCharacterCodePoint의 해당 인스턴스를 설정하여 동등한 모든 대상 문자에 대한 대체 코드 포인트를 지정합니다. 대상 문자를 바꾸지 않고 삭제하려면 TargetCharacterCodePoint의 해당 인스턴스를 0으로 설정합니다. CharacterReplacements 배열에서 SourceCharacterCodePoint 및 TargetCharacterCodePoint 설정의 추가 쌍을 지정하여 원하는 만큼 서로 다른 값의 대상 문자를 바꾸거나 삭제할 수 있습니다. SourceCharacterCodePoint의 여러 인스턴스에 동일한 값을 지정하는 경우, TargetCharacterCodePoint의 마지막 해당 설정 값이 대상에 적용됩니다.

예를 들어 CharacterReplacements에 다음과 같은 값을 지정한다고 가정하겠습니다.

```
"CharacterSetSettings": {
  "CharacterReplacements": [ {
    "SourceCharacterCodePoint": 62,
    "TargetCharacterCodePoint": 61
  }, {
    "SourceCharacterCodePoint": 42,
    "TargetCharacterCodePoint": 41
  }
]
}
```

이 예제에서는 대상의 소스 코드 포인트 16진수 값이 62인 모든 문자를 코드 포인트 값이 61인 문자로 바 AWS DMS 끝니다. 또한 대상의 소스 코드 포인트 42가 있는 모든 문자를 코드 포인트 값 41이 있는 문자로 바 AWS DMS 끝니다. 다시 말해, AWS DMS 는 대상에 있는 문자 'b'의 모든 인스턴스를 문자 'a'로 바꿉니다. 마찬가지로는 대상 'B'에 있는 문자의 모든 인스턴스를 문자로 바 AWS DMS 끝니다 'A'.

- 문자 세트 검증 및 교체 - 개별 문자 교체가 완료되면 모든 대상 문자에 지정한 단일 문자 세트에 유효한 유니코드 코드 포인트가 있는지 AWS DMS 확인할 수 있습니다. CharacterSetSettings에서 CharacterSetSupport를 사용하여 이 대상 문자 검증 및 수정을 구성할 수 있습니다. 검증 문자 집합을 지정하려면 CharacterSetSupport의 CharacterSet를 문자 집합의 문자열 값으로 설정합니다. (CharacterSet에 대해 가능한 값은 다음과 같습니다.) 다음 방법 중 하나로 잘못된 대상 문자를 AWS DMS 수정할 수 있습니다.

- 현재 코드 포인트에 관계없이 유효하지 않은 모든 대상 문자에 대해 단일 대체 유니코드 코드 포인트를 지정합니다. 이 대체 코드 포인트를 구성하려면 CharacterSetSupport의 ReplaceWithCharacterCodePoint를 지정된 값으로 설정합니다.

- ReplaceWithCharacterCodePoint를 0으로 설정하여 유효하지 않은 모든 대상 문자의 삭제를 구성합니다.

예를 들어 CharacterSetSupport에 다음과 같은 값을 지정한다고 가정하겠습니다.

```
"CharacterSetSettings": {
  "CharacterSetSupport": {
    "CharacterSet": "UTF16_PlatformEndian",
    "ReplaceWithCharacterCodePoint": 0
  }
}
```

이 예제에서는 대상에서 발견된 문자 중 문자 집합에서 유효하지 않은 "UTF16_PlatformEndian" 문자를 AWS DMS 삭제합니다. 따라서 16진수 값 2FB6으로 지정된 모든 문자가 삭제됩니다. 이 값은 4바이트 유니코드 코드 포인트이고 UTF16 문자 집합은 2바이트 코드 포인트 문자만 수락하므로 유효하지 않습니다.

Note

복제 작업은 테이블 매핑을 통해 지정한 전역 또는 테이블 수준 변환을 시작하기 전에 지정된 문자 대체를 모두 완료합니다. 테이블 매핑에 관한 자세한 내용은 [작업 설정을 지정하기 위한 테이블 매핑 사용](#) 섹션을 참조하세요.

문자 대체는 LOB 데이터 유형을 지원하지 않습니다. 여기에는 DMS가 LOB 데이터 유형으로 간주하는 모든 데이터 유형이 포함됩니다. 예를 들어, Oracle의 Extended 데이터 유형은 LOB로 간주됩니다. 소스 데이터 유형에 대한 자세한 내용은 [Oracle용 소스 데이터 형식](#) 섹션을 참조하세요.

에서에 대해 AWS DMS 지원하는 값은 다음 표에 CharacterSet 나와 있습니다.

UTF-8	ibm-860_P100-1995	ibm-280_P100-1995
UTF-16	ibm-861_P100-1995	ibm-284_P100-1995
UTF-16BE	ibm-862_P100-1995	ibm-285_P100-1995
UTF-16LE	ibm-863_P100-1995	ibm-290_P100-1995

UTF-32	ibm-864_X110-1999	ibm-297_P100-1995
UTF-32BE	ibm-865_P100-1995	ibm-420_X120-1999
UTF-32LE	ibm-866_P100-1995	ibm-424_P100-1995
UTF16_PlatformEndian	ibm-867_P100-1998	ibm-500_P100-1995
UTF16_OppositeEndian	ibm-868_P100-1995	ibm-803_P100-1999
UTF32_PlatformEndian	ibm-869_P100-1995	ibm-838_P100-1995
UTF32_OppositeEndian	ibm-878_P100-1996	ibm-870_P100-1995
UTF-16BE,version=1	ibm-901_P100-1999	ibm-871_P100-1995
UTF-16LE,version=1	ibm-902_P100-1999	ibm-875_P100-1995
UTF-16,version=1	ibm-922_P100-1999	ibm-918_P100-1995
UTF-16,version=2	ibm-1168_P100-2002	ibm-930_P120-1999
UTF-7	ibm-4909_P100-1999	ibm-933_P110-1995
IMAP-mailbox-name	ibm-5346_P100-1998	ibm-935_P110-1999
SCSU	ibm-5347_P100-1998	ibm-937_P110-1999
BOCU-1	ibm-5348_P100-1997	ibm-939_P120-1999
CESU-8	ibm-5349_P100-1998	ibm-1025_P100-1995
ISO-8859-1	ibm-5350_P100-1998	ibm-1026_P100-1995
US-ASCII	ibm-9447_P100-2002	ibm-1047_P100-1995
gb18030	ibm-9448_X100-2005	ibm-1097_P100-1995
ibm-912_P100-1995	ibm-9449_P100-2002	ibm-1112_P100-1995
ibm-913_P100-2000	ibm-5354_P100-1998	ibm-1122_P100-1999

ibm-914_P100-1995	ibm-1250_P100-1995	ibm-1123_P100-1995
ibm-915_P100-1995	ibm-1251_P100-1995	ibm-1130_P100-1997
ibm-1089_P100-1995	ibm-1252_P100-2000	ibm-1132_P100-1998
ibm-9005_X110-2007	ibm-1253_P100-1995	ibm-1137_P100-1999
ibm-813_P100-1995	ibm-1254_P100-1995	ibm-4517_P100-2005
ibm-5012_P100-1999	ibm-1255_P100-1995	ibm-1140_P100-1997
ibm-916_P100-1995	ibm-5351_P100-1998	ibm-1141_P100-1997
ibm-920_P100-1995	ibm-1256_P110-1997	ibm-1142_P100-1997
iso-8859_10-1998	ibm-5352_P100-1998	ibm-1143_P100-1997
iso-8859_11-2001	ibm-1257_P100-1995	ibm-1144_P100-1997
ibm-921_P100-1995	ibm-5353_P100-1998	ibm-1145_P100-1997
iso-8859_14-1998	ibm-1258_P100-1997	ibm-1146_P100-1997
ibm-923_P100-1998	macos-0_2-10.2	ibm-1147_P100-1997
ibm-942_P12A-1999	macos-6_2-10.4	ibm-1148_P100-1997
ibm-943_P15A-2003	macos-7_3-10.2	ibm-1149_P100-1997
ibm-943_P130-1999	macos-29-10.2	ibm-1153_P100-1999
ibm-33722_P12A_P12 A-2009_U2	macos-35-10.2	ibm-1154_P100-1999
ibm-33722_P120-1999	ibm-1051_P100-1995	ibm-1155_P100-1999
ibm-954_P101-2007	ibm-1276_P100-1995	ibm-1156_P100-1999
euc-jp-2007	ibm-1006_P100-1995	ibm-1157_P100-1999
ibm-1373_P100-2002	ibm-1098_P100-1995	ibm-1158_P100-1999

windows-950-2000	ibm-1124_P100-1996	ibm-1160_P100-1999
ibm-950_P110-1999	ibm-1125_P100-1997	ibm-1164_P100-1999
ibm-1375_P100-2008	ibm-1129_P100-1997	ibm-1364_P110-2007
ibm-5471_P100-2006	ibm-1131_P100-1997	ibm-1371_P100-1999
ibm-1386_P100-2001	ibm-1133_P100-1997	ibm-1388_P103-2001
windows-936-2000	ISO_2022,locale=ja ,version=0	ibm-1390_P110-2003
ibm-1383_P110-1999	ISO_2022,locale=ja ,version=1	ibm-1399_P110-2003
ibm-5478_P100-1995	ISO_2022,locale=ja ,version=2	ibm-5123_P100-1999
euc-tw-2014	ISO_2022,locale=ja ,version=3	ibm-8482_P100-1999
ibm-964_P110-1999	ISO_2022,locale=ja ,version=4	ibm-16684_P110-2003
ibm-949_P110-1999	ISO_2022,locale=ko ,version=0	ibm-4899_P100-1998
ibm-949_P11A-1999	ISO_2022,locale=ko ,version=1	ibm-4971_P100-1999
ibm-970_P110_P110- 2006_U2	ISO_2022,locale=zh ,version=0	ibm-9067_X100-2005
ibm-971_P100-1995	ISO_2022,locale=zh ,version=1	ibm-12712_P100-1998
ibm-1363_P11B-1998	ISO_2022,locale=zh ,version=2	ibm-16804_X110-1999

ibm-1363_P110-1997	HZ	ibm-37_P100-1995,s waplfnl
windows-949-2000	x11-compound-text	ibm-1047_P100-1995 ,swaplfnl
windows-874-2000	ISCII,version=0	ibm-1140_P100-1997 ,swaplfnl
ibm-874_P100-1995	ISCII,version=1	ibm-1141_P100-1997 ,swaplfnl
ibm-1162_P100-1999	ISCII,version=2	ibm-1142_P100-1997 ,swaplfnl
ibm-437_P100-1995	ISCII,version=3	ibm-1143_P100-1997 ,swaplfnl
ibm-720_P100-1997	ISCII,version=4	ibm-1144_P100-1997 ,swaplfnl
ibm-737_P100-1997	ISCII,version=5	ibm-1145_P100-1997 ,swaplfnl
ibm-775_P100-1996	ISCII,version=6	ibm-1146_P100-1997 ,swaplfnl
ibm-850_P100-1995	ISCII,version=7	ibm-1147_P100-1997 ,swaplfnl
ibm-851_P100-1995	ISCII,version=8	ibm-1148_P100-1997 ,swaplfnl
ibm-852_P100-1995	LMBCS-1	ibm-1149_P100-1997 ,swaplfnl
ibm-855_P100-1995	ibm-37_P100-1995	ibm-1153_P100-1999 ,swaplfnl

ibm-856_P100-1995	ibm-273_P100-1995	ibm-12712_P100-1998,swaplfn1
ibm-857_P100-1995	ibm-277_P100-1995	ibm-16804_X110-1999,swaplfn1
ibm-858_P100-1997	ibm-278_P100-1995	ebcdic-xml-us

이전 이미지 작업 설정

Kinesis 또는 Kafka 같은 데이터 스트리밍 대상에 CDC 업데이트를 작성하는 경우 업데이트를 통한 변경 전에 소스 데이터베이스 행의 원래 값을 볼 수 있습니다. 이를 가능하게 하기 위해서는 소스 데이터베이스 엔진에서 제공하는 데이터를 기반으로 업데이트 이벤트의 이전 이미지를 AWS DMS 채웁니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

이를 위해 BeforeImageSettings 파라미터를 사용하면 소스 데이터베이스 시스템에서 수집된 값을 사용하여 모든 업데이트 작업에 새 JSON 속성이 추가됩니다.

BeforeImageSettings는 전체 로드와 CDC 작업 또는 CDC 전용 작업에만 적용해야 합니다. 전체 로드와 CDC 작업은 기존 데이터를 마이그레이션하고 지속적인 변경 사항을 복제합니다. CDC 전용 작업은 데이터 변경 사항만 복제합니다.

전체 로드 전용 작업에는 BeforeImageSettings를 적용하지 마세요.

BeforeImageSettings의 옵션은 다음과 같습니다.

- EnableBeforeImage – true로 설정하면 이전 이미징이 활성화됩니다. 기본값은 false입니다.
- FieldName – 새 JSON 속성에 이름을 지정합니다. EnableBeforeImage가 true인 경우 FieldName은 필수이며 비워 둘 수 없습니다.
- ColumnFilter – 이전 이미징을 사용하여 추가할 열을 지정합니다. 테이블 프라이머리 키의 일부에 속하는 열만 추가하려면 기본값 pk-only를 사용합니다. 이전 이미지 값이 있는 모든 열을 추가하려면 all을 사용합니다. 참고로 이전 이미지는 CLOB 및 BLOB와 같은 대용량 이진 객체(LOB) 데이터 유형을 지원하지 않습니다.

다음은 BeforeImageSettings의 용례를 보여줍니다.

```
"BeforeImageSettings": {
```

```

    "EnableBeforeImage": true,
    "FieldName": "before-image",
    "ColumnFilter": "pk-only"
  }

```

추가 테이블 매핑 설정을 비롯해 Kinesis 사용 시 이전 이미지 설정에 관한 자세한 내용은 [Kinesis 데이터 스트림 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기](#) 섹션을 참조하세요.

추가 테이블 매핑 설정을 비롯해 Kafka 사용 시 이전 이미지 설정에 관한 자세한 내용은 [Apache Kafka 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기](#) 섹션을 참조하세요.

오류 처리 작업 설정

다음 설정을 사용하여 복제 작업의 오류 처리 동작을 설정할 수 있습니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 [작업 설정 예제](#)를 참조하세요.

- **DataErrorPolicy** - 레코드 수준에서 데이터 처리와 관련된 오류가 있을 때 AWS DMS가 수행하는 작업을 결정합니다. 일부 데이터 처리 오류 예제에는 전환 오류, 변환 오류, 잘못된 데이터가 포함됩니다. 기본값은 LOG_ERROR입니다.
 - **IGNORE_RECORD** - 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. `DataErrorEscalationCount` 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
 - **LOG_ERROR** - 작업은 계속되고 오류가 작업 로그에 기록됩니다.
 - **SUSPEND_TABLE** - 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
 - **STOP_TASK** - 작업은 중지되고 수동 개입이 필요합니다.
- **DataTruncationErrorPolicy** - 데이터를 자를 때 AWS에서 수행하는 작업을 결정합니다. 기본값은 LOG_ERROR입니다.
 - **IGNORE_RECORD** - 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. `DataErrorEscalationCount` 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
 - **LOG_ERROR** - 작업은 계속되고 오류가 작업 로그에 기록됩니다.
 - **SUSPEND_TABLE** - 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
 - **STOP_TASK** - 작업은 중지되고 수동 개입이 필요합니다.
- **DataErrorEscalationPolicy** - 최대 오류 수(`DataErrorEscalationCount` 파라미터에서 설정됨)에 도달할 때 AWS DMS에서 수행하는 작업을 결정합니다. 기본값은 SUSPEND_TABLE입니다.

- **SUSPEND_TABLE** – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
- **STOP_TASK** – 작업은 중지되고 수동 개입이 필요합니다.
- **DataErrorEscalationCount** – 특정 레코드의 데이터에 발생할 수 있는 최대 오류 수를 설정합니다. 이 수치에 도달하면 오류 레코드를 포함하는 테이블의 데이터는 **DataErrorEscalationPolicy**에 설정된 정책에 따라 처리됩니다. 기본값은 0입니다.
- **EventErrorPolicy** - 작업 관련 이벤트를 보내는 동안 오류가 발생할 때 AWS DMS가 수행하는 작업을 결정합니다. 가능한 값은 다음과 같습니다.
 - **IGNORE** – 작업은 계속되며 해당 이벤트와 관련된 모든 데이터는 무시됩니다.
 - **STOP_TASK** – 작업은 중지되고 수동 개입이 필요합니다.
- **TableErrorPolicy** – 특정 테이블의 데이터 또는 메타데이터를 처리할 때 오류가 발생하는 경우 AWS DMS에서 수행할 작업을 결정합니다. 이 오류는 일반 테이블 데이터에만 적용되고 특정 레코드와 관련된 오류가 아닙니다. 기본값은 **SUSPEND_TABLE**입니다.
 - **SUSPEND_TABLE** – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
 - **STOP_TASK** – 작업은 중지되고 수동 개입이 필요합니다.
- **TableErrorEscalationPolicy** – 최대 오류 수(**TableErrorEscalationCount** 파라미터를 사용하여 설정됨)에 도달할 때 AWS에서 수행하는 작업을 결정합니다. 기본값과 유일한 사용자 설정은 **STOP_TASK**이며 여기에서 작업은 중지되고 수동 개입이 필요합니다.
- **TableErrorEscalationCount** – 특정 테이블의 일반 데이터 또는 메타데이터에 발생할 수 있는 오류의 최대 수입니다. 이 수치에 도달하면 테이블의 데이터는 **TableErrorEscalationPolicy**에 설정된 정책에 따라 처리됩니다. 기본값은 0입니다.
- **RecoverableErrorCount** – 환경 오류가 발생할 때 작업을 다시 시작하기 위한 시도의 최대 횟수입니다. 시스템이 작업을 지정된 횟수만큼 다시 시작하려고 시도하고 나면, 이 작업은 중지되고 수동 개입이 필요합니다. 기본값은 -1입니다.

이 값을 -1로 설정하면 DMS가 시도하는 재시도 횟수는 다음과 같이 반환된 오류 유형에 따라 달라집니다.

- 실행 상태, 복구 가능 오류: 연결 끊김 또는 대상 적용 실패와 같은 복구 가능 오류가 발생하면 DMS는 태스크를 9회 재시도합니다.
- 시작 상태, 복구 가능 오류: DMS가 태스크를 6회 재시도합니다.
- 실행 상태, DMS에서 처리하는 치명적인 오류: DMS는 태스크를 6회 재시도합니다.
- 실행 중 상태, DMS에서 처리하지 않은 치명적인 오류: DMS는 태스크를 다시 시도하지 않습니다.

- 위 이외의 경우:는 작업을 무기한 AWS DMS 재시도합니다.

이 값을 0으로 설정하면 작업을 다시 시작하기 위한 시도를 하지 않습니다.

DMS 태스크가 제대로 복구될 수 있도록 RecoverableErrorCount 및 RecoverableErrorInterval을 충분한 간격으로 재시도할 수 있는 값으로 설정하는 것이 좋습니다. 치명적인 오류가 발생하면 DMS는 대부분의 시나리오에서 재시작 시도를 중지합니다.

- RecoverableErrorInterval - AWS DMS가 작업을 다시 시작하려는 시도 사이에 대기하는 초 수입니다. 기본값은 5입니다.
- RecoverableErrorThrottling - 활성화되면 작업 재시작 시도 간격이 RecoverableErrorInterval 값을 기준으로 연속적으로 늘어납니다. 예를 들어 RecoverableErrorInterval을 5초로 설정하면 다음 재시도는 10초 후, 20초 후, 40초 후 등등의 시간 경과 후에 다시 시도됩니다. 기본값은 true입니다.
- RecoverableErrorThrottlingMax -가 활성화된 경우 AWS DMS가 작업을 다시 시작하려는 시도 사이에 대기하는 최대 초 RecoverableErrorThrottling 수입니다. 기본값은 1800입니다.
- RecoverableErrorStopRetryAfterThrottlingMax- 기본값은 로 설정되며 trueDMS 는에 따라 복구 시도 사이에 AWS DMS 대기하는 최대 초 수에 도달한 후 작업 재개를 중지합니다RecoverableErrorStopRetryAfterThrottlingMax. 로 설정하면 falseDMS는 복구 시도 사이에 AWS DMS 대기하는 최대 초 수에 도달한 후에 도달할 RecoverableErrorStopRetryAfterThrottlingMax 때까지에 따라 작업을 계속 재 개RecoverableErrorCount합니다.
- ApplyErrorDeletePolicy - DELETE 작업과 충돌이 있을 때 AWS 에서 수행하는 작업을 결정합니다. 기본값은 IGNORE_RECORD입니다. 가능한 값은 다음과 같습니다.
 - IGNORE_RECORD - 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. ApplyErrorEscalationCount 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
 - LOG_ERROR - 작업은 계속되고 오류가 작업 로그에 기록됩니다.
 - SUSPEND_TABLE - 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동 하고 데이터가 복제되지 않습니다.
 - STOP_TASK - 작업은 중지되고 수동 개입이 필요합니다.
- ApplyErrorInsertPolicy - INSERT 작업과 충돌이 있을 때 AWS DMS에서 수행하는 작업을 결정합니다. 기본값은 LOG_ERROR입니다. 가능한 값은 다음과 같습니다.
 - IGNORE_RECORD - 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. ApplyErrorEscalationCount 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.

- LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다.
- SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
- STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- INSERT_RECORD – 삽입한 원본 레코드와 동일한 프라이머리 키를 사용하는 기존 대상 레코드가 있으면 대상 레코드가 업데이트됩니다.

Note

- 트랜잭션 적용 모드: 이 프로세스에서 시스템은 먼저 레코드 삽입을 시도합니다. 프라이머리 키 충돌로 인해 삽입이 실패하면 기존 레코드가 삭제된 다음 새 레코드를 삽입합니다.
- 배치 적용 모드: 프로세스가 전체 새 레코드 세트를 삽입하기 전에 대상 배치의 기존 레코드를 모두 제거하여 데이터를 완전히 대체합니다.
- 이 프로세스는 데이터 중복을 방지하지만 기본 정책에 비해 약간의 성능 비용이 발생합니다. 정확한 성능 영향은 특정 워크로드 특성에 따라 달라집니다.

- ApplyErrorUpdatePolicy – UPDATE 작업과 누락 데이터 충돌이 있을 때 AWS DMS에서 수행하는 작업을 결정합니다. 기본값은 LOG_ERROR입니다. 가능한 값은 다음과 같습니다.
- IGNORE_RECORD – 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. ApplyErrorEscalationCount 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
- LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다.
- SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
- STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- UPDATE_RECORD - 대상 레코드가 누락된 경우 누락된 대상 레코드가 대상 테이블에 삽입됩니다. 작업에 대한 LOB 열 지원을 AWS DMS 완전히 비활성화합니다. 이 옵션을 선택하는 경우, Oracle이 소스 데이터베이스이면 모든 원본 테이블 열에서 전체 보충 로깅을 활성화해야 합니다.

Note

- 트랜잭션 적용 모드: 이 프로세스에서 시스템은 먼저 레코드 업데이트를 시도합니다. 대상에 누락된 레코드로 인해 업데이트가 실패하면 실패한 레코드에 대해 삭제를 실행한

다음 새 레코드를 삽입합니다. 이 프로세스에는 Oracle 소스 데이터베이스에 대한 전체 보충 로깅이 필요하며 DMS는 이 작업에 대한 LOB 열 지원을 비활성화합니다.

- 배치 적용 모드: 프로세스가 전체 새 레코드 세트를 삽입하기 전에 대상 배치의 기존 레코드를 모두 제거하여 데이터를 완전히 대체합니다.

- `ApplyErrorEscalationPolicy` - 최대 오류 수(`ApplyErrorEscalationCount` 파라미터를 사용하여 설정)에 도달할 때 AWS DMS가 수행하는 작업을 결정합니다. 기본값은 `LOG_ERROR`입니다.
- `LOG_ERROR` - 작업은 계속되고 오류가 작업 로그에 기록됩니다.
- `SUSPEND_TABLE` - 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
- `STOP_TASK` - 작업은 중지되고 수동 개입이 필요합니다.
- `ApplyErrorEscalationCount` - 이 옵션은 변경 프로세스 작업 중에 특정 테이블에서 발생할 수 있는 최대 `APPLY` 충돌 개수를 설정합니다. 이 개수에 도달하면 테이블 데이터는 `ApplyErrorEscalationPolicy` 파라미터에 설정된 정책에 따라 처리됩니다. 기본값은 0입니다.
- `ApplyErrorFailOnTruncationDdl` - CDC 중에 추적되는 테이블에 대해 잘림이 수행될 때 작업이 실패하도록 하려면 이 옵션을 `true`로 설정합니다. 기본값은 `false`입니다.

이 방법은 DDL 테이블 잘림을 복제하지 않는 PostgreSQL 버전 11.x나 이전 버전 또는 다른 소스 엔드포인트에서는 사용할 수 없습니다.

- `FailOnNoTablesCaptured` - 작업이 시작될 때 작업에 대해 정의된 테이블 매핑이 테이블을 찾지 못하는 경우 작업이 실패하도록 하려면 이 옵션을 `true`로 설정합니다. 기본값은 `true`입니다.
- `FailOnTransactionConsistencyBreached` - 이 옵션은 CDC와 함께 사용하기 위한 원본으로 Oracle을 사용하는 작업에 적용됩니다. 기본값은 `false`입니다. 트랜잭션이 지정된 제한 시간보다 더 오랫동안 열려 있고 삭제될 수 있는 경우 작업이 실패하도록 하려면 `true`로 설정합니다.

CDC 작업이 Oracle로 시작되면 CDC를 시작하기 전에 가장 오래된 열린 트랜잭션이 종료될 때까지 제한된 시간 동안 AWS DMS 기다립니다. 가장 오래된 열린 트랜잭션이 제한 시간에 도달할 때까지 달하지 않으면 대부분의 경우는 해당 트랜잭션을 무시하고 CDC를 AWS DMS 시작합니다. 이 옵션을 `true`로 설정하면 작업이 실패합니다.

- `FullLoadIgnoreConflicts` - 캐시된 이벤트를 적용할 때 "영향을 받는 행 0" 및 "중복" 오류를 AWS DMS 무시하도록 하려면 이 옵션을 `true`로 설정합니다. `true`로 설정하면 모든 오류를 무시하는 대신 `false` AWS DMS 보고합니다. 기본값은 `true`입니다.
- `DataMaskingErrorPolicy` - 호환되지 않는 데이터 유형 또는 기타 이유로 인해 데이터 마스킹이 실패할 때 AWS DMS 취하는 작업을 결정합니다. 다음과 같은 옵션을 사용할 수 있습니다.

- STOP_TASK(기본값) – 작업은 중지되고 수동 개입이 필요합니다.
- IGNORE_RECORD – 작업은 계속되고 해당 레코드의 데이터는 무시됩니다.
- LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다. 마스킹되지 않은 데이터는 대상 테이블에 로드됩니다.
- SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.

Note

대상으로서 Redshift의 테이블 로드 오류는에 보고됩니다STL_LOAD_ERRORS. 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서에서 [STL_LOAD_ERRORS](#)를 참조하세요.

Note

복구 가능한 오류와 관련된 파라미터 변경은 DMS 작업을 중지하고 재개한 후에만 적용됩니다. 현재 실행에는 변경 사항이 적용되지 않습니다.

작업 설정 저장

다른 작업의 설정을 다시 사용해야 하는 경우에 작업 설정을 JSON 파일로 저장할 수 있습니다. 작업의 개요 세부 정보 섹션에서 JSON 파일에 복사할 작업 설정을 찾을 수 있습니다.

Note

작업 설정을 다른 작업에 재사용하는 동안에는 모든 CloudWatchLogGroup 및 CloudWatchLogStream 속성을 제거하세요. 그렇지 않으면 다음 오류가 발생합니다. 시스템 오류 메시지: 작업 설정 생성 시 CloudWatchLogGroup 또는 CloudWatchLogStream을 설정할 수 없습니다.

예를 들어, 다음 JSON 파일에는 작업에 대해 저장된 설정이 포함됩니다.

```
{
  "TargetMetadata": {
```

```

    "TargetSchema": "",
    "SupportLobs": true,
    "FullLobMode": false,
    "LobChunkSize": 0,
    "LimitedSizeLobMode": true,
    "LobMaxSize": 32,
    "InlineLobMaxSize": 0,
    "LoadMaxFileSize": 0,
    "ParallelLoadThreads": 0,
    "ParallelLoadBufferSize": 0,
    "BatchApplyEnabled": false,
    "TaskRecoveryTableEnabled": false,
    "ParallelLoadQueuesPerThread": 0,
    "ParallelApplyThreads": 0,
    "ParallelApplyBufferSize": 0,
    "ParallelApplyQueuesPerThread": 0
  },
  "FullLoadSettings": {
    "TargetTablePrepMode": "DO_NOTHING",
    "CreatePkAfterFullLoad": false,
    "StopTaskCachedChangesApplied": false,
    "StopTaskCachedChangesNotApplied": false,
    "MaxFullLoadSubTasks": 8,
    "TransactionConsistencyTimeout": 600,
    "CommitRate": 10000
  },
  "Logging": {
    "EnableLogging": true,
    "LogComponents": [
      {
        "Id": "TRANSFORMATION",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },
      {
        "Id": "SOURCE_UNLOAD",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },
      {
        "Id": "IO",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },
      {
        "Id": "TARGET_LOAD",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      }
    ]
  }
}

```

```
    },
    {
      "Id": "PERFORMANCE",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "SOURCE_CAPTURE",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "SORTER",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "REST_SERVER",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "VALIDATOR_EXT",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "TARGET_APPLY",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "TASK_MANAGER",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "TABLES_MANAGER",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "METADATA_MANAGER",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "FILE_FACTORY",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "COMMON",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    }
```

```

    },
    {
      "Id": "ADDONS",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "DATA_STRUCTURE",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "COMMUNICATION",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    },
    {
      "Id": "FILE_TRANSFER",
      "Severity": "LOGGER_SEVERITY_DEFAULT"
    }
  ]
},
"ControlTablesSettings": {
  "ControlSchema": "",
  "HistoryTimeslotInMinutes": 5,
  "HistoryTableEnabled": false,
  "SuspendedTablesTableEnabled": false,
  "StatusTableEnabled": false,
  "FullLoadExceptionTableEnabled": false
},
"StreamBufferSettings": {
  "StreamBufferCount": 3,
  "StreamBufferSizeInMB": 8,
  "CtrlStreamBufferSizeInMB": 5
},
"ChangeProcessingDdlHandlingPolicy": {
  "HandleSourceTableDropped": true,
  "HandleSourceTableTruncated": true,
  "HandleSourceTableAltered": true
},
"ErrorBehavior": {
  "DataErrorPolicy": "LOG_ERROR",
  "DataTruncationErrorPolicy": "LOG_ERROR",
  "DataErrorEscalationPolicy": "SUSPEND_TABLE",
  "DataErrorEscalationCount": 0,
  "TableErrorPolicy": "SUSPEND_TABLE",
  "TableErrorEscalationPolicy": "STOP_TASK",

```

```

    "TableErrorEscalationCount": 0,
    "RecoverableErrorCount": -1,
    "RecoverableErrorInterval": 5,
    "RecoverableErrorThrottling": true,
    "RecoverableErrorThrottlingMax": 1800,
    "RecoverableErrorStopRetryAfterThrottlingMax": true,
    "ApplyErrorDeletePolicy": "IGNORE_RECORD",
    "ApplyErrorInsertPolicy": "LOG_ERROR",
    "ApplyErrorUpdatePolicy": "LOG_ERROR",
    "ApplyErrorEscalationPolicy": "LOG_ERROR",
    "ApplyErrorEscalationCount": 0,
    "ApplyErrorFailOnTruncationDdl": false,
    "FullLoadIgnoreConflicts": true,
    "FailOnTransactionConsistencyBreached": false,
    "FailOnNoTablesCaptured": true
  },
  "ChangeProcessingTuning": {
    "BatchApplyPreserveTransaction": true,
    "BatchApplyTimeoutMin": 1,
    "BatchApplyTimeoutMax": 30,
    "BatchApplyMemoryLimit": 500,
    "BatchSplitSize": 0,
    "MinTransactionSize": 1000,
    "CommitTimeout": 1,
    "MemoryLimitTotal": 1024,
    "MemoryKeepTime": 60,
    "StatementCacheSize": 50
  },
  "PostProcessingRules": null,
  "CharacterSetSettings": null,
  "LoopbackPreventionSettings": null,
  "BeforeImageSettings": null,
  "FailTaskWhenCleanTaskResourceFailed": false
}

```

AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정

경우에 따라 대용량 이진 객체(LOB)는 시스템 간에 마이그레이션하기 어려울 수 있습니다. AWS DMS에는 LOB 열을 튜닝할 때 도움이 되는 다양한 옵션이 있습니다. 어떤 데이터 형식이 LOBs로 간주되는지 확인하려면 설명서를 AWS DMS참조 AWS DMS 하세요.

다른 데이터베이스로 데이터를 마이그레이션할 때에는 LOB 저장 방법을 재고해 볼 수 있으며, 특히 이중 마이그레이션에서 그러합니다. 이 작업을 수행할 경우, LOB 데이터를 마이그레이션할 필요가 없습니다.

LOB를 포함하기로 결정하는 경우 다른 LOB 설정을 결정할 수 있습니다.

- LOB 모드는 LOB가 처리되는 방식을 결정합니다.
 - 전체 LOB 모드 - 전체 LOB 모드에서는 크기에 관계없이 모든 LOBs 소스에서 대상으로 AWS DMS 마이그레이션합니다. 이 구성에서 AWS DMS 는 예상할 수 있는 최대 LOBs 크기에 대한 정보가 없습니다. 또한 LOB는 한 번에 하나씩, 조각별로 마이그레이션됩니다. 전체 LOB 모드는 매우 느릴 수 있습니다.
 - 제한적 LOB 모드 - 제한적 LOB 모드에서는 DMS가 수용해야 하는 최대 LOB 크기를 설정합니다. 이를 통해 DMS는 메모리를 사전 할당하고 LOB 데이터를 대량으로 로드할 수 있습니다. 최대 LOB 크기를 초과하는 LOB는 잘리고 경고가 로그 파일에 기록됩니다. 제한적 LOB 모드는 전체 LOB 모드에 비해 상당한 성능 이점이 있습니다. 가능하다면 제한적 LOB 모드 사용을 권장합니다. 이 파라미터의 최대값은 102,400KB(100MB)입니다.

Note

63KB보다 큰 값의 최대 LOB 크기(K) 옵션을 사용하면 제한된 LOB 모드에서 실행하도록 구성된 전체 로드의 성능에 영향을 줍니다. 전체 로드 시 DMS는 최대 LOB 크기(k) 값에 커밋 비율을 곱하여 메모리를 할당하며, 해당 제품에는 LOB 열 수를 곱합니다. DMS가 해당 메모리를 사전 할당할 수 없는 경우 전체 로드 작업의 성능에 부정적인 영향을 미치는 SWAP 메모리를 사용합니다. 제한된 LOB 모드를 사용할 때 성능 문제가 발생하는 경우, 적절한 수준의 성능에 도달할 때까지 커밋 비율을 낮추는 것이 좋습니다. CDC 모드에서 DMS는 LOB 열 수에 제한된 LOB 작업 설정에 지정된 최대 LOB 크기 파라미터를 곱한 다음 레코드 크기를 곱하여 메모리를 할당합니다. DMS CDC 프로세스는 DMS 작업당 단일 스레드입니다. 자세한 내용은 [처리 튜닝 설정 변경](#)을 참조하세요.

제한된 LOB 크기를 확인하려면 ValidationPartialLobSize를 LobMaxSize(K)와 같은 값으로 설정해야 합니다.

- 인라인 LOB 모드 - 인라인 LOB 모드에서는 DMS가 인라인으로 전송하는 최대 LOB 크기를 설정합니다. 지정된 크기보다 작은 LOB는 인라인으로 전송됩니다. 지정된 크기보다 큰 LOB는 전체 LOB 모드를 사용하여 복제됩니다. 대부분의 LOB가 작은 경우, 이 옵션을 선택하여 작은 LOB와 큰 LOB를 모두 복제할 수 있습니다. DMS는 S3 및 Redshift와 같이 전체 LOB 모드를 지원하지 않는 엔드포인트에 대해서는 인라인 LOB 모드를 지원하지 않습니다.

Note

Oracle을 통해 LOB는 가능할 때마다 VARCHAR 데이터 형식으로 처리됩니다. 이 접근 방식은가 데이터베이스에서 대량으로 AWS DMS 가져오므로 다른 방법보다 훨씬 빠릅니다. Oracle에서 VARCHAR의 최대 크기는 32K이므로 Oracle이 소스 데이터베이스일 때 32K 미만의 제한적인 LOB 크기가 최적입니다.

- 작업이 제한적 LOB 모드에서 실행되도록 구성된 경우 최대 LOB 크기(K) 옵션은 AWS DMS 가 허용하는 최대 크기의 LOB를 설정합니다. 이 값보다 큰 LOB는 이 값으로 잘립니다.
- 태스크가 전체 LOB 모드를 사용하도록 구성된 경우는 LOBs 여러 조각으로 AWS DMS 검색합니다. LOB 청크 크기(K) 옵션은 각 조각의 크기를 결정합니다. 이 옵션을 설정하면 네트워크 구성에서 허용된 최대 패킷 크기에 특히 유의해야 합니다. LOB 청크 크기가 최대 허용 패킷 크기를 초과하면 연결 해제 오류가 나타날 수 있습니다. LobChunkSize의 권장값은 64킬로바이트[KB]입니다. LobChunkSize의 값을 64KB 이상으로 늘리면 작업이 실패할 수 있습니다.
- 작업이 인라인 LOB 모드에서 실행되도록 구성된 경우, InlineLobMaxSize 설정에 따라 DMS가 인라인으로 전송할 LOB가 결정됩니다.

Note

프라이머리 키는 변경 데이터 캡처(CDC) 작업 중에 LOB 열이 포함된 테이블에 필수입니다. DMS는이 키를 사용하여 소스 테이블에서 LOB 값을 조회합니다. 이 요구 사항은 CDC 작업에만 적용됩니다. 전체 로드 작업은 제한 없이 전체 LOB 열을 소스에서 대상으로 직접 읽고 복사할 수 있습니다.

이 옵션을 지정하는 작업 설정에 관한 자세한 내용은 [대상 메타데이터 작업 설정](#) 섹션을 참조하세요

소스 테이블의 최대 LOB 열 길이를 확인하는 SQL 명령

다음 SQL 명령을 사용하여 최대 LOB 열 길이를 확인하고 마이그레이션에서 데이터가 잘리지 않도록 DMS 제한 LOB 설정을 구성합니다.

Oracle

```
SELECT dbms_lob.getlength(<COL_NAME>) as LOB_LENGTH
FROM <TABLE_NAME>
ORDER BY dbms_lob.getlength(<COL_NAME>) DESC
FETCH FIRST 10 ROWS ONLY;
```

```
Select ((max(length(<COL_NAME>)))/(1024)) from <TABLE_NAME>
```

SQL Server

```
Select top 10 datalength(<COL_NAME>) as fieldsize from <TABLE_NAME> order by  
datalength(<COL_NAME>) desc;
```

MySQL

```
Select (max(length(<COL_NAME>)))/(1024)) as "Size in KB" from <TABLE_NAME>;
```

PostgreSQL

```
Select max((octet_length(<COL_NAME>))/(1024.0)) as "Size in KB" from <TABLE_NAME>;
```

Db2 LUW

```
-- Method 1: Using SYSCAT.COLUMNS (converting to KB)
```

```
SELECT TABSCHEMA, TABNAME, COLNAME, LENGTH/1024 as LENGTH_KB, TYPENAME FROM  
SYSCAT.COLUMNS WHERE TYPENAME IN ('BLOB', 'CLOB', 'DBCLOB') ORDER BY LENGTH DESC;
```

```
-- Method 2: For specific table with KB conversion
```

```
SELECT COLNAME, LENGTH/1024 as LENGTH_KB, TYPENAME FROM SYSCAT.COLUMNS WHERE  
TABSCHEMA = 'YOUR_SCHEMA' AND TABNAME = 'YOUR_TABLE' AND TYPENAME IN ('BLOB', 'CLOB',  
'DBCLOB');
```

```
-- Method 3: Using SYSIBM.SYSCOLUMNS
```

```
SELECT TBCREATOR, TBNAME, NAME, LENGTH/1024 as LENGTH_KB, COLTYPE FROM  
SYSIBM.SYSCOLUMNS WHERE COLTYPE IN ('BLOB', 'CLOB', 'DBCLOB') ORDER BY LENGTH DESC;
```

```
SYBASE :
```

```
SELECT c.name as column_name, t.name as data_type, (c.length)/1024 as length_KB FROM  
syscolumns c JOIN systypes t ON c.usertype = t.usertype WHERE object_name(c.id) =  
'YOUR_TABLE_NAME' AND t.name IN ('text', 'image', 'unitext') ORDER BY c.length DESC;
```


여러 작업 생성

일부 마이그레이션 시나리오에서는 여러 마이그레이션 작업을 생성해야 할 수도 있습니다. 작업은 독립적으로 수행되며 동시에 실행할 수 있습니다. 각 작업에는 자체 초기 로드, CDC, 로그 읽기 프로세스가 있습니다. 데이터 조작 언어(DML)를 통해 관련된 테이블은 동일한 작업의 일부여야 합니다.

마이그레이션에서 여러 작업을 생성해야 하는 이유는 주로 다음과 같습니다.

- 작업용 대상 테이블은 시스템을 팬아웃하거나 여러 시스템으로 나눌 경우 등 각기 다른 데이터베이스에 상주합니다.
- 대규모 테이블의 마이그레이션을 여러 작업으로 나누려면 필터링을 사용해야 합니다.

Note

각 작업에는 자체 변경 캡처와 로그 읽기 프로세스가 있기 때문에 변경 사항은 여러 작업에서 조정되지 않습니다. 따라서 여러 작업을 사용하여 마이그레이션을 수행할 때에는 개별 원본 트랜잭션이 단일 작업 내에 완전히 포함되도록 해야 합니다. 개별 트랜잭션이 여러 작업으로 분할되지 않은 경우, 여러 작업을 사용하여 마이그레이션을 수행할 수 있습니다.

를 사용하여 지속적 복제를 위한 작업 생성 AWS DMS

소스 데이터 스토어에서 진행 중인 변경 사항을 캡처하는 AWS DMS 작업을 생성할 수 있습니다. 데이터를 마이그레이션하는 동안에도 이 변경 사항을 캡처할 수 있습니다. 작업을 생성하여 지원된 대상 데이터 저장소로 초기(전체 로드) 마이그레이션을 완료한 후 지속적 변경 사항을 캡처할 수도 있습니다. 이 프로세스를 지속적 복제 또는 변경 데이터 캡처(CDC)라고 합니다.는 소스 데이터 스토어에서 지속적 변경 사항을 복제할 때 이 프로세스를 AWS DMS 사용합니다. 이 프로세스는 데이터베이스 엔진의 기본 API를 사용하여 데이터베이스 로그에 대한 변경 사항을 수집합니다.

Note

뷰는 전체 로드 작업만 사용하여 마이그레이션할 수 있습니다. 작업이 CDC 전용 작업이거나 완료된 후 CDC를 시작하는 전체 로드 작업인 경우, 소스의 테이블만 마이그레이션에 포함됩니다. 전체 로드 전용 작업을 사용하면 뷰 또는 테이블 및 뷰 조합을 마이그레이션할 수 있습니다. 자세한 내용은 [JSON을 사용하여 테이블 선택 및 변환 지정](#) 단원을 참조하십시오.

각 소스 엔진에는 이 변경 스트림을 지정된 사용자 계정에 노출하기 위한 특정 구성 요구 사항이 있습니다. 대다수 엔진에서는 데이터 손실 없이 캡처 프로세스를 지원할 수 있도록 유의미한 방식으로 변경 데이터를 소모하도록 해주는 일부 추가 구성이 필요합니다. 예를 들어, Oracle은 보충 로깅 추가를 필요로 하고, MySQL은 행 수준의 2진법 로깅(빈 로깅)을 필요로 합니다.

소스 데이터베이스에서 진행 중인 변경 사항을 읽으려면는 엔진별 API 작업을 AWS DMS 사용하여 소스 엔진의 트랜잭션 로그에서 변경 사항을 읽습니다. 다음은 이를 수행하는 방법의 몇 가지 예 AWS DMS 입니다.

- Oracle의 경우 Oracle LogMiner API 또는 바이너리 리더 API(bfile API)를 AWS DMS 사용하여 진행 중인 변경 사항을 읽습니다.는 시스템 변경 번호(SCN)를 기반으로 온라인 또는 아카이브 다시 실행 로그에서 진행 중인 변경 사항을 AWS DMS 읽습니다.
- Microsoft SQL Server의 경우 MS-Replication 또는 MS-CDC를 AWS DMS 사용하여 SQL Server 트랜잭션 로그에 정보를 씁니다. 그러면 SQL 서버에서 fn_dblog() 또는 fn_dump_dblog() 함수를 사용해 로그 시퀀스 번호(LSN)에 토대를 둔 트랜잭션 로그의 변경 사항을 읽습니다.
- MySQL의 경우는 행 기반 바이너리 로그(binlog)의 변경 사항을 AWS DMS 읽고 해당 변경 사항을 대상으로 마이그레이션합니다.
- PostgreSQL의 경우 논리적 복제 슬롯 AWS DMS 을 설정하고 test_decoding 플러그인을 사용하여 소스에서 변경 사항을 읽고 대상으로 마이그레이션합니다.
- Amazon RDS를 소스로 할 경우, CDC를 설정할 수 있는 백업을 권장합니다. 충분한 시간 동안(대개 24시간이면 충분) 소스 데이터베이스가 변경 로그를 유지하도록 구성하는 것이 좋습니다. 엔드포인트에 대한 특정 설정은 다음을 참조하세요.
 - Amazon RDS for Oracle: [AWS에 대한 관리형 Oracle 소스 구성 AWS DMS](#).
 - Amazon RDS for MySQL 및 Aurora MySQL: [AWS관리형 MySQL 호환 데이터베이스들의 소스로 사용 AWS DMS](#).
 - Amazon RDS for SQL Server: [클라우드 SQL Server DB 인스턴스에서 지속적 복제 설정](#).
 - Amazon RDS for PostgreSQL 및 Aurora PostgreSQL: PostgreSQL은 필요한 WAL을 자동으로 유지합니다.

다음은 진행 중인 복제 작업의 두 가지 유형입니다.

- 전체 로드+CDC – 이 작업은 기존 데이터를 마이그레이션한 다음, 소스 데이터베이스의 변경 내용을 기반으로 대상 데이터베이스를 업데이트합니다.
- CDC만 해당 – 대상 데이터베이스에 데이터를 확보한 후 진행 중인 변경 내용을 마이그레이션하는 작업입니다.

CDC 시작 지점에서 복제를 시작

여러 지점에서 AWS DMS 지속적 복제 작업(변경 데이터 캡처만 해당)을 시작할 수 있습니다. 여기에는 다음이 포함됩니다.

- 사용자 지정 CDC 시작 시간부터 - AWS Management Console 또는를 사용하여 복제 AWS CLI 를 시작할 타임스탬프 AWS DMS 를 제공할 수 있습니다. AWS DMS 그런 다음이 사용자 지정 CDC 시작 시간부터 지속적 복제 작업을 시작합니다.는 지정된 타임스탬프(UTC)를 SQL Server용 LSN 또는 Oracle용 SCN과 같은 기본 시작 지점으로 AWS DMS 변환합니다. AWS DMS 는 엔진별 메서드를 사용하여 소스 엔진의 변경 스트림을 기반으로 마이그레이션 작업을 시작할 위치를 결정합니다.

Note

StartFromContext 추가 연결 속성을 필수 타임스탬프로 설정하면 Db2가 소스로 사용자 지정 CDC 시작 시간을 제공합니다.

PostgreSQL을 소스로 사용할 경우 사용자 지정 CDC 시작 시간을 지원하지 않습니다.

PostgreSQL 데이터베이스 엔진에는 타임스탬프를 LSN 또는 SCN으로 매핑하는 방법이 없기 때문입니다. Oracle 및 SQL Server에는 그러한 방법이 있습니다.

- CDC 원래 시작 지점부터 - 소스 엔진 트랜잭션 로그의 원래 포인트에서 시작할 수도 있습니다. 타임스탬프가 트랜잭션 로그의 여러 네이티브 포인트를 나타낼 수 있기 때문에 경우에 따라이 접근 방식을 선호할 수 있습니다.는 다음 소스 엔드포인트에 대해이 기능을 AWS DMS 지원합니다.

- SQL Server
- PostgreSQL
- Oracle
- MySQL
- MariaDB

Note

다음 데이터베이스 엔드포인트는 CDC 네이티브 시작점 기능을 지원하지 않습니다.

- Amazon Aurora MySQL
- Amazon Aurora PostgreSQL
- Amazon DocumentDB(MongoDB 호환)
- Amazon S3
- IBM Db2 for z/OS

- IBM DB2 LUW
- Microsoft Azure SQL Database
- Microsoft Azure SQL 관리형 인스턴스
- MongoDB
- SAP Sybase ASE

작업이 생성되면 CDC 시작점을 AWS DMS 표시하고 변경할 수 없습니다. 다른 CDC 시작 지점을 사용하려면 새 작업을 만드세요.

Note

CDC(Change Data Capture) 시작점을 지정할 때 마이그레이션 전 평가는 여전히 현재 환경의 기존 메타데이터 및 파라미터에 대한 전체 분석을 수행합니다. 이렇게 하면 지정된 CDC 시작점에 관계없이 모든 현재 구성 및 설정이 평가됩니다.

중요: 특정 작업에 대해 지난 7일 이내에 평가를 수행하지 않은 경우 데이터 완전성과 일관성을 보장하기 위해 마이그레이션 전 평가가 재개 및 재로드 모드 모두에서 자동으로 실행됩니다.

기술 세부 정보:

- 메타데이터 분석은 포괄적으로 유지됩니다.
- 파라미터 평가는 현재 상태를 다룹니다.
- CDC 시작점은 평가 범위를 제한하지 않습니다.
- 전체 시스템 구성 검토가 유지됩니다.
- 다음과 같은 경우 재개 및 다시 로드 모드에서 자동 실행:

마지막 평가 > 7일 전.

이전 평가 레코드를 찾을 수 없습니다.

이렇게 하면 지정된 CDC 지점과 현재 시스템 상태 간의 데이터 일관성을 유지하면서 정확한 마이그레이션 계획을 수립할 수 있습니다.

CDC 기본 시작점 결정

CDC 기본 시작점은 CDC를 시작할 수 있는 시간을 정의하는 데이터베이스 엔진 로그의 지점입니다. 예를 들어, 대량 데이터 덤프가 이미 대상에 적용된 경우를 생각해볼 수 있습니다. 진행 중인 복제 전용 작업의 기본 시작점을 찾아볼 수 있습니다. 데이터 불일치를 방지하려면 복제 전용 작업의 시작 지점을 신중하게 선택하세요. DMS는 선택한 CDC 시작 지점 이후에 시작된 트랜잭션을 캡처합니다.

다음은 지원되는 소스 엔진에서 CDC 기본 시작점을 찾는 방법에 대한 예입니다.

SQL Server

SQL Server의 로그 시퀀스 번호(LSN)는 세 부분으로 구성되어 있습니다.

- 가상 로그 파일(VLF) 시퀀스 번호
- 로그 블록의 시작 오프셋
- 슬롯 번호

LSN의 예는 00000014:00000061:0001과 같습니다.

트랜잭션 로그 백업 설정에 따라 SQL Server 마이그레이션 작업의 시작 지점을 확인하려면 SQL Server의 `fn_dblog()` 또는 `fn_dump_dblog()` 함수를 사용합니다.

SQL Server에서 CDC 기본 시작점을 사용하려면 진행 중인 복제에 참여하는 테이블에 게시를 만드세요. AWS DMS는 CDC 기본 시작점을 사용하지 않고 CDC를 사용하면 자동으로 간행물을 만듭니다.

PostgreSQL

PostgreSQL 소스 데이터베이스에 CDC 복구 체크포인트를 사용할 수 있습니다. 이 체크포인트 값은 다양한 지점에서 소스 데이터베이스에 대해 지속적 복제 작업(상위 작업)으로 생성됩니다. 일반적인 체크포인트에 관한 자세한 내용은 [체크포인트를 CDC 시작 지점으로 사용](#) 섹션을 참조하세요.

기본 시작점으로 사용할 체크포인트를 식별하려면 데이터베이스 `pg_replication_slots` 보기 또는 상위 작업의 개요 세부 정보를 사용합니다. AWS Management Console

콘솔에서 상위 작업에 대한 개요 세부 정보를 찾으려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

IAM 사용자로 로그인한 경우, AWS DMS에 액세스하려면 적절한 권한이 있어야 합니다. 필요한 권한에 관한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

2. 탐색 창에서 데이터베이스 마이그레이션 작업을 선택합니다.
3. 데이터베이스 마이그레이션 작업 페이지의 목록에서 상위 작업을 선택합니다. 그러면 개요 세부 정보가 표시된 상위 작업 페이지가 열립니다.
4. 변경 데이터 캡처(CDC), 변경 데이터 캡처(CDC) 시작 위치 및 변경 데이터 캡처(CDC) 복구 체크포인트 아래에서 체크포인트 값을 찾습니다.

이 값은 다음과 유사합니다.

```
checkpoint:V1#1#000004AF/B00000D0#0#0#*#0#0
```

여기서 4AF/B00000D0 구성 요소는 이 기본 CDC 시작점을 지정하는 데 필요한 것입니다. PostgreSQL 원본의 이 시작 지점에서 복제를 시작하는 CDC 작업을 만들 때 DMS API `CdcStartPosition` 파라미터를 이 값으로 설정합니다. 를 사용하여이 CDC 작업을 생성하는 AWS CLI 방법에 대한 자세한 내용은 섹션을 참조하세요 [를 사용하여 AWS관리형 PostgreSQL DB 인스턴스로 CDC 활성화 AWS DMS](#).

Oracle

시스템 변경 번호(SCN)는 Oracle 데이터베이스에서 사용하는 논리적 내부 타임스탬프입니다. SCN은 트랜잭션의 ACID 속성을 충족하는 데 필요한 (데이터베이스 내부에서 발생하는) 이벤트를 배열합니다. Oracle 데이터베이스는 SCN을 사용하여 디스크에 작성된 모든 변화가 진행된 위치를 표시합니다. 그리하여 복원 작업이 이미 작성된 변경 사항을 적용하지 않습니다. 또한 Oracle은 SCN을 사용하여 복구가 중지되지 않도록 데이터세트에 다시 실행이 없는 지점을 표시합니다.

다음 명령을 실행해 Oracle 데이터베이스의 현재 SCN을 가져옵니다.

```
SELECT CURRENT_SCN FROM V$DATABASE
```

SCN 또는 타임스탬프를 사용하여 CDC 작업을 시작하면 열린 트랜잭션의 결과를 놓치고 해당 결과를 마이그레이션하지 못하게 됩니다. 열린 트랜잭션은 작업 시작 지점 이전에 시작되고 작업 시작 지점 이후에 커밋된 트랜잭션입니다. SCN과 타임스탬프를 식별하여 모든 열린 트랜잭션이 포함된 시점에서 CDC 작업을 시작할 수 있습니다. 자세한 내용은 Oracle 온라인 설명서의 [트랜잭션](#)을 참조하세요. 버전 3.5.1 이상에서는 SCN 또는 타임스탬프를 사용하여 작업을 시작하는 경우 `openTransactionWindow` 엔드포인트 설정을 사용하여 CDC 전용 작업에 대한 열린 트랜잭션을 AWS DMS 지원합니다.

`openTransactionWindow` 설정을 사용하는 경우 열린 트랜잭션을 처리하려면 창을 분 단위로 제공해야 합니다.는 캡처 위치를 AWS DMS 전환하고 데이터 캡처를 시작할 새 위치를 찾습니다.는

필요한 Oracle 다시 실행 또는 아카이브된 다시 실행 로그에서 열린 트랜잭션을 스캔하기 위해 새 시작 위치를 AWS DMS 사용합니다.

MySQL

MySQL 버전 5.6.3이 출시되기 전에는 MySQL 로그 시퀀스 번호(LSN)는 4바이트의 부호 없는 정수였습니다. redo 로그 파일 크기 한도가 4GB에서 512GB로 증가한 MySQL 5.6.3부터 LSN은 8바이트의 부호 없는 정수가 되었습니다. 추가 사이즈 정보를 저장하기 위해 증가되면 바이트 추가가 요청됩니다. LSN 값을 사용하는 MySQL 5.6.3 또는 이후 버전에 빌드된 애플리케이션은 LSN 값을 저장해 비교하기 위해 32비트 대신 64비트 변수를 사용해야 합니다. MySQL LSN에 관한 자세한 내용은 [MySQL 설명서](#)를 참조하세요.

다음 명령을 실행해 MySQL 데이터베이스의 현재 LSN을 가져옵니다.

```
mysql> show master status;
```

쿼리는 binlog 파일 이름, 위치 및 기타 값을 반환합니다. CDC 기본 시작점은 binlogs 파일 이름과 위치의 조합으로 표시됩니다. 예를 들어, mysql-bin-changelog.000024:373입니다. 이 예제에서 mysql-bin-changelog.000024는 binlogs 파일 이름이고 373는가 변경 사항 캡처 AWS DMS 를 시작해야 하는 위치입니다.

체크포인트를 CDC 시작 지점으로 사용

지속적 복제 작업은 변경 사항을 마이그레이션하고 AWS DMS 때때로와 관련된 체크포인트 정보를 AWS DMS 캐싱합니다. AWS DMS 가 생성한 체크포인트에는 정보가 포함되어 있으므로 복제 엔진이 변경 스트림의 복구 지점을 파악할 수 있습니다. 체크포인트를 사용하여 변경 타임라인으로 되돌아가거나 실패한 마이그레이션 작업을 복구합니다. 또한 이 체크포인트를 사용해 특정 시간에 다른 대상으로 다른 지속적 복제 작업을 시작할 수 있습니다.

다음 세 가지 방법 중 하나로 체크포인트 정보를 얻을 수 있습니다.

- API 작업 DescribeReplicationTasks를 실행하고 결과를 검토합니다. 작업 별로 정보를 필터링하고 체크포인트를 검색할 수 있습니다. 작업이 중지/실패 상태일 때 마지막 체크포인트를 검색할 수 있습니다. 작업이 삭제되면 이 정보는 없어집니다.
- 대상 인스턴스에서 awsdms_txn_state라는 메타데이터를 봅니다. 테이블을 쿼리하여 체크포인트 정보를 얻을 수 있습니다. 메타데이터 테이블을 생성하기 위해서 작업을 생성할 때 TaskRecoveryTableEnabled 파라미터를 Yes로 설정합니다. 이 설정을 사용하면 AWS DMS 가 대상 메타데이터 테이블에 체크포인트 정보를 지속적으로 기록합니다. 작업이 삭제되면 이 정보는 없어집니다.

예를 들어, `checkpoint:V1#34#00000132/0F000E48#0#0#*#0#121`은 메타데이터 테이블의 체크포인트 예입니다.

- 탐색 창에서 데이터베이스 마이그레이션 작업을 선택하고 데이터베이스 마이그레이션 작업 페이지에 나타나는 목록에서 상위 작업을 선택합니다. 개요 세부 정보가 표시된 상위 작업 페이지가 열립니다. 변경 데이터 캡처(CDC), 변경 데이터 캡처(CDC) 시작 위치 및 변경 데이터 캡처(CDC) 복구 체크포인트 아래에서 체크포인트 값을 찾습니다. 체크포인트 값은 다음과 유사합니다.

```
checkpoint:V1#1#000004AF/B00000D0#0#0#*#0#0
```

커밋 또는 서버 시점에서 작업 중지

CDC 네이티브 시작점이 도입됨에 따라 다음 시점에서 작업을 중지할 수도 AWS DMS 있습니다.

- 소스의 약정 시간
- 복제 인스턴스의 서버 시간

필요에 따라 작업을 중지할 수 있도록 작업을 수정하고 시간을 UTC로 설정할 수 있습니다. 설정한 약정 또는 서버 시간에 따라 작업은 자동으로 중지됩니다. 또는 작업 생성 시 마이그레이션 작업을 중지할 적절한 시점을 알고 있다면 작업을 생성할 때 중지 시간을 설정할 수 있습니다.

Note

새 AWS DMS Serverless 복제를 처음 시작할 때 모든 리소스를 초기화하는 데 최대 40분이 걸릴 수 있습니다. 단, 이 `server_time` 옵션은 리소스 초기화가 완료된 후에만 적용할 수 있습니다.

대상 재로드로 작업 시작

재로드 옵션("DMS API의 재로드 대상")을 사용하여 기존 데이터 AWS DMS 마이그레이션을 시작하고 진행 중인 변경 작업을 복제할 수 있습니다. 이 경우 마이그레이션은 처음부터 시작되며, 각 테이블 데이터를 다시 로드하고 전체 로드 및 CDC 지원 작업 설정을 사용하여 데이터 복제를 계속합니다.

재로드 옵션으로 작업 시작을 사용하려면 다음 조건이 적용되어야 합니다.

- 작업을 중지해야 합니다.
- 작업의 마이그레이션 방법이 전체 로드 또는 CDC로 전체 로드여야 합니다.

DMS는 테이블을 다시 로드하기 전에 TargetTablePrepMode 설정을 적용합니다. 를 TargetTablePrepMode로 설정한 경우 먼저 테이블을 수동으로 잘라내DO_NOTHING야 합니다.

Note

대상 재로드 옵션으로 DMS 작업이 시작되면 사전 마이그레이션 평가는 현재 환경의 기존 메타데이터 및 파라미터에 대한 전체 분석을 수행합니다. 이렇게 하면 실제 작업 상태에 관계없이 모든 현재 구성 및 설정이 평가됩니다.

Important

특정 작업에 대해 지난 7일 이내에 평가를 수행하지 않은 경우 데이터 완전성과 일관성을 보장하기 위해 마이그레이션 전 평가가 자동으로 실행됩니다.

양방향 복제 수행

AWS DMS 작업을 사용하여 두 시스템 간에 양방향 복제를 수행할 수 있습니다. 양방향 복제에서는 두 시스템 간에 동일한 테이블(또는 테이블 집합)의 데이터를 두 방향으로 모두 복제할 수 있습니다.

예를 들어 데이터베이스 A에서 데이터베이스 B로 EMPLOYEE 테이블을 복사하고 데이터베이스 A에서 데이터베이스 B로 테이블에 변경 사항을 복제할 수 있습니다. 또한 데이터베이스 B에서 A로 다시 EMPLOYEE 테이블에 변경 사항을 복제할 수 있습니다. 즉, 양방향 복제를 수행할 수 있습니다.

Note

AWS DMS 양방향 복제는 프라이머리 노드, 충돌 해결 등을 포함한 전체 멀티 마스터 솔루션으로 사용하기 위한 것이 아닙니다.

서로 다른 노드의 데이터가 운영상 분리되는 경우에 양방향 복제를 사용합니다. 즉, 노드 A에서 작동하는 애플리케이션에 의해 변경된 데이터 요소가 있고 노드 A가 노드 B와 양방향 복제를 수행하는 경우, 노드 A의 데이터 요소는 노드 B에서 작동하는 애플리케이션에 의해 변경되지 않습니다.

AWS DMS 는 다음과 같은 데이터베이스 엔진에서 양방향 복제를 지원합니다.

- Oracle

- SQL Server
- MySQL
- PostgreSQL
- Amazon Aurora MySQL 호환 버전
- Aurora PostgreSQL 호환 버전

양방향 복제 작업 생성

AWS DMS 양방향 복제를 활성화하려면 데이터베이스(A 및 B) 모두에 대해 소스 및 대상 엔드포인트를 구성합니다. 예를 들어 데이터베이스 A의 소스 엔드포인트, 데이터베이스 B의 소스 엔드포인트, 데이터베이스 A의 대상 엔드포인트 및 데이터베이스 B의 대상 엔드포인트를 구성합니다.

그런 다음 두 가지 작업(데이터를 대상 B로 이동하기 위한 소스 A에 대한 작업과 데이터를 대상 A로 이동하기 위한 소스 B에 대한 작업)을 만듭니다. 또한 각 작업이 루프백 방지로 구성되어 있는지 확인합니다. 작업이 루프백 방지가 설정되어 있으면 동일한 변경 사항이 두 작업의 대상에 적용되지 않으므로 하나 이상의 작업에 대한 데이터가 손상됩니다. 자세한 내용은 [루프백 방지](#) 단원을 참조하십시오.

가장 간단한 접근 방식은 데이터베이스 A와 데이터베이스 B 모두에서 동일한 데이터세트로 시작합니다. 그런 다음 두 개의 CDC 전용 작업(A에서 B로 데이터를 복제하기 위한 작업과 B에서 A로 데이터를 복제하기 위한 작업)을 만듭니다.

AWS DMS 를 사용하여 노드 A의 노드 B에서 새 데이터 세트(데이터베이스)를 인스턴스화하려면 다음을 수행합니다.

1. 전체 로드 및 CDC 작업을 사용하여 데이터베이스 A에서 B로 데이터를 이동합니다. 이 때 데이터베이스 B의 데이터를 수정하는 애플리케이션이 없는지 확인합니다.
2. 전체 로드가 완료되고 애플리케이션이 데이터베이스 B의 데이터를 수정할 수 있도록 허용되기 전에 데이터베이스 B의 시작 시간 또는 CDC 시작 위치를 기록해 둡니다. 자세한 내용은 [CDC 시작 지점에서 복제를 시작](#) 섹션을 참조하세요.
3. 이 시작 시간 또는 CDC 시작 위치를 사용하여 데이터베이스 B에서 A로 데이터를 다시 이동하는 CDC 전용 작업을 만듭니다.

Note

양방향 쌍에서 하나의 작업만 전체 로드 및 CDC일 수 있습니다.

루프백 방지

루프백 방지를 표시하려면 작업에서 T1 AWS DMS reads가 소스 데이터베이스 A에서 로그를 변경하고 대상 데이터베이스 B에 변경 사항을 적용한다고 가정합니다.

그런 다음 두 번째 작업 T2는 소스 데이터베이스 B에서 변경 로그를 읽고 변경 사항을 대상 데이터베이스 A에 다시 적용합니다. T2가 이를 수행하기 전에 DMS는 소스 데이터베이스 A의 대상 데이터베이스 B에 적용된 동일한 변경 사항이 소스 데이터베이스 A에 적용되지 않았는지 확인해야 합니다. 즉, DMS는 이러한 변경 사항이 대상 데이터베이스 A에 반영(순환)되지 않는지 확인해야 합니다. 그렇지 않으면 데이터베이스 A의 데이터가 손상될 수 있습니다.

변경 사항의 루프백을 방지하려면 각 양방향 복제 작업에 다음 작업 설정을 추가합니다. 이렇게 하면 루프백 데이터 손상이 어느 방향으로도 발생하지 않습니다.

```
{
  . . .

  "LoopbackPreventionSettings": {
    "EnableLoopbackPrevention": Boolean,
    "SourceSchema": String,
    "TargetSchema": String
  },
  . . .
}
```

LoopbackPreventionSettings 작업 설정은 트랜잭션이 새 트랜잭션인지 또는 반대 복제 작업의 에코인지를 결정합니다. AWS DMS가 대상 데이터베이스에 트랜잭션을 적용하면 변경 표시와 함께 DMS 테이블(awsdms_loopback_prevention)을 갱신됩니다. 각 트랜잭션을 대상에 적용하기 전에 DMS는 이 awsdms_loopback_prevention 테이블에 대한 참조를 포함하는 모든 트랜잭션을 무시합니다. 따라서 변경 사항이 적용되지 않습니다.

양방향 쌍으로 된 각 복제 작업에 이러한 작업 설정을 포함합니다. 이러한 설정은 루프백 방지를 활성화합니다. 또한 각 엔드포인트에 대한 awsdms_loopback_prevention 테이블을 포함하는 작업의 각 소스 데이터베이스 및 대상 데이터베이스에 대한 스키마를 지정합니다.

각 작업에서 이러한 에코를 식별하고 무시하도록 하려면 EnableLoopbackPrevention을 true로 설정합니다. awsdms_loopback_prevention을 포함하는 소스에서 스키마를 지정하려면 SourceSchema를 소스 데이터베이스의 해당 스키마 이름으로 설정합니다. 동일한 테이블을 포함하는 대상에서 스키마를 지정하려면 TargetSchema를 대상 데이터베이스의 해당 스키마 이름으로 설정합니다.

다음 예제에서는 복제 작업 T1 및 해당 반대 복제 작업 T2에 대한 SourceSchema 및 TargetSchema 설정이 반대 설정으로 지정됩니다.

작업 T1에 대한 설정은 다음과 같습니다.

```
{
  . . .

  "LoopbackPreventionSettings": {
    "EnableLoopbackPrevention": true,
    "SourceSchema": "LOOP-DATA",
    "TargetSchema": "loop-data"
  },
  . . .
}
```

반대 작업 T2에 대한 설정은 다음과 같습니다.

```
{
  . . .

  "LoopbackPreventionSettings": {
    "EnableLoopbackPrevention": true,
    "SourceSchema": "loop-data",
    "TargetSchema": "LOOP-DATA"
  },
  . . .
}
```

Note

를 사용하는 경우 create-replication-task 또는 modify-replication-task 명령만 AWS CLI 사용하여 양방향 복제 작업 LoopbackPreventionSettings에서 구성합니다.

양방향 복제의 제한 사항

에 대한 양방향 복제 AWS DMS 에는 다음과 같은 제한 사항이 있습니다.

- 루프백 방지는 데이터 조작 언어(DML) 문만 추적합니다. AWS DMS 는 데이터 정의 언어(DDL) 루프백 방지를 지원하지 않습니다. 이렇게 하려면 양방향 쌍의 작업 중 하나를 구성하여 DDL 문을 필터링합니다.
- 루프백 방지를 사용하는 작업은 배치로 변경 사항을 커밋하는 것을 지원하지 않습니다. 루프백 방지로 작업을 구성하려면 `BatchApplyEnabled`를 `false`로 설정해야 합니다.
- DMS 양방향 복제에는 충돌 감지 또는 충돌 해결이 포함되지 않습니다. 데이터 불일치를 감지하려면 두 작업 모두에서 데이터 검증을 사용합니다.
- 양방향 복제를 설정하려면 SQL Server 소스 엔드포인트 `setUpMsCdcForTables`를 `true`로 설정해야 합니다.

작업 수정

작업 설정, 테이블 매핑 또는 기타 설정을 변경해야 하는 경우 작업을 수정할 수 있습니다. 수정된 태스크를 실행하기 전에 마이그레이션 전 태스크 평가를 활성화하고 실행할 수 있습니다. 콘솔에서 태스크를 수정하려면 태스크를 선택하고 수정을 선택합니다. CLI 명령 또는 API 작업 [ModifyReplicationTask](#)를 사용할 수도 있습니다.

작업을 수정하는 몇 가지 제한 사항이 있습니다. 여기에는 다음이 포함됩니다.

- 작업의 원본 또는 대상 엔드포인트를 수정할 수 없습니다.
- 작업의 마이그레이션 유형은 변경할 수 없습니다.
- 실행한 태스크의 상태가 중지됨 또는 실패여야 수정할 수 있습니다.

태스크 이동

사용 사례에 다음과 같은 상황이 적용될 경우 태스크를 다른 복제 인스턴스로 이동할 수 있습니다.

- 현재 특정 유형의 인스턴스를 사용 중이고 다른 인스턴스 유형으로 전환하려고 하는 경우.
- 현재 인스턴스에 많은 복제 태스크로 인해 과부하가 걸린 상태이고, 부하를 여러 인스턴스로 분산하려고 하는 경우.
- 인스턴스 스토리지가 꽉 차 있고, 스토리지가 컴퓨팅이 크기를 조정하는 대신 해당 인스턴스에서 더 강력한 인스턴스로 태스크를 이동하려는 경우.
- 의 새로 릴리스된 기능을 사용하고 싶지 AWS DMS만 새 작업을 생성하고 마이그레이션을 다시 시작하지 않으려고 합니다. 대신 기능을 지원하는 새 AWS DMS 버전으로 복제 인스턴스를 실행하고 기존 작업을 해당 인스턴스로 이동하는 것이 좋습니다.

콘솔에서 태스크를 이동하려면 태스크를 선택하고 이동을 선택합니다. CLI 명령 또는 API 작업 `MoveReplicationTask`를 실행하여 태스크를 이동할 수도 있습니다. 데이터베이스 엔진을 대상 엔드포인트로 보유한 태스크를 이동할 수 있습니다.

대상 복제 인스턴스에 이동 중인 태스크를 수용할 수 있는 충분한 스토리지 공간이 있는지 확인합니다. 그렇지 않을 경우, 태스크를 이동하기 전에 스토리지의 크기를 조정하여 대상 복제 인스턴스를 위한 공간을 확보합니다.

또한 대상 복제 인스턴스가 현재 복제 인스턴스와 동일하거나 더 높은 AWS DMS 엔진 버전으로 생성되었는지 확인합니다.

Note

- 태스크는 현재 상주하고 있는 동일한 복제 인스턴스로 이동할 수 없습니다.
- 이동 중에는 태스크 설정을 수정할 수 없습니다.
- 태스크를 이동하려면 실행한 태스크의 상태가 중지됨, 실패 또는 이동 실패 상태여야 합니다.

DMS 태스크 이동과 관련된 태스크 상태로는 이동 중과 이동 실패라는 두 가지 상태가 있습니다. 이러한 태스크 상태에 대한 자세한 내용은 [작업 상태](#) 섹션을 참조하세요.

태스크를 이동한 후에는 이동된 태스크를 실행하기 전에 마이그레이션 전 평가를 활성화하고 실행하여 차단 문제가 있는지 확인할 수 있습니다.

작업 중 테이블 다시 로드

작업을 실행하는 동안 원본의 데이터를 사용하여 대상 데이터베이스 테이블을 다시 로드할 수 있습니다. 작업 중에 파티션 작업으로 인해 오류가 발생하거나 데이터가 변경될 경우(예: Oracle을 사용하는 경우) 테이블을 다시 로드해야 할 수도 있습니다. 작업에서 최대 10개 테이블을 다시 로드할 수 있습니다.

테이블을 다시 로드해도 태스크가 중단되지는 않습니다.

테이블을 다시 로드하려면, 다음 조건을 적용해야 합니다.

- 작업이 실행 중이어야 합니다.
- 작업의 마이그레이션 방법이 전체 로드 또는 CDC로 전체 로드여야 합니다.

- 중복 테이블은 허용되지 않습니다.
- AWS DMS 는 이전에 읽은 테이블 정의를 유지하며 다시 로드 작업 중에 다시 생성하지 않습니다. ALTER TABLE ADD COLUMN 또는 DROP COLUMN 등 테이블이 다시 로드되기 전에 테이블에 만들어진 DDL 문으로 인해 다시 로드 작업이 실패할 수 있습니다.

 Note

DMS는 테이블을 다시 로드하기 전에 TargetTablePrepMode 설정을 적용합니다. TargetTablePrepMode를 DO_NOTHING으로 설정하는 경우 먼저 테이블을 수동으로 잘라야 합니다.

AWS Management Console

AWS DMS 콘솔을 사용하여 테이블을 다시 로드하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

IAM 사용자로 로그인한 경우, AWS DMS에 액세스하려면 적절한 권한이 있어야 합니다. 필요한 권한에 관한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

2. 탐색 창에서 작업을 선택합니다.
3. 다시 로드할 테이블이 있는 실행 중인 작업을 선택합니다.
4. 테이블 통계 탭을 선택합니다.

The screenshot shows the AWS DMS console interface for a task named 'move-data'. At the top, there are buttons for 'Create task', 'Assess', 'Modify', 'Start/Resume', 'Stop', and 'Delete'. Below these is a 'Filter' input field. A table lists the task details: ID 'move-data', Status 'Running', Source 'from-mysql-sou', Target 'to-pgsql-target', and Type 'Full Load'. Below this is a progress bar. The main section is titled 'move-data' and contains tabs for 'Overview', 'Task monitoring', 'Table statistics' (which is selected), 'Logs', and 'Assessment results'. Under the 'Table statistics' tab, there are buttons for 'Revalidate' and 'Reload table data', with the latter being circled in red. Below these buttons is another 'Filter' input field and a table showing table statistics.

	Schema	Table	Load State	Inserts	Deletes	Updates	DDLs	Full Load
☒	employees	departments	Table completed	0	0	0	0	9
☐	employees	dept_emp	Table completed	0	0	0	0	331,6
☐	employees	dept_manager	Table completed	0	0	0	0	24

- 다시 로드할 테이블을 선택합니다. 작업이 더 이상 실행되고 있지 않으면 테이블을 다시 로드할 수 없습니다.
- 테이블 데이터 다시 로드를 선택합니다.

AWS DMS 가 테이블을 다시 로드할 준비를 하면 콘솔이 테이블 상태를 다시 로드 중인 테이블로 변경합니다.

작업 설정을 지정하기 위한 테이블 매핑 사용

테이블 매핑에서는 여러 유형의 규칙을 사용하여 데이터 소스, 소스 스키마, 데이터, 작업 중 발생하는 모든 변환을 지정합니다. 테이블 매핑을 사용하여 마이그레이션할 데이터베이스의 개별 테이블과 마이그레이션에 사용할 스키마를 지정할 수 있습니다.

테이블 매핑 작업 시 필터를 사용하여 테이블 열에서 복제할 데이터를 지정할 수 있습니다. 또한 변환을 사용하여 선택한 스키마, 테이블 또는 뷰를 대상 데이터베이스에 기록하기 전에 수정할 수 있습니다.

주제

- [콘솔에서 테이블 선택 및 변환 규칙 지정](#)
- [JSON을 사용하여 테이블 선택 및 변환 지정](#)
- [선택 규칙 및 작업](#)
- [테이블 매핑의 와일드카드](#)
- [변환 규칙 및 작업](#)
- [변환 규칙 표현식을 사용하여 열 내용 정의](#)
- [테이블 및 컬렉션 설정 규칙과 작업](#)
- [데이터 마스킹을 사용하여 민감한 정보 숨기기](#)

Note

MongoDB 소스 엔드포인트에 대한 테이블 매핑 작업을 수행하는 경우, 필터를 사용하여 복제할 데이터를 지정하고 `schema_name` 대신 데이터베이스 이름을 지정할 수 있습니다. 아니면 기본 "%"를 사용할 수 있습니다.

콘솔에서 테이블 선택 및 변환 규칙 지정

AWS Management Console 를 사용하여 테이블 선택 및 변환 지정을 포함하여 테이블 매핑을 수행할 수 있습니다. 콘솔에서 위치 섹션을 사용하여 스키마, 테이블, 작업을 지정합니다(포함 또는 제외). 필터 섹션을 사용하여 테이블에서 열 이름과 복제 작업에 적용할 조건을 지정합니다. 이와 함께 이 두 작업은 선택 규칙을 생성합니다.

1개 이상의 선택 규칙을 지정한 후 테이블 매핑에 변환을 포함시킬 수 있습니다. 변환을 사용하여 스키마나 테이블 이름을 변경하거나, 스키마나 테이블에 접두사나 접미사를 추가하거나, 또는 테이블 열을 제거할 수 있습니다.

 Note

AWS DMS 는 스키마 수준, 테이블 수준 또는 열 수준당 둘 이상의 변환 규칙을 지원하지 않습니다.

다음 절차에서는 **EntertainmentAgencySample**이라는 스키마에서 호출된 **Customers**라는 테이블을 기반으로 선택 규칙을 설정하는 방법을 보여줍니다.

콘솔을 사용하여 테이블 선택, 필터 기준, 변환을 지정하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

IAM 사용자로 로그인한 경우, AWS DMS에 액세스하려면 적절한 권한이 있어야 합니다. 필요한 권한에 관한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

2. 대시보드 페이지에서 데이터베이스 마이그레이션 작업을 선택합니다.
3. 작업 생성을 선택합니다.
4. 작업 구성 섹션에서 작업 식별자, 복제 인스턴스, 소스 데이터베이스 엔드포인트, 대상 데이터베이스 엔드포인트, 마이그레이션 유형 등의 작업 정보를 입력합니다.

DMS > Database migration tasks > Create database migration task

Create database migration task

Task configuration

Task identifier

Replication instance

Source database endpoint

Target database endpoint

Migration type [Info](#)

- 테이블 매핑 섹션에서 스키마 이름과 테이블 이름을 입력합니다. 스키마 이름 또는 테이블 이름을 지정할 때 "%"를 와일드카드 값으로 사용할 수 있습니다. 사용할 수 있는 다른 와일드카드에 관한 자세한 내용은 [the section called “테이블 매핑의 와일드카드”](#)을 참조하세요. 수행할 작업을 지정하여 필터에서 정의한 데이터를 포함하거나 제외합니다.

Table mappings

Editing mode [Info](#)

☒ **Wizard**
You can enter only a subset of the available table mappings.

☐ **JSON editor**
You can enter all available table mappings directly in JSON format.

Specify at least one selection rule with an include action. After you do this, you can add one or more transformation rules.

▼ Selection rules

Choose the schema and/or tables you want to include with, or exclude from, your migration task. [Info](#) **Add new selection rule**

▼ where schema name is like 'MySchema' and table name is like '%', include

Schema
Enter a schema ▼

Schema name
Use the % character as a wildcard
MySchema

Table name
Use the % character as a wildcard
%

Action
Choose "Include" to migrate your selected objects, or "Exclude" to ignore them during the migration.
Include ▼

6. 열 필터 추가와 조건 추가 링크를 사용하여 필터 정보를 지정합니다.

- a. 열 필터 추가를 선택하여 열과 조건을 지정합니다.
- b. 조건 추가를 선택하여 조건을 추가할 수 있습니다.

다음은 01 및 85 사이에 **AgencyIDs**를 포함하는 **Customers** 테이블에 대한 필터를 보여주는 예입니다.

Source filters [Info](#) [Add column filter](#)

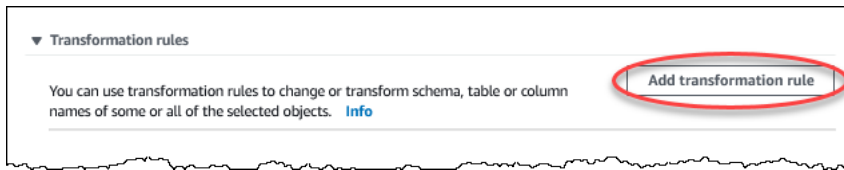
▼ Column filter 1 X

Column name
AgencyId

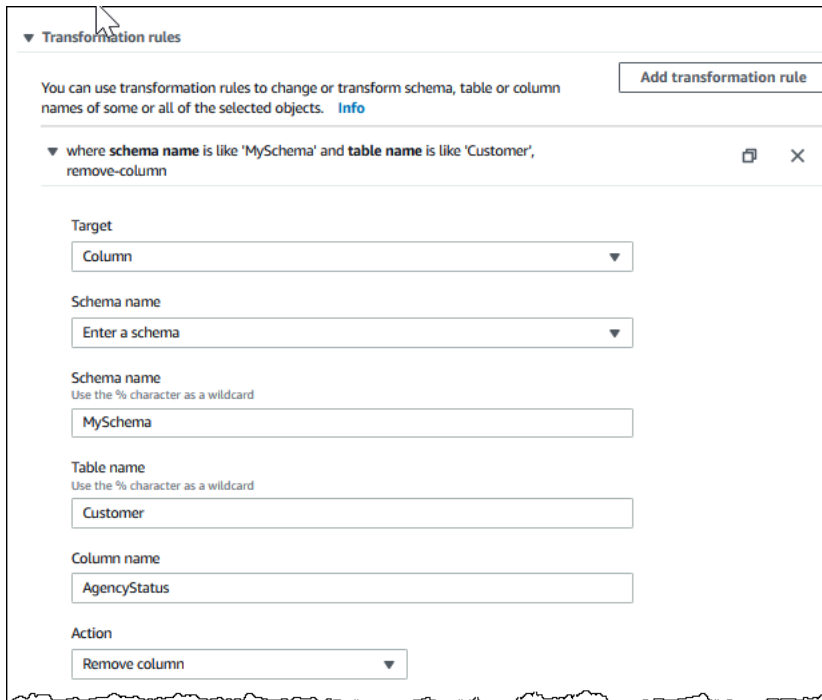
Condition 1
Equal to or between two values ▼ 01 85

[Add condition](#)

7. 원하는 대로 선택하고 새 선택 규칙 추가를 선택합니다.
8. 최소 1개 이상의 선택 규칙을 생성한 후 작업에 변환을 추가할 수 있습니다. 변환 규칙 추가를 선택합니다.



9. 변환할 대상을 선택하고 요청된 추가 정보를 입력합니다. 다음은 **Customer** 테이블에서 **AgencyStatus** 열을 삭제하는 변환을 보여주는 예입니다.



10. 변환 규칙 추가를 선택합니다.

11. 작업 생성을 선택합니다.

Note

AWS DMS 는 스키마 수준, 테이블 수준 또는 열 수준당 둘 이상의 변환 규칙을 지원하지 않습니다.

JSON을 사용하여 테이블 선택 및 변환 지정

마이그레이션 중에 적용할 테이블 매핑을 지정하려면 JSON 파일을 만들 수 있습니다. 콘솔을 사용하여 마이그레이션 작업을 만드는 경우, 이 JSON 파일을 찾아보거나 JSON을 테이블 매핑 상자에 직접 입력할 수 있습니다. CLI 또는 API를 사용하여 마이그레이션을 수행하는 경우,

CreateReplicationTask 또는 ModifyReplicationTask API 작업의 TableMappings 파라미터를 사용하여 이 파일을 지정할 수 있습니다.

AWS DMS 는 최대 2MB 크기의 테이블 매핑 JSON 파일만 처리할 수 있습니다. DMS 작업을 수행하는 동안에는 매핑 규칙 JSON 파일 크기를 2MB 미만으로 유지하는 것이 좋습니다. 이렇게 하면 작업 생성 또는 수정 중에 예상치 못한 오류가 발생하지 않습니다. 매핑 규칙 파일이 2MB 제한을 초과하는 경우, 테이블을 여러 작업으로 분할하여 매핑 규칙 파일이 이 한도 이하로 유지되도록 크기를 줄이는 것이 좋습니다.

사용할 테이블, 뷰 및 스키마를 지정할 수 있습니다. 또한 테이블, 뷰 및 스키마 변환을 수행하고 AWS DMS 에서 개별 테이블 및 뷰를 로드하는 방법에 대한 설정을 지정할 수 있습니다. 다음 규칙 유형을 사용하여 이러한 옵션에 대한 테이블 매핑 규칙을 만듭니다.

- selection 규칙 – 로드할 소스 테이블, 뷰 및 스키마의 유형과 이름을 식별합니다. 자세한 내용은 [선택 규칙 및 작업](#) 단원을 참조하십시오.
- transformation 규칙 – 대상에 로드되기 전에 소스의 특정 소스 테이블 및 스키마에 대한 특정 변경 사항 또는 추가 사항을 지정합니다. 자세한 내용은 [변환 규칙 및 작업](#) 단원을 참조하십시오.

또한 변환 규칙 내에서 표현식을 사용하여 새 열과 기존 열의 내용을 정의할 수 있습니다. 자세한 내용은 [변환 규칙 표현식을 사용하여 열 내용 정의](#) 단원을 참조하십시오.

- table-settings 규칙 – DMS 작업이 개별 테이블에 대한 데이터를 로드하는 방법을 지정합니다. 자세한 내용은 [테이블 및 컬렉션 설정 규칙과 작업](#) 단원을 참조하십시오.

Note

Amazon S3 대상의 경우, post-processing 규칙 유형과 add-tag 규칙 작업을 사용하여 선택된 테이블 및 스키마로 매핑된 S3 객체에 태그를 지정할 수도 있습니다. 자세한 내용은 [Amazon S3 객체 태그 지정](#) 단원을 참조하십시오.

다음 대상의 경우 object-mapping 규칙 유형을 사용하여 선택한 스키마와 테이블을 대상으로 마이그레이션하는 방법과 위치를 지정할 수 있습니다.

- Amazon DynamoDB – 자세한 내용은 [객체 매핑을 사용하여 데이터를 DynamoDB로 마이그레이션](#)을 참조하세요.
- Amazon Kinesis – 자세한 내용은 [객체 매핑을 사용하여 데이터를 Kinesis 데이터 스트림으로 마이그레이션](#)을 참조하세요.
- Apache Kafka – 자세한 내용은 [객체 매핑을 사용하여 데이터를 Kafka 주제로 마이그레이션](#)을 참조하세요.

선택 규칙 및 작업

테이블 매핑을 사용하여 사용할 테이블, 뷰 및 스키마를 지정하기 위해 선택 규칙과 작업을 이용할 수 있습니다. 선택 규칙 유형을 사용하는 테이블 매핑 규칙의 경우, 다음 값을 적용할 수 있습니다.

Warning

이러한 규칙 내에 민감한 데이터를 포함하지 마세요.

파라미터	가능한 값	설명
rule-type	selection	선택 규칙입니다. 테이블 매핑을 지정할 때에는 1개 이상의 선택 규칙을 정의합니다.
rule-id	숫자 값.	규칙을 식별하기 위한 고유한 숫자 값입니다. 콘솔을 사용하여 규칙을 생성하면 콘솔에서 대신 이 값을 생성합니다.
rule-name	영숫자 값입니다.	규칙을 식별하기 위한 고유한 이름입니다. 콘솔을 사용하여 규칙을 생성하면 콘솔에서 대신 이 값을 생성합니다.
rule-action	include, exclude, explicit	규칙에서 선택한 특정 객체(단일 또는 다수)를 포함하거나 제외하는 값입니다. explicit이 지정되면 명시적으로 지정된 테이블 및 스키마에 해당하는 객체 하나만을 선택 및 포함할 수 있습니다.
object-locator	객체는 다음 파라미터를 사용합니다. - schema-name – 스키마의 이름입니다. - table-name – 테이블의 이름입니다.	규칙이 적용되는 각 스키마와 테이블 또는 뷰의 이름입니다. 또한 규칙에 테이블만 포함되는지, 뷰만 포함되는지, 또는 테이블과 뷰가 모두 포함되는지 여부를 지정할 수도 있습니다. rule-action 이 include이거나

파라미터	가능한 값	설명
	(선택 사항) table-type - table view all은 table-name 이 테이블, 뷰 또는 테이블과 뷰 모두를 지칭하는지를 나타냅니다. 기본값은 table입니다. AWS DMS 는 전체 로드 작업에서만 뷰를 로드합니다. 전체 로드 및 변경 데이터 캡처(CDC) 작업만 있는 경우, 뷰를 로드하도록 전체 로드 전용 작업을 1개 이상 구성하세요. 모든 대상 엔드포인트가 전체 로드 상태에서도 뷰를 복제 소스로 받아들이는 것은 아닙니다(예: Amazon OpenSearch Service). 대상 엔드포인트의 한계를 확인하세요.  Warning DMS 선택 규칙은 대/소문자를 구분합니다. 그러나 선택 결과는 소스 엔드포인트 데이터베이스 구성에 따라 달라집니다. 소스 엔드포인트가 대/소문자를 구분하지 않는 것으로 구성된 경우 객체 로케이터 값의 대/소문자는 중요하지 않습니다. 대/소문자를 구분하지 않는 엔드포인트의 DMS 선택 규칙에 올바른 객체 식별자가 사용되는지 확인합니다.	exclude인 경우 각 schema-name 및 table-name 파라미터의 값의 전부 또는 일부에 대한 와일드카드로 '%' 퍼센트 기호를 사용할 수 있습니다. 사용할 수 있는 다른 와일드카드에 관한 자세한 내용은 the section called “테이블 매핑의 와일드카드”을 참조하세요. 따라서 다음 항목들을 일치시킬 수 있습니다. - 단일 스키마의 단일한 테이블, 뷰 또는 컬렉션 - 일부 또는 전체 스키마의 단일한 테이블, 뷰 또는 컬렉션 - 단일 스키마의 일부 또는 모든 테이블과 뷰 또는 단일 데이터베이스의 컬렉션 - 일부 또는 모든 스키마의 일부 또는 전체 테이블과 뷰 또는 일부 또는 전체 데이터베이스의 컬렉션 rule-action 이 explicit인 경우, 단일 테이블 또는 뷰 및 스키마의 정확한 이름만을 지정할 수 있습니다 (와일드카드 없음). 뷰에 지원되는 소스에는 다음이 포함됩니다. - Oracle - Microsoft SQL Server - PostgreSQL - IBM DB2 LUW - IBM DB2 z/OS

파라미터	가능한 값	설명
		• SAP Adaptive Server Enterprise(ASE) • MySQL • AURORA MySQL • MariaDB  Note AWS DMS 는 소스 뷰를 대상 뷰에 로드하지 않습니다. 소스 뷰는 소스의 뷰와 이름이 동일한 대상의 동등한 테이블에 로드됩니다. 컬렉션이 포함된 데이터베이스에 대해 지원되는 소스는 다음과 같습니다. • MongoDB • Amazon DocumentDB
load-order	양의 정수입니다. 최대값은 2,147,483,647입니다.	테이블 및 뷰를 로드하기 위한 우선순위입니다. 우선순위 값이 높은 테이블 및 뷰가 먼저 로드됩니다.
filters	객체의 배열입니다.	소스를 필터링하기 위한 하나 이상의 객체입니다. 소스의 단일 열에서 필터링할 객체 파라미터를 지정합니다. 여러 열에서 필터링할 여러 객체를 지정합니다. 자세한 내용은 소스 필터 사용 단원을 참조하십시오.

Example스키마에서 모든 테이블 마이그레이션

다음은 소스에서 이름이 Test인 스키마의 모든 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      },
      "rule-action": "include"
    }
  ]
}
```

Example스키마에서 일부 테이블 마이그레이션

다음은 소스에서 이름이 Test인 스키마에서 DMS로 시작하는 테이블을 제외한 모든 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "selection",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "Test",
```

```

        "table-name": "DMS%"
      },
      "rule-action": "exclude"
    }
  ]
}

```

Example 단일 스키마에서 지정된 단일 테이블 마이그레이션

다음은 소스의 NewCust 스키마에서 Customer 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "NewCust",
        "table-name": "Customer"
      },
      "rule-action": "explicit"
    }
  ]
}

```

Note

여러 선택 규칙을 지정하여 여러 테이블 및 스키마에서 명시적으로 선택할 수 있습니다.

Example 설정된 순서대로 테이블 마이그레이션

테이블과 뷰는 로드 순서 값에 따라 마이그레이션되며, 값이 높을수록 마이그레이션 시퀀스에서 우선 순위를 받습니다. 다음 예제에서는 우선 순위 값이 2인 loadfirst, 우선 순위 값이 1인 loadsecond, 두 테이블을 마이그레이션합니다. 마이그레이션 작업은 loadsecond 테이블로 진행하기 전에 먼저 loadfirst 테이블을 처리합니다. 이 우선 순위 지정 메커니즘을 사용하면 마이그레이션 프로세스 중에 데이터베이스 객체 간의 종속성을 준수할 수 있습니다.

```

{

```

```

"rules": [
  {
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "Test",
      "table-name": "loadsecond"
    },
    "rule-action": "include",
    "load-order": "1"
  },
  {
    "rule-type": "selection",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
      "schema-name": "Test",
      "table-name": "loadfirst"
    },
    "rule-action": "include",
    "load-order": "2"
  }
]
}

```

Note

load-order는 테이블 초기화에 적용됩니다. MaxFullLoadSubTasks가 1보다 큰 경우 연속 테이블 로드는 이전 테이블 로드가 완료될 때까지 기다리지 않습니다.

Example스키마에서 일부 뷰 마이그레이션

다음은 소스에서 이름이 Test인 스키마의 일부 뷰를 대상의 동등한 테이블로 마이그레이션하는 예입니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "2",

```

```

        "rule-name": "2",
        "object-locator": {
            "schema-name": "Test",
            "table-name": "view_DMS%",
            "table-type": "view"
        },
        "rule-action": "include"
    }
]
}

```

Example스키마에서 모든 테이블 및 뷰 마이그레이션

다음은 소스에서 이름이 report인 스키마의 모든 테이블 및 뷰를 대상의 동등한 테이블로 마이그레이션하는 예입니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "3",
      "rule-name": "3",
      "object-locator": {
        "schema-name": "report",
        "table-name": "%",
        "table-type": "all"
      },
      "rule-action": "include"
    }
  ]
}

```

테이블 매핑의 와일드카드

이 섹션에서는 테이블 매핑을 위한 스키마와 테이블 이름을 지정할 때 사용할 수 있는 와일드카드에 대해 설명합니다.

와일드카드	일치 항목
%	Zero or more characters
_	A single character

와일드카드	일치 항목
[_]	A literal underscore character
[ab]	A set of characters. For example, [ab] matches either 'a' or 'b'.
[a-d]	A range of characters. For example, [a-d] matches either 'a', 'b', 'c', or 'd'.

Oracle 소스 및 대상 엔드포인트의 경우, `escapeCharacter` 추가 연결 속성을 사용하여 이스케이프 문자를 지정할 수 있습니다. 이스케이프 문자를 사용하면 와일드카드 문자를 와일드카드가 아닌 것처럼 표현식에 사용할 수 있습니다. 예를 들어, `escapeCharacter=#`를 사용하면 이 샘플 코드에서처럼 표현식에 '#'을 사용하여 와일드카드 문자가 일반 문자로 작동하도록 할 수 있습니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "542485267",
      "rule-name": "542485267",
      "object-locator": { "schema-name": "ROOT", "table-name": "TEST#_T%" },
      "rule-action": "include",
      "filters": []
    }
  ]
}
```

여기서 '#' 이스케이프 문자는 '_' 와일드카드 문자를 일반 문자로 만듭니다.는 스키마에서 테이블을 AWS DMS 선택합니다. ROOT여기서 각 테이블의 이름은 접두사TEST_T로가 붙습니다.

변환 규칙 및 작업

변환 작업을 사용하여 선택한 스키마, 테이블 또는 뷰에 적용할 변환을 지정합니다. 변환 규칙은 선택적입니다.

제한 사항

- 동일한 객체(스키마, 테이블, 열, 테이블-테이블스페이스 또는 인덱스-테이블스페이스)에 대해 둘 이상의 변환 규칙 작업을 적용할 수 없습니다. 각 변형 작업이 서로 다른 객체에 적용되는 한, 모든 레벨에 여러 변환 규칙 작업을 적용할 수 있습니다. 그러나 동일한 열에 대해 ADD-COLUMN 또는 CHANGE-DATA-TYPE와 같은 다른 변환을 수행할 수 있는 데이터 마스킹 변환 규칙을 사용할 때는 이 제한이 적용되지 않습니다.
- 변환 규칙의 테이블 이름과 열 이름은 대소문자를 구분합니다. 예를 들어, Oracle 또는 Db2 데이터베이스의 테이블 이름과 열 이름을 대문자로 제공해야 합니다.
- 오른쪽에서 왼쪽으로 쓰는 언어를 사용하는 열 이름에는 변환이 지원되지 않습니다.
- 이름에 특수 문자(예: #, \, /, -)가 포함된 열에서는 변환을 수행할 수 없습니다.
- BLOB/CLOB 데이터 형식에 매핑된 열에 대해 지원되는 유일한 변환은 대상에서 열을 삭제하는 것입니다.
- AWS DMS 는 두 소스 테이블을 단일 대상 테이블로 복제하는 것을 지원하지 않습니다.는 AWS DMS 복제 작업의 변환 규칙에 따라 테이블에서 테이블로, 열에서 열로 레코드를 복제합니다. 객체 이름은 중복되지 않도록 고유한 이름이어야 합니다.

예를 들어, 소스 테이블에는 이름이 ID인 열이 있고 그에 상응하는 대상 테이블에는 id라는 기존 열이 있습니다. 하나의 규칙에서 ADD-COLUMN 문을 사용하여 id라는 새 열을 추가하고 SQLite 문을 사용하여 열을 사용자 지정 값으로 채우는 경우, 이름이 id인 모호한 중복 객체가 생성되며 이 객체는 지원되지 않습니다.

- 변환 규칙을 생성할 때는 예를 들어 column-name를 %로 설정할 때 선택 규칙이 여러 열을 지정하는 경우에만 data-type 파라미터를 사용하는 것이 좋습니다. data-type을 사용하여 단일 열을 선택하는 것은 권장하지 않습니다.
- AWS DMS 는 소스 객체와 대상 객체(테이블)가 동일한 데이터베이스/스키마에 있는 변환 규칙을 지원하지 않습니다. 변환 규칙에서 소스 및 대상과 동일한 테이블을 사용하면 테이블 데이터의 의도하지 않은 변경, 테이블 구조 수정 또는 테이블 삭제를 포함하되 이에 국한되지 않는 예기치 않고 잠재적으로 유해한 결과가 발생할 수 있습니다.

값

변환 규칙 유형을 사용하는 테이블 매핑 규칙의 경우, 다음 값을 적용할 수 있습니다.

파라미터	가능한 값	설명
rule-type	transformation	선택 규칙에서 지정한 각 객체에 규칙을 적용하는 값입니다. 달리 지정되지 않는 한, transformation 을 사용합니다.
rule-id	숫자 값.	규칙을 식별하기 위한 고유한 숫자 값입니다. 동일한 객체(스키마, 테이블, 열, 테이블 간 공간 또는 인덱스 테이블 공간)에 대해 여러 변환 규칙을 지정하는 경우는 더 낮은 rule-id로 변환 규칙을 AWS DMS 적용합니다.
rule-name	영숫자 값입니다.	규칙을 식별하기 위한 고유한 이름입니다.
object-locator	객체는 다음 파라미터를 사용합니다. • schema-name – 스키마의 이름입니다. MongoDB 및 Amazon DocumentDB 엔드포인트의 경우, 컬렉션 세트를 보관하는 데이터베이스의 이름입니다. • table-name – 테이블, 뷰 또는 컬렉션의 이름입니다. • table-tablespace-name – 기존 테이블 테이블스페이스의 이름입니다. • index-tablespace-name – 기존 인덱스 테이블스페이스의 이름입니다. • column-name – 기존 열의 이름입니다. • data-type – 기존 열 데이터 형식의 이름입니다.	규칙이 적용되는 각 스키마, 테이블 또는 뷰, 테이블 테이블스페이스, 인덱스 테이블스페이스 및 열의 이름입니다. 각 object-locator 파라미터 (data-type 제외) 값의 전부 또는 일부에 '%' (퍼센트 기호)를 와일드카드로 사용할 수 있습니다. 따라서 다음 항목들을 일치시킬 수 있습니다. • 단일 스키마의 단일 테이블 또는 뷰 • 일부 또는 전체 스키마의 단일 테이블 또는 뷰 • 단일 스키마의 일부 또는 전체 테이블 및 뷰 • 일부 또는 전체 스키마의 일부 또는 전체 테이블 및 뷰 • (하나 또는 둘 이상의) 지정된 테이블, 뷰 및 스키마에 있는 하나 이상의 열.

파라미터	가능한 값	설명
		- 예를 들어 <code>column-name</code> 을 %로 설정할 때와 같이 선택 규칙이 여러 열을 지정하는 경우에만 <code>data-type</code> 파라미터를 사용하는 것이 좋습니다. 단일 열에는 이 파라미터를 사용하지 않는 것이 좋습니다. 또한 <code>table-tablespace-name</code> 또는 <code>index-tablespace-name</code> 파라미터는 Oracle 소스 엔드포인트와 일치시키는 데만 사용할 수 있습니다. 단일 규칙에서 <code>table-tablespace-name</code> 또는 <code>index-tablespace-name</code> 를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 따라서 다음 항목 중 하나와 일치시킬 수 있습니다. - 하나의, 일부 또는 모든 테이블 테이블스페이스 - 하나의, 일부 또는 모든 인덱스 테이블스페이스

파라미터	가능한 값	설명
rule-action	add-column , include-column , remove-column rename convert-lowercase , convert- uppercase add-prefix , remove-prefix , replace-prefix add-suffix , remove-suffix , replace-suffix define-primary-key change-data-type add-before-image-columns data-masking-digits-mask data-masking-digits- randomize data-masking-hash-mask	객체에 적용할 변환. 모든 변환 규칙 작업은 대소문자를 구분합니다. rule-action 파라미터의 add-column 값은 테이블에 열을 추가합니다. 하지만 같은 테이블의 기존 열과 이름이 같은 새 열을 추가할 수는 없습니다. expression 및 data-type 파라미터와 함께 사용할 경우 add-column 은 새 열 데이터의 값을 지정합니다. rule-action 에 대한 change-data-type 값은 column 규칙 대상에만 사용할 수 있습니다. rule-action 파라미터의 include-column 값은 기본적으로 모든 열을 삭제하고 지정된 열을 포함하도록 테이블 모드를 변경합니다. include-column 규칙을 여러 번 호출하면 여러 열이 대상에 포함됩니다. define-primary-key 규칙의 스키마 또는 테이블 이름에 와일드카드 (%)가 있으면 이 규칙을 사용할 수 없습니다. 기존 작업의 경우 remove-column , rename 또는 add-prefix 같은 대상 테이블 스키마를 변경하는 변환 규칙 작업은 작업을 다시 시작할 때까지 적용되지 않습니다. 변환 규칙을 추가

파라미터	가능한 값	설명
		한 후 작업을 재개하면 변경된 열에 예기치 않은 동작이 발생할 수 있으며, 여기에는 열 데이터 누락이 포함될 수 있습니다. 변환 규칙이 제대로 작동하려면 작업을 다시 시작해야 합니다. <code>data-masking-digits-mask</code>, <code>data-masking-digits-randomize</code> 및 <code>data-masking-hash-mask</code> 는 대상에 로드할 때 테이블의 하나 이상의 열에 포함된 민감한 정보를 마스킹하기 위한 것입니다. 이러한 변환은 열 규칙 대상에만 사용할 수 있습니다. 자세한 내용은 데이터 마스킹을 사용하여 민감한 정보 숨기기 섹션을 참조하세요.
<code>rule-target</code>	<code>schema</code> , <code>table</code> , <code>column</code> , <code>table-tablespace</code> , <code>index-tablespace</code>	변환할 객체의 유형입니다. <code>table-tablespace</code> 및 <code>index-tablespace</code> 값은 Oracle 대상 엔드 포인트에만 사용할 수 있습니다. <code>object-locator</code> : <code>table-tablespace-name</code> 또는 <code>index-tablespace-name</code> 이름의 일부로 지정하는 파라미터의 값을 지정해야 합니다.
<code>value</code>	대상 유형에서 명명 규칙을 따르는 영숫자 값입니다.	<code>rename</code> 과 같이 입력이 필요한 작업의 새 값입니다.
<code>old-value</code>	대상 유형에서 명명 규칙을 따르는 영숫자 값입니다.	<code>replace-prefix</code> 와 같이 대체가 필요한 작업의 이전 값입니다.

파라미터	가능한 값	설명
data-type	type – rule-action 이 add-column 인 경우 사용할 데이터 형식 또는 rule-action 이 change-data-type 인 경우 대체 데이터 형식입니다. 또는 rule-action 이 change-data-type 이고, column-name 의 값이 "%"이고, 기존 데이터 형식을 식별하기 위한 추가 data-type 파라미터가 object-locator 에 포함된 경우 대체 데이터 형식의 이름입니다. AWS DMS 는 다음 DMS 데이터 형식에 대한 열 데이터 형식 변환을 지원합니다. "bytes", "date", "time", "datetime", "int1", "int2", "int4", "int8", "numeric", "real4", "real8", "string", "uint1", "uint2", "uint4", "uint8", "wstring", "blob", "nclob", "clob", "boolean", "set", "list", "map", "tuple"  Note AWS DMS 는 지원되는 형식으로만 한 유형에서 다른 유형으로 변환을 적용할 수 있습니다. 예: DATE는 YYYY:MM:DD/YYYY-MM-DD 로 표시되어야 하며 YYYY:MM:DD HH:MM:SS/YYYY-MM-DD	다음은 대체할 기존 데이터 형식을 지정하는 data-type 파라미터의 예입니다. <pre data-bbox="987 415 1442 1717"> { "rules": [{ "rule-type": "selection", "rule-id": "1", "rule-name": "1", "object-locator": { "schema-name": "%", "table-name": "%" }, "rule-action": "include" }, { "rule-type": "transformation", "rule-id": "2", "rule-name": "2", "rule-target": "column", "object-locator": { "schema-name": "test", "table-name": "table_t" }, "column-name": "col10" }, { "rule-action": "change-data-type", "data-type": { "type": "string", "length": "4092", "scale": "" } }] } </pre>

파라미터	가능한 값	설명
	HH:MM:SS DATETIME은 로 표시되어야 합니다. TIME은 HH:MM:SS에 표시되어야 합니다. precision – 추가된 열 또는 대체 데이터 형식에 정밀도가 있는 경우 정밀도를 지정하는 정수 값. scale – 추가된 열 또는 대체 데이터 형식에 척도가 있는 경우 척도를 지정하는 정수 값 또는 날짜 시간 값. length – 새 열 데이터의 길이(add-column 과 함께 사용하는 경우).	여기서는 table_t 테이블의 col10 열이 string 데이터 유형으로 변경됩니다.

파라미터	가능한 값	설명
expression	SQLite 구문을 따르는 영숫자 값입니다.	rename-schema 로 설정된 rule-action 과 함께 사용되는 경우 expression 파라미터는 새 스키마를 지정합니다. rename-table 로 설정된 rule-action 과 함께 사용되는 경우 expression 은 새 테이블을 지정합니다. rename-column 으로 설정된 rule-action 과 함께 사용되는 경우 expression 은 새 열 이름을 지정합니다. add-column 으로 설정된 rule-action 과 함께 사용되는 경우 expression 은 새 열을 구성하는 데이터를 지정합니다. 이 파라미터에는 표현식만 지원되는 점에 유의하세요. 연산자와 명령은 지원되지 않습니다. 변환 규칙에 표현식을 사용하는 방법에 관한 자세한 내용은 변환 규칙 표현식을 사용하여 열 내용 정의 섹션을 참조하세요. SQLite 표현식에 관한 자세한 내용은 SQLite 함수를 사용하여 표현식 작성 섹션을 참조하세요.

파라미터	가능한 값	설명
primary-key-def	객체는 다음 파라미터를 사용합니다. - name – 테이블 또는 뷰에 대한 새 프라이머리 키 또는 고유 인덱스의 이름입니다. (선택 사항) origin – 정의할 고유 키의 유형: primary-key (기본값) 또는 unique-index . - columns – 프라이머리 키 또는 고유 인덱스에서 나타나는 순서대로 열의 이름을 나열하는 문자열 배열입니다.	이 파라미터는 변환된 테이블 또는 뷰에서 고유 키의 이름, 유형 및 콘텐츠를 정의할 수 있습니다. 이는 rule-action 이 define-primary-key 로 설정되고 rule-target 이 table로 설정된 경우에 해당됩니다. 기본적으로 고유 키는 프라이머리 키로 정의됩니다.

파라미터	가능한 값	설명
before-image-def	객체는 다음 파라미터를 사용합니다. - column-prefix – 열 이름 앞에 추가되는 값입니다. 기본값은 BI_입니다. - column-suffix – 열 이름에 뒤에 추가되는 값입니다. 기본값은 비어 있음입니다. - column-filter – 다음 값 중 하나가 필요합니다. pk-only(기본값), non-lob(선택 사항) 및 all(선택 사항).	이 파라미터는 이전 이미지 열을 식별하는 명명 규칙을 정의하고 대상에 이전 이미지 열을 만들 수 있는 소스 열을 식별하는 필터를 지정합니다. rule-action 이 add-before-image-columns 로 설정되고 rule-target 이 column으로 설정된 경우 이 파라미터를 지정할 수 있습니다. column-prefix 와 column-suffix 를 모두 빈 문자열로 설정하지 마세요. column-filter 에서 다음을 선택합니다. - pk-only – 테이블 프라이머리 키의 일부인 열만 추가합니다. - non-lob – LOB 유형이 아닌 열만 추가합니다. - all – 이전 이미지 값이 있는 열을 추가합니다.  Note before-image-def 파라미터는 CLOB 및 BLOB와 같은 대용량 이진 객체(LOB) 데이터 유형을 지원하지 않습니다. 데이터 유형이 LOB로 설정된 경우 테이블에 빈 열이 생성됩니다.

파라미터	가능한 값	설명
		AWS DMS 대상 엔드포인트에서 이전 이미지 지원에 관한 자세한 내용은 다음을 참조하세요. Kinesis 데이터 스트림 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기 Apache Kafka 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기

예제

Example스키마 이름 바꾸기

다음은 소스의 스키마 이름 Test를 대상의 Test1로 변경하는 예입니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-action": "rename",
      "rule-target": "schema",
      "object-locator": {
        "schema-name": "Test"
      }
    }
  ]
}
```

```

        "value": "Test1"
      }
    ]
  }

```

Example 테이블 이름 바꾸기

다음은 소스의 테이블 이름 Actor를 대상의 Actor1로 변경하는 예입니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-action": "rename",
      "rule-target": "table",
      "object-locator": {
        "schema-name": "Test",
        "table-name": "Actor"
      },
      "value": "Actor1"
    }
  ]
}

```

Example 열 이름 바꾸기

다음은 Actor 테이블에서 소스의 열 이름 first_name을 대상의 fname으로 변경하는 예입니다.

```

{
  "rules": [
    {

```

```

        "rule-type": "selection",
        "rule-id": "1",
        "rule-name": "1",
        "object-locator": {
            "schema-name": "test",
            "table-name": "%"
        },
        "rule-action": "include"
    },
    {
        "rule-type": "transformation",
        "rule-id": "4",
        "rule-name": "4",
        "rule-action": "rename",
        "rule-target": "column",
        "object-locator": {
            "schema-name": "test",
            "table-name": "Actor",
            "column-name" : "first_name"
        },
        "value": "fname"
    }
]
}

```

Example Oracle 테이블 테이블스페이스의 이름 바꾸기

다음은 Oracle 소스에서 Actor라는 테이블에 대해 SetSpace라는 테이블 테이블스페이스의 이름을 Oracle 대상 엔드포인트의 SceneTblSpace로 바꾸는 예입니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "Play",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {

```

```

        "rule-type": "transformation",
        "rule-id": "2",
        "rule-name": "2",
        "rule-action": "rename",
        "rule-target": "table-tablespace",
        "object-locator": {
            "schema-name": "Play",
            "table-name": "Actor",
            "table-tablespace-name": "SetSpace"
        },
        "value": "SceneTblSpace"
    }
]
}

```

Example Oracle 인덱스 테이블스페이스의 이름 바꾸기

다음은 Oracle 소스에서 Actor라는 테이블에 대해 SetISpace라는 인덱스 테이블스페이스의 이름을 Oracle 대상 엔드포인트의 SceneIdxSpace로 바꾸는 예입니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "Play",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-action": "rename",
      "rule-target": "table-tablespace",
      "object-locator": {
        "schema-name": "Play",
        "table-name": "Actor",
        "table-tablespace-name": "SetISpace"
      },
    },
  ],
}

```

```

        "value": "SceneIdxSpace"
      }
    ]
  }

```

Example 열 추가

다음 예에서는 스키마 test의 테이블 Actor에 datetime 열을 추가합니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "test",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-action": "add-column",
      "rule-target": "column",
      "object-locator": {
        "schema-name": "test",
        "table-name": "actor"
      },
      "value": "last_updated",
      "data-type": {
        "type": "datetime",
        "precision": 6
      }
    }
  ]
}

```

Example 열 제거

다음은 소스의 Actor라는 테이블을 변환하여 col 문자로 시작하는 모든 열을 대상에서 제거하는 예입니다.

```
{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "%"
    },
    "rule-action": "include"
  }, {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-action": "remove-column",
    "rule-target": "column",
    "object-locator": {
      "schema-name": "test",
      "table-name": "Actor",
      "column-name": "col%"
    }
  }
]
```

Example 소문자로 변환

다음은 소스의 테이블 이름 ACTOR를 대상의 actor로 변환하는 예입니다.

```
{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "%"
    },
    "rule-action": "include"
  }, {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-action": "rename",
    "rule-target": "table",
    "object-locator": {
      "schema-name": "test",
      "table-name": "ACTOR",
      "column-name": ""
    }
  }
]
```

```

}, {
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "convert-lowercase",
  "rule-target": "table",
  "object-locator": {
    "schema-name": "test",
    "table-name": "ACTOR"
  }
}]
}

```

Example 대문자로 변환

다음은 모든 테이블의 전체 열과 모든 스키마를 소스의 소문자에서 대상의 대문자로 변환하는 예제입니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "test",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-action": "convert-uppercase",
      "rule-target": "column",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%",
        "column-name": "%"
      }
    }
  ]
}

```

```
}
```

Example접두사 추가

다음은 소스의 모든 테이블을 변환하여 대상에서 테이블에 DMS_ 접두사를 추가하는 예입니다.

```
{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "%"
    },
    "rule-action": "include"
  }, {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-action": "add-prefix",
    "rule-target": "table",
    "object-locator": {
      "schema-name": "test",
      "table-name": "%"
    },
    "value": "DMS_"
  }]
}
```

Example접두사 바꾸기

다음은 소스에서 Pre_ 접두사를 포함하는 모든 열을 변환하여 이 접두사를 대상의 NewPre_ 접두사로 대체하는 예입니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
```



```

        "object-locator": {
            "schema-name": "test",
            "table-name": "%"
        },
        "rule-action": "include"
    },
    {
        "rule-type": "transformation",
        "rule-id": "2",
        "rule-name": "2",
        "rule-action": "replace-prefix",
        "rule-target": "column",
        "object-locator": {
            "schema-name": "%",
            "table-name": "%",
            "column-name": "%"
        },
        "value": "NewPre_",
        "old-value": "Pre_"
    }
]
}

```

Example 접미사 제거

다음은 소스의 모든 테이블을 변환하여 대상의 테이블에서 _DMS 접미사를 제거하는 예입니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "%"
    },
    "rule-action": "include"
  }, {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-action": "remove-suffix",
    "rule-target": "table",
    "object-locator": {

```

```

    "schema-name": "test",
    "table-name": "%"
  },
  "value": "_DMS"
}]
}

```

Example 프라이머리 키 정의

대상 엔드포인트로 마이그레이션된 ITEM-primary-key 테이블의 세 열에서 ITEM라는 프라이머리 키를 정의하는 예입니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "inventory",
      "table-name": "%"
    },
    "rule-action": "include"
  }, {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-action": "define-primary-key",
    "rule-target": "table",
    "object-locator": {
      "schema-name": "inventory",
      "table-name": "ITEM"
    },
    "primary-key-def": {
      "name": "ITEM-primary-key",
      "columns": [
        "ITEM-NAME",
        "BOM-MODEL-NUM",
        "BOM-PART-NUM"
      ]
    }
  }
]
}

```

Example고유 인덱스 정의

대상 엔드포인트로 마이그레이션된 ITEM 테이블의 세 열에서 ITEM-unique-idx라는 고유 인덱스를 정의하는 예입니다.

```
{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "inventory",
      "table-name": "%"
    },
    "rule-action": "include"
  }, {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-action": "define-primary-key",
    "rule-target": "table",
    "object-locator": {
      "schema-name": "inventory",
      "table-name": "ITEM"
    },
    "primary-key-def": {
      "name": "ITEM-unique-idx",
      "origin": "unique-index",
      "columns": [
        "ITEM-NAME",
        "BOM-MODEL-NUM",
        "BOM-PART-NUM"
      ]
    }
  }
]
```

Example대상 열의 데이터 형식 변경

다음 예제에서는 이름이 SALE_AMOUNT인 대상 열의 데이터 형식을 기존 데이터 형식에서 int8로 변경합니다.

```
{
```

```

    "rule-type": "transformation",
    "rule-id": "1",
    "rule-name": "RuleName 1",
    "rule-action": "change-data-type",
    "rule-target": "column",
    "object-locator": {
        "schema-name": "dbo",
        "table-name": "dms",
        "column-name": "SALE_AMOUNT"
    },
    "data-type": {
        "type": "int8"
    }
}

```

Example이전 이미지 열 추가

emp_no라는 소스 열에 대해 다음 예의 변환 규칙은 대상에 BI_emp_no라는 새 열을 추가합니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-target": "column",
    "object-locator": {
      "schema-name": "%",
      "table-name": "employees"
    },
    "rule-action": "add-before-image-columns",
    "before-image-def": {
      "column-prefix": "BI_",
      "column-suffix": "",
      "column-filter": "pk-only"
    }
  }
]
}

```

```

    }
  }
]
}

```

여기서 다음 문은 해당 행의 BI_emp_no 열을 1로 채웁니다.

```
UPDATE employees SET emp_no = 3 WHERE BI_emp_no = 1;
```

지원되는 AWS DMS 대상에 CDC 업데이트를 작성할 때 BI_emp_no 열을 사용하면 emp_no 열에 업데이트된 값이 있는 행을 알 수 있습니다.

변환 규칙 표현식을 사용하여 열 내용 정의

변환 규칙 내에서 표현식을 사용하여 새 열과 기존 열의 내용을 정의할 수 있습니다. 예를 들어, 표현식을 사용하여 열을 추가하거나 소스 테이블 헤더를 대상에 복제할 수 있습니다. 표현식을 사용하여 대상 테이블의 레코드를 소스에서 삽입, 업데이트 또는 삭제된 것으로 플래그를 지정할 수도 있습니다.

주제

- [표현식을 사용하여 열 추가](#)
- [표현식을 사용하여 대상 레코드에 플래그 지정](#)
- [표현식을 사용하여 소스 테이블 헤더 복제](#)
- [SQLite 함수를 사용하여 표현식 작성](#)
- [표현식을 사용하여 대상 테이블에 메타데이터 추가](#)

표현식을 사용하여 열 추가

변환 규칙에서 표현식을 사용하여 테이블에 열을 추가하려면 add-column 규칙 작업 및 column 규칙 대상을 사용합니다.

다음 예에서는 ITEM 테이블에 새 열을 추가합니다. 50자 길이의 string 데이터 형식을 사용하여 새 열 이름을 FULL_NAME으로 설정합니다. 표현식은 기존의 두 열 FIRST_NAME 및 LAST_NAME의 값을 결합하여 FULL_NAME으로 평가합니다. schema-name, table-name 및 표현식 파라미터는 소스 데이터베이스 테이블의 객체를 참조합니다. Value 및 data-type블록은 대상 데이터베이스 테이블의 객체를 참조합니다.

```
{
```

```

"rules": [
  {
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "Test",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "rule-action": "add-column",
    "rule-target": "column",
    "object-locator": {
      "schema-name": "Test",
      "table-name": "ITEM"
    },
    "value": "FULL_NAME",
    "expression": "$FIRST_NAME||'_'||$LAST_NAME",
    "data-type": {
      "type": "string",
      "length": 50
    }
  }
]
}

```

표현식을 사용하여 대상 레코드에 플래그 지정

변환 규칙에서 표현식을 사용하여 대상 테이블의 레코드를 소스 테이블에서 삽입, 업데이트 또는 삭제된 것으로 플래그를 지정할 수 있습니다. 표현식은 `operation_indicator` 함수를 사용하여 레코드에 플래그를 지정합니다. 소스에서 삭제된 레코드는 대상에서 삭제되지 않습니다. 그 대신, 대상 레코드가 소스에서 삭제되었음을 나타내는 사용자 제공 값으로 플래그가 지정됩니다.

Note

이 `operation_indicator` 함수는 원본 및 대상 데이터베이스에 프라이머리 키가 있는 테이블에서만 작동합니다.

예를 들어, 다음 변환 규칙은 먼저 대상 테이블에 새 Operation 열을 추가합니다. 그런 다음, 레코드가 소스 테이블에서 삭제될 때마다 D 값으로 열을 업데이트합니다.

```
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "%",
    "table-name": "%"
  },
  "rule-action": "add-column",
  "value": "Operation",
  "expression": "operation_indicator('D', 'U', 'I')",
  "data-type": {
    "type": "string",
    "length": 50
  }
}
```

표현식을 사용하여 소스 테이블 헤더 복제

기본적으로 소스 테이블의 헤더는 대상에 복제되지 않습니다. 복제할 헤더를 나타내려면 테이블 열 헤더를 포함하는 표현식과 함께 변환 규칙을 사용합니다.

표현식에서 다음 열 헤더를 사용할 수 있습니다.

헤더	지속적 복제의 값	전체 로드의 값	데이터 유형
AR_H_STRE AM_POSITION	소스의 스트림 위치 값. 이 값은 소스 엔드 포인트에 따라 시스템 변경 번호(SCN) 또는 로그 시퀀스 번호(LSN)일 수 있습니다.	빈 문자열.	STRING
AR_H_TIMESTAMP	변경 시간을 나타내는 타임스탬프.	데이터가 대상에 도달한 현재 시간을 나타내는 타임스탬프입니다.	DATETIME(크기 = 7)

헤더	지속적 복제의 값	전체 로드의 값	데이터 유형
AR_H_COMM IT_TIMESTAMP	커밋 시간을 나타내는 타임 스탬프.	현재 시간을 나타내는 타임스탬프.	DATETIME(크기 = 7)
AR_H_OPERATION	INSERT, UPDATE 또 는 DELETE	INSERT	STRING
AR_H_USER	변경한 사용자에게 대해 소스에서 제공되는 사 용자 이름, ID 또는 기 타 정보. 이 헤더는 SQL Server 및 Oracle(버전 11.2.0.3 이상) 소스 엔 드포인트에서만 지원 됩니다.	빈 문자열.	STRING
AR_H_CHAN GE_SEQ	타임스탬프와 자동 증 분 숫자로 구성된 소 스 데이터베이스의 고 유한 증분 숫자입니다. 값은 소스 데이터베이 스 시스템에 따라 달라 집니다.	빈 문자열.	STRING

다음 예에서는 소스의 스트림 위치 값을 사용하여 대상에 새 열을 추가합니다. SQL Server의 경우, 스트림 위치 값은 소스 엔드포인트의 LSN입니다. Oracle의 경우, 스트림 위치 값은 소스 엔드포인트의 SCN입니다.

```
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "%",
```



```

    "table-name": "%"
  },
  "rule-action": "add-column",
  "value": "transact_id",
  "expression": "$AR_H_STREAM_POSITION",
  "data-type": {
    "type": "string",
    "length": 50
  }
}

```

다음 예제에서는 원본의 고유한 증분 번호가 있는 새 열을 대상에 추가합니다. 이 값은 작업 수준에서 35자리 고유 숫자를 나타냅니다. 처음 16자리는 타임스탬프의 일부이고, 마지막 19자리는 DBMS에서 증분한 record_id 번호입니다.

```

{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "%",
    "table-name": "%"
  },
  "rule-action": "add-column",
  "value": "transact_id",
  "expression": "$AR_H_CHANGE_SEQ",
  "data-type": {
    "type": "string",
    "length": 50
  }
}

```

SQLite 함수를 사용하여 표현식 작성

테이블 설정을 사용하여 지정된 작업에서 선택한 테이블 또는 뷰에 적용할 설정을 지정합니다. 테이블 설정 규칙은 선택 사항입니다.

Note

MongoDB 및 DocumentDB 데이터베이스는 테이블과 뷰의 개념 대신 데이터 레코드를 컬렉션에 함께 수집된 문서로 저장합니다. 따라서 MongoDB 또는 DocumentDB 소스에서 마이그레이션할 때는 테이블과 뷰보다는 선택한 컬렉션에 대한 병렬 로드 설정의 범위 세그먼트화 유형을 고려하세요.

주제

- [CASE 표현식 사용](#)
- [예제:](#)

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 문자열 함수를 찾을 수 있습니다.

문자열 함수	설명
<code>lower(x)</code>	이 <code>lower(x)</code> 함수는 모든 문자가 소문자로 변환된 <code>x</code> 문자열의 복사본을 반환합니다. 기본 내장 <code>lower</code> 함수는 ASCII 문자에만 사용할 수 있습니다.
<code>upper(x)</code>	이 <code>upper(x)</code> 함수는 모든 문자가 소문자로 변환된 <code>x</code> 문자열의 복사본을 반환합니다. 기본 내장 <code>upper</code> 함수는 ASCII 문자에만 사용할 수 있습니다.
<code>ltrim(x,y)</code>	이 <code>ltrim(x,y)</code> 함수는 <code>x</code> 의 왼쪽에서부터 <code>y</code> 에 표시되는 모든 문자를 제거하여 구성된 문자열을 반환합니다. <code>y</code> 에 값이 없는 경우, <code>ltrim(x)</code> 는 <code>x</code> 의 왼쪽에서 공백을 제거합니다.
<code>replace(x,y,z)</code>	이 <code>replace(x,y,z)</code> 함수는 문자열 <code>x</code> 에서 나타나는 모든 문자열 <code>y</code> 를 문자열 <code>z</code> 로 대체하여 구성된 문자열을 반환합니다.
<code>rtrim(x,y)</code>	이 <code>rtrim(x,y)</code> 함수는 <code>x</code> 의 오른쪽에서부터 <code>y</code> 에 표시되는 모든 문자를 제거하여 구성된 문자열을 반환합니다. <code>y</code> 에 값이 없는 경우, <code>rtrim(x)</code> 는 <code>x</code> 의 오른쪽에서 공백을 제거합니다.

문자열 함수	설명
<code>substr(<i>x</i>,<i>y</i>,<i>z</i>)</code>	이 <code>substr(<i>x</i>,<i>y</i>,<i>z</i>)</code> 함수는 입력 문자열의 첫 <i>y</i>번째 문자로 시작하는 하위 문자열 <i>x</i>를 반환합니다. 이 문자열의 길이는 <i>z</i>입니다. <i>z</i>를 생략하면 <code>substr(<i>x</i>,<i>y</i>)</code>은 <i>y</i>번째 문자로 시작하는 <i>x</i> 문자열의 끝까지 모든 문자를 반환합니다. <i>x</i>의 가장 왼쪽 문자는 숫자 1입니다. <i>y</i>가 음수인 경우, 왼쪽이 아닌 오른쪽부터 계수하여 하위 문자열의 첫 번째 문자를 찾습니다. <i>z</i>가 음수인 경우, 첫 <i>y</i> 번째 문자 앞의 <code>abs(<i>z</i>)</code> 문자가 반환됩니다. <i>x</i>가 문자열인 경우, 문자의 인덱스는 실제 UTF-8 문자를 참조합니다. <i>x</i>가 BLOB인 경우, 인덱스는 바이트를 나타냅니다.
<code>trim(<i>x</i>,<i>y</i>)</code>	이 <code>trim(<i>x</i>,<i>y</i>)</code> 함수는 <i>x</i>의 양쪽에서부터 <i>y</i>에 표시되는 모든 문자를 제거하여 구성된 문자열을 반환합니다. <i>y</i>에 값이 없는 경우, <code>trim(<i>x</i>)</code>는 <i>x</i>의 오른쪽에서 공백을 제거합니다.

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 LOB 함수를 찾을 수 있습니다.

LOB 함수	설명
<code>hex(<i>x</i>)</code>	이 <code>hex</code> 함수는 BLOB를 인수로 받고 BLOB 내용의 대문자 16진수 문자열 버전을 반환합니다.
<code>randblob (<i>N</i>)</code>	이 <code>randblob(<i>N</i>)</code> 함수는 의사 랜덤 바이트를 포함하는 <i>N</i> 바이트 BLOB를 반환합니다. <i>N</i> 이 1보다 작으면 1바이트 임의의 BLOB가 반환됩니다.
<code>zeroblob(<i>N</i>)</code>	이 <code>zeroblob(<i>N</i>)</code> 함수는 <i>N</i> 바이트의 0x00으로 구성된 BLOB를 반환합니다.

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 numeric 함수를 찾을 수 있습니다.

Numeric 함수	설명
abs(<i>x</i>)	이 abs(<i>x</i>) 함수는 숫자 인수 <i>x</i> 의 절대값을 반환합니다. <i>x</i> 가 NULL인 경우 abs(<i>x</i>) 함수는 NULL을 반환합니다. <i>x</i> 가 숫자 값으로 변환할 수 없는 문자열 또는 BLOB인 경우 abs(<i>x</i>) 함수는 0.0을 반환합니다.
random()	이 random 함수는 -9,223,372,036,854,775,808 ~ +9,223,372,036,854,775,807 사이의 의사 랜덤 정수를 반환합니다.
round (<i>x</i> , <i>y</i>)	이 round (<i>x</i> , <i>y</i>) 함수는 소수점 오른쪽의 <i>y</i> 자리 숫자로 반올림된 부동 소수점 값 <i>x</i> 를 반환합니다. <i>y</i> 에 값이 없으면 0으로 간주됩니다.
max (<i>x</i> , <i>y</i> ...)	다중 인수 max 함수는 최대값이 포함된 인수를 반환하며 인수가 NULL이면 NULL을 반환합니다. max 함수는 인수의 왼쪽에서 오른쪽으로 데이터 정렬 함수를 정의하는 인수를 검색합니다. 검색으로 찾은 인수는 모든 문자열 비교에 해당 데이터 정렬 함수를 사용합니다. 데이터 정렬 함수를 정의하는 인수가 max에 하나도 없다면 BINARY 데이터 정렬 함수가 사용됩니다. max 함수는 인수가 두 개 이상이면 단순 함수이지만 인수가 하나뿐이면 집계 함수로 작동합니다.
min (<i>x</i> , <i>y</i> ...)	다중 인수 min 함수는 최소값이 포함된 인수를 반환합니다. min 함수는 인수의 왼쪽에서 오른쪽으로 데이터 정렬 함수를 정의하는 인수를 검색합니다. 검색으로 찾은 인수는 모든 문자열 비교에 해당 데이터 정렬 함수를 사용합니다. 데이터 정렬 함수를 정의하는 인수가 min에 하나도 없다면 BINARY 데이터 정렬 함수가 사용됩니다. min 함수는 인수가 두 개 이상이면 단순 함수이지만 인수가 하나뿐이면 집계 함수로 작동합니다.

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 NULL 체크 함수를 찾을 수 있습니다.

NULL 체크 함수	설명
<code>coalesce (x,y...)</code>	이 <code>coalesce</code> 함수는 NULL이 아닌 첫 번째 인수의 복사본을 반환하지만 모든 인수가 NULL이면 NULL을 반환합니다. <code>coalesce</code> 함수에는 인수가 두 개 이상 있습니다.
<code>ifnull(x,y)</code>	이 <code>ifnull</code> 함수는 NULL이 아닌 첫 번째 인수의 복사본을 반환하지만 두 인수가 모두 NULL이면 NULL을 반환합니다. 이 <code>ifnull</code> 함수에는 정확히 두 개의 인수가 있습니다. <code>ifnull</code> 함수는 두 개의 인수가 있는 <code>coalesce</code> 와 같은 함수입니다.
<code>nullif(x,y)</code>	<code>nullif(x,y)</code> 함수는 인수가 다르면 첫 번째 인수의 복사본을 반환하지만 인수가 같으면 NULL을 반환합니다. <code>nullif(x,y)</code> 함수는 인수의 왼쪽에서 오른쪽으로 데이터 정렬 함수를 정의하는 인수를 검색합니다. 검색으로 찾은 인수는 모든 문자열 비교에 해당 데이터 정렬 함수를 사용합니다. 데이터 정렬 함수를 정의하는 인수가 <code>nullif</code>에 하나도 없다면 BINARY 데이터 정렬 함수가 사용됩니다.

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 날짜 및 시간 함수를 찾을 수 있습니다.

날짜 및 시간 함수	설명
<code>date(timestring , modifier, modifier...)</code>	이 <code>date</code> 함수는 YYYY-MM-DD 형식으로 날짜를 반환합니다.
<code>time(timestring , modifier, modifier...)</code>	이 <code>time</code> 함수는 HH:MM:SS 형식으로 시간을 반환합니다.
<code>datetime(timestring , modifier, modifier...)</code>	이 <code>datetime</code> 함수는 YYYY-MM-DD HH:MM:SS 형식으로 날짜 및 시간을 반환합니다.
<code>julianday(timestring , modifier, modifier...)</code>	이 <code>julianday</code> 함수는 기원전 4714년 11월 24일 그리니치에서 정오 이후 경과한 일수를 반환합니다.

날짜 및 시간 함수	설명
<code>strftime(<i>format</i>, <i>timestring</i> , <i>modifier</i>, <i>modifier</i>...)</code>	이 <code>strftime</code> 함수는 다음 변수 중 하나를 사용하여 첫 번째 인수로 지정된 형식 문자열에 따라 날짜를 반환합니다. <code>%d</code>: 날짜(월, 일) <code>%H</code>: 00~24시 <code>%f</code>: ** 소수 초(SS.SSS) <code>%j</code>: 연중 일(001~366) <code>%J</code>: ** 율리우스력 날수 <code>%m</code>: 01~12개월 차 <code>%M</code>: 00~59분 <code>%s</code>: 1970-01-01 이후 경과 시간(초) <code>%S</code>: 00~59초 <code>%w</code>: 주중 일(0~6, 일요일 =0) <code>%W</code>: 연중 주(00~53) <code>%Y</code>: 0000~9999년도 <code>%%</code>: %

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 해시 함수를 찾을 수 있습니다.

해시 함수	설명
<code>hash_sha256(<i>x</i>)</code>	이 <code>hash</code> 함수는 (SHA-256 알고리즘을 사용하여) 입력 열에 대한 해시 값을 생성하고 생성된 해시 값의 16진수 값을 반환합니다. 표현식에서 <code>hash</code> 함수를 사용하려면 표현식에 <code>hash_sha256(<i>x</i>)</code> 함수를 추가하고 <i>x</i>를 소스 열 이름으로 바꾸세요.

CASE 표현식 사용

SQLite CASE 표현식은 조건 목록을 평가하고 결과를 기반으로 표현식을 반환합니다. 다음 구문이 나와 있습니다.

```
CASE case_expression
  WHEN when_expression_1 THEN result_1
  WHEN when_expression_2 THEN result_2
  ...
  [ ELSE result_else ]
END
```

Or

```
CASE
  WHEN case_expression THEN result_1
  WHEN case_expression THEN result_2
  ...
  [ ELSE result_else ]
END
```

예제:

Example CASE 조건을 사용하여 대상 테이블에 새 문자열 열을 추가하는 방법

예를 들어, 다음 변환 규칙은 먼저 대상 테이블 employee에 새 문자열 열 emp_seniority를 추가합니다. 급여 열에 SQLite round 함수를 사용하고 이 때 급여가 20,000과 같거나 이를 초과하는지 확인하는 CASE 조건을 함께 적용합니다. 값이 맞으면 해당 열에 값 SENIOR이 적용되고 다른 모든 항목은 해당 값 JUNIOR를 갖습니다.

```
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "add-column",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "public",
    "table-name": "employee"
  },
  "value": "emp_seniority",
  "expression": " CASE WHEN round($emp_salary)>=20000 THEN 'SENIOR' ELSE 'JUNIOR'
END",
```

```

    "data-type": {
      "type": "string",
      "length": 50
    }
  }
}

```

Example SUBSTR 함수를 사용하여 대상 테이블에 새 문자열 열을 추가하는 방법

다음 예제 변환 규칙은 SQLite 연산자 또는 함수를 사용하여 열에 데이터를 정의하는 새 문자열 열을 추가합니다. 이 접근 방식에는 SQLite 함수를 사용하여 Postgresql 대상 테이블에 삽입하기 전에 Oracle에서 UUID 형식으로 로드된 GUID 데이터를 변환하는 작업이 포함됩니다.

다음 규칙은 SQLite 하위 문자열(SUBSTR), 16진수 함수(HEX) 및 소문자(LOWER) 함수를 사용하여 GUID 데이터를 하이픈으로 구분된 여러 그룹, 특히 8자리 그룹과 4자리 그룹 3개, 12자리 그룹 12자리 그룹으로 나눕니다. 총 32자리는 128비트를 나타냅니다.

다음은 변환 규칙을 통한 대상 사후 처리에 대한 샘플 소스 데이터 및 출력입니다.

소스 테이블(Oracle GUID 형식)

T_COL2

```

06F6949D234911EE80670242AC120002
1A2B3C4D5E6F11EE80670242AC120003
F5E4D3C2B1A011EE80670242AC120004

```

대상 테이블(PostgreSQL UUID 형식)

T_COL2_TMP

```

06f6949d-2349-11ee-8067-0242ac120002
1a2b3c4d-5e6f-11ee-8067-0242ac120003
f5e4d3c2-b1a0-11ee-8067-0242ac120004

```

```

{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "add-column",

```



```

"rule-target": "column",
"object-locator": {
  "schema-name": "SPORTS",
  "table-name": "TEST_TBL_2"
},
"value": "t_col2_tmp",
"expression": "CASE LOWER(SUBSTR(HEX($T_COL2), 1, 8) || '-' ||
SUBSTR(HEX($T_COL2), 9, 4) || '-' || SUBSTR(HEX($T_COL2), 13, 4) || '-' ||
SUBSTR(HEX($T_COL2), 17, 4) || '-' || SUBSTR(HEX($T_COL2), 21, 12)) WHEN '----'
THEN NULL ELSE LOWER(SUBSTR(HEX($T_COL2), 1, 8) || '-' || SUBSTR(HEX($T_COL2), 9,
4) || '-' || SUBSTR(HEX($T_COL2), 13, 4) || '-' || SUBSTR(HEX($T_COL2), 17, 4) ||
'-' || SUBSTR(HEX($T_COL2), 21, 12)) END",
"data-type": {
  "type": "string",
  "length": 60
}
}

```

Example 대상 테이블에 새 날짜 열 추가

다음 예에서는 대상 테이블 employee에 새 날짜 열 createdate를 추가합니다. SQLite 날짜 함수 datetime을 사용하면 삽입된 각 행에 대해 새로 생성된 테이블에 날짜가 추가됩니다.

```

{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "add-column",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "public",
    "table-name": "employee"
  },
  "value": "createdate",
  "expression": "datetime ()",
  "data-type": {
    "type": "datetime",
    "precision": 6
  }
}

```

Example대상 테이블에 새 숫자 열 추가

다음 예에서는 대상 테이블 employee에 새 숫자 열 rounded_emp_salary를 추가합니다. SQLite round 함수를 사용하여 반올림된 급여를 추가합니다.

```
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "add-column",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "public",
    "table-name": "employee"
  },
  "value": "rounded_emp_salary",
  "expression": "round($emp_salary)",
  "data-type": {
    "type": "int8"
  }
}
```

Example해시 함수를 사용하여 대상 테이블에 새 문자열 열을 추가하는 방법

다음 예에서는 대상 테이블 employee에 새 문자열 열 hashed_emp_number를 추가합니다. SQLite hash_sha256(x) 함수는 소스 열 emp_number의 대상에 해시된 값을 생성합니다.

```
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "add-column",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "public",
    "table-name": "employee"
  },
  "value": "hashed_emp_number",
  "expression": "hash_sha256($emp_number)",
  "data-type": {
    "type": "string",
    "length": 64
  }
}
```

```
}
}
```

표현식을 사용하여 대상 테이블에 메타데이터 추가

다음 표현식을 사용하여 대상 테이블에 메타데이터 정보를 추가할 수 있습니다.

- \$AR_M_SOURCE_SCHEMA – 소스 스키마의 이름입니다.
- \$AR_M_SOURCE_TABLE_NAME – 소스 테이블의 이름입니다.
- \$AR_M_SOURCE_COLUMN_NAME – 소스 테이블의 열 이름입니다.
- \$AR_M_SOURCE_COLUMN_DATATYPE – 소스 테이블 내 열의 데이터 형식입니다.

Example 소스의 스키마 이름을 사용하여 스키마 이름에 대한 열을 추가하는 방법

다음 예에서는 소스의 스키마 이름을 사용하여 대상에 schema_name이라는 새 열을 추가합니다.

```
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "add-column",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "%",
    "table-name": "%"
  },
  "rule-action": "add-column",
  "value": "schema_name",
  "expression": "$AR_M_SOURCE_SCHEMA",
  "data-type": {
    "type": "string",
    "length": 50
  }
}
```

테이블 및 컬렉션 설정 규칙과 작업

테이블 설정을 사용하여 지정된 작업에서 선택한 테이블 또는 뷰에 적용할 설정을 지정합니다. 테이블 설정 규칙은 엔드포인트 및 마이그레이션 요구 사항에 따라 선택 사항입니다.

MongoDB 및 Amazon DocumentDB 데이터베이스는 테이블과 뷰의 개념 대신 데이터 레코드를 컬렉션에 함께 수집된 문서로 저장합니다. MongoDB 또는 Amazon DocumentDB 엔드포인트에 대한 단일 데이터베이스는 데이터베이스 이름으로 식별되는 특정 컬렉션 집합입니다.

MongoDB 또는 Amazon DocumentDB 소스에서 마이그레이션할 때는 병렬 로드 설정을 사용하는 방식이 약간 다릅니다. 이 경우, 테이블과 뷰보다는 선택한 컬렉션에 대한 자동 세분화 또는 범위 세그먼트화 유형의 병렬 로드 설정을 고려해 보세요.

주제

- [테이블 설정의 와일드카드는 제한됨](#)
- [선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용](#)
- [선택한 테이블 또는 뷰의 LOB 설정 지정](#)
- [테이블 설정 예제](#)

테이블 설정 규칙 유형을 사용하는 테이블 매핑 규칙의 경우, 다음 파라미터를 적용할 수 있습니다.

파라미터	가능한 값	설명
rule-type	table-settings	선택 규칙에서 지정한 테이블, 뷰 또는 컬렉션에 규칙을 적용하는 값입니다.
rule-id	숫자 값.	규칙을 식별하기 위한 고유한 숫자 값입니다.
rule-name	영숫자 값입니다.	규칙을 식별하기 위한 고유한 이름입니다.
object-locator	객체는 다음 파라미터를 사용합니다. • schema-name – 스키마의 이름입니다. MongoDB 및 Amazon DocumentDB 엔드포인트의 경우, 컬렉션 세트를 보관하는 데이터베이스의 이름입니다. • table-name – 테이블, 뷰 또는 컬렉션의 이름입니다.	특정 스키마와 테이블 또는 뷰의 이름 또는 특정 데이터베이스 및 컬렉션의 이름(와일드카드 없음)입니다.

파라미터	가능한 값	설명
parallel-load	객체는 다음 파라미터를 사용합니다. - type – 병렬 로드를 활성화할지 여부를 지정합니다. 사용할 경우 이 파라미터는 동시에 로드할 테이블 또는 뷰 파티션, 하위 파티션 또는 기타 세그먼트를 식별하는 메커니즘을 지정합니다. 파티션은 소스 테이블 또는 뷰에서 이미 이름별로 정의되고 식별된 세그먼트입니다. MongoDB 및 Amazon DocumentDB 엔드포인트의 경우 파티션은 세그먼트입니다. 연결된 자동 분할 파라미터를 자동으로 계산할 AWS DMS 수 있습니다. 또는 범위 세그먼트화 파라미터를 사용하여 이러한 파라미터를 수동으로 지정할 수도 있습니다. Oracle 엔드포인트의 경우에 한해 하위 파티션은 이미 소스 테이블 또는 뷰에서 이름별로 정의되고 식별된 추가 수준의 세그먼트입니다. 하나 이상의 테이블 또는 뷰 열에 대한 값 범위의 경계를 지정하여 table-settings 규칙에서 다른 세그먼트를 식별할 수 있습니다. - partitions – type이 partitions-list 일 때 이 값은 동시에 로드할 모든 파티션을 지정합니다.	object-locator 옵션으로 식별되는 테이블 또는 뷰에서 병렬 로드(다중 스레드) 작업을 지정하는 값입니다. 이 경우, 다음과 같은 방법으로 동시에 로드할 수 있습니다. - 사용 가능한 모든 파티션이나 하위 파티션으로 지정된 세그먼트별로 로드합니다. - 선택한 파티션 및 하위 파티션별로 로드합니다. - 자동 세분화로 로드하거나 지정한 범위 기반 세그먼트별로 로드합니다. 병렬 로드와 관련한 자세한 내용은 선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용 섹션을 참조하세요.

파라미터	가능한 값	설명
	• <code>subpartitions</code> – Oracle 엔드포인트의 경우에 한해 <code>type이 partitions-list</code> 일 때 이 값은 동시에 로드할 모든 하위 파티션을 지정합니다. • <code>columns – type이 ranges</code> 일 때 이 값은 동시에 로드할 범위 기반 세그먼트를 식별하는 데 사용되는 열의 이름을 지정합니다. • <code>boundaries – type이 ranges</code> 일 때 이 값은 동시에 로드할 범위 기반 세그먼트를 식별하는 데 사용되는 <code>columns</code>의 값을 지정합니다.	

파라미터	가능한 값	설명
type	parallel-load 에 다음 중 하나를 수행하세요. • <code>partitions-auto</code> – 테이블 또는 뷰의 모든 파티션이 동시에 로드됩니다. 모든 파티션은 자체 스레드에 할당됩니다. 이는 MongoDB 및 Amazon DocumentDB 소스 엔드포인트가 병렬 전체 로드의 자동 세분화 옵션을 사용하기 위한 필수 설정입니다. • <code>subpartitions-auto</code> – (Oracle 엔드포인트 전용) 테이블 또는 뷰의 모든 하위 파티션이 동시에 로드됩니다. 모든 하위 파티션은 자체 스레드에 할당됩니다. • <code>partitions-list</code> – 테이블 또는 뷰에서 지정된 모든 파티션이 동시에 로드됩니다. Oracle 엔드포인트의 경우에 한해 테이블 또는 뷰에서 지정된 모든 하위 파티션이 동시에 로드됩니다. 지정한 각 파티션 및 하위 파티션은 자체 스레드에 할당됩니다. 파티션 이름 (<code>partitions</code>) 및 하위 파티션 이름(<code>subpartitions</code>)에 따라 동시에 로드할 파티션과 하위 파티션을 지정합니다. • <code>ranges</code> – 테이블, 뷰 또는 컬렉션에서 범위가 지정된 모든 세그먼트가 동시에 로드됩니다. 식별하는 각 테이블, 뷰 또는 컬렉션 세그먼트는 자체 스레드에 할당됩니다. 이러한	동시에 로드할 테이블, 뷰 또는 컬렉션 파티션, 하위 파티션 또는 세그먼트를 식별하는 메커니즘입니다.

파라미터	가능한 값	설명
	세그먼트는 열 이름(columns) 및 열 값(boundaries)으로 지정합니다. PostgreSQL 엔드포인트는 이러한 유형의 병렬 로드만 지원합니다. MongoDB와 Amazon DocumentDB는 소스 엔드포인트로 이 범위 세그먼트화 유형과 병렬 전체 로드(partitions-auto)의 자동 세분화 유형을 모두 지원합니다. • none – 테이블, 뷰 또는 컬렉션은 파티션 또는 하위 파티션에 관계없이 단일 스레드 작업(기본값)으로 로드됩니다. 자세한 내용은 작업 생성 단원을 참조하십시오.	
number-of-partitions	(선택 사항) type이 MongoDB 또는 Amazon DocumentDB 엔드포인트의 지정된 컬렉션에 대한 partitions-auto 인 경우, 이 파라미터는 마이그레이션에 사용되는 총 파티션(세그먼트) 수를 지정합니다. 기본값은 16입니다.	동시에 로드할 파티션의 정확한 이름을 지정합니다.
collection-count-from-metadata	(선택 사항) type가 MongoDB 또는 Amazon DocumentDB 엔드포인트의 지정된 컬렉션 partitions-auto에 대해 이고이 파라미터가 로 설정된 경우 true예상 컬렉션 수를 AWS DMS 사용하여 파티션 수를 결정합니다. 이 파라미터가 로 설정된 경우 실제 컬렉션 수를 false AWS DMS 사용합니다. 기본값은 true입니다.	동시에 로드할 파티션 수를 계산할 때 예상 컬렉션 수를 사용할지 아니면 실제 컬렉션 수를 사용할지를 지정합니다.

파라미터	가능한 값	설명
max-records-skip-per-page	(선택 사항) MongoDB 또는 Amazon DocumentDB 엔드포인트의 지정된 컬렉션에 대한 type이 partitions-auto 인 경우, 이것은 각 파티션의 경계를 결정할 때 한 번에 건너뛰어야 하는 레코드 수를 나타냅니다. AWS DMS 는 페이지징된 건너뛰기 방식을 사용하여 파티션의 최소 경계를 결정합니다. 기본값은 10,000입니다.	각 파티션의 경계를 결정할 때 한 번에 건너뛴 레코드 수를 지정합니다. 기본값보다 비교적 큰 값을 설정하면 커서 제한 시간이 초과되고 작업이 실패할 수 있습니다. 기본값보다 비교적 작은 값을 설정하면 페이지당 작업 수가 늘어나고 전체 로드 속도가 느려집니다.
batch-size	(선택 사항) MongoDB 또는 Amazon DocumentDB 엔드포인트의 지정된 컬렉션에 대한 type이 partitions-auto 인 경우, 이 정수 값은 한 번의 왕복 배치에서 반환되는 문서 수를 제한합니다. 배치 크기가 0인 경우, 커서는 서버에서 정의한 최대 배치 크기를 사용합니다. 기본값은 0입니다.	하나의 배치에 반환된 최대 문서 수를 지정합니다. 배치마다 서버를 왕복해야 합니다.
partitions	type이 partitions-list 일 때 동시에 로드할 파티션의 이름을 지정하는 문자열 배열입니다.	동시에 로드할 파티션의 이름입니다.
subpartitions	(Oracle 엔드포인트 전용) type이 partitions-list 일 때 동시에 로드할 하위 파티션의 이름을 지정하는 문자열 배열입니다.	동시에 로드할 하위 파티션의 이름입니다.
columns	type이 ranges일 때 동시에 로드할 범위 기반 테이블, 뷰 또는 컬렉션 세그먼트를 식별하는 열 이름으로 설정된 문자열 배열입니다.	동시에 로드할 범위 기반 테이블, 뷰 또는 컬렉션 세그먼트를 식별하는 열 이름입니다.

파라미터	가능한 값	설명
boundaries	type이 ranges일 때 열-값 배열의 배열입니다. 각 열-값 배열은 columns에 의해 지정된 수량과 순서로 열 값을 포함합니다. 열-값 배열은 테이블, 뷰 또는 컬렉션 세그먼트의 상한을 지정합니다. 각 추가 열-값 배열은 하나의 추가 테이블, 뷰 또는 컬렉션 세그먼트에 대한 상한을 추가합니다. 이러한 모든 범위 기반 테이블, 뷰 또는 컬렉션 세그먼트는 동시에 로드됩니다.	동시에 로드할 범위 기반 테이블, 뷰 또는 컬렉션 파티션을 식별하는 열 값입니다.
lob-settings	객체는 다음 파라미터를 사용합니다. - mode – LOB에 대한 마이그레이션 처리 모드를 지정합니다. - bulk-max-size – mode 설정에 따라 LOB의 최대 크기를 지정합니다.	object-locator 옵션으로 식별된 테이블 또는 뷰에 대한 LOB 처리를 지정하는 값입니다. 지정된 LOB 처리는 이 테이블 또는 뷰에만 지정된 작업 LOB 설정을 재정의합니다. LOB 설정 파라미터 사용에 관한 자세한 내용은 선택한 테이블 또는 뷰의 LOB 설정 지정 섹션을 참조하세요.

파라미터	가능한 값	설명
mode	다음 값을 사용하여 지정한 테이블 또는 뷰의 LOB에 대한 마이그레이션 처리를 지정합니다. • limited – (기본값) 이 값은 모든 LOB가 테이블 또는 뷰의 다른 모든 열 데이터 형식과 함께 인라인으로 마이그레이션되는 제한적 LOB 모드로 마이그레이션을 설정합니다. 이 값은 주로 작은 LOB(100MB 이하)를 복제하는 경우에 사용합니다. 또한 <code>bulk-max-size</code> 값을 지정합니다(0은 유효하지 않음). 마이그레이션된 모든 LOB가 <code>bulk-max-size</code> 보다 클 경우 사용자가 설정한 크기로 잘립니다. • unlimited – 이 값은 마이그레이션을 전체 LOB 모드로 설정합니다. 이 값은 복제하려는 LOB의 전체 또는 대부분이 1GB보다 큰 경우 사용합니다. <code>bulk-max-size</code> 값을 0으로 지정하면 모든 LOB가 표준 전체 LOB 모드로 마이그레이션됩니다. 이 형식의 <code>unlimited</code> 모드에서 모든 LOB는 소스 테이블 또는 뷰의 조회를 사용하여 다른 열 데이터 형식과 별도로 마이그레이션됩니다. <code>bulk-max-size</code> 값을 0보다 큰 값으로 지정하면 모든 LOB가 조합 전체 LOB 모드로 마이그레이션됩니다. 이 형식의 <code>unlimited</code> 모드에서는 <code>bulk-max-size</code> 보다 큰 LOB가 표준 전체 LOB 모드와 비슷한 방식으로 소스 테이블 또는	LOB를 마이그레이션하는데 사용되는 메커니즘입니다.

파라미터	가능한 값	설명
	는 뷰 조회를 사용하여 마이그레이션됩니다. 그렇지 않으면 제한적 LOB 모드와 유사하게 이 크기 이하의 LOB가 인라인으로 마이그레이션됩니다. 사용하는 형식에 상관없이 unlimited 모드에서는 LOB가 잘리지 않습니다. • none – 모든 테이블 또는 뷰 LOB는 작업 LOB 설정에 따라 마이그레이션됩니다. 작업 LOB 설정에 관한 자세한 내용은 대상 메타데이터 작업 설정 섹션을 참조하세요. LOB를 마이그레이션하는 방법 및 이러한 작업 LOB 설정을 지정하는 방법에 관한 자세한 내용은 AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정 섹션을 참조하세요.	
bulk-max-size	이 값의 효과는 mode에 따라 다릅니다.	LOB의 최대 크기입니다(킬로바이트 단위). 작은 LOB를 복제해야 하거나 대상 엔드포인트가 무제한 LOB 크기를 지원하지 않는 경우에만 이 옵션을 지정하세요.

테이블 설정의 와일드카드는 제한됨

"table-settings" 규칙에 퍼센트 와일드카드("%")를 사용하는 것은 다음과 같이 소스 데이터베이스에서 지원되지 않습니다.

```
{
  "rule-type": "table-settings",
```

```

"rule-id": "8",
"rule-name": "8",
"object-locator": {
  "schema-name": "ipeline-prod",
  "table-name": "%"
},
"parallel-load": {
  "type": "partitions-auto",
  "number-of-partitions": 16,
  "collection-count-from-metadata": "true",
  "max-records-skip-per-page": 1000000,
  "batch-size": 50000
}
}

```

다음과 같이 "table-settings" 규칙에 "%"를 사용하는 경우는 다음과 같은 예외를 AWS DMS 반환합니다.

```

Error in mapping rules. Rule with ruleId = x failed validation. Exact
schema and table name required when using table settings rule.

```

또한에서 단일 작업을 사용하여 많은 수의 대용량 컬렉션을 로드하지 않는 것이 AWS 좋습니다. `parallel-load`. 단, AWS DMS 는 `MaxFullLoadSubTasks` 작업 설정 파라미터의 값에 따라 동시에 로드되는 세그먼트 수는 물론, 리소스 경합도 제한하며 최대값은 49입니다.

그 대신 각 "schema-name" 및 "table-name"을 개별적으로 지정하여 가장 큰 컬렉션에 대한 소스 데이터베이스의 모든 컬렉션을 지정하세요. 또한 마이그레이션 규모를 적절하게 확장하세요. 예를 들어, 충분한 수의 복제 인스턴스에서 여러 작업을 실행하여 데이터베이스의 수많은 대규모 컬렉션을 처리할 수 있습니다.

선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용

선택한 관계형 테이블, 뷰 및 컬렉션에 병렬 로드를 사용하여 마이그레이션 속도와 효율성을 향상시킬 수 있습니다. 즉, 여러 스레드를 동시에 사용하여 단일한 세그먼트화 테이블, 뷰 또는 컬렉션을 마이그레이션할 수 있습니다. 이를 위해서는 전체 로드 작업을 스레드로 AWS DMS 분할하고 각 테이블 세그먼트를 자체 스레드에 할당합니다.

이 병렬 로드 프로세스를 사용하면 먼저 다중 스레드가 소스 엔드포인트에서 여러 테이블, 뷰 및 컬렉션을 동시에 언로드할 수 있습니다. 그런 다음, 여러 스레드가 동일한 테이블, 뷰 및 컬렉션을 대상 엔드포인트에 동시에 마이그레이션하고 로드할 수 있습니다. 일부 데이터베이스 엔진의 경우, 기존 파티션

이나 하위 파티션별로 테이블 및 뷰를 세그먼트화할 수 있습니다. 다른 데이터베이스 엔진의 경우 특정 파라미터(자동 분할)에 따라 컬렉션을 AWS DMS 자동으로 분할할 수 있습니다. 그렇지 않으면 지정한 열 값의 범위로 테이블, 뷰 또는 컬렉션을 세그먼트화할 수 있습니다.

병렬 로드가 지원되는 소스 엔드포인트는 다음과 같습니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- IBM DB2 LUW
- SAP Adaptive Server Enterprise(ASE)
- MongoDB(병렬 전체 로드의 자동 세분화 및 범위 세그먼트화 옵션만 지원)
- Amazon DocumentDB(병렬 전체 로드의 자동 세분화 및 범위 세그먼트화 옵션만 지원)

MongoDB 및 Amazon DocumentDB 엔드포인트의 경우는 병렬 전체 로드의 범위 분할 옵션에 대한 파티션 키인 열에 대해 다음 데이터 형식을 AWS DMS 지원합니다.

- 배정밀도 실수
- 문자열
- ObjectId
- 32비트 정수
- 64비트 정수

테이블 설정 규칙과 함께 사용할 병렬 로드는 다음 대상 엔드포인트에 대해 지원됩니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- Amazon S3
- SAP Adaptive Server Enterprise(ASE)
- Amazon Redshift

- MongoDB(병렬 전체 로드의 자동 세분화 및 범위 세그먼트화 옵션만 지원)
- Amazon DocumentDB(병렬 전체 로드의 자동 세분화 및 범위 세그먼트화 옵션만 지원)
- Db2 LUW

동시에 로드할 최대 테이블 및 뷰 수를 지정하려면 `MaxFullLoadSubTasks` 작업 설정을 사용합니다.

병렬 로드 작업의 지원되는 대상에 대해 테이블 또는 뷰당 최대 스레드 수를 지정하려면 열-값 경계를 사용하여 더 많은 세그먼트를 정의하세요.

Important

`MaxFullLoadSubTasks`는 동시에 로드할 테이블 또는 테이블 세그먼트의 수를 제어합니다. `ParallelLoadThreads`는 마이그레이션 작업에서 로드를 동시에 실행하는 데 사용하는 스레드의 수를 제어합니다. 이러한 설정은 배수로 적용됩니다. 따라서 전체 로드 작업 중에 사용된 스레드의 총 개수는 대략 `ParallelLoadThreads` 의 값에 `MaxFullLoadSubTasks`의 값을 곱한 결과(즉, `ParallelLoadThreads * MaxFullLoadSubtasks`))입니다. 전체 로드 하위 작업의 수가 많고 병렬 로드 스레드의 수가 많은 작업을 만들면 작업이 너무 많은 메모리를 소비하여 실패할 수 있습니다.

Amazon DynamoDB, Amazon Kinesis Data Streams, Apache Kafka 또는 Amazon Elasticsearch Service 대상의 테이블당 최대 스레드 수를 지정하려면 `ParallelLoadThreads` 대상 메타데이터 작업 설정을 사용하세요.

`ParallelLoadThreads` 사용 시 병렬 로드 작업의 버퍼 크기를 지정하려면 `ParallelLoadBufferSize` 대상 메타데이터 작업 설정을 사용합니다.

`ParallelLoadThreads` 및 `ParallelLoadBufferSize`의 가용성과 설정은 대상 엔드포인트에 따라 다릅니다.

`ParallelLoadThreads` 및 `ParallelLoadBufferSize` 설정에 관한 자세한 내용은 [대상 메타데이터 작업 설정](#)을 참조하세요. `MaxFullLoadSubTasks` 설정에 관한 자세한 내용은 [전체 로드 작업 설정](#)를 참조하세요. 대상 엔드포인트에 관한 자세한 내용은 관련 주제를 참조하세요.

병렬 로드를 사용하려면 `parallel-load` 옵션을 사용하여 `table-settings` 유형의 테이블 매핑 규칙을 만듭니다. `table-settings` 규칙 내에서 동시에 로드하려는 단일한 테이블, 뷰 또는 컬렉션의 세그먼트화 기준을 지정할 수 있습니다. 이렇게 하려면 `parallel-load` 옵션의 `type` 파라미터를 여러 옵션 중 하나로 설정합니다.

이러한 설정을 수행하는 방법은 병렬 로드를 위해 테이블, 뷰 또는 컬렉션을 세그먼트화하려는 방법에 따라 다릅니다.

- 파티션별(세그먼트별) – `partitions-auto` 유형을 사용하여 기존 테이블 또는 뷰 파티션(세그먼트)을 모두 로드합니다. 또는 지정된 파티션 배열이 있는 `partitions-list` 유형을 사용하여 선택한 파티션만 로드합니다.

MongoDB 및 Amazon DocumentDB 엔드포인트에만 `partitions-auto` 유형 및 추가 선택적 `table-settings` 파라미터를 사용하여가 AWS DMS 자동으로 계산하는 세그먼트별로 모든 또는 지정된 컬렉션을 로드합니다.

- (Oracle 엔드포인트 전용) 하위 파티션별 – `subpartitions-auto` 유형을 사용하여 기존 테이블 또는 뷰 하위 파티션을 모두 로드합니다. 또는 지정된 `subpartitions` 배열이 있는 `partitions-list` 유형을 사용하여 선택한 하위 파티션만 로드합니다.
- 정의한 세그먼트별 – 열-값 경계를 사용하여 정의한 테이블, 뷰 또는 컬렉션 세그먼트를 로드합니다. 이렇게 하려면 지정된 `ranges` 및 `columns` 배열이 있는 `boundaries` 유형을 사용하세요.

Note

PostgreSQL 엔드포인트는 이러한 유형의 병렬 로드만 지원합니다. MongoDB와 Amazon DocumentDB는 소스 엔드포인트로 이 범위 세그먼트화 유형과 병렬 전체 로드 (`partitions-auto`)의 자동 세분화 유형을 모두 지원합니다.

동시에 로드할 추가 테이블, 뷰 또는 컬렉션을 식별하려면 `parallel-load` 옵션을 사용하여 추가 `table-settings` 객체를 지정합니다.

다음 절차에서는 가장 간단한 것부터 가장 복잡한 것까지 각 병렬 로드 유형별로 JSON을 코드화하는 방법을 확인할 수 있습니다.

모든 테이블, 뷰 또는 컬렉션 파티션이나 모든 테이블 또는 뷰 하위 파티션을 지정하려면

- `parallel-load`를 `partitions-auto` 유형 또는 `subpartitions-auto` 유형으로 지정하세요(동시 지정은 불가).

그러면 모든 테이블, 뷰 또는 컬렉션 파티션(또는 세그먼트) 또는 하위 파티션이 자동으로 자체 스레드에 할당됩니다.

일부 엔드포인트의 경우, 파티션 또는 하위 파티션은 테이블 또는 뷰에 이미 정의된 경우에만 병렬 로드 포함됩니다. MongoDB 및 Amazon DocumentDB 소스 엔드포인트의 경우 선택적 추가

파라미터를 기반으로 파티션(또는 세그먼트)을 AWS DMS 자동으로 계산할 수 있습니다. 여기에는 number-of-partitions, collection-count-from-metadata, max-records-skip-per-page, batch-size가 포함됩니다.

선택한 테이블 또는 뷰 파티션, 하위 파티션 또는 둘 다 지정하려면

1. partitions-list 유형으로 parallel-load를 지정합니다.
2. (선택 사항) 파티션 이름의 배열을 partitions 값으로 지정하여 파티션을 포함합니다.

그러면 지정된 각 파티션이 자체 스레드에 할당됩니다.

 Important

Oracle 엔드포인트의 경우, 병렬 로드를 위해 파티션과 하위 파티션을 선택할 때 파티션과 하위 파티션이 겹치지 않는지 확인하세요. 겹치는 파티션과 하위 파티션을 사용하여 데이터를 동시에 로드하면 항목이 중복되거나 프라이머리 키 중복 위반으로 인해 오류가 발생합니다.

3. (선택 사항) Oracle 엔드포인트의 경우 한해 하위 파티션 이름의 배열을 subpartitions 값으로 지정하여 하위 파티션을 포함합니다.

그러면 지정된 각 하위 파티션이 자체 스레드에 할당됩니다.

 Note

파티션 또는 하위 파티션은 테이블 또는 뷰에 이미 정의된 경우에만 병렬 로드에는 포함됩니다.

테이블 또는 뷰 세그먼트를 열 값의 범위로 지정할 수 있습니다. 이렇게 하려면 다음 열 특성을 알고 있어야 합니다.

- 인덱싱된 열을 지정하면 성능이 크게 향상됩니다.
- 최대 10개의 열을 지정할 수 있습니다.
- 열을 사용하여 DOUBLE, FLOAT, BLOB, CLOB 및 NCLOB AWS DMS 데이터 형식으로 세그먼트 경계를 정의할 수 없습니다.
- null 값의 레코드는 복제되지 않습니다.

테이블, 뷰 또는 컬렉션 세그먼트를 열 값의 범위로 지정하려면

1. `ranges` 유형으로 `parallel-load`를 지정합니다.
2. 열 이름의 배열을 `columns` 값으로 지정하여 테이블 또는 뷰 세그먼트 사이의 경계를 정의합니다. 테이블 또는 뷰 세그먼트 사이의 경계를 정의하려는 모든 열에 대해 이 작업을 수행합니다.

열의 순서는 중요합니다. 다음 설명과 같이 각 경계를 정의하는 데 있어 첫 번째 열이 가장 중요하고 마지막 열은 가장 덜 중요합니다.

3. 모든 테이블 또는 뷰 세그먼트의 데이터 범위를 정의하려면 경계 배열의 값을 `boundaries`로 지정하세요. 경계 배열은 열-값 배열의 배열입니다. 이 작업을 수행하려면 다음 단계를 수행하세요.
 - a. 열-값 배열의 각 요소를 각 열에 맞는 값으로 지정합니다. 열-값 배열은 정의하려는 각 테이블 또는 뷰 세그먼트의 상한을 나타냅니다. `columns` 배열에서 해당 열을 지정한 순서대로 각 열을 지정합니다.

소스에서 지원하는 형식으로 DATE 열의 값을 입력합니다.

- b. 테이블 또는 뷰의 맨 아래에서 맨 위의 열 세그먼트까지 각 세그먼트의 순서에 따라 각 열-값 배열을 상한으로 지정합니다. 지정한 맨 위 상한선 위에 행이 있으면 이 행은 테이블 또는 뷰의 맨 위 세그먼트를 완료합니다. 따라서 범위 기반 세그먼트의 수가 경계 배열의 세그먼트 경계 수보다 하나 더 많을 수 있습니다. 이러한 범위 기반 세그먼트는 각각 자체 스레드에 할당됩니다.

테이블 또는 뷰의 모든 열에 데이터 범위를 정의하지 않더라도 null이 아닌 모든 데이터가 복제됩니다.

예를 들어, 다음과 같이 열 COL1, COL2 및 COL3에 세 개의 열-값 배열을 정의한다고 가정합니다.

COL1	COL2	COL3
10	30	105
20	20	120
100	12	99

가능한 총 4개 세그먼트에 세 개의 세그먼트 경계를 정의했습니다.

각 세그먼트별로 복제할 행의 범위를 식별하기 위해 복제 인스턴스는 각각의 4개 세그먼트에서 이 3개 열에 검색을 적용합니다. 검색은 다음과 같습니다.

세그먼트 1

첫 번째 두 열의 값이 해당 세그먼트 1 상한값보다 작거나 같을 경우, 해당 모든 행을 복제합니다. 세 번째 열의 값이 세그먼트 1 상한 값 미만인 경우 모든 행을 복제합니다.

세그먼트 2

세그먼트 1 행을 제외한 모든 행을 복제하는 경우는 처음 두 열의 값이 해당 세그먼트 2 상한 값 이하입니다. 세 번째 열의 값이 세그먼트 2 상한 값 미만일 때 해당합니다.

세그먼트 3

세그먼트 2 행을 제외한 모든 행을 복제하는 경우는 처음 두 열의 값이 해당 세그먼트 3 상한 값 이하이고 세 번째 열의 값이 세그먼트 3 상한 값 미만일 때 해당합니다.

세그먼트 4

세그먼트 1, 2 및 3 행을 제외한 나머지 모든 행을 복제합니다.

이 경우, 복제 인스턴스는 다음과 같이 WHERE 절을 작성하여 각 세그먼트를 로드합니다.

세그먼트 1

```
((COL1 < 10) OR ((COL1 = 10) AND (COL2 < 30)) OR ((COL1 = 10) AND (COL2 = 30) AND (COL3 < 105)))
```

세그먼트 2

```
NOT ((COL1 < 10) OR ((COL1 = 10) AND (COL2 < 30)) OR ((COL1 = 10) AND (COL2 = 30) AND (COL3 < 105))) AND ((COL1 < 20) OR ((COL1 = 20) AND (COL2 < 20)) OR ((COL1 = 20) AND (COL2 = 20) AND (COL3 < 120)))
```

세그먼트 3

```
NOT ((COL1 < 20) OR ((COL1 = 20) AND (COL2 < 20)) OR ((COL1 = 20) AND (COL2 = 20) AND (COL3 < 120))) AND ((COL1 < 100) OR ((COL1 = 100) AND (COL2 < 12)) OR ((COL1 = 100) AND (COL2 = 12) AND (COL3 < 99)))
```

세그먼트 4

```
NOT ((COL1 < 100) OR ((COL1 = 100) AND (COL2 < 12)) OR ((COL1 = 100)
AND (COL2 = 12) AND (COL3 < 99)))
```

선택한 테이블 또는 뷰의 LOB 설정 지정

하나 이상의 table-settings 객체에 대해 lob-settings 옵션으로 table-settings 유형의 테이블 매핑 규칙을 생성하여 하나 이상의 테이블에 대한 작업 LOB 설정을 지정할 수 있습니다.

선택한 테이블 또는 뷰에 대한 LOB 설정 지정은 다음 소스 엔드포인트에서 지원됩니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- IBM DB2는 mode 및 bulk-max-size 설정에 따라 다음과 같이 설명됩니다.
- SAP Adaptive Server Enterprise(ASE)는 다음과 같이 mode 및 bulk-max-size 설정에 따라 다릅니다.

선택한 테이블 또는 뷰에 대한 LOB 설정 지정은 다음 대상 엔드포인트에서 지원됩니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- SAP ASE는 mode 및 bulk-max-size 설정에 따라 다음과 같이 설명됩니다.

Note

프라이머리 키를 포함하는 테이블 및 뷰에서만 LOB 데이터 형식을 사용할 수 있습니다.

선택한 테이블 또는 뷰의 LOB 설정을 사용하려면 table-settings 옵션을 사용하여 lob-settings 유형의 테이블 매핑 규칙을 생성하세요. 이렇게 하면 object-locator 옵션으로 식별된

테이블 또는 뷰에 대한 LOB 처리가 지정됩니다. `table-settings` 규칙 내에서 다음 파라미터를 사용하여 `lob-settings` 객체를 지정할 수 있습니다.

- `mode` – 다음과 같이 선택한 테이블 또는 뷰에 LOB 마이그레이션 처리 메커니즘을 지정합니다.
- `limited` – 기본 모드인 제한적 LOB 모드는 가장 빠르고 효율적입니다. 모든 LOB의 크기가 작거나(100MB 이내) 대상 엔드포인트가 무제한 LOB 크기를 지원하지 않는 경우에만 이 모드를 사용하세요. 또한 `limited`를 사용하는 경우 모든 LOB는 `bulk-max-size`에 설정된 크기 내에 있어야 합니다.

전체 로드 작업의 경우, 이 모드에서 복제 인스턴스는 기본 테이블 또는 뷰 스토리지의 일부로 다른 열 데이터 형식과 함께 모든 LOB를 인라인으로 마이그레이션합니다. 그러나 인스턴스는 `bulk-max-size` 값보다 큰 마이그레이션된 LOB를 모두 지정된 크기로 자릅니다. CDC(Change Data Capture) 로드 작업의 경우, 인스턴스는 표준 전체 LOB 모드(다음 참조)에서처럼 소스 테이블 조회를 사용하여 모든 LOB를 마이그레이션합니다.

Note

뷰는 전체 로드 작업에 대해서만 마이그레이션할 수 있습니다.

- `unlimited` – 전체 LOB 모드의 마이그레이션 메커니즘은 다음과 같이 `bulk-max-size`에 설정한 값에 따라 다릅니다.
- 표준 전체 LOB 모드 – `bulk-max-size`를 0으로 설정하면 복제 인스턴스가 표준 전체 LOB 모드를 사용하여 모든 LOB를 마이그레이션합니다. 이 모드에서는 크기에 관계없이 모든 LOB를 마이그레이션하려면 소스 테이블 또는 뷰를 조회해야 합니다. 이 경우, 일반적으로 제한적 LOB 모드보다 마이그레이션 속도가 훨씬 느립니다. 대부분 또는 모든 LOB가 큰 경우에만(1GB 이상) 이 모드를 사용하세요.
- 조합 전체 LOB 모드 – `bulk-max-size`를 0이 아닌 값으로 설정하면 이 전체 LOB 모드는 제한적 LOB 모드와 표준 전체 LOB 모드를 조합하여 사용합니다. 전체 로드 작업에서 LOB 크기가 `bulk-max-size` 값 이내에 있는 경우, 인스턴스는 제한적 LOB 모드에서처럼 LOB를 인라인으로 마이그레이션합니다. LOB 크기가 이 값보다 큰 경우, 인스턴스는 표준 전체 LOB 모드에서와 같이 소스 테이블 또는 뷰 조회를 사용하여 LOB를 마이그레이션합니다. CDC(Change Data Capture) 로드 작업의 경우, 인스턴스는 표준 전체 LOB 모드(다음 참조)에서처럼 소스 테이블 조회를 사용하여 모든 LOB를 마이그레이션합니다. LOB 크기와는 관계없습니다.

Note

뷰는 전체 로드 작업에 대해서만 마이그레이션할 수 있습니다.

이 모드의 마이그레이션 속도는 빠른 제한적 LOB 모드와 느린 표준 전체 LOB 모드 사이에서 절충한 수준이 됩니다. 작고 큰 LOB가 혼합되어 있지만 대부분의 LOB가 작은 경우에만 이 모드를 사용하세요.

이 조합된 전체 LOB 모드는 다음 엔드포인트에서만 사용 가능합니다.

- IBM DB2 소스로만 사용
- SAP ASE 소스 또는 대상으로 사용

unlimited 모드를 지정하는 메커니즘에 관계없이, 인스턴스는 모든 LOB를 자르지 않고 온전히 마이그레이션합니다.

- none – 복제 인스턴스는 작업 LOB 설정을 사용하여 선택한 테이블 또는 뷰의 LOB를 마이그레이션합니다. 이 옵션을 사용하면 선택한 테이블 또는 뷰의 LOB 설정 유무와 관계없이 마이그레이션 결과를 쉽게 비교할 수 있습니다.

복제에 포함된 LOB가 있는 테이블 또는 뷰를 지정한 경우 limited LOB 모드 사용 시 BatchApplyEnabled 작업 설정을 true로만 설정할 수 있습니다.

경우에 따라 BatchApplyEnabled를 true로, BatchApplyPreserveTransaction을 false로 설정해야 할 수 있습니다. 이러한 경우, 테이블 또는 뷰에 LOB가 있고 소스 및 대상 엔드포인트가 Oracle이면 인스턴스가 BatchApplyPreserveTransaction을 true로 설정합니다.

- bulk-max-size – 이전 항목에서 설명한 대로, mode에 따라 이 값을 0 또는 0이 아닌 값(킬로바이트)으로 설정합니다. limited 모드에서는 이 파라미터에 0이 아닌 값을 설정해야 합니다.

인스턴스는 LOB를 이진 형식으로 변환합니다. 따라서 복제해야 하는 가장 큰 LOB를 지정하려면 크기에 3을 곱하세요. 예를 들어, 가장 큰 LOB가 2MB라면 bulk-max-size를 6,000(6MB)으로 설정합니다.

테이블 설정 예제

다음은 테이블 설정 사용법을 보여주는 몇 가지 예제입니다.

Example 파티션별로 세그먼트화된 테이블 로드

다음 예제는 모든 파티션을 기반으로 소스의 SALES 테이블을 동시에 로드하기 때문에 보다 효율적입니다.

```
{
```

```

"rules": [{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
    "schema-name": "%",
    "table-name": "%"
  },
  "rule-action": "include"
},
{
  "rule-type": "table-settings",
  "rule-id": "2",
  "rule-name": "2",
  "object-locator": {
    "schema-name": "HR",
    "table-name": "SALES"
  },
  "parallel-load": {
    "type": "partitions-auto"
  }
}
]
}

```

Example하위 파티션별로 세그먼트화된 테이블 로드

다음 예제는 모든 하위 파티션을 기반으로 Oracle 소스의 SALES 테이블을 동시에 로드하기 때문에 보다 효율적입니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "table-settings",

```

```

        "rule-id": "2",
        "rule-name": "2",
        "object-locator": {
            "schema-name": "HR",
            "table-name": "SALES"
        },
        "parallel-load": {
            "type": "subpartitions-auto"
        }
    }
]
}

```

Example 파티션 목록별로 세그먼트화된 테이블 로드

다음 예제는 특정 파티션 목록별로 소스의 SALES 테이블을 동시에 로드합니다. 여기서 지정된 파티션은 알파벳의 일부로 시작하는 값을 따라 이름이 지정됩니다(예: ABCD, EFGH 등).

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "table-settings",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
      "schema-name": "HR",
      "table-name": "SALES"
    },
    "parallel-load": {
      "type": "partitions-list",
      "partitions": [
        "ABCD",
        "EFGH",
        "IJKL",

```



```

        "MNOP",
        "QRST",
        "UVWXYZ"
    ]
}
}
]
}

```

Example선택한 파티션 및 하위 파티션 목록으로 세그먼트화된 Oracle 테이블 로드

다음 예제는 선택한 파티션 및 하위 파티션 목록별로 Oracle 소스의 SALES 테이블을 동시에 로드합니다. 여기서 지정된 파티션은 알파벳의 일부로 시작하는 값을 따라 이름이 지정됩니다(예: ABCD, EFGH 등). 지정된 하위 파티션의 이름은 숫자로 시작하는 값을 따라 지정됩니다(예: 01234 및 56789).

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "table-settings",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
      "schema-name": "HR",
      "table-name": "SALES"
    },
    "parallel-load": {
      "type": "partitions-list",
      "partitions": [
        "ABCD",
        "EFGH",
        "IJKL",
        "MNOP",
        "QRST",
        "UVWXYZ"
      ]
    }
  }
]
}

```

```

        ],
        "subpartitions": [
            "01234",
            "56789"
        ]
    }
}
]
}

```

Example 열 값 범위별로 세그먼트화된 테이블 로드

다음 예에서는 SALES_NO 및 REGION 열 값의 범위로 지정된 세그먼트별로 소스의 SALES 테이블을 동시에 로드합니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "table-settings",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
      "schema-name": "HR",
      "table-name": "SALES"
    },
    "parallel-load": {
      "type": "ranges",
      "columns": [
        "SALES_NO",
        "REGION"
      ],
      "boundaries": [
        "1000",

```

```

        "NORTH"
      ],
      [
        "3000",
        "WEST"
      ]
    ]
  }
}

```

여기에서는 이름이 SALES_NO 및 REGION인 세그먼트 범위에 두 개의 열이 지정됩니다. 두 경계는 두 열 값 집합(["1000", "NORTH"] 및 ["3000", "WEST"])으로 지정됩니다.

따라서 이 두 경계는 동시에 로드할 다음 세 개 테이블 세그먼트를 식별합니다.

세그먼트 1

SALES_NO가 1,000 이하이고 REGION이 'NORTH'보다 작은 행. 즉, EAST 리전에서 1,000 이하의 매출 수치입니다.

세그먼트 2

세그먼트 1 이외의 행이 3,000개 SALES_NO 이하이고 'WEST' REGION 미만입니다. 즉, NORTH 및 SOUTH 리전에서 1000보다 크고 3,000 이하인 매출 수치입니다.

세그먼트 3

세그먼트 1 및 세그먼트 2 이외의 나머지 모든 행. 즉, 'WEST' 리전에서 3,000을 초과하는 매출 수치입니다.

Example두 개의 테이블 로드: 하나는 범위별로 세그먼트화, 다른 하나는 파티션별로 세그먼트화됨

다음 예제에서는 식별하는 세그먼트 경계로 SALES 테이블을 동시에 로드합니다. 또한 이전 예제와 같이 모든 파티션으로 ORDERS 테이블을 동시에 로드합니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {

```

```

        "schema-name": "%",
        "table-name": "%"
    },
    "rule-action": "include"
},
{
    "rule-type": "table-settings",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
        "schema-name": "HR",
        "table-name": "SALES"
    },
    "parallel-load": {
        "type": "ranges",
        "columns": [
            "SALES_NO",
            "REGION"
        ],
        "boundaries": [
            [
                "1000",
                "NORTH"
            ],
            [
                "3000",
                "WEST"
            ]
        ]
    }
},
{
    "rule-type": "table-settings",
    "rule-id": "3",
    "rule-name": "3",
    "object-locator": {
        "schema-name": "HR",
        "table-name": "ORDERS"
    },
    "parallel-load": {
        "type": "partitions-auto"
    }
}
]

```

```
}
```

Example작업 LOB 설정을 사용하여 LOB 포함 테이블 로드

다음 예제는 작업 LOB 설정을 사용하여 모든 LOB를 포함하여 소스의 ITEMS 테이블을 로드합니다. 100MB의 bulk-max-size 설정은 무시되며 limited 또는 unlimited 모드로 신속하게 재설정하는 경우에만 유지됩니다.

```
{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "table-settings",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
      "schema-name": "INV",
      "table-name": "ITEMS"
    },
    "lob-settings": {
      "mode": "none",
      "bulk-max-size": "100000"
    }
  }
  ]
}
```

Example제한적 LOB 모드를 사용하여 LOB 포함 테이블 로드

다음 예제는 제한적 LOB 모드(기본값)를 사용하여 소스의 LOB를 포함하는 ITEMS 테이블을 로드하며, 잘리지 않는 최대 크기로 100MB를 지원합니다. 이 크기보다 큰 LOB는 100MB로 잘립니다. 모든 LOB는 다른 모든 열 데이터 형식과 인라인으로 로드됩니다.

```
{
```

```

"rules": [{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
    "schema-name": "%",
    "table-name": "%"
  },
  "rule-action": "include"
},
{
  "rule-type": "table-settings",
  "rule-id": "2",
  "rule-name": "2",
  "object-locator": {
    "schema-name": "INV",
    "table-name": "ITEMS"
  },
  "lob-settings": {
    "bulk-max-size": "100000"
  }
}
]
}

```

Example표준 전체 LOB 모드를 사용하여 LOB 포함 테이블 로드

다음 예제는 표준 전체 LOB 모드를 사용하여 절단하지 않은 모든 LOB를 포함하여 소스의 ITEMS 테이블을 로드합니다. 크기에 관계없이 모든 LOB는 소스 테이블의 각 LOB 조회를 사용하여 다른 데이터 형식과 별도로 로드됩니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {

```

```

        "rule-type": "table-settings",
        "rule-id": "2",
        "rule-name": "2",
        "object-locator": {
            "schema-name": "INV",
            "table-name": "ITEMS"
        },
        "lob-settings": {
            "mode": "unlimited",
            "bulk-max-size": "0"
        }
    }
]
}

```

Example조합 전체 LOB 모드를 사용하여 LOB 포함 테이블 로드

다음 예제는 조합 전체 LOB 모드를 사용하여 절단하지 않은 모든 LOB를 포함하여 소스의 ITEMS 테이블을 로드합니다. 제한적 LOB 모드에서와 같이, 크기 100MB 이내의 모든 LOB는 다른 데이터 형식과 함께 인라인으로 로드됩니다. 크기가 100MB를 초과하는 모든 LOB는 다른 데이터 형식과 별도로 로드됩니다. 이 별도의 로드는 표준 전체 LOB 모드에서와 같이 소스 테이블의 각 LOB에 대해 조회를 사용합니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "%",
      "table-name": "%"
    },
    "rule-action": "include"
  },
  {
    "rule-type": "table-settings",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
      "schema-name": "INV",
      "table-name": "ITEMS"
    },
    "lob-settings": {

```

```

        "mode": "unlimited",
        "bulk-max-size": "100000"
    }
}
]
}

```

데이터 마스킹을 사용하여 민감한 정보 숨기기

마이그레이션 중인 테이블의 하나 이상의 열에 저장된 민감한 데이터를 숨기려면 데이터 마스킹 변환 규칙 작업을 활용할 수 있습니다. 버전 3.5.4부터를 AWS DMS 사용하면 테이블 매핑에서 데이터 마스킹 변환 규칙 작업을 사용할 수 있으므로 마이그레이션 프로세스 중에 하나 이상의 열의 내용을 변경할 수 있습니다.는 수정된 데이터를 대상 테이블에 AWS DMS 로드합니다.

AWS Database Migration Service 는 데이터 마스킹 변환 규칙 작업을 위한 세 가지 옵션을 제공합니다.

- 데이터 마스킹: 숫자 마스킹
- 데이터 마스킹: 숫자 임의화
- 데이터 마스킹: 해싱 마스크

이러한 데이터 마스킹 변환 규칙 작업은 다른 변환 규칙과 마찬가지로 복제 작업의 테이블 매핑에서 구성할 수 있습니다. 규칙 대상은 열 수준으로 설정해야 합니다.

마스킹 문자를 사용하여 열 데이터의 번호 마스킹

"데이터 마스킹: 숫자 마스킹" 변환 규칙 작업을 사용하면 숫자를 지정한 단일 ASCII 인쇄 가능 문자(빈 문자 또는 공백 문자 제외)로 대체하여 하나 이상의 열에서 숫자 데이터를 마스킹할 수 있습니다.

다음은 customer_master 테이블에 있는 cust_passport_no 열의 모든 숫자를 '#' 마스킹 문자로 마스킹하고 마스킹된 데이터를 대상 테이블에 로드하는 예제입니다.

```

    {
      "rules": [
        {
          "rule-type": "selection",
          "rule-id": "1",
          "rule-name": "1",

```



```

        "object-locator": {
            "schema-name": "cust_schema",
            "table-name": "%"
        },
        "rule-action": "include"
    },
    {
        "rule-type": "transformation",
        "rule-id": "2",
        "rule-name": "2",
        "rule-target": "column",
        "object-locator": {
            "schema-name": "cust_schema",
            "table-name": "customer_master",
            "column-name": "cust_passport_no"
        },
        "rule-action": "data-masking-digits-mask",
        "value": "#"
    }
]
}

```

예를 들어 소스 테이블의 열 `cust_passport_no`에 "C6BGJ566669K" 레코드가 포함된 경우 AWS DMS 작업은 이 데이터를 대상 테이블에 기록합니다 "C#BGJ#####K".

열의 숫자를 난수로 바꾸기

변환 규칙 "데이터 마스킹: 숫자 임의화"를 사용하면 하나 이상의 열에 있는 각 숫자 숫자를 난수로 바꿀 수 있습니다. 다음 예제에서는 소스 테이블 `cust_passport_no` 열의 모든 숫자를 난수 `customer_master`로 AWS DMS 바꾸고 수정된 데이터를 대상 테이블에 씁니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "cust_schema",
        "table-name": "%"
      },
    },
  ],
}

```

```

        "rule-action": "include"
    },
    {
        "rule-type": "transformation",
        "rule-id": "2",
        "rule-name": "2",
        "rule-target": "column",
        "object-locator": {
            "schema-name": "cust_schema",
            "table-name": "customer_master",
            "column-name": "cust_passport_no"
        },
        "rule-action": "data-masking-digits-randomize"
    }
]
}

```

예를 들어 AWS DMS 작업은 소스 테이블의 `cust_passport_no` 열 "C6BGJ566669K"에 있는 값을 로 변환 "C1BGJ842170K"하고 대상 데이터베이스에 씁니다.

열 데이터를 해시 값으로 바꾸기

변환 규칙 "데이터 마스킹: 해싱 마스크"를 사용하면 열 데이터를 SHA256 알고리즘을 사용하여 생성된 해시로 바꿀 수 있습니다. 해시 길이는 항상 64자이므로 대상 테이블 열 길이는 최소 64자여야 합니다. 또는 열에 `change-data-type` 변환 규칙 작업을 추가하여 대상 테이블의 열 너비를 늘릴 수 있습니다.

다음 예제에서는 소스 테이블의 `cust_passport_no` 열에 있는 데이터에 대해 64자 길이의 해시 값을 생성하고 열 길이를 늘린 후 변환된 데이터를 대상 테이블에 `customer_master` 로드합니다.

```

{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "cust_schema",
        "table-name": "%"
      },
    },
  ],
}

```

```

"rule-action": "include"
},
{
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "cust_schema",
    "table-name": "customer_master",
    "column-name": "cust_passport_no"
  },
  "rule-action": "change-data-type",
  "data-type": {
    "type": "string",
    "length": "100",
    "scale": ""
  }
},
{
  "rule-type": "transformation",
  "rule-id": "3",
  "rule-name": "3",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "cust_schema",
    "table-name": "customer_master",
    "column-name": "cust_passport_no"
  },
  "rule-action": "data-masking-hash-mask"
}
]
}

```

예를 들어 소스 테이블 `cust_passport_no`의 열에 값이 포함된 경우 “C6BGJ566669K” AWS DMS 작업은 대상 테이블에 해시 “7CB06784764C9030CCC41E25C15339FEB293FFE9B329A72B5FED564E99900C75”를 씁니다.

제한 사항

- 각 데이터 마스킹 변환 규칙 옵션은 특정 AWS DMS 데이터 유형에 대해서만 지원됩니다.

- 데이터 마스킹: 데이터 형식의 열인 WSTRING 및 STRING에 대해 숫자 마스킹이 지원됩니다.
- 데이터 마스킹: Digits Randomize는 서명되지 않은 WSTRING, STRING; NUMERIC, INT1, INT2, INT4, and INT8 데이터 형식의 열에 대해 지원됩니다.
- 데이터 마스킹: 해싱 마스크는 WSTRING 및 STRING 데이터 유형의 열에 대해 지원됩니다.

소스 엔진의 AWS DMS 데이터 형식에 데이터 형식을 매핑하는 방법에 대해 자세히 알아보려면 소스 엔진과 데이터 형식의 AWS DMS 데이터 형식 매핑을 참조하세요. [Oracle용 소스 데이터 형식](#), [SQL Server용 소스 데이터 형식](#), [PostgreSQL용 소스 데이터 형식](#) 및 [MySQL용 소스 데이터 형식](#)에 대한 소스 데이터 유형을 참조하세요.

- 데이터 유형이 호환되지 않는 열에 데이터 마스킹 규칙 작업을 사용하면 DMS 작업에서 오류가 발생합니다. 오류 처리 동작을 지정하려면 DMS 작업 설정의 DataMaskingErrorPolicy를 참조하세요. DataMaskingErrorPolicy에 대한 자세한 내용은 [오류 처리 작업 설정](#) 섹션을 참조하세요.
- 소스 열 유형이 사용하려는 마스킹 옵션에 대해 지원되지 않는 경우 change-data-type 변환 규칙 작업을 추가하여 열의 데이터 유형을 호환되는 유형으로 변경할 수 있습니다. 데이터 형식이 마스킹 전에 변경되도록 change-data-type 변환의 rule-id는 마스킹 변환의 규칙 ID보다 작은 숫자여야 합니다.
- 생성된 해시 값이 고유하고 일관되므로 데이터 마스킹 사용: 프라이머리 키/고유 키/외부 키 열을 마스킹하기 위한 해싱 마스크 작업입니다. 다른 두 마스킹 옵션은 고유성을 보장할 수 없습니다.
- 데이터 마스킹: 숫자 마스킹 및 데이터 마스킹: 숫자 임의화는 열 데이터의 숫자에만 영향을 미치고 데이터 길이에는 영향을 미치지 않지만, 데이터 마스킹: 해싱 마스크는 전체 열을 수정하고 데이터 길이는 64자로 변경됩니다. 따라서 그에 따라 생성할 대상 테이블 또는 마스킹되는 열에 대해 change-data-type 변환 규칙을 추가해야 합니다.
- 데이터 마스킹 변환 규칙 작업이 지정된 열은 AWS DMS의 데이터 검증에서 제외됩니다. 프라이머리 키/고유 키 열이 마스킹된 경우 이 테이블에 대한 데이터 검증이 실행되지 않습니다. 이러한 테이블의 검증 상태는 No Primary key와 같습니다.

소스 필터 사용

소스 필터를 사용하여 소스에서 대상으로 전송되는 레코드의 수와 유형을 제한할 수 있습니다. 예를 들어, 본사 사업장에서 근무하는 직원만이 대상 데이터베이스로 이동하도록 지정할 수 있습니다. 필터는 선택 규칙의 일부입니다. 데이터 열에서 필터를 적용합니다.

소스 필터는 다음 제약 조건을 준수해야 합니다.

- 선택 규칙에는 필터가 전혀 없거나 필터가 하나 이상 있을 수 있습니다.

- 모든 필터에는 하나 이상의 필터 조건이 있을 수 있습니다.
- 필터를 2개 이상 사용하는 경우, 필터 사이에서 AND 연산자를 사용하는 것처럼 필터 목록이 결합됩니다.
- 단일 필터 내에서 필터 조건을 2개 이상 사용하면 필터 조건 사이에서 OR 연산자를 사용하는 것처럼 필터 조건 목록이 결합됩니다.
- `rule-action = 'include'`인 경우 필터만이 적용됩니다.
- 필터에는 열 이름과 필터 조건 목록이 필요합니다. 필터 조건에는 연산자에 따라 값이 한 개, 두 개 또는 아무 값에도 연결되지 않는 필터 연산자가 있어야 합니다.
- 열 이름, 테이블 이름, 조회 이름 및 스키마 이름은 대소문자를 구분합니다. Oracle 및 Db2는 항상 대문자를 사용해야 합니다.
- 필터는 정확한 이름을 가진 테이블만 지원합니다. 필터는 와일드카드를 지원하지 않습니다.

소스 필터 사용 시 다음과 같은 제한 사항이 적용됩니다.

- 필터는 오른쪽에서 왼쪽으로 쓰는 언어의 열을 계산하지 않습니다.
- LOB 열에 필터를 적용하지 마세요.
- 생성 후 업데이트되지 않는 변경 불가능 열에만 필터를 적용합니다. 생성 후 업데이트할 수 있는 변경 가능 열에 소스 필터가 적용되면 불리한 동작이 발생할 수 있습니다.

예를 들어, 열의 특정 행을 제외하거나 포함하는 필터는 나중에 행이 변경되더라도 항상 지정된 행을 제외하거나 포함합니다. A열에 1~10 행을 제외하거나 포함시키고 나중에 11~20 행이 되도록 변경한다고 가정합니다. 이 경우, 데이터가 더 이상 동일하지 않은 경우에도 계속 제외되거나 포함됩니다.

마찬가지로 필터 범위 외부의 행이 나중에 업데이트(또는 업데이트 및 삭제)되고, 필터에 의해 정의된 대로 제외되거나 포함되어야 한다고 가정합니다. 이 경우, 대상에서 복제됩니다.

소스 필터를 사용할 때 다음과 같은 추가 문제가 적용됩니다.

- 필터링 정의에 포함된 열과 프라이머리 키를 사용하여 인덱스를 생성하는 것이 좋습니다.

JSON에서 소스 필터 규칙 생성

선택 규칙의 `JSON filters` 파라미터를 사용하여 소스 필터를 생성할 수 있습니다. `filters` 파라미터는 1개 이상의 JSON 객체 배열을 지정합니다. 각 객체는 소스 필터 유형, 열 이름 및 필터 조건을 지

정하는 파라미터를 가지고 있습니다. 이러한 필터 조건에는 하나 이상의 필터 연산자 및 필터 값이 포함됩니다.

다음 표에는 `filters` 객체에서 소스 필터링을 지정하기 위한 파라미터가 나와 있습니다.

파라미터	값
<code>filter-type</code>	<code>source</code>
<code>column-name</code>	필터를 적용할 원본 열의 이름이 있는 파라미터입니다. 이름은 대/소문자를 구분합니다.
<code>filter-conditions</code>	<code>filter-operator</code> 값에 따라 <code>filter-operator</code> 파라미터와 0 개 이상의 관련 값 파라미터를 포함하는 하나 이상의 객체로 구성된 배열입니다.
<code>filter-operator</code>	다음 값 중 하나를 가진 파라미터: <code>lte</code> – 하나의 값보다 작거나 같음 <code>ste</code> – 하나의 값보다 작거나 같음(<code>lte</code> 별칭) <code>gte</code> – 하나의 값보다 크거나 같음 <code>eq</code> – 하나의 값과 같음 <code>noteq</code> – 하나의 값과 같지 않음 <code>between</code> – 두 값과 같음 또는 두 값 사이의 값임 <code>notbetween</code> – 두 값과 같지 않음 또는 두 값 사이의 값이 아님 <code>null</code> – NULL 값 <code>notnull</code> – NULL 값 없음
<code>value</code> 또는	0개 이상의 값 파라미터가 <code>filter-operator</code> 과 연관되어 있음:
<code>start-value</code> 및 <code>end-value</code> 또는	<code>filter-operator</code> 가 <code>lte</code>, <code>ste</code>, <code>gte</code>, <code>eq</code> 또는 <code>noteq</code>인 경우, 값 파라미터 하나를 지정하는 데 <code>value</code>를 사용합니다. <code>filter-operator</code> 가 <code>between</code> 또는 <code>notbetween</code> 인 경우, <code>start-value</code> 및 <code>end-value</code> 를 사용하여 두 개의 값 파라미터를 지정합니다.
값 없음	

파라미터	값
	• <code>filter-operator</code> 가 <code>null</code> 또는 <code>notnull</code>인 경우, 값 없음 파라미터를 지정하세요.

다음 예제에서는 소스 필터를 사용하는 일반적인 방법에 대해 다룹니다.

Example 단일 필터

다음 필터는 `empid >= 100`인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "employee"
    },
    "rule-action": "include",
    "filters": [{
      "filter-type": "source",
      "column-name": "empid",
      "filter-conditions": [{
        "filter-operator": "gte",
        "value": "100"
      }]
    }]
  }]
}
```

Example 여러 필터 연산자

다음 필터는 여러 필터 연산자를 단일 데이터 열에 적용합니다. 이 필터는 (`empid <= 10`) OR (`empid is between 50 and 75`) OR (`empid >= 100`)인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
  "rules": [{
```

```

    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "employee"
    },
    "rule-action": "include",
    "filters": [{
      "filter-type": "source",
      "column-name": "empid",
      "filter-conditions": [{
        "filter-operator": "lte",
        "value": "10"
      }, {
        "filter-operator": "between",
        "start-value": "50",
        "end-value": "75"
      }, {
        "filter-operator": "gte",
        "value": "100"
      }]
    }]
  }]
}

```

Example 여러 필터

다음 필터는 여러 필터를 테이블의 열 2개에 적용합니다. 이 필터는 (empid <= 100) AND (dept = tech)인 모든 직원을 대상 데이터베이스로 복제합니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "employee"
    },
    "rule-action": "include",

```



```

    "filters": [{
      "filter-type": "source",
      "column-name": "empid",
      "filter-conditions": [{
        "filter-operator": "lte",
        "value": "100"
      }]
    }, {
      "filter-type": "source",
      "column-name": "dept",
      "filter-conditions": [{
        "filter-operator": "eq",
        "value": "tech"
      }]
    }]
  }
}

```

Example NULL 값 필터링

다음 필터는 빈 값을 필터링하는 방법을 보여줍니다. 이 필터는 dept = NULL인 모든 직원을 대상 데이터베이스로 복제합니다.

```

{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "employee"
    },
    "rule-action": "include",
    "filters": [{
      "filter-type": "source",
      "column-name": "dept",
      "filter-conditions": [{
        "filter-operator": "null"
      }]
    }]
  }]
}

```

```
}

```

Example NOT 연산자를 사용한 필터링

일부 연산자는 음수 형식으로 사용할 수 있습니다. 다음 필터는 (empid is < 50) OR (empid is > 75)인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "employee"
    },
    "rule-action": "include",
    "filters": [{
      "filter-type": "source",
      "column-name": "empid",
      "filter-conditions": [{
        "filter-operator": "notbetween",
        "start-value": "50",
        "end-value": "75"
      }]
    }]
  }]
}
```

Example혼합 필터 연산자 사용

AWS DMS 버전 3.5.0부터 포함 연산자와 음수 연산자를 혼합할 수 있습니다.

다음 필터는 (empid != 50) AND (dept is not NULL)인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
  "rules": [{
    "rule-type": "selection",
```

```

    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
        "schema-name": "test",
        "table-name": "employee"
    },
    "rule-action": "include",
    "filters": [{
        "filter-type": "source",
        "column-name": "empid",
        "filter-conditions": [{
            "filter-operator": "noteq",
            "value": "50"
        }]
    }, {
        "filter-type": "source",
        "column-name": "dept",
        "filter-conditions": [{
            "filter-operator": "notnull"
        }]
    }]
}]
}

```

null을 다른 필터 연산자와 함께 사용할 때는 다음 사항에 유의하세요.

- 동일한 필터 내에서 포함, 음수 및 null 필터 조건을 함께 사용하면 NULL 값이 있는 레코드가 복제되지 않습니다.
- 동일한 필터 내에서 포함 필터 조건 없이 음수 필터 조건과 null 필터 조건을 함께 사용하면 데이터가 복제되지 않습니다.
- null 필터 조건을 명시적으로 설정하지 않고 음수 필터 조건을 사용하면 NULL 값이 있는 레코드가 복제되지 않습니다.

시간과 날짜를 기준으로 필터링

가져올 데이터를 선택할 때 필터 기준의 일부로 날짜 또는 시간을 지정할 수 있습니다.는 필터링에 날짜 형식 YYYY-MM-DD 및 시간 형식 YYYY-MM-DD HH:MM:SS.SSS를 AWS DMS 사용합니다. AWS DMS 비교 함수는 SQLite 규칙을 따릅니다. SQLite 데이터 형식과 날짜 비교에 관한 자세한 내용은 SQLite 설명서의 [SQLite 버전 3의 데이터 형식](#)을 참조하세요.

다음 필터는 날짜에서 필터링하는 방법을 보여줍니다. 이 필터는 `empstartdate >= January 1, 2002`인 모든 직원을 대상 데이터베이스로 복제합니다.

Example 단일 날짜 필터

```
{
  "rules": [{
    "rule-type": "selection",
    "rule-id": "1",
    "rule-name": "1",
    "object-locator": {
      "schema-name": "test",
      "table-name": "employee"
    },
    "rule-action": "include",
    "filters": [{
      "filter-type": "source",
      "column-name": "empstartdate",
      "filter-conditions": [{
        "filter-operator": "gte",
        "value": "2002-01-01"
      }]
    }]
  }]
}
```

작업에 대한 마이그레이션 전 평가 활성화 및 활용

마이그레이션 전 평가에서는 데이터베이스 마이그레이션 작업의 특정 구성 요소를 평가하여 마이그레이션 작업이 예상대로 실행되는 데 방해가 될 수 있는 문제를 식별하는 데 도움이 됩니다. 이 평가를 통해 새 작업이나 수정된 태스크를 실행하기 전에 문제를 식별하고 수정할 수 있습니다. 이를 통해 요구 사항 누락 또는 알려진 제한으로 인한 태스크 실패와 관련된 지연을 방지할 수 있습니다.

AWS DMS 는 마이그레이션 전 평가를 위한 두 가지 옵션에 대한 액세스를 제공합니다.

- 데이터 유형 평가: 제한된 범위의 평가를 제공하는 레거시 보고서입니다.
- 마이그레이션 전 평가 실행: 데이터 유형 평가 결과를 포함하여 다양한 유형의 개별 평가를 포함합니다.

Note

마이그레이션 전 평가 실행을 선택하는 경우 데이터 유형 평가를 별도로 선택할 필요가 없습니다.

다음 주제에서는 이 옵션에 대해 설명합니다.

- [마이그레이션 전 평가 실행 지정, 시작, 보기](#): 마이그레이션 전 평가(권장) 평가 실행은 신규 또는 기존 마이그레이션 작업 구성을 기반으로 실행할 하나 이상의 개별 평가를 지정합니다. 개별 평가에서는 마이그레이션 유형, 지원되는 객체, 인덱스 구성, 기타 태스크 설정(예: 마이그레이션할 스키마와 테이블을 식별하는 테이블 매핑) 등의 기준 관점에서 지원되는 소스 및/또는 대상 데이터베이스의 특정 요소를 평가합니다.

예를 들어, 개별 평가에서는 AWS DMS 엔진 버전을 기반으로 마이그레이션할 수 있거나 마이그레이션할 수 없는 소스 데이터 유형 또는 프라이머리 키 포맷을 평가할 수 있습니다. AWS DMS Management Console을 사용하거나 AWS CLI 및 SDKs를 사용하여 AWS DMS API에 액세스하여 최신 평가 실행의 결과를 보고 작업에 대한 모든 이전 평가 실행의 결과를 볼 수 있습니다. 또한 이러한 결과를 저장 AWS DMS 하기 위해 선택한 Amazon S3 버킷의 작업에 대한 이전 평가 실행 결과를 볼 수 있습니다.

Note

사용 가능한 개별 평가의 수와 유형은 시간이 지남에 따라 증가할 수 있습니다. 주기적 업데이트에 관한 자세한 내용은 [개별 평가 지정](#)을 참조하세요.

- [데이터 유형\(레거시\) 평가 시작 및 보기](#): 데이터 유형(레거시) 평가는 단일 JSON 구조 즉, 지원되는 관계형 소스 데이터베이스 인스턴스에서 올바르게 마이그레이션되지 않을 수도 있는 데이터 유형을 사용하여 단일 유형의 마이그레이션 전 평가의 결과를 단일 JSON 구조로 반환합니다. 이 보고서는 마이그레이션을 위해 선택된 소스 데이터베이스의 모든 스키마 및 테이블에서 발견된 모든 문제 데이터 유형에 대한 결과를 반환합니다.

마이그레이션 전 평가를 위한 사전 조건 생성

이 섹션에서는 마이그레이션 전 평가를 생성하는 데 필요한 Amazon S3 및 IAM 리소스에 대해 설명합니다.

⚠ Important

다음 사전 조건은 자체 Amazon S3 버킷 및 IAM 역할을 제공하는 경우에만 필요합니다.

S3 버킷 생성

AWS DMS 는 마이그레이션 전 평가 보고서를 S3 버킷에 저장합니다. S3 버킷을 생성하려면 다음을 수행합니다.

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/s3/> Amazon S3 콘솔을 엽니다.
2. 버킷 생성을 선택합니다.
3. 버킷 만들기 페이지에서 버킷의 로그인 이름이 포함된 글로벌로 고유한 이름(예: dms-bucket-*yoursignin*)을 입력합니다.
4. DMS 마이그레이션 작업의 AWS 리전 를 선택합니다.
5. 나머지 설정을 그대로 유지하고 버킷 만들기를 선택합니다.

IAM 리소스 생성

DMS는 IAM 역할 및 정책을 사용하여 S3 버킷에 액세스하여 마이그레이션 전 평가 결과를 저장합니다.

IAM 정책을 생성하려면 다음을 수행합니다.

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 탐색 창에서 Policies를 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON 코드를 편집기에 붙여 넣으면서 예제 코드를 바꿉니다. *amzn-s3-demo-bucket*을 이전 섹션에서 생성한 Amazon S3 버킷의 이름으로 바꿉니다.

JSON

```
{
```

```

    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "s3:PutObject",
          "s3:DeleteObject",
          "s3:GetObject",
          "s3:PutObjectTagging"
        ],
        "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket/*"
        ]
      },
      {
        "Effect": "Allow",
        "Action": [
          "s3:ListBucket",
          "s3:GetBucketLocation"
        ],
        "Resource": [
          "arn:aws:s3:::amzn-s3-demo-bucket"
        ]
      }
    ]
  }
}

```

6. 다음: 태그를 선택한 후 다음: 검토를 선택합니다.
7. 이름*의 경우 **DMSPremigrationAssessmentS3Policy**를 입력한 후 정책 생성을 선택합니다.

IAM 역할을 생성하려면 다음을 수행합니다.

1. IAM 콘솔의 탐색 창에서 역할을 선택합니다.
2. 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 선택 페이지의 신뢰할 수 있는 엔터티 유형 아래에서 AWS 서비스를 선택합니다. 다른 AWS 서비스의 사용 사례에서 DMS를 선택합니다.
4. DMS 확인란을 선택한 다음 다음을 선택합니다.
5. 권한 추가 페이지에서 **DMSPremigrationAssessmentS3Policy**를 선택합니다. 다음을 선택합니다.
6. 이름 지정, 검토 및 생성 페이지에서 역할 이름에 **DMSPremigrationAssessmentS3Role**을 입력하고 역할 생성을 선택합니다.

7. 역할 페이지에서 역할 이름에 **DMSPremigrationAssessmentS3Role**을 입력합니다.
DMSPremigrationAssessmentS3Role을 선택합니다.
8. DMSPremigrationAssessmentS3Role 페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.
9. 신뢰 정책 편집 페이지에서 기존 텍스트를 대체하여 다음 JSON을 편집기에 붙여넣습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

이 정책은 DMS에 마이그레이션 전 평가 실행 결과를 S3 버킷에 넣을 수 있는 sts:AssumeRole 권한을 부여합니다.

10. 정책 업데이트를 선택합니다.

마이그레이션 전 평가 실행 지정, 시작, 보기

마이그레이션 전 평가는 신규 또는 기존 마이그레이션 태스크 구성을 기반으로 실행할 하나 이상의 개별 평가를 지정합니다. 개별 평가에서는 마이그레이션 유형, 지원되는 객체, 인덱스 구성, 기타 작업 설정(예: 마이그레이션할 스키마와 테이블을 식별하는 테이블 매핑) 등의 고려 사항에 따라 소스 또는 대상 데이터베이스의 특정 요소를 평가합니다. 예를 들어, 개별 평가에서는 마이그레이션할 수 있거나 없는 소스 데이터 유형 또는 프라이머리 키 포맷을 평가할 수 있습니다.

개별 평가 지정

새 평가 실행을 생성할 경우 태스크 구성에 적용할 수 있는 일부 또는 모든 개별 평가를 실행하도록 선택할 수 있습니다.

AWS DMS 는 다음과 같은 관계형 소스 및 대상 데이터베이스 엔진에 대한 사전 마이그레이션 평가 실행을 지원합니다.

- [Oracle 평가](#)
- [SQL Server 평가](#)
- [MySQL 평가](#)(MariaDB 및 Amazon Aurora MySQL - 호환 버전 포함)
- [PostgreSQL 평가](#)(Amazon Aurora PostgreSQL - 호환 버전 포함)
- [the section called “MariaDB 평가”](#)
- [the section called “Db2 LUW 평가”](#)

마이그레이션 전 평가 실행 시작 및 보기

AWS DMS Management Console, AWS CLI 및 AWS DMS API를 사용하여 신규 또는 기존 마이그레이션 작업에 대한 사전 마이그레이션 평가 실행을 시작할 수 있습니다.

마이그레이션 전 평가를 시작하려면 새 작업이나 기존 작업에 대해 평가를 실행하세요.

1. AWS DMS 관리 콘솔의 데이터베이스 마이그레이션 작업 페이지에서 다음 중 하나를 수행하세요.

- 새 작업을 생성하고 평가하려면 태스크 생성을 선택합니다. 데이터베이스 마이그레이션 작업 생성 페이지가 열립니다.
 1. 테이블 매핑을 포함하여 작업을 생성하는 데 필요한 작업 설정을 입력합니다.
 2. 마이그레이션 전 평가 섹션에서 마이그레이션 전 평가 실행 확인란을 선택합니다. 이 페이지에는 새 태스크에 대한 평가 실행을 지정하는 옵션이 포함되어 있습니다.

Note

새 작업을 생성할 때 마이그레이션 전 평가 실행을 활성화하면 작업 생성 시 작업을 자동으로 시작하는 옵션이 비활성화됩니다. 평가 실행이 완료된 후 수동으로 작업을 시작할 수 있습니다.

- 기존 태스크를 평가하려면 데이터베이스 마이그레이션 태스크 페이지에서 기존 태스크의 식별자를 선택합니다. 선택한 기존 작업의 작업 페이지가 열립니다.

1. 작업을 선택하고 마이그레이션 전 평가 생성을 선택합니다. 기존 작업에 대한 평가 실행을 지정하는 옵션이 포함된 프리미엄 평가 생성 페이지가 열립니다.
2. 평가 실행의 고유한 이름을 입력하거나 기본값을 그대로 둡니다.
3. 이 평가 실행에 포함하려는 사용 가능한 개별 평가를 선택합니다. 현재 작업 설정을 기반으로 사용 가능한 개별 평가만 선택할 수 있습니다. 기본적으로 사용 가능한 모든 개별 평가가 활성화되고 선택됩니다.
4. 계정에서 평가 결과 보고서를 저장할 Amazon S3 버킷과 폴더를 검색하고 선택합니다. 평가 실행을 위한 리소스 설정에 대한 자세한 내용은 [마이그레이션 전 평가를 위한 사전 조건 생성](#) 섹션을 참조하세요.
5. 선택한 Amazon S3 버킷 및 폴더에 대한 전체 계정 액세스 권한이 있는 IAM 역할을 선택하거나 입력합니다. 평가 실행을 위한 리소스 설정에 대한 자세한 내용은 [마이그레이션 전 평가를 위한 사전 조건 생성](#) 섹션을 참조하세요.
6. Amazon S3 버킷에 있는 평가 결과 보고서를 암호화하는 설정을 선택할 수 있습니다. S3 버킷 암호화에 대한 자세한 내용은 [Amazon S3 버킷에 대한 기본 서버 측 암호화 동작 설정](#)을 참조하세요.
7. 새 작업에 대해 작업 생성을 선택하거나 기존 작업에 대해 생성을 선택합니다.

데이터베이스 마이그레이션 작업 페이지가 열리고 새 작업이나 수정된 작업이 나열되며 상태는 생성 중...입니다. 그리고 해당 작업이 생성되면 마이그레이션 전 평가 실행이 시작된다는 배너 메시지도 함께 표시됩니다.

AWS DMS 는 AWS DMS 관리 콘솔 AWS CLI, 또는 AWS DMS API를 사용하여 최신 및 모든 이전 마이그레이션 전 평가 실행에 대한 액세스를 제공합니다.

평가 실행의 결과를 보려면

1. AWS DMS 관리 콘솔의 데이터베이스 마이그레이션 작업 페이지에서 기존 작업의 식별자를 선택합니다. 기존 작업의 작업 페이지가 열립니다.
2. 기존 작업 페이지에서 마이그레이션 전 평가 탭을 선택합니다. 그러면 해당 페이지의 마이그레이션 전 평가 섹션이 열리고 평가 실행 결과가 이름별로 역순으로 나열됩니다. 최신 결과가 목록 상단에 나타납니다. 결과를 조회할 평가 실행의 이름을 선택합니다.

이러한 평가 실행 결과는 최근 평가 실행의 이름과 그 상태에 관한 개요를 시작으로 하여 지정된 개별 평가 및 그 상태를 일련의 목록으로 보여줍니다. 그런 다음, 목록에서 평가 이름을 선택하여 개별 평가의 상태 세부 정보를 탐색할 수 있으며 평가 결과는 표의 열 수준까지 확인할 수 있습니다.

평가 실행의 상태 개요와 개별 평가에서는 상태 값이 모두 표시됩니다. 이 값은 평가 실행의 전체 상태와 개별 평가의 유사한 상태를 나타냅니다. 평가 실행의 상태 값을 목록으로 열거하면 다음과 같습니다.

- "cancelling" – 평가 실행이 취소되었습니다.
- "deleting" – 평가 실행이 삭제되었습니다.
- "failed" – failed 상태가 지정된 개별 평가가 하나 이상 완료되었습니다. 이 상태는 오류 조건을 포함한 다른 모든 상태보다 우선합니다.
- "error-provisioning" – 리소스가 프로비저닝되는 동안 (provisioning 상태 도중) 내부 오류가 발생했습니다. 프로비저닝 오류로 인해 검증 실패로 이어질 수 있는 평가가 실행되지 않았을 수 있으므로 개별 평가의 상태가 실패하지 않은 경우에만 이 상태가 할당됩니다.
- "error-executing" – 개별 평가를 실행하는 동안 (running 상태 도중) 내부 오류가 발생했습니다. 이 상태는 개별 평가의 상태가 실패하지 않은 경우에만 할당됩니다. 오류 조건으로 인해 검증 실패로 이어질 수 있는 평가를 완료하지 못했을 수 있기 때문입니다.
- "invalid state" – 평가 실행이 알 수 없는 상태입니다.
- "passed" - 모든 개별 평가가 실패, 경고 또는 오류 상태 없이 성공적으로 완료되었습니다.
- "provisioning" – 개별 평가를 실행하는 데 필요한 리소스가 프로비저닝되고 있습니다.
- "running" - 개별 평가가 실행 중입니다.
- "starting" – 평가 실행이 시작되었지만 개별 평가를 위한 리소스가 아직 프로비저닝되지 않았습니다.
- "warning" - warning 상태인 개별 평가가 하나 이상 완료되었으며 실패한 평가 또는 오류 상태가 없습니다.

평가 실행의 개별 평가에 대한 상태 값을 목록으로 열거하면 다음과 같습니다.

- "cancelled" – 평가 실행 취소의 일환으로 개별 평가가 취소되었습니다.
- "error" – 개별 평가가 성공적으로 완료되지 않았습니다.
- "failed" – 개별 평가가 성공적으로 완료되었지만 검증에 실패했다는 결과: 자세한 내용은 해당 결과의 세부 정보를 참조하세요.
- "invalid state" – 개별 평가의 상태가 알 수 없는 상태입니다.
- "passed" – 개별 평가가 완료되었고 성공적인 검증 결과를 얻었습니다.
- "pending" – 개별 평가가 실행 대기 중입니다.
- "running" – 개별 평가가 실행 중입니다.

- "warning" - 경고 상태로 완료된 개별 평가입니다.
- "skipped" - 평가 실행 중에 개별 평가를 건너뛰었습니다.

Amazon S3의 평가 실행 결과에 대한 JSON 파일도 볼 수 있습니다.

Amazon S3의 평가 실행에 대한 JSON 파일을 보려면

1. AWS DMS Management Console에서 평가 실행의 상태 개요에 표시된 Amazon S3 버킷 링크를 선택합니다. 그러면 버킷에 저장된 버킷 폴더 및 기타 Amazon S3 객체 목록이 표시됩니다. 결과가 버킷 폴더에 저장되어 있다면 폴더를 엽니다.
2. 여러 JSON 파일에서 평가 실행 결과를 찾을 수 있습니다. summary.json 파일에는 평가 실행의 전체 결과가 포함됩니다. 나머지 파일에는 평가 실행을 위해 지정된 개별 평가의 이름(예: unsupported-data-types-in-source.json)이 각각 지정됩니다. 각 파일에는 선택한 평가 실행의 해당 개별 평가 결과가 포함되어 있습니다.

기존 마이그레이션 작업에 대한 사전 마이그레이션 평가 실행의 결과를 시작하고 보려면 다음 CLI 명령 및 AWS DMS API 작업을 실행할 수 있습니다.

- CLI: [describe-applicable-individual-assessments](#), API: [DescribeApplicableIndividualAssessments](#) – 하나 이상의 작업 구성 파라미터가 주어지면 신규 마이그레이션 전 평가 실행에 대해 지정할 수 있는 개별 평가를 목록으로 보여줍니다.
- CLI: [start-replication-task-assessment-run](#), API: [StartReplicationTaskAssessmentRun](#) – 기존 마이그레이션 작업에 대한 하나 이상의 개별 평가에 대해 신규 마이그레이션 전 평가 실행을 시작합니다.
- CLI: [describe-replication-task-assessment-runs](#), API: [DescribeReplicationTaskAssessmentRuns](#) – 필터 설정에 따라 마이그레이션 전 실행의 페이지가 매겨진 목록을 반환합니다.
- CLI: [describe-replication-task-individual-assessments](#), API: [DescribeReplicationTaskIndividualAssessments](#) – 필터 설정에 따라 개별 평가의 페이지가 매겨진 목록을 반환합니다.
- CLI: [cancel-replication-task-assessment-run](#), API: [CancelReplicationTaskAssessmentRun](#) – 단일 마이그레이션 전 평가 실행을 취소하지만 삭제하지는 않습니다.

- CLI: [delete-replication-task-assessment-run](#), API: [DeleteReplicationTaskAssessmentRun](#) – 단일 마이그레이션 전 평가 실행의 기록을 삭제합니다.

개별 평가

이 섹션에서는 개별 마이그레이션 전 평가에 대해 설명합니다.

AWS DMS API를 사용하여 개별 사전 마이그레이션 평가를 생성하려면 [StartReplicationTaskAssessmentRun](#) 작업의 IncludeOnly 파라미터에 나열된 API 키를 사용합니다.

주제

- [모든 엔드포인트 유형에 대한 평가](#)
- [Oracle 평가](#)
- [SQL Server 평가](#)
- [MySQL 평가](#)
- [MariaDB 평가](#)
- [PostgreSQL 평가](#)
- [Db2 LUW 평가](#)

모든 엔드포인트 유형에 대한 평가

이 섹션에서는 모든 엔드포인트 유형에 대한 개별 마이그레이션 전 평가를 설명합니다.

주제

- [지원되지 않는 데이터 형식](#)
- [대형 객체\(LOB\)가 사용되지만 대상 LOB 열은 Null을 허용할 수 없음](#)
- [대형 객체\(LOB\)는 있지만 프라이머리 키나 고유 제약 조건은 없는 소스 테이블](#)
- [CDC 또는 전체 로드 및 CDC 태스크만을 위한 프라이머리 키가 없는 소스 테이블](#)
- [CDC 태스크만을 위한 프라이머리 키가 없는 대상 테이블](#)
- [지원되지 않는 소스 프라이머리 키 유형 - 복합 프라이머리 키](#)

지원되지 않는 데이터 형식

API 키: `unsupported-data-types-in-source`

DMS가 지원하지 않는 소스 엔드포인트의 데이터 유형을 확인합니다. 모든 데이터 형식을 엔진 간에 마이그레이션할 수 있는 것은 아닙니다.

대형 객체(LOB)가 사용되지만 대상 LOB 열은 Null을 허용할 수 없음

API 키: `full-lob-not-nullable-at-target`

복제가 전체 LOB 모드 또는 인라인 LOB 모드를 사용할 때 대상에서 LOB 열의 null 허용 여부를 확인합니다. DMS는 이러한 LOB 모드를 사용할 때 LOB 열이 null이어야 합니다. 이 평가를 수행하려면 소스 및 대상 데이터베이스가 관계형이어야 합니다.

대형 객체(LOB)는 있지만 프라이머리 키나 고유 제약 조건은 없는 소스 테이블

API 키: `table-with-lob-but-without-primary-key-or-unique-constraint`

LOB는 있지만 프라이머리 키나 고유 키는 없는 소스 테이블이 있는지 확인합니다. 테이블에는 DMS가 LOB를 마이그레이션하기 위한 프라이머리 키 또는 고유 키가 있어야 합니다. 이 평가를 수행하려면 소스 데이터베이스가 관계형이어야 합니다.

CDC 또는 전체 로드 및 CDC 태스크만을 위한 프라이머리 키가 없는 소스 테이블

API 키: `table-with-no-primary-key-or-unique-constraint`

전체 로드 및 변경 데이터 캡처(CDC) 마이그레이션 또는 CDC 전용 마이그레이션을 위해 소스 테이블에 프라이머리 키나 고유 키가 있는지 확인합니다. 프라이머리 키나 고유 키가 없으면 CDC 마이그레이션 중에 성능 문제가 발생할 수 있습니다. 이 평가를 수행하려면 소스 데이터베이스가 관계형이어야 하고 마이그레이션 유형에 CDC가 포함되어야 합니다.

CDC 태스크만을 위한 프라이머리 키가 없는 대상 테이블

API 키: `target-table-has-unique-key-or-primary-key-for-cdc`

CDC 전용 마이그레이션을 위해 이미 생성된 대상 테이블에 프라이머리 키 또는 고유 키가 있는지 확인합니다. 프라이머리 키나 고유 키가 없으면 DMS가 업데이트 적용 및 삭제 시 대상에서 전체 테이블 스캔이 발생할 수 있습니다. 이로 인해 CDC 마이그레이션 중에 성능 문제가 발생할 수 있습니다. 이 평가를 수행하려면 대상 데이터베이스가 관계형이어야 하며 마이그레이션 유형에는 CDC가 포함되어야 합니다.

지원되지 않는 소스 프라이머리 키 유형 - 복합 프라이머리 키

API 키: `unsupported-source-pk-type-for-elasticsearch-target`

Amazon OpenSearch Service로 마이그레이션할 때 소스 테이블에 복합 프라이머리 키가 있는지 확인합니다. 소스 테이블의 프라이머리 키는 단일 열로 구성되어야 합니다. 이 평가를 수행하려면 소스 데이터베이스가 관계형이어야 하고 대상 데이터베이스가 DynamoDB입니다.

Note

DMS는 소스 프라이머리 키가 여러 열로 구성된 OpenSearch Service 대상으로 소스 데이터베이스를 마이그레이션할 수 있도록 지원합니다.

Oracle 평가

Oracle을 소스로 사용할 때 권한에 대한 자세한 내용은 [용 자체 관리형 Oracle 소스에 필요한 사용자 계정 권한 AWS DMS](#) 또는 [용 AWS관리형 Oracle 소스에 필요한 사용자 계정 권한을 참조하세요 AWS DMS](#).

Note

이 섹션에서는 소스 또는 AWS DMS용 대상으로 Oracle을 사용하는 마이그레이션 작업에 대한 개별 마이그레이션 전 평가에 대해 설명합니다.

자체 관리형 Oracle 데이터베이스를 소스로 사용하는 경우 다음 권한 세트를 사용 AWS DMS 하세요.

```
grant select on gv_$parameter to dms_user;
grant select on v_$instance to dms_user;
grant select on v_$version to dms_user;
grant select on gv_$ASM_DISKGROUP to dms_user;
grant select on gv_$database to dms_user;
grant select on DBA_DB_LINKS to dms_user;
grant select on gv_$log_History to dms_user;
grant select on gv_$log to dms_user;
grant select on dba_types to dms_user;
grant select on dba_users to dms_user;
grant select on dba_directories to dms_user;
grant execute on SYS.DBMS_XMLGEN to dms_user;
```

자체 관리형 Oracle 데이터베이스를 AWS DMS Serverless의 소스로 사용하는 경우 추가 권한이 필요합니다.

```
grant select on dba_segments to dms_user;
```

```
grant select on v_$tablespace to dms_user;
grant select on dba_tab_subpartitions to dms_user;
grant select on dba_extents to dms_user;
```

AWS관리형 Oracle 데이터베이스를 소스로 사용하는 경우 다음 권한 세트를 AWS DMS사용합니다.

```
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('V_$PARAMETER', 'dms_user',
'SELECT');
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('V_$INSTANCE',
'dms_user', 'SELECT');
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('V_
$VERSION', 'dms_user', 'SELECT');
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('GV_
$ASM_DISKGROUP', 'dms_user', 'SELECT');
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('GV_
$DATABASE', 'dms_user', 'SELECT');
EXEC
RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('DBA_DB_LINKS', 'dms_user', 'SELECT');
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('GV_
$LOG_HISTORY', 'dms_user', 'SELECT');
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('GV_
$LOG', 'dms_user', 'SELECT');
EXEC
RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('DBA_TYPES', 'dms_user', 'SELECT');
EXEC
RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('DBA_USERS', 'dms_user', 'SELECT');
EXEC
RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('DBA_DIRECTORIES', 'dms_user',
'SELECT');
GRANT SELECT ON RDSADMIN.RDS_CONFIGURATION to dms_user;
GRANT EXECUTE ON SYS.DBMS_XMLGEN TO dms_user;
```

AWS관리형 Oracle 데이터베이스를 AWS DMS Serverless의 소스로 사용하는 경우 추가 권한이 필요합니다.

```
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('DBA_SEGMENTS', 'dms_user',
'SELECT');
EXEC
RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('DBA_TAB_SUBPARTITIONS', 'dms_user',
'SELECT');
```



```
EXEC
RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('DBA_EXTENTS','dms_user','SELECT');
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('V_
$TABLESPACE','dms_user','SELECT');
```

자체 관리형 Oracle 데이터베이스를 대상으로 사용하는 경우 다음 권한 세트를 AWS DMS사용합니다.

```
grant select on v_$instance to dms_user;
grant execute on SYS.DBMS_XMLGEN to dms_user;
```

AWS관리형 Oracle 데이터베이스를 대상으로 사용하는 경우 다음 권한 세트를 AWS DMS사용합니다.

```
EXEC RDSADMIN.RDSADMIN_UTIL.GRANT_SYS_OBJECT('V_$INSTANCE','dms_user',
'SELECT');
GRANT EXECUTE ON SYS.DBMS_XMLGEN TO dms_user;
```

주제

- [BatchApplyEnabled가 활성화된 경우에만 제한된 LOB 모드가 사용되는지 확인](#)
- [소스의 테이블에 숫자 데이터 유형에 대해 지정된 스케일이 없는 열이 있는지 검증](#)
- [대상 데이터베이스에서 트리거 검증](#)
- [소스에 Archivelog DEST_ID가 0으로 설정되어 있는지 검증](#)
- [전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 활성화되었는지 확인](#)
- [BatchApplyEnabled와 함께 DMS 태스크 범위에 사용되는 테이블에 999개 이상의 열이 있는지 검증](#)
- [데이터베이스 수준 보충 로깅 확인](#)
- [대기에 필요한 DB 링크가 생성되었는지 확인합니다.](#)
- [LOB 데이터 형식 및 바이너리 리더 구성 여부에 대한 Oracle 검증](#)
- [데이터베이스가 CDB인지 확인하세요.](#)
- [Oracle Database Edition을 확인하세요.](#)
- [DMS에 대한 Oracle CDC 방법을 검증하세요.](#)
- [DMS용 Oracle RAC 구성을 검증하세요.](#)
- [DMS 사용자에게 대상에 대한 권한이 있는지 검증](#)

- 모든 열에 보충 로깅이 필요한지 검증
- 기본 또는 고유 키를 사용하여 테이블에 추가 로깅이 활성화되어 있는지 검증
- SecureFile LOBs가 있고 태스크가 전체 LOB 모드로 구성되어 있는지 검증
- 태스크 범위에 포함된 테이블 내에서 함수 기반 인덱스가 사용되고 있는지 확인합니다.
- 태스크 범위에 포함된 테이블에 글로벌 임시 테이블이 사용되고 있는지 확인합니다.
- 오버플로 세그먼트가 있는 인덱스 구성 테이블이 태스크 범위에 포함된 테이블에 사용되고 있는지 확인합니다.
- 태스크 범위에 포함된 테이블에 다중 수준 중첩 테이블이 사용되는지 확인합니다.
- 태스크 범위에 포함된 테이블에 보이지 않는 열이 사용되는지 확인합니다.
- 태스크 범위에 포함된 테이블에 ROWID 열을 기반으로 구체화된 뷰가 사용되는지 확인합니다.
- Active Data Guard DML 리디렉션 기능이 사용되는지 확인합니다.
- 파티셔닝된 하이브리드 테이블이 사용되는지 확인합니다.
- 스키마 전용 Oracle 계정이 사용되는지 확인
- 가상 열이 사용되는지 검증
- 태스크 범위에 정의된 테이블 이름에 아포스트로피가 포함되어 있는지 확인합니다.
- 태스크 범위에 정의된 열에 XMLType, Long 또는 Long Raw 데이터 유형이 있는지 확인하고 태스크 설정에서 LOB 모드 구성을 확인합니다.
- 소스 Oracle 버전이에서 지원되는지 확인합니다 AWS DMS.
- 대상 Oracle 버전이에서 지원되는지 확인합니다 AWS DMS.
- DMS 사용자에게 데이터 검증을 사용하는 데 필요한 권한이 있는지 검증합니다.
- DMS 사용자에게 Oracle ASM과 함께 Binary Reader를 사용할 수 있는 권한이 있는지 확인합니다.
- DMS 사용자에게 Oracle ASM과 함께 Binary Reader를 사용할 수 있는 권한이 있는지 확인합니다.
- DMS 사용자에게 CopyToTempFolder 메서드와 함께 Binary Reader를 사용할 수 있는 권한이 있는지 검증
- DMS 사용자에게 Oracle Standby를 소스로 사용할 수 있는 권한이 있는지 검증
- DMS 소스가 애플리케이션 컨테이너 PDB에 연결되어 있는지 검증
- 테이블에 태스크 범위에 포함된 XML 데이터 유형이 있는지 확인합니다.
- 소스 데이터베이스에서 Archivelog 모드가 활성화되어 있는지 확인합니다.
- RDS Oracle의 아카이브로그 보존을 검증합니다.
- 테이블에 태스크 범위에 포함된 확장 데이터 유형이 있는지 확인합니다.

- [태스크 범위에 포함된 객체 이름의 길이를 확인합니다.](#)
- [DMS 소스가 Oracle PDB에 연결되어 있는지 확인](#)
- [테이블에 태스크 범위에 공간 열이 포함되어 있는지 확인합니다.](#)
- [DMS 소스가 Oracle Standby에 연결되어 있는지 확인합니다.](#)
- [소스 데이터베이스 테이블스페이스가 TDE를 사용하여 암호화되었는지 확인합니다.](#)
- [소스 데이터베이스가 자동 스토리지 관리\(ASM\)를 사용하는지 검증합니다.](#)
- [배치 적용이 활성화되어 있는지, 대상 Oracle 데이터베이스의 테이블에 테이블 또는 인덱스 수준에서 병렬 처리가 활성화되어 있는지 확인합니다.](#)
- [작업 범위의 테이블을 검증하여 “벌크 배열 크기” 파라미터 권장](#)
- [HandleCollationDiff 작업 설정이 구성되었는지 확인](#)
- [DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있고 해당 상태가 VALID인지 검증합니다](#)
- [Binary Reader가 Oracle Standby에 소스로 사용되는지 확인](#)
- [AWS DMS 사용자에게 Oracle RDS Standby 데이터베이스에서 데이터를 복제하는 데 필요한 디렉터리 권한이 있는지 확인합니다.](#)
- [복제에 사용되는 Oracle Standby 유형 검증](#)
- [RDS Oracle 대기에 필요한 디렉터리가 생성되었는지 확인](#)
- [배치 적용 대상에 프라이머리 키 또는 고유 인덱스가 있는지 확인](#)
- [배치 적용 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 확인](#)
- [지원되지 않는 HCC 수준이 전체 로드에서 사용되는지 확인](#)
- [CDC를 사용한 전체 로드에서 지원되지 않는 HCC 수준이 사용되는지 검증](#)
- [CDC에 지원되지 않는 HCC 압축이 사용되는지 검증](#)
- [소스 압축 방법을 기반으로 한 CDC 권장 사항](#)
- [배치 적용이 활성화되어 있는지 확인하고 테이블에 999개 이상의 열이 있는지 확인합니다](#)
- [Digits Randomize 변환 규칙 확인](#)
- [Digits Mask에 대한 변환 규칙 확인](#)
- [해싱 마스크에 대한 변환 규칙 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인](#)

- [스트리밍 대상에 대한 복제에 LOBs 또는 확장 데이터 유형 열이 포함되어 있지 않은지 확인](#)
- [CDC 전용 작업이 OpenTransactionWindow 엔드포인트 설정을 사용하도록 구성되어 있는지 확인합니다.](#)
- [선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.](#)
- [마이그레이션을 위해 대상 외래 키 제약 조건이 비활성화되었는지 확인](#)
- [Oracle 데이터베이스와 AWS DMS 버전이 호환되는지 확인](#)
- [보조 제약 조건 및 인덱스\(비기본\)가 소스 데이터베이스에 있는지 검증합니다.](#)
- [세션 제한 시간 설정\(IDLE_TIME\)이 로 설정되어 있는지 확인합니다. UNLIMITED](#)
- [AWS DMS 사용자에게 소스 데이터베이스에 대한 모든 필수 권한이 있는지 확인](#)
- [Oracle LogMiner를 사용할 때 XMLTYPE 또는 LOB 열이 존재하는지 확인](#)
- [대상 엔드포인트가 읽기 전용 복제본이 아닌지 확인](#)
- [직접 경로 로드를 사용할 때 Oracle 대상에 CONTEXT 인덱스가 없는지 확인](#)
- [기존 LOB 열에서 제한된 LOB 모드를 사용할 때 FailTasksOnLobTruncation가 활성화되었는지 확인](#)
- [EnableHomogenousPartitionOps 엔드포인트 설정이 활성화되었는지 확인](#)

BatchApplyEnabled가 활성화된 경우에만 제한된 LOB 모드가 사용되는지 확인

API 키: `oracle-batch-apply-lob-mode`

이 마이그레이션 전 평가는 DMS 작업의 테이블에 LOB 열이 포함되어 있는지 확인합니다. LOB 열이 작업 범위에 포함된 경우 제한된 LOB 모드와 함께 BatchApplyEnabled를 사용해야 합니다.

자세한 내용은 [대상 메타데이터 태스크 설정](#)을 참조하세요.

소스의 테이블에 숫자 데이터 유형에 대해 지정된 스케일이 없는 열이 있는지 검증

API 키: `oracle-number-columns-without-scale`

이 마이그레이션 전 평가는 DMS 태스크에 스케일이 지정되지 않은 NUMBER 데이터 유형의 열이 포함되어 있는지 확인합니다. 엔드포인트 설정 NumberDataTypeScale을 평가 보고서에 지정된 값으로 설정하는 것이 좋습니다.

자세한 내용은 [Oracle을 AWS DMS의 소스로 사용할 때 엔드포인트 설정](#)을 참조하세요.

대상 데이터베이스에서 트리거 검증

API 키: `oracle-target-triggers-are-enabled`

이 마이그레이션 전 평가는 대상 데이터베이스에서 트리거가 활성화되었는지 여부를 검증합니다. 트리거가 활성화된 경우 평가가 실패합니다. 마이그레이션 중에 트리거를 비활성화하거나 제거하는 것이 좋습니다.

자세한 내용은 [DMS 모범 사례 참조](#)를 참조하세요.

소스에 Archivelog **DEST_ID**가 0으로 설정되어 있는지 검증

API 키: oracle-zero-archive-log-dest-id

이 마이그레이션 전 평가는 아카이브된 로그 useZeroDestid=true가 0으로 설정된 경우 소스에 대한 엔드포인트 추가 연결 속성 DEST_ID가 설정되었는지 여부를 검증합니다.

자세한 내용은 [장애 조치 시나리오에서 Oracle 데이터베이스와 함께 사용할 때 AWS DMS 복제를 처리하는 방법을](#) 참조하세요.

전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 활성화되었는지 확인

API 키: oracle-check-secondary-indexes

이 마이그레이션 전 평가는 대상 데이터베이스의 전체 로드 중에 보조 인덱스가 활성화되었는지 여부를 검증합니다. 전체 로드 중에 보조 인덱스를 비활성화하거나 제거하는 것이 좋습니다.

자세한 내용은 [AWS Database Migration Service에 대한 모범 사례](#)를 참조하세요.

BatchApplyEnabled와 함께 DMS 태스크 범위에 사용되는 테이블에 999개 이상의 열이 있는지 검증

API 키: oracle-batch-apply-lob-999

배치 최적화 적용 모드가 활성화된 테이블은 총 999개 이상의 열을 가질 수 없습니다. 열이 999개 이상인 테이블은 AWS DMS가 배치를 하나씩 처리하여 지연 시간을 늘립니다. DMS는 수식 $2 * \text{columns\_in\_original\_table} + \text{columns\_in\_primary\_key} \leq 999$ 를 사용하여 배치 최적화 적용 모드에서 지원되는 테이블당 총 열 수를 계산합니다.

자세한 내용은 [AWS Database Migration Service에 대한 대상으로 Oracle에 대한 제한 사항](#)을 참조하세요.

데이터베이스 수준 보충 로깅 확인

API 키: oracle-supplemental-db-level

이 마이그레이션 전 평가는 데이터베이스 수준에서 최소 보충 로깅이 활성화되는지 여부를 검증합니다. Oracle 데이터베이스를 마이그레이션 소스로 사용하려면 보충 로깅을 활성화해야 합니다.

보충 로깅을 활성화하려면 다음 쿼리를 사용하세요.

```
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA
```

자세한 내용은 [보충 로깅 설정](#) 단원을 참조하십시오.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

대기에 필요한 DB 링크가 생성되었는지 확인합니다.

API 키: oracle-validate-standby-dblink

이 마이그레이션 전 평가는 Oracle 대기 데이터베이스 소스에 대해 Dblink가 생성되었는지 여부를 검증합니다. AWS_DMS_DBLINK는 대기 데이터베이스를 원본으로 사용하기 위한 전제 조건입니다. Oracle Standby를 소스로 사용하는 경우, AWS DMS 는 기본적으로 열린 트랜잭션의 유효성을 검사하지 않습니다.

자세한 내용은 [의 소스로 자체 관리형 Oracle 데이터베이스 작업 AWS DMS](#) 단원을 참조하십시오.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

LOB 데이터 형식 및 바이너리 리더 구성 여부에 대한 Oracle 검증

API 키: oracle-binary-lob-source-validation

이 마이그레이션 전 평가는 Oracle 데이터베이스 엔드포인트 버전 12c 또는 이후 버전에서 Oracle LogMiner를 사용하는지 여부를 검증합니다. AWS DMS 는 Oracle 데이터베이스 버전 12c에서 LOB 열을 마이그레이션하는 경우 Oracle LogMiner를 지원하지 않습니다. 또한 이 평가에서는 LOB 열이 있는지 확인하고 적절한 권장 사항을 제공합니다.

Oracle LogMiner를 사용하지 않도록 마이그레이션을 구성하려면 소스 엔드포인트에 다음 구성을 추가하세요.

```
useLogMinerReader=N;useBfile=Y;
```

자세한 내용은 [CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용](#) 단원을 참조하십시오.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

데이터베이스가 CDB인지 확인하세요.

API 키: `oracle-validate-cdb`

이 마이그레이션 전 평가는 데이터베이스가 컨테이너 데이터베이스인지 검증합니다. AWS DMS 는 멀티테넌트 컨테이너 루트 데이터베이스(CDB\$ROOT)는 지원하지 않습니다.

 Note

이 평가는 Oracle 12.1.0.1 또는 이후 버전에만 필요합니다. 이 평가는 12.1.0.1 이전의 Oracle 버전에는 적용되지 않습니다.

자세한 내용은 [Oracle을의 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

Oracle Database Edition을 확인하세요.

API 키: `oracle-check-cdc-support-express-edition`

이 마이그레이션 전 평가는 Oracle 소스 데이터베이스가 Express Edition인지 검증합니다. AWS DMS 는 Oracle Express Edition(Oracle Database XE) 버전 18.0 및 이후 버전에 대한 CDC는 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

DMS에 대한 Oracle CDC 방법을 검증하세요.

API 키: `oracle-recommendation-cdc-method`

이 사전 마이그레이션 평가는 지난 7일 동안 다시 실행 로그 생성을 검증하고 CDC에 AWS DMS Binary Reader 또는 Oracle LogMiner를 사용할지 여부를 권장합니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

사용할 CDC 방법을 결정하는 방법에 관한 자세한 내용은 [CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용](#) 섹션을 참조하세요.

DMS용 Oracle RAC 구성을 검증하세요.

API 키: `oracle-check-rac`

이 마이그레이션 전 평가는 Oracle 데이터베이스가 Real Application Cluster인지 여부를 검증합니다. Real Application Cluster 데이터베이스를 올바르게 구성해야 합니다. 데이터베이스가 RAC를 기반으로 하는 경우 Oracle LogMiner 대신 CDC용 AWS DMS Binary Reader를 사용하는 것이 좋습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 내용은 [CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용](#) 단원을 참조하십시오.

DMS 사용자에게 대상에 대한 권한이 있는지 검증

API 키: `oracle-validate-permissions-on-target`

이 마이그레이션 전 평가는 DMS 사용자에게 대상 데이터베이스에 필요한 모든 권한이 있는지 확인합니다.

모든 열에 보충 로깅이 필요한지 검증

API 키: `oracle-validate-supplemental-logging-all-columns`

이 마이그레이션 전 평가는 작업 범위에 언급된 테이블에 대해 프라이머리 키 또는 고유 키 없이 테이블의 모든 열에 추가 로깅이 추가되었는지 확인합니다. 기본 또는 고유 키가 없는 테이블의 모든 열에 대한 추가 로깅이 없으면 데이터의 전후 이미지를 다시 실행 로그에서 사용할 수 없습니다. DMS는 DML 문을 생성하려면 기본 또는 고유 키가 없는 테이블에 대한 추가 로깅이 필요합니다.

기본 또는 고유 키를 사용하여 테이블에 추가 로깅이 활성화되어 있는지 검증

API 키: `oracle-validate-supplemental-logging-for-pk`

이 마이그레이션 전 평가는 프라이머리 키 또는 고유 인덱스가 있는 테이블에 대해 추가 로깅이 활성화되어 있는지 여부를 검증하고 엔드포인트 수준에서 `AddSupplementalLogging`이 활성화되어 있는지도 확인합니다. DMS가 변경 사항을 복제할 수 있도록 하려면 프라이머리 키 또는 고유 키를 기반으로 테이블 수준에서 추가 로깅을 수동으로 추가하거나 복제된 테이블에 ALTER 권한이 있는 DMS 사용자를 사용하여 엔드포인트 설정 `AddSupplementalLogging = true`를 활용할 수 있습니다.

SecureFile LOBs가 있고 태스크가 전체 LOB 모드로 구성되어 있는지 검증

API 키: `oracle-validate-securefile-lobs`

이 마이그레이션 전 평가에서는 작업 범위 내의 테이블에 SecureFile LOBs 있는지 확인하고 해당 LOB 설정을 확인합니다. 전체 LOB 모드에서 태스크를 실행하면 성능이 느려질 수 있으므로 성능을 향상시키기 위해 별도의 태스크에 LOB 테이블을 할당하는 것이 좋습니다.

태스크 범위에 포함된 테이블 내에서 함수 기반 인덱스가 사용되고 있는지 확인합니다.

API 키: `oracle-validate-function-based-indexes`

이 마이그레이션 전 평가에서는 태스크 범위 내의 테이블에 함수 기반 인덱스가 있는지 확인합니다. AWS DMS 는 함수 기반 인덱스 복제를 지원하지 않습니다. 대상 데이터베이스에서 마이그레이션한 후 인덱스를 생성하는 것이 좋습니다.

태스크 범위에 포함된 테이블에 글로벌 임시 테이블이 사용되고 있는지 확인합니다.

API 키: `oracle-validate-global-temporary-tables`

이 마이그레이션 전 평가는 글로벌 임시 테이블이 작업 테이블 매핑 범위 내에서 사용되는지 확인합니다. AWS DMS 는 글로벌 임시 테이블 마이그레이션 또는 복제를 지원하지 않습니다.

오버플로 세그먼트가 있는 인덱스 구성 테이블이 태스크 범위에 포함된 테이블에 사용되고 있는지 확인합니다.

API 키: `oracle-validate-iot-overflow-segments`

오버플로 세그먼트가 있는 인덱스 구성 테이블이 작업 범위에 포함된 테이블에 사용되고 있는지 확인합니다. 오버플로 세그먼트가 있는 인덱스 구성 테이블에 대해 CDC를 지원하지 AWS DMS 않습니다.

태스크 범위에 포함된 테이블에 다중 수준 중첩 테이블이 사용되는지 확인합니다.

API 키: `oracle-validate-more-than-one-nesting-table-level`

이 마이그레이션 전 평가는 작업 범위에 사용되는 중첩 테이블의 중첩 수준을 확인합니다. AWS DMS 는 테이블 중첩의 한 수준만 지원합니다.

태스크 범위에 포함된 테이블에 보이지 않는 열이 사용되는지 확인합니다.

API 키: `oracle-validate-invisible-columns`

이 마이그레이션 전 평가는 작업 범위에 사용되는 테이블에 보이지 않는 열이 있는지 확인합니다. AWS DMS 는 소스 데이터베이스의 보이지 않는 열에서 데이터를 마이그레이션하지 않습니다. 보이지 않는 열을 마이그레이션하려면 열이 표시되도록 수정해야 합니다.

태스크 범위에 포함된 테이블에 ROWID 열을 기반으로 구체화된 뷰가 사용되는지 확인합니다.

API 키: `oracle-validate-rowid-based-materialized-views`

이 사전 마이그레이션 평가는 마이그레이션에 사용되는 구체화된 뷰가 ROWID 열을 기반으로 생성되는지 확인합니다. AWS DMS 는 ROWID 데이터 유형 또는 ROWID 열을 기반으로 구체화된 뷰를 지원하지 않습니다.

Active Data Guard DML 리디렉션 기능이 사용되는지 확인합니다.

API 키: `oracle-validate-adg-redirect-dml`

이 마이그레이션 전 평가는 Active Data Guard DML 리디렉션 기능이 사용되는지 여부를 검증합니다. Oracle 19.0을 소스로 사용하는 경우 Data Guard DML 리디렉션 기능을 지원하지 AWS DMS 않습니다.

파티셔닝된 하이브리드 테이블이 사용되는지 확인합니다.

API 키: `oracle-validate-hybrid-partitioned-tables`

이 마이그레이션 전 평가는 파티셔닝된 하이브리드 테이블이 태스크 범위에 정의된 테이블에 사용되는지 여부를 검증합니다.

스키마 전용 Oracle 계정이 사용되는지 확인

API 키: `oracle-validate-schema-only-accounts`

이 마이그레이션 전 평가는 스키마 전용 계정이 태스크 범위 내에 있는지 확인합니다.

가상 열이 사용되는지 검증

API 키: `oracle-validate-virtual-columns`

이 마이그레이션 전 평가는 Oracle 인스턴스에 태스크 범위 내의 테이블에 가상 열이 있는지 확인합니다.

태스크 범위에 정의된 테이블 이름에 아포스트로피가 포함되어 있는지 확인합니다.

API 키: `oracle-validate-names-with-apostrophes`

이 마이그레이션 전 평가는 태스크 범위에 사용되는 테이블에 아포스트로피가 포함되어 있는지 확인합니다. AWS DMS 는 아포스트로피가 포함된 이름으로 테이블을 복제하지 않습니다. 식별된 경우 해

당 테이블의 이름을 바꾸는 것이 좋습니다. 또는 이러한 테이블을 로드하기 위해 아포스트로피 없는 뷰 또는 구체화된 뷰를 생성할 수 있습니다.

태스크 범위에 정의된 열에 **XMLType**, **Long** 또는 **Long Raw** 데이터 유형이 있는지 확인하고 태스크 설정에서 LOB 모드 구성을 확인합니다.

API 키: `oracle-validate-limited-lob-mode-for-longs`

이 사전 마이그레이션 평가는 작업 범위에 정의된 테이블에 데이터 형식이 XMLType, Long 또는 인지 확인하고 작업 설정이 제한된 크기의 LOB 모드를 사용하도록 구성되어 있는지 Long Raw 확인합니다. AWS DMS 는 FULL LOB 모드를 사용하여 이러한 데이터 형식의 복제를 지원하지 않습니다. 이러한 데이터 유형으로 테이블을 식별할 때 제한된 크기 LOB 모드를 사용하도록 태스크 설정을 변경하는 것이 좋습니다.

소스 Oracle 버전이에서 지원되는지 확인합니다 AWS DMS.

API 키: `oracle-validate-supported-versions-of-source`

이 사전 마이그레이션 평가는 소스 Oracle 인스턴스 버전이에서 지원되는지 확인합니다 AWS DMS.

대상 Oracle 버전이에서 지원되는지 확인합니다 AWS DMS.

API 키: `oracle-validate-supported-versions-of-target`

이 사전 마이그레이션 평가는 대상 Oracle 인스턴스 버전이에서 지원되는지 확인합니다 AWS DMS.

DMS 사용자에게 데이터 검증을 사용하는 데 필요한 권한이 있는지 검증합니다.

API 키: `oracle-prerequisites-privileges-of-validation-feature`

이 마이그레이션 전 평가는 DMS 사용자에게 DMS 데이터 검증을 사용하는 데 필요한 권한이 있는지 확인합니다. 데이터 검증을 사용하지 않으려는 경우 이 검증 활성화를 무시할 수 있습니다.

DMS 사용자에게 Oracle ASM과 함께 Binary Reader를 사용할 수 있는 권한이 있는지 확인합니다.

API 키: `oracle-prerequisites-privileges-of-binary-reader-asm`

이 마이그레이션 전 평가는 DMS 사용자에게 Oracle ASM 인스턴스에서 Binary Reader를 사용하는 데 필요한 권한이 있는지 확인합니다. 소스가 Oracle ASM 인스턴스가 아니거나 CDC용 Binary Reader를 사용하지 않는 경우 이 평가 활성화를 무시할 수 있습니다.

DMS 사용자에게 Oracle ASM과 함께 Binary Reader를 사용할 수 있는 권한이 있는지 확인합니다.

API 키: `oracle-prerequisites-privileges-of-binary-reader-non-asm`

이 마이그레이션 전 평가는 DMS 사용자에게 Oracle 비ASM 인스턴스에서 Binary Reader를 사용하는 데 필요한 권한이 있는지 확인합니다. 이 평가는 Oracle 비ASM 인스턴스가 있는 경우에만 유효합니다.

DMS 사용자에게 CopyToTempFolder 메서드와 함께 Binary Reader를 사용할 수 있는 권한이 있는지 검증

API 키: `oracle-prerequisites-privileges-of-binary-reader-copy-to-temp-folder`

이 마이그레이션 전 평가는 DMS 사용자에게 '임시 폴더에 복사' 메서드와 함께 Binary Reader를 사용하는 데 필요한 권한이 있는지 확인합니다. 이 평가는 CopyToTempFolder를 사용하여 Binary Reader를 사용하는 동안 CDC 변경 사항을 읽고 소스에 연결된 ASM 인스턴스가 있는 경우에만 관련이 있습니다. CopyToTempFolder 기능을 사용하지 않으려는 경우 이 평가 활성화를 무시할 수 있습니다.

CopyToTempFolder 기능은 더 이상 사용되지 않으므로 사용하지 않는 것이 좋습니다.

DMS 사용자에게 Oracle Standby를 소스로 사용할 수 있는 권한이 있는지 검증

API 키: `oracle-prerequisites-privileges-of-standby-as-source`

이 마이그레이션 전 평가는 DMS 사용자에게 StandBy Oracle 인스턴스를 소스로 사용하는 데 필요한 권한이 있는지 확인합니다. StandBy Oracle 인스턴스를 소스로 사용하지 않으려는 경우 이 평가 활성화를 무시할 수 있습니다.

DMS 소스가 애플리케이션 컨테이너 PDB에 연결되어 있는지 검증

API 키: `oracle-check-app-pdb`

이 마이그레이션 전 평가는 DMS 소스가 애플리케이션 컨테이너 PDB에 연결되어 있는지 확인합니다. DMS는 애플리케이션 컨테이너 PDB에서의 복제를 지원하지 않습니다.

테이블에 태스크 범위에 포함된 XML 데이터 유형이 있는지 확인합니다.

API 키: `oracle-check-xml-columns`

이 마이그레이션 전 평가는 태스크 범위에 사용되는 테이블에 XML 데이터 유형이 있는지 확인합니다. 또한 테이블에 XML 데이터 유형이 포함된 경우 태스크가 제한된 LOB 모드로 구성되어 있는지 확인합니다. DMS는 Oracle XML 열 마이그레이션을 위한 제한된 LOB 모드만 지원합니다.

소스 데이터베이스에서 Archivelog 모드가 활성화되어 있는지 확인합니다.

API 키: `oracle-check-archivelog-mode`

이 마이그레이션 전 평가는 소스 데이터베이스에서 Archivelog 모드가 활성화되었는지 여부를 검증합니다. DMS가 변경 사항을 복제하려면 소스 데이터베이스에서 아카이브 로그 모드를 활성화해야 합니다.

RDS Oracle의 아카이브로그 보존을 검증합니다.

API 키: `oracle-check-archivelog-retention-rds`

이 마이그레이션 전 평가는 RDS Oracle 데이터베이스의 Archivelog 보존이 최소 24시간 동안 구성되어 있는지 확인합니다.

테이블에 태스크 범위에 포함된 확장 데이터 유형이 있는지 확인합니다.

API 키: `oracle-check-extended-columns`

이 마이그레이션 전 평가는 태스크 범위에 사용되는 테이블에 확장 데이터 유형이 있는지 확인합니다. 확장 데이터 유형은 DMS 버전 3.5 이상에서만 지원됩니다.

태스크 범위에 포함된 객체 이름의 길이를 확인합니다.

API 키: `oracle-check-object-30-bytes-limit`

이 마이그레이션 전 평가는 객체 이름의 길이가 30바이트를 초과하는지 확인합니다. DMS는 긴 객체 이름(30바이트 이상)을 지원하지 않습니다.

DMS 소스가 Oracle PDB에 연결되어 있는지 확인

API 키: `oracle-check-pdb-enabled`

이 마이그레이션 전 평가는 DMS 소스가 PDB에 연결되어 있는지 확인합니다. DMS는 Oracle PDB와 함께 Binary Reader를 소스로 사용하는 경우에만 CDC를 지원합니다. 또한 DMS가 Oracle PDB에 연결될 때 작업이 Binary Reader를 사용하도록 구성되어 있는지 평가합니다.

테이블에 태스크 범위에 공간 열이 포함되어 있는지 확인합니다.

API 키: `oracle-check-spatial-columns`

이 마이그레이션 전 평가는 테이블에 태스크 범위에 포함된 공간 열이 있는지 확인합니다. DMS는 전체 LOB 모드를 사용하는 공간 데이터 유형만 지원합니다. 또한 DMS가 공간 열을 식별할 때 태스크가 전체 LOB 모드를 사용하도록 구성되어 있는지 여부를 평가합니다.

DMS 소스가 Oracle Standby에 연결되어 있는지 확인합니다.

API 키: `oracle-check-standby-db`

이 마이그레이션 전 평가는 소스가 Oracle Standby에 연결되어 있는지 확인합니다. DMS는 Oracle Standby와 함께 Binary Reader를 소스로 사용할 때만 CDC를 지원합니다. 또한 DMS가 Oracle Standby에 연결될 때 작업이 이진 리더를 사용하도록 구성되어 있는지 평가합니다.

소스 데이터베이스 테이블스페이스가 TDE를 사용하여 암호화되었는지 확인합니다.

API 키: `oracle-check-tde-enabled`

이 마이그레이션 전 평가는 소스에 테이블스페이스에서 활성화된 TDE 암호화가 있는지 확인합니다. DMS는 Oracle LogMiner for RDS Oracle을 사용할 때 암호화된 테이블스페이스에서만 TDE를 지원합니다.

소스 데이터베이스가 자동 스토리지 관리(ASM)를 사용하는지 검증합니다.

API 키: `oracle-check-asm`

이 사전 마이그레이션 평가는 소스 데이터베이스가 ASM을 사용하는지 여부를 감지합니다. 최적의 성능을 위해 엔드포인트 설정에서 `parallelASMReadThreads` 및 `readAheadBlocks` 파라미터를 사용하여 Binary Reader를 구성합니다. 이러한 설정은 ASM 스토리지로 작업할 때 데이터 추출 성능을 향상시킵니다.

자세한 내용은 [CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용](#) 단원을 참조하십시오.

배치 적용이 활성화되어 있는지, 대상 Oracle 데이터베이스의 테이블에 테이블 또는 인덱스 수준에서 병렬 처리가 활성화되어 있는지 확인합니다.

API 키: `oracle-check-degree-of-parallelism`

AWS DMS 는 대상 데이터베이스의 테이블에 병렬 처리가 활성화되어 있는지 확인합니다. 대상 데이터베이스에서 병렬 처리를 활성화하면 배치 프로세스가 실패합니다. 따라서 배치 적용 기능을 사용할 때는 테이블 또는 인덱스 수준에서 병렬 처리를 비활성화해야 합니다.

작업 범위의 테이블을 검증하여 “벌크 배열 크기” 파라미터 권장

API 키: `oracle-check-bulk-array-size`

이 평가에서는 작업 범위 내에 LOB(Large Object) 데이터 형식이 있는 테이블이 없는 경우 BulkArraySize ECA(Extra Connection Attribute)를 설정하는 것이 좋습니다. BulkArraySize ECA를 설정하면 마이그레이션의 전체 로드 단계의 성능이 향상될 수 있습니다. 소스/대상 엔드포인트의 ECA를 사용하여 대량 배열 크기를 설정하여 마이그레이션의 전체 로드 단계에서 최적의 성능을 얻을 수 있습니다.

HandleCollationDiff 작업 설정이 구성되었는지 확인

API 키: oracle-check-handlecollationdiff

이 평가는 DMS 작업이 검증을 위해 구성되어 있는지 검증하고 Oracle과 PostgreSQL 간의 데이터를 검증하는 동안 잘못된 검증 결과를 방지하기 위해 HandleCollationDiff 작업 설정을 권장합니다.

자세한 내용은 [the section called “데이터 검증 작업 설정”](#) 단원을 참조하십시오.

DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있고 해당 상태가 VALID인지 검증합니다

API 키: oracle-check-pk-validity

데이터를 검증하려면 테이블에 프라이머리 키나 고유한 인덱스가 있어야 합니다.

자세한 내용은 [데이터 유효성 검사](#) 단원을 참조하십시오.

Binary Reader가 Oracle Standby에 소스로 사용되는지 확인

API 키: oracle-check-binary-reader

이 평가는 소스 데이터베이스가 대기 데이터베이스이고 변경 데이터 캡처용 이진 리더(CDC)를 사용하고 있는지 확인합니다.

자세한 내용은 [the section called “Oracle을 소스로 사용”](#) 단원을 참조하십시오.

AWS DMS 사용자에게 Oracle RDS Standby 데이터베이스에서 데이터를 복제하는 데 필요한 디렉터리 권한이 있는지 확인합니다.

API 키: oracle-check-directory-permissions

이 평가는 소스 데이터베이스가 Oracle RDS Standby일 때 AWS DMS 사용자에게 ARCHIVELOG_DIR_% 및 ONLINELOG_DIR_% 디렉터리에 대한 필수 읽기 권한이 있는지 확인합니다.

자세한 내용은 [the section called “AWS관리형 Oracle 소스 작업”](#) 단원을 참조하십시오.

복제에 사용되는 Oracle Standby 유형 검증

API 키: oracle-check-physical-standby-with-apply

이 평가는 AWS DMS 복제에 사용되는 Oracle 대기 데이터베이스의 유형을 검증합니다.는 다시 실행 로그가 자동으로 적용되는 읽기 전용 모드에서 열어야 하는 물리적 대기 데이터베이스 AWS DMS 만 지원합니다. AWS DMS 는 복제를 위한 스냅샷 또는 논리적 대기 데이터베이스를 지원하지 않습니다.

자세한 내용은 [the section called “Binary Reader와 함께 Oracle Standby를 소스로 사용”](#) 단원을 참조하십시오.

RDS Oracle 대기에 필요한 디렉터리가 생성되었는지 확인

API 키: `oracle-check-rds-standby-directories`

이 평가는 RDS Standby 인스턴스의 아카이브 로그 및 온라인 로그에 필요한 Oracle 디렉터리가 생성되었는지 확인합니다.

자세한 내용은 [the section called “CDC용 Binary Reader와 함께 Oracle Standby 소스 사용”](#) 단원을 참조하십시오.

배치 적용 대상에 프라이머리 키 또는 고유 인덱스가 있는지 확인

API 키: `oracle-check-batch-apply-target-pk-ui-absence`

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 동시에 존재하는 테이블은 배치가 실패하며 변경 사항이 하나씩 처리됩니다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키를 생성할 수 있습니다.

자세한 내용은 [the section called “Oracle을 대상으로 사용”](#) 단원을 참조하십시오.

배치 적용 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 확인

API 키: `oracle-check-batch-apply-target-pk-ui-simultaneously`

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 동시에 있는 테이블은 배치 처리 시 실패하며, 변경 사항은 하나씩 처리됩니다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키(들) 또는 프라이머리 키를 삭제하고 마이그레이션을 수행하는 경우 다시 빌드할 수 있습니다.

자세한 내용은 [the section called “Oracle을 대상으로 사용”](#) 단원을 참조하십시오.

지원되지 않는 HCC 수준이 전체 로드에서 사용되는지 확인

API 키: `oracle-check-binary-reader-hcc-full-load`

Oracle 소스 엔드포인트가 Binary Reader를 사용하도록 구성된 경우, HCC 압축 방법의 Query Low 수준은 전체 로드 작업에 대해서만 지원됩니다.

자세한 내용은 [the section called “소스로서의 Oracle에 대한 압축 방법”](#) 단원을 참조하십시오.

CDC를 사용한 전체 로드에는 지원되지 않는 HCC 수준이 사용되는지 검증

API 키: `oracle-check-binary-reader-hcc-full-load-and-cdc`

Oracle 소스 엔드포인트가 Binary Reader를 사용하도록 구성된 경우 Query low 모드의 HCC는 전체 로드 작업에서만 지원됩니다.

[the section called “소스로서의 Oracle에 대한 압축 방법”](#)

CDC에 지원되지 않는 HCC 압축이 사용되는지 검증

API 키: `oracle-check-binary-reader-hcc-cdc`

Oracle 소스 엔드포인트는 Binary Reader를 사용하도록 구성됩니다. Binary Reader는 CDC 작업에서 Query low를 지원하지 않습니다.

자세한 내용은 [the section called “CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용”](#) 단원을 참조하십시오.

소스 압축 방법을 기반으로 한 CDC 권장 사항

API 키: `oracle-recommend-cdc-method-by-compression`

압축된 객체가 감지됩니다. 추가 권장 사항을 보려면 특정 평가의 결과 섹션으로 이동하세요.

자세한 내용은 [the section called “CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용”](#) 단원을 참조하십시오.

배치 적용이 활성화되어 있는지 확인하고 테이블에 999개 이상의 열이 있는지 확인합니다

API 키: `oracle-batch-apply-lob-999`

DMS는 $2 * \text{columns\_in\_original\_table} + \text{columns\_in\_primary\_key}$ 공식을 사용하여 고객 테이블의 열 수를 결정합니다. 이 공식을 기반으로 열이 999개 이상인 테이블을 식별했습니다. 이는 배치 프로세스에 영향을 미치므로 실패하고 one-by-one 모드로 전환됩니다.

자세한 내용은 [the section called “대상으로서 Oracle에 적용되는 제한 사항”](#) 단원을 참조하십시오.

Digits Randomize 변환 규칙 확인

API 키: `oracle-datamasking-digits-randomize`

이 평가는 테이블 매핑에 사용되는 열이 Digits Randomize 변환 규칙과 호환되는지 확인합니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

Digits Mask에 대한 변환 규칙 확인

API 키: `oracle-datamasking-digits-mask`

이 평가는 테이블 매핑에 사용되는 열이 Digits Mask 변환 규칙에서 지원되지 않는지 확인합니다. 또한 Digits Mask 변환을 해당 열에 적용하면 고유성을 보장할 수 없으므로 DMS 작업 실패가 발생할 수 있으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

해싱 마스크에 대한 변환 규칙 확인

API 키: `oracle-datamasking-hash-mask`

이 평가는 테이블 매핑에 사용되는 열 중 하나라도 해싱 마스크 변환 규칙에서 지원되지 않는지 확인합니다. 또한 소스 열의 길이가 64자를 초과하는지 확인합니다. 해시 마스킹을 지원하려면 대상 열 길이가 64자를 초과하는 것이 가장 좋습니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인

API 키: `all-to-all-validation-with-datamasking-digits-randomize`

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인

API 키: `all-to-all-validation-with-datamasking-hash-mask`

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인

API 키: `all-to-all-validation-with-digit-mask`

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

스트리밍 대상에 대한 복제에 LOBs 또는 확장 데이터 유형 열이 포함되어 있지 않은지 확인

API 키: `oracle-validate-lob-to-streaming-target`

이 평가는 LOB 또는 확장 데이터 유형을 스트리밍 대상 엔드포인트(예: S3, Kinesis 또는 Kafka)로 마이그레이션할 때 발생할 수 있는 데이터 손실을 식별합니다. Oracle 데이터베이스는 로그 파일에서 이러한 데이터 유형에 대한 변경 사항을 추적하지 않으므로 DMS가 스트리밍 대상에 NULL 값을 씁니다. 데이터 손실을 방지하기 위해 소스 데이터베이스에 'before' 트리거를 구현하여 Oracle이 이러한 변경 사항을 기록하도록 할 수 있습니다.

CDC 전용 작업이 **OpenTransactionWindow** 엔드포인트 설정을 사용하도록 구성되어 있는지 확인합니다.

API 키: `oracle-check-cdc-open-tx-window`

CDC 전용 작업의 경우 OpenTransactionWindow를 사용하여 누락된 데이터를 방지합니다. 자세한 내용은 [AWS DMS를 사용하여 지속적인 복제를 위한 작업 생성](#)을 참조하세요.

선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.

API 키: `all-check-source-selection-rules`

이 사전 마이그레이션 평가는 와일드카드 기반 규칙에 대한 패턴 일치를 포함하여 선택 규칙에 지정된 하나 이상의 객체가 소스 데이터베이스에 존재하는지 확인합니다.

마이그레이션을 위해 대상 외래 키 제약 조건이 비활성화되었는지 확인

API 키: `oracle-target-foreign-key-constraints-check`

이 사전 마이그레이션 평가는 대상 데이터베이스에서 마이그레이션 실패를 일으킬 수 있는 활성 외래 키 제약 조건을 감지합니다(ORA-02291).

Oracle 데이터베이스와 AWS DMS 버전이 호환되는지 확인

API 키: `oracle-dms-compatibility-version-check`

이 사전 마이그레이션 평가는 Oracle 데이터베이스 버전이 버전과 호환되지 않는지 여부를 감지합니다 AWS DMS . 이 불일치로 인해 지원되지 않는 Oracle Redo 호환성 설정으로 인해 작업이 실패할 수 있습니다.

보조 제약 조건 및 인덱스(비기본)가 소스 데이터베이스에 있는지 검증합니다.

API 키: `all-check-secondary-constraints`

이 사전 마이그레이션 평가는 보조 제약 조건 및 인덱스(외래 키, 검사 제약 조건, 클러스터링되지 않은 인덱스)가 소스 데이터베이스에 있는지 확인합니다.

세션 제한 시간 설정(**IDLE_TIME**)이 로 설정되어 있는지 확인합니다. **UNLIMITED**

API 키: `oracle-check-idle-time`

이 사전 마이그레이션 평가는 Oracle 데이터베이스 IDLE_TIME 파라미터가 AWS DMS 사용자에게 UNLIMITED 대 로 설정되어 있는지 확인합니다. 세션 제한 시간으로 인해 연결 제한 시간으로 인해 마이그레이션 작업이 실패할 수 있습니다.

AWS DMS 사용자에게 소스 데이터베이스에 대한 모든 필수 권한이 있는지 확인

API 키: `oracle-validate-permissions-on-source`

이 사전 마이그레이션 평가는 AWS DMS 사용자가 소스 데이터베이스에 필요한 모든 권한으로 구성되어 있는지 확인합니다.

Oracle LogMiner를 사용할 때 **XMLTYPE** 또는 LOB 열이 존재하는지 확인

API 키: `oracle-update-lob-columns`

이 사전 마이그레이션 평가는 Oracle LogMiner를 사용할 때 소스 데이터베이스에 XMLTYPE 또는 LOB 열이 있음을 경고합니다.

대상 엔드포인트가 읽기 전용 복제본이 아닌지 확인

API 키: `all-check-target-read-replica`

이 사전 마이그레이션 평가는 대상 엔드포인트가 읽기 전용 복제본으로 구성되지 않았는지 확인합니다. 대상 데이터베이스에 대한 쓰기 액세스 권한이 AWS DMS 필요하며 읽기 전용 복제본으로 복제할 수 없습니다.

직접 경로 로드를 사용할 때 Oracle 대상에 CONTEXT 인덱스가 없는지 확인

API 키: `oracle-check-direct-path-context-indexes`

이 사전 마이그레이션 평가는 대상 Oracle 테이블에 CONTEXT 인덱스가 포함되어 있는지 확인합니다. AWS DMS 는 전체 로드 모드에서 직접 경로를 사용할 때 CONTEXT 인덱스를 지원하지 않습니다. 장애를 방지하려면 직접 경로 전체 로드 모드를 비활성화하거나 로드 전에 CONTEXT 인덱스를 제거합니다.

자세한 내용은 [Oracle을의 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

기존 LOB 열에서 제한된 LOB 모드를 사용할 때 **FailTasksOnLobTruncation**가 활성화되었는지 확인

API 키: oracle-pg-lobb-with-failtasksonlobtruncation

이 사전 마이그레이션 평가는 선택한 테이블에 LOB 열이 있고 제한된 LOB 모드가 지정된 경우 **FailTasksOnLobTruncation** 추가 연결 속성이 true로 설정되어 있는지 확인합니다. 이 설정은 마이그레이션 중에 LOB 데이터가 잘리면 작업에 실패하여 자동 데이터 손실을 방지합니다.

자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

EnableHomogenousPartitionOps 엔드포인트 설정이 활성화되었는지 확인

API 키: oracle-homogenous-partition-ops

이 사전 마이그레이션 평가는 Oracle 동종 마이그레이션에 대해 **EnableHomogenousPartitionOps** 엔드포인트 설정이 활성화되어 있는지 확인합니다. 이 설정은 Oracle 파티션 및 하위 파티션 DDL 작업을 복제 AWS DMS 하는 데 필요합니다.

자세한 내용은 [Oracle을의 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

SQL Server 평가

이 섹션에서는 Microsoft SQL Server 원본 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 마이그레이션 전 평가에 대해 설명합니다.

주제

- [전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 활성화되었는지 확인](#)
- [BatchApplyEnabled가 true로 설정된 경우에만 제한된 LOB 모드가 사용되는지 확인](#)
- [대상 데이터베이스에 태스크 범위의 테이블에 활성화된 트리거가 있는지 확인합니다.](#)
- [작업 범위의 테이블에 계산된 열이 포함되어 있는지 확인합니다.](#)
- [작업 범위의 테이블에 열 저장소 인덱스가 있는지 확인합니다.](#)
- [메모리 최적화 테이블이 작업 범위에 속하는지 확인합니다.](#)
- [임시 테이블이 작업 범위에 속하는지 확인합니다.](#)
- [데이터베이스 수준에서 지연된 내구성이 활성화되어 있는지 확인합니다.](#)
- [데이터베이스 수준에서 가속화된 데이터 복구가 활성화되어 있는지 확인하세요.](#)

- [테이블 매핑에 프라이머리 키가 포함된 1만 개 이상의 테이블이 있는지 확인하세요.](#)
- [소스 데이터베이스에 특수 문자가 포함된 테이블 또는 스키마 이름이 있는지 확인합니다.](#)
- [소스 데이터베이스에 마스킹된 데이터가 있는 열 이름이 있는지 확인](#)
- [소스 데이터베이스에 암호화된 백업이 있는지 확인](#)
- [소스 데이터베이스에 URL 또는 Windows Azure에 저장된 백업이 있는지 확인합니다.](#)
- [소스 데이터베이스에 여러 디스크에 백업이 있는지 확인](#)
- [소스 데이터베이스에 전체 백업이 하나 이상 있는지 확인](#)
- [소스 데이터베이스에 최소 열과 열 구조 압축이 있는지 확인합니다.](#)
- [소스 데이터베이스 인스턴스에 SQL Server 2008 또는 SQL Server 2008 R2에 대한 서버 수준 감사가 있는지 확인](#)
- [소스 데이터베이스에 전체 LOB 모드에 대한 지오메트리 열이 있는지 확인](#)
- [소스 데이터베이스에 Identity 속성이 있는 열이 있는지 확인합니다.](#)
- [DMS 사용자에게 FULL LOAD 권한이 있는지 확인](#)
- [DMS 사용자에게 FULL LOAD 및 CDC 또는 CDC 전용 권한이 있는지 확인합니다.](#)
- [온프레미스 또는 EC2 데이터베이스에서 CDC에 대해 MS 복제가 활성화되어 있는지 확인합니다.](#)
- [DMS 사용자에게 VIEW DEFINITION 권한이 있는지 확인합니다.](#)
- [DMS 사용자에게 Sysadmin 역할이 없는 사용자를 위한 MASTER 데이터베이스에 대한 데이터베이스 상태 보기 권한이 있는지 확인합니다.](#)
- [DMS 사용자에게 VIEW SERVER STATE 권한이 있는지 확인합니다.](#)
- [텍스트 리플렉션 크기 파라미터가 무제한이 아닌지 검증](#)
- [배치 적용 대상에 프라이머리 키 또는 고유 인덱스가 있는지 확인](#)
- [배치 적용이 활성화된 경우, 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 검증합니다](#)
- [DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있는지 검증](#)
- [AWS DMS 사용자에게 대상에 필요한 권한이 있는지 확인](#)
- [MaxFullLoadSubTasks 설정 사용에 대한 권장 사항](#)
- [Digits Randomize 변환 규칙 확인](#)
- [Digits Mask에 대한 변환 규칙 확인](#)
- [해싱 마스크에 대한 변환 규칙 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인](#)

- [데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인](#)
- [선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.](#)
- [보조 제약 조건 및 인덱스\(비기본\)가 소스 데이터베이스에 있는지 검증합니다.](#)
- [대상 엔드포인트가 읽기 전용 복제본이 아닌지 확인](#)
- [백업 체인 검증](#)
- [EXCLUSIVE_AUTOMATIC_TRUNCATION 보호 정책 적용에 대한 데이터베이스 사용자 권한 확인](#)
- [보조 노드 연결 및 AWS DMS 소스 엔드포인트에 필요한 보호 속성 확인](#)
- [가 보조 노드에 연결될 때 엔드포인트에 필요한 추가 연결 속성\(ECAs\) AWS DMS 이 모두 있는지 확인합니다.](#)

전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 활성화되었는지 확인

API 키: sqlserver-check-secondary-indexes

이 마이그레이션 전 평가는 대상 데이터베이스의 전체 로드 중에 보조 인덱스가 활성화되었는지 여부를 검증합니다. 보조 인덱스를 비활성화하거나 제거하는 것이 좋습니다.

자세한 내용은 [AWS Database Migration Service에 대한 모범 사례](#)를 참조하세요.

BatchApplyEnabled가 true로 설정된 경우에만 제한된 LOB 모드가 사용되는지 확인

API 키: sqlserver-batch-apply-lob-mode

이 마이그레이션 전 평가는 DMS 태스크에 LOB 열이 포함되어 있는지 확인합니다. LOB 열이 태스크 범위에 포함된 경우 제한된 LOB 모드와 함께 BatchApplyEnabled를 사용해야 합니다. 이러한 테이블에 대해 별도의 태스크를 생성하고 대신 트랜잭션 적용 모드를 사용하는 것이 좋습니다.

자세한 내용은 [DMS 배치 적용 기능을 사용하여 CDC 복제 성능을 개선하려면 어떻게 해야 합니까?](#)를 참조하세요.

대상 데이터베이스에 태스크 범위의 테이블에 활성화된 트리거가 있는지 확인합니다.

API 키: sqlserver-check-for-triggers

AWS DMS 는 대상 데이터베이스에서 전체 로드 DMS 작업의 성능과 대상 지연 시간에 영향을 미칠 수 있는 트리거를 식별했습니다. 이러한 트리거는 태스크 실행 중에 비활성화되고 전환 기간 중에 활성화되어야 합니다.

작업 범위의 테이블에 계산된 열이 포함되어 있는지 확인합니다.

API 키: sqlserver-check-for-computed-fields

이 사전 마이그레이션 평가는 계산된 열이 있는지 확인합니다. AWS DMS 는 SQL Server에서 계산된 열의 변경 사항 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

작업 범위의 테이블에 열 저장소 인덱스가 있는지 확인합니다.

API 키: sqlserver-check-for-columnstore-indexes

이 사전 마이그레이션 평가는 열 저장소 인덱스가 있는 테이블이 있는지 확인합니다. AWS DMS 는 열 저장소 인덱스가 있는 SQL Server 테이블의 변경 사항 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

메모리 최적화 테이블이 작업 범위에 속하는지 확인합니다.

API 키: sqlserver-check-for-memory-optimized-tables

이 사전 마이그레이션 평가는 메모리 최적화 테이블이 있는지 확인합니다. AWS DMS 는 메모리 최적화 테이블의 변경 사항 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

임시 테이블이 작업 범위에 속하는지 확인합니다.

API 키: sqlserver-check-for-temporal-tables

이 사전 마이그레이션 평가는 임시 테이블이 있는지 확인합니다. AWS DMS 는 임시 테이블의 변경 사항 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

데이터베이스 수준에서 지연된 내구성이 활성화되어 있는지 확인합니다.

API 키: sqlserver-check-for-delayed-durability

이 사전 마이그레이션 평가는 지연된 내구성의 존재 여부를 확인합니다. AWS DMS 는 지연된 내구성을 사용하는 트랜잭션의 변경 사항 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

데이터베이스 수준에서 가속화된 데이터 복구가 활성화되어 있는지 확인하세요.

API 키: sqlserver-check-for-accelerated-data-recovery

이 사전 마이그레이션 평가는 가속화된 데이터 복구가 있는지 확인합니다. AWS DMS 는 가속화된 데이터 복구로 데이터베이스의 변경 사항 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

테이블 매핑에 프라이머리 키가 포함된 1만 개 이상의 테이블이 있는지 확인하세요.

API 키: sqlserver-large-number-of-tables

이 마이그레이션 전 평가에서는 프라이머리 키가 포함된 10,000개 이상의 테이블이 있는지 확인합니다. MS-Replication으로 구성된 데이터베이스는 프라이머리 키가 포함된 테이블이 너무 많으면 작업 실패가 발생할 수 있습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

MS-Replication 구성에 관한 자세한 내용은 [SQL Server에서 지속적인 복제를 위한 데이터 변경 사항 캡처](#) 섹션을 참조하세요.

소스 데이터베이스에 특수 문자가 포함된 테이블 또는 스키마 이름이 있는지 확인합니다.

API 키: `sqlserver-check-for-special-characters`

이 마이그레이션 전 평가는 소스 데이터베이스에 다음 세트의 문자가 포함된 테이블 또는 스키마 이름이 있는지 확인합니다.

```
\\ -- \n \" \b \r ' \t ;
```

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스에 마스킹된 데이터가 있는 열 이름이 있는지 확인

API 키: `sqlserver-check-for-masked-data`

이 마이그레이션 전 평가는 소스 데이터베이스에 마스킹된 데이터가 있는지 확인합니다. 는 마스킹 없이 마스킹된 데이터를 AWS DMS 마이그레이션합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스에 암호화된 백업이 있는지 확인

API 키: `sqlserver-check-for-encrypted-backups`

이 마이그레이션 전 평가는 소스 데이터베이스에 암호화된 백업이 있는지 확인합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스에 URL 또는 Windows Azure에 저장된 백업이 있는지 확인합니다.

API 키: `sqlserver-check-for-backup-url`

이 마이그레이션 전 평가는 소스 데이터베이스에 URL에 저장된 백업이 있는지 또는 Windows Azure에 저장된 백업이 있는지 확인합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스에 여러 디스크에 백업이 있는지 확인

API 키: `sqlserver-check-for-backup-multiple-stripes`

이 마이그레이션 전 평가는 소스 데이터베이스에 여러 디스크에 대한 백업이 있는지 확인합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스에 전체 백업이 하나 이상 있는지 확인

API 키: sqlserver-check-for-full-backup

이 마이그레이션 전 평가는 소스 데이터베이스에 전체 백업이 하나 이상 있는지 확인합니다. SQL Server는 전체 백업에 맞게 구성되어야 하며, 사용자는 데이터 복제를 시작하기 전에 백업을 수행해야 합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스에 희소 열과 열 구조 압축이 있는지 확인합니다.

API 키: sqlserver-check-for-sparse-columns

이 마이그레이션 전 평가는 소스 데이터베이스에 희소 열과 열 구조 압축이 있는지 확인합니다. DMS는 희소 열과 열 구조 압축을 지원하지 않습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스 인스턴스에 SQL Server 2008 또는 SQL Server 2008 R2에 대한 서버 수준 감사가 있는지 확인

API 키: sqlserver-check-for-audit-2008

이 마이그레이션 전 평가는 소스 데이터베이스가 SQL Server 2008 또는 SQL Server 2008 R2에 대한 서버 수준 감사를 활성화했는지 확인합니다. DMS에는 SQL Server 2008 및 2008 R2와 관련된 알려진 문제가 있습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스에 전체 LOB 모드에 대한 지오메트리 열이 있는지 확인

API 키: sqlserver-check-for-geometry-columns

이 마이그레이션 전 평가는 SQL Server를 소스로 사용할 때 소스 데이터베이스에 전체 대형 객체 (LOB) 모드에 대한 지오메트리 열이 있는지 확인합니다. 데이터베이스에 지오메트리 열이 포함된 경우 제한된 LOB 모드를 사용하거나 인라인 LOB 모드를 사용하도록 InlineLobMaxSize 태스크 설정을 지정하는 것이 좋습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

소스 데이터베이스에 Identity 속성이 있는 열이 있는지 확인합니다.

API 키: sqlserver-check-for-identity-columns

이 마이그레이션 전 평가는 소스 데이터베이스에 IDENTITY 속성이 있는 열이 있는지 확인합니다. DMS는 이 속성을 해당 대상 데이터베이스 열로 마이그레이션하지 않습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

DMS 사용자에게 FULL LOAD 권한이 있는지 확인

API 키: sqlserver-check-user-permission-for-full-load-only

이 마이그레이션 전 평가는 DMS 작업의 사용자에게 FULL LOAD 모드에서 작업을 실행할 수 있는 권한이 있는지 확인합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

DMS 사용자에게 FULL LOAD 및 CDC 또는 CDC 전용 권한이 있는지 확인합니다.

API 키: sqlserver-check-user-permission-for-cdc

이 마이그레이션 전 평가는 DMS 사용자에게 FULL LOAD and CDC 또는 CDC only 모드에서 작업을 실행할 수 있는 권한이 있는지 확인합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

온프레미스 또는 EC2 데이터베이스에서 CDC에 대해 MS 복제가 활성화되어 있는지 확인합니다.

API 키: sqlserver-check-attribute-for-enable-ms-cdc-onprem

온프레미스 또는 EC2 데이터베이스에서 CDC에 대해 MS 복제가 활성화되어 있는지 확인합니다.

MS-Replication 구성에 관한 자세한 내용은 [온프레미스 또는 Amazon EC2에서 자체 관리형 SQL Server의 데이터 변경 캡처](#) 섹션을 참조하세요.

DMS 사용자에게 VIEW DEFINITION 권한이 있는지 확인합니다.

API 키: sqlserver-check-user-permission-on-view-definition

이 마이그레이션 전 평가는 엔드포인트 설정에 지정된 사용자에게 VIEW DEFINITION 권한이 있는지 확인합니다. DMS에는 객체 정의를 볼 수 있는 VIEW DEFINITION 권한이 필요합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

DMS 사용자에게 Sysadmin 역할이 없는 사용자를 위한 MASTER 데이터베이스에 대한 데이터베이스 상태 보기 권한이 있는지 확인합니다.

API 키: sqlserver-check-user-permission-on-view-database-state

이 마이그레이션 전 평가는 엔드포인트 설정에 지정된 사용자에게 VIEW DATABASE STATE 권한이 있는지 확인합니다. DMS에서는 MASTER 데이터베이스의 데이터베이스 객체에 액세스하려면 이 권한이 필요합니다. DMS에서는 사용자에게 sysadmin 권한이 없는 경우에도 이 권한이 필요합니다. DMS에서는 함수, 인증서 및 로그인을 생성하고 자격 증명을 부여하려면 이 권한이 필요합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

DMS 사용자에게 VIEW SERVER STATE 권한이 있는지 확인합니다.

API 키: sqlserver-check-user-permission-on-view-server-state

이 마이그레이션 전 평가는 추가 연결 속성(ECA)에 지정된 사용자에게 VIEW SERVER STATE 권한이 있는지 확인합니다. VIEW SERVER STATE는 사용자가 서버 전체 정보 및 상태를 볼 수 있는 서버 수준 권한입니다. 이 권한은 SQL Server 인스턴스에 대한 정보를 노출하는 동적 관리 뷰(DMV) 및 동적 관리 함수(DMF)에 대한 액세스를 제공합니다. DMS 사용자가 CDC 리소스에 액세스하려면 이 권한이 필요합니다. FULL LOAD and CDC 또는 CDC only 모드에서 DMS 작업을 실행하려면 이 권한이 필요합니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

텍스트 리플렉션 크기 파라미터가 무제한이 아닌지 검증

API 키: sqlserver-check-for-max-text-repl-size

데이터베이스에서 최대 텍스트 리플렉션 크기 파라미터를 설정하면 LOB 열에 대한 데이터 마이그레이션 오류가 발생할 수 있습니다. DMS는 이를 -1로 설정하는 것을 적극 권장합니다.

자세한 내용은 [the section called “Microsoft SQL Server 관련 문제 해결”](#) 단원을 참조하십시오.

배치 적용 대상에 프라이머리 키 또는 고유 인덱스가 있는지 확인

API 키: sqlserver-check-batch-apply-target-pk-ui-absence

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 동시에 존재하는 테이블은 배치가 실패하며 변경 사항이 하나씩 처리됩니다.

다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키를 생성할 수 있습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

배치 적용이 활성화된 경우, 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 검증합니다

API 키:sqlserver-check-batch-apply-target-pk-ui-simultaneously

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 동시에 있는 테이블은 배치 처리 시 실패하며, 변경 사항은 하나씩 처리됩니다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유 키 또는 프라이머리 키를 삭제하고 마이그레이션할 때 다시 빌드할 수 있습니다.

자세한 내용은 [SQL Server를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있는지 검증

API 키:sqlserver-check-pk-validity

데이터를 검증하려면 테이블에 프라이머리 키나 고유한 인덱스가 있어야 합니다.

자세한 내용은 [데이터 유효성 검사](#) 단원을 참조하십시오.

AWS DMS 사용자에게 대상에 필요한 권한이 있는지 확인

API 키:sqlserver-check-target-privileges

AWS DMS 사용자는 대상 데이터베이스에 최소한 db_owner 사용자 역할이 있어야 합니다.

자세한 내용은 [SQL Server를의 대상으로 사용할 때의 보안 요구 사항 AWS Database Migration Service](#) 단원을 참조하십시오.

MaxFullLoadSubTasks 설정 사용에 대한 권장 사항

API 키:sqlserver-tblnum-for-max-fullload-subtasks

이 평가는 작업에 포함된 테이블 수를 확인하고 전체 로드 프로세스 중에 최적의 성능을 위해 MaxFullLoadSubTasks 파라미터를 늘릴 것을 권장합니다. 기본적으로는 8개의 테이블을 동시에 AWS DMS 마이그레이션합니다. MaxFullLoadSubTasks 파라미터를 더 높은 값으로 변경하면 전체 로드 성능이 향상됩니다.

자세한 내용은 [전체 로드 작업 설정](#) 단원을 참조하십시오.

Digits Randomize 변환 규칙 확인

API 키: sqlserver-datamasking-digits-randomize

이 평가는 테이블 매핑에 사용되는 열이 Digits Randomize 변환 규칙과 호환되는지 확인합니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

Digits Mask에 대한 변환 규칙 확인

API 키: sqlserver-datamasking-digits-mask

이 평가는 테이블 매핑에 사용되는 열이 Digits Mask 변환 규칙에서 지원되지 않는지 확인합니다. 또한 Digits Mask 변환을 해당 열에 적용하면 고유성을 보장할 수 없으므로 DMS 작업 실패가 발생할 수 있으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

해싱 마스크에 대한 변환 규칙 확인

API 키: sqlserver-datamasking-hash-mask

이 평가는 테이블 매핑에 사용되는 열 중 하나라도 해싱 마스크 변환 규칙에서 지원되지 않는지 확인합니다. 또한 소스 열의 길이가 64자를 초과하는지 확인합니다. 해시 마스킹을 지원하려면 대상 열 길이가 64자를 초과하는 것이 가장 좋습니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-datamasking-digits-randomize

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-datamasking-hash-mask

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-digit-mask

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.

API 키: all-check-source-selection-rules

이 사전 마이그레이션 평가는 와일드카드 기반 규칙에 대한 패턴 일치를 포함하여 선택 규칙에 지정된 하나 이상의 객체가 소스 데이터베이스에 존재하는지 확인합니다.

보조 제약 조건 및 인덱스(비기본)가 소스 데이터베이스에 있는지 검증합니다.

API 키: all-check-secondary-constraints

이 사전 마이그레이션 평가는 보조 제약 조건 및 인덱스(외래 키, 검사 제약 조건, 클러스터링되지 않은 인덱스)가 소스 데이터베이스에 있는지 확인합니다.

대상 엔드포인트가 읽기 전용 복제본이 아닌지 확인

API 키: all-check-target-read-replica

이 사전 마이그레이션 평가는 대상 엔드포인트가 읽기 전용 복제본으로 구성되지 않았는지 확인합니다. 대상 데이터베이스에 대한 쓰기 액세스 권한이 AWS DMS 필요하며 읽기 전용 복제본으로 복제할 수 없습니다.

백업 체인 검증

API 키: sqlserver-check-for-backup-broken-chain

이 사전 마이그레이션 평가는 소스 데이터베이스 백업 체인이 손상되지 않았는지 확인합니다. 끊어진 백업 체인은 AWS DMS 가 CDC 복제에 필요한 트랜잭션 로그에 액세스하는 것을 방지할 수 있습니다.

EXCLUSIVE_AUTOMATIC_TRUNCATION 보호 정책 적용에 대한 데이터베이스 사용자 권한 확인

API 키: sqlserver-safeguard-permissions

이 사전 마이그레이션 평가는 데이터베이스 사용자에게 EXCLUSIVE_AUTOMATIC_TRUNCATION 보호 정책을 사용하는 데 필요한 권한이 있는지 확인합니다. 사용자는 `dbo.syscategories` 및 `dbo.sysjobs` 시스템 객체에 대한 SELECT 권한을 `dmsuser`에 부여해야 합니다.

자세한 내용은 [SQL Server를의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

보조 노드 연결 및 AWS DMS 소스 엔드포인트에 필요한 보호 속성 확인

API 키: sqlserver-check-sec-node-sg-policy

이 사전 마이그레이션 평가는 보호 기능이 활성화된 보조 노드에 연결할 때 소스 엔드포인트에 필수 추가 연결 속성(ECAs)이 구성되어 있는지 확인합니다.

자세한 내용은 [SQL Server를의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#) 단원을 참조하십시오.

가 보조 노드에 연결될 때 엔드포인트에 필요한 추가 연결 속성(ECAs) AWS DMS 이 모두 있는지 확인합니다.

API 키: sqlserver-check-sec-node-without-eca

이 사전 마이그레이션 평가는 소스 엔드포인트가 보조 노드에 연결할 때 필요한 모든 추가 연결 속성(ECAs)이 구성되어 있는지 확인합니다.

자세한 내용은 [자체 관리형 SQL Server AlwaysOn 가용성 그룹 사용](#) 단원을 참조하십시오.

MySQL 평가

이 섹션에서는 MySQL, Aurora MySQL-Compatible Edition 또는 Aurora MySQL-Compatible Edition Serverless 소스 엔드포인트를 사용하는 마이그레이션 태스크에 대한 개별 마이그레이션 전 평가를 설명합니다.

주제

- [바이너리 로그 트랜잭션 압축이 비활성화되었는지 확인](#)
- [DMS 사용자에게 소스 데이터베이스에 대한 REPLICATION CLIENT 및 REPLICATION SLAVE 권한이 있는지 확인](#)
- [DMS 사용자에게 소스 데이터베이스 테이블에 대한 SELECT 권한이 있는지 확인](#)
- [소스 데이터베이스에서 server_id가 1 이상으로 설정되어 있는지 확인](#)
- [DMS 사용자에게 대상으로서 MySQL 데이터베이스에 대한 필수 권한이 있는지 확인](#)
- [소스 데이터베이스에 바이너리 로그의 자동 제거가 설정되어 있는지 확인](#)
- [BatchApplyEnabled가 true로 설정된 경우에만 제한된 LOB 모드가 사용되는지 확인](#)
- [테이블이 Innodb 이외의 스토리지 엔진을 사용하는지 확인](#)
- [마이그레이션에 사용되는 테이블에 자동 증분이 활성화되어 있는지 확인](#)

- [DMS CDC를 지원하도록 데이터베이스 binlog 이미지가 FULL로 설정되어 있는지 확인](#)
- [소스 데이터베이스가 MySQL 읽기 전용 복제본인지 확인](#)
- [테이블에 파티션이 있는지 확인하고 전체 로드 태스크 설정에 target_table_prep_mode 권장](#)
- [DMS가 데이터베이스 버전을 지원하는지 확인](#)
- [대상 데이터베이스가 local_infile을 1로 설정하도록 구성되어 있는지 확인](#)
- [대상 데이터베이스에 외부 키가 있는 테이블이 있는지 확인](#)
- [태스크 범위의 소스 테이블에 캐스케이드 제약 조건이 있는지 확인](#)
- [제한 시간 값이 MySQL 소스 또는 대상에 적합한지 확인](#)
- [max_statement_time 데이터베이스 파라미터 검증](#)
- [배치 적용 대상에 프라이머리 키 또는 고유 인덱스가 있는지 확인](#)
- [배치 적용 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 확인](#)
- [전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 활성화되었는지 확인](#)
- [DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있는지 검증](#)
- [MaxFullLoadSubTasks 설정 사용에 대한 권장 사항](#)
- [Digits Randomize 변환 규칙 확인](#)
- [Digits Mask에 대한 변환 규칙 확인](#)
- [해싱 마스크에 대한 변환 규칙 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인](#)
- [소스 Amazon Aurora MySQL 인스턴스가 읽기 전용 복제본이 아닌지 확인](#)
- [이진 로그 보존 시간이 올바르게 설정되었는지 확인](#)
- [소스 테이블에 보이지 않는 열이 없는지 확인합니다.](#)
- [DMS CDC를 지원하도록 데이터베이스 binlog 형식이 ROW로 설정되어 있는지 확인](#)
- [선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.](#)
- [생성된 열이 있는 테이블이 소스 데이터베이스에 존재하는지 검증](#)
- [분할된 테이블에 대해 skipTableSuspensionForPartitionDdl이 활성화되어 있는지 확인합니다.](#)
- [max_allowed_packet 크기가 소스 LOB 열을 처리할 수 있는지 확인](#)
- [보조 제약 조건 및 인덱스\(비기본\)가 소스 데이터베이스에 있는지 검증합니다.](#)

바이너리 로그 트랜잭션 압축이 비활성화되었는지 확인

API 키: `mysql-check-binlog-compression`

이 마이그레이션 전 평가는 바이너리 로그 트랜잭션 압축이 비활성화되었는지 확인합니다. AWS DMS 는 바이너리 로그 트랜잭션 압축을 지원하지 않습니다.

자세한 내용은 [MySQL 데이터베이스를 AWS DMS의 소스로 사용하는 것에 대한 제한 사항](#)을 참조하세요.

DMS 사용자에게 소스 데이터베이스에 대한 REPLICATION CLIENT 및 REPLICATION SLAVE 권한이 있는지 확인

API 키: `mysql-check-replication-privileges`

이 마이그레이션 전 평가는 DMS 태스크 마이그레이션 유형이 CDC 또는 전체 로드 + CDC인 경우 소스 엔드포인트 연결 설정에 지정된 DMS 사용자에게 소스 데이터베이스에 대한 REPLICATION CLIENT 및 REPLICATION SLAVE 권한이 있는지 확인합니다.

자세한 내용은 [MySQL 호환 데이터베이스를 AWS DMS의 원본으로 사용](#)을 참조하세요.

DMS 사용자에게 소스 데이터베이스 테이블에 대한 SELECT 권한이 있는지 확인

API 키: `mysql-check-select-privileges`

이 마이그레이션 전 평가는 소스 엔드포인트 연결 설정에 지정된 DMS 사용자에게 소스 데이터베이스 테이블에 대한 SELECT 권한이 있는지 확인합니다.

자세한 내용은 [MySQL 호환 데이터베이스를 AWS DMS의 원본으로 사용](#)을 참조하세요.

소스 데이터베이스에서 `server_id`가 1 이상으로 설정되어 있는지 확인

API 키: `mysql-check-server-id`

이 마이그레이션 전 평가는 CDC 마이그레이션 유형에 대한 소스 데이터베이스에서 `server_id` 서버 변수가 1 이상으로 설정되었는지 확인합니다.

소스에 대한 자세한 내용은 자체 관리형 MySQL 호환 데이터베이스를 소스로 사용을 AWS DMS참조하세요. [MySQL AWS DMS](#)

DMS 사용자에게 대상으로서 MySQL 데이터베이스에 대한 필수 권한이 있는지 확인

API 키: `mysql-check-target-privileges`

이 마이그레이션 전 평가는 대상 엔드포인트 연결 설정에 지정된 DMS 사용자에게 MySQL 데이터베이스에 필요한 권한이 대상인지 확인합니다.

MySQL 소스 엔드포인트 사전 조건에 대한 자세한 내용은 [AWS DMS의 원본으로 MySQL 호환 데이터베이스 사용](#)을 참조하세요.

소스 데이터베이스에 바이너리 로그의 자동 제거가 설정되어 있는지 확인

API 키: `mysql-check-expire-logs-days`

이 마이그레이션 전 평가는 데이터베이스가 바이너리 로그를 자동으로 제거하도록 구성되어 있는지 확인합니다. 마이그레이션 중에 디스크 공간이 과도하게 사용되지 않도록 EXPIRE_LOGS_DAYS 또는 BINLOG_EXPIRE_LOGS_SECONDS 전역 시스템 변수의 값은 0보다 커야 합니다.

소스에 대한 자세한 내용은 자체 관리형 MySQL 호환 데이터베이스를 소스로 사용을 AWS DMS참조하세요. [MySQL AWS DMS](#)

BatchApplyEnabled가 true로 설정된 경우에만 제한된 LOB 모드가 사용되는지 확인

API 키: `mysql-batch-apply-lob-mode`

이 마이그레이션 전 평가는 DMS 태스크에 LOB 열이 포함되어 있는지 확인합니다. LOB 열이 태스크 범위에 포함된 경우 제한된 LOB 모드와 함께 BatchApplyEnabled를 사용해야 합니다.

BatchApplyEnabled 설정에 대한 자세한 내용은 [DMS 배치 적용 기능을 사용하여 CDC 복제 성능을 개선하려면 어떻게 해야 합니까?](#)를 참조하세요.

테이블이 InnoDB 이외의 스토리지 엔진을 사용하는지 확인

API 키: `mysql-check-table-storage-engine`

이 마이그레이션 전 평가는 소스 MySQL 데이터베이스의 테이블에 사용되는 스토리지 엔진이 InnoDB 이외의 엔진인지 확인합니다. 기본적으로 DMS에서는 InnoDB 스토리지 엔진이 있는 대상 테이블을 생성합니다. InnoDB 이외의 스토리지 엔진을 사용해야 하는 경우 대상 데이터베이스에 테이블을 수동으로 생성하고 TRUNCATE_BEFORE_LOAD 또는 DO_NOTHING을 전체 로드 태스크 설정으로 사용하도록 DMS 태스크를 구성해야 합니다. 전체 로드 태스크 설정에 관한 자세한 내용은 [전체 로드 작업 설정](#) 섹션을 참조하세요.

Note

Aurora MySQL-Compatible Edition 또는 Aurora MySQL-Compatible Edition Serverless에서는 이 마이그레이션 전 평가를 사용할 수 없습니다.

MySQL 엔드포인트 제한 사항에 대한 자세한 내용은 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요.

마이그레이션에 사용되는 테이블에 자동 증분이 활성화되어 있는지 확인

API 키: `mysql-check-auto-increment`

이 마이그레이션 전 평가는 태스크에 사용되는 소스 테이블에 자동 증분이 활성화되어 있는지 확인합니다. DMS에서는 열의 AUTO_INCREMENT 속성은 대상 데이터베이스로 마이그레이션하지 않습니다.

MySQL 엔드포인트 제한 사항에 대한 자세한 내용은 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요. MySQL에서 자격 증명 열을 처리하는 방법에 대한 자세한 내용은 [AWS DMS 2부의 IDENTITY 열 처리를 참조하세요](#).

DMS CDC를 지원하도록 데이터베이스 binlog 이미지가 **FULL**로 설정되어 있는지 확인

API 키: `mysql-check-binlog-image`

이 마이그레이션 전 평가는 소스 데이터베이스의 binlog 이미지가 FULL로 설정되어 있는지 확인합니다. MySQL에서 `binlog_row_image` 변수는 ROW 형식을 사용할 때 바이너리 로그 이벤트가 기록되는 방식을 결정합니다. DMS와의 호환성을 보장하고 CDC를 지원하려면 `binlog_row_image` 변수를 FULL로 설정합니다. 이 설정을 통해 DMS는 마이그레이션 중에 대상 데이터베이스에 대한 전체 데이터 조작 언어(DML)를 구성할 수 있는 충분한 정보를 수신할 수 있습니다.

binlog 이미지를 FULL로 설정하려면 다음을 수행합니다.

- Amazon RDS의 경우 이 값은 기본적으로 FULL입니다.
- 온프레미스 또는 Amazon EC2에서 호스팅되는 데이터베이스의 경우 `binlog_row_image` 값을 `my.ini`(Microsoft Windows) 또는 `my.cnf`(UNIX)로 설정합니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

소스 데이터베이스가 MySQL 읽기 전용 복제본인지 확인

API 키: `mysql-check-database-role`

이 마이그레이션 전 평가는 소스 데이터베이스가 읽기 전용 복제본인지 확인합니다. 읽기 전용 복제본에 연결할 때 DMS에 대한 CDC 지원을 활성화하려면 `log_slave_updates` 파라미터를 True로 설

정합니다. 자체 관리형 MySQL 데이터베이스 사용에 대한 자세한 내용은 [자체 관리형 MySQL 호환 데이터베이스를의 소스로 사용 AWS DMS](#) 섹션을 참조하세요.

log_slave_updates 값을 True로 설정하려면 다음을 수행합니다.

- Amazon RDS의 경우 데이터베이스의 파라미터 그룹을 사용합니다. RDS 데이터베이스 파라미터 그룹 사용에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [파라미터 그룹 작업](#)을 참조하세요.
- 온프레미스 또는 Amazon EC2에서 호스팅되는 데이터베이스의 경우 log_slave_updates 값을 my.ini(Microsoft Windows) 또는 my.cnf(UNIX)로 설정합니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

테이블에 파티션이 있는지 확인하고 전체 로드 태스크 설정에 **target_table_prep_mode** 권장

API 키: mysql-check-table-partition

이 마이그레이션 전 평가는 소스 데이터베이스에 파티션이 있는 테이블이 있는지 확인합니다. DMS는 MySQL 대상에 파티션이 없는 테이블을 생성합니다. 파티션된 테이블을 대상의 파티션된 테이블로 마이그레이션하려면 다음을 수행해야 합니다.

- 대상 MySQL 데이터베이스에서 파티션된 테이블을 미리 생성합니다.
- TRUNCATE_BEFORE_LOAD 또는 DO_NOTHING을 전체 로드 태스크 설정으로 사용하도록 DMS 태스크를 구성합니다.

MySQL 엔드포인트 제한 사항에 대한 자세한 내용은 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요.

DMS가 데이터베이스 버전을 지원하는지 확인

API 키: mysql-check-supported-version

이 마이그레이션 전 평가는 소스 데이터베이스 버전이 DMS와 호환되는지 확인합니다. 지원되는 MySQL 버전에 대한 자세한 내용은 [데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트](#) 섹션을 참조하세요.

대상 데이터베이스가 **local_infile**을 1로 설정하도록 구성되어 있는지 확인

API 키: mysql-check-target-localinfile-set

이 마이그레이션 전 평가는 대상 데이터베이스의 `local_infile` 파라미터가 1로 설정되었는지 확인합니다. DMS를 사용하려면 대상 데이터베이스에 전체 로드하는 동안 'local_infile' 파라미터를 1로 설정해야 합니다. 자세한 내용은 [를 사용하여 MySQL에서 MySQL로 마이그레이션 AWS DMS](#) 단원을 참조하십시오.

이 평가는 전체 로드 또는 전체 로드 및 CDC 태스크에 대해서만 유효합니다.

대상 데이터베이스에 외부 키가 있는 테이블이 있는지 확인

API 키: `mysql-check-fk-target`

이 마이그레이션 전 평가는 MySQL 데이터베이스로 마이그레이션하는 전체 로드 또는 전체 및 CDC 태스크에 외부 키가 있는 테이블이 있는지 확인합니다. DMS의 기본 설정은 알파벳순으로 테이블을 로드하는 것입니다. 상위 테이블과 하위 테이블이 동시에 로드되지 않을 수 있으므로 외부 키와 참조 무결성 제약이 있는 테이블은 로드가 실패할 수 있습니다.

DMS의 참조 무결성에 대한 자세한 내용은 [AWS DMS 마이그레이션 성능 개선](#) 주제의 인덱스, 트리거 및 참조 무결성 제약 조건 작업을 참조하세요.

태스크 범위의 소스 테이블에 캐스케이드 제약 조건이 있는지 확인

API 키: `mysql-check-cascade-constraints`

이 마이그레이션 전 평가에서는 MySQL 소스 테이블에 캐스케이드 제약 조건이 있는지 확인합니다. MySQL은 이러한 이벤트에 대한 변경 사항을 binlog에 기록하지 않기 때문에 캐스케이드 제약 조건은 DMS 태스크로 마이그레이션되거나 복제되지 않습니다. AWS DMS 는 이러한 제약 조건을 지원하지 않지만 관계형 데이터베이스 대상에 대한 해결 방법을 사용할 수 있습니다.

캐스케이드 제약 조건 및 기타 제약 조건 지원에 대한 자세한 내용은 AWS DMS의 마이그레이션 태스크 문제 해결 주제의 [인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음](#) 섹션을 참조하세요.

제한 시간 값이 MySQL 소스 또는 대상에 적합한지 확인

API 키: `mysql-check-target-network-parameter`

이 마이그레이션 전 평가는 작업의 MySQL 엔드포인트에 `net_read_timeout`, `net_write_timeout` 및 `wait_timeout` 설정이 300초 이상으로 설정되어 있는지 확인합니다. 이는 마이그레이션 중에 연결이 끊어지지 않도록 하기 위해 필요합니다.

자세한 내용은 [작업 중에 대상 MySQL 인스턴스 연결이 끊김](#) 단원을 참조하십시오.

max_statement_time 데이터베이스 파라미터 검증

API 키: mysql-check-max-statement-time

소스 파라미터 - MySQL 기반 소스용 max_Statement_time을 확인합니다. 10억보다 큰 테이블이 있는 경우 max_Statement_time 값을 검증하고 잠재적인 데이터 손실을 방지하려면 더 높은 값으로 설정하는 것이 좋습니다.

배치 적용 대상에 프라이머리 키 또는 고유 인덱스가 있는지 확인

API 키: mysql-check-batch-apply-target-pk-ui-absence

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 동시에 존재하는 테이블은 배치가 실패하며 변경 사항이 하나씩 처리됩니다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키를 생성할 수 있습니다.

자세한 내용은 [AWS Database Migration Service용 대상으로 MySQL 호환 데이터베이스 사용](#)을 참조하세요.

배치 적용 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 확인

API 키: mysql-check-batch-apply-target-pk-ui-simultaneously

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 동시에 있는 테이블은 배치 처리 시 실패하며, 변경 사항은 하나씩 처리됩니다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키(들) 또는 프라이머리 키를 삭제하고 마이그레이션을 수행하는 경우 다시 빌드할 수 있습니다.

자세한 내용은 [AWS Database Migration Service용 대상으로 MySQL 호환 데이터베이스 사용](#)을 참조하세요.

전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 활성화되었는지 확인

API 키: mysql-check-secondary-indexes

대상 데이터베이스에서 보조 인덱스를 비활성화하거나 제거하는 것이 좋습니다. 보조 인덱스는 전체 로드 중에 마이그레이션 성능에 영향을 미칠 수 있습니다. 캐시된 변경 사항을 적용하기 전에 보조 인덱스를 활성화하는 것이 좋습니다.

자세한 내용은 [AWS Database Migration Service모범 사례](#)를 참조하세요.

DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있는지 검증

API 키: `mysql-check-pk-validity`

데이터를 검증하려면 테이블에 프라이머리 키나 고유한 인덱스가 있어야 합니다.

자세한 내용은 [AWS DMS 데이터 검증](#) 섹션을 참조하세요.

MaxFullLoadSubTasks 설정 사용에 대한 권장 사항

API 키: `mysql-tblnum-for-max-fullload-subtasks`

이 평가는 작업에 포함된 테이블 수를 확인하고 전체 로드 프로세스 중에 최적의 성능을 위해 MaxFullLoadSubTasks 파라미터를 늘릴 것을 권장합니다. 기본적으로는 8개의 테이블을 동시에 AWS DMS 마이그레이션합니다. MaxFullLoadSubTasks 파라미터를 더 높은 값으로 변경하면 전체 로드 성능이 향상됩니다.

자세한 내용은 [전체 로드 작업 설정](#) 섹션을 참조하세요.

Digits Randomize 변환 규칙 확인

API 키: `mysql-datamasking-digits-randomize`

이 평가는 테이블 매핑에 사용되는 열이 Digits Randomize 변환 규칙과 호환되는지 확인합니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

Digits Mask에 대한 변환 규칙 확인

API 키: `mysql-datamasking-digits-mask`

이 평가는 테이블 매핑에 사용되는 열이 Digits Mask 변환 규칙에서 지원되지 않는지 확인합니다. 또한 Digits Mask 변환을 해당 열에 적용하면 고유성을 보장할 수 없으므로 DMS 작업 실패가 발생할 수 있으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

해싱 마스크에 대한 변환 규칙 확인

API 키: `mysql-datamasking-hash-mask`

이 평가는 테이블 매핑에 사용되는 열 중 하나라도 해싱 마스크 변환 규칙에서 지원되지 않는지 확인합니다. 또한 소스 열의 길이가 64자를 초과하는지 확인합니다. 해시 마스킹을 지원하려면 대상 열 길이가 64자를 초과하는 것이 가장 좋습니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지

않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인

API 키: `all-to-all-validation-with-datamasking-digits-randomize`

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인

API 키: `all-to-all-validation-with-datamasking-hash-mask`

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인

API 키: `all-to-all-validation-with-digit-mask`

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

소스 Amazon Aurora MySQL 인스턴스가 읽기 전용 복제본이 아닌지 확인

API 키: `mysql-check-aurora-read-only`

이 마이그레이션 전 평가는 두 개의 Amazon Aurora MySQL 클러스터 간 마이그레이션 시, 소스 엔드포인트가 읽기/쓰기 인스턴스여야 하며, 복제 인스턴스일 수 없음을 검증합니다.

이진 로그 보존 시간이 올바르게 설정되었는지 확인

API 키: `mysql-check-binlog-retention-time`

이 마이그레이션 전 평가는 'binlog retention hours' 값이 24시간보다 큰지 확인합니다.

소스 테이블에 보이지 않는 열이 없는지 확인합니다.

API 키: `mysql-check-invisible-columns`

이 사전 마이그레이션 평가는 소스 테이블에 보이지 않는 열이 없는지 확인합니다.는 소스 데이터베이스의 보이지 않는 열에서 데이터를 마이그레이션하지 AWS DMS 않습니다.

DMS CDC를 지원하도록 데이터베이스 binlog 형식이 ROW로 설정되어 있는지 확인

API 키: `mysql-check-binlog-format`

이 마이그레이션 전 평가는 소스 데이터베이스 binlog 형식이 DMS 변경 데이터 캡처(CDC)를 지원하도록 ROW로 설정되어 있는지 확인합니다. binlog 형식을 ROW로 설정하려면 다음을 수행합니다.

- Amazon RDS의 경우 데이터베이스의 파라미터 그룹을 사용합니다. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 [단일 AZ 데이터베이스에 대한 MySQL 바이너리 로깅 구성](#)을 참조하세요.
- 온프레미스 또는 Amazon EC2에서 호스팅되는 데이터베이스의 경우 `binlog_format` 값을 `my.ini`(Microsoft Windows) 또는 `my.cnf` (UNIX)로 설정합니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다. 자체 호스트형 MySQL 서버에 대한 자세한 내용은 [자체 관리형 MySQL 호환 데이터베이스를 AWS DMS의 소스로 사용](#)을 참조하세요.

선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.

API 키: `all-check-source-selection-rules`

이 사전 마이그레이션 평가는 와일드카드 기반 규칙에 대한 패턴 일치를 포함하여 선택 규칙에 지정된 하나 이상의 객체가 소스 데이터베이스에 존재하는지 확인합니다.

생성된 열이 있는 테이블이 소스 데이터베이스에 존재하는지 검증

API 키: `mysql-check-generated-columns`

이 사전 마이그레이션 평가는 MySQL 소스 테이블에 생성된 열이 있는지 확인합니다. AWS DMS 작업은 생성된 열을 마이그레이션하거나 복제하지 않습니다. 생성된 열을 마이그레이션하는 방법에 대한 자세한 내용은 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요.

분할된 테이블에 대해 `skipTableSuspensionForPartitionDdl`이 활성화되어 있는지 확인합니다.

API 키: `mysql-check-skip-table-suspension-partition-ddl`

이 사전 마이그레이션 평가는 소스 데이터베이스에서 분할된 테이블을 감지하고 `skipTableSuspensionForPartitionDdl` 파라미터 설정을 확인합니다. 이 파라미터를 설정하지 않으면 마이그레이션 중에 테이블이 불필요하게 일시 중지될 수 있습니다. 자세한 내용은 링크를 참조하세요 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#).

max_allowed_packet 크기가 소스 LOB 열을 처리할 수 있는지 확인

API 키: mysql-check-max-allowed-packet-lob

AWS DMS 는 현재 max_allowed_packet 설정을 초과하는 소스 테이블의 LOB 열을 감지합니다. 이러한 불일치로 인해 데이터 마이그레이션 중에 복제에 실패할 수 있습니다. 자세한 내용은 [MySQL 관련 문제 해결](#) 단원을 참조하십시오.

보조 제약 조건 및 인덱스(비기본)가 소스 데이터베이스에 있는지 검증합니다.

API 키: all-check-secondary-constraints

이 사전 마이그레이션 평가는 보조 제약 조건 및 인덱스(외래 키, 검사 제약 조건, 클러스터링되지 않은 인덱스)가 소스 데이터베이스에 있는지 확인합니다.

MariaDB 평가

이 섹션에서는 MariaDB 소스 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 마이그레이션 전 평가에 대해 설명합니다.

AWS DMS API를 사용하여 개별 사전 마이그레이션 평가를 생성하려면

[StartReplicationTaskAssessmentRun](#) 작업의 Include 파라미터에 나열된 API 키를 사용합니다.

주제

- [소스 데이터베이스에서 server_id가 1 이상으로 설정되어 있는지 확인](#)
- [소스 데이터베이스에 바이너리 로그의 자동 제거가 설정되어 있는지 확인](#)
- [BatchApplyEnabled가 true로 설정된 경우에만 제한된 LOB 모드가 사용되는지 확인](#)
- [바이너리 로그 트랜잭션 압축이 비활성화되었는지 확인](#)
- [DMS 사용자에게 소스 데이터베이스에 대한 REPLICATION CLIENT 및 REPLICATION SLAVE 권한이 있는지 확인](#)
- [DMS 사용자에게 소스 데이터베이스 테이블에 대한 SELECT 권한이 있는지 확인](#)
- [DMS 사용자에게 대상으로서의 MySQL 호환 데이터베이스에 필요한 권한이 있는지 확인](#)
- [테이블이 InnoDB 이외의 스토리지 엔진을 사용하는지 확인](#)
- [마이그레이션에 사용되는 테이블에 자동 증분이 활성화되어 있는지 확인](#)
- [DMS CDC를 지원하도록 데이터베이스 binlog 형식이 ROW로 설정되어 있는지 확인](#)
- [DMS CDC를 지원하도록 데이터베이스 binlog 이미지가 FULL로 설정되어 있는지 확인](#)

- [소스 데이터베이스가 MariaDB 읽기 전용 복제본인지 확인](#)
- [테이블에 파티션이 있는지 확인하고 전체 로드 작업 설정에 TRUNCATE_BEFORE_LOAD 또는 DO_NOTHING을 권장합니다.](#)
- [DMS가 데이터베이스 버전을 지원하는지 확인](#)
- [대상 데이터베이스가 local_infile을 1로 설정하도록 구성되어 있는지 확인](#)
- [대상 데이터베이스에 외부 키가 있는 테이블이 있는지 확인](#)
- [태스크 범위의 소스 테이블에 캐스케이드 제약 조건이 있는지 확인](#)
- [태스크 범위의 소스 테이블에 열이 생성되었는지 확인](#)
- [제한 시간 값이 MariaDB 소스에 적합한지 확인](#)
- [제한 시간 값이 MariaDB 대상에 적합한지 확인](#)
- [max_statement_time 데이터베이스 파라미터 검증](#)
- [배치 적용 대상에 프라이머리 키 또는 고유 인덱스가 있는지 확인](#)
- [배치 적용 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 확인](#)
- [전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 활성화되었는지 확인](#)
- [DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있는지 검증](#)
- [MaxFullLoadSubTasks 설정 사용에 대한 권장 사항](#)
- [Digits Randomize 변환 규칙 확인](#)
- [Digits Mask에 대한 변환 규칙 확인](#)
- [해싱 마스크에 대한 변환 규칙 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인](#)
- [이진 로그 보존 시간이 올바르게 설정되었는지 확인](#)
- [소스 테이블에 보이지 않는 열이 없는지 확인](#)
- [선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.](#)
- [분할된 테이블에 대해 skipTableSuspensionForPartitionDdl가 활성화되어 있는지 확인](#)
- [보조 제약 조건 및 인덱스\(비기본\)가 소스 데이터베이스에 있는지 검증합니다.](#)

소스 데이터베이스에서 **server_id**가 1 이상으로 설정되어 있는지 확인

API 키: mariadb-check-server-id

이 마이그레이션 전 평가는 CDC 마이그레이션 유형에 대한 소스 데이터베이스에서 `server_id` 서버 변수가 1 이상으로 설정되었는지 확인합니다.

MariaDB 엔드포인트 제한에 대한 자세한 내용은 [자체 관리형 MySQL 호환 데이터베이스를 AWS DMS의 소스로 사용](#)을 참조하세요.

소스 데이터베이스에 바이너리 로그의 자동 제거가 설정되어 있는지 확인

API 키: `mariadb-check-expire-logs-days`

이 마이그레이션 전 평가는 데이터베이스가 바이너리 로그를 자동으로 제거하도록 구성되어 있는지 확인합니다. 마이그레이션 중에 디스크 공간이 과도하게 사용되지 않도록 `EXPIRE_LOGS_DAYS` 또는 `BINLOG_EXPIRE_LOGS_SECONDS` 전역 시스템 변수의 값은 0보다 커야 합니다.

MariaDB 엔드포인트 제한에 대한 자세한 내용은 [자체 관리형 MySQL 호환 데이터베이스를 AWS DMS의 소스로 사용](#)을 참조하세요.

BatchApplyEnabled가 true로 설정된 경우에만 제한된 LOB 모드가 사용되는지 확인

API 키: `mariadb-batch-apply-lob-mode`

복제에 LOB 열이 포함되면 BatchApplyEnabled를 제한된 LOB 모드에서만 사용할 수 있습니다. LOB 모드의 다른 옵션을 사용하면 배치가 실패하고 변경 사항이 하나씩 AWS DMS 처리됩니다. 이러한 테이블을 자체 태스크로 이동하고 대신 트랜잭션 적용 모드를 사용하는 것이 좋습니다.

BatchApplyEnabled 설정에 대한 자세한 내용은 [DMS 배치 적용 기능을 사용하여 CDC 복제 성능을 개선하려면 어떻게 해야 하나요?](#)를 참조하세요.

바이너리 로그 트랜잭션 압축이 비활성화되었는지 확인

API 키: `mariadb-check-binlog-compression`

이 사전 마이그레이션 평가는 이진 로그 트랜잭션 압축이 비활성화되었는지 확인합니다. AWS DMS는 이진 로그 트랜잭션 압축을 지원하지 않습니다.

자세한 내용은 [MySQL 데이터베이스를 AWS DMS의 소스로 사용하는 것에 대한 제한 사항](#)을 참조하세요.

DMS 사용자에게 소스 데이터베이스에 대한 REPLICATION CLIENT 및 REPLICATION SLAVE 권한이 있는지 확인

API 키: `mariadb-check-replication-privileges`

이 마이그레이션 전 평가는 DMS 작업 마이그레이션 유형이 CDC 또는 전체 로드 + CDC인 경우 소스 엔드포인트 연결 설정에 지정된 DMS 사용자에게 소스 데이터베이스에 대한 REPLICATION CLIENT 및 REPLICATION SLAVE 권한이 있는지 확인합니다.

자세한 내용은 [MySQL 호환 데이터베이스를 AWS DMS의 원본으로 사용](#)을 참조하세요.

DMS 사용자에게 소스 데이터베이스 테이블에 대한 SELECT 권한이 있는지 확인

API 키: mariadb-check-select-privileges

이 마이그레이션 전 평가는 소스 엔드포인트 연결 설정에 지정된 DMS 사용자에게 소스 데이터베이스 테이블에 대한 SELECT 권한이 있는지 확인합니다.

자세한 내용은 [MySQL 호환 데이터베이스를 AWS DMS의 원본으로 사용](#)을 참조하세요.

DMS 사용자에게 대상으로서의 MySQL 호환 데이터베이스에 필요한 권한이 있는지 확인

API 키: mariadb-check-target-privileges

이 마이그레이션 전 평가는 대상 엔드포인트 연결 설정에 지정된 DMS 사용자에게 MySQL 호환 데이터베이스에 필요한 권한이 대상인지 여부를 검증합니다.

자세한 내용은 [MySQL 호환 데이터베이스를 AWS DMS의 원본으로 사용](#)을 참조하세요.

테이블이 InnoDB 이외의 스토리지 엔진을 사용하는지 확인

API 키: mariadb-check-table-storage-engine

이 마이그레이션 전 평가는 소스 MariaDB 데이터베이스의 테이블에 사용되는 스토리지 엔진이 InnoDB 이외의 엔진인지 확인합니다. 기본적으로 DMS에서는 InnoDB 스토리지 엔진이 있는 대상 테이블을 생성합니다. InnoDB 이외의 스토리지 엔진을 사용해야 하는 경우 대상 데이터베이스에 테이블을 수동으로 생성하고 TRUNCATE_BEFORE_LOAD 또는 DO_NOTHING을 전체 로드 태스크 설정으로 사용하도록 DMS 태스크를 구성해야 합니다. 전체 로드 태스크 설정에 관한 자세한 내용은 [전체 로드 작업 설정](#) 섹션을 참조하세요.

MariaDB 엔드포인트 제한 사항에 대한 자세한 내용은 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요.

마이그레이션에 사용되는 테이블에 자동 증분이 활성화되어 있는지 확인

API 키: mariadb-check-auto-increment

이 마이그레이션 전 평가는 태스크에 사용되는 소스 테이블에 자동 증분이 활성화되어 있는지 확인합니다. DMS에서는 열의 AUTO_INCREMENT 속성은 대상 데이터베이스로 마이그레이션하지 않습니다.

MariaDB 엔드포인트 제한 사항에 대한 자세한 내용은 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요. MariaDB의 자격 증명 열 처리에 대한 자세한 내용은 [IDENTITY 열 처리 AWS DMS: 2부를 참조하세요](#).

DMS CDC를 지원하도록 데이터베이스 binlog 형식이 **ROW**로 설정되어 있는지 확인

API 키: mariadb-check-binlog-format

이 마이그레이션 전 평가는 소스 데이터베이스 binlog 형식이 DMS 변경 데이터 캡처(CDC)를 지원하도록 ROW로 설정되어 있는지 확인합니다.

binlog 형식을 ROW로 설정하려면 다음을 수행합니다.

- Amazon RDS의 경우 데이터베이스의 파라미터 그룹을 사용합니다. RDS 파라미터 그룹 사용에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [MySQL 바이너리 로깅 구성](#)을 참조하세요.
- 온프레미스 또는 Amazon EC2에서 호스팅되는 데이터베이스의 경우 binlog_format 값을 my.ini(Microsoft Windows) 또는 my.cnf(UNIX)로 설정합니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자체 호스팅 MariaDB 서버에 대한 자세한 내용은 [자체 관리형 MySQL 호환 데이터베이스들의 소스로 사용 AWS DMS](#) 섹션을 참조하세요.

DMS CDC를 지원하도록 데이터베이스 binlog 이미지가 **FULL**로 설정되어 있는지 확인

API 키: mariadb-check-binlog-image

이 마이그레이션 전 평가는 소스 데이터베이스의 binlog 이미지가 FULL로 설정되어 있는지 확인합니다. MariaDB에서 binlog_row_image 변수는 ROW 형식을 사용할 때 바이너리 로그 이벤트가 기록되는 방식을 결정합니다. DMS와의 호환성을 보장하고 CDC를 지원하려면 binlog_row_image 변수를 FULL로 설정합니다. 이 설정을 통해 DMS는 마이그레이션 중에 대상 데이터베이스에 대한 전체 데이터 조작 언어(DML)를 구성할 수 있는 충분한 정보를 수신할 수 있습니다.

binlog 이미지를 FULL로 설정하려면 다음을 수행합니다.

- Amazon RDS의 경우 이 값은 기본적으로 FULL입니다.

- 온프레미스 또는 Amazon EC2에서 호스팅되는 데이터베이스의 경우 `binlog_row_image` 값을 `my.ini`(Microsoft Windows) 또는 `my.cnf`(UNIX)로 설정합니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자체 호스팅 MariaDB 서버에 대한 자세한 내용은 [자체 관리형 MySQL 호환 데이터베이스들의 소스로 사용 AWS DMS](#) 섹션을 참조하세요.

소스 데이터베이스가 MariaDB 읽기 전용 복제본인지 확인

API 키: `mariadb-check-database-role`

이 마이그레이션 전 평가는 소스 데이터베이스가 읽기 전용 복제본인지 확인합니다. 읽기 전용 복제본에 연결할 때 DMS에 대한 CDC 지원을 활성화하려면 `log_slave_updates` 파라미터를 True로 설정합니다. 자체 관리형 MySQL 데이터베이스 사용에 대한 자세한 내용은 [자체 관리형 MySQL 호환 데이터베이스들의 소스로 사용 AWS DMS](#) 섹션을 참조하세요.

`log_slave_updates` 값을 True로 설정하려면 다음을 수행합니다.

- Amazon RDS의 경우 데이터베이스의 파라미터 그룹을 사용합니다. RDS 데이터베이스 파라미터 그룹 사용에 대한 자세한 내용은 Amazon RDS 사용 설명서의 [파라미터 그룹 작업](#)을 참조하세요.
- 온프레미스 또는 Amazon EC2에서 호스팅되는 데이터베이스의 경우 `log_slave_updates` 값을 `my.ini`(Microsoft Windows) 또는 `my.cnf`(UNIX)로 설정합니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

테이블에 파티션이 있는지 확인하고 전체 로드 작업 설정에 **TRUNCATE_BEFORE_LOAD** 또는 **DO_NOTHING**을 권장합니다.

API 키: `mariadb-check-table-partition`

이 마이그레이션 전 평가는 소스 데이터베이스에 파티션이 있는 테이블이 있는지 확인합니다. DMS는 MariaDB 대상에 파티션이 없는 테이블을 생성합니다. 파티션된 테이블을 대상의 파티션된 테이블로 마이그레이션하려면 다음을 수행해야 합니다.

- 대상 MariaDB 데이터베이스에 파티션된 테이블을 미리 생성합니다.
- **TRUNCATE_BEFORE_LOAD** 또는 **DO_NOTHING**을 전체 로드 태스크 설정으로 사용하도록 DMS 태스크를 구성합니다.

MariaDB 엔드포인트 제한 사항에 대한 자세한 내용은 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#) 섹션을 참조하세요.

DMS가 데이터베이스 버전을 지원하는지 확인

API 키: mariadb-check-supported-version

이 마이그레이션 전 평가는 소스 데이터베이스 버전이 DMS와 호환되는지 확인합니다. CDC는 Amazon RDS MariaDB 버전 10.4 이하 또는 MySQL 버전 10.11 이상에서는 지원되지 않습니다. 지원되는 MariaDB 버전에 대한 자세한 내용은 [데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트](#) 섹션을 참조하세요.

대상 데이터베이스가 **local_infile**을 1로 설정하도록 구성되어 있는지 확인

API 키: mariadb-check-target-localinfile-set

이 마이그레이션 전 평가는 대상 데이터베이스의 local_infile 파라미터가 1로 설정되었는지 확인합니다. DMS를 사용하려면 대상 데이터베이스에 전체 로드하는 동안 'local_infile' 파라미터를 1로 설정해야 합니다. 자세한 내용은 [를 사용하여 MySQL에서 MySQL로 마이그레이션 AWS DMS](#) 단원을 참조하십시오.

이 평가는 전체 로드 태스크에만 유효합니다.

대상 데이터베이스에 외부 키가 있는 테이블이 있는지 확인

API 키: mariadb-check-fk-target

이 마이그레이션 전 평가는 MariaDB 데이터베이스로 마이그레이션하는 전체 로드 또는 전체 및 CDC 태스크에 외부 키가 있는 테이블이 있는지 확인합니다. DMS의 기본 설정은 알파벳순으로 테이블을 로드하는 것입니다. 상위 테이블과 하위 테이블이 동시에 로드되지 않을 수 있으므로 외부 키와 참조 무결성 제약이 있는 테이블은 로드가 실패할 수 있습니다.

DMS의 참조 무결성에 대한 자세한 내용은 [AWS DMS 마이그레이션 성능 개선](#) 주제의 인덱스, 트리거 및 참조 무결성 제약 조건 작업을 참조하세요.

태스크 범위의 소스 테이블에 캐스케이드 제약 조건이 있는지 확인

API 키: mariadb-check-cascade-constraints

이 마이그레이션 전 평가는 MariaDB 소스 테이블에 캐스케이드 제약 조건이 있는지 확인합니다. MariaDB는 이러한 이벤트에 대한 변경 사항을 binlog에 기록하지 않기 때문에 캐스케이드 제약 조건은

DMS 작업으로 마이그레이션되거나 복제되지 않습니다. AWS DMS 는 이러한 제약 조건을 지원하지 않지만 관계형 데이터베이스 대상에 대한 해결 방법을 사용할 수 있습니다.

캐스캐이드 제약 조건 및 기타 제약 조건 지원에 대한 자세한 내용은 AWS DMS의 마이그레이션 태스크 문제 해결 주제의 [인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음](#) 섹션을 참조하세요.

태스크 범위의 소스 테이블에 열이 생성되었는지 확인

API 키: mariadb-check-generated-columns

이 마이그레이션 전 평가는 MariaDB 소스 테이블에 열이 생성되었는지 확인합니다. DMS 태스크는 생성된 열을 마이그레이션하거나 복제하지 않습니다.

생성된 열을 마이그레이션하는 방법에 대한 자세한 내용은 [작업 중에 대상 MySQL 인스턴스 연결이 끊김](#) 섹션을 참조하세요.

제한 시간 값이 MariaDB 소스에 적합한지 확인

API 키: mariadb-check-source-network-parameter

이 마이그레이션 전 평가는 작업의 MariaDB 소스 엔드포인트에 net_read_timeout, net_write_timeout 및 wait_timeout 설정이 300초 이상으로 설정되어 있는지 확인합니다. 이는 마이그레이션 중에 연결이 끊어지지 않도록 하기 위해 필요합니다.

자세한 내용은 [작업 중에 대상 MySQL 인스턴스 연결이 끊김](#) 단원을 참조하십시오.

제한 시간 값이 MariaDB 대상에 적합한지 확인

API 키: mariadb-check-target-network-parameter

이 마이그레이션 전 평가에서는 작업의 MariaDB 대상 엔드포인트에 net_read_timeout, net_write_timeout 및 wait_timeout 설정이 300초 이상으로 설정되어 있는지 확인합니다. 이는 마이그레이션 중에 연결이 끊어지지 않도록 하기 위해 필요합니다.

자세한 내용은 [작업 중에 대상 MySQL 인스턴스 연결이 끊김](#) 단원을 참조하십시오.

max_statement_time 데이터베이스 파라미터 검증

API 키: mariadb-check-max-statement-time

AWS DMS 는 데이터베이스 소스 파라미터 max_statement_time가 0 이외의 값으로 설정되어 있는지 확인합니다. DMS 전체 로드 프로세스를 수용하려면 이 파라미터를 0으로 설정해야 합니다. 전체 로

드가 완료된 후 파라미터 값을 0 이외의 값으로 설정하면 데이터가 손실될 수 있으므로 파라미터 값을 변경하는 것을 고려할 수 있습니다.

배치 적용 대상에 프라이머리 키 또는 고유 인덱스가 있는지 확인

API 키: mariadb-check-batch-apply-target-pk-ui-absence

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 없는 테이블은 배치가 실패하고 변경 사항을 하나씩 처리합니다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키를 생성할 수 있습니다.

자세한 내용은 [the section called “MySQL을 대상으로 사용”](#) 단원을 참조하십시오.

배치 적용 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 확인

API 키: mariadb-check-batch-apply-target-pk-ui-simultaneously

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 동시에 존재하는 테이블은 배치가 실패하며 변경 사항이 하나씩 처리됩니다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키(들) 또는 프라이머리 키를 삭제하고 마이그레이션을 수행하는 경우 다시 빌드할 수 있습니다.

자세한 내용은 [the section called “MySQL을 대상으로 사용”](#) 단원을 참조하십시오.

전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 활성화되었는지 확인

API 키: mariadb-check-secondary-indexes

대상 데이터베이스에서 보조 인덱스를 비활성화하거나 제거하는 것을 고려해야 합니다. 보조 인덱스는 전체 로드 중에 마이그레이션 성능에 영향을 미칠 수 있습니다. 캐시된 변경 사항을 적용하기 전에 보조 인덱스를 활성화하는 것이 좋습니다.

자세한 내용은 [모범 사례](#) 단원을 참조하십시오.

DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있는지 검증

API 키: mariadb-check-pk-validity

데이터를 검증하려면 테이블에 프라이머리 키나 고유한 인덱스가 있어야 합니다.

자세한 내용은 [데이터 유효성 검사](#) 단원을 참조하십시오.

MaxFullLoadSubTasks 설정 사용에 대한 권장 사항

이 평가는 작업에 포함된 테이블 수를 확인하고 전체 로드 프로세스 중에 최적의 성능을 위해 MaxFullLoadSubTasks 파라미터를 늘릴 것을 권장합니다. 기본적으로는 8개의 테이블을 동시에 AWS DMS 마이그레이션합니다. MaxFullLoadSubTasks 파라미터를 더 높은 값으로 변경하면 전체 로드 성능이 향상됩니다.

자세한 내용은 [the section called “전체 로드 작업 설정”](#) 단원을 참조하십시오.

Digits Randomize 변환 규칙 확인

API 키: mariadb-datamasking-digits-randomize

이 평가는 테이블 매핑에 사용되는 열이 Digits Randomize 변환 규칙과 호환되는지 확인합니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

Digits Mask에 대한 변환 규칙 확인

API 키: mariadb-datamasking-digits-mask

이 평가는 테이블 매핑에 사용되는 열이 Digits Mask 변환 규칙에서 지원되지 않는지 확인합니다. 또한 Digits Mask 변환을 해당 열에 적용하면 고유성을 보장할 수 없으므로 DMS 작업 실패가 발생할 수 있으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

해싱 마스크에 대한 변환 규칙 확인

API 키: mariadb-datamasking-hash-mask

이 평가는 테이블 매핑에 사용되는 열 중 하나라도 해싱 마스크 변환 규칙에서 지원되지 않는지 확인합니다. 또한 소스 열의 길이가 64자를 초과하는지 확인합니다. 해시 마스크를 지원하려면 대상 열 길이가 64자를 초과하는 것이 가장 좋습니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

데이터 검증 작업 설정과 데이터 마스크 Digits Randomize가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-datamasking-digits-randomize

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스크 Digits Randomize가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인

API 키: `all-to-all-validation-with-datamasking-hash-mask`

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인

API 키: `all-to-all-validation-with-digit-mask`

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

이진 로그 보존 시간이 올바르게 설정되었는지 확인

API 키: `mariadb-check-binlog-retention-time`

이 마이그레이션 전 평가는 'binlog retention hours' 값이 24시간보다 큰지 확인합니다.

소스 테이블에 보이지 않는 열이 없는지 확인

API 키: `mariadb-check-invisible-columns`

이 사전 마이그레이션 평가는 소스 테이블에 보이지 않는 열이 없는지 확인합니다.는 소스 데이터베이스의 보이지 않는 열에서 데이터를 마이그레이션하지 AWS DMS 않습니다.

선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.

API 키: `all-check-source-selection-rules`

이 사전 마이그레이션 평가는 와일드카드 기반 규칙에 대한 패턴 일치를 포함하여 선택 규칙에 지정된 하나 이상의 객체가 소스 데이터베이스에 존재하는지 확인합니다.

분할된 테이블에 대해 **skipTableSuspensionForPartitionDdl**가 활성화되어 있는지 확인

API 키: `mariadb-check-skip-table-suspension-partition-ddl`

이 사전 마이그레이션 평가는 소스 데이터베이스에서 분할된 테이블을 감지하고 `skipTableSuspensionForPartitionDdl` 파라미터 설정을 확인합니다. 이 파라미터를 설정하지 않으면 마이그레이션 중에 테이블이 불필요하게 일시 중지될 수 있습니다.

자세한 내용은 [MySQL 데이터베이스를 소스로 사용할 때의 제한 사항 AWS DMS](#) 단원을 참조하십시오.

보조 제약 조건 및 인덱스(비기본)가 소스 데이터베이스에 있는지 검증합니다.

API 키: `all-check-secondary-constraints`

이 사전 마이그레이션 평가는 보조 제약 조건 및 인덱스(외래 키, 검사 제약 조건, 클러스터링되지 않은 인덱스)가 소스 데이터베이스에 있는지 확인합니다.

PostgreSQL 평가

이 섹션에서는 PostgreSQL 소스 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 마이그레이션 전 평가에 대해 설명합니다.

주제

- [DDL 이벤트 트리거가 ENABLE ALWAYS로 설정되어 있는지 확인](#)
- [소스 데이터베이스에 PostGIS 열이 있는지 확인](#)
- [전체 로드 프로세스 중에 대상 테이블에서 외부 키 제약 조건이 비활성화되었는지 확인](#)
- [이름이 유사한 테이블이 존재하는지 확인](#)
- [프라이머리 키가 없는 ARRAY 데이터 유형의 테이블이 있는지 확인](#)
- [BatchApplyEnabled가 활성화된 경우 대상 테이블에 프라이머리 키 또는 고유 인덱스가 존재하는지 확인](#)
- [대상 데이터베이스의 테이블에 전체 로드 마이그레이션 태스크에 대한 보조 인덱스가 있는지 확인](#)
- [BatchApplyEnabled가 true로 설정된 경우에만 제한된 LOB 모드가 사용되는지 확인](#)
- [마이그레이션을 위해 소스 데이터베이스 버전이 DMS에서 지원되는지 확인](#)
- [소스 데이터베이스에서 logical_decoding_work_mem 파라미터 확인](#)
- [소스 데이터베이스에 오래 실행되는 트랜잭션이 있는지 확인](#)
- [소스 데이터베이스 파라미터 max_slot_wal_keep_size 확인](#)
- [소스 데이터베이스 파라미터 postgres-check-maxwalsenders가 CDC를 지원하도록 설정되어 있는지 확인합니다.](#)
- [소스 데이터베이스가 PGLOGICAL에 대해 구성되어 있는지 확인](#)
- [소스 테이블 프라이머리 키가 LOB 데이터 유형인지 확인](#)
- [소스 테이블에 프라이머리 키가 있는지 확인](#)
- [소스 데이터베이스에 준비된 트랜잭션이 있는지 확인](#)
- [wal_sender_timeout이 DMS CDC를 지원하는 데 필요한 최소 값으로 설정되어 있는지 확인](#)
- [wal_level이 소스 데이터베이스에서 logical로 설정되어 있는지 확인](#)

- [배치 적용 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 확인](#)
- [LOB 객체가 발견될 때 최대 LOB 설정 권장](#)
- [DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있고 상태가 양호한지 검증](#)
- [AWS DMS 사용자에게 대상에 필요한 권한이 있는지 확인](#)
- [CDC에 대한 무료 복제 슬롯의 가용성을 검증합니다](#)
- [DMS 사용자 전체 로드 권한 확인](#)
- [Digits Randomize 변환 규칙 확인](#)
- [Digits Mask에 대한 변환 규칙 확인](#)
- [해싱 마스크에 대한 변환 규칙 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인](#)
- [선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.](#)
- [대상 PostgreSQL 데이터베이스에 생성된 열이 포함되어 있는지 확인](#)
- [구체화된 뷰가 동종 PostgreSQL 마이그레이션에 존재하는지 검증](#)
- [REPLICA IDENTITY FULL이 pglogical 플러그인 사용과 충돌하는지 확인](#)
- [보조 제약 조건 및 인덱스\(비기본\)가 소스 데이터베이스에 있는지 검증합니다.](#)
- [Oracle로 마이그레이션하기 위한 CHAR/VARCHAR 열 호환성 검증](#)
- [소스 데이터베이스에 idle_in_transaction_session_timeout 설정이 구성되어 있는지 확인](#)
- [AWS DMS 사용자에게 AWS관리형 PostgreSQL 데이터베이스에 필요한 역할이 있는지 확인](#)
- [대상 엔드포인트가 읽기 전용 복제본이 아닌지 확인](#)
- [소스 Aurora PostgreSQL 읽기 전용 복제본 버전 확인](#)
- [소스 PostgreSQL 읽기 전용 복제본 버전 확인](#)

DDL 이벤트 트리거가 ENABLE ALWAYS로 설정되어 있는지 확인

API 키: `postgres-check-ddl-event-trigger`

이 마이그레이션 전 평가는 DDL 이벤트 트리거가 ENABLE ALWAYS로 설정되어 있는지 확인합니다. 소스 데이터베이스가 다른 타사 복제 시스템의 대상이기도 한 경우, CDC 중에는 DDL 변경 내용이 마이그레이션되지 않을 수 있습니다. 이 상황에서는 DMS가 `awsdms_intercept_ddl` 이벤트를 트리거하지 못할 수 있습니다. 이 상황을 해결하려면 소스 데이터베이스에서 해당 트리거를 다음 예제와 같이 수정하세요.


```
alter event trigger awsdms_intercept_ddl enable always;
```

자세한 내용은 [PostgreSQL 데이터베이스를 DMS의 소스로 사용하는 것에 대한 제한 사항](#)을 참조하세요.

소스 데이터베이스에 PostGIS 열이 있는지 확인

API 키: postgres-check-postgis-data-type

이 사전 마이그레이션 평가는 사례 소스 엔진과 대상 엔진에 있는 PostGIS 데이터 형식의 열이 다른지 확인합니다. 동종(like-to-like) 마이그레이션에 대해서만 PostGIS 데이터 형식을 AWS DMS 지원합니다.

자세한 내용은 [PostgreSQL 데이터베이스를 DMS의 소스로 사용하는 것에 대한 제한 사항](#)을 참조하세요.

전체 로드 프로세스 중에 대상 테이블에서 외부 키 제약 조건이 비활성화되었는지 확인

API 키: postgres-check-session-replication-role

이 마이그레이션 전 평가는 전체 로드 단계에서 외부 키 제약 조건을 비활성화하기 위해 session_replication_role parameter가 대상에서 REPLICA로 설정되어 있는지 확인합니다. 전체 로드 마이그레이션 유형의 경우 외부 키 제약 조건을 비활성화해야 합니다.

PostgreSQL 엔드포인트 제한에 대한 자세한 내용은 [PostgreSQL 데이터베이스를 AWS Database Migration Service의 대상으로 사용](#)을 참조하세요.

이름이 유사한 테이블이 존재하는지 확인

API 키: postgres-check-similar-table-name

이 마이그레이션 전 평가는 소스에 비슷한 이름의 테이블이 있는지 확인합니다. 동일한 이름의 테이블을 서로 다른 경우에 갖고 있으면 복제 중에 예측할 수 없는 동작이 발생할 수 있습니다.

PostgreSQL 엔드포인트 제한에 대한 자세한 내용은 [PostgreSQL 데이터베이스를 DMS 소스로 사용하는 제한 사항](#)을 참조하세요.

프라이머리 키가 없는 ARRAY 데이터 유형의 테이블이 있는지 확인

API 키: postgres-check-table-with-array

이 마이그레이션 전 평가는 프라이머리 키 없이 배열 데이터 유형이 있는 테이블이 있는지 확인합니다. 프라이머리 키가 누락된 ARRAY 데이터 유형의 테이블은 전체 로드 중에 무시됩니다.

PostgreSQL 엔드포인트 제한에 대한 자세한 내용은 [PostgreSQL 데이터베이스를 DMS 소스로 사용하는 제한 사항](#)을 참조하세요.

BatchApplyEnabled가 활성화된 경우 대상 테이블에 프라이머리 키 또는 고유 인덱스가 존재하는지 확인

API 키: postgres-check-batch-apply-target-pk-ui-absence

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 없는 테이블은 배치가 실패하고 변경 AWS DMS 사항을 하나씩 처리합니다. 이러한 테이블에 대해 별도의 태스크를 생성하고 대신 트랜잭션 적용 모드를 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키를 생성할 수 있습니다.

자세한 내용은 [PostgreSQL 데이터베이스를 AWS Database Migration Service 대상으로 사용](#)을 참조하세요.

대상 데이터베이스의 테이블에 전체 로드 마이그레이션 태스크에 대한 보조 인덱스가 있는지 확인

API 키: postgres-check-target-secondary-indexes

이 마이그레이션 전 평가는 전체 로드 마이그레이션 태스크 범위에 보조 인덱스가 있는 테이블이 있는지 확인합니다. 전체 로드 태스크 기간 동안 보조 인덱스를 삭제하는 것이 좋습니다.

자세한 내용은 [PostgreSQL 데이터베이스를 AWS Database Migration Service 대상으로 사용](#)을 참조하세요.

BatchApplyEnabled가 true로 설정된 경우에만 제한된 LOB 모드가 사용되는지 확인

API 키: postgres-batch-apply-lob-mode

복제에 LOB 열이 포함되면 BatchApplyEnabled를 제한된 LOB 모드에서만 사용할 수 있습니다. LOB 모드의 다른 옵션을 사용하면 배치가 실패하고 변경 AWS DMS 사항이 하나씩 처리됩니다. 이러한 테이블을 자체 태스크로 이동하고 대신 트랜잭션 적용 모드를 사용하는 것이 좋습니다.

BatchApplyEnabled 설정에 대한 자세한 내용은 [DMS 배치 적용 기능을 사용하여 CDC 복제 성능을 개선하려면 어떻게 해야 합니까?](#)를 참조하세요.

마이그레이션을 위해 소스 데이터베이스 버전이 DMS에서 지원되는지 확인

API 키: postgres-check-dbversion

이 사전 마이그레이션 평가는 소스 데이터베이스 버전이와 호환되는지 확인합니다 AWS DMS.

소스 데이터베이스에서 **logical_decoding_work_mem** 파라미터 확인

API 키: `postgres-check-for-logical-decoding-work-mem`

이 마이그레이션 전 평가에서는 소스 데이터베이스에서 `logical_decoding_work_mem` 파라미터를 조정하는 것이 좋습니다. 트랜잭션이 많고 트랜잭션이 오래 실행되거나 하위 트랜잭션이 많을 수 있는 데이터베이스에서는 논리적 디코딩 메모리 사용량이 증가하고 디스크에 유출해야 할 수 있습니다. 이로 인해 복제 중에 DMS 소스 지연 시간이 높아집니다. 이러한 시나리오에서는 `logical_decoding_work_mem`을 조정해야 할 수 있습니다. 이 파라미터는 PostgreSQL 버전 13 이상에서 지원됩니다.

소스 데이터베이스에 오래 실행되는 트랜잭션이 있는지 확인

API 키: `postgres-check-longrunningtxn`

이 마이그레이션 전 평가는 소스 데이터베이스에 10분 이상 지속된 장기 실행 트랜잭션이 있는지 확인합니다. 기본적으로 DMS는 태스크를 시작하는 동안 진행 중인 트랜잭션이 있는지 확인하기 때문에 태스크를 시작하지 못할 수 있습니다.

소스 데이터베이스 파라미터 **max_slot_wal_keep_size** 확인

API 키: `postgres-check-maxslot-wal-keep-size`

이 마이그레이션 전 평가는 `max_slot_wal_keep_size`에 대해 구성된 값을 확인합니다. `max_slot_wal_keep_size`가 기본값이 아닌 값으로 설정되면 필요한 WAL 파일을 제거하여 DMS 태스크가 실패할 수 있습니다.

소스 데이터베이스 파라미터 **postgres-check-maxwalsenders**가 CDC를 지원하도록 설정되어 있는지 확인합니다.

API 키: `postgres-check-maxwalsenders`

이 마이그레이션 전 평가는 소스 데이터베이스의 `max_wal_senders`에 대해 구성된 값을 확인합니다. 데이터 캡처 변경(CDC)을 지원하려면 DMS는 `max_wal_senders`를 1보다 크게 설정해야 합니다.

소스 데이터베이스가 **PGLOGICAL**에 대해 구성되어 있는지 확인

API 키: `postgres-check-pglogical`

이 마이그레이션 전 평가는 CDC에 대해 PGLLOGICAL을 지원하도록 `shared_preload_libraries` 값이 `pglogical`로 설정되어 있는지 확인합니다. 논리적 복제에 테스트 디코딩을 사용할 계획인 경우 이 평가를 무시할 수 있습니다.

소스 테이블 프라이머리 키가 LOB 데이터 유형인지 확인

API 키: postgres-check-pk-lob

이 마이그레이션 전 평가는 테이블의 프라이머리 키가 대형 객체(LOB) 데이터 유형인지 확인합니다. 소스 테이블에 프라이머리 키로 LOB 열이 있는 경우 DMS는 복제를 지원하지 않습니다.

소스 테이블에 프라이머리 키가 있는지 확인

API 키: postgres-check-pk

이 마이그레이션 전 평가는 태스크 범위에 사용되는 테이블에 프라이머리 키가 있는지 확인합니다. DMS는 소스 테이블에서 복제본 자격 증명이 full로 설정되지 않은 한 프라이머리 키가 없는 테이블에 대한 복제를 지원하지 않습니다.

소스 데이터베이스에 준비된 트랜잭션이 있는지 확인

API 키: postgres-check-preparedtxn

이 마이그레이션 전 평가는 소스 데이터베이스에 준비된 트랜잭션이 있는지 확인합니다. 복제 슬롯 생성은 소스 데이터베이스에 준비된 트랜잭션이 있는 경우 응답을 중지할 수 있습니다.

wal_sender_timeout이 DMS CDC를 지원하는 데 필요한 최소 값으로 설정되어 있는지 확인

API 키: postgres-check-walsenderstimeout

이 마이그레이션 전 평가는 wal_sender_timeout이 최소 10,000밀리초(10초)로 설정되어 있는지 확인합니다. CDC가 있는 DMS 태스크에는 최소 10,000밀리초(10초)가 필요하며, 값이 10,000보다 작으면 태스크가 실패합니다.

wal_level이 소스 데이터베이스에서 logical로 설정되어 있는지 확인

API 키: postgres-check-wallevel

이 마이그레이션 전 평가는 wal_level이 logical로 설정되어 있는지 확인합니다. DMS CDC가 작동하려면 소스 데이터베이스에서 이 파라미터를 활성화해야 합니다.

배치 적용 대상에 프라이머리 키와 고유 인덱스가 모두 존재하는지 확인

API 키: postgres-check-batch-apply-target-pk-ui-simultaneously

배치 적용은 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있는 테이블에서만 지원됩니다. 프라이머리 키 또는 고유 인덱스가 동시에 존재하는 테이블은 배치가 실패하며 변경 사항이 하나씩 처리됩니다.

다. 이러한 테이블을 자체 작업으로 이동하고 트랜잭션 적용 모드를 대신 사용하는 것이 좋습니다. 또는 대상 테이블에 고유한 키(들) 또는 프라이머리 키를 삭제하고 마이그레이션을 수행하는 경우 다시 빌드할 수 있습니다.

자세한 내용은 [the section called “자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화”](#) 단원을 참조하십시오.

LOB 객체가 발견될 때 최대 LOB 설정 권장

API 키: postgres-check-limited-lob-size

PostgreSQL의 LOB 크기 계산은 다른 엔진과 다릅니다. 데이터 잘림을 방지하려면 작업 설정에서 올바른 최대 LOB 크기를 설정했는지 확인합니다.

자세한 내용은 [데이터 유효성 검사](#) 단원을 참조하십시오.

DMS 검증이 활성화된 경우 테이블에 프라이머리 키 또는 고유 인덱스가 있고 상태가 양호한지 검증

API 키: postgres-check-pk-validity

데이터를 검증하려면 테이블에 프라이머리 키나 고유한 인덱스가 있어야 합니다.

자세한 내용은 [데이터 유효성 검사](#) 단원을 참조하십시오.

AWS DMS 사용자에게 대상에 필요한 권한이 있는지 확인

API 키: postgres-check-target-privileges

AWS DMS 사용자는 대상 데이터베이스에 최소한 db_owner 사용자 역할이 있어야 합니다.

자세한 내용은 [the section called “대상으로서 PostgreSQL의 보안 요구 사항”](#) 단원을 참조하십시오.

CDC에 대한 무료 복제 슬롯의 가용성을 검증합니다

API 키: postgres-check-replication-slots-count

이 평가는 CDC가 변경 사항을 복제하는 데 복제 슬롯을 사용할 수 있는지 확인합니다.

DMS 사용자 전체 로드 권한 확인

API 키: postgres-check-select-object-privileges

이 평가는 DMS 사용자에게 전체 로드 작업에 필요한 테이블에 대한 필수 SELECT 권한이 있는지 확인합니다.

Digits Randomize 변환 규칙 확인

API 키: postgres-datamasking-digits-randomize

이 평가는 테이블 매핑에 사용되는 열이 Digits Randomize 변환 규칙과 호환되는지 확인합니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

Digits Mask에 대한 변환 규칙 확인

API 키: postgres-datamasking-digits-mask

이 평가는 테이블 매핑에 사용되는 열이 Digits Mask 변환 규칙에서 지원되지 않는지 확인합니다. 또한 Digits Mask 변환을 해당 열에 적용하면 고유성을 보장할 수 없으므로 DMS 작업 실패가 발생할 수 있으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

해싱 마스크에 대한 변환 규칙 확인

API 키: postgres-datamasking-hash-mask

이 평가는 테이블 매핑에 사용되는 열 중 하나라도 해싱 마스크 변환 규칙에서 지원되지 않는지 확인합니다. 또한 소스 열의 길이가 64자를 초과하는지 확인합니다. 해싱 마스크를 지원하려면 대상 열 길이가 64자를 초과하는 것이 가장 좋습니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

데이터 검증 작업 설정과 데이터 마스크 Digits Randomize가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-datamasking-digits-randomize

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스크 Digits Randomize가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스크 해싱 마스크가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-datamasking-hash-mask

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스크 해싱 마스크가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스크 Digits Mask가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-digit-mask

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.

API 키: all-check-source-selection-rules

이 사전 마이그레이션 평가는 와일드카드 기반 규칙에 대한 패턴 일치를 포함하여 선택 규칙에 지정된 하나 이상의 객체가 소스 데이터베이스에 존재하는지 확인합니다.

대상 PostgreSQL 데이터베이스에 생성된 열이 포함되어 있는지 확인

API 키: postgres-check-target-generated-cols

이 사전 마이그레이션 평가는 대상 PostgreSQL 데이터베이스에 마이그레이션 중에 특별한 처리가 필요할 수 있는 생성된 열(STORED 및 VIRTUAL 유형 모두 포함)이 포함되어 있는지 확인합니다. 다른 열에서 값을 계산하는 생성된 열은 대상 PostgreSQL 버전과의 호환성과 마이그레이션 후 적절한 데이터 일관성을 보장하기 위해 특정 확인이 필요합니다.

구체화된 뷰가 동종 PostgreSQL 마이그레이션에 존재하는지 검증

API 키: postgres-check-materialized-views

PostgreSQL 데이터베이스 간에 마이그레이션할 때 구체화된 뷰를 마이그레이션 AWS DMS 할 수 없습니다. 마이그레이션 후 대상 데이터베이스에서 구체화된 뷰를 수동으로 생성해야 합니다.

자세한 내용은 [PostgreSQL 데이터베이스를 DMS 소스로 사용 시 제한 사항](#) 단원을 참조하십시오.

REPLICA IDENTITY FULL이 pglogical 플러그인 사용과 충돌하는지 확인

API 키: postgres-check-pglogical-replica-identity-full

이 사전 마이그레이션 평가는 REPLICA IDENTITY FULL을 사용하여 테이블을 감지합니다. test_decoding 플러그인을 사용하여 REPLICA IDENTITY FULL을 지원하지만 pglogical과 함께 사용하면 업데이트가 올바르게 복제되지 않습니다. REPLICA IDENTITY 설정을 DEFAULT/INDEX로 변경하거나 test_decoding 플러그인으로 전환하여 REPLICA IDENTITY FULL을 유지합니다.

자세한 내용은 [논리적 복제를 사용하여 변경 데이터 캡처\(CDC\) 활성화](#) 단원을 참조하십시오.

보조 제약 조건 및 인덱스(비기본)가 소스 데이터베이스에 있는지 검증합니다.

API 키: all-check-secondary-constraints

이 사전 마이그레이션 평가는 보조 제약 조건 및 인덱스(외래 키, 검사 제약 조건, 클러스터링되지 않은 인덱스)가 소스 데이터베이스에 있는지 확인합니다.

Oracle로 마이그레이션하기 위한 CHAR/VARCHAR 열 호환성 검증

API 키: postgres-to-oracle-check-varchar-columns

이 사전 마이그레이션 평가는 대상 데이터베이스에 사용되는 NCHAR/NVARCHAR2 데이터 유형 열이 소스 데이터베이스의 CHAR/VARCHAR 열과 호환되는지 확인합니다.

소스 데이터베이스에 **idle_in_transaction_session_timeout** 설정이 구성되어 있는지 확인

API 키: postgres-check-idle-in-transaction-session-timeout

이 사전 마이그레이션 평가는 소스 데이터베이스에서 **idle_in_transaction_session_timeout** 파라미터가 0으로 설정되지 않았는지 확인합니다.

AWS DMS 사용자에게 AWS관리형 PostgreSQL 데이터베이스에 필요한 역할이 있는지 확인

API 키: postgres-check-rds-roles

이 사전 마이그레이션 평가는 AWS DMS 사용자가 AWS관리형 PostgreSQL 데이터베이스에 필요한 모든 역할로 구성되었는지 확인합니다. 역할이 충분하지 않으면 마이그레이션 작업이 실패할 수 있습니다.

대상 엔드포인트가 읽기 전용 복제본이 아닌지 확인

API 키: all-check-target-read-replica

이 사전 마이그레이션 평가는 대상 엔드포인트가 읽기 전용 복제본으로 구성되지 않았는지 확인합니다. 대상 데이터베이스에 대한 쓰기 액세스 권한이 AWS DMS 필요하며 읽기 전용 복제본으로 복제할 수 없습니다.

소스 Aurora PostgreSQL 읽기 전용 복제본 버전 확인

API 키: postgres-aurora-check-source-replica-role-cdc

이 사전 마이그레이션 평가는 소스 엔드포인트가 버전 16 이상을 실행하는 Aurora PostgreSQL 읽기 전용 복제본을 사용하는지 확인합니다. CDC 작업에는 Aurora PostgreSQL이 16 이전 버전의 읽기 전용 노드에서 지원하지 않는 복제 슬롯이 필요합니다.

자세한 내용은 [읽기 전용 복제본을 PostgreSQL의 소스로 사용](#) 단원을 참조하십시오.

소스 PostgreSQL 읽기 전용 복제본 버전 확인

API 키: postgres-check-source-replica-role-cdc

이 사전 마이그레이션 평가는 소스 엔드포인트가 버전 16 이상을 실행하는 PostgreSQL 읽기 전용 복제본을 사용하는지 확인합니다. CDC 작업에는 PostgreSQL이 16 이전 버전의 읽기 전용 노드에서 지원하지 않는 복제 슬롯이 필요합니다.

자세한 내용은 [읽기 전용 복제본을 PostgreSQL의 소스로 사용](#) 단원을 참조하십시오.

Db2 LUW 평가

이 섹션에서는 Db2 LUW 소스 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 마이그레이션 전 평가에 대해 설명합니다.

주제

- [IBM DB2 LUW 데이터베이스가 복구 가능하도록 구성되어 있는지 확인합니다.](#)
- [DMS 사용자에게 전체 로드를 수행하는 데 필요한 소스 데이터베이스에 대한 권한이 있는지 확인](#)
- [DMS 사용자에게 소스 데이터베이스에서 CDC를 수행하는 데 필요한 권한이 있는지 확인](#)
- [소스 IBM DB2 LUW 소스 테이블에 Db2 XML 데이터 유형이 있는지 확인](#)
- [소스 IBM Db2 LUW 버전이에서 지원되는지 확인 AWS DMS](#)
- [대상 IBM Db2 LUW 버전이에서 지원되는지 확인 AWS DMS](#)
- [Digits Randomize 변환 규칙 확인](#)
- [Digits Mask에 대한 변환 규칙 확인](#)
- [해싱 마스크에 대한 변환 규칙 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인](#)
- [데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인](#)
- [배치 적용 호환성을 위해 대상 테이블에 올바른 인덱스 구성\(프라이머리 키 또는 고유 인덱스, 둘 다 아님\)이 있는지 확인합니다](#)
- [BatchApplyEnabled가 true로 설정된 경우에만 '제한된 LOB 모드'가 사용되는지 확인](#)
- [전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 비활성화되었는지 확인](#)
- [선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.](#)
- [보조 제약 조건 및 인덱스\(비기본\)가 소스 데이터베이스에 있는지 검증합니다.](#)

IBM DB2 LUW 데이터베이스가 복구 가능하도록 구성되어 있는지 확인합니다.

API 키: db2-check-archive-config-param

이 마이그레이션 전 평가는 Db2 LUW 데이터베이스에 LOGARCHMETH1 및 LOGARCHMETH2 데이터베이스 구성 파라미터가 하나 또는 둘 다 있고 ON으로 설정되어 있는지 확인합니다.

DMS 사용자에게 전체 로드를 수행하는 데 필요한 소스 데이터베이스에 대한 권한이 있는지 확인

API 키: db2-check-full-load-privileges

이 마이그레이션 전 평가는 전체 로드 작업을 위해 DMS 사용자가 소스 데이터베이스에서 필요한 모든 권한을 보유하고 있는지 검증합니다.

DMS 사용자에게 소스 데이터베이스에서 CDC를 수행하는 데 필요한 권한이 있는지 확인

API 키: db2-check-cdc-privileges

이 마이그레이션 전 평가는 CDC 작업을 위해 DMS 사용자가 소스 데이터베이스에서 필요한 모든 권한을 보유하고 있는지 검증합니다.

소스 IBM DB2 LUW 소스 테이블에 Db2 XML 데이터 유형이 있는지 확인

API 키: db2-check-xml-data-type

이 마이그레이션 전 평가는 소스 IBM DB2 LUW 테이블에 Db2 XML 데이터 형식이 있는지 확인합니다.

소스 IBM Db2 LUW 버전이에서 지원되는지 확인 AWS DMS

API 키: db2-validate-supported-versions-source

이 사전 마이그레이션 평가는 소스 IBM Db2 LUW 버전이에서 지원되는지 확인합니다 AWS DMS.

대상 IBM Db2 LUW 버전이에서 지원되는지 확인 AWS DMS

API 키: db2-validate-supported-versions-target

이 사전 마이그레이션 평가는 대상 IBM Db2 LUW 버전이에서 지원되는지 확인합니다 AWS DMS.

Digits Randomize 변환 규칙 확인

API 키: db2-datamasking-digits-randomize

이 평가는 테이블 매핑에 사용되는 열이 Digits Randomize 변환 규칙과 호환되는지 확인합니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

Digits Mask에 대한 변환 규칙 확인

API 키: db2-datamasking-digits-mask

이 평가는 테이블 매핑에 사용되는 열이 Digits Mask 변환 규칙에서 지원되지 않는지 확인합니다. 또한 Digits Mask 변환을 해당 열에 적용하면 고유성을 보장할 수 없으므로 DMS 작업 실패가 발생할 수 있으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

해싱 마스크에 대한 변환 규칙 확인

API 키: db2-datamasking-hash-mask

이 평가는 테이블 매핑에 사용되는 열 중 하나라도 해싱 마스크 변환 규칙에서 지원되지 않는지 확인합니다. 또한 소스 열의 길이가 64자를 초과하는지 확인합니다. 해시 마스킹을 지원하려면 대상 열 길이가 64자를 초과하는 것이 가장 좋습니다. 또한 Digits Randomize 변환을 적용해도 고유성이 보장되지 않으므로 평가에서는 변환을 위해 선택한 열이 프라이머리 키, 고유 제약 조건 또는 외래 키의 일부인지 확인합니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-datamasking-digits-randomize

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Randomize가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-datamasking-hash-mask

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 해싱 마스크가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

데이터 검증 작업 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인

API 키: all-to-all-validation-with-digit-mask

이 마이그레이션 전 평가는 데이터 검증 설정과 데이터 마스킹 Digits Mask가 동시에 활성화되지 않았는지 확인합니다. 이러한 기능은 호환되지 않기 때문입니다.

배치 적용 호환성을 위해 대상 테이블에 올바른 인덱스 구성(프라이머리 키 또는 고유 인덱스, 둘 다 아님)이 있는지 확인합니다

API 키: db2-check-batch-apply-target-pk-ui-absence

배치 적용을 사용하려면 대상 테이블에 기본 또는 고유 키가 있어야 하지만 둘 다 있어야 하는 것은 아닙니다. 테이블에 프라이머리 키와 고유 키가 모두 포함된 경우 적용 모드가 배치에서 트랜잭션으로 변경됩니다.

BatchApplyEnabled가 true로 설정된 경우에만 '제한된 LOB 모드'가 사용되는지 확인

API 키: db2-check-for-batch-apply-lob-mode

이 마이그레이션 전 평가는 DMS 작업에 LOB 열이 포함되어 있는지 확인합니다. LOB 열이 작업 범위에 포함된 경우 BatchApplyEnabled=true를 사용하려면 '제한된 LOB 모드'를 사용해야 합니다.

전체 로드 중에 대상 데이터베이스에서 보조 인덱스가 비활성화되었는지 확인

API 키: db2-check-secondary-indexes

이 마이그레이션 전 평가는 대상 데이터베이스의 전체 로드 중에 보조 인덱스가 비활성화되었는지 여부를 검증합니다. 전체 로드 중에 보조 인덱스를 비활성화하거나 제거해야 합니다.

선택한 객체가 소스 데이터베이스에 하나 이상 존재하는지 확인합니다.

API 키: all-check-source-selection-rules

이 사전 마이그레이션 평가는 와일드카드 기반 규칙에 대한 패턴 일치를 포함하여 선택 규칙에 지정된 하나 이상의 객체가 소스 데이터베이스에 존재하는지 확인합니다.

보조 제약 조건 및 인덱스(비기본)가 소스 데이터베이스에 있는지 검증합니다.

API 키: all-check-secondary-constraints

이 사전 마이그레이션 평가는 보조 제약 조건 및 인덱스(외래 키, 검사 제약 조건, 클러스터링되지 않은 인덱스)가 소스 데이터베이스에 있는지 확인합니다.

데이터 유형(레거시) 평가 시작 및 보기

Note

이 섹션에서는 레거시 콘텐츠를 설명합니다. 앞의 [마이그레이션 전 평가 실행 지정, 시작, 보기](#)에 설명된 마이그레이션 전 평가 실행을 사용하는 것이 좋습니다.

콘솔에서는 데이터 유형 평가를 사용할 수 없습니다. API 또는 CLI를 통해서만 데이터 유형 평가를 실행할 수 있으며, 작업의 S3 버킷에서 데이터 유형 평가의 결과만 볼 수 있습니다.

마이그레이션 전 평가는 다음 조건에서 자동으로 실행됩니다.

- 작업 시작 중: 작업 생성 중에 평가를 수동으로 실행하지 않은 경우.
- 작업 재개 중: 지난 7일 이내에 완료된 평가가 없는 경우.

데이터 유형 평가는 대상에서 지원하지 않기 때문에 소스 데이터베이스에서 제대로 마이그레이션되지 않을 수 있는 데이터 유형을 식별합니다. 이 평가 중에는 마이그레이션 작업에 대한 소스 데이터베이스 스키마를 AWS DMS 읽고 열 데이터 유형 목록을 생성합니다. 그런 다음이 목록을에서 지원하는 사전 정의된 데이터 유형 목록과 비교합니다 AWS DMS. 마이그레이션 태스크에 지원되지 않는 데이터 형식이 있는 경우 AWS DMS 는 마이그레이션 태스크에 지원되지 않는 데이터 형식이 있는지 확인할 수 있는 보고서를 생성합니다. AWS DMS 는 마이그레이션 태스크에 지원되지 않는 데이터 형식이 없는 경우 보고서를 생성하지 않습니다.

AWS DMS 는 다음 관계형 데이터베이스에 대한 데이터 형식 평가 보고서 생성을 지원합니다.

- Oracle
- SQL Server
- PostgreSQL
- MySQL
- MariaDB
- Amazon Aurora

CLI 및 SDKs를 사용하여 AWS DMS API에 액세스하여 데이터 유형 평가 보고서를 시작하고 볼 수 있습니다.

- CLI는 [start-replication-task-assessment](#) 명령을 사용하여 데이터 형식 평가를 시작하고 [describe-replication-task-assessment-results](#) 명령을 사용하여 JSON 형식의 최신 데이터 형식 평가 보고서를 확인합니다.
- AWS DMS API는 [StartReplicationTaskAssessment](#) 작업을 사용하여 데이터 유형 평가를 시작하고 [DescribeReplicationTaskAssessmentResults](#) 작업을 사용하여 최신 데이터 유형 평가 보고서를 JSON 형식으로 봅니다.

데이터 형식 평가 보고서는 지원되지 않는 데이터 형식과 각 데이터 형식에 대한 열 수를 나열한 요약이 포함된 단일 JSON 파일입니다. 여기에는 지원되지 않는 데이터 형식을 가진 스키마, 테이블 및 열을 포함하여 지원되지 않는 각 데이터 형식에 대한 데이터 구조 목록이 포함됩니다. 보고서를 사용하여 소스 데이터 형식을 수정하고 마이그레이션을 개선하여 성공적으로 수행될 수 있도록 합니다.

지원되지 않는 데이터 형식에는 두 가지 수준이 있습니다. 보고서에 지원되지 않는다고 표시되는 데이터 형식은 마이그레이션할 수 없습니다. 보고서에 부분적으로 지원된다고 표시되는 데이터 형식은 다른 데이터 형식으로 변환되어 예상대로 마이그레이션되지 않을 수 있습니다.

다음 예제에서는 볼 수 있는 샘플 데이터 형식 평가 보고서를 보여 줍니다.

```
{
  "summary":{
    "task-name":"test15",
    "not-supported":{
      "data-type": [
        "sql-variant"
      ],
      "column-count":3
    },
    "partially-supported":{
      "data-type":[
        "float8",
        "jsonb"
      ],
      "column-count":2
    }
  },
  "types":[
    {
      "data-type":"float8",
      "support-level":"partially-supported",
      "schemas":[
        {
          "schema-name":"schema1",
          "tables":[
            {
              "table-name":"table1",
              "columns":[
                "column1",
                "column2"
              ]
            },
            {
              "table-name":"table2",
              "columns":[
                "column3",
                "column4"
              ]
            }
          ]
        }
      ]
    }
  ]
}
```

```
]
}
]
},
{
  "schema-name": "schema2",
  "tables": [
    {
      "table-name": "table3",
      "columns": [
        "column5",
        "column6"
      ]
    },
    {
      "table-name": "table4",
      "columns": [
        "column7",
        "column8"
      ]
    }
  ]
},
{
  "datatype": "int8",
  "support-level": "partially-supported",
  "schemas": [
    {
      "schema-name": "schema1",
      "tables": [
        {
          "table-name": "table1",
          "columns": [
            "column9",
            "column10"
          ]
        },
        {
          "table-name": "table2",
          "columns": [
            "column11",
            "column12"
          ]
        }
      ]
    }
  ]
}
```

```

    ]
  }
  ]
}
]
}
]
}

```

AWS DMS 는 계정에서 생성한 Amazon S3 버킷 AWS DMS 에 최신 및 모든 이전 데이터 유형 평가를 저장합니다. Amazon S3 버킷 이름의 형식은 다음과 같습니다. 여기서 *customerId*는 고객 ID이고 *customerDNS*는 내부 식별자입니다.

dms-*customerId*-*customerDNS*

Note

기본적으로 AWS 계정 각각에서 최대 100개의 Amazon S3 버킷을 만들 수 있습니다. 는 계정에 버킷을 AWS DMS 생성하므로 버킷 제한을 초과하지 않도록 해야 합니다. 그렇지 않으면 데이터 형식 평가가 실패합니다.

지정된 마이그레이션 작업에 대한 모든 데이터 형식 평가 보고서는 작업 식별자로 이름이 지정된 버킷 폴더에 저장됩니다. 각 보고서의 파일 이름은 yyyy-mm-dd-hh-mm 형식으로 된 데이터 형식 평가의 날짜로 표시됩니다. Amazon S3 관리 콘솔에서 이전 데이터 형식 평가 보고서를 조회하고 비교할 수 있습니다.

AWS DMS 는 이러한 보고서에 대해 생성된 S3 버킷에 대한 액세스를 허용하는 AWS Identity and Access Management (IAM) 역할도 생성합니다. 역할 이름은 dms-access-for-tasks입니다. 이 역할에는 AmazonDMSRedshiftS3Role 정책이 사용됩니다. StartReplicationTaskAssessment를 실행할 때 ResourceNotFoundFault 오류가 발생하는 경우 dms-access-for-tasks 역할 수동 생성에 대한 자세한 내용은 문제 해결 섹션의 [ResourceNotFoundFault](#) 섹션을 참조하세요.

평가 실행의 문제 해결

그 다음 AWS Database Migration Service를 사용하여 평가 보고서를 실행하는 데 발생하는 문제 해결에 대한 주제를 찾을 수 있습니다. 이러한 주제는 일반적인 문제를 해결하는 데 도움이 될 수 있습니다.

주제

- [StartReplicationTaskAssessment 실행 시 ResourceNotFoundFault](#)

StartReplicationTaskAssessment 실행 시 ResourceNotFoundFault

[StartReplicationTaskAssessment](#) 작업을 실행할 때 다음과 같은 예외가 발생할 수 있습니다.

An error occurred (ResourceNotFoundFault) when calling the StartReplicationTaskAssessment operation: Task assessment has not been run or dms-access-for-tasks IAM Role not configured correctly

이 예외가 발생하면 다음을 수행하여 dms-access-for-tasks 역할을 생성합니다.

1. IAM 콘솔(<https://console.aws.amazon.com/iam/>)을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지에서 신뢰할 수 있는 엔터티 유형으로 사용자 지정 신뢰 정책을 선택합니다.
5. 편집기에서 다음 JSON을 붙여 넣으면서 기존 텍스트를 바꿉니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

위의 정책은 sts:AssumeRole 권한을 부여합니다 AWS DMS. AmazonDMSRedshiftS3Role 정책을 추가하면 DMS가 계정에 S3 버킷을 생성하고 데이터 유형 평가 결과를 이 S3 버킷에 넣을 수 있습니다.

6. 다음을 선택합니다.
7. 권한 추가 페이지에서 AmazonDMSRedshiftS3Role 정책을 검색하고 추가합니다. 다음을 선택합니다.
8. 이름, 검토 및 생성 페이지에서 dms-access-for-tasks 역할의 이름을 지정합니다. 역할 생성을 선택합니다.

작업 설정에 대한 보충 데이터 지정

일부 AWS DMS 엔드포인트에 대한 복제 작업을 생성하거나 수정할 때 마이그레이션을 수행하기 위해 작업에 추가 정보가 필요할 수 있습니다. DMS 콘솔의 옵션을 사용하여 이 추가 정보를 지정할 수 있습니다. 또는 DMS API 작업 CreateReplicationTask 또는 ModifyReplicationTask에 대한 TaskData 파라미터를 사용하여 지정할 수 있습니다.

대상 엔드포인트가 Amazon Neptune인 경우, 테이블 매핑을 보완하는 매핑 데이터를 지정해야 합니다. 이 보충 매핑 데이터는 소스 관계형 데이터를 Neptune 데이터베이스가 사용할 수 있는 대상 그래프 데이터로 변환하는 방법을 지정합니다. 이 경우 두 가지 형식 중 하나를 사용할 수 있습니다. 자세한 내용은 [Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정](#) 단원을 참조하십시오.

AWS DMS 작업 모니터링

모니터링은 AWS DMS 및 AWS 솔루션의 안정성, 가용성 및 성능을 유지하는 데 중요한 부분입니다. 다중 지점 장애가 발생할 경우 보다 쉽게 디버깅할 수 있도록 AWS 솔루션의 모든 부분으로부터 모니터링 데이터를 수집해야 합니다. AWS DMS 작업 및 리소스를 모니터링하고 잠재적 인시던트에 대응하기 위한 몇 가지 도구를 AWS 제공합니다.

AWS DMS 이벤트 및 알림

AWS DMS 는 Amazon Simple Notification Service(Amazon SNS)를 사용하여 복제 인스턴스의 생성 또는 삭제와 같은 AWS DMS 이벤트가 발생할 때 알림을 제공합니다. 이벤트를 구독할 수 있는 범주로 AWS DMS 그룹화하므로 해당 범주의 이벤트가 발생할 때 알림을 받을 수 있습니다. 예를 들어 임의의 복제 인스턴스에 대한 생성 카테고리를 구독할 경우 생성 관련 이벤트가 발생하여 복제 인스턴스에 영향을 끼칠 때마다 알림 메시지가 수신됩니다. 이메일 메시지, 문자 메시지 또는 HTTP 엔드포인트 호출과 같이 AWS 리전에 대해 Amazon SNS에서 지원하는 모든 형식으로 이러한 알림을 사용할 수 있습니다. 자세한 내용은 [에서 Amazon SNS 이벤트 및 알림 작업 AWS Database Migration Service](#) 섹션을 참조하세요.

작업 상태

작업 상태를 확인하고 작업의 제어 테이블을 확인하여 작업 진행률을 모니터링할 수 있습니다. 작업 상태는 AWS DMS 작업 및 관련 리소스의 상태를 나타냅니다. 여기에는 작업이 생성, 시작, 실행 또는 중지되는 경우와 같은 표시가 포함됩니다. 또한 테이블 전체 로드 시작되었거나 진행 중인 경우와 같이 작업이 마이그레이션하는 테이블의 현재 상태와 테이블에서 발생한 삽입, 삭제 및 업데이트 횟수와 같은 세부 정보가 포함됩니다. 작업 및 작업 리소스 상태 모니터링에 대한 자세한 내용은 [작업 상태](#) 및 [작업 중 테이블 상태](#) 섹션을 참조하세요. 제어 테이블에 대한 자세한 내용은 [제어 테이블 작업 설정](#) 섹션을 참조하세요.

Amazon CloudWatch 경보 및 로그

Amazon CloudWatch 경보를 사용하여 지정한 기간 동안 하나 이상의 작업 지표를 감시합니다. 지표가 지정된 임계값을 초과하면 Amazon SNS 주제로 알림이 전송됩니다. CloudWatch 경보는 단순히 특정 상태에 있다고 해서 작업을 호출하지 않습니다. 대신 상태가 변경되어 지정한 기간 동안 유지되어야 합니다. AWS DMS 또한 CloudWatch를 사용하여 마이그레이션 프로세스 중에 작업 정보를 로깅합니다. 또는 AWS DMS API를 사용하여 작업 로그에 AWS CLI 대한 정보를 볼 수 있습니다. 에서 CloudWatch를 사용하는 방법에 대한 자세한 내용은 [AWS DMS Amazon CloudWatch를 사용한 복제 태스크 모니터링](#) 섹션을 참조하세요. AWS DMS 지표 모니터링에 대한 자세한 내용은 [AWS Database Migration Service 지표](#) 섹션을 참조하세요. AWS DMS 작업 로그 사용에 대한 자세한 내용은 [AWS DMS 작업 로그 보기 및 관리](#) 섹션을 참조하세요.

Time Travel 로그

복제 작업을 로깅하고 디버깅하려면 AWS DMS Time Travel을 사용할 수 있습니다. 이 방식에서는 Amazon S3를 사용하여 로그를 저장하고 암호화 키를 사용하여 로그를 암호화합니다. 날짜-시간 필터를 사용하여 S3 로그를 검색한 다음, 필요에 따라 로그를 보고 다운로드하고 난독화할 수 있습니다. 이렇게 하면 ‘과거로 이동’하여 데이터베이스 활동을 조사할 수 있습니다.

Time Travel은 DMS가 지원하는 PostgreSQL 소스 엔드포인트와 DMS가 지원하는 PostgreSQL 및 MySQL 대상 엔드포인트와 함께 사용할 수 있습니다. Time Travel은 전체 로드 및 CDC 태스크 및 CDC 전용 태스크에만 사용할 수 있습니다. Time Travel을 활성화하거나 기존의 Time Travel 설정을 수정하려면 태스크를 중지해야 합니다.

Time Travel 로그에 대한 자세한 내용은 [Time Travel 작업 설정](#) 섹션을 참조하세요. Time Travel 로그 사용에 대한 모범 사례는 [Time Travel을 통한 복제 태스크 문제 해결](#) 섹션을 참조하세요.

AWS CloudTrail 로그

AWS DMS 는 사용자 AWS CloudTrail, IAM 역할 또는 서비스가 수행한 작업에 대한 레코드를 제공하는 AWS 서비스와 통합됩니다 AWS DMS. CloudTrail은 콘솔의 호출 및 API 작업에 대한 코드 호출을 AWS DMS 포함하여에 대한 모든 AWS DMS API 호출을 이벤트 AWS DMS 로 캡처합니다. 추적을 생성하면 이벤트를 포함하여 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다 AWS DMS. 추적을 구성하지 않은 경우에도 이벤트 기록에서 CloudTrail 콘솔의 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 수행된 요청, 요청이 수행된 AWS DMS IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다. 자세한 내용은 [를 사용하여 AWS DMS API 호출 로깅 AWS CloudTrail](#) 단원을 참조하십시오.

데이터베이스 로그

또는 데이터베이스 서비스의 API를 사용하여 작업 엔드포인트에 대한 AWS 데이터베이스 로그를 보고, 다운로드하고 AWS Management Console AWS CLI, 볼 수 있습니다. 자세한 내용은 [AWS 설명서](#)에서 해당 데이터베이스 서비스 설명서를 참조하세요.

자세한 내용은 다음 항목을 참조하세요.

주제

- [작업 상태](#)
- [작업 중 테이블 상태](#)
- [Amazon CloudWatch를 사용한 복제 태스크 모니터링](#)
- [AWS Database Migration Service 지표](#)

- [AWS DMS 작업 로그 보기 및 관리](#)
- [를 사용하여 AWS DMS API 호출 로깅 AWS CloudTrail](#)
- [AWS DMS 컨텍스트 로깅](#)
- [향상된 모니터링 대시보드](#)
- [AWS DMS 이벤트 보기](#)

작업 상태

태스크 상태는 말 그대로 태스크의 상태를 나타냅니다. 다음 테이블은 작업이 가질 수 있는 가능한 상태를 나타냅니다.

작업 상태	설명
생성 중	AWS DMS 가 작업을 생성하고 있습니다.
실행 중	작업이 지정된 마이그레이션 업무를 수행하고 있습니다.
중지됨	작업이 중지됩니다.
중지 중	작업이 중지되고 있습니다. 대개 작업에서 사용자 개입을 나타냅니다.
삭제 중	보통 사용자 개입 요청을 통해 작업이 삭제되고 있습니다.
실패	작업에 실패했습니다. 자세한 내용은 태스크 로그 파일을 참조하세요.
오류	오류로 인해 태스크가 중지되었습니다. 태스크 오류에 대한 간략한 설명은 개요 탭의 마지막 실패 메시지 섹션에 나와 있습니다.
실행 중(오류 포함)	태스크가 오류 상태로 실행 중입니다. 이는 일반적으로 태스크에 있는 하나 이상의 테이블을 마이그레이션할 수 없음을 나타냅니다. 태스크는 선택 규칙에 따라 다른 테이블을 계속 로드합니다.
시작됨	작업이 복제 인스턴스와 원본 및 대상 엔드포인트에 연결되고 있습니다. 모든 필터와 변환이 적용되고 있습니다.

작업 상태	설명
준비됨	작업을 실행할 준비가 되어 있습니다. 이 상태 다음에는 대개 "생성" 상태가 나옵니다.
수정 중	작업이 수정되고 있습니다. 대개 작업 설정을 수정한 사용자 작업으로 인한 것입니다.
이동 중	태스크가 다른 복제 인스턴스로 이동하는 중입니다. 이동이 완료될 때까지 복제는 이 상태로 유지됩니다. 복제 태스크가 이동 중인 동안에는 태스크 삭제 작업만 허용됩니다.
이동 실패	어떤 이유로든 태스크 이동이 실패했습니다(예: 대상 복제 인스턴스에 스토리지 공간이 부족). 복제 태스크가 이 상태 일 때는 태스크를 시작, 수정, 이동 또는 삭제할 수 있습니다.
테스트	이 태스크에 지정된 데이터베이스 마이그레이션이 StartReplicationTaskAssessmentRun 또는 StartReplicationTaskAssessment 작업 실행에 대한 응답으로 테스트되고 있습니다.

작업 상태 표시줄은 작업 진행률의 추정치를 나타냅니다. 이 추정치의 품질은 소스 데이터베이스의 테이블 통계 품질에 따라 달라지며, 테이블 통계 품질이 좋을수록 추정 정확도가 높아집니다. 추정 행 통계 없이 단 하나의 테이블을 사용하는 작업의 경우, 모든 유형의 완료율 추정치를 제공할 수 없습니다. 이 경우에 작업 상태와 로드되는 행 표시를 사용하여 작업이 실제로 실행되고 진행되고 있음을 확인할 수 있습니다.

DMS 콘솔은 '마지막 업데이트' 열에 AWS DMS 가 테이블의 테이블 통계 레코드를 마지막으로 업데이트한 시간만 표시합니다. 테이블에 대한 마지막 업데이트 시간을 표시하지 않습니다.

DMS 콘솔을 사용하는 것 외에도 다음 예와 같이 [AWS CLI](#)의 `aws dms describe-replication-tasks` 명령을 사용하여 태스크 상태를 비롯한 현재 복제 태스크에 대한 설명을 출력할 수 있습니다.

```
{
  "ReplicationTasks": [
    {
      "ReplicationTaskIdentifier": "moveit2",
```

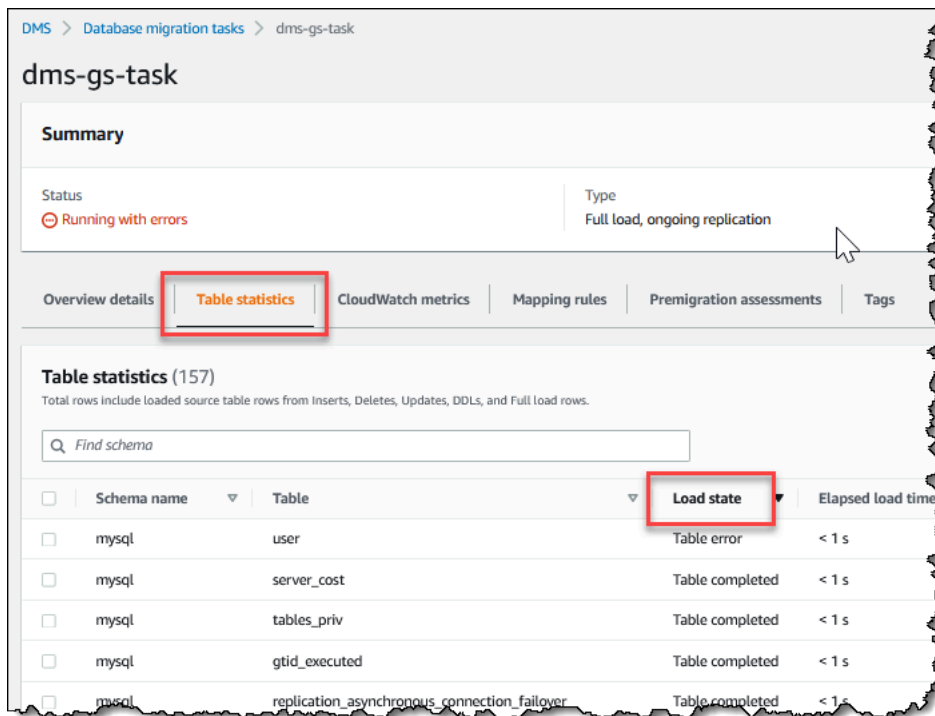
```

        "SourceEndpointArn": "arn:aws:dms:us-east-1:123456789012:endpoint:6GGI6YPWYGAYUVLKIB732KEVWA",
        "TargetEndpointArn": "arn:aws:dms:us-east-1:123456789012:endpoint:E0M4SFKCZEYHZBFGAGZT3QEC5U",
        "ReplicationInstanceArn": "arn:aws:dms:us-east-1:123456789012:rep:T30M70UB5NM2LCVZF7JPGJRNUE",
        "MigrationType": "full-load",
        "TableMappings": ...output omitted... ,
        "ReplicationTaskSettings": ...output omitted... ,
        "Status": "stopped",
        "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
        "ReplicationTaskCreationDate": 1590524772.505,
        "ReplicationTaskStartDate": 1590619805.212,
        "ReplicationTaskArn": "arn:aws:dms:us-east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII",
        "ReplicationTaskStats": {
            "FullLoadProgressPercent": 100,
            "ElapsedTimeMillis": 0,
            "TablesLoaded": 0,
            "TablesLoading": 0,
            "TablesQueued": 0,
            "TablesErrored": 0,
            "FreshStartDate": 1590619811.528,
            "StartDate": 1590619811.528,
            "StopDate": 1590619842.068
        }
    }
}

```

작업 중 테이블 상태

AWS DMS 콘솔은 마이그레이션 중에 테이블의 상태에 대한 정보를 업데이트합니다. 다음 표에는 가능한 상태 값이 나와 있습니다.



State	설명
테이블 없음	AWS DMS가 소스 엔드포인트에서 테이블을 찾을 수 없습니다.
로드 전	전체 로드 프로세스가 활성화되었지만, 아직 시작되지 않았습니다.
전체 로드	전체 로드 프로세스가 진행되고 있습니다.
테이블 완료	전체 로드가 완료되었습니다.
테이블 취소	테이블 로드가 취소되었습니다.
테이블 오류	테이블을 로드하는 동안 오류가 발생했습니다.

Amazon CloudWatch를 사용한 복제 태스크 모니터링

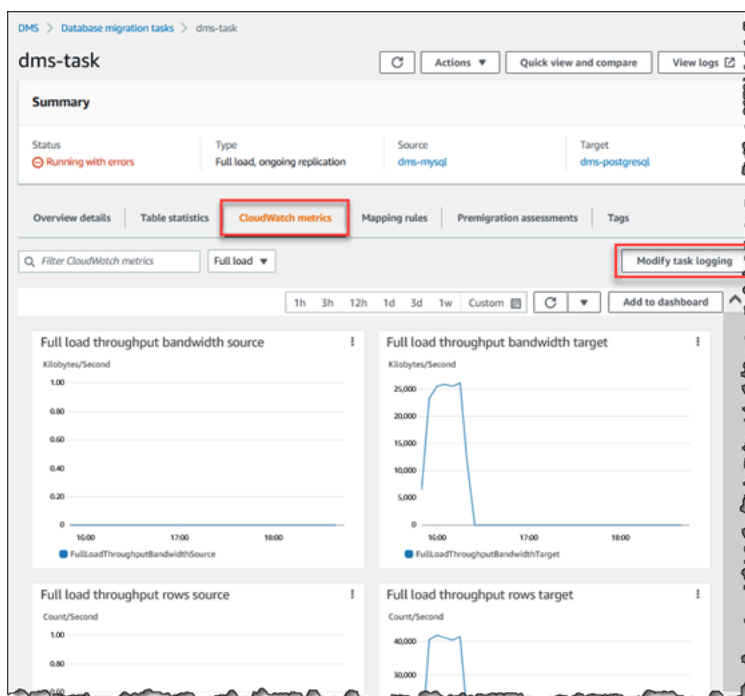
Amazon CloudWatch 경보 또는 이벤트를 사용하여 마이그레이션을 더욱 면밀하게 추적할 수 있습니다. Amazon CloudWatch에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 [Amazon](#)

[CloudWatch, Amazon CloudWatch Events 및 Amazon CloudWatch Logs란 무엇입니까?](#)를 참조하세요. Amazon CloudWatch 사용 시 비용이 청구됩니다.

복제 태스크에서 CloudWatch 로그가 생성되지 않는 경우 문제 해결 안내서에서 [AWS DMS 는 CloudWatch 로그를 생성하지 않습니다](#). 섹션을 참조하세요.

AWS DMS 콘솔에는 다음과 같이 작업 상태, 완료율, 경과 시간 및 테이블 통계를 포함하여 각 작업에 대한 기본 CloudWatch 통계가 표시됩니다. 복제 태스크를 선택한 다음 CloudWatch 지표 탭을 선택합니다.

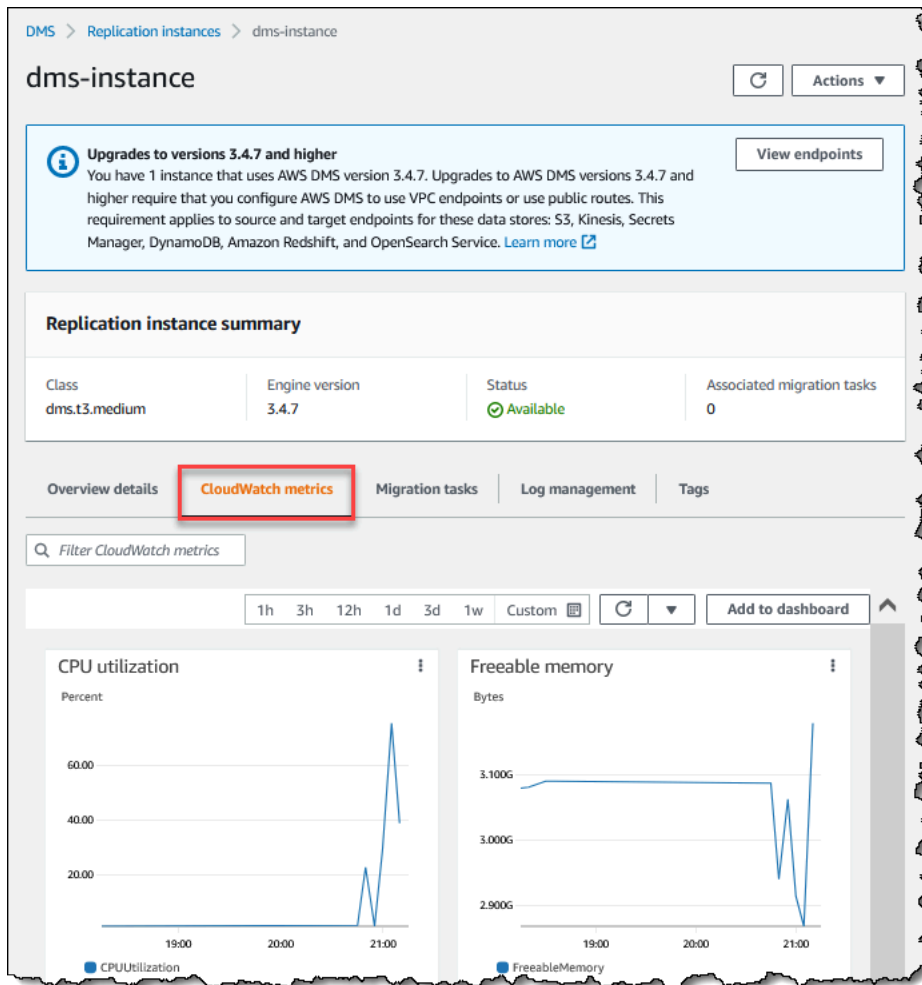
CloudWatch 태스크 로그 설정을 보고 수정하려면 태스크 로깅 수정을 선택합니다. 자세한 내용은 [작업 설정 로깅](#) 단원을 참조하십시오.



AWS DMS 콘솔에는 테이블 통계 탭을 선택할 때 삽입, 삭제 및 업데이트 수를 포함하여 각 테이블에 대한 성능 통계가 표시됩니다.

Schema name	Table	Load state	Elapsed load time	Inserts	Deletes	Updates	DOLs	Applied inserts
mysql	user	Table error	1 s	0	0	0	0	0
mysql	server_cost	Table completed	1 s	0	0	0	0	0
dms_sample	seat_type	Table completed	< 1 s	0	0	0	0	0
mysql	tables_priv	Table completed	1 s	0	0	0	0	0

또한, 복제 인스턴스 페이지에서 복제 인스턴스를 선택하는 경우 CloudWatch 지표 탭을 선택하여 인스턴스의 성능 지표를 확인할 수 있습니다.



AWS Database Migration Service 지표

AWS DMS는 다음에 대한 통계를 제공합니다.

- 호스트 지표 - Amazon CloudWatch에서 제공하는 복제 호스트에 대한 성능 및 사용률 통계. 사용 가능한 전체 측정치 목록은 [복제 인스턴스 지표](#) 섹션을 참조하세요.
- 복제 태스크 지표 - 들어오는 변경 사항과 커밋된 변경 사항, 복제 호스트와 소스 및 대상 데이터베이스 사이의 지연 시간을 포함한 복제 태스크 통계. 사용 가능한 전체 측정치 목록은 [복제 작업 지표](#) 섹션을 참조하세요.
- 테이블 지표 - 완료된 삽입, 업데이트, 삭제 및 DDL 문 개수를 포함하여 마이그레이션 중인 테이블에 대한 통계.

작업 지표는 복제 호스트와 원본 엔드포인트 간의 통계, 그리고 복제 호스트와 대상 엔드포인트 간의 통계로 구분됩니다. CDCLatencySource와 CDCLatencyTarget을 사용하여 이러한 관련 통계를 비교하여 작업의 지연 시간 원인을 식별할 수 있습니다. 예를 들어 CDCLatencySource 값이 CDCLatencyTarget 값과 거의 동일한 경우 먼저 소스 측을 확인해야 합니다. 그러나 CDCLatencyTarget이 CDCLatencySource보다 높은 경우 대상 측 지연 시간을 확인하는 데 집중해야 합니다.

작업 지표 값은 원본 데이터베이스에서 현재 작업의 영향을 받을 수 있습니다. 예를 들어, 트랜잭션이 시작되었지만 커밋되지 않은 경우, CDCLatencySource 지표는 트랜잭션이 커밋될 때까지 계속 증가합니다.

복제 인스턴스의 경우, FreeableMemory 측정치에 대한 명확한 이해가 필요합니다. 여유 메모리는 사용 가능한 실제 여유 메모리를 나타내지 않습니다. 해제하여 다른 용도로 사용할 수 있는 현재 사용 중인 메모리로, 여유 복제 인스턴스에 사용 중인 버퍼 및 캐시의 조합입니다.

FreeableMemory 측정치에는 사용 가능한 실제 여유 메모리가 반영되지 않지만, FreeableMemory 및 SwapUsage 측정치를 보고 복제 인스턴스가 오버로드되었는지 여부를 알 수 있습니다.

다음 조건에 대해 이 두 측정치를 모니터링합니다.

- FreeableMemory 측정치가 0에 가까움.
- SwapUsage 측정치가 증가하거나 변동이 있음.

이러한 두 가지 상태 중 하나에 해당하는 경우 이는 대규모 복제 인스턴스로 전환하는 것을 고려해야 함을 나타냅니다. 또한 복제 인스턴스에서 실행 중인 작업 수와 유형을 줄이기는 것도 고려해 보아야 합니다. 전체 로드 작업에는 변경 내용만을 복제하는 작업보다 더 많은 메모리가 요구됩니다.

AWS DMS 마이그레이션 작업의 실제 메모리 요구 사항을 대략적으로 추정하려면 다음 파라미터를 사용할 수 있습니다.

LOB 열

마이그레이션 범위 내 각 테이블의 평균 LOB 열 수입니다.

병렬로 로드할 수 있는 최대 테이블 수

한 작업에서 병렬로 AWS DMS 로드되는 최대 테이블 수입니다.

기본값은 8입니다.

LOB 청크 크기

가 대상 데이터베이스에 데이터를 복제하는 데 AWS DMS 사용하는 킬로바이트 단위의 LOB 청크 크기입니다.

전체 로드 중 커밋 비율

가 병렬로 전송할 AWS DMS 수 있는 최대 레코드 수입니다.

기본값은 10,000입니다.

LOB 크기

개별 LOB의 최대 크기(KB)입니다.

대량 배열 크기

엔드포인트 드라이버에서 가져오거나 처리하는 최대 행 수입니다. 이 값은 드라이버 설정에 따라 달라집니다.

기본값은 1,000입니다.

이러한 값을 결정한 후 다음 방법 중 하나를 사용하여 마이그레이션 태스크에 필요한 메모리 양을 추정할 수 있습니다. 이러한 방법은 마이그레이션 태스크의 LOB 열 설정에서 선택한 옵션에 따라 달라집니다.

- 전체 LOB 모드의 경우 다음 공식을 사용합니다.

$$\text{Required memory} = (\text{LOB columns}) * (\text{Maximum number of tables to load in parallel}) * (\text{LOB chunk size}) * (\text{Commit rate during full load})$$

소스 테이블에 평균 2개의 LOB 열이 포함되어 있고 LOB 청크 크기가 64KB인 경우를 예로 들어 보겠습니다. Maximum number of tables to load in parallel 및 Commit rate during full load에 기본값을 사용하는 경우 태스크에 필요한 메모리 양은 다음과 같습니다.

$$\text{Required memory} = 2 * 8 * 64 * 10,000 = 10,240,000 \text{ KB}$$

Note

전체 로드 중에 커밋 속도 값을 줄이려면 AWS DMS 콘솔을 열고 데이터베이스 마이그레이션 작업을 선택한 다음 작업을 생성하거나 수정합니다. 고급 설정을 확장하고 전체 로드 중 커밋 비율 값을 입력합니다.

- 제한적 LOB 모드의 경우 다음 공식을 사용합니다.

$$\text{Required memory} = (\text{LOB columns}) * (\text{Maximum number of tables to load in parallel}) * (\text{LOB size}) * (\text{Bulk array size})$$

소스 테이블에 평균 2개의 LOB 열이 포함되어 있고 개별 LOB의 최대 크기가 4,096KB인 경우를 예로 들어 보겠습니다. Maximum number of tables to load in parallel 및 Bulk array size에 기본값을 사용하는 경우 태스크에 필요한 메모리 양은 다음과 같습니다.

$$\text{Required memory} = 2 * 8 * 4,096 * 1,000 = 65,536,000 \text{ KB}$$

가 변환 AWS DMS 을 최적으로 수행하려면 변환이 발생할 때 CPU를 사용할 수 있어야 합니다. CPU를 오버로드하고 CPU 리소스가 충분하지 않으면 마이그레이션 속도가 느려질 수 있습니다. 특히 Oracle에서 PostgreSQL로 마이그레이션하는 것과 같은 이기종 마이그레이션 및 복제를 수행할 때 CPU 집약적일 수 있습니다.

복제 인스턴스 지표

복제 인스턴스 모니터링에는 다음 통계의 Amazon CloudWatch 지표가 포함됩니다.

지표	설명
AvailableMemory	스와핑 없이 새 애플리케이션을 시작하는 데 사용할 수 있는 메모리의 크기. 자세한 내용은 Linux man-pages 페이지의 /proc/memInfo 섹션에서 MemAvailable 값을 참조하세요. 단위: 바이트
CPUAllocated	태스크에 최대 할당된 CPU의 비율(0은 제한이 없음을 의미함). AWS DMS 는 CloudWatch 콘솔 ReplicationTaskIdentifier 에서 ReplicationInstanceIdentifier 및의 결합된 차원에 대해 이 지표를 높입니다. ReplicationInstanceIdentifier, ReplicationTaskIdentifier 범주를 사용하여 이 지표를 볼 수 있습니다. 단위: 백분율
CPUUtilization	인스턴스에서 현재 사용 중인 할당된 vCPU(가상 CPU)의 비율입니다.

지표	설명
	단위: 백분율
DiskQueueDepth	디스크 액세스를 대기 중인 읽기/쓰기 요청(I/O) 수입니다. 단위: 개
FreeStorageSpace	사용 가능한 스토리지 공간 크기입니다. 단위: 바이트
FreeMemory	애플리케이션, 페이지 캐시, 커널 자체 데이터 구조에서 사용할 수 있는 물리적 메모리의 양입니다. 자세한 내용은 Linux man-pages 페이지의 /proc/memInfo 섹션에서 MemFree 값을 참조하세요. 단위: 바이트
FreeableMemory	사용 가능한 RAM 크기입니다. 단위: 바이트
MemoryAllocated	태스크에 대한 최대 메모리 할당량입니다(0은 제한이 없음을 의미함). AWS DMS 는 CloudWatch 콘솔ReplicationTaskIdentifier 에서 ReplicationInstanceIdentifier 및의 결합된 차원에 대해 이 지표를 높입니다. ReplicationInstanceIdentifier, ReplicationTaskIdentifier 범주를 사용하여 이 지표를 볼 수 있습니다. 단위: MiB
WriteIOPS	초당 평균 디스크 쓰기 I/O 연산 수. 단위: 개수/초
ReadIOPS	초당 평균 디스크 읽기 I/O 연산 수 단위: 개수/초

지표	설명
WriteThroughput	초당 디스크에 쓴 평균 바이트 수. 단위: 바이트/초
ReadThroughput	초당 디스크에서 읽은 평균 바이트 수입니다. 단위: 바이트/초
WriteLatency	디스크 I/O (출력) 연산당 평균 처리 시간. 단위: 밀리초
ReadLatency	디스크 I/O (입력) 연산당 평균 처리 시간. 단위: 밀리초
SwapUsage	복제 인스턴스에서 사용된 스왑 공간 크기. 단위: 바이트
NetworkTransmitThroughput	복제 인스턴스에서 발생하는 송신(Transmit) 네트워크 트래픽으로, 고객 데이터베이스 트래픽을 포함합니다. 단위: 바이트/초
NetworkReceiveThroughput	복제 인스턴스에서 발생하는 수신(Receive) 네트워크 트래픽으로, 고객 데이터베이스 트래픽을 포함합니다. 단위: 바이트/초

복제 작업 지표

복제 태스크 모니터링에는 다음 통계의 지표가 포함됩니다.

지표	설명
FullLoadThroughput BandwidthTarget	대상 전체 로드의 발신 데이터(초당 KB 단위).

지표	설명
FullLoadThroughput RowsTarget	대상 전체 로드의 발신 변경 사항(초당 행 수 단위).
CDCIncomingChanges	대상에 적용되기를 기다리고 있는 특정 시점의 총 변경 이벤트 수입니다. 이 값은 소스 엔드포인트의 트랜잭션 변경 비율 측정치와 같지 않습니다. 이 측정치의 값이 크면 일반적으로 AWS DMS 가 캡처한 변경 사항을 적시에 적용할 수 없어서 대상 지연 시간이 길어짐을 의미합니다.
CDCChange sMemorySource	메모리에서 누적되면서 원본에서 커밋되기를 대기하는 행 개수. 이 지표를 CDCChangesDiskSource와 함께 확인할 수 있습니다.
CDCChange sMemoryTarget	메모리에서 누적되면서 대상으로 커밋되기를 대기하는 행 개수. 이 지표를 CDCChangesDiskTarget과 함께 확인할 수 있습니다.
CDCChangesDiskSource	디스크에서 누적되면서 원본에서 커밋되기를 대기하는 행 개수. 이 지표를 CDCChangesMemorySource와 함께 확인할 수 있습니다.
CDCChangesDiskTarget	디스크에서 누적되면서 대상으로 커밋되기를 대기하는 행 개수. 이 지표를 CDCChangesMemoryTarget과 함께 확인할 수 있습니다.
CDCThroughputBandwidthTarget	대상의 발신 데이터(초당 KB 단위). CDCThroughputBandwidth는 샘플링 포인트에서 전송된 송신 데이터를 기록합니다. 어떠한 태스크 네트워크 트래픽도 찾을 수 없다면 이 값은 0입니다. CDC는 장시간 트랜잭션을 할 수 없기 때문에 네트워크 트래픽은 기록되지 않을 수 있습니다.
CDCThroughputRowsSource	원본의 수신 작업 수신 변경 사항 개수(초당 행 수 단위).
CDCThroughputRowsTarget	대상의 발신 작업 변경 사항 개수(초당 행 수 단위).

지표	설명
CDCLatencySource	소스 엔드포인트에서 캡처한 마지막 이벤트와 AWS DMS 인스턴스의 현재 시스템 타임스탬프 간의 간격(초)입니다. CDCLatencySource는 소스 인스턴스와 복제 인스턴스 간의 지연 시간을 나타냅니다. CDCLatencySource가 높으면 소스에서 변경 사항을 캡처하는 프로세스가 지연되었음을 나타냅니다. 지속적 복제의 지연 시간을 식별하기 위해 이 지표를 CDCLatencyTarget과 함께 확인할 수 있습니다. CDCLatencySource와 CDCLatencyTarget이 모두 높으면 CDCLatencySource를 먼저 조사하세요. 소스 인스턴스와 복제 인스턴스 간에 복제 지연이 없는 경우 CDCSourceLatency는 0이 될 수 있습니다. 복제 태스크에서 소스의 트랜잭션 로그에서 다음 이벤트를 읽으려고 시도하고 소스에서 마지막으로 읽은 때와 비교하여 새 이벤트가 없는 경우에도 CDCSourceLatency는 0이 될 수 있습니다. 이 경우 태스크는 CDCSourceLatency를 0으로 재설정합니다.
CDCLatencyTarget	대상에 커밋되기를 대기하는 첫 이벤트 타임스탬프와 AWS DMS 인스턴스의 현재 타임스탬프 사이의 간격(초)입니다. 대상 지연 시간은 복제 인스턴스 서버 시간과 대상 구성 요소에 전달된 확인되지 않은 가장 오래된 이벤트 ID 간의 차이입니다. 즉, 대상 지연 시간은 복제 인스턴스와 적용되었지만 TRG 엔드포인트에서 확인되지 않은 가장 오래된 이벤트 간의 타임스탬프 차이입니다(99%). CDCLatencyTarget이 높으면 대상에 변경 이벤트를 적용하는 프로세스가 지연되었음을 나타냅니다. 지속적 복제의 지연 시간을 식별하기 위해 이 지표를 CDCLatencySource와 함께 확인할 수 있습니다. CDCLatencyTarget은 높지만 CDCLatencySource는 높지 않은 경우 다음을 조사하세요. • 대상에 프라이머리 키 또는 인덱스가 없음 • 대상 또는 복제 인스턴스에서 리소스 병목 현상이 발생함 • 복제 인스턴스와 대상 사이에 네트워크 문제가 있음

지표	설명
CPUUtilization	태스크가 여러 코어에 걸쳐 사용하는 CPU의 비율입니다. 태스크 CPUUtilization의 의미는 복제 CPUUtilization과 약간 다릅니다. 하나의 vCPU가 완전히 사용되는 경우 100%를 나타내지만 여러 vCPU가 사용되는 경우에는 값이 100%를 초과할 수 있습니다. 단위: 백분율
SwapUsage	태스크가 사용하는 스왑의 양. 단위: 바이트
MemoryUsage	태스크에서 소비한 제어 그룹(cgroup) memory.usage_in_bytes. DMS는 cgroup을 사용하여 메모리 및 CPU와 같은 시스템 리소스의 사용을 제어합니다. 이 지표는 해당 태스크에 할당된 cgroup 내에서 태스크의 메모리 사용량을 메가바이트 단위로 나타냅니다. cgroup 한도는 DMS 복제 인스턴스 클래스에 사용할 수 있는 리소스를 기반으로 합니다. memory.usage_in_bytes는 메모리의 RSS(상주 세트 크기), 캐시 및 스왑 구성 요소로 구성됩니다. 필요한 경우 운영 체제가 캐시 메모리를 회수할 수 있습니다. 복제 인스턴스 지표인 AvailableMemory도 모니터링하는 것이 좋습니다. AWS DMS는 CloudWatch 콘솔ReplicationTaskIdentifier에서 ReplicationInstanceIdentifier 및의 결합된 차원에 대해 이 지표를 높입니다. ReplicationInstanceIdentifier, ReplicationTaskIdentifier 범주를 사용하여 이 지표를 볼 수 있습니다.

AWS DMS 작업 로그 보기 및 관리

Amazon CloudWatch를 사용하여 AWS DMS 마이그레이션 프로세스 중에 작업 정보를 로깅할 수 있습니다. 작업 설정을 선택할 때 로깅을 활성화합니다. 자세한 내용은 [작업 설정 로깅](#) 섹션을 참조하세요.

실행된 작업의 로그를 보려면 다음 단계를 따르세요.

1. AWS DMS 콘솔을 열고 탐색 창에서 데이터베이스 마이그레이션 작업을 선택합니다. 데이터베이스 마이그레이션 태스크 대화 상자가 나타납니다.

2. 작업 이름을 선택합니다. 개요 세부 정보 대화 상자가 나타납니다.
3. 마이그레이션 태스크 로그 섹션을 찾아 CloudWatch Logs 보기를 선택합니다.

또한 AWS CLI 또는 AWS DMS API를 사용하여 작업 로그에 대한 정보를 볼 수 있습니다. 이렇게 하려면 `describe-replication-instance-task-logs` AWS CLI 명령 또는 AWS DMS API 작업을 사용합니다. `DescribeReplicationInstanceTaskLogs`.

예를 들어 다음 AWS CLI 명령은 작업 로그 메타데이터를 JSON 형식으로 보여줍니다.

```
$ aws dms describe-replication-instance-task-logs \
  --replication-instance-arn arn:aws:dms:us-east-1:237565436:rep:CDSFSFSFFFSSUFCAY
```

명령의 샘플 응답은 다음과 같습니다.

```
{
  "ReplicationInstanceTaskLogs": [
    {
      "ReplicationTaskArn": "arn:aws:dms:us-east-1:237565436:task:MY34U6Z4MSY52GRTIX304AY",
      "ReplicationTaskName": "mysql-to-ddb",
      "ReplicationInstanceTaskLogSize": 3726134
    }
  ],
  "ReplicationInstanceArn": "arn:aws:dms:us-east-1:237565436:rep:CDSFSFSFFFSSUFCAY"
}
```

이 응답에 복제 인스턴스와 관련된 한 가지 작업 로그(mysql-to-ddb)가 있습니다. 이 로그의 크기는 3,726,124바이트입니다.

`describe-replication-instance-task-logs`에서 반환한 정보를 사용하여 작업 로그 문제를 진단하고 해결할 수 있습니다. 예를 들어, 태스크의 상세 디버그 로깅을 활성화하면 태스크 로그가 빠르게 증가하게 되어 복제 인스턴스에서 사용 가능한 스토리지를 모두 소비하고 인스턴스 상태가 `storage-full`로 변경될 수 있습니다. 작업 로그를 기술하면 더 이상 필요 없는 항목을 파악해 삭제하여 스토리지 공간을 확보할 수 있습니다.

작업의 작업 로그를 삭제하려면 작업 설정 DeleteTaskLogs를 true로 설정하세요. 예를 들어 다음 JSON은 명령 또는 AWS DMS API ModifyReplicationTask 작업을 사용하여 AWS CLI modify-replication-task 작업을 수정할 때 작업 로그를 삭제합니다.

```
{
  "Logging": {
    "DeleteTaskLogs":true
  }
}
```

Note

각 복제 인스턴스에 대해서는 10일이 지난 로그를 AWS DMS 삭제합니다.

를 사용하여 AWS DMS API 호출 로깅 AWS CloudTrail

AWS DMS 는 사용자 AWS CloudTrail, 역할 또는 서비스가 수행한 작업에 대한 레코드를 제공하는 AWS 서비스와 통합됩니다 AWS DMS. CloudTrail은 AWS DMS 콘솔의 호출 및 API 작업에 대한 코드 호출을 포함하여에 대한 모든 AWS DMS API 호출을 이벤트 AWS DMS 로 캡처합니다. 추적을 생성하면 이벤트를 포함하여 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다 AWS DMS. 추적을 구성하지 않은 경우에도 이벤트 기록에서 CloudTrail 콘솔의 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 수행된 요청, 요청이 수행된 AWS DMS IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 설명은 [AWS CloudTrail 사용자 가이드](#)를 참조하십시오.

AWS DMS CloudTrail의 정보

AWS 계정을 생성할 때 계정에서 CloudTrail이 활성화됩니다. 에서 활동이 발생하면 AWS DMS해당 활동이 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에서 최근 이벤트를 보고 검색하고 다운로드할 수 있습니다. 자세한 내용은 [CloudTrail 이벤트 기록을 사용하여 이벤트 보기](#)를 참조하세요.

에 대한 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 AWS DMS생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 기본적으로 콘솔에서 추적을 생성하면 추적이 모든 AWS 리전에 적용됩니다. 추적은 AWS 파티션의 모든 AWS 리

전에서 이벤트를 로깅하고 지정한 Amazon S3 버킷으로 로그 파일을 전송합니다. 또한 CloudTrail 로그에서 수집된 이벤트 데이터를 추가로 분석하고 조치를 취하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음을 참조하세요.

- [추적 생성 개요](#)
- [CloudTrail 지원 서비스 및 통합](#)
- [CloudTrail에 대한 Amazon SNS 알림 구성](#)
- [여러 AWS 리전에서 CloudTrail 로그 파일 수신](#) 및 [여러 계정에서 CloudTrail 로그 파일 수신](#)

모든 AWS DMS 작업은 CloudTrail에서 로깅되며 [AWS Database Migration Service API 참조](#)에 문서화됩니다. 예를 들어 CreateReplicationInstance, TestConnection, StartReplicationTask 작업을 직접 호출하면 CloudTrail 로그 파일에 항목이 생성됩니다.

모든 이벤트 또는 로그 항목에는 요청을 생성했던 사용자에 관한 정보가 포함됩니다. ID 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 IAM 사용자 자격 증명 정보로 했는지 여부.
- 역할 또는 페더레이션 사용자의 임시 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 요청이 다른 AWS 서비스에서 이루어졌는지 여부입니다.

자세한 내용은 [CloudTrail userIdentity 요소](#)를 참조하세요.

AWS DMS 로그 파일 항목 이해

트레일이란 지정한 S3 버킷에 이벤트를 로그 파일로 입력할 수 있게 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스의 단일 요청을 나타내며 요청된 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보를 포함하고 있습니다. CloudTrail 로그 파일은 퍼블릭 API 직접 호출에 대한 순서 지정된 스택 추적이 아니기 때문에 특정 순서로 표시되지 않습니다.

다음은 RebootReplicationInstance 작업을 보여 주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
```

```
"principalId": "AKIAIOSFODNN7EXAMPLE:johndoe",
"arn": "arn:aws:sts::123456789012:assumed-role/admin/johndoe",
"accountId": "123456789012",
"accessKeyId": "ASIAIOSFODNN7EXAMPLE",
"sessionContext": {
  "attributes": {
    "mfaAuthenticated": "false",
    "creationDate": "2018-08-01T16:42:09Z"
  },
  "sessionIssuer": {
    "type": "Role",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::123456789012:role/admin",
    "accountId": "123456789012",
    "userName": "admin"
  }
},
"eventTime": "2018-08-02T00:11:44Z",
"eventSource": "dms.amazonaws.com",
"eventName": "RebootReplicationInstance",
"awsRegion": "us-east-1",
"sourceIPAddress": "72.21.198.64",
"userAgent": "console.amazonaws.com",
"requestParameters": {
  "forceFailover": false,
  "replicationInstanceArn": "arn:aws:dms:us-east-1:123456789012:rep:EX4MBJ2NMRDL3BMAYJ0XUGYPUE"
},
"responseElements": {
  "replicationInstance": {
    "replicationInstanceIdentifier": "replication-instance-1",
    "replicationInstanceStatus": "rebooting",
    "allocatedStorage": 50,
    "replicationInstancePrivateIpAddresses": [
      "172.31.20.204"
    ],
    "instanceCreateTime": "Aug 1, 2018 11:56:21 PM",
    "autoMinorVersionUpgrade": true,
    "engineVersion": "2.4.3",
    "publiclyAccessible": true,
    "replicationInstanceClass": "dms.t2.medium",
    "availabilityZone": "us-east-1b",
```

```
"kmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
f7bc0f8e-1a3a-4ace-9faa-e8494fa3921a",
  "replicationSubnetGroup": {
    "vpcId": "vpc-1f6a9c6a",
    "subnetGroupStatus": "Complete",
    "replicationSubnetGroupArn": "arn:aws:dms:us-
east-1:123456789012:subgrp:EDHRVRBAAAPONQAIYWP4NUW22M",
    "subnets": [
      {
        "subnetIdentifier": "subnet-cbfff283",
        "subnetAvailabilityZone": {
          "name": "us-east-1b"
        },
        "subnetStatus": "Active"
      },
      {
        "subnetIdentifier": "subnet-d7c825e8",
        "subnetAvailabilityZone": {
          "name": "us-east-1e"
        },
        "subnetStatus": "Active"
      },
      {
        "subnetIdentifier": "subnet-6746046b",
        "subnetAvailabilityZone": {
          "name": "us-east-1f"
        },
        "subnetStatus": "Active"
      },
      {
        "subnetIdentifier": "subnet-bac383e0",
        "subnetAvailabilityZone": {
          "name": "us-east-1c"
        },
        "subnetStatus": "Active"
      },
      {
        "subnetIdentifier": "subnet-42599426",
        "subnetAvailabilityZone": {
          "name": "us-east-1d"
        },
        "subnetStatus": "Active"
      },
    ]
  }
}
```

```

        "subnetIdentifier": "subnet-da327bf6",
        "subnetAvailabilityZone": {
            "name": "us-east-1a"
        },
        "subnetStatus": "Active"
    },
    ],
    "replicationSubnetGroupIdentifier": "default-vpc-1f6a9c6a",
    "replicationSubnetGroupDescription": "default group created by console
for vpc id vpc-1f6a9c6a"
},
    "replicationInstanceEniId": "eni-0d6db8c7137cb9844",
    "vpcSecurityGroups": [
        {
            "vpcSecurityGroupId": "sg-f839b688",
            "status": "active"
        }
    ],
    "pendingModifiedValues": {},
    "replicationInstancePublicIpAddresses": [
        "18.211.48.119"
    ],
    "replicationInstancePublicIpAddress": "18.211.48.119",
    "preferredMaintenanceWindow": "fri:22:44-fri:23:14",
    "replicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:EX4MBJ2NMRDL3BMAYJ0XUGYPUE",
    "replicationInstanceEniIds": [
        "eni-0d6db8c7137cb9844"
    ],
    "multiAZ": false,
    "replicationInstancePrivateIpAddress": "172.31.20.204",
    "patchingPrecedence": 0
    }
},
    "requestID": "a3c83c11-95e8-11e8-9d08-4b8f2b45bfd5",
    "eventID": "b3c4adb1-e34b-4744-bdeb-35528062a541",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}

```


AWS DMS 컨텍스트 로깅

AWS DMS 는 컨텍스트 로깅을 사용하여 진행 중인 마이그레이션에 대한 정보를 제공합니다. 컨텍스트 로깅은 다음과 같은 정보를 태스크의 CloudWatch 로그에 기록합니다.

- 태스크의 소스 및 대상 데이터베이스 연결에 대한 정보.
- 복제 태스크 동작. 태스크 로그를 사용하여 복제 문제를 진단할 수 있습니다.
- 소스 및 대상 데이터베이스에서 AWS DMS 실행되는 데이터가 없는 SQL 문입니다. SQL 로그를 사용하여 예상치 못한 마이그레이션 동작을 진단할 수 있습니다.
- 각 CDC 이벤트의 위치 세부 정보를 스트리밍합니다.

컨텍스트 로깅은 AWS DMS 버전 3.5.0 이상에서만 사용할 수 있습니다.

AWS DMS 는 기본적으로 컨텍스트 로깅을 켭니다. 컨텍스트 로깅을 제어하려면 `EnableLogContext` 태스크 설정을 `true` 또는 `false`로 설정하거나 콘솔에서 태스크를 수정합니다.

AWS DMS 는 3분마다 CloudWatch 로그의 복제 작업에 컨텍스트 로그 정보를 기록합니다. 복제 인스턴스에 애플리케이션 로그를 위한 충분한 공간이 있는지 확인하세요. 태스크 로그 관리에 대한 자세한 내용은 [AWS DMS 작업 로그 보기 및 관리](#) 섹션을 참조하세요.

주제

- [객체 유형](#)
- [로깅 예제](#)
- [제한 사항](#)

객체 유형

AWS DMS 는 다음 객체 유형에 대해 CloudWatch에서 컨텍스트 로깅을 생성합니다.

객체 유형	설명
TABLE_NAME	이러한 로그 항목에는 현재 태스크 매핑 규칙의 범위 내에 있는 테이블에 대한 정보가 들어 있습니다. 이러한 항목을 사용하여 마이그레이션 중 특정 기간 동안의 테이블 이벤트를 검사할 수 있습니다.

객체 유형	설명
SCHEMA_NAME	이러한 로그 항목에는 현재 태스크 매핑 규칙에서 사용하는 스키마에 대한 정보가 들어 있습니다. 이러한 항목을 사용하여 마이그레이션 중에 특정 기간 동안 사용 AWS DMS 중인 스키마를 확인할 수 있습니다.
TRANSACTION_ID	이러한 항목에는 소스 데이터베이스에서 캡처된 각 DML/DDI 변경 사항에 대한 트랜잭션 ID가 포함됩니다. 이러한 로그 항목을 사용하여 특정 트랜잭션 중에 발생한 변경 사항을 확인할 수 있습니다.
CONNECTION_ID	이러한 항목에는 연결 ID가 포함됩니다. 이러한 로그 항목을 사용하여 각 마이그레이션 단계에서 AWS DMS가 사용하는 연결을 확인할 수 있습니다.
STATEMENT	이러한 항목에는 각 마이그레이션 변경 사항을 가져오고, 처리하고, 적용하는 데 사용되는 SQL 코드가 포함됩니다.
STREAM_POSITION	이러한 항목에는 소스 데이터베이스에 대한 각 마이그레이션 작업의 트랜잭션 로그 파일 내 위치가 포함됩니다. 이러한 항목의 형식은 소스 데이터베이스 엔진 유형에 따라 다릅니다. CDC 전용 복제를 구성할 때 이 정보를 사용하여 복구 체크포인트의 시작 위치를 결정할 수도 있습니다.

로깅 예제

이 섹션에는 복제를 모니터링하고 복제 문제를 진단하는 데 사용할 수 있는 로그 레코드 예제가 포함되어 있습니다.

연결 로그 예제

이 섹션에는 연결 ID가 포함된 로그 샘플이 포함되어 있습니다.

```
2023-02-22T10:09:29 [SOURCE_CAPTURE ]I: Capture record 1 to internal
queue from Source {operation:START_REGULAR (43), connectionId:27598,
streamPosition:0000124A/6800A778.NOW} (streamcomponent.c:2920)

2023-02-22T10:12:30 [SOURCE_CAPTURE ]I: Capture record 0 to internal queue from
Source {operation:IDLE (51), connectionId:27598} (streamcomponent.c:2920)

2023-02-22T11:25:27 [SOURCE_CAPTURE ]I: Capture record 0 to internal queue
from Source {operation:IDLE (51), columnName:region, connectionId:27598}
(streamcomponent.c:2920)
```

태스크 동작 로그 예제

이 섹션에는 복제 태스크 로그 동작에 대한 로그 샘플이 포함되어 있습니다. 이 정보를 사용하여 IDLE 상태의 태스크와 같은 복제 문제를 진단할 수 있습니다.

다음 SOURCE_CAPTURE 로그는 소스 데이터베이스 로그 파일에서 읽을 수 있는 이벤트가 없음을 나타내며, 대상 데이터베이스에 적용할 AWS DMS CDC 구성 요소로부터 받은 이벤트가 없음을 나타내는 TARGET_APPLY 레코드를 포함합니다. 이러한 이벤트에는 이전에 적용된 이벤트 관련 컨텍스트 세부 정보도 포함됩니다.

```
2023-02-22T11:23:24 [SOURCE_CAPTURE ]I: No Event fetched from wal log
(postgres_endpoint_wal_engine.c:1369)
2023-02-22T11:24:29 [TARGET_APPLY ]I: No records received to load
or apply on target , waiting for data from upstream. The last context
is {operation:INSERT (1), tableName:sales_11, schemaName:public,
txnId:18662441, connectionId:17855, statement:INSERT INTO
"public"."sales_11"("sales_no","dept_name","sale_amount","sale_date","region") values
(?,?,?,?,?),
```

SQL 문 로그 예제

이 섹션에는 소스 및 대상 데이터베이스에서 실행되는 SQL 문에 대한 로그 샘플이 포함되어 있습니다. 로그에 표시되는 SQL 문에는 SQL 문만 표시되고 데이터는 표시되지 않습니다. 다음 TARGET_APPLY 로그는 대상에서 실행된 INSERT 문을 보여줍니다.

```
2023-02-22T11:26:07 [TARGET_APPLY    ]I: Applied record 2193305 to
target {operation:INSERT (1), tableName:sales_111, schemaName:public,
txnId:18761543, connectionId:17855, statement:INSERT INTO
"public"."sales_111"("sales_no","dept_name","sale_amount","sale_date","region") values
(?,?,?,?/?),
```

제한 사항

AWS DMS 컨텍스트 로깅에는 다음과 같은 제한 사항이 적용됩니다.

- 는 모든 엔드포인트 유형에 대해 최소 로깅을 AWS DMS 생성하지만 광범위한 엔진별 컨텍스트 로깅은 다음 엔드포인트 유형에만 사용할 수 있습니다. 이러한 엔드포인트 유형을 사용할 때는 컨텍스트 로깅을 활성화하는 것이 좋습니다.
- MySQL
- PostgreSQL
- Oracle
- Microsoft SQL Server
- MongoDB/Amazon DocumentDB
- Amazon S3

향상된 모니터링 대시보드

향상된 모니터링 대시보드는 모니터링 작업 및 복제 인스턴스와 관련된 중요 지표에 대한 포괄적인 가시성을 제공합니다. 이를 통해 추적하려는 특정 리소스에 대한 지표를 필터링, 집계 및 시각화할 수 있습니다. 대시보드는 기존 CloudWatch 지표를 직접 게시하므로 데이터 포인트 샘플링 시간을 변경하지 않고도 리소스 성능을 모니터링할 수 있습니다.

주제

- [향상된 모니터링 대시보드 개요](#)
- [향상된 모니터링 대시보드에서 지표 보기](#)
- [향상된 모니터링 대시보드 보기](#)
- [향상된 모니터링 지표 보존](#)

향상된 모니터링 대시보드 개요

향상된 모니터링 대시보드는 AWS DMS 콘솔에서 사용할 수 있습니다. 직관적인 시각화 및 차트가 포함된 사용자 친화적 인터페이스를 제공하므로 데이터 마이그레이션 프로세스를 효과적으로 모니터링, 분석 및 최적화할 수 있습니다. 향상된 모니터링을 사용하면 중앙 위치에서 모든 관련 정보를 확인하여 모니터링 프로세스를 간소화하고 잠재적 문제를 빠르게 식별할 수 있습니다.

향상된 모니터링 대시보드에서 태스크 및 복제 인스턴스에 대한 지표와 엔드포인트에 대한 세부 정보를 볼 수 있습니다. 또한 현재 리전의 활성 CloudWatch 경보 수와 서비스 상태도 추적할 수 있습니다. 대시보드는 AWS DMS를 사용할 수 있는 모든 상용 리전에서 사용할 수 있습니다. 이 대시보드를 사용하는 데 드는 추가 비용은 없습니다.

Note

향상된 모니터링 대시보드는 AWS DMS Serverless 복제를 지원하지 않습니다.

향상된 모니터링 대시보드에서 지표 보기

향상된 모니터링 대시보드에서 지표를 보려면 CloudWatch 권한이 있는 IAM 역할이 있어야 합니다. 또한 지표를 보려면 `cloudwatch:DescribeAlarms` 및 `health:DescribeEvents` 권한이 필요합니다. 마지막으로 향상된 모니터링 대시보드에서 CloudWatch 지표에 액세스하려면 IAM 정책 내에서 [GetMetricData](#) API에 액세스할 수 있는 권한도 있어야 합니다. 이러한 권한 또는 API의 하위 집합에 대한 권한이 없으면 향상된 모니터링 대시보드에서 지표를 볼 수 없습니다. 필요한 IAM 권한에 대한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

향상된 모니터링 대시보드 보기

AWS DMS 향상된 모니터링 대시보드에는 태스크 및 복제 인스턴스 보기의 두 가지 유형의 보기가 있습니다.

태스크 보기

AWS DMS의 향상된 대시보드의 태스크 보기는 데이터 마이그레이션 태스크에 대한 포괄적인 개요를 제공합니다. 이 보기는 직관적인 차트 및 시각화를 통해 태스크의 다양한 측면을 모니터링하고 분석할 수 있는 중앙 위치를 제공합니다.

태스크 보기는 마이그레이션 태스크와 관련된 주요 지표 및 통계에 대한 인사이트를 제공하므로 성능, 진행 상황 및 전반적인 상태를 더 잘 이해할 수 있습니다. 태스크 보기의 일부 주요 기능 및 이점은 다음과 같습니다.

- **태스크 상태 요약:** 이 차트는 실행, 중지, 실패 또는 완료와 같은 다양한 상태 범주에 걸친 태스크의 배포를 표시합니다. 주의가 필요한 태스크를 빠르게 식별하고 적절한 조치를 취할 수 있습니다.
- **성능 지표:** 이 차트는 처리량, 지연 시간, CPU 사용률 등과 같은 지표를 포함하여 태스크의 성능을 보여줍니다. 이러한 지표는 잠재적인 병목 현상을 식별하고 마이그레이션 프로세스를 최적화하는 데 도움이 됩니다.
- **오류 분석:** 태스크 실패 또는 오류가 발생하는 경우 태스크 보기는 발생한 오류 및 관련 로그 항목에 대한 자세한 정보를 제공합니다. 이 정보는 문제를 더 효율적으로 해결하는 데 도움이 될 수 있습니다.
- **과거 추세:** 대시보드는 과거 데이터를 통합하여 일정 기간 동안 태스크가 어떻게 수행되었는지 분석할 수 있습니다. 이러한 과거 추세를 기반으로 패턴을 식별하고, 진행 상황을 추적하고, 정보에 입각한 결정을 내릴 수 있습니다.
- **필터링 및 정렬:** 태스크 보기를 사용하면 태스크 이름, 태그 또는 특정 시간 범위와 같은 다양한 기준에 따라 태스크를 필터링하고 정렬할 수 있습니다. 이러한 유연성을 통해 요구 사항과 가장 관련이 있는 태스크 또는 측면에 집중할 수 있습니다.

다음 목록은 태스크 보기에서 볼 수 있는 지표를 설명합니다.

- **전체 로드 처리량 대역폭 - 소스:** 소스에서 전체 로드로 전송된 수신 데이터를 초당 KB 단위로 나타냅니다.
- **전체 로드 처리량 대역폭 - 대상:** 대상의 전체 로드에서 전송된 발신 데이터를 초당 KB 단위로 나타냅니다.
- **전체 로드 처리량 행 - 소스:** 대상의 전체 로드에서 들어오는 변경 사항을 초당 행 단위로 나타냅니다.
- **전체 로드 처리량 행 - 대상:** 대상에 대한 전체 로드의 발신 변경 사항을 초당 행 단위로 나타냅니다.
- **CDC 처리량 대역폭 - 소스:** 소스의 네트워크 대역폭을 초당 KB 단위로 나타냅니다.

CDC 처리량 대역폭은 샘플 포인트의 대역폭을 기록합니다. 어떠한 네트워크 트래픽도 찾을 수 없다면 이 값은 0입니다. CDC는 장시간 트랜잭션을 발행할 수 없기 때문에 네트워크 트래픽은 기록되지 않을 수 있습니다.

- **CDC 처리량 대역폭 - 대상:** 대상의 네트워크 대역폭을 초당 KB 단위로 나타냅니다.

CDC 처리량 대역폭은 샘플 포인트의 대역폭을 기록합니다. 어떠한 네트워크 트래픽도 찾을 수 없다면 이 값은 0입니다. CDC는 장시간 트랜잭션을 발행할 수 없기 때문에 네트워크 트래픽은 기록되지 않을 수 있습니다.

- **CDC 처리량 행 - 소스:** 소스에서 들어오는 태스크 변경 사항을 초당 행 단위로 나타냅니다.

- CDC 처리량 행 - 대상: 대상에 대한 발신 태스크 변경 사항을 초당 행 단위로 나타냅니다.
- CDC 지연 - 소스: 소스 엔드포인트에서 캡처한 마지막 이벤트와 AWS DMS 인스턴스의 현재 시스템 타임스탬프 간의 간격(초)입니다. 태스크 범위에서 변경 내용이 캡처되지 않은 경우 AWS DMS가 이 값을 0으로 설정합니다.
- CDC 지연 - 대상: 대상에 커밋되기를 대기하는 첫 이벤트 타임스탬프와 AWS DMS 인스턴스의 현재 타임스탬프 사이의 간격(초)입니다. 대상에 의해 처리되지 않은 트랜잭션이 있는 경우 이 값이 발생합니다. 그렇지 않고 모든 트랜잭션이 적용되는 경우 대상 지연 시간이 소스 지연 시간과 동일합니다. 대상 지연 시간은 소스 지연 시간보다 작지 않아야 합니다.
- CPU 사용률: 태스크가 여러 코어에 걸쳐 사용하는 CPU의 비율입니다. 태스크 CPUUtilization의 의미 체계는 복제 인스턴스 CPUUtilizaiton의 의미 체계와 약간 다릅니다. 하나의 vCPU가 완전히 사용되는 경우 100%를 나타내지만 여러 vCPU가 사용되는 경우에는 값이 100%를 초과할 수 있습니다.
- 메모리 사용량: 태스크에서 소비한 제어 그룹(cgroup) memory.usage_in_bytes를 나타냅니다. AWS DMS는 cgroup을 사용하여 메모리 및 CPU와 같은 시스템 리소스의 사용량을 제어합니다. 이 지표는 해당 태스크에 할당된 cgroup 내에서 태스크의 메모리 사용량을 메가바이트 단위로 나타냅니다.

cgroup 한도는 AWS DMS 복제 인스턴스 클래스에 사용할 수 있는 리소스를 기반으로 합니다. memory.usage_in_bytes는 메모리의 RSS(상주 세트 크기), 캐시 및 스왑 구성 요소로 구성됩니다. 필요한 경우 운영 체제가 캐시 메모리를 회수할 수 있습니다. 복제 인스턴스 지표인 AvailableMemory도 모니터링하는 것이 좋습니다.

AWS DMS는 CloudWatch 콘솔의 ReplicationInstanceIdentifier 및 ReplicationTaskIdentifier 차원을 합친 것과 비교하여 이 지표를 높입니다. ReplicationInstanceIdentifier, ReplicationTaskIdentifier 범주를 사용하여 이 지표를 볼 수 있습니다.

- 검증 레코드 수: 이 차트는 AWS DMS 태스크에 검증이 활성화된 경우에만 표시됩니다. 이는 AWS DMS 태스크에 사용할 수 있는 검증 지표의 조합으로, 다음을 포함합니다.
 - ValidationSucceededRecordCount - AWS DMS에서 검증한 분당 행 수입니다.
 - ValidationAttemptedRecordCount - 검증을 시도한 분당 행 수입니다.
 - ValidationFailedOverallCount - 검증이 실패한 행 수입니다.
 - ValidationSuspendedOverallCount - 검증이 일시 중지된 행 수입니다.
 - ValidationPendingOverallCount - 검증이 보류 중인 행 수입니다.

위의 지표 외에도 태스크 보기를 사용자 지정하고 위젯으로 추가하여 추가 지표를 포함할 수 있습니다. 이러한 지표에 대한 자세한 내용은 다음 설명서를 검토하는 것이 좋습니다.

- 마이그레이션 및 복제 AWS DMS 태스크에 사용할 수 있는 모든 지표는 [AWS DMS 작업 모니터링](#) 섹션을 참조하세요.
- 사용 가능한 모든 검증 관련 지표는 [데이터 유효성 검사](#) 섹션을 참조하세요.

복제 인스턴스 보기

복제 인스턴스 보기는 복제 인스턴스에 대한 포괄적인 개요를 제공하므로 데이터 복제 인프라를 효과적으로 모니터링하고 관리할 수 있습니다. 이 보기는 직관적인 차트 및 시각화를 통해 복제 인스턴스의 다양한 측면을 분석할 수 있는 중앙 위치를 제공합니다.

복제 인스턴스 보기는 복제 인스턴스와 관련된 주요 지표 및 통계에 대한 인사이트를 제공하므로 성능, 리소스 사용률 및 전반적인 상태를 더 잘 이해할 수 있습니다. 다음은 복제 인스턴스 보기의 몇 가지 주요 기능과 이점입니다.

- 인스턴스 상태 요약: 이 차트에는 가용 영역, 인스턴스 클래스, 엔진 버전, 할당된 스토리지, 현재 상태 등 선택한 복제 인스턴스의 세부 정보가 표시됩니다.
- 리소스 사용률: 대시보드는 CPU, 메모리 및 디스크 사용량과 같은 지표를 포함하여 복제 인스턴스의 리소스 사용률을 보여주는 차트를 제공합니다. 이러한 지표는 잠재적 리소스 제약 조건을 식별하고 복제 인프라를 최적화하는 데 도움이 됩니다.
- 과거 추세: 대시보드는 과거 데이터를 통합하여 일정 기간 동안 복제 인스턴스의 성능을 분석할 수 있습니다. 이러한 과거 추세를 기반으로 패턴을 식별하고, 변경 사항을 추적하고, 정보에 입각한 결정을 내릴 수 있습니다.
- 필터링 및 정렬: 복제 인스턴스 보기를 사용하면 인스턴스 이름 또는 특정 시간 범위와 같은 다양한 기준에 따라 복제 인스턴스를 필터링하고 정렬할 수 있습니다. 이러한 유연성을 통해 요구 사항과 가장 관련이 있는 인스턴스 또는 측면에 집중할 수 있습니다.

다음 목록은 복제 인스턴스 보기에서 볼 수 있는 지표를 설명합니다.

- CloudWatch 경보: AWS/DMS 네임스페이스의 경보 요약을 나타냅니다.
- CPU 사용률: 태스크가 여러 코어에 걸쳐 사용하는 CPU의 비율입니다. 태스크 CPUUtilization의 의미 체계는 복제 CPUUtilizaiton의 의미 체계와 약간 다릅니다. 하나의 vCPU가 완전히 사용되는 경우 100%를 나타내지만 여러 vCPU가 사용되는 경우에는 값이 100%를 초과할 수 있습니다.

- 인스턴스당 작업별 메모리 사용량: 작업이 차지하는 상주 설정 크기(RSS)를 나타냅니다. 이 크기는 기본 메모리(RAM)에 있는 작업이 차지하는 메모리 부분을 나타냅니다. 차지하는 메모리의 일부가 페이징 아웃되거나 실행 파일의 일부가 로드되지 않으므로 MemoryUsage에는 스왑 공간이나 파일 시스템에 있는 메모리가 포함하지 않습니다.
- 메모리: 사용 가능하거나 사용 중이거나 사용할 수 있는 메모리 양과 사용된 스왑 공간을 나타냅니다.
- 사용 가능한 메모리: 스와핑 없이 새 애플리케이션을 시작하는 데 사용할 수 있는 메모리의 추정 크기를 나타냅니다.
- 여유 메모리: 애플리케이션, 페이지 캐시, 커널 자체 데이터 구조에서 사용할 수 있는 물리적 메모리의 양을 나타냅니다.
- 사용 가능한 메모리: 사용 가능한 메모리는 사용 가능한 실제 여유 메모리를 나타내지 않습니다. 현재 사용 중인 메모리로, 다른 용도로 사용할 수 있습니다. 복제에 사용되는 버퍼와 캐시의 조합입니다.
- 스왑 사용량: 복제 인스턴스에서 사용된 스왑 공간의 크기를 나타냅니다.
- 여유 스토리지: 사용 가능한 스토리지 공간의 양을 바이트 단위로 나타냅니다.

이전 지표 외에도 복제 인스턴스 보기를 사용자 지정하고 위젯으로 추가하여 추가 지표를 포함할 수 있습니다. 이러한 지표에 대한 자세한 내용은 다음 설명서를 검토하는 것이 좋습니다.

- 마이그레이션 및 복제 AWS DMS 태스크에 사용할 수 있는 모든 지표는 [AWS DMS 작업 모니터링](#) 섹션을 참조하세요.
- 사용 가능한 모든 검증 관련 지표는 [데이터 유효성 검사](#) 섹션을 참조하세요.

향상된 모니터링 지표 보존

기본적으로 향상된 모니터링 지표는 CloudWatch 지표에 대한 보존 정책을 따릅니다. 보존 정책 변경에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 [CloudWatch Logs에서 로그 데이터 보존 기간 변경](#)을 참조하세요.

AWS DMS 이벤트 보기

AWS DMS 리소스에 대한 다음 이벤트 정보를 검색할 수 있습니다.

- 리소스 이름

- 리소스 유형
- 이벤트의 날짜 및 시간
- 이벤트 메시지 요약

지난 7일간의 이벤트를 표시하는 AWS 관리 콘솔의 이벤트 탭에서 이벤트에 액세스할 수 있습니다. [describe-events](#) AWS CLI 명령 또는 [DescribeEvents](#) DMS API 작업을 사용하여 이벤트를 검색할 수도 있습니다. AWS CLI 또는 DMS API를 사용하여 이벤트를 보는 경우 최대 지난 14일 동안의 이벤트를 검색할 수 있습니다.

Note

더 오랜 시간 동안 이벤트를 저장해야 하는 경우 이벤트를 EventBridge로 보내고 AWS DMS 이벤트 세부 정보를 다른 서비스에 전송하여 저장할 수 있습니다. 자세한 내용은 [AWS DMS에 대해 Amazon EventBridge 이벤트 사용하기](#)를 참조하세요.

AWS DMS 이벤트에 대한 설명은 [AWS DMS 이벤트 범주 및 이벤트 메시지를](#) 참조하세요.

DMS 콘솔

DMS 콘솔에서 지난 7일 동안의 모든 AWS DMS 이벤트를 보려면:

1. AWS Management Console에 로그인하고 <https://console.aws.amazon.com/dms/v2> DMS 콘솔을 엽니다.
2. 탐색 페이지에서 이벤트를 선택합니다.
3. 검색어를 입력하여 결과를 필터링합니다.

다음 예에서는 **replication-task** 문자로 필터링된 이벤트 목록을 보여줍니다.

☰ DMS > Events

Events (12) info			
Name	Type	Date	Event
dms-onboard-task-v1	replication-task	April 9, 2025 at 13:06:38 (UTC-07:00)	Replication task has stopped. Stop Reason FULL_LOAD_ONLY_FINISHED.
dms-onboard-task-v1	replication-task	April 9, 2025 at 13:05:33 (UTC-07:00)	Replication task has started.

AWS CLI

지난 한 시간 동안 생성된 모든 이벤트를 보려면 파라미터 없이 [describe-events](#) AWS DMS CLI 명령을 실행합니다.

```
aws dms describe-events
```

다음 샘플 출력은 복제 작업이 수정되었음을 보여줍니다.

```
{
  "Events": [
    {
      "SourceIdentifier": "dms-task",
      "SourceType": "replication-task",
      "Message": "Replication task has modified.",
      "EventCategories": [
        "configuration change"
      ],
      "Date": "2025-04-09T12:48:22.231000-07:00"
    }
  ]
}
```

지난 14일(20160분) 동안의 모든 AWS DMS 이벤트를 보려면 [describe-events](#) AWS CLI 명령을 실행하고 --duration 파라미터를 로 설정합니다**20160**.

```
aws dms describe-events --duration 20160
```

다음 예에서는 복제 dms-replication-instance에 대한 지정된 시간 범위의 이벤트를 보여 줍니다.

```
aws dms describe-events \
  --source-identifier dms-replication-instance \
  --source-type replication-instance \
  --start-time 2025-04-10T22:00Z \
  --end-time 2025-04-12T23:59Z
```

다음 샘플 출력은 복제 인스턴스의 상태를 보여줍니다.

```
{
  "Events": [
    {
      "SourceIdentifier": "dms-replication-instance",
      "SourceType": "replication-instance",
      "Message": "Replication application shutdown",

```

```
    "EventCategories": [],
    "Date": "2025-04-12T03:11:43.270000-07:00"
  },
  {
    "SourceIdentifier": "dms-replication-instance",
    "SourceType": "replication-instance",
    "Message": "Replication application restarted",
    "EventCategories": [],
    "Date": "2025-04-12T03:19:58.214000-07:00"
  }
]
}
```

AWS DMS API

[DescribeEvents](#) DMS API 작업을 실행하고 `--duration` 파라미터를 로 설정하여 지난 14일 동안의 모든 AWS DMS 인스턴스 이벤트를 볼 수 있습니다**20160**.

에서 Amazon EventBridge 이벤트 및 알림 작업 AWS Database Migration Service

Amazon EventBridge를 사용하여 복제 인스턴스 생성 또는 삭제와 같은 AWS DMS 이벤트가 발생할 때 알림을 제공할 수 있습니다. EventBridge는 이벤트를 수신하고 이벤트 규칙에 정의된 대로 이벤트 알림을 라우팅합니다. AWS 리전에 대해 Amazon EventBridge에서 지원하는 모든 형식으로 알림을 사용할 수 있습니다. Amazon EventBridge 사용에 대한 자세한 내용은 Amazon EventBridge 사용 설명서에서 [Amazon EventBridge란 무엇입니까?](#)를 참조하세요.

Note

Amazon EventBridge 이벤트 작업은 AWS DMS 버전 3.4.5 이상에서 지원됩니다.

EventBridge는 AWS DMS 환경 변경의 지표인 이벤트를 수신하고 규칙을 적용하여 이벤트를 알림 메커니즘으로 라우팅합니다. 규칙은 이벤트의 구조(이벤트 패턴이라고 함)를 기반으로 이벤트를 알림 메커니즘과 일치시킵니다.

AWS DMS 는 이벤트를 이벤트 규칙을 적용할 수 있는 범주로 그룹화하므로 해당 범주의 이벤트가 발생할 때 알림을 받을 수 있습니다. 예를 들어, 특정 복제 인스턴스의 생성 카테고리에 EventBridge 이벤트 규칙을 적용한다고 가정해 보겠습니다. 그러면 복제 인스턴스에 영향을 주는 생성 관련 이벤트가 발생할 때마다 알림을 받게 됩니다. 복제 인스턴스의 구성 변경 카테고리에 규칙을 적용하면 복제 인스턴스의 구성이 변경될 때마다 메시지가 수신됩니다. 에서 제공하는 이벤트 범주 목록은 다음 AWS DMS 이벤트 범주 및 이벤트 메시지를 AWS DMS참조하세요.

Note

events.amazonaws.com로부터 게시를 허용하려면 Amazon SNS 주제의 액세스 정책을 업데이트해야 합니다. 자세한 내용은 Amazon EventBridge 사용 설명서의 [Amazon EventBridge에 대한 리소스 기반 정책 사용](#)을 참조하세요.

이벤트 구독을 Amazon EventBridge로 이전하는 방법에 대한 자세한 내용은 [활성 이벤트 구독을 DMS에서 Amazon EventBridge로 마이그레이션](#)을 참조하세요.

Amazon SNS 문자 메시지 사용에 대한 자세한 내용은 [Amazon SNS를 사용한 SMS 알림 메시지 전송 및 수신](#)을 참조하세요.

에 Amazon EventBridge 이벤트 규칙 사용 AWS DMS

Amazon EventBridge는 사용자가 EventBridge 이벤트 규칙을 생성할 때 입력한 주소로 이벤트 알림을 보냅니다. 여러 가지 규칙을 생성할 수 있습니다. 예를 들어 모든 이벤트 알림 메시지를 수신하는 하나의 규칙과 프로덕션 DMS 리소스의 중요 이벤트만 포함하는 다른 규칙을 생성할 수 있습니다. EventBridge에서 이벤트 알림을 활성화 또는 비활성화할 수도 있습니다.

AWS DMS 이벤트에 반응하는 Amazon EventBridge 규칙을 생성하려면

- [Amazon EventBridge 사용 설명서의 이벤트에 반응하는 Amazon EventBridge 규칙 생성](#)에 설명된 단계를 수행하고 AWS DMS 이벤트에 대한 규칙을 생성합니다. EventBridge
 - a. EventBridge가 규칙의 이벤트 패턴과 일치하는 이벤트를 수신할 때 수행할 작업을 지정합니다. 이벤트가 일치하면 EventBridge는 이벤트를 전송하고 규칙에 정의된 작업을 간접 호출합니다.
 - b. 서비스 제공업체에서 AWS를 선택합니다.
 - c. 서비스 이름에서 Server Migration Service (SMS)를 선택합니다.

이제부터 이벤트 알림 메시지를 수신할 수 있습니다.

다음 JSON 예제는 AWS DMS 서비스에 대한 EventBridge 이벤트 모델을 보여줍니다.

```
{
  "version": "0",
  "id": "11a11b11-222b-333a-44d4-01234a5b67890",
  "detail-type": "DMS Replication Task State Change",
  "source": "aws.dms",
  "account": "0123456789012",
  "time": "1970-01-01T00:00:00Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:dms:us-east-1:012345678901:task:AAAABBBB0CCCCDDDEEEEEE1FFFF2GGG3FFFFFFF3"
  ],
  "detail": {
    "type": "REPLICATION_TASK",
    "category": "StateChange",
    "eventType": "REPLICATION_TASK_STARTED",
    "eventId": "DMS-EVENT-0069",
    "resourceLink": "https://console.aws.amazon.com/dms/v2/home?region=us-east-1#taskDetails/taskName",
  }
}
```

```

    "detailMessage": "Replication task started, with flag = fresh start"
  }
}

```

알림을 받을 수 있는 카테고리 및 이벤트 목록은 다음 섹션을 참조하세요.

AWS DMS 이벤트 범주 및 이벤트 메시지

AWS DMS는 식별할 수 있는 범주에서 상당한 수의 이벤트를 생성합니다. 각 카테고리는 복제 인스턴스 또는 복제 태스크 소스 형식에 적용됩니다.

주제

- [복제 인스턴스 이벤트 메시지](#)
- [복제 태스크 이벤트 메시지](#)
- [복제 이벤트 메시지](#)

복제 인스턴스 이벤트 메시지

다음 표에는 복제 인스턴스 소스 유형에서 가능한 카테고리 및 이벤트가 나와 있습니다.

카테고리	이벤트 ID	설명
만들기	DMS-EVENT-0067	복제 인스턴스를 생성 중입니다.
삭제	DMS-EVENT-0066	복제 인스턴스를 삭제 중입니다.
구성 변경	DMS-EVENT-0012	이 복제 인스턴스의 복제 인스턴스 클래스가 변경되고 있습니다.
구성 변경	DMS-EVENT-0018	복제 인스턴스의 스토리지가 증가하고 있습니다.
구성 변경	DMS-EVENT-0024	복제 인스턴스가 다중 AZ 구성으로 전환 중입니다.

카테고리	이벤트 ID	설명
구성 변경	DMS-EVENT-0030	복제 인스턴스가 단일 AZ 구성으로 전환 중입니다.
정비	DMS-EVENT-0026	복제 인스턴스의 오프라인 유지 관리가 진행 중입니다. 현재 복제 인스턴스를 사용할 수 없습니다.
만들기	DMS-EVENT-0005	복제 인스턴스를 생성 중입니다.
삭제	DMS-EVENT-0003	복제 인스턴스가 삭제되었습니다.
구성 변경	DMS-EVENT-0014	이 복제 인스턴스의 복제 인스턴스 클래스가 변경되었습니다.
구성 변경	DMS-EVENT-0017	복제 인스턴스의 스토리지가 증가되었습니다.
구성 변경	DMS-EVENT-0025	복제 인스턴스가 다중 AZ 구성으로 전환이 완료되었습니다.
구성 변경	DMS-EVENT-0029	복제 인스턴스가 단일 AZ 구성으로 전환이 완료되었습니다.
정비	DMS-EVENT-0047	복제 인스턴스의 관리 소프트웨어가 업데이트되었습니다.
유지 관리	DMS-EVENT-0027	복제 인스턴스의 오프라인 유지 관리가 완료되었습니다. 이제 복제 인스턴스를 사용할 수 있습니다.
정비	DMS-EVENT-0068	복제 인스턴스가 업그레이드할 수 없는 상태입니다.
장애 조치	DMS-EVENT-0034	장애 조치를 너무 자주 요청하면 일반 장애 조치 이벤트 대신 이 이벤트가 발생합니다.

카테고리	이벤트 ID	설명
실패	DMS-EVENT-0031	복제 인스턴스가 %s 상태로 전화되었습니다.
실패	DMS-EVENT-0036	호환되지 않는 네트워크로 인해 복제 인스턴스에 장애가 발생했습니다.
실패	DMS-EVENT-0037	서비스가 데이터 볼륨을 암호화하는 데 사용된 KMS 키에 액세스할 수 없습니다.
실패		복제 인스턴스에서 호환되지 않는 파라미터를 사용했습니다.
장애 조치		안전한 상태에서 사용자 요청 장애 조치를 시작하기 위한 대기 시간이 초과되었습니다.
장애 조치	DMS-EVENT-0013	다중 AZ 복제 인스턴스에서 장애 조치가 시작되었습니다.
장애 조치	DMS-EVENT-0049	다중 AZ 복제 인스턴스에서 장애 조치가 완료되었습니다.
장애 조치	DMS-EVENT-0050	다중 AZ 활성화가 시작되었습니다.
장애 조치	DMS-EVENT-0051	다중 AZ 활성화가 완료되었습니다.
StateChange		일반 및 느린 쿼리 로그가 자동으로 %s(으)로 전환되었습니다.

카테고리	이벤트 ID	설명
StateChange		AWS DMS 가 애플리케이션 인스턴스 %s의 KMS 암호화 키에 액세스할 수 없습니다. 이는 키가 비활성화되었거나 키에 액세스할 수 AWS DMS 없기 때문일 수 있습니다. 이 상태가 계속되면 애플리케이션이 액세스할 수 없음 상태가 됩니다. 자세한 내용은 AWS DMS 설명서의 문제 해결 섹션을 참조하세요.
StateChange		AWS DMS 이제가 애플리케이션 인스턴스 %s의 KMS 암호화 키에 성공적으로 액세스할 수 있습니다.
StateChange		Amazon DMS가 애플리케이션 인스턴스 %s의 KMS 암호화 키에 액세스할 수 없습니다. 이 애플리케이션은 액세스할 수 없음 상태가 됩니다. 자세한 내용은 Amazon RDS 설명서의 문제 해결 섹션을 참조하세요.
StateChange		복제 인스턴스 생성의 일환으로 HM에서 앱을 다시 시작
StateChange		복제 인스턴스 삭제의 일환으로 HM에서 앱을 종료
장애 조치	DMS-EVENT-0015	스탠바이로의 다중 AZ 장애 조치.
LowStorage	DMS-EVENT-0007	복제 인스턴스의 여유 스토리지가 부족합니다.
LowStorage		할당된 inode가 모두 소진되었습니다. 해결하려면 스토리지 규모를 조정해야 합니다.

복제 태스크 이벤트 메시지

다음 표에는 복제 태스크 소스 유형에서 가능한 카테고리 및 이벤트가 나와 있습니다.

카테고리	이벤트 ID	설명
실패	DMS-EVENT-0078	복제 태스크가 실패했습니다.
실패	DMS-EVENT-0082	태스크 데이터 정리 호출이 실패했습니다.
상태 변경	DMS-EVENT-0081	테이블 세부 정보 다시 로드가 요청되었습니다.
상태 변경		복제 태스크가 복사되었습니다.
상태 변경		복제 태스크 복사가 실패했습니다.
상태 변경		복제 태스크가 이동되었습니다.
상태 변경		복제 상태 이동이 실패했습니다.
상태 변경		대상 태스크를 생성하지 못했습니다.
상태 변경		복제 태스크 평가 실행이 시작되었습니다.
상태 변경		복제 태스크 평가 실행이 성공적으로 완료되었습니다.
상태 변경		복제 태스크 평가 실행이 완료되었지만 실패했습니다.
StateChange		복제 태스크 평가 실행이 경고와 함께 완료되었습니다.
StateChange		복제 태스크 평가 실행이 오류와 함께 완료되었습니다.

카테고리	이벤트 ID	설명
StateChange		복제 태스크 평가 실행 %s이(가) 취소되었습니다.
StateChange		복제 태스크 평가 실행 %s이(가) 삭제되었습니다.
StateChange		복제 태스크 평가 실행이 리소스를 프로비저닝하지 못했습니다.
StateChange		복제 태스크가 실패했습니다.
만들기		복제 태스크가 생성되었습니다.
ConfigurationChange		복제 태스크가 수정되었습니다.
실패		복제 태스크가 실패했습니다.
StateChange	DMS-EVENT-0091	읽기가 일시 중지되었습니다. 스왑 파일 한도에 도달했습니다.
StateChange	DMS-EVENT-0092	읽기가 일시 중지되었습니다. 디스크 사용량 한도에 도달했습니다.
StateChange	DMS-EVENT-0093	읽기가 다시 시작되었습니다.
StateChange	DMS-EVENT-0069	복제 태스크가 다음 설정으로 시작되었습니다. taskType: %s, startType: %s
StateChange	DMS-EVENT-0079	복제 태스크가 중지되었습니다.
삭제	DMS-EVENT-0073	복제 태스크가 삭제되었습니다.

복제 이벤트 메시지

다음 표에는 복제 소스 유형에서 가능한 카테고리 및 이벤트가 나와 있습니다.

범주	설명
상태 변경	DMS 복제 스케일 업 이벤트.
상태 변경	DMS 복제 스케일 다운 이벤트.
상태 변경	DMS 복제 규모 조정 이벤트가 완료되었습니다.
상태 변경	DMS 복제가 생성되었습니다.
상태 변경	DMS 복제가 초기화되고 있습니다.
상태 변경	DMS 복제가 메타데이터 수집을 위한 리소스를 준비하고 있습니다.
상태 변경	DMS 복제에 연결된 연결을 테스트 중입니다.
상태 변경	DMS 복제가 메타데이터를 가져오는 중입니다.
상태 변경	DMS 복제가 용량을 계산하는 중입니다.
상태 변경	DMS 복제가 용량을 프로비저닝하는 중입니다.
상태 변경	DMS 복제가 프로비저닝되었습니다.
상태 변경	DMS 복제가 시작되었습니다.
상태 변경	DMS 복제가 실행 중입니다.
상태 변경	DMS 복제가 중지 중입니다.
상태 변경	DMS 복제가 중지되었습니다.
상태 변경	DMS 복제가 수정되고 있습니다.
상태 변경	DMS 복제를 삭제 중입니다.
상태 변경	DMS 복제가 용량을 프로비저닝 해제하는 중입니다.

범주	설명
상태 변경	DMS 복제가 프로비저닝 해제되었습니다.
실패	DMS 복제가 실패했습니다.

에서 Amazon SNS 이벤트 및 알림 작업 AWS Database Migration Service

AWS DMS 3.4.5 릴리스부터 이후 버전에서는 AWS DMS 이벤트가 발생할 때 Amazon EventBridge를 사용하여 알림을 제공하는 것이 좋습니다. 에서 EventBridge 이벤트를 사용하는 방법에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [에서 Amazon EventBridge 이벤트 및 알림 작업 AWS Database Migration Service](#).

이벤트 구독을 Amazon EventBridge로 이전

다음 AWS CLI 명령을 사용하여 활성 이벤트 구독을 DMS에서 Amazon EventBridge로 한 번에 최대 10개까지 마이그레이션할 수 있습니다.

```
update-subscriptions-to-event-bridge [--force-move | --no-force-move]
```

기본적으로는 복제 인스턴스가 AWS DMS 3.4.5 이상 버전인 경우에만 활성 이벤트 구독을 AWS DMS 마이그레이션합니다. 이 기본 동작을 재정의하려면 `--force-move` 옵션을 사용합니다. 하지만 복제 인스턴스가 업그레이드되지 않은 경우 Amazon EventBridge에서 일부 유형의 이벤트를 사용하지 못할 수 있습니다.

`update-subscriptions-to-event-bridge` CLI 명령을 실행하려면 AWS Identity and Access Management (IAM) 사용자는 다음 정책 권한이 있어야 합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "SNS:GetTopicAttributes",
        "SNS:SetTopicAttributes",
        "events:PutTargets",
        "events:EnableRule",
        "events:PutRule"
      ],
      "Resource": "*"
    }
  ]
}
```

```
]
}
```

구독을 EventBridge로 이전하는 방법에 대한 자세한 내용은 AWS Database Migration Service API 참조의 [UpdateSubscriptionsToEventBridge](#)를 참조하세요.

Amazon SNS 이벤트 및 알림 작업

AWS DMS 버전 3.4.5 이하에서는 다음과 같이 이벤트 및 알림 작업을 지원합니다.

AWS Database Migration Service(AWS DMS)는 Amazon Simple Notification Service(Amazon SNS)를 사용하여 복제 인스턴스 생성 또는 삭제와 같은 AWS DMS 이벤트가 발생할 때 알림을 제공할 수 있습니다. AWS 리전에 따라 Amazon SNS에서 지원하는 형식으로 이메일 메시지, 문자 또는 HTTP 엔드포인트 직접 호출 등의 알림을 사용할 수 있습니다.

AWS DMS는 이벤트를 구독할 수 있는 범주로 그룹화하므로 해당 범주의 이벤트가 발생할 때 알림을 받을 수 있습니다. 예를 들어 임의의 복제 인스턴스에 대한 생성 카테고리를 구독할 경우 생성 관련 이벤트가 발생하여 복제 인스턴스에 영향을 끼칠 때마다 알림 메시지가 수신됩니다. 복제 인스턴스에서 구성 변경 카테고리를 구독하면 복제 인스턴스의 구성이 변경될 때마다 메시지가 수신됩니다. 또한 이벤트 알림 메시지 구독이 변경되어도 알림 메시지가 수신됩니다. AWS DMS에서 제공하는 이벤트 범주 목록은 [AWS SNS 알림을 위한 DMS 이벤트 범주 및 이벤트 메시지](#) 다음 단원을 참조하십시오.

AWS DMS는 이벤트 구독을 생성할 때 제공한 주소로 이벤트 알림을 보냅니다. 모든 이벤트 알림 메시지를 수신하거나 프로덕션 DMS 리소스의 중요 이벤트만 수신하는 등 다른 구독을 복수로 생성하는 것도 가능합니다. DMS 콘솔에서 활성화됨 옵션을 선택 취소하거나 AWS DMS API를 사용하여 Enabled 파라미터를 false로 설정하여 구독을 삭제하지 않고도 알림을 쉽게 끌 수 있습니다.

Note

AWS SMS 문자 메시지를 사용하는 DMS 이벤트 알림은 현재 Amazon SNS가 지원되는 모든 AWS 리전의 AWS DMS 리소스에서 사용할 수 있습니다. Amazon SNS가 SMS 메시징을 지원하는 AWS 리전 및 국가 목록은 [지원되는 리전 및 국가](#)를 참조하세요.

SNS 문자 메시지 사용에 대한 자세한 내용은 [Amazon SNS를 사용한 SMS 알림 메시지 전송 및 수신](#)을 참조하세요.

AWS DMS 이벤트 알림은 CloudWatch 또는 EventBridge의 CloudTrail 이벤트와 다릅니다. CloudWatch EventBridge CloudTrail 이벤트 알림은 모든 API 간접 호출을 통해 생성할 수 있습니다. DMS는 DMS 이벤트가 발생한 경우에만 알림을 보냅니다.

AWS DMS는 구독 식별자를 사용하여 각 구독을 식별합니다. 동일한 Amazon SNS 주제에 여러 AWS DMS 이벤트 구독을 게시할 수 있습니다. 이벤트 알림을 사용하면 Amazon SNS 요금이 적용됩니다. Amazon SNS 비용 청구에 대한 자세한 내용은 [Amazon SNS 요금](#)을 참조하세요.

Amazon SNS를 사용하여 AWS DMS 이벤트를 구독하려면 다음 프로세스를 사용합니다.

1. Amazon SNS 주제를 생성합니다. 주제에서 수신할 알림 유형과 알림이 도달할 주소 또는 번호를 지정합니다.
2. AWS Management Console AWS CLI, 또는 AWS DMS API를 사용하여 AWS DMS 이벤트 알림 구독을 생성합니다.
3. AWS DMS는 구독과 함께 제출한 주소로 승인 이메일 또는 SMS 메시지를 보냅니다. 구독을 확인하려면, 승인 이메일이나 SMS 메시지의 링크를 클릭합니다.
4. 구독을 확인하면 AWS DMS 콘솔의 이벤트 구독 섹션에서 구독 상태가 업데이트됩니다.
5. 이제부터 이벤트 알림 메시지가 수신됩니다.

알림을 받을 수 있는 카테고리 및 이벤트 목록은 다음 섹션을 참조하세요. AWS DMS 이벤트 구독 및 작업에 대한 자세한 내용은 섹션을 참조하세요 [SNS를 사용하여 AWS DMS 이벤트 알림 구독](#).

AWS SNS 알림을 위한 DMS 이벤트 범주 및 이벤트 메시지

Important

AWS DMS 3.4.5 릴리스부터 이후 버전에서는 AWS DMS 이벤트가 발생할 때 Amazon EventBridge를 사용하여 알림을 제공하는 것이 좋습니다. 에서 EventBridge 이벤트를 사용하는 방법에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [에서 Amazon EventBridge 이벤트 및 알림 작업 AWS Database Migration Service](#).

AWS DMS는 범주에서 DMS 콘솔 또는 AWS DMS AWS API를 사용하여 구독할 수 있는 많은 수의 이벤트를 생성합니다. 각 범주는 소스 유형에 적용되며, 현재 AWS DMS는 복제 인스턴스 및 복제 작업 소스 유형을 지원합니다.

다음 표에는 복제 인스턴스 원본 형식에서 가능한 카테고리 및 이벤트가 나와 있습니다.

카테고리	DMS 이벤트 ID	설명
구성 변경	DMS-EVENT-0012	이 복제 인스턴스의 복제 인스턴스 클래스가 변경되고 있습니다.
구성 변경	DMS-EVENT-0014	이 복제 인스턴스의 복제 인스턴스 클래스가 변경되었습니다.
구성 변경	DMS-EVENT-0018	복제 인스턴스의 스토리지가 증가하고 있습니다.
구성 변경	DMS-EVENT-0017	복제 인스턴스의 스토리지가 증가되었습니다.
구성 변경	DMS-EVENT-0024	복제 인스턴스가 다중 AZ 구성으로 전환 중입니다.
구성 변경	DMS-EVENT-0025	복제 인스턴스가 다중 AZ 구성으로 전환이 완료되었습니다.
구성 변경	DMS-EVENT-0030	복제 인스턴스가 단일 AZ 구성으로 전환 중입니다.
구성 변경	DMS-EVENT-0029	복제 인스턴스가 단일 AZ 구성으로 전환이 완료되었습니다.
만들기	DMS-EVENT-0067	복제 인스턴스를 생성 중입니다.
만들기	DMS-EVENT-0005	복제 인스턴스가 생성되었습니다.
삭제	DMS-EVENT-0066	복제 인스턴스를 삭제 중입니다.
삭제	DMS-EVENT-0003	복제 인스턴스가 삭제되었습니다.
정비	DMS-EVENT-0047	복제 인스턴스의 관리 소프트웨어가 업데이트되었습니다.
정비	DMS-EVENT-0026	복제 인스턴스의 오프라인 유지 관리가 진행 중입니다. 현재 복제 인스턴스를 사용할 수 없습니다.
정비	DMS-EVENT-0027	복제 인스턴스의 오프라인 유지 관리가 완료되었습니다. 이제 복제 인스턴스를 사용할 수 있습니다.
정비	DMS-EVENT-0068	복제 인스턴스가 업그레이드할 수 없는 상태입니다.

카테고리	DMS 이벤트 ID	설명
LowStorage	DMS-EVENT-0007	복제 인스턴스가 할당된 스토리지의 90% 이상을 사용하였습니다. 여유 스토리지 공간 지표를 사용하여 복제 인스턴스에 대한 스토리지 공간을 모니터링할 수 있습니다.
장애 조치	DMS-EVENT-0013	다중 AZ 복제 인스턴스에서 장애 조치가 시작되었습니다.
장애 조치	DMS-EVENT-0049	다중 AZ 복제 인스턴스에서 장애 조치가 완료되었습니다.
장애 조치	DMS-EVENT-0015	스탠바이로의 다중 AZ 장애 조치가 완료되었습니다.
장애 조치	DMS-EVENT-0050	다중 AZ 활성화가 시작되었습니다.
장애 조치	DMS-EVENT-0051	다중 AZ 활성화가 완료되었습니다.
장애 조치	DMS-EVENT-0034	장애 조치를 너무 자주 요청하면 일반 장애 조치 이벤트 대신 이 이벤트가 발생합니다.
실패	DMS-EVENT-0031	복제 인스턴스에 스토리지 장애가 발생했습니다.
실패	DMS-EVENT-0036	호환되지 않는 네트워크로 인해 복제 인스턴스에 장애가 발생했습니다.
실패	DMS-EVENT-0037	서비스는 데이터 볼륨을 암호화하는 데 사용되는 AWS KMS 키에 액세스할 수 없습니다.

다음 표에는 복제 작업 원본 형식에서 가능한 카테고리 및 이벤트가 나와 있습니다.

카테고리	DMS 이벤트 ID	설명
상태 변경	DMS-EVENT-0069	복제 태스크가 시작되었습니다.
상태 변경	DMS-EVENT-0081	테이블 세부 정보 다시 로드가 요청되었습니다.

카테고리	DMS 이벤트 ID	설명
상태 변경	DMS-EVENT-0079	복제가 중지되었습니다.
상태 변경	DMS-EVENT-0091	읽기가 일시 중지되었습니다. 스왑 파일 한도에 도달했습니다.
상태 변경	DMS-EVENT-0092	읽기가 일시 중지되었습니다. 디스크 사용량 한도에 도달했습니다.
상태 변경	DMS-EVENT-0093	읽기가 재개되었습니다.
실패	DMS-EVENT-0078	복제 태스크가 실패했습니다.
실패	DMS-EVENT-0082	태스크 삭제 호출이 태스크 데이터를 정리하지 못했습니다.
구성 변경	DMS-EVENT-0080	복제 태스크가 수정되었습니다.
삭제	DMS-EVENT-0073	복제 태스크가 삭제되었습니다.
만들기	DMS-EVENT-0074	복제 태스크가 생성되었습니다.

다음 표에는 마이그레이션 전 평가 소스 유형에서 가능한 카테고리 및 이벤트가 나와 있습니다.

카테고리	DMS 이벤트 ID	설명
상태 변경	DMS-EVENT-0083	마이그레이션 전 평가가 시작되었습니다.
상태 변경	DMS-EVENT-0084	마이그레이션 전 평가가 통과되었습니다.
상태 변경	DMS-EVENT-0085	마이그레이션 전 평가는 실패와 함께 완료되었습니다.
상태 변경	DMS-EVENT-0086	마이그레이션 전 평가가 경고와 함께 완료되었습니다.

카테고리	DMS 이벤트 ID	설명
상태 변경	DMS-EVENT-0087	실행 중 마이그레이션 전 평가에서 및 오류가 발생했습니다.
상태 변경	DMS-EVENT-0088	마이그레이션 전 평가가 취소되었습니다.
상태 변경	DMS-EVENT-0089	마이그레이션 전 평가가 삭제되었습니다.
상태 변경	DMS-EVENT-0090	리소스를 프로비저닝하는 동안 마이그레이션 전 평가에서 오류가 발생했습니다.

다음 예제에서는 상태 변경 범주가 있는 AWS DMS 이벤트 구독을 보여줍니다.

```
Resources:
  DMSEvent:
    Type: AWS::DMS::EventSubscription
    Properties:
      Enabled: true
      EventCategories: State Change
      SnsTopicArn: arn:aws:sns:us-east-1:123456789:testSNS
      SourceIds: []
      SourceType: replication-task
```

SNS를 사용하여 AWS DMS 이벤트 알림 구독

Important

AWS DMS 3.4.5 릴리스부터 이후 버전에서는 AWS DMS 이벤트가 발생할 때 Amazon EventBridge를 사용하여 알림을 제공하는 것이 좋습니다. 에서 EventBridge 이벤트를 사용하는 방법에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [에서 Amazon EventBridge 이벤트 및 알림 작업 AWS Database Migration Service](#).

AWS DMS 이벤트가 발생할 때 알림을 받을 수 있도록 AWS DMS 이벤트 알림 구독을 생성할 수 있습니다. 구독을 생성하는 가장 간단한 방법은 AWS DMS 콘솔을 사용하는 것입니다. 알림 구독에서는 알림을 보내는 방법 및 위치를 선택합니다. 알림을 받을 소스 유형을 지정합니다. 현재 복제 인스턴스 및 복제 작업 소스 유형을 AWS DMS 지원합니다. 선택한 소스 유형에 따라 이벤트 알림 메시지를 수신하려는 이벤트 카테고리 및 소스를 선택합니다.

사용 AWS Management Console

Important

AWS DMS 3.4.5 릴리스부터 이후 버전에서는 AWS DMS 이벤트가 발생할 때 Amazon EventBridge를 사용하여 알림을 제공하는 것이 좋습니다. 에서 EventBridge 이벤트를 사용하는 방법에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [에서 Amazon EventBridge 이벤트 및 알림 작업 AWS Database Migration Service](#).

콘솔을 사용하여 Amazon SNS로 AWS DMS 이벤트 알림을 구독하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

IAM 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다.

2. 탐색 창에서 [이벤트 구독]을 선택합니다.
3. 이벤트 구독 페이지에서 이벤트 구독 생성을 선택합니다.
4. 이벤트 구독 생성 페이지에서 다음을 수행합니다.
 - a. 세부 정보의 이름에서 이벤트 알림 구독의 이름을 입력합니다.
 - b. 활성화됨을 선택하여 구독을 활성화합니다. 구독만 생성하고 알림 메시지를 아직 전송하지 않으려면 활성화됨을 선택하지 마세요.
 - c. 대상에서 기존 주제, 새 이메일 주제 생성 또는 새 SMS 주제 생성을 선택하여 알림을 전송합니다. 알림을 전송할 기존 Amazon SNS 주제가 있거나 주제를 생성해야 합니다. 주제를 생성하는 경우 알림이 전송될 이메일 주소를 입력할 수 있습니다.
 - d. 이벤트 소스의 소스 유형에서 소스 유형을 선택합니다. 옵션은 복제 인스턴스 및 복제 태스크뿐입니다.
 - e. 선택한 소스 유형에 따라 이벤트 알림 메시지를 수신하고자 하는 이벤트 카테고리 및 소스를 선택합니다.

Create event subscription

Details

Name

The name for your event subscription

☒ Enabled

Target

Send notification to

- ☐ Existing topics
- ☒ Create new email topic
- ☐ Create new SMS topic

Topic name**With these recipients**

Email addresses or phone numbers of SMS enabled devices to send the notifications to

Event source

Source type

Source Type of resource this subscription will consume events from

Event categories

- ☒ All event categories
- ☐ Select specific event categories

Replication instance

- ☒ All instances
- ☐ Select specific instances

- f. 이벤트 구독 생성을 선택합니다.

AWS DMS 콘솔은 구독이 생성 중임을 나타냅니다.

Note

AWS DMS API 및 CLI를 사용하여 Amazon SNS 이벤트 알림 구독을 생성할 수도 있습니다. 자세한 내용은 AWS DMS API 참조의 [CreateEventSubscription](#) 및 AWS DMS CLI 참조의 [create-event-subscription](#)을 참조하세요.

SNS 주제의 액세스 정책 검증

SNS 액세스 정책에는 AWS DMS가 SNS 주제에 이벤트를 게시할 수 있는 권한이 필요합니다. 다음 절차에 설명된 대로 액세스 정책을 검증하고 업데이트할 수 있습니다.

액세스 정책을 검증하려면

1. Amazon SNS 콘솔을 엽니다.
2. 탐색 패널에서 주제를 선택하고 DMS 알림을 받을 주제를 선택합니다.
3. 액세스 정책 탭을 선택합니다.

SNS 액세스 정책에서 AWS DMS가 SNS 주제에 이벤트를 게시하도록 허용하지 않는 경우 정책을 업데이트할 수 있습니다.

액세스 정책을 업데이트하려면

1. 주제 페이지의 세부 정보 섹션에서 편집을 선택합니다.
2. 액세스 정책 섹션을 확장하고 다음 정책을 JSON 편집기에 연결합니다.

```
{
  "Sid": "dms-allow-publish",
  "Effect": "Allow",
  "Principal": {
    "Service": "dms.amazonaws.com"
  },
  "Action": "sns:Publish",
  "Resource": "your-SNS-topic-ARN"
}
```



```
}
```

이벤트를 주제에 게시하는 DMS EventSubscription Arn인 `aws:SourceArn` 조건을 지정하여 SNS 주제에 대한 액세스를 추가로 제한하는 것이 좋습니다.

```
...
"Resource": "your-SNS-topic-ARN"
"Condition": {
  "StringEquals": {
    "aws:SourceArn": "arn:partition:dms:your-AWS-region:your-AWS-account-ID:es:your-dms-es-arn or *"
  }
}
```

3. 변경 사항 저장을 선택합니다.

AWS DMS 데이터 검증

주제

- [복제 작업 통계](#)
- [Amazon CloudWatch를 사용한 복제 태스크 통계](#)
- [작업 중 테이블 다시 검증](#)
- [JSON 편집기를 사용하여 검증 규칙 수정](#)
- [검증 전용 태스크](#)
- [문제 해결](#)
- [Redshift 검증 성능](#)
- [에 대한 향상된 데이터 검증 AWS Database Migration Service](#)
- [제한 사항](#)
- [Amazon S3 대상 데이터 검증](#)
- [AWS DMS 데이터 재동기화](#)

AWS DMS 는 데이터가 소스에서 대상으로 정확하게 마이그레이션되었는지 확인하기 위해 데이터 검증을 지원합니다. 활성화된 경우 테이블에 대해 전체 로드가 수행된 직후 검증이 시작됩니다. 검증은 CDC 지원 태스크에 대해 증분 변경 사항이 발생할 경우 이러한 변경 사항을 비교합니다.

데이터 검증 중에는 소스의 각 행을 대상의 해당 행과 AWS DMS 비교하고, 행에 동일한 데이터가 포함되어 있는지 확인하고, 불일치를 보고합니다. 이를 위해 AWS DMS 적절한 쿼리를 실행하여 데이터를 검색합니다. 이러한 쿼리는 소스 및 대상의 추가 리소스와 추가 네트워크 리소스를 사용합니다.

검증이 활성화된 CDC 전용 태스크의 경우, 새 데이터의 검증을 시작하기 전에 테이블에 있는 기존 데이터를 모두 검증합니다.

데이터 검증은가 소스 엔드포인트로 AWS DMS 지원하는 모든 곳에서 다음 소스 데이터베이스와 함께 작동합니다.

- Oracle
- PostgreSQL 호환 데이터베이스(PostgreSQL, Aurora PostgreSQL 또는 Aurora Serverless for PostgreSQL)
- MySQL 호환 데이터베이스(MySQL, MariaDB, Aurora MySQL 또는 Aurora Serverless for MySQL)
- Microsoft SQL Server

- IBM DB2 LUW

데이터 검증은가 대상 엔드포인트로 AWS DMS 지원하는 모든 곳에서 다음 대상 데이터베이스와 함께 작동합니다.

- Oracle
- PostgreSQL 호환 데이터베이스(PostgreSQL, Aurora PostgreSQL 또는 Aurora Serverless for PostgreSQL)
- MySQL 호환 데이터베이스(MySQL, MariaDB, Aurora MySQL 또는 Aurora Serverless for MySQL)
- Microsoft SQL Server
- IBM DB2 LUW
- Amazon Redshift
- Amazon S3. Amazon S3 대상 데이터 유효성 검사에 대한 자세한 내용은 [Amazon S3 대상 데이터 검증](#) 섹션을 참조하세요.

지원되는 엔드포인트에 대한 자세한 내용은 [AWS DMS 엔드포인트 작업](#) 섹션을 참조하세요.

데이터 검증에는 마이그레이션 자체에 소요되는 시간 외에도 추가 시간이 요구됩니다. 필요한 추가 시간은 마이그레이션되는 데이터의 양에 따라 달라집니다.

이러한 설정에 대한 자세한 내용은 [데이터 검증 작업 설정](#) 섹션을 참조하세요.

JSON 파일의 ValidationSettings 태스크 설정에 대한 예제는 [작업 설정 예제](#) 섹션을 참조하세요.

복제 작업 통계

데이터 검증이 활성화되면는 테이블 수준에서 다음 통계를 AWS DMS 제공합니다.

- ValidationState - 테이블의 검증 상태입니다. 이 파라미터의 값은 다음과 같습니다.
 - 활성화되지 않음 - 마이그레이션 작업의 테이블에 대해 검증이 활성화되지 않습니다.
 - 보류 중인 레코드 - 테이블의 일부 레코드가 검증 대기 중입니다.
 - 일치하지 않는 레코드 - 테이블의 일부 레코드가 원본과 대상 간에 일치하지 않습니다. 불일치가 발생하는 이유에는 여러 가지가 있으므로 자세한 내용은 대상 엔드포인트의 `awsdms_control.awsdms_validation_failures_v1` 테이블을 참조하세요.
 - 일시 중지된 레코드 - 테이블의 일부 레코드를 검증할 수 없습니다.
 - 프라이머리 키 없음 - 프라이머리 키가 없어 테이블을 검증할 수 없습니다.

- 테이블 오류 - 테이블이 오류 상태이고 일부 데이터가 마이그레이션되지 않아 테이블이 검증되지 않았습니다.
- 검증됨 - 테이블의 모든 행이 검증되었습니다. 테이블이 업데이트된 경우 상태가 Validated에서 변경할 수 있습니다.
- 오류 - 예상치 못한 오류로 인해 테이블을 검증할 수 없습니다.
- 검증 보류 중 - 테이블이 검증 대기 중입니다.
- 테이블 준비 중 - 마이그레이션 태스크에서 활성화된 테이블을 검증하기 위해 준비 중입니다.
- 재검증 보류 중 - 테이블이 업데이트된 후 테이블의 모든 행이 검증 보류 중입니다.
- ValidationPending - 대상에 마이그레이션되었지만 아직 검증되지 않은 레코드 수입니다.
- ValidationSuspended - 비교할 AWS DMS 수 없는 레코드 수입니다. 예를 들어 소스의 레코드가 지속적으로 업데이트되는 경우는 소스와 대상을 비교할 AWS DMS 수 없습니다.
- ValidationFailed - 데이터 검증 단계를 통과하지 않은 레코드 수입니다.

JSON 파일의 ValidationSettings 태스크 설정에 대한 예제는 [작업 설정 예제](#) 섹션을 참조하세요.

콘솔, AWS CLI 또는 AWS DMS API를 사용하여 데이터 검증 정보를 볼 수 있습니다.

- 콘솔에서 작업을 생성하거나 수정할 때 작업을 검증하도록 선택할 수 있습니다. 콘솔을 사용하여 데이터 검증 보고서를 보려면 작업 페이지에서 작업을 선택하고 세부 정보 섹션에서 테이블 통계 탭을 선택합니다.
- 데이터 검증을 시작하도록 작업을 생성하거나 수정하려는 경우 CLI를 사용하여 EnableValidation 파라미터를 true로 설정합니다. 다음 예제에서는 작업을 생성하고 데이터 검증을 활성화합니다.

```
create-replication-task
--replication-task-settings '{"ValidationSettings":{"EnableValidation":true}}'
--replication-instance-arn arn:aws:dms:us-east-1:5731014:
    rep:36KWVMB7Q
--source-endpoint-arn arn:aws:dms:us-east-1:5731014:
    endpoint:CSZAEFQURFYMM
--target-endpoint-arn arn:aws:dms:us-east-1:5731014:
    endpoint:CGPP7MF6WT4JQ
--migration-type full-load-and-cdc
--table-mappings '{"rules": [{"rule-type": "selection", "rule-id": "1",
    "rule-name": "1", "object-locator": {"schema-name": "data_types", "table-name":
"%"},
    "rule-action": "include"}]}'
```

`describe-table-statistics` 명령을 사용하여 JSON 형식의 데이터 검증 보고서를 받습니다. 다음 명령은 데이터 검증 보고서를 표시합니다.

```
aws dms describe-table-statistics --replication-task-arn arn:aws:dms:us-east-1:5731014:rep:36KWVMB7Q
```

이 보고서는 다음과 비슷합니다.

```
{
  "ReplicationTaskArn": "arn:aws:dms:us-west-2:5731014:task:VFPPTYKK2RYSI",
  "TableStatistics": [
    {
      "ValidationPendingRecords": 2,
      "Inserts": 25,
      "ValidationState": "Pending records",
      "ValidationSuspendedRecords": 0,
      "LastUpdateTime": 1510181065.349,
      "FullLoadErrorRows": 0,
      "FullLoadCondtnlChkFailedRows": 0,
      "Ddls": 0,
      "TableName": "t_binary",
      "ValidationFailedRecords": 0,
      "Updates": 0,
      "FullLoadRows": 10,
      "TableState": "Table completed",
      "SchemaName": "d_types_s_sqlserver",
      "Deletes": 0
    }
  ]
}
```

- AWS DMS API를 사용하여 `CreateReplicationTask` 작업을 사용하여 작업을 생성하고 `EnableValidation` 파라미터를 `true`로 설정하여 작업에 의해 마이그레이션된 데이터를 검증합니다. `DescribeTableStatistics` 작업을 사용하여 JSON 형식의 데이터 검증 보고서를 받습니다.

Amazon CloudWatch를 사용한 복제 태스크 통계

Amazon CloudWatch가 활성화되면는 다음과 같은 복제 작업 통계를 AWS DMS 제공합니다.

- `ValidationSucceededRecordCount` - 분당 AWS DMS 검증한 행 수입니다.

- ValidationAttemptedRecordCount - 검증을 시도한 분당 행 수입니다.
- ValidationFailedOverallCount - 검증이 실패한 행 수입니다.
- ValidationSuspendedOverallCount - 검증이 일시 중지된 행 수입니다.
- ValidationPendingOverallCount - 검증이 보류 중인 행 수입니다.
- ValidationBulkQuerySourceLatency - AWS DMS 는 변경 사항이 많을 경우, 전체 복제 또는 지속적 복제 동안 특정 시나리오를 중심으로 일괄적으로 데이터를 검증할 수 있습니다. 이 지표는 소스 엔드 포인트 대량의 데이터세트를 읽는 데 필요한 지연 시간을 나타냅니다.
- ValidationBulkQueryTargetLatency - AWS DMS 는 변경 사항이 많을 경우, 전체 복제 또는 지속적 복제 동안 특정 시나리오를 중심으로 일괄적으로 데이터를 검증할 수 있습니다. 이 지표는 대상 엔드 포인트의 대량의 데이터세트를 읽는 데 필요한 지연 시간을 나타냅니다.
- ValidationItemQuerySourceLatency - 지속적 복제 동안 데이터 검증은 지속적 변경 사항을 식별하고 이러한 변경 사항을 검증합니다. 이 지표는 소스에서 이런 변경을 읽는 데 필요한 지연 시간을 나타냅니다. 검증은 변경의 수를 근거로, 검증 동안 오류가 있을 시 필요한 것보다 더 많은 쿼리를 실행할 수 있습니다.
- ValidationItemQueryTargetLatency - 지속적 복제 동안 데이터 검증은 지속적 변경 사항을 식별하고 행 단위로 변경 사항을 검증할 수 있습니다. 이 지표는 대상에서 이런 변경을 읽는 데 필요한 지연 시간을 제공합니다. 검증은 변경의 수를 근거로, 검증 동안 오류가 있을 시 필요한 것보다 더 많은 쿼리를 실행할 수도 있습니다.

CloudWatch를 사용한 통계에서 데이터 검증 정보를 수집하려면 콘솔을 사용하여 태스크를 생성하거나 수정할 때 CloudWatch 로그 활성화를 선택합니다. 그런 다음, 데이터가 소스에서 대상으로 정확히 마이그레이션되었는지 확인하기 위한 데이터 검증 정보를 확인하려면 다음 작업을 수행합니다.

1. 데이터베이스 마이그레이션 태스크 페이지에서 태스크를 선택합니다.
2. CloudWatch 지표 탭을 선택합니다.
3. 드롭다운 메뉴에서 검증을 선택합니다.

작업 중 테이블 다시 검증

작업이 실행되는 동안 AWS DMS 에 데이터 검증 수행을 요청할 수 있습니다.

AWS Management Console

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

AWS Identity and Access Management (IAM) 사용자로 로그인한 경우 액세스할 수 있는 적절한 권한이 있는지 확인합니다 AWS DMS. 필요한 권한은 [섹션을 참조하세요](#) [를 사용하는 데 필요한 IAM 권한 AWS DMS](#).

2. 탐색 창에서 작업을 선택합니다.
3. 다시 검증할 테이블이 있는 실행 중인 작업을 선택합니다.
4. 테이블 통계 탭을 선택합니다.
5. 다시 검증할 테이블을 선택합니다(한 번에 최대 10개의 테이블을 선택할 수 있음). 작업이 더 이상 실행되고 있지 않으면 테이블을 다시 검증할 수 없습니다.
6. 다시 검증을 선택합니다.

JSON 편집기를 사용하여 검증 규칙 수정

AWS DMS 콘솔의 JSON 편집기를 사용하여 작업에 검증 규칙을 추가하려면 다음을 수행합니다.

1. 데이터베이스 마이그레이션 태스크를 선택합니다.
2. 마이그레이션 태스크 목록에서 태스크를 선택합니다.
3. 태스크가 실행 중인 경우 작업 드롭다운 메뉴에서 중지를 선택합니다.
4. 태스크가 중지된 후 태스크를 수정하려면 작업 드롭다운 메뉴에서 수정을 선택합니다.
5. 테이블 매핑 섹션에서 JSON 편집기를 선택하고 테이블 매핑에 검증 규칙을 추가합니다.

예를 들어, 다음과 같은 검증 규칙을 추가하여 소스에서 replace 함수를 실행할 수 있습니다. 이 경우 검증 규칙에서 null 바이트가 발견되면 해당 바이트는 공백으로 확인됩니다.

```
{
  "rule-type": "validation",
  "rule-id": "1",
  "rule-name": "1",
  "rule-target": "column",
  "object-locator": {
    "schema-name": "Test-Schema",
    "table-name": "Test-Table",
    "column-name": "Test-Column"
  },
  "rule-action": "override-validation-function",
  "source-function": "REPLACE(${column-name}, chr(0), chr(32))",
  "target-function": "${column-name}"
}
```

```
}
```

Note

`override-validation-function`은 열이 프라이머리 키의 일부인 경우 적용되지 않습니다.

검증 전용 태스크

마이그레이션이나 데이터 복제를 수행하지 않고도 데이터를 미리 보고 검증하는 검증 전용 태스크를 생성할 수 있습니다. 검증 전용 태스크를 생성하려면 `EnableValidation` 및 `ValidationOnly` 설정을 `true`로 설정합니다. `ValidationOnly`를 활성화하면 추가 요구 사항이 적용됩니다. 자세한 내용은 [데이터 검증 작업 설정](#) 단원을 참조하십시오.

전체 로드 전용 마이그레이션 유형의 경우, 실패가 많이 보고되면 검증 전용 태스크가 CDC 태스크보다 훨씬 더 빨리 완료됩니다. 그러나 소스 또는 대상 엔드포인트에 대한 변경 사항은 전체 로드 모드에서 실패로 보고되며, 이는 단점이 될 수 있습니다.

CDC 검증 전용 태스크는 평균 지연 시간을 기준으로 검증을 지연시키며, 실패를 보고하기 전에 여러 번 재시도합니다. 대부분의 데이터 비교 결과가 실패로 이어질 경우, CDC 모드의 검증 전용 태스크는 속도가 매우 느리며, 이는 단점이 될 수 있습니다.

검증 전용 작업은 복제 작업과 같은 방향으로 설정해야 하며, 특히 CDC의 경우에는 더욱 그렇습니다. CDC 검증 전용 태스크는 소스의 변경 로그를 기준으로, 어떤 행이 변경되었고 재검증해야 하는지 감지하기 때문입니다. 대상이 소스로 지정된 경우, DMS에서 대상으로 전송한 변경 사항만 인식하며 복제 오류를 포착하지 못할 수 있습니다.

전체 로드 검증 전용

AWS DMS 버전 3.4.6 이상부터 전체 로드 검증 전용 작업은 소스 및 대상 테이블의 모든 행을 한 번의 패스로 빠르게 비교하고 즉시 오류를 보고한 다음 종료합니다. 이 모드에서는 실패로 인해 검증이 일시 중지되지 않으며, 속도에 최적화되어 있습니다. 그러나 소스 또는 대상 엔드포인트에 대한 변경 사항은 실패로 보고됩니다.

Note

AWS DMS 버전 3.4.6 이상부터이 검증 동작은 검증이 활성화된 전체 로드 마이그레이션 작업에도 적용됩니다.

CDC 검증 전용

CDC 검증 전용 태스크는 새로 시작할 때 소스 테이블과 대상 테이블 사이에 있는 모든 기존 행을 검증합니다. 또한 CDC 검증 전용 태스크는 지속적으로 실행되며, 지속적 복제의 변경 사항을 재검증하고, 각 패스에서 보고된 실패 횟수를 제한하고, 불일치 행이 실패하기 전에 다시 시도합니다. 이는 거짓 긍정을 방지하도록 최적화되었습니다.

FailureMaxCount 또는 TableFailureMaxCount 임계값을 위반하면 테이블(또는 전체 태스크)에 대한 검증이 일시 중지됩니다. 이는 검증이 활성화된 CDC 또는 전체 로드+CDC 마이그레이션 태스크에도 적용됩니다. 그리고 검증이 활성화된 CDC 태스크는 평균 소스 및 대상 지연 시간을 기준으로 변경된 각 행에 대한 재검증을 지연시킵니다.

하지만 CDC 검증 전용 태스크는 데이터를 마이그레이션하지 않으며 지연 시간도 발생하지 않습니다. 기본적으로 ValidationQueryCdcDelaySeconds는 180으로 설정됩니다. 그리고 지연 시간이 긴 환경을 고려하여 용량을 늘리고 거짓 긍정을 방지할 수 있습니다.

검증 전용 사용 사례

마이그레이션 또는 복제 태스크의 데이터 검증 부분을 별도의 검증 전용 태스크로 분할하는 사용 사례에는 다음 경우가 포함되나, 이에 국한되지 않습니다.

- 검증이 수행되는 시기를 정확하게 제어 - 검증 쿼리는 소스 및 대상 엔드포인트 양쪽 모두에 추가적인 부하를 가중시킵니다. 따라서 먼저 한 태스크에서 데이터를 마이그레이션 또는 복제한 다음, 다른 태스크에서 결과를 검증하는 편이 유리할 수 있습니다.
- 복제 인스턴스의 부하 감소 - 데이터 검증을 분할하여 자체 인스턴스에서 실행하는 편이 유리할 수 있습니다.
- 특정 시점에 일치하지 않는 행의 수를 신속하게 파악 - 예를 들어, 유지 관리 기간 직전에 또는 도중에 프로덕션이 대상 엔드포인트로 전환될 경우 전체 로드 검증 전용 태스크를 생성하여 질문에 대한 답을 얻을 수 있습니다.
- CDC 구성 요소를 사용한 마이그레이션 태스크에서 검증 실패가 예상될 경우 - 예를 들어, Oracle varchar2를 PostgreSQL jsonb으로 마이그레이션할 경우 CDC 검증은 이러한 실패한 행을 계속

재시도하고 매번 보고되는 실패 횟수를 제한합니다. 하지만 전체 로드 검증 전용 태스크를 생성하면 더 빠른 답을 얻을 수 있습니다.

- 검증 실패 테이블을 읽는 데이터 복구 스크립트/유틸리티를 개발한 경우 - ([문제 해결](#)도 참조하세요). 전체 로드 검증 태스크는 데이터 복구 스크립트가 조치를 취해야 할 오류를 신속하게 보고합니다.

JSON 파일의 ValidationSettings 태스크 설정에 대한 예제는 [작업 설정 예제](#) 섹션을 참조하세요.

문제 해결

검증 중에는 대상 엔드포인트에서 새 테이블을 AWS DMS 생성합니

다awsdms_control.awsdms_validation_failures_v1. 어떤 레코드가

ValidationSuspended 또는 ValidationFailed 상태가 되면는 진단 정보들에 AWS DMS 기록합니

다awsdms_control.awsdms_validation_failures_v1. 이 테이블을 쿼리하여 검증 오류 문제를 해결할 수 있습니다.

대상에서 테이블이 생성되는 기본 스키마를 변경하는 방법에 대한 내용은 [제어 테이블 태스크 설정](#) 섹션을 참조하세요.

다음은 awsdms_control.awsdms_validation_failures_v1 테이블에 대한 설명입니다.

열 이름	데이터 유형	설명
TASK_NAME	VARCHAR(128) NOT NULL	AWS DMS 작업 식별자입니다.
TABLE_OWNER	VARCHAR(128) NOT NULL	테이블의 스키마(소유자).
TABLE_NAME	VARCHAR(128) NOT NULL	테이블 이름.
FAILURE_TIME	DATETIME(3) NOT NULL	실패가 발생한 시간.
KEY_TYPE	VARCHAR(128) NOT NULL	나중에 사용할 수 있도록 예약됩니다(값은 항상 'Row').
KEY	TEXT NOT NULL	이 키는 행 레코드 유형의 프라이머리 키입니다.

열 이름	데이터 유형	설명
FAILURE_TYPE	VARCHAR(128) NOT NULL	검증 오류의 심각도. RECORD_DIFF , MISSING_SOURCE , MISSING_TARGET 또는 TABLE_WARNING 입니다.
DETAILS	VARCHAR(8000) NOT NULL	지정된 키와 일치하지 않는 모든 소스/대상 열 값의 JSON 형식 문자열입니다.

다음은 awsdms_control.awsdms_validation_failures_v1 테이블을 쿼리하여 태스크에 대한 모든 실패를 표시하는 MySQL 대상에 대한 샘플 쿼리입니다. 스키마 이름과 쿼리 구문은 대상 엔진 버전마다 다릅니다. 작업 이름은 작업의 외부 리소스 ID여야 합니다. 작업의 외부 리소스 ID는 작업 ARN의 마지막 값입니다. 예를 들어, ARN 값이 arn:aws:dms:us-west-2:5599:task:VFPFKH4FJR3FTYKK2RYSI인 작업의 경우, 작업의 외부 리소스 ID는 VFPFKH4FJR3FTYKK2RYSI입니다.

```
select * from awsdms_validation_failures_v1 where TASK_NAME = 'VFPFKH4FJR3FTYKK2RYSI'
```

```
TASK_NAME      VFPFKH4FJR3FTYKK2RYSI
TABLE_OWNER    DB2PERF
TABLE_NAME     PERFTTEST
FAILURE_TIME    2020-06-11 21:58:44
KEY_TYPE       Row
KEY            {"key": ["3451491"]}
FAILURE_TYPE    RECORD_DIFF
DETAILS        [{"MYREAL": '+1.10106036e-01'}, {'MYREAL': '+1.10106044e-01'}],]
```

DETAILS 필드를 보고 일치하지 않는 열을 확인할 수 있습니다. 실패한 레코드의 프라이머리 키가 있으므로, 소스 및 대상 엔드포인트를 쿼리하여 레코드에서 일치하지 않는 부분을 찾아낼 수 있습니다.

awsdms_validation_failures_v2 제어 테이블

검증 중에 AWS DMS 버전 3.6.1 이상에서 DMS는 PostgreSQL 대상 엔드포인트에 새 테이블을 생성합니다. awsdms_validation_failures_v2. 이 테이블은 데이터 검증이 활성화된 모든 DMS 작업에 대한 실패로 구성됩니다. awsdms_validation_failures_v2 테이블이 생성되면 검증 및 재 동기화가 활성화된 모든 작업에 오류가 발생할 수 있으므로 테이블을 삭제하거나 자르면 안 됩니다.

awsdms_validation_failures_v2 테이블에는 자동 증분 프라이머리 키 기능이 있습니다. 이 테이블은 데이터 재동기화 기능을 지원하는 새 열로 구성됩니다. 스크립트는 다음과 같습니다.

RESYNC_RESULT

값: SUCCESS 또는 FAILURE.

RESYNC_TIME

밀리초 정밀도의 타임스탬프입니다. 이 실패에 대해 데이터 재동기화를 시도하지 않는 경우 기본값은 NULL입니다.

RESYNC_ACTION

값: UPSERT 또는 DELETE.

RESYNC_ID

자동 증분이 활성화된 프라이머리 키 열입니다.

awsdms_validation_failures_v2 테이블에서 인덱스가 TASK_NAME, TABLE_OWNER, TABLE_NAME, FAILURE_TYPE, 및 FAILURE_TIME 열에 추가되어 대상 데이터베이스의 지정된 테이블에 대한 실패를 효율적으로 읽습니다. 다음은 awsdms_validation_failures_v2 테이블을 생성하기 위한 생성 문 예제입니다.

```
CREATE TABLE public.awsdms_validation_failures_v2 (  
    "RESYNC_ID" int8 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1 MINVALUE 1  
    MAXVALUE 9223372036854775807 START 1 CACHE 1 NO CYCLE) NOT NULL,  
    "TASK_NAME" varchar(128) NOT NULL,  
    "TABLE_OWNER" varchar(128) NOT NULL,  
    "TABLE_NAME" varchar(128) NOT NULL,  
    "FAILURE_TIME" timestamp NOT NULL,  
    "KEY_TYPE" varchar(128) NOT NULL,  
    "KEY" varchar(7800) NOT NULL,  
    "FAILURE_TYPE" varchar(128) NOT NULL,  
    "DETAILS" varchar(7000) NOT NULL,  
    "RESYNC_RESULT" varchar(128) NULL,  
    "RESYNC_TIME" timestamp NULL,  
    "RESYNC_ACTION" varchar(128) NULL,  
    CONSTRAINT awsdms_validation_failures_v2_pkey PRIMARY KEY ("RESYNC_ID")  
);
```

Redshift 검증 성능

Amazon Redshift는 열 기반 스토리지, MPP, 데이터 압축 및 기타 요소 등 여러 가지 면에서 관계형 데이터베이스와 다릅니다. 이러한 차이로 인해 Redshift는 관계형 데이터베이스와는 다른 성능 프로파일을 제공합니다.

전체 로드 복제 단계 중 검증에서는 PartitionSize 설정에 따라 데이터 크기가 제어되는 범위 쿼리를 사용합니다. 이러한 범위 기반 쿼리에서는 소스 테이블의 모든 레코드가 선택됩니다.

지속적인 복제의 경우 쿼리는 범위 기반과 개별 레코드 가져오기 간에 전환됩니다. 쿼리 유형은 다음과 같은 여러 요인에 따라 동적으로 결정됩니다.

- 쿼리 볼륨
- 소스 테이블의 DML 쿼리 유형
- 작업 지연 시간
- 총 레코드 수
- PartitionSize 등의 검증 설정

검증 쿼리로 인해 Amazon Redshift 클러스터에 추가 로드가 발생할 수 있습니다. 위의 요인은 사용 사례에 따라 다르므로 검증 쿼리 성능을 검토하고 그에 따라 클러스터와 테이블을 조정해야 합니다. 성능 문제를 완화하기 위한 몇 가지 옵션은 다음과 같습니다.

- PartitionSize 및 ThreadCount 설정을 줄이면 전체 로드 검증 중에 워크로드를 줄이는 데 도움이 됩니다. 이렇게 하면 데이터 검증 속도가 느려진다는 점에 유의하세요.
- Redshift는 기본 키를 적용하지 않지만 AWS DMS 는 기본 키를 사용하여 데이터 검증을 위해 대상의 레코드를 고유하게 식별합니다. 전체 로드 검증 쿼리가 더 빠르게 실행되기 위해 가능하면 프라이머리 키가 정렬 키를 미러링하도록 설정합니다.

에 대한 향상된 데이터 검증 AWS Database Migration Service

AWS Database Migration Service 는 데이터베이스 마이그레이션을 위한 데이터 검증 성능을 개선하여 고객이 처리 시간이 훨씬 빨라진 대규모 데이터 세트를 검증할 수 있도록 합니다. 이제 CDC 마이그레이션 작업을 통한 전체 로드 및 전체 로드 모두에 대해 복제 엔진 버전 3.5.4에서 향상된 데이터 검증을 사용할 수 있습니다. 현재이 개선 사항은 Oracle에서 PostgreSQL로, SQL Server에서 PostgreSQL로, Oracle에서 Oracle로, SQL Server에서 SQL Server로의 마이그레이션 경로를 지원하며 향후 릴리스에 추가 마이그레이션 경로가 계획되어 있습니다.

사전 조건

- Oracle: Oracle 엔드포인트에 액세스하는 SYS.DBMS_CRYPTO 사용자 계정에 대한 EXECUTE 권한을 부여합니다.

```
GRANT EXECUTE ON SYS.DBMS_CRYPTO TO dms_endpoint_user;
```

- PostgreSQL 데이터베이스에 pgcrypto 확장 프로그램을 생성합니다.

자체 관리형 PostgreSQL 인스턴스의 경우 contrib 모듈 라이브러리를 설치하고 확장을 생성해야 합니다.

- contrib 모듈 라이브러리를 설치합니다. 예를 들어 Amazon Linux 및 PostgreSQL 15가 있는 Amazon EC2 인스턴스의 경우:

```
sudo dnf install postgresql15-contrib
```

- pgcrypto 확장 기능을 생성합니다.

```
CREATE EXTENSION IF NOT EXISTS pgcrypto;
```

- Amazon RDS for PostgreSQL 인스턴스의 경우 AWS DMS 엔드포인트에 대한 SSL 모드를 구성합니다.
 - 기본적으로 Amazon RDS는 SSL 연결을 강제로 실행합니다. Amazon RDS for PostgreSQL 인스턴스에 대한 AWS DMS 엔드포인트를 생성할 때 "SSL 모드" 옵션 = "필수"를 사용합니다.
 - "SSL 모드" 옵션 = "없음"을 사용하려면 RDS rds.force_ssl 파라미터 그룹에서 파라미터를 0으로 설정합니다.
- PostgreSQL 12 및 13의 경우 BIT_XOR 집계를 생성합니다.

```
CREATE OR REPLACE AGGREGATE BIT_XOR(IN v bit) (SFUNC = bitxor, STYPE = bit);
```

향상된 데이터 검증 제한 사항

이 향상된 데이터 검증 기능에는 다음과 같은 제한 사항이 있습니다.

- 데이터베이스 엔드포인트 요구 사항: 이 개선 사항은 다음 기준을 충족하는 데이터베이스 엔드포인트에서만 활성화됩니다.
 - AWS Secrets Manager 를 사용하여 자격 증명을 저장합니다.

- Microsoft SQL Server의 경우 Kerberos 인증도 지원됩니다.
- 데이터베이스 버전 지원:
 - PostgreSQL 12 이상
 - Oracle 12.1 이상
 - 2019년 이전의 Microsoft SQL Server 버전에서는 NCHAR 및 NVARCHAR 데이터 형식의 검증이 지원되지 않습니다.

제한 사항

- 데이터를 검증하려면 테이블에 프라이머리 키나 고유한 인덱스가 있어야 합니다.
 - 프라이머리 키 열은 CLOB, BLOB, BINARY 또는 BYTE 형식이 아니어야 합니다.
 - VARCHAR 또는 CHAR 형식의 프라이머리 키 열은 길이가 1024보다 작아야 합니다. 데이터 유형에 길이를 지정해야 합니다. 무제한 데이터 유형을 데이터 검증의 프라이머리 키로 사용할 수 없습니다.
 - NOVALIDATE 절을 사용하여 생성된 Oracle 키는 프라이머리 키 또는 고유 인덱스로 간주되지 않습니다.
 - 프라이머리 키가 없고 고유 키만 있는 Oracle 테이블의 경우에는 고유 제약 조건이 있는 열에도 NOT NULL 제약 조건이 있어야 합니다.
- NULL PK/UK 값의 검증은 지원되지 않습니다.
- 대상 PostgreSQL 인스턴스의 프라이머리 키 열의 콜레이션이 "C"로 설정되어 있지 않으면 프라이머리 키의 정렬 순서가 Oracle의 정렬 순서와 달라집니다. PostgreSQL과 Oracle의 정렬 순서가 다르면 데이터 검증을 통해 레코드를 검증할 수 없습니다.
- 데이터 검증을 수행하면 원본 및 대상 데이터베이스에 대해 추가 쿼리가 생성됩니다. 두 데이터베이스에 이 추가 로드를 처리할 수 있을 만큼 충분한 리소스가 확보되었는지 확인해야 합니다. Redshift 대상의 경우 특히 그렇습니다. 자세한 내용은 [Redshift 검증 성능](#) 섹션을 참조하세요.
- 여러 데이터베이스를 하나로 통합할 때는 데이터 검증이 지원되지 않습니다.
- 소스 또는 대상 Oracle 엔드포인트의 경우를 AWS DMS 사용합니다 DBMS_CRYPT0. Oracle 엔드포인트에서 데이터 검증을 사용하는 경우, Oracle 엔드포인트에 액세스하는 데 사용되는 사용자 계정에 dbms_crypto에 대한 실행 권한을 부여해야 합니다. 다음 명령문을 실행하여 이 권한을 호출할 수 있습니다

```
grant execute on sys.dbms_crypto to dms_endpoint_user;
```

- 검증 AWS DMS 중에 대상 데이터베이스가 외부에서 수정되면 불일치가 정확하게 보고되지 않을 수 있습니다. 이 결과는 애플리케이션 중 하나가 동일한 테이블에서 검증을 수행하는 동안 대상 테이블에 데이터를 쓰 AWS DMS 는 경우에 발생할 수 있습니다.
- 검증 중에 하나 이상의 행이 지속적으로 수정되는 경우는 해당 행을 검증할 AWS DMS 수 없습니다.
- 가 10,000개 이상의 실패하거나 일시 중지된 레코드를 AWS DMS 감지하면 검증이 중지됩니다. 계속 진행하기 전에 데이터에 잠재된 문제를 해결하세요.
- AWS DMS 는 뷰의 데이터 검증을 지원하지 않습니다.
- AWS DMS 는 문자 대체 작업 설정을 사용할 때 데이터 검증을 지원하지 않습니다.
- AWS DMS 는 Oracle LONG 유형의 검증을 지원하지 않습니다.
- AWS DMS 는 이기종 마이그레이션 중에 Oracle Spatial 유형의 검증을 지원하지 않습니다.
- 데이터 검증은 테이블 매핑에 데이터 마스킹 변환이 있는 테이블의 열을 무시합니다.
- 데이터 검증은 PK/UK 열에 대한 데이터 마스킹 변환 규칙이 있는 경우 전체 테이블을 건너뜁니다. 검증 상태는 해당 테이블에 대한 프라이머리 키 없음으로 표시됩니다.
- Amazon Aurora PostgreSQL Limitless에서는 데이터 검증이 작동하지 않습니다. Limitless Database의 테이블을 검증하려고 하면 검증 상태에 이러한 테이블에 대한 "프라이머리 키 없음"이 표시됩니다.

S3 대상 검증 사용에 대한 제한 사항은 [S3 대상 검증 사용에 대한 제한 사항](#) 섹션을 참조하세요.

Amazon S3 대상 데이터 검증

AWS DMS 는 Amazon S3 대상에서 복제된 데이터의 검증을 지원합니다. 는 복제된 데이터를 Amazon S3에 플랫 파일로 AWS DMS 저장하기 때문에 [Amazon Athena](#) CREATE TABLE AS SELECT(CTAS) 쿼리를 사용하여 데이터를 검증합니다.

Amazon S3에 저장된 데이터에 대한 쿼리는 컴퓨팅 집약적입니다. 따라서 하루에 한 번만 변경 데이터 캡처(CDC) 중에 자정(00:00) UTC에 Amazon S3 데이터에 대한 검증을 AWS DMS 실행합니다. 가 AWS DMS 실행하는 각 일일 검증을 간격 검증이라고 합니다. 간격 검증 중에는 이전 24시간 동안 대상 Amazon S3 버킷으로 마이그레이션된 모든 변경 레코드를 AWS DMS 검증합니다. 간격 검증의 제한 사항에 대한 자세한 내용은 [S3 대상 검증 사용에 대한 제한 사항](#) 섹션을 참조하세요.

Amazon S3 대상 검증은 Amazon Athena를 사용하므로 추가 비용이 적용됩니다. 자세한 내용은 [Amazon Athena 요금](#)을 참조하세요.

Note

S3 대상 검증에는 AWS DMS 버전 3.5.0 이상이 필요합니다.

주제

- [S3 대상 검증 사전 조건](#)
- [S3 대상 검증을 사용하기 위한 권한](#)
- [S3 대상 검증 사용에 대한 제한 사항](#)
- [S3 대상 검증과 함께 검증 전용 태스크 사용](#)

S3 대상 검증 사전 조건

S3 대상 검증을 사용하기 전에 다음과 같은 설정 및 권한을 확인합니다.

- 엔드포인트의 [S3Settings](#)에 대한 DataFormat 값을 parquet로 설정합니다. 자세한 내용은 [S3의 Parquet 설정](#) 단원을 참조하십시오.
- 마이그레이션 태스크를 생성하는 데 사용된 사용자 계정에 할당된 역할은 올바른 권한 집합을 보유하고 있어야 합니다. 아래의 [권한](#) 섹션을 참조하세요.

지속적 복제(CDC)를 사용하는 태스크의 경우 다음 설정을 확인하세요.

- CDC 데이터에 전체 레코드를 남기려면 보충 로깅을 활성화합니다. 보충 로깅 활성화에 대한 내용은 이 가이드의 [문제 해결 및 진단 지원](#) 섹션에서 [보충 로깅을 Oracle 소스 엔드포인트에 자동으로 추가합니다](#) 부분을 참조하세요.
- 대상 엔드포인트의 TimestampColumnName 파라미터를 설정합니다. 타임스탬프 열 이름에는 제한 사항이 없습니다. 자세한 내용은 [S3Settings](#) 섹션을 참조하세요.
- 대상의 날짜 기반 폴더 분할을 설정합니다. 자세한 내용은 [날짜 기반 폴더 파티셔닝 사용](#) 단원을 참조하십시오.

S3 대상 검증을 사용하기 위한 권한

S3 대상 검증을 위한 액세스 권한을 설정하려면 마이그레이션 작업 생성에 사용된 사용자 계정에 할당된 역할은 다음과 같은 권한 집합을 보유하고 있어야 합니다. 샘플 값을 내가 생각한 값으로 바꿉니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "athena:StartQueryExecution",
        "athena:GetQueryExecution",
        "athena:CreateWorkGroup"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "glue:CreateDatabase",
        "glue>DeleteDatabase",
        "glue:GetDatabase",
        "glue:GetTables",
        "glue:CreateTable",
        "glue>DeleteTable",
        "glue:GetTable"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation",
        "s3:GetObject",
        "s3:ListBucketMultipartUploads",
        "s3:AbortMultipartUpload",
        "s3:ListMultipartUploadParts"
      ],
      "Resource": "*"
    }
  ]
}
```

S3 대상 검증 사용에 대한 제한 사항

S3 대상 검증을 사용할 때 적용되는 아래의 추가적인 제한 사항을 확인합니다. 모든 검증에 적용되는 제한 사항은 [제한 사항](#) 섹션을 참조하세요.

- DatePartitionSequence 값에는 '일' 구성 요소가 필요합니다. S3 대상 검증은 YYYYMM 형식을 지원하지 않습니다.
- CDC 중에 간격 검증을 실행하면 awsdms_validation_failures_v1 표에 거짓 검증 오류가 표시될 수 있습니다. 이러한 오류는가 간격 검증 중에 도착한 변경 사항을 다음 날의 파티션 폴더로 AWS DMS 마이그레이션하기 때문에 발생합니다. 일반적으로 이러한 변경 사항은 당일의 파티션 폴더에 작성됩니다. 이러한 거짓 오류는 동적 소스 데이터베이스에서 정적 대상(예: Amazon S3)으로 복제된 항목을 검증하는 과정에서 발생하는 제한 사항입니다. 이러한 거짓 오류를 조사하려면 이런 오류가 일반적으로 나타나는 검증 기간이 종료되는 시점에(00:00 UTC) 레코드를 확인합니다.

거짓 오류의 수를 최소화하려면 태스크에 대한 CDCLatencySource가 낮아야 합니다. 모니터링 지연 시간에 대한 내용은 [복제 작업 지표](#) 섹션을 참조하세요.

- failed 또는 stopped 상태의 태스크는 전날의 변경 사항을 검증하지 않습니다. 예상치 못한 실패로 인한 검증 오류를 최소화하려면 동일한 테이블 매핑, 소스 및 대상 엔드포인트가 포함된 별도의 검증 전용 태스크를 생성합니다. 검증 전용 태스크에 대한 자세한 내용은 [S3 대상 검증과 함께 검증 전용 태스크 사용](#) 섹션을 참조하세요.
- 테이블 통계의 검증 상태 열에는 가장 최근의 간격 검증 상태가 반영됩니다. 따라서 불일치 항목이 있는 테이블은 다음 날 간격 검증이 끝난 후에 검증된 것으로 표시될 수 있습니다. 대상 Amazon S3 버킷의 s3_validation_failures folder를 검사하여 하루 이상 전에 발생한 불일치 항목이 있는지 확인합니다.
- S3 검증은 Amazon Athena의 버킷 테이블 기능을 사용합니다. 이렇게 하면 S3 검증이 대상 테이블 데이터의 버킷 복사본을 만들 수 있습니다. 즉, 테이블 데이터의 사본이 DMS 검증의 내부 파티셔닝과 일치하는 하위 집합으로 분할됩니다. Athena 버킷 테이블의 한도는 버킷 100,000개입니다. 이 제한을 초과하는 테이블에 대한 S3 검증 시도는 검증 실패로 나타납니다. S3 Validation에서 생성하려고 시도하는 버킷 수는 다음과 같습니다.

`(#records in the table) / (validation partition size setting)`

이 제한을 해결하려면 S3 Validation에서 생성한 버킷 수가 100,000개 미만이 되도록 검증 파티션 크기 설정을 늘리세요. 버킷화에 대한 자세한 내용은 Amazon Athena 사용 설명서의 [Athena에서 파티셔닝 및 버킷화](#)를 참조하세요.

- 테이블 이름에는 밑줄을 제외한 특수 문자가 포함되어서는 안 됩니다.

S3 검증은 테이블 이름에 특수 문자(밀줄 제외)를 지원하지 않는 Amazon Athena를 사용합니다. 자세한 내용은 Amazon Athena 사용 설명서의 [CREATE TABLE](#) 주제를 참조하세요.

- AWS Lake Formation에서 관리하는 Amazon S3 대상과 함께 AWS DMS 데이터 검증 기능을 사용하면 검증 프로세스가 실패합니다. 이로 인해 데이터 일관성 문제가 발생할 수 있습니다.

S3 대상 검증과 함께 검증 전용 태스크 사용

검증 전용 태스크는 마이그레이션을 실행하지 않고 마이그레이션할 데이터에 대한 검증을 실행합니다.

마이그레이션 작업이 중지되더라도 검증 전용 작업은 계속 실행되므로 00:00 UTC 간격 검증 기간을 놓치지 AWS DMS 않습니다.

Amazon S3 대상 엔드포인트에서 검증 전용 태스크를 사용할 경우 다음과 같은 제한 사항이 적용됩니다.

- 검증 전용 설정이 활성화된 전체 로드 작업에 대해 Amazon S3 검증이 지원되지만, 다른 엔드포인트의 전체 로드 및 검증 전용 작업과는 다르게 작동합니다. S3를 대상으로 하는 경우, 이 유형의 태스크는 S3 대상의 전체 로드 데이터에 대해서만 검증을 수행하며 CDC 마이그레이션의 일부로 마이그레이션된 데이터에 대해서는 검증을 수행하지 않습니다. 이 기능은 전체 로드 전용 태스크로 생성된 데이터를 검증할 때만 사용하세요. 이 모드를 사용하여 활성 CDC 태스크가 실행 중인 대상의 데이터를 검증하면 효과적인 검증이 이루어지지 않습니다.
- 검증 전용 태스크는 마지막 간격 검증 기간(00:00 UTC) 이후의 변경 사항만 검증합니다. 검증 전용 태스크는 전날의 전체 로드 데이터 또는 CDC 데이터를 검증하지 않습니다.

AWS DMS 데이터 재동기화

AWS Database Migration Service (AWS DMS) 데이터 재동기화는 소스 데이터베이스와 대상 데이터베이스 간의 데이터 검증을 통해 식별된 데이터 불일치를 자동으로 수정합니다. 이 기능은 기존 DMS 마이그레이션 작업의 일부로 작동하여 작업 구성, 연결 설정, 테이블 매핑 및 변환에 따라 적절한 업데이트가 이루어지도록 합니다.

데이터 재동기화 기능은 대상 데이터베이스의 제어 테이블에서 검증 실패를 읽고 적절한 수정 작업을 실행하여 작동합니다. 불일치가 감지되면 현재 데이터는 실패 레코드에 저장된 프라이머리 키를 사용하여 소스에서 검색되고 구성된 변환을 준수하면서 대상에 적용됩니다. 자세한 내용은 [the section called “awsdms_validation_failures_v2 제어 테이블”](#) 단원을 참조하십시오.

동작은 마이그레이션 유형에 따라 다릅니다. full-load-only 작업의 경우 데이터 재동기화는 초기 로드 및 검증이 완료된 후 한 번 실행됩니다. 변경 데이터 캡처(CDC)가 있는 작업의 경우 데이터 재동기화는 구성된 일정에 따라 작동하며 수정 사항이 적용되는 동안 복제 및 검증을 일시적으로 일시 중지합니다.

CDC 재동기화 작업 중:

- 복제 및 검증이 일시적으로 일시 중지됩니다.
- 데이터 재동기화는 기존 검증 실패를 처리합니다.
- 일반 복제 및 검증이 재개됩니다.
- 구성된 일정에 따라 프로세스가 반복됩니다.

데이터 재동기화는 각 수정 작업의 상태를 자동으로 추적하고 테이블 통계를 통해 자세한 지표를 제공합니다.

사전 조건:

데이터 재동기화 기능에는 다음과 같은 사전 조건이 필요합니다.

- AWS DMS 엔진 버전 3.6.1 이상이 있어야 합니다.
- 복제가 진행 중인 작업에 대한 일정 및 타이밍 기간 설정을 구성해야 합니다. 전체 로드 전용 작업에는 이러한 설정이 필요하지 않습니다.

제한 사항

데이터 재동기화 기능에는 다음과 같은 제한이 있습니다.

- 데이터 재동기화는 Oracle 및 SQL Server만 소스 데이터베이스로 지원합니다.
- 데이터 재동기화는 PostgreSQL 및 Amazon Aurora PostgreSQL 호환 엔진을 대상 데이터베이스로 지원합니다.
- 소스 및 대상 데이터베이스의 모든 테이블에는 프라이머리 키가 있어야 합니다. 검증은 프라이머리 키 또는 고유 키가 없는 테이블을 지원하지 않습니다. 유효한 기본 또는 고유 키가 없는 모든 테이블은 검증이 일시 중지되고 검증 실패는 보고되지 않습니다.
- Full-load-only 작업을 실행할 때는 데이터 검증을 활성화해야 합니다.
- 검증 전용 작업은 데이터를 복제하지 않으므로 데이터 재동기화를 활성화할 수 없습니다. 검증 전용 taskID만 제공하여 상위 복제 작업에서 재동기화를 활성화할 수 있습니다. 자세한 내용은 [검증 전용 작업](#) 섹션을 참조하세요.

- 검증 전용 작업에 작업 설정에 ControlSchema 파라미터 설정이 구성되어 있는 경우 데이터 재동기화가 올바른 검증 실패를 찾으려면 복제 작업에도 동일한 파라미터 구성이 있어야 합니다.
- CDC 작업에 대한 일정 및 타이밍 기간 설정을 구성해야 합니다.
- 재동기화 기간 동안 데이터 재동기화는 DMS의 복제 지연 시간에 영향을 미칠 수 있습니다.

데이터 재동기화 AWS DMS 종의 검증 문제 해결에 대한 자세한 내용은 [AWS DMS 데이터 검증](#) 아래의 [문제 해결](#) 섹션을 참조하세요.

예약 및 타이밍

CDC 작업의 경우 데이터 재동기화가 작동하는 시기와 기간을 구성해야 합니다. 이렇게 하면 정상적인 복제 작업에 미치는 영향을 방지할 수 있습니다. 다음을 지정합니다.

- cron 형식을 사용하여 재동기화 작업이 발생할 수 있는 시기를 정의하는 일정입니다.
- 재동기화 작업이 최대 사용 기간으로 연장되지 않도록 하기 위한 최대 기간입니다.

사용량이 적은 시간 또는 소스 데이터베이스에 변경이 거의 또는 전혀 없는 기간 동안 재동기화 작업을 예약하는 것이 좋습니다.

Note

데이터 재동기화와 일반 복제를 동시에 실행할 수 없으므로 예약된 시간에는 대상 적용 스트림이 비어 있을 때까지 기다리는 것이 포함됩니다.

사용 사례

데이터 재동기화 기능을 사용하면 사용자가 소스 시스템과 대상 시스템 간의 데이터 불일치를 조정할 수 있습니다. 일치하지 않는 레코드를 식별하고 동기화하여 분산 환경 간에 데이터 일관성을 유지합니다. 다음 사용 사례는 데이터 재동기화 기능이 데이터 일관성 문제를 해결하는 일반적인 시나리오를 보여줍니다.

시나리오 1: 전체 로드 작업 - 동일한 DMS 작업을 사용하여 재동기화 실행

기존 DMS 전체 로드 마이그레이션 작업에서 다음을 수행할 수 있습니다.

- 검증 활성화: Validation with data migration = true.
- 재동기화 활성화: Data resync = true

시나리오 2: 전체 로드 및 CDC, CDC 전용 작업 - 동일한 DMS 작업을 사용하여 재동기화 실행

기존 DMS CDC 마이그레이션 작업에서 다음을 수행할 수 있습니다.

- 검증 활성화: `Validation with data migration = true`.
- 재동기화 활성화: `Data resync = true`
- 재동기화 일정 지정: `"ResyncSchedule": "0 0,2,4,6 * * *"`.
- 재동기화 시간 지정: `MaxResyncTime": 60`

시나리오 3: 검증 전용 작업과 함께 복제 및 재동기화를 위한 전체 로드 및 CDC 또는 CDC 전용 작업

재동기화를 사용할 때 다른 DMS 작업에서 검증 전용 작업을 수행하려면 다음을 수행할 수 있습니다.

- 검증 전용 DMS CDC 작업을 생성합니다.

Note

데이터 재동기화 중에 이 작업의 ID를 기록하고 지정해야 합니다.

- 기본 CDC 작업에서 `Data validation = false` 검증을 비활성화합니다.
- 재동기화 활성화: `Data resync = true`
- 재동기화 일정 지정: `"ResyncSchedule": "0 0,2,4,6 * * *"`.
- 재동기화 시간 지정: `MaxResyncTime": 60`.
- 검증 전용 DMS CDC 작업의 ID를 지정합니다. 검증 전용 작업 ID는 ARN 끝에 추가됩니다. 예제 ARN: `arn:aws:dms:us-west-2:123456789012:task:6DG4CLGJ5JSJR67CFD7UDXFY7KV6CYGRICL6KWI` 및 예제 검증 전용 작업 ID: `6DG4CLGJ5JSJR67CFD7UDXFY7KV6CYGRICL6KWI`.

모범 사례

의 데이터 재동기 기능을 활용하여 복제 작업의 내구성 AWS Database Migration Service 을 개선하고 일관성을 확보할 수 있습니다. 데이터 재동기화 기능을 사용하는 모범 사례는 다음과 같습니다.

- 데이터 재동기화의 일부로 불일치가 있는 레코드는 소스에서 가져와 대상 데이터베이스에 적용하여 수정됩니다. 재동기화 기간 중에 소스 데이터베이스가 업데이트되면 재동기화는 최신 레코드 값을 읽어 대상에 적용합니다. 이로 인해 CDC가 이벤트를 적용하지 못하고 대상 데이터베이스에 일시적

인 불일치가 발생할 수 있습니다. 이를 방지하려면 업무 외 시간 또는 소스 데이터베이스의 변경 사항이 0이거나 최소인 기간에 재동기화 기간을 예약해야 합니다.

- 최소 소스 데이터베이스 활동 기간 동안 허용 가능한 목표 지연 시간 임계값 내에서 재동기화 기간을 설정합니다. 재동기화 간격이 작으면 처리되지 않은 검증 불일치가 누적될 수 있는 반면, 확인 실패가 많을 경우 기간이 클수록 복제 지연 시간이 늘어날 수 있습니다. 검증 실패 및 재동기화 속도를 모니터링하여 소스 비활성 기간 동안 최적의 재동기화 기간을 결정합니다. 재동기화 기간을 설정하는 몇 가지 예는 다음과 같습니다.
- 여러 짧은 기간 구성:

```
"ResyncSchedule": "0 0,2,4,6 * * *",  
"MaxResyncTime": 60
```

- 단일 일일 기간 구성:

```
"ResyncSchedule": "0 0 * * *",  
"MaxResyncTime": 360
```

- 재동기화 기간 동안 DMS의 복제 지연 시간을 모니터링하고 그에 따라 일정을 조정하여 대규모 급증을 완화합니다.
- 테이블 통계를 통해 또는 대상 데이터베이스에서 `awsdms_validation_failures_v2` 테이블을 쿼리하여 재동기화 결과를 검토할 수 있습니다. 자세한 내용은 [Amazon CloudWatch를 사용하여 복제 지표 모니터링](#)을 참조하세요.
- 작업이 지속적 복제 단계에 있는 경우 재동기화 기간 동안 개별 테이블에 대한 재로드를 시작하지 마세요.
- CDC 복제 작업 모범 사례:
 - 데이터베이스의 모든 테이블이 로드 프로세스를 완료합니다.
 - 불일치는 진행 중인 검증 프로세스에서 식별됩니다.
 - 재동기화 예약 기간에 따라 복제 작업이 잠시 일시 중지됩니다.
 - 데이터 재동기화는 검증 프로세스 중에 식별된 문제를 해결합니다.
 - 복제 프로세스가 재개되고 일정에 따라 반복됩니다.

데이터 재동기화 구성 및 예제

데이터 재동기화 설정 구성:

DMS에서 복제 작업에 대한 재동기화를 구성할 수 있습니다. 다음은 작업에서 데이터 재동기화 설정 구성의 예입니다.

```
"ResyncSettings": {
  "EnableResync": true,
  "ResyncSchedule": "0 0,2,4,6 * * *", // Run at 12AM, 2AM, 4AM, and 6AM daily
  "MaxResyncTime": 60, // Run for maximum of 60 minutes, or 1
  hour
  "ValidationTaskId": "TASK-ID-IF-NEEDED" //Optional, used only if validation is
  performed as a separate Validation only task
}
```

일반적인 재동기화 예약 패턴의 예:

- 0 0 * * *: 매일 자정에 한 번 실행합니다.
- 0 0,12 * * *: 매일 자정과 정오에 두 번 실행합니다.
- 0 0,2,4,6, * * *: 자정부터 오전 6시 사이에 2시간마다 실행합니다.
- 0 1 * * 1: 매주 월요일 오전 1시에 실행합니다.

Note

0부터 6까지 매일 숫자를 지정해야 합니다. 자세한 내용은 [cron 표현식](#) 섹션을 참조하세요.

재동기화 작업 모니터링:

테이블 통계를 통해 재동기화 작업을 모니터링할 수 있습니다. 다음은 예시 출력입니다.

```
{
  "TableStatistics": {
    ...
    "ValidationFailedRecords": 1000,
    ...
    "ResyncRowsAttempted": 1000,
```

```

    "ResyncRowsSucceeded": 995,
    "ResyncRowsFailed": 5,
    "ResyncProgress": 99.5, // ratio of ResyncRowsSucceeded/
ValidationFailedRecords
    "ResyncState": "Last resync at: 2024-03-14T06:00:00Z"
  }
}

```

에서 데이터 재동기화 기능을 구성하려면 다양한 재동기화 파라미터와 해당 구성 설정을 검토할 AWS DMS 수 있습니다. 자세한 내용은 [the section called “데이터 재동기 설정”](#) 단원을 참조하십시오. 데이터 재동기 로깅 설정에 대한 자세한 내용은 [the section called “작업 설정 로깅”](#) 섹션을 참조하세요.

검증 및 문제 해결

검증:

데이터 평가 기능이 활성화되면는 다음 구조를 사용하여 대상 데이터베이스에 검증 실패 테이블을 AWS DMS 생성합니다.

```

CREATE TABLE awsdms_validation_failures_v2 (
  "RESYNC_ID" bigint NOT NULL,
  "TASK_NAME" varchar(128) NOT NULL,
  "TABLE_OWNER" varchar(128) NOT NULL,
  "TABLE_NAME" varchar(128) NOT NULL,
  "FAILURE_TIME" timestamp NOT NULL,
  "KEY_TYPE" varchar(128) NOT NULL,
  "KEY" varchar(7800) NOT NULL,
  "FAILURE_TYPE" varchar(128) NOT NULL,
  "DETAILS" varchar(7000) NOT NULL,
  "RESYNC_RESULT" varchar(128) NULL,
  "RESYNC_TIME" timestamp NULL,
  "RESYNC_ACTION" varchar(128) NULL
);

```

이 테이블에 쿼리를 작성하여 발견된 데이터 불일치와 불일치 해결 방법을 이해할 수 있습니다.

검증이 활성화되면는 대상 데이터베이스에 검증 실패 테이블을 AWS DMS 생성합니다. 문제가 있는 경우 awsdms_control.awsdms_validation_failures_v2 테이블을 쿼리하여 발견된 데이터 불일치와 불일치 해결 방법을 이해할 수 있습니다. 자세한 내용은 [AWS DMS 데이터 검증의 문제 해결](#) 섹션을 참조하세요.

일반적인 워크플로:

데이터 재동기화에서 검증하는 동안 표준 워크플로는 다음과 같습니다.

전체 로드 전용 작업:

1. 데이터베이스의 모든 테이블이 로드 프로세스를 완료합니다.
2. 불일치는 진행 중인 검증 프로세스에서 식별됩니다.
3. 데이터 재동기화는 검증 프로세스 중에 식별된 문제를 해결합니다.
4. 검증 프로세스는 수정을 검증합니다.
5. 마이그레이션 작업이 성공적으로 완료되었습니다.

CDC 작업:

1. 데이터베이스의 모든 테이블이 로드 프로세스를 완료합니다.
2. 불일치는 진행 중인 검증 프로세스에서 식별됩니다.
3. 재동기화 예약 기간에 따라 복제 작업이 잠시 일시 중지됩니다.
4. 데이터 재동기화는 검증 프로세스 중에 식별된 문제를 해결합니다.
5. 복제 프로세스가 재개되고 일정에 따라 반복됩니다.

재동기화 작업 중에 복제 작업을 중지하거나 테이블을 다시 로드하고 재검증하는 등 작업을 수정하면 작업의 동작과 결과에 영향을 미칠 수 있습니다. 알려진 동작 변경 사항 중 일부는 다음과 같습니다.

재동기화 작업이 진행되는 동안 복제 작업을 중지하는 경우:

- 재동기화 작업은 자동으로 재개되지 않습니다. 다시 시작해야 합니다.
- 향후 재동기화 작업은 구성된 일정에 따라 수행됩니다.
- 완료되지 않은 수정은 다음 재동기화 일정 창에서 시도됩니다.

데이터베이스에서 테이블을 다시 로드하는 경우:

- 재동기화 작업은 재로드 중인 테이블을 건너뜁니다.
- 다시 로드된 테이블에 대한 이전 검증 실패는 무시됩니다.
- 재로드 작업이 완료된 후 새 검증이 시작됩니다.

데이터베이스에서 테이블을 재검증하는 경우:

- 재동기화 작업에 대한 모든 통계가 재설정됩니다.
- 재검증된 테이블에 대한 이전 검증 실패는 무시됩니다.

Note

작업을 DMS 버전 3.6.1 이상으로 업그레이드하거나 이동할 때 `awsdms_control.awsdms_validation_failures_v1` 테이블의 장애가 다시 동기화되지 않습니다. `awsdms_validation_failures_v2` 테이블의 실패만 다시 동기화됩니다. `awsdms_control.awsdms_validation_failures_v2` 테이블에서 실패를 다시 동기화하려면 작업을 다시 로드하거나, 작업에 하나 이상의 테이블을 다시 로드하거나, 하나 이상의 테이블을 다시 검증해야 합니다. 자세한 내용은 다음 링크를 참조하세요.

- 작업을 다시 로드하려면 [StartReplicationTask API 참조](#)를 참조하세요.
- 작업에서 하나 이상의 테이블을 다시 로드하려면 AWS CLI 명령 참조 설명서의 [reload-tables](#) 섹션을 참조하세요.
- 하나 이상의 테이블을 다시 검증하려면 AWS CLI 명령 참조 설명서의 [reload-tables](#) 섹션에 있는 `validate-only` 옵션을 참조하세요.

Cron 표현식 규칙

에서 복제 작업 중에 데이터 재동기화 작업을 구성하려면 cron 표현식 규칙을 사용할 AWS DMS 수 있습니다. 이러한 규칙을 사용하면 재동기화 기간을 사용자 지정하고 비즈니스 요구 사항에 따라 예약할 수 있습니다. 분, 시간, 일, 월, 요일과 같은 다양한 파라미터를 사용할 수 있습니다. 각 파라미터에 대한 cron 표현식 규칙은 다음과 같습니다.

분:

- 분 범위는 0~59입니다.
- (-), or/and를 사용하여 범위를 지정할 수 있습니다. 쉼표(,)로 구분된 최대 10개의 항목.
- 예:
 - 2-5는 2, 3, 4, 5와 같습니다.
 - 1-2, 3-4, 5, 7-10은 유효한 범위입니다.
 - 1, 2, 3, 4, 5, 6, 7, 8, 9, 10은 유효한 범위입니다.

- 1,2,3,4,5,6,7,8,9,10,11은 유효한 범위가 아닙니다. 재동기화 작업은 10번째 범위 항목 이후에 건너뛰니다.
- (*)를 사용할 수 있습니다. 예: *는 0-59와 같습니다.
- (/)는 (-) 또는 (*)와 함께만 사용할 수 있습니다.

예:

- 2-7/2는 2,4,6과 같습니다.
- */15는 0,15,30,45와 같습니다.

시간:

‘분’과 동일하지만 유효한 범위는 0 ~ 23입니다.

일:

- ‘분’과 동일하지만 유효한 범위는 1 ~ 31입니다.
- L의 사용은 재동기화 구성에서 지원됩니다. 해당 월의 마지막 날로 통합됩니다. 다른 구문과 함께 사용해서는 안 됩니다.

월:

"분"과 동일하지만 유효한 범위는 1 ~ 12입니다.

요일:

- "분"과 동일하지만 유효한 범위는 0 ~ 6입니다.
- 해당 주의 이름에 문자열 값을 추가할 수 없습니다.

AWS Database Migration Service의 리소스에 태그 지정

AWS Database Migration Service(AWS DMS)에서 태그를 사용하여 리소스에 메타데이터를 추가할 수 있습니다. 또한 AWS Identity and Access Management(IAM) 정책과 함께 이러한 태그를 사용하여 AWS DMS 리소스에 대한 액세스를 관리하고 AWS DMS 리소스에 적용 가능한 작업을 제어할 수 있습니다. 마지막으로 비슷하게 태그가 지정된 리소스에 대한 비용을 그룹화하여 이러한 태그로 비용을 추적할 수 있습니다.

모든 AWS DMS 리소스에 태그를 지정할 수 있습니다.

- Certificates
- 데이터 공급자
- 데이터 마이그레이션
- Endpoints
- 이벤트 구독
- 인스턴스 프로파일
- 마이그레이션 프로젝트
- 복제 인스턴스
- 복제 서브넷 그룹
- 복제 작업

AWS DMS 태그는 사용자가 정의하고 AWS DMS 리소스와 연결하는 이름-값 페어입니다. 이 이름을 키라고 합니다. 키 값을 제공하는 것은 선택 사항입니다. 태그를 사용하여 AWS DMS 리소스에 임의의 정보를 배정할 수 있습니다. 범주 정의 등에 태그 키를 사용할 수 있으며 태그 값은 해당 범주의 항목일 수 있습니다. 예를 들어, 태그 키를 'project'로 정의하고 태그 값을 'Salix'로 정의하여 AWS DMS 리소스가 Salix 프로젝트에 할당됨을 나타냅니다. environment=test나 environment=production 등의 키를 사용하여 태그로 AWS DMS 리소스를 테스트나 프로덕션에 사용되는 것으로 지정할 수도 있습니다. AWS DMS 리소스와 연결된 메타데이터를 더 쉽게 추적할 수 있게 일관성 있는 태그 키 세트를 사용하는 것이 좋습니다.

태그를 사용하여 비용 구조를 반영하도록 AWS 청구서를 구성합니다. 이렇게 하려면 가입하여 태그 키 값이 포함된 AWS 계정 청구서를 가져옵니다. 그런 다음 같은 태그 키 값을 가진 리소스에 따라 결제 정보를 구성하여 리소스 비용의 합을 볼 수 있습니다. 예를 들어, 특정 애플리케이션 이름으로 여러 리소스에 태그를 지정한 다음 결제 정보를 구성하여 여러 서비스에 걸친 해당 애플리케이션의 총 비용을 볼 수 있습니다. 자세한 내용은 AWS Billing 사용 설명서의 [비용 할당 태그 사용](#)을 참조하십시오.

각 AWS DMS 리소스에는 해당 AWS DMS 리소스에 지정되는 모든 태그를 포함하는 태그 세트가 있습니다. 태그 세트는 최대 10개의 태그를 포함하거나 비어 있을 수 있습니다. AWS DMS 리소스의 기존 태그와 동일한 키를 갖는 태그를 리소스에 추가하면 새 값이 이전 값을 덮어씁니다.

AWS는 태그에 의미론적 의미를 적용하지 않으며 태그는 엄격히 문자열로 해석됩니다. AWS DMS는 리소스를 생성할 때 사용하는 설정에 따라, AWS DMS 리소스에 태그를 설정할 수 있습니다.

다음 목록에서는 AWS DMS 태그의 특징을 설명합니다.

- 태그 키는 태그의 필수 이름입니다. 문자열 값은 길이가 1~128자(유니코드 문자)이며 "aws:" 또는 "dms:"로 시작할 수 없습니다. 문자열에는 유니코드 문자, 숫자, 공백, '_', ':', '/', '=', '+', '-'(Java regex: `"^([\p{L}\p{Z}\p{N}_.: / = + \-]*)$"`)만 포함될 수 있습니다.

- 태그 값은 태그의 선택적 문자열 값입니다. 문자열 값은 길이가 1~256자(유니코드 문자)이며 "aws:" 또는 "dms:"로 시작할 수 없습니다. 문자열에는 유니코드 문자, 숫자, 공백, '_', ':', '/', '=', '+', '-'(Java regex: `"^([\p{L}\p{Z}\p{N}_.: / = + \-]*)$"`)만 포함될 수 있습니다.

값은 태그 세트에서 고유할 필요는 없으며 null일 수 있습니다. 예를 들어, project/Trinity 및 cost-center/Trinity의 태그 세트에 키-값 페어가 있을 수 있습니다.

AWS CLI 또는 AWS DMS API를 사용하여 AWS DMS 리소스에서 태그를 추가, 나열, 삭제할 수 있습니다. AWS CLI 또는 AWS DMS API를 사용할 때는 작업하려는 AWS DMS 리소스에 대한 Amazon 리소스 이름(ARN)을 제공해야 합니다. ARN 생성에 대한 자세한 내용은 [에 대한 Amazon 리소스 이름\(ARN\) 구성 AWS DMS](#) 주제단원을 참조하십시오.

권한 부여 목적으로 태그가 캐시됩니다. 이 때문에 AWS DMS 리소스의 태그에 대한 추가나 업데이트가 제공되는 데 몇 분 정도 걸릴 수 있습니다.

API

AWS DMS API를 사용한 AWS DMS 리소스에 대한 태그를 추가, 나열 또는 제거할 수 있습니다.

- AWS DMS 리소스에 태그를 추가하려면 [AddTagsToResource](#) 작업을 사용합니다.
- AWS DMS 리소스에 할당된 태그를 나열하려면 [ListTagsForResource](#) 작업을 사용합니다.
- AWS DMS 리소스에서 태그를 제거하려면 [RemoveTagsFromResource](#) 작업을 사용합니다.

필수 ARN을 생성하는 방법에 대해 자세히 알아보려면 [에 대한 Amazon 리소스 이름\(ARN\) 구성 AWS DMS](#) 단원을 참조하십시오.

AWS DMS API를 사용한 XML 작업 시 다음 스키마를 사용합니다.

```
<Tagging>
  <TagSet>
    <Tag>
      <Key>Project</Key>
      <Value>Trinity</Value>
    </Tag>
    <Tag>
      <Key>User</Key>
      <Value>Jones</Value>
    </Tag>
  </TagSet>
</Tagging>
```

다음 표에는 허용되는 XML 태그와 해당 특성의 목록이 나와 있습니다. Key 및 Value 값은 대/소문자를 구분합니다. 예를 들어, project=Trinity와 PROJECT=Trinity는 서로 다른 두 개의 태그입니다.

태그 지정 요소	설명
TagSet	태그 세트에는 Amazon RDS 리소스에 배정된 모든 태그가 포함됩니다. 리소스당 하나의 태그 세트만 있을 수 있습니다. AWS DMS API를 통해서만 TagSet로 작업합니다.
Tag	태그는 사용자가 정의하는 키-값 페어입니다. 태그 세트에 1~10개의 태그가 있을 수 있습니다.
키	키는 태그의 필수 이름입니다. 문자열 값은 길이가 1~128자(유니코드 문자)이며 "dms:" 또는 "aws:"로 시작할 수 없습니다. 문자열에는 유니코드 문자, 숫자, 공백, '_', ':', '/', '=', '+', '-'(Java regex: "<code>^([\\p{L}\\p{Z}\\p{N}_.:/+\\-]*)\$</code>")만 포함될 수 있습니다. 키는 태그 집합에 대해 고유해야 합니다. 예를 들어, 태그 세트에 project/Trinity와 project/Xanadu처럼 키는 같지만 값은 다른 키-페어가 있을 수 없습니다.
값	값은 태그의 선택적 값입니다. 문자열 값은 길이가 1~256자(유니코드 문자)이며 "dms:" 또는 "aws:"로 시작할 수 없습니다. 문자열에는 유니코드 문자, 숫자, 공백, '_', ':', '/', '=', '+', '-'(Java regex: "<code>^([\\p{L}\\p{Z}\\p{N}_.:/+\\-]*)\$</code>")만 포함될 수 있습니다.

태그 지정 요소	설명
	값은 태그 세트에서 고유할 필요는 없으며 null일 수 있습니다. 예를 들어, project/Trinity 및 cost-center/Trinity의 태그 세트에 키-값 페어가 있을 수 있습니다.

의 보안 AWS Database Migration Service

의 클라우드 보안 AWS 이 최우선 순위입니다. AWS 고객은 보안에 가장 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 이점을 누릴 수 있습니다.

보안은 AWS 와 사용자 간의 공동 책임입니다. [공동 책임 모델](#)은 이를 클라우드의 보안과 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 는 클라우드에서 AWS AWS 서비스를 실행하는 인프라를 보호할 책임이 있습니다. AWS 또한는 안전하게 사용할 수 있는 서비스를 제공합니다. 서드 파티 감사원은 정기적으로 [AWS 규정 준수 프로그램](#)의 일환으로 보안 효과를 테스트하고 검증합니다. 에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 [AWS 규정 준수 프로그램별 범위 내 서비스를 AWS DMS참조하세요](#).
- 클라우드의 보안 - 사용자의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 데이터의 민감도, 조직의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다 AWS DMS. 다음 주제에서는 보안 및 규정 준수 목표를 충족하도록 AWS DMS 를 구성하는 방법을 보여줍니다. 또한 AWS DMS 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법을 알아봅니다.

AWS DMS 리소스 및 데이터베이스(DBs)에 대한 액세스를 관리할 수 있습니다. 액세스를 관리하는 데 사용하는 방법은 수행해야 하는 복제 작업에 따라 다릅니다 AWS DMS.

- AWS Identity and Access Management (IAM) 정책을 사용하여 AWS DMS 리소스를 관리할 수 있는 사용자를 결정하는 권한을 할당합니다. IAM 사용자로 로그인하는 경우 적절한 권한이 있어야 AWS DMS 합니다. 예를 들면, IAM을 사용하여 DB 인스턴스 및 클러스터를 생성, 설명, 수정, 삭제하거나 리소스에 태그를 지정하거나 보안 그룹을 수정할 수 있는 사용자를 결정할 수 있습니다. IAM 및와 함께 사용하는 방법에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [에 대한 자격 증명 및 액세스 관리 AWS Database Migration Service](#).
- AWS DMS 는 전송 계층 보안(TLS)과의 엔드포인트 연결에 SSL(Secure Sockets Layer)을 사용합니다. 에서 SSL/TLS를 사용하는 방법에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [에서 SSL 사용 AWS Database Migration Service](#).
- AWS DMS 는 AWS Key Management Service (AWS KMS) 암호화 키를 사용하여 복제 인스턴스에 사용되는 스토리지와 해당 엔드포인트 연결 정보를 암호화합니다. AWS DMS 또한는 AWS KMS 암호

호화 키를 사용하여 Amazon S3 및 Amazon Redshift 대상 엔드포인트에 대한 저장 데이터를 보호합니다. 자세한 내용은 [암호화 키 설정 및 AWS KMS 권한 지정](#) 단원을 참조하십시오.

- AWS DMS 는 항상 가능한 최대의 네트워크 액세스 제어를 위해 Amazon VPC 서비스를 기반으로 Virtual Private Cloud(VPC)에 복제 인스턴스를 생성합니다. DB 인스턴스 및 인스턴스 클러스터의 경우 복제 인스턴스와 동일한 VPC를 사용하거나 이 액세스 제어 수준에 맞게 추가 VPC를 사용합니다. 사용하는 각 Amazon VPC는 모든 포트의 모든 트래픽이 VPC 외부로 나가도록(송신) 허용하는 규칙이 있는 보안 그룹과 연결되어야 합니다. 해당 엔드포인트에서 올바른 수신이 활성화되어 있는 한 이 접근 방식을 통해 통신을 복제 인스턴스에서 원본과 대상 데이터베이스 엔드포인트로 전달할 수 있습니다.

사용 가능한 네트워크 구성에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [복제 인스턴스용으로 네트워크 설정](#). VPC에서 DB 인스턴스 또는 인스턴스 클러스터를 생성하는 방법에 대한 자세한 내용은 [AWS 설명서](#)에서 Amazon 데이터베이스의 보안 및 클러스터 관리 설명서를 참조하십시오. AWS DMS 를 지원하는 네트워크 구성에 대한 자세한 내용은 [복제 인스턴스용으로 네트워크 설정](#) 섹션을 참조하십시오.

- 데이터베이스 마이그레이션 로그를 확인하려면, 사용 중인 IAM 역할에 적합한 Amazon CloudWatch Logs 권한이 필요합니다. AWS DMS로그에 대한 자세한 내용은 [Amazon CloudWatch를 사용한 복제 태스크 모니터링](#) 섹션을 참조하십시오.

주제

- [의 데이터 보호 AWS Database Migration Service](#)
- [에 대한 자격 증명 및 액세스 관리 AWS Database Migration Service](#)
- [에 대한 규정 준수 검증 AWS Database Migration Service](#)
- [의 복원력 AWS Database Migration Service](#)
- [의 인프라 보안 AWS Database Migration Service](#)
- [리소스 이름 및 태그를 사용하여 세분화된 액세스 제어](#)
- [AWS DMS 동종 마이그레이션을 위한 암호화](#)
- [암호화 키 설정 및 AWS KMS 권한 지정](#)
- [에 대한 네트워크 보안 AWS Database Migration Service](#)
- [에서 SSL 사용 AWS Database Migration Service](#)
- [데이터베이스 암호 변경](#)
- [에서 Kerberos 인증 사용 AWS Database Migration Service](#)

의 데이터 보호 AWS Database Migration Service

데이터 암호화

지원되는 AWS DMS 대상 엔드포인트의 데이터 리소스에 대해 암호화를 활성화할 수 있습니다. AWS DMS 또한 AWS DMS 와 모든 소스 AWS DMS 및 대상 엔드포인트 간의 연결을 암호화합니다. 또한 AWS DMS 및 지원되는 대상 엔드포인트가 암호화를 활성화하는 데 사용하는 키를 관리할 수 있습니다.

주제

- [저장 시 암호화](#)
- [전송 중 암호화](#)
- [키 관리](#)

저장 시 암호화

AWS DMS 는 복제된 데이터를 지원되는 AWS DMS 대상 엔드포인트로 복사하기 전에 Amazon S3로 푸시하는 데 사용할 서버 측 암호화 모드를 지정할 수 있도록 하여 저장 데이터 암호화를 지원합니다. 엔드포인트에 대한 encryptionMode 추가 연결 속성을 설정하여 이 암호화 모드를 지정할 수 있습니다. 이 encryptionMode 설정이 KMS 키 암호화 모드를 지정하는 경우 사용자 지정 AWS KMS 키를 생성하여 다음 대상 엔드포인트에 대한 AWS DMS 대상 데이터를 암호화할 수도 있습니다.

- Amazon Redshift - encryptionMode 설정에 대한 자세한 내용은 [Amazon Redshift를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요. 사용자 지정 AWS KMS 암호화 키 생성에 대한 자세한 내용은 섹션을 참조하세요 [AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화](#).
- Amazon S3 - encryptionMode 설정에 대한 자세한 내용은 [Amazon S3를의 대상으로 사용 시 엔드포인트 설정 AWS DMS](#) 섹션을 참조하세요. 사용자 지정 AWS KMS 암호화 키 생성에 대한 자세한 내용은 섹션을 참조하세요 [Amazon S3 대상 객체를 암호화하는 AWS KMS 키 생성](#).

전송 중 암호화

AWS DMS 는 복제하는 데이터가 소스 엔드포인트에서 대상 엔드포인트로 안전하게 이동하도록 하여 전송 중 암호화를 지원합니다. 여기에는 데이터가 복제 파이프라인을 통해 이동할 때 복제 작업에서 중간 스토리지에 사용하는 복제 인스턴스의 S3 버킷을 암호화하는 작업이 포함됩니다. 소스 및 대상 엔드포인트에 대한 작업 연결을 암호화하려면 SSL(Secure Socket Layer) 또는 TLS(전송 계층 보안)

를 AWS DMS 사용합니다. 두 엔드포인트에 대한 연결을 암호화하여는 소스 엔드포인트에서 복제 작업으로, 작업에서 대상 엔드포인트로 모두 이동할 때 데이터가 안전한지 AWS DMS 확인합니다. 에서 SSL/TLS를 사용하는 방법에 대한 자세한 내용은 섹션을 [AWS DMS참조하세요. 에서 SSL 사용 AWS Database Migration Service](#)

AWS DMS 는 중간 복제 스토리지 및 연결 정보를 모두 암호화하기 위해 기본 키와 사용자 지정 키를 모두 지원합니다. AWS KMS를 사용하여 이러한 키를 관리합니다. 자세한 내용은 [암호화 키 설정 및 AWS KMS 권한 지정](#) 섹션을 참조하세요.

키 관리

AWS DMS 는 복제 스토리지, 연결 정보 및 특정 대상 엔드포인트의 대상 데이터 스토리지를 암호화하는 기본 또는 사용자 지정 키를 지원합니다. 를 사용하여 이러한 키를 관리합니다 AWS KMS. 자세한 내용은 [암호화 키 설정 및 AWS KMS 권한 지정](#) 단원을 참조하십시오.

인터넷워크 트래픽 개인 정보

연결에는 온프레미스에서 실행되든 클라우드에서 AWS 서비스의 일부로 실행되든 동일한 AWS 리전의 AWS DMS 와 소스 및 대상 엔드포인트 간의 보호 기능이 제공됩니다. (소스 또는 대상 엔드포인트를 하나 이상 클라우드에서 AWS 서비스의 일부로 실행해야 합니다.) 이 보호는 이러한 구성 요소가 동일한 Virtual Private Cloud(VPC)를 공유하는지 아니면 VPCs가 모두 동일한 AWS 리전에 있는 경우 별도의 VPCs에 존재하는지에 관계없이 적용됩니다. 에 지원되는 네트워크 구성에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [복제 인스턴스용으로 네트워크 설정](#). 이러한 네트워크 구성 사용 시 보안 고려 사항에 대한 자세한 내용은 [에 대한 네트워크 보안 AWS Database Migration Service](#) 섹션을 참조하세요.

DMS Fleet Advisor의 데이터 보호

DMS Fleet Advisor는 데이터베이스 메타데이터를 수집 및 분석하여 마이그레이션 대상의 적절한 크기를 결정합니다. DMS Fleet Advisor는 테이블의 데이터에 액세스하지 않으며 데이터를 전송하지도 않습니다. 또한 DMS Fleet Advisor는 데이터베이스 기능 사용을 추적하지 않으며 사용 통계에 액세스하지 않습니다.

DMS Fleet Advisor가 데이터베이스 작업에 사용하는 데이터베이스 사용자를 생성하면 데이터베이스에 대한 액세스를 제어할 수 있습니다. 이러한 사용자에게 필요한 권한을 부여합니다. DMS Fleet Advisor를 사용하려면 데이터베이스 사용자에게 읽기 권한을 부여해야 합니다. DMS Fleet Advisor는 데이터베이스를 수정하지 않으며 쓰기 권한이 필요하지 않습니다. 자세한 내용은 [AWS DMS Fleet Advisor용 데이터베이스 사용자 생성](#) 단원을 참조하십시오.

데이터베이스에서 데이터 암호화를 사용할 수 있습니다. AWS DMS 또한은 DMS Fleet Advisor 및 데이터 수집기 내에서 연결을 암호화합니다.

DMS 데이터 수집기는 데이터 보호 애플리케이션 프로그래밍 인터페이스(DPAPI)를 사용하여 고객의 환경 및 데이터베이스 자격 증명 정보에 대한 정보를 암호화, 보호, 저장합니다. DMS Fleet Advisor는 이 암호화된 데이터를 DMS 데이터 수집기가 작동하는 서버의 파일에 저장합니다. DMS Fleet Advisor는 이 데이터를 이 서버 외부로 전송하지 않습니다. DPAPI에 대한 자세한 내용은 [방법: 데이터 보호 사용](#)을 참조하세요.

DMS 데이터 수집기를 설치한 후에는 이 애플리케이션이 지표를 수집하기 위해 실행하는 모든 쿼리를 볼 수 있습니다. 오프라인 모드에서 DMS 데이터 수집기를 실행한 다음 서버에서 수집된 데이터를 검토할 수 있습니다. 또한 Amazon S3 버킷에서 이렇게 수집된 데이터를 검토할 수 있습니다. 자세한 내용은 [DMS 데이터 수집기는 어떻게 작동합니까?](#) 단원을 참조하십시오.

에서 서비스 개선을 위한 데이터 사용 옵트아웃 AWS Database Migration Service

AWS Organizations 옵트아웃 정책을 AWS DMS 사용하여 데이터를 개발 및 개선하는 데 사용하지 않도록 선택할 수 있습니다. AWS DMS가 현재 이러한 데이터를 수집하지 않는 경우에도 옵트아웃을 선택할 수 있습니다. 자세한 내용은 AWS 조직 사용 설명서의 [AI 서비스 옵트아웃 정책](#)을 참조하세요.

현재 AWS Database Migration Service (AWS DMS)는 사용자를 대신하여 처리하는 데이터를 수집하지 않습니다. DMS와 다른 AWS 서비스의 기능을 개발하고 개선하기 위해 DMS는 향후 이러한 데이터를 수집할 수 있습니다. DMS가 데이터를 수집하도록 구성된 경우이 설명서 페이지를 업데이트합니다. 언제든지 옵트아웃할 수 있는 기회가 있습니다.

Note

옵트아웃 정책을 사용하려면 AWS Organizations에서 AWS 계정을 중앙에서 관리해야 합니다. AWS 계정에 대한 조직을 생성하지 않은 경우 [Organizations 사용 설명서의 AWS Managing an organization with](#) Organizations를 참조하세요. AWS

옵트아웃은 다음과 같은 효과가 있습니다.

- AWS DMS는 옵트아웃 전에 서비스 개선을 위해 수집 및 저장된 데이터를 삭제합니다(있는 경우).
- 옵트아웃한 후에는 서비스 개선을 위해 더 이상이 데이터를 수집하거나 저장 AWS DMS 하지 마십시오.

의 교차 리전 추론 AWS Database Migration Service

특정 AWS Database Migration Service 기능은 리전 간 AI 추론을 사용하여 지리 AWS 리전 내에서 최적의 결과를 자동으로 선택하여 추론 요청을 처리합니다. 이 접근 방식은 사용 가능한 컴퓨팅 리소스와 모델 가용성을 극대화하고 고품질 고객 경험을 제공하도록 설계되었습니다. 교차 리전 추론을 사용하면 다음을 얻을 수 있습니다.

- 최첨단 AI 기능 및 기능에 대한 액세스
- 수요가 많은 기간에 처리량 및 복원력 향상

교차 리전 AI 추론 요청은 기본과 동일한 지리의 일부 AWS 리전 인 내에 유지됩니다 AWS 리전. 예를 들어 AWS 리전 미국의 기본에서 이루어진 요청은 AWS 리전 미국의 내에 보관됩니다. 데이터는 기본에만 저장됩니다 AWS 리전. 모든 데이터는 Amazon의 보안 네트워크를 통해 암호화되어 전송됩니다.

Note

Amazon CloudWatch 및 AWS CloudTrail 로그는 AI 추론이 발생하는 AWS 리전 를 지정하지 않습니다.

DMS Schema Conversion의 교차 리전 추론

DMS Schema Conversion에서 생성형 AI 기능을 사용하는 경우 익명화된 코드 조각 및 관련 스키마 메타데이터가 AI 처리를 위해 동일한 지리 AWS 리전 내의 다른 로 전송될 수 있습니다. 프로덕션 데이터는 기본 AWS 리전 에 유지되며 액세스되거나 전송되지 않습니다.

Important

DMS Schema Conversion에서 생성형 AI 기능을 사용하는 경우 교차 리전 추론이 항상 활성화됩니다. 스키마 변환 처리를 기본에 상주하게 유지하려면 비활성화된 생성형 AI 기능과 함께 스키마 변환을 AWS 리전사용합니다.

DMS Schema Conversion의 생성형 AI 기능은 현재 제한된 수의 리전에서 사용할 수 있습니다. 다음 표에서는 기본 AWS 리전 요청에 따라 라우팅할 수 있는 항목을 설명합니다 AWS 리전.

기본 AWS 리전	추론 AWS 리전
아시아 태평양(도쿄)(ap-northeast-1)	아시아 태평양(도쿄)(ap-northeast-1) 아시아 태평양(오사카) (ap-northeast-3)
아시아 태평양(오사카) (ap-northeast-3)	아시아 태평양(도쿄)(ap-northeast-1) 아시아 태평양(오사카) (ap-northeast-3)
아시아 태평양(시드니)(ap-southeast-2)	아시아 태평양(시드니)(ap-southeast-2) 아시아 태평양(멜버른)(ap-southeast-4)
캐나다(중부)(ca-central-1)	캐나다(중부)(ca-central-1) 미국 동부(버지니아 북부)(us-east-1) 미국 동부(오하이오)(us-east-2) 미국 서부(오리건)(us-west-2)
유럽(프랑크푸르트)(eu-central-1)	유럽(프랑크푸르트)(eu-central-1) 유럽(스톡홀름)(eu-north-1) 유럽(밀라노) (eu-south-1) 유럽(스페인)(eu-south-2) 유럽(아일랜드)(eu-west-1) 유럽(파리) (eu-west-3)
유럽(스톡홀름)(eu-north-1)	유럽(프랑크푸르트) (eu-central-1) 유럽(스톡홀름)(eu-north-1) 유럽(밀라노) (eu-south-1) 유럽(스페인)(eu-south-2) 유럽(아일랜드)(eu-west-1)

기본 AWS 리전	추론 AWS 리전
	유럽(파리)(eu-west-3)
유럽(아일랜드)(eu-west-1)	유럽(프랑크푸르트)(eu-central-1)
	유럽(스톡홀름)(eu-north-1)
	유럽(밀라노) (eu-south-1)
	유럽(스페인)(eu-south-2)
	유럽(아일랜드)(eu-west-1)
	유럽(파리)(eu-west-3)
유럽(런던)(eu-west-2)	유럽(프랑크푸르트)(eu-central-1)
	유럽(스톡홀름)(eu-north-1)
	유럽(밀라노) (eu-south-1)
	유럽(스페인)(eu-south-2)
	유럽(아일랜드)(eu-west-1)
	유럽(런던) (eu-west-2)
	유럽(파리)(eu-west-3)
유럽(파리)(eu-west-3)	유럽(프랑크푸르트)(eu-central-1)
	유럽(스톡홀름)(eu-north-1)
	유럽(밀라노) (eu-south-1)
	유럽(스페인)(eu-south-2)
	유럽(아일랜드)(eu-west-1)
	유럽(파리)(eu-west-3)

기본 AWS 리전	추론 AWS 리전
미국 동부(버지니아 북부)(us-east-1)	미국 동부(버지니아 북부)(us-east-1) 미국 동부(오하이오)(us-east-2) 미국 서부(오리건)(us-west-2)
미국 동부(오하이오)(us-east-2)	미국 동부(버지니아 북부)(us-east-1) 미국 동부(오하이오)(us-east-2) 미국 서부(오리건)(us-west-2)
미국 서부(오리건)(us-west-2)	미국 동부(버지니아 북부)(us-east-1) 미국 동부(오하이오)(us-east-2) 미국 서부(오리건)(us-west-2)

에 대한 자격 증명 및 액세스 관리 AWS Database Migration Service

AWS Identity and Access Management (IAM)는 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어하는 데 도움이 되는 AWS 서비스입니다. IAM 관리자는 누가 AWS DMS 리소스를 사용할 수 있는 인증(로그인) 및 권한(권한 있음)을 받을 수 있는지 제어합니다. IAM은 추가 비용 없이 사용할 수 있는 AWS 서비스입니다.

주제

- [대상](#)
- [ID를 통한 인증](#)
- [정책을 사용하여 액세스 관리](#)
- [AWS Database Migration Service 에서 IAM을 사용하는 방법](#)
- [AWS Database Migration Service 자격 증명 기반 정책 예제](#)
- [에 대한 리소스 기반 정책 예제 AWS KMS](#)
- [보안 암호를 사용하여 AWS Database Migration Service 엔드포인트 액세스](#)
- [에 대한 서비스 연결 역할 사용 AWS DMS](#)
- [AWS Database Migration Service 자격 증명 및 액세스 문제 해결](#)
- [를 사용하는 데 필요한 IAM 권한 AWS DMS](#)
- [와 함께 사용할 IAM 역할 생성 AWS DMS](#)
- [교차 서비스 혼동된 대리인 방지](#)
- [AWS 에 대한 관리형 정책 AWS Database Migration Service](#)

대상

AWS Identity and Access Management (IAM)를 사용하는 방법은 역할에 따라 다릅니다.

- 서비스 사용자 - 기능에 액세스할 수 없는 경우 관리자에게 권한 요청([참조 AWS Database Migration Service 자격 증명 및 액세스 문제 해결](#))
- 서비스 관리자 - 사용자 액세스 결정 및 권한 요청 제출([AWS Database Migration Service 에서 IAM 을 사용하는 방법](#) 참조)
- IAM 관리자 - 액세스를 관리하기 위한 정책 작성([AWS Database Migration Service 자격 증명 기반 정책 예제](#) 참조)

ID를 통한 인증

인증은 자격 증명 자격 증명을 AWS 사용하여 로그인하는 방법입니다. AWS 계정 루트 사용자, IAM 사용자 또는 IAM 역할을 수임하여 인증되어야 합니다.

AWS IAM Identity Center (IAM Identity Center), Single Sign-On 인증 또는 Google/Facebook 자격 증명과 같은 자격 증명 소스의 자격 증명을 사용하여 페더레이션 자격 증명으로 로그인할 수 있습니다. 로그인하는 방법에 대한 자세한 내용은 AWS 로그인 사용 설명서의 [AWS 계정에 로그인하는 방법](#) 섹션을 참조하세요.

프로그래밍 방식 액세스를 위해서는 요청에 암호화 방식으로 서명할 수 있는 SDK 및 CLI를 AWS 제공합니다. 자세한 내용은 IAM 사용 설명서의 [API 요청용 AWS Signature Version 4](#) 섹션을 참조하세요.

AWS 계정 루트 사용자

를 생성할 때 모든 AWS 서비스 및 리소스에 대한 완전한 액세스 권한이 있는 AWS 계정 theroot 사용자라는 하나의 로그인 자격 증명으로 AWS 계정시작합니다. 일상적인 태스크에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자가 필요한 작업 목록은 IAM 사용자 설명서의 [루트 사용자 자격 증명이 필요한 작업](#)을 참조하세요.

IAM 사용자 및 그룹

[IAM 사용자](#)는 단일 개인 또는 애플리케이션에 대한 특정 권한을 가진 ID입니다. 장기 자격 증명이 있는 IAM 사용자 대신 임시 자격 증명을 사용하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 [자격 증명 공급자와의 페더레이션을 사용하여 임시 자격 증명을 AWS 사용하여 액세스하도록 인간 사용자에게 요구](#)하기를 참조하세요.

[IAM 그룹](#)은 IAM 사용자 모음을 지정하고 대규모 사용자 집합에 대한 관리 권한을 더 쉽게 만듭니다. 자세한 내용은 IAM 사용 설명서의 [IAM 사용자 사용 사례](#) 섹션을 참조하세요.

IAM 역할

[IAM 역할](#)은 임시 자격 증명을 제공하는 특정 권한이 있는 자격 증명입니다. [사용자에서 IAM 역할\(콘솔\)로 전환하거나 또는 API 작업을 호출하여 역할을](#) 수임할 수 있습니다. AWS CLI AWS 자세한 내용은 IAM 사용 설명서의 [역할 수임 방법](#)을 참조하세요.

IAM 역할은 페더레이션 사용자 액세스, 임시 IAM 사용자 권한, 교차 계정 액세스, 교차 서비스 액세스 및 Amazon EC2에서 실행되는 애플리케이션에 유용합니다. 자세한 내용은 IAM 사용 설명서의 [교차 계정 리소스 액세스](#)를 참조하세요.

정책을 사용하여 액세스 관리

정책을 AWS 생성하고 자격 증명 또는 리소스에 연결하여 AWS 에서 액세스를 제어합니다. 정책은 자격 증명 또는 리소스와 연결될 때 권한을 정의합니다. 보안 주체가 요청할 때 이러한 정책을 AWS 평가합니다. 대부분의 정책은 JSON 문서 AWS 로 저장됩니다. JSON 정책 문서에 대한 자세한 내용은 IAM 사용 설명서의 [JSON 정책 개요](#) 섹션을 참조하세요.

정책을 사용하여 관리자는 어떤 보안 주체가 어떤 리소스에 대해 어떤 조건에서 작업을 수행할 수 있는지 정의하여 누가 무엇을 액세스할 수 있는지 지정합니다.

기본적으로 사용자 및 역할에는 어떠한 권한도 없습니다. IAM 관리자는 IAM 정책을 생성하고 사용자가 수임할 수 있는 역할에 추가합니다. IAM 정책은 작업을 수행하기 위해 사용하는 방법과 관계없이 작업에 대한 권한을 정의합니다.

ID 기반 정책

ID 기반 정책은 ID(사용자, 사용자 그룹 또는 역할)에 연결하는 JSON 권한 정책 문서입니다. 이러한 정책은 자격 증명이 수행할 수 있는 작업, 대상 리소스 및 이에 관한 조건을 제어합니다. ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 [고객 관리형 정책으로 사용자 지정 IAM 권한 정의](#)를 참조하세요.

ID 기반 정책은 인라인 정책(단일 ID에 직접 포함) 또는 관리형 정책(여러 ID에 연결된 독립 실행형 정책)일 수 있습니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 [관리형 정책 및 인라인 정책 중에서 선택](#) 섹션을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 예를 들어 IAM 역할 신뢰 정책 및 Amazon S3 버킷 정책이 있습니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 리소스 기반 정책에서 [보안 주체를 지정](#)해야 합니다.

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. 리소스 기반 정책에서는 IAM의 AWS 관리형 정책을 사용할 수 없습니다.

액세스 제어 목록(ACL)

액세스 제어 목록(ACL)은 어떤 위탁자(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

Amazon S3 AWS WAF 및 Amazon VPC는 ACLs. ACL에 관한 자세한 내용은 Amazon Simple Storage Service 개발자 가이드의 [액세스 제어 목록\(ACL\) 개요](#)를 참조하세요.

기타 정책 타입

AWS 는 보다 일반적인 정책 유형에서 부여한 최대 권한을 설정할 수 있는 추가 정책 유형을 지원합니다.

- 권한 경계 - ID 기반 정책에서 IAM 엔터티에 부여할 수 있는 최대 권한을 설정합니다. 자세한 정보는 IAM 사용 설명서의 [IAM 엔터티의 권한 범위](#)를 참조하세요.
- 서비스 제어 정책(SCP) - AWS Organizations내 조직 또는 조직 단위에 대한 최대 권한을 지정합니다. 자세한 내용은AWS Organizations 사용 설명서의 [서비스 제어 정책](#)을 참조하세요.
- 리소스 제어 정책(RCP) - 계정의 리소스에 사용할 수 있는 최대 권한을 설정합니다. 자세한 내용은 AWS Organizations 사용 설명서의 [리소스 제어 정책\(RCP\)](#)을 참조하세요.
- 세션 정책 - 역할 또는 페더레이션 사용자에게 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 자세한 내용은 IAM 사용 설명서의 [세션 정책](#)을 참조하세요.

여러 정책 유형

여러 정책 유형이 요청에 적용되는 경우, 결과 권한은 이해하기가 더 복잡합니다. 에서 여러 정책 유형이 관련될 때 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 [정책 평가 로직](#)을 참조하세요.

AWS Database Migration Service 에서 IAM을 사용하는 방법

IAM을 사용하여 액세스를 관리하기 전에 사용할 수 있는 IAM 기능을 이해해야 AWS DMS합니다 AWS DMS. AWS DMS 및 기타 AWS 서비스에서 IAM을 사용하는 방법을 개괄적으로 알아보려면 IAM 사용 설명서의 [AWS IAM으로 작업하는 서비스를](#) 참조하세요.

주제

- [AWS DMS 자격 증명 기반 정책](#)
- [AWS DMS 리소스 기반 정책](#)
- [AWS DMS 태그 기반 권한 부여](#)
- [예 대한 IAM 역할 AWS DMS](#)
- [DMS Fleet Advisor의 ID 및 액세스 관리](#)

AWS DMS 자격 증명 기반 정책

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업 및 리소스를 지정할 수 있을 뿐 아니라 작업이 허용되거나 거부되는 조건도 지정할 수 있습니다. AWS DMS 는 특정한 작업, 리소스 및 조건 키를 지원합니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용자 설명서의 [IAM JSON 정책 요소 참조](#)를 참조하세요.

작업

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 Action요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 작업을 설명합니다. 연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하세요.

의 정책 작업은 작업 앞에 접두사를 AWS DMS 사용합니다 dms:. 예를 들어 API 작업을 사용하여 복제 작업을 생성할 수 AWS DMS CreateReplicationTask 있는 권한을 부여하려면 해당 정책에 dms:CreateReplicationTask 작업을 포함합니다. 정책 설명에는 Action 또는 NotAction 요소가 포함되어야 합니다. 이 서비스로 수행할 수 있는 작업을 설명하는 고유한 작업 세트를 AWS DMS 정의합니다.

명령문 하나에 여러 작업을 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [
    "dms:action1",
    "dms:action2"
```

와일드카드(*)를 사용하여 여러 작업을 지정할 수 있습니다. 예를 들어, Describe라는 단어로 시작하는 모든 작업을 지정하려면 다음 작업을 포함합니다.

```
"Action": "dms:Describe*"
```

AWS DMS 작업 목록을 보려면 IAM 사용자 설명서의 [에서 정의한 작업을 AWS Database Migration Service](#) 참조하세요.

리소스

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

Resource JSON 정책 요소는 작업이 적용되는 하나 이상의 객체를 지정합니다. 모범 사례에 따라 [Amazon 리소스 이름\(ARN\)](#)을 사용하여 리소스를 지정합니다. 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"
```

AWS DMS 는 다음 리소스와 함께 작동합니다.

- 인증서
- 엔드포인트
- 이벤트 구독
- 복제 인스턴스
- 복제 서브넷(보안) 그룹
- 복제 작업

에 AWS DMS 필요한 리소스는 호출하는 작업에 따라 달라집니다. 리소스 ARN에서 지정한 관련 리소스에 대해 이러한 작업을 허용하는 정책이 필요합니다.

예를 들어 AWS DMS 엔드포인트 리소스의 ARN은 다음과 같습니다.

```
arn:${Partition}:dms:${Region}:${Account}:endpoint/${InstanceId}
```

ARN 형식에 대한 자세한 내용은 [Amazon 리소스 이름\(ARNs\) 및 AWS 서비스 네임스페이스를 참조하세요](#).

예를 들어 문에서 us-east-2 리전의 1A2B3C4D5E6F7G8H9I0J1K2L3M 엔드포인트 인스턴스를 지정하려면 다음 ARN을 사용합니다.

```
"Resource": "arn:aws:dms:us-east-2:987654321098:endpoint/1A2B3C4D5E6F7G8H9I0J1K2L3M"
```

특정 계정에 속하는 모든 엔드포인트를 지정하려면 와일드카드(*)를 사용합니다.

```
"Resource": "arn:aws:dms:us-east-2:987654321098:endpoint/*"
```

리소스를 생성하기 위한 작업과 같은 일부 AWS DMS 작업은 특정 리소스에서 수행할 수 없습니다. 이러한 경우, 와일드카드(*)를 사용해야 합니다.


```
"Resource": "*"

```

일부 AWS DMS API 작업에는 여러 리소스가 포함됩니다. 예를 들어 StartReplicationTask를 시작하고 복제 작업을 두 개의 데이터베이스 엔드포인트 리소스(소스 및 대상)에 연결하므로 IAM 사용하는 소스 엔드포인트를 읽고 대상 엔드포인트에 쓸 수 있는 권한이 있어야 합니다. 단일 문에서 여러 리소스를 지정하려면 ARN을 쉼표로 구분합니다.

```
"Resource": [
    "resource1",
    "resource2" ]

```

정책을 사용하여 AWS DMS 리소스에 대한 액세스를 제어하는 방법에 대한 자세한 내용은 섹션을 참조하세요 [리소스 이름을 사용하여 액세스 제어](#). AWS DMS 리소스 유형 및 해당 ARN의 목록을 보려면 IAM 사용자 설명서의 [AWS Database Migration Service에서 정의된 리소스](#)를 참조하세요. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 [AWS Database Migration Service가 정의한 작업](#)을 참조하세요.

조건 키

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

Condition 요소는 정의된 기준에 따라 문이 실행되는 시기를 지정합니다. 같음(equals) 또는 미만(less than)과 같은 [조건 연산자](#)를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다. 모든 AWS 전역 조건 키를 보려면 IAM 사용 설명서의 [AWS 전역 조건 컨텍스트 키](#)를 참조하세요.

AWS DMS 는 자체 조건 키 세트를 정의하고 일부 전역 조건 키 사용을 지원합니다. 모든 AWS 전역 조건 키를 보려면 IAM 사용 설명서의 [AWS 전역 조건 컨텍스트 키](#)를 참조하세요.

AWS DMS 는 조건 키에 사용할 수 있는 표준 태그 세트를 정의하고 사용자 지정 태그를 정의할 수도 있습니다. 자세한 내용은 [태그를 사용하여 액세스 제어](#) 단원을 참조하십시오.

AWS DMS 조건 키 목록을 보려면 IAM 사용 설명서의 [대한 조건 키를 참조하세요 AWS Database Migration Service](#). 조건 키를 사용할 수 있는 작업과 리소스에 대해 알아보려면 [AWS Database Migration Service에서 정의한 작업](#) 및 [AWS Database Migration Service에서 정의한 리소스](#)를 참조하세요.

예제

자격 AWS DMS 증명 기반 정책의 예를 보려면 섹션을 참조하세요 [AWS Database Migration Service 자격 증명 기반 정책 예제](#).

AWS DMS 리소스 기반 정책

리소스 기반 정책은 지정된 보안 주체가 지정된 AWS DMS 리소스에 대해 수행할 수 있는 작업과 조건을 지정하는 JSON 정책 문서입니다.는 지원되는 대상 엔드포인트로 마이그레이션된 데이터를 암호화하기 위해 생성하는 AWS KMS 암호화 키에 대한 리소스 기반 권한 정책을 AWS DMS 지원합니다. 지원되는 대상 엔드포인트에는 Amazon Redshift 및 Amazon S3 등이 있습니다. 리소스 기반 정책을 사용하여 각 대상 엔드포인트의 다른 계정에 이러한 암호화 키를 사용할 수 있는 권한을 부여할 수 있습니다.

교차 계정 액세스를 활성화하려는 경우, 전체 계정이나 다른 계정의 IAM 엔터티를 [리소스 기반 정책의 위탁자](#)로 지정할 수 있습니다. 리소스 기반 정책에 교차 계정 위탁자를 추가하는 것은 트러스트 관계 설정의 절반밖에 되지 않는다는 것을 유념하세요. 보안 주체와 리소스가 서로 다른 AWS 계정에 있는 경우 보안 주체 엔터티에 리소스에 액세스할 수 있는 권한도 부여해야 합니다. 엔터티에 자격 증명 기반 정책을 연결하여 권한을 부여합니다. 하지만 리소스 기반 정책이 동일 계정의 위탁자에 액세스를 부여하는 경우 추가 자격 증명 기반 정책이 필요하지 않습니다. 자세한 내용은 IAM 사용자 설명서의 [IAM 역할과 리소스 기반 정책의 차이](#)를 참조하세요.

이 AWS DMS 서비스는 AWS KMS 암호화 키에 연결된 키 정책이라는 한 가지 유형의 리소스 기반 정책만 지원합니다. 이 정책은 지원되는 대상 엔드포인트에서 마이그레이션된 데이터를 암호화할 수 있는 위탁자 엔터티(계정, 사용자, 역할 및 페더레이션 사용자)를 정의합니다.

지원되는 대상 엔드포인트에 대해 생성한 암호화 키에 리소스 기반 정책을 연결하는 방법을 알아보려면 [AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화](#) 및 [Amazon S3 대상 객체를 암호화하는 AWS KMS 키 생성](#) 섹션을 참조하세요.

예제

AWS DMS 리소스 기반 정책의 예는 섹션을 참조하세요 [에 대한 리소스 기반 정책 예제 AWS KMS](#).

AWS DMS 태그 기반 권한 부여

AWS DMS 리소스에 태그를 연결하거나 요청에서 태그를 전달할 수 있습니다 AWS DMS. 태그를 기반으로 액세스를 제어하려면, `dms:ResourceTag/key-name` `aws:RequestTag/key-name` 또는 [조건 키를 사용하여 정책의 조건 요소에](#) 태그 정보를 제공합니다.는 조건 키에 사용할 수 있는 표준 태그 세트를 AWS DMS 정의하고 사용자 지정 태그를 정의할 수도 있습니다. `aws:TagKeys` 자세한 내용은 [태그를 사용하여 액세스 제어](#) 단원을 참조하십시오.

태그를 기반으로 리소스에 대한 액세스를 제한하는 자격 증명 기반 정책의 예는 [태그를 기반으로 AWS DMS 리소스 액세스](#) 섹션을 참조하세요.

에 대한 IAM 역할 AWS DMS

[IAM 역할](#)은 특정 권한이 있는 AWS 계정 내 엔터티입니다.

에서 임시 자격 증명 사용 AWS DMS

임시 자격 증명을 사용하여 페더레이션으로 로그인하거나 IAM 역할을 수입하거나 교차 계정 역할을 수입할 수 있습니다. [AssumeRole](#) 또는 [GetFederationToken](#)과 같은 AWS STS API 작업을 호출하여 임시 보안 자격 증명을 얻을 수 있습니다.

AWS DMS 는 임시 자격 증명 사용을 지원합니다.

서비스 연결 역할

[서비스 연결 역할](#)을 사용하면 AWS 서비스가 다른 서비스의 리소스에 액세스하여 사용자를 대신하여 작업을 완료할 수 있습니다. 서비스 연결 역할은 IAM 계정에 나타나고 서비스가 소유합니다. IAM 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집할 수 없습니다.

AWS DMS 서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 섹션을 참조하세요 [서비스 연결 역할 사용](#).

서비스 역할

이 기능을 사용하면 서비스가 사용자를 대신하여 [서비스 역할](#)을 수입할 수 있습니다. 이 역할을 사용하면 서비스가 다른 서비스의 리소스에 액세스해 사용자를 대신해 작업을 완료할 수 있습니다. 서비스 역할은 IAM 계정에 나타나고, 해당 계정이 소유합니다. 즉, IAM 관리자가 이 역할에 대한 권한을 변경할 수 있습니다. 그러나 권한을 변경하면 서비스의 기능이 손상될 수 있습니다.

AWS DMS 는 특정 소스 또는 대상 엔드포인트를 사용하기 위해 생성해야 하는 두 가지 유형의 서비스 역할을 지원합니다.

- 다음 소스 및 대상 엔드포인트(또는 해당 리소스)에 대한 AWS DMS 액세스를 허용할 권한이 있는 역할:
 - 대상으로서 Amazon DynamoDB – 자세한 내용은 [DynamoDB를의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service](#)를 참조하세요.
 - OpenSearch 대상으로서 - 자세한 내용은 단원을 참조하십시오 [Amazon OpenSearch Service를의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service](#).

- 대상으로서 Amazon Kinesis – 자세한 내용은 [Kinesis 데이터 스트림을의 대상으로 사용하기 위한 사전 조건 AWS Database Migration Service](#)를 참조하세요.
- 대상으로서 Amazon Redshift – 대상 데이터를 암호화하기 위한 사용자 지정 KMS 암호화 키를 생성하거나 중간 작업 스토리지를 보관할 사용자 지정 S3 버킷을 지정하는 경우에만 지정된 역할을 생성해야 합니다. 자세한 내용은 [AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화](#) 또는 [Amazon S3 버킷 설정](#)을 참조하세요.
- 소스 또는 대상으로서 Amazon S3 – 자세한 내용은 [Amazon S3를의 소스로 사용할 때의 사전 조건 AWS DMS](#) 또는 [Amazon S3를 대상으로 사용하기 위한 사전 조건](#) 섹션을 참조하세요.

예를 들어, S3 소스 엔드포인트에서 데이터를 읽거나 S3 대상 엔드포인트로 데이터를 푸시하려면 이러한 엔드포인트 작업 각각에서 S3에 액세스하기 위한 전제 조건으로 서비스 역할을 생성해야 합니다.

- AWS DMS 콘솔, AWS CLI 및 AWS DMS API를 사용하는 데 필요한 권한이 있는 역할 - 생성해야 하는 두 가지 IAM 역할은 dms-vpc-role 및 dms-cloudwatch-logs-role입니다. Amazon Redshift를 대상 데이터베이스로 사용하는 경우 IAM 역할도 생성하고 dms-access-for-endpoint AWS 계정에 추가해야 합니다. 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 단원을 참조하십시오.

에서 IAM 역할 선택 AWS DMS

데이터베이스 마이그레이션에 AWS DMS 콘솔, AWS CLI 또는 AWS DMS API를 사용하는 경우 AWS DMS의 기능을 사용하려면 먼저 AWS 계정에 특정 IAM 역할을 추가해야 합니다. 이 중 두 가지는 dms-vpc-role과 dms-cloudwatch-logs-role입니다. Amazon Redshift를 대상 데이터베이스로 사용하는 경우 dms-access-for-endpoint AWS 계정에 IAM 역할도 추가해야 합니다. 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 단원을 참조하십시오.

DMS Fleet Advisor의 ID 및 액세스 관리

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스를 지정할 수 있으며 작업이 허용되거나 거부되는 조건도 지정할 수 있습니다. DMS Fleet Advisor는 특정 작업, 리소스 및 조건 키를 지원합니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용자 설명서의 [IAM JSON 정책 요소 참조](#)를 참조하세요.

DMS Fleet Advisor는 IAM 역할을 사용하여 Amazon Simple Storage Service에 액세스합니다. [IAM 역할](#)은 특정 권한이 있는 AWS 계정 내 엔티티입니다. 자세한 내용은 [IAM 리소스 생성](#) 단원을 참조하십시오.

AWS Database Migration Service 자격 증명 기반 정책 예제

기본적으로 IAM 사용자 및 역할은 AWS DMS 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 또한 AWS Management Console AWS CLI 또는 AWS API를 사용하여 작업을 수행할 수 없습니다. IAM 관리자는 지정된 리소스에서 특정 API 작업을 수행할 수 있는 권한을 사용자와 역할에게 부여하는 IAM 정책을 생성해야 합니다. 그런 다음 관리자는 해당 권한이 필요한 IAM 사용자 또는 그룹에 이러한 정책을 연결해야 합니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용자 설명서의 [JSON 탭에서 정책 생성](#)을 참조하세요.

주제

- [정책 모범 사례](#)
- [AWS DMS 콘솔 사용](#)
- [사용자가 자신의 고유한 권한을 볼 수 있도록 허용](#)
- [하나의 Amazon S3 버킷에 액세스](#)
- [태그를 기반으로 AWS DMS 리소스 액세스](#)

정책 모범 사례

자격 증명 기반 정책에 따라 계정에서 사용자가 AWS DMS 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부가 결정됩니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책을 시작하고 최소 권한으로 전환 - 사용자 및 워크로드에 권한 부여를 시작하려면 많은 일반적인 사용 사례에 대한 권한을 부여하는 AWS 관리형 정책을 사용합니다. 에서 사용할 수 있습니다 AWS 계정. 사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 추가로 줄이는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 [AWS 관리형 정책](#) 또는 [AWS 직무에 대한 관리형 정책](#)을 참조하세요.
- 최소 권한 적용 - IAM 정책을 사용하여 권한을 설정하는 경우, 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 [IAM의 정책 및 권한](#)을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 - 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어, SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. AWS 서비스와 같은 특징을 통해 사용되는 경우 조건을 사용하여 서

비즈니스 작업에 대한 액세스 권한을 부여할 수도 있습니다 CloudFormation. 자세한 내용은 IAM 사용 설명서의 [IAM JSON 정책 요소: 조건](#)을 참조하세요.

- IAM Access Analyzer를 통해 IAM 정책을 확인하여 안전하고 기능적인 권한 보장 - IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 새로운 및 기존 정책을 확인합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 내용은 IAM 사용 설명서의 [IAM Access Analyzer에서 정책 검증](#)을 참조하세요.
- 다중 인증(MFA) 필요 -에서 IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 MFA를 AWS 계정입니다. API 작업을 직접적으로 호출할 때 MFA가 필요하다면 정책에 MFA 조건을 추가합니다. 자세한 내용은 IAM 사용 설명서의 [MFA를 통한 보안 API 액세스](#)를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 [IAM의 보안 모범 사례](#)를 참조하세요.

AWS DMS 콘솔 사용

다음 정책은 AWS DMS 콘솔을 포함하여 AWS DMS에 대한 액세스 권한을 부여하고 Amazon EC2와 같은 다른 Amazon 서비스에 필요한 특정 작업에 대한 권한도 지정합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "dms:*",
      "Resource": "arn:aws:dms:*:123456789012:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:ListAliases",
        "kms:DescribeKey"
      ],
      "Resource": "arn:aws:kms:*:123456789012:key/*"
    },
    {
      "Effect": "Allow",
      "Action": [
```

```

        "iam:GetRole",
        "iam:CreateRole",
        "iam:AttachRolePolicy"
    ],
    "Resource": "arn:aws:iam::123456789012:role/*"
},
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "arn:aws:iam::123456789012:role/*",
    "Condition": {
        "StringEquals": {
            "iam:PassedToService": "dms.amazonaws.com"
        }
    }
},
{
    "Effect": "Allow",
    "Action": [
        "ec2:DescribeVpcs",
        "ec2:DescribeInternetGateways",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:ModifyNetworkInterfaceAttribute",
        "ec2:CreateNetworkInterface",
        "ec2>DeleteNetworkInterface"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "cloudwatch:Get*",
        "cloudwatch:List*"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams",
        "logs:FilterLogEvents",

```

```

        "logs:GetLogEvents"
      ],
      "Resource": "arn:aws:logs:*:123456789012:*"
    }
  ]
}

```

이러한 권한을 분석하면 콘솔을 사용하는 데 각 권한이 필요한 이유를 더 잘 이해할 수 있습니다.

다음 섹션은 사용자가 콘솔에 표시할 수 있는 AWS KMS 키와 별칭을 나열할 수 있도록 허용하는 데 필요합니다. KMS 키에 대한 Amazon 리소스 이름(ARN)을 알고 있고 AWS Command Line Interface (AWS CLI)만을 사용할 때는 이 항목이 필요하지 않습니다.

```

{
    "Effect": "Allow",
    "Action": [
        "kms:ListAliases",
        "kms:DescribeKey"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}

```

엔드포인트를 사용하여 역할 ARN을 전달해야 하는 특정 엔드포인트 유형에는 다음 섹션이 필요합니다. 또한 필요한 AWS DMS 역할을 미리 생성하지 않으면 AWS DMS 콘솔에서 역할을 생성할 수 있습니다. 모든 역할이 사전에 구성된 경우, iam:GetRole 및 iam:PassRole만 있으면 됩니다. 역할에 관한 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.

```

{
    "Effect": "Allow",
    "Action": [
        "iam:GetRole",
        "iam:PassRole",
        "iam:CreateRole",
        "iam:AttachRolePolicy"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}

```


다음 섹션은 Amazon EC2 인스턴스를 생성하고 생성된 복제 인스턴스에 대한 네트워크를 구성 AWS DMS 해야 하기 때문에 필요합니다. 이러한 리소스는 고객의 계정에 있으므로 고객 대신에 이러한 작업을 수행하는 기능이 필요합니다.

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:DescribeVpcs",
        "ec2:DescribeInternetGateways",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:ModifyNetworkInterfaceAttribute",
        "ec2:CreateNetworkInterface",
        "ec2>DeleteNetworkInterface"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}
```

다음 섹션에서는 사용자가 복제 인스턴스 지표를 볼 수 있도록 허용해야 합니다.

```
{
    "Effect": "Allow",
    "Action": [
        "cloudwatch:Get*",
        "cloudwatch:List*"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}
```

이 섹션에서는 사용자가 복제 로그를 볼 수 있도록 허용해야 합니다.

```
{
    "Effect": "Allow",
    "Action": [
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams",
        "logs:FilterLogEvents",
        "logs:GetLogEvents"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}
```

마이그레이션에 AWS DMS 콘솔, AWS Command Line Interface (AWS CLI) 또는 AWS DMS API를 사용하는 경우 계정에 여러 역할을 추가해야 합니다. 이러한 역할을 추가하는 방법에 관한 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.

이 정책을 사용하여 AWS DMS에 액세스하기 위한 요구 사항에 대한 자세한 내용은 섹션을 참조하세요 [요를 사용하는 데 필요한 IAM 권한 AWS DMS](#).

사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 ID에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 AWS CLI 또는 AWS API를 사용하여 프로그래밍 방식으로 이 작업을 완료할 수 있는 권한이 포함됩니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupForUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    },
    {
      "Sid": "NavigateInConsole",
      "Effect": "Allow",
      "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}
```

```
    ]
}
```

하나의 Amazon S3 버킷에 액세스

AWS DMS는 Amazon S3 버킷을 데이터베이스 마이그레이션을 위한 중간 스토리지로 사용합니다. 일반적으로 AWS DMS는 이러한 목적으로 기본 S3 버킷을 관리합니다. 그러나 경우에 따라 특히 AWS CLI 또는 AWS DMS API를 사용하는 경우 AWS DMS를 사용하면 대신 자체 S3 버킷을 지정할 수 있습니다. 예를 들어, 데이터를 Amazon Redshift 대상 엔드포인트로 마이그레이션하기 위한 자체 S3 버킷을 지정할 수 있습니다. 이 경우 AWS관리형 AmazonDMSRedshiftS3Role 정책에 따라 권한이 있는 역할을 생성해야 합니다.

다음 예제에서는 AmazonDMSRedshiftS3Role 정책의 한 버전을 보여 줍니다. 이를 통해 AWS DMS는 AWS 계정의 IAM 사용자에게 Amazon S3 버킷 중 하나에 대한 액세스 권한을 부여할 수 있습니다. 또한 사용자가 객체를 추가, 업데이트 및 삭제할 수 있게 허용합니다.

이 정책에서는 s3:PutObject, s3:GetObject 및 s3:DeleteObject 권한을 사용자에게 부여할 뿐만 아니라 s3:ListAllMyBuckets, s3:GetBucketLocation 및 s3:ListBucket 권한도 부여합니다. 이러한 권한은 콘솔에 필요한 추가 권한입니다. 다른 권한을 통해 AWS DMS는 버킷 수명 주기를 관리할 수 있습니다. 또한 객체를 복사할 수 있도록 하려면 s3:GetObjectAcl 작업이 필요합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:ListBucket",
        "s3:DeleteBucket",
        "s3:GetBucketLocation",
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject",
        "s3:GetObjectVersion",
        "s3:GetBucketPolicy",
        "s3:PutBucketPolicy",
        "s3:GetBucketAcl",

```

```

        "s3:PutBucketVersioning",
        "s3:GetBucketVersioning",
        "s3:PutLifecycleConfiguration",
        "s3:GetLifecycleConfiguration",
        "s3:DeleteBucketPolicy"
    ],
    "Resource": "arn:aws:s3:::dms-*"
}
]
}

```

이 정책을 기반으로 역할을 생성하는 방법에 관한 자세한 내용은 [Amazon S3 버킷 설정](#) 섹션을 참조하세요.

태그를 기반으로 AWS DMS 리소스 액세스

자격 증명 기반 정책의 조건을 사용하여 태그를 기반으로 AWS DMS 리소스에 대한 액세스를 제어할 수 있습니다. 이 예제에서는 모든 AWS DMS 엔드포인트에 대한 액세스를 허용하는 정책을 생성하는 방법을 보여줍니다. 하지만 권한은 엔드포인트 데이터베이스 태그 Owner가 해당 사용자의 사용자 이름을 값으로 갖는 경우에만 부여됩니다.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "dms:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "dms:endpoint-tag/Owner": "${aws:username}"
        }
      }
    }
  ]
}

```

이 정책을 계정의 IAM 사용자에게 연결할 수 있습니다. 라는 사용자가 AWS DMS 엔드포인트에 액세스하려고 richard-roe 하면 엔드포인트 데이터베이스에 Owner=richard-roe 또는 태그를 지정해야 합니다owner=richard-roe. 그렇지 않으면 이 사용자는 액세스가 거부됩니다. 조건 키 이름은 대소문자를 구분하지 않기 때문에 조건 태그 키 Owner는 Owner 및 owner와 모두 일치합니다. 자세한 내용은 IAM 사용자 설명서의 [IAM JSON 정책 요소: 조건](#)을 참조하세요.

에 대한 리소스 기반 정책 예제 AWS KMS

AWS DMS를 사용하면 사용자 지정 AWS KMS 암호화 키를 생성하여 지원되는 대상 엔드포인트 데이터를 암호화할 수 있습니다. 지원되는 대상 데이터 암호화를 위해 생성하는 암호화 키에 대한 키 정책을 생성하고 연결하는 방법을 알아보려면 [AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화](#) 및 [Amazon S3 대상 객체를 암호화하는 AWS KMS 키 생성](#) 섹션을 참조하세요.

주제

- [Amazon Redshift 대상 데이터를 암호화하기 위한 사용자 지정 암호화 AWS KMS 키에 대한 정책](#)
- [Amazon S3 대상 데이터를 암호화하기 위한 사용자 지정 암호화 AWS KMS 키에 대한 정책](#)

Amazon Redshift 대상 데이터를 암호화하기 위한 사용자 지정 암호화 AWS KMS 키에 대한 정책

다음 예제에서는 Amazon Redshift 대상 데이터를 암호화하기 위해 생성하는 AWS KMS 암호화 키에 대해 생성된 키 정책의 JSON을 보여줍니다.

JSON

```
{
  "Id": "key-consolepolicy-3",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:root"
        ]
      },
      "Action": "kms:*",
      "Resource": "*"
    }
  ]
}
```

```
    },
    {
      "Sid": "Allow access for Key Administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/Admin"
        ]
      },
      "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",
        "kms:Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow use of the key",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/DMS-Redshift-endpoint-access-role"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ],
```

```

{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:iam::987654321098:role/DMS-Redshift-endpoint-access-role"
    ]
  },
  "Action": [
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
  ],
  "Resource": "*",
  "Condition": {
    "Bool": {
      "kms:GrantIsForAWSResource": true
    }
  }
}

```

여기서 키 정책이 키를 생성하기 전에 Amazon Redshift 대상 엔드포인트 데이터에 액세스하기 위해 생성한 역할을 참조하는 위치를 확인할 수 있습니다. 이 예제에서는 DMS-Redshift-endpoint-access-role입니다. 또한 다른 위탁자(사용자 및 역할)에게 허용된 다양한 키 작업도 볼 수 있습니다. 예를 들어 DMS-Redshift-endpoint-access-role이 부여된 모든 사용자는 대상 데이터를 암호화, 해독 및 다시 암호화할 수 있습니다. 이러한 사용자는 외부의 데이터를 암호화하기 위해 내보낼 데이터 키를 생성할 수도 있습니다 AWS KMS. 또한 방금 생성한 AWS KMS 키와 같은 키에 대한 세부 정보를 반환할 수 있습니다. 이 밖에도 이러한 사용자는 대상 엔드포인트와 같은 AWS 리소스에 대한 연결을 관리할 수 있습니다.

Amazon S3 대상 데이터를 암호화하기 위한 사용자 지정 암호화 AWS KMS 키에 대한 정책

다음 예제에서는 Amazon S3 대상 데이터를 암호화하기 위해 생성하는 AWS KMS 암호화 키에 대해 생성된 키 정책의 JSON을 보여줍니다.

JSON

```
{
  "Id": "key-consolepolicy-3",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:root"
        ]
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow access for Key Administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/Admin"
        ]
      },
      "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",
        "kms>Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
      ],
      "Resource": "*"
    }
  ],
}
```



```

    {
      "Sid": "Allow use of the key",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/DMS-S3-endpoint-access-role"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow attachment of persistent resources",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/DMS-S3-endpoint-access-role"
        ]
      },
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": "*",
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
  ]
}

```

여기서 키 정책이 키를 생성하기 전에 Amazon S3 대상 엔드포인트 데이터에 액세스하기 위해 생성한 역할을 참조하는 위치를 확인할 수 있습니다. 이 예제에서는 DMS-S3-endpoint-access-role입니다.

다. 또한 다른 위탁자(사용자 및 역할)에게 허용된 다양한 키 작업도 볼 수 있습니다. 예를 들어 DMS-S3-endpoint-access-role이 부여된 모든 사용자는 대상 데이터를 암호화, 해독 및 다시 암호화할 수 있습니다. 이러한 사용자는 외부의 데이터를 암호화하기 위해 내보낼 데이터 키를 생성할 수도 있습니다 AWS KMS. 또한 방금 생성한 AWS KMS 키와 같은 키에 대한 세부 정보를 반환할 수 있습니다. 이 밖에도 이러한 사용자는 대상 엔드포인트와 같은 AWS 리소스에 대한 연결을 관리할 수 있습니다.

보안 암호를 사용하여 AWS Database Migration Service 엔드포인트 액세스

의 경우 AWS DMS 보안 암호는 보안 암호 인증을 통해 지원되는 AWS DMS 소스 또는 대상 엔드포인트에 대한 데이터베이스 연결을 인증하기 위한 사용자 자격 증명 세트를 나타내는 데 사용할 수 있는 암호화된 키입니다. Oracle Automatic Storage Management(ASM)도 사용하는 Oracle 엔드포인트의 경우는 Oracle ASM에 액세스하려면 사용자 자격 증명을 나타내는 추가 보안 암호가 AWS DMS 필요합니다.

클라우드 및 온프레미스에서 애플리케이션, 서비스 및 IT 리소스에 액세스하기 위한 보안 인증 정보를 안전하게 생성, 저장 및 검색하는 서비스 AWS Secrets Manager를 사용하여 보안 인증에 AWS DMS 필요한 보안 암호 또는 보안 암호를 생성할 수 있습니다. 여기에는 사용자 개입 없이 암호화된 보안 암호 값을 자동으로 주기적으로 교체할 수 있는 지원이 포함되므로 자격 증명에 대한 보안 수준이 더 향상됩니다. AWS Secrets Manager 또한에서 보안 암호 값 교체를 활성화하면 보안 암호에 의존하는 데이터베이스 마이그레이션에 영향을 주지 않고 보안 암호 값 교체가 이루어집니다. 엔드포인트 데이터베이스 연결을 비밀리에 인증하려면 SecretsManagerSecretId에 할당하는 ID 또는 ARN에 대한 보안 암호를 생성하세요. 이 암호는 엔드포인트 설정에 포함해야 합니다. Oracle 엔드포인트의 일부로서 Oracle ASM을 비밀리에 인증하려면 SecretsManagerOracleAsmSecretId에 할당하는 ID 또는 ARN에 대한 보안 암호를 생성하세요. 이 암호는 엔드포인트 설정에도 포함해야 합니다.

Note

Amazon RDS Aurora에서 관리하는 마스터 자격 증명은 사용할 수 없습니다. 이러한 자격 증명에는 연결을 AWS DMS 설정해야 하는 호스트 또는 포트 정보가 포함되지 않습니다. 그 대신에 새 사용자와 보안 암호를 생성합니다. 사용자 및 보안 암호 생성에 관한 자세한 내용은 다음 [AWS Management Console](#)를 사용하여 보안 암호 및 보안 암호 액세스 역할 생성을 참조하세요.

에 대한 자세한 내용은 AWS Secrets Manager 사용 설명서의 [Secrets Manager란 AWS 무엇입니까?](#)를 AWS Secrets Manager참조하세요.

AWS DMS 는 지원되는 소스 및 대상 엔드포인트에서 다음 온프레미스 또는 AWS관리형 데이터베이스에 대한 보안 인증을 지원합니다.

- Amazon DocumentDB
- IBM DB2 LUW
- Microsoft SQL Server
- MongoDB
- MySQL
- Oracle
- PostgreSQL
- Amazon Redshift
- SAP ASE

이러한 데이터베이스에 연결하려면 엔드포인트 설정의 일부로 다음 값 집합 중 하나를 입력할 수 있지만 둘 다 입력할 수는 없습니다.

- Username, Password, ServerName 및 Port 설정을 사용하여 데이터베이스 연결을 인증하기 위한 일반 텍스트 값. Oracle ASM을 함께 사용하는 Oracle 엔드포인트의 경우, AsmUserName, AsmPassword 및 AsmServerName 설정을 사용하여 ASM을 인증하기 위한 추가 일반 텍스트 값을 포함하세요.
- SecretsManagerSecretId 및 SecretsManagerAccessRoleArn 설정 값을 사용하는 보안 암호 인증. Oracle ASM을 사용하는 Oracle 엔드포인트의 경우, SecretsManagerOracleAsmSecretId 및 SecretsManagerOracleAsmAccessRoleArn 설정에 대한 추가 값을 포함하세요. 이러한 설정의 보안 암호 값에는 다음과 같은 항목이 포함될 수 있습니다.
 - SecretsManagerSecretId – AWS Secrets Manager에서 엔드포인트 데이터베이스 액세스를 위해 생성한 보안 암호의 전체 Amazon 리소스 이름(ARN), 부분 ARN 또는 기억하기 쉬운 이름.
 - SecretsManagerAccessRoleArn - 사용자를 대신하여 이 보안 암호에 대한 액세스를 제공하기 위해 IAM에서 생성한 SecretsManagerSecretId 보안 암호 AWS DMS 액세스 역할의 ARN입니다.
 - SecretsManagerOracleAsmSecretId – AWS Secrets Manager에서 Oracle ASM에 액세스하기 위해 생성한 보안 암호의 전체 Amazon 리소스 이름(ARN), 부분 ARN 또는 기억하기 쉬운 이름.
 - SecretsManagerOracleAsmAccessRoleArn – 사용자를 대신하여 이 SecretsManagerOracleAsmSecretId 보안 암호에 대한 AWS DMS 액세스를 제공하기 위해 IAM에서 생성한 보안 암호 액세스 역할의 ARN.

Note

또한 단일 보안 암호 액세스 역할을 사용하여 SecretsManagerSecretId 보안 암호와 SecretsManagerOracleAsmSecretId 보안 암호 모두에 대한 AWS DMS 액세스를 제공할 수 있습니다. 두 보안 암호에 대해 이 단일 보안 암호 액세스 역할을 생성하는 경우, 이 액세스 역할에 대한 동일한 ARN을 SecretsManagerAccessRoleArn 및 SecretsManagerOracleAsmAccessRoleArn에 모두 할당해야 합니다. 예를 들어, 두 보안 암호에 대한 보안 액세스 역할에서 해당 ARN ARN2xsecrets이 변수에 할당된 경우, 다음과 같이 ARN 설정을 지정할 수 있습니다.

```
SecretsManagerAccessRoleArn = ARN2xsecrets;
SecretsManagerOracleAsmAccessRoleArn = ARN2xsecrets;
```

이러한 값을 생성하는 방법에 관한 자세한 내용은 [AWS Management Console 를 사용하여 보안 암호 및 보안 암호 액세스 역할 생성](#) 섹션을 참조하세요.

엔드포인트에 필요한 보안 암호 및 보안 암호 액세스 역할 엔드포인트 설정을 생성하고 지정한 후에는 이러한 보안 암호 세부 정보로 CreateEndpoint 또는 ModifyEndpoint API 요청을 실행할 사용자 계정의 권한을 업데이트하세요. 이러한 계정 권한에 보안 액세스 역할에 대한 IAM:GetRole 권한과 보안 암호에 대한 SecretsManager:DescribeSecret 권한이 포함되어 있는지 확인합니다.는 액세스 역할과 보안 암호를 모두 검증하려면 이러한 권한이 AWS DMS 필요합니다.

필수 사용자 권한을 제공하고 확인하려면

1. 에 로그인 AWS Management Console 하고에서 AWS Identity and Access Management 콘솔을 엽니다<https://console.aws.amazon.com/iam/>.
2. 사용자를 선택한 다음, CreateEndpoint 및 ModifyEndpoint API 직접 호출에 사용되는 사용자 ID를 선택합니다.
3. 권한 탭에서 {} JSON을 선택합니다.
4. 사용자에게 다음과 같은 권한이 있는지 확인하세요.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": [
```

```

    "iam:GetRole",
    "iam:PassRole"
  ],
  "Resource": "SECRET_ACCESS_ROLE_ARN"
},
{
  "Effect": "Allow",
  "Action": "secretsmanager:DescribeSecret",
  "Resource": "SECRET_ARN"
}
]
}

```

5. 사용자에게 해당 권한이 없는 경우 권한을 추가하세요.
6. IAM 역할을 사용하여 DMS API 직접 호출을 하는 경우, 각 역할에 대해 위 단계를 반복하세요.
7. 터미널을 열고 AWS CLI 를 사용하여 위에서 사용한 역할 또는 사용자를 수임하여 권한이 올바르게 부여되었는지 확인합니다.
 - a. IAM `get-role` 명령을 사용하여 `SecretAccessRole`에 대한 사용자 권한을 확인합니다.

```
aws iam get-role --role-name ROLE_NAME
```

*ROLE_NAME*을 `SecretsManagerAccessRole`의 이름으로 바꿉니다.

명령에서 오류 메시지가 반환되는 경우, 권한이 올바르게 제공되었는지 확인하세요.

- b. Secrets Manager `describe-secret` 명령을 사용하여 보안 암호에 대한 사용자 권한을 확인합니다.

```
aws secretsmanager describe-secret --secret-id SECRET_NAME OR SECRET_ARN --
region=REGION_NAME
```

사용자는 기억하기 쉬운 이름, 부분 ARN 또는 전체 ARN일 수 있습니다. 자세한 내용은 [describe-secret](#)을 참조하세요.

명령에서 오류 메시지가 반환되는 경우, 권한이 올바르게 제공되었는지 확인하세요.

AWS Management Console 를 사용하여 보안 암호 및 보안 암호 액세스 역할 생성

AWS Management Console 를 사용하여 엔드포인트 인증을 위한 보안 암호를 생성하고가 사용자를 대신하여 보안 암호에 AWS DMS 액세스할 수 있도록 허용하는 정책 및 역할을 생성할 수 있습니다.

를 사용하여 소스 및 대상 엔드포인트 연결에 대한 데이터베이스를 인증하는 데 사용할 AWS DMS 수 AWS Management Console 있는 보안 암호를 생성하려면

1. 에 로그인 AWS Management Console 하고에서 AWS Secrets Manager 콘솔을 엽니다<https://console.aws.amazon.com/secretsmanager/>.
2. 새 보안 암호 저장을 선택합니다.
3. 새 보안 암호 저장 페이지의 보안 암호 유형 선택에서 다른 유형의 보안 암호를 선택한 다음, 일반 텍스트를 선택합니다.

Note

이 시점부터 엔드포인트 데이터베이스에 연결하기 위해 일반 텍스트 자격 증명 정보를 입력해야 하는 곳은 여기뿐입니다.

4. 일반 텍스트 필드에서:
 - SecretsManagerSecretId에 할당할 ID에 대한 보안 암호의 경우, 다음 JSON 구조를 입력합니다.

```
{
  "username": db_username,
  "password": db_user_password,
  "port": db_port_number,
  "host": db_server_name
}
```

Note

이것은 엔드포인트 데이터베이스를 인증하는 데 필요한 JSON 멤버의 최소 목록입니다. 원하는 모든 추가 JSON 엔드포인트 설정을 모두 소문자로 JSON 멤버로 추가할 수 있습니다. 하지만 AWS DMS 는 엔드포인트 인증을 위한 추가 JSON 멤버를 무시합니다.

여기서 *db_username*은 다음 예제와 같이 데이터베이스에 액세스하는 사용자의 이름이고, *db_user_password*는 데이터베이스 사용자의 비밀번호이며, *db_port_number*는 데이터베이스에 액세스하기 위한 포트 번호이고, *db_server_name*은 웹상의 데이터베이스 서버 이름(주소)입니다.

```
{
  "username": "admin",
  "password": "some_password",
  "port": "8190",
  "host": "oracle101.abcdefghij.us-east-1.rds.amazonaws.com"
}
```

- SecretsManagerOracleAsmSecretId에 할당할 ID에 대한 보안 암호의 경우, 다음 JSON 구조를 입력합니다.

```
{
  "asm_user": asm_username,
  "asm_password": asm_user_password,
  "asm_server": asm_server_name
}
```

Note

이것은 Oracle 엔드포인트에 대한 Oracle ASM을 인증하는 데 필요한 JSON 멤버의 최소 목록입니다. 또한 사용 가능한 Oracle ASM 엔드포인트 설정을 기반으로 지정할 수 있는 전체 목록이기도 합니다.

여기서 *asm_username*은 다음 예제와 같이 Oracle ASM에 액세스하는 사용자의 이름이고, *asm_user_password*는 Oracle ASM 사용자의 비밀번호이며, *asm_server_name*은 포트를 포함한 웹상의 Oracle ASM 서버 이름(주소)입니다.

```
{
  "asm_user": "oracle_asm_user",
  "asm_password": "oracle_asm_password",
  "asm_server": "oracle101.abcdefghij.us-east-1.rds.amazonaws.com:8190/+ASM"
}
```

5. AWS KMS 암호화 키를 선택하여 보안 암호를 암호화합니다. 에서 서비스에 AWS Secrets Manager 대해 생성한 기본 암호화 키를 수락하거나 생성한 AWS KMS 키를 선택할 수 있습니다.
6. 이 보안 암호를 참조할 이름과 설명(선택 사항)을 지정합니다. 이 이름은 `SecretsManagerSecretId` 또는 `SecretsManagerOracleAsmSecretId` 값으로 사용할 수 있는 기억하기 쉬운 이름입니다.
7. 보안 암호에서 자동 교체를 활성화하려면 설명된 대로 보안 암호의 자격 증명을 교체할 수 있는 권한이 있는 AWS Lambda 함수를 선택하거나 생성해야 합니다. 그러나 Lambda 함수를 사용하도록 자동 교체를 설정하기 전에 함수에 대한 구성 설정이 `EXCLUDE_CHARACTERS` 환경 변수 값에 다음 4자를 추가하는지 확인하세요.

```
;.:+[]*&,%\
```

AWS DMS 는 엔드포인트 자격 증명에 사용되는 암호에 이러한 문자를 허용하지 않습니다. 이러한 문자를 제외하도록 Lambda 함수를 구성하면 AWS Secrets Manager 은 이러한 문자를 교체된 비밀번호 값의 일부로 생성할 수 없습니다. Lambda 함수를 사용하도록 자동 교체를 설정하면가 AWS Secrets Manager 즉시 보안 암호를 교체하여 보안 암호 구성을 검증합니다.

Note

데이터베이스 엔진 구성에 따라 데이터베이스는 교체된 자격 증명을 가져오지 못할 수도 있습니다. 이 경우, 자격 증명을 새로 고치려면 작업을 수동으로 다시 시작해야 합니다.

8. 보안 암호를 검토하고에 저장합니다 AWS Secrets Manager. 그런 다음 친숙한 이름으로 각 보안 암호를 조회한 다음 AWS Secrets Manager보안 암호 ARN을 값으로 또는 `SecretsManagerSecretId` `SecretsManagerOracleAsmSecretId` 적절하게 검색하여 엔드포인트 데이터베이스 연결 및 Oracle ASM(사용된 경우)에 대한 액세스를 인증할 수 있습니다.

보안 암호 액세스 정책 및 역할을 생성하여 **`SecretsManagerAccessRoleArn`** 또는를 설정하면가 적절한 보안 암호 AWS DMS 에 액세스할 수 **`SecretsManagerOracleAsmAccessRoleArn`**있는 AWS Secrets Manager 에 액세스할 수 있습니다.

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> AWS Identity and Access Management (IAM) 콘솔을 엽니다.
2. 정책을 선택한 후 정책 생성을 선택합니다.
3. JSON을 선택하고 다음 정책을 입력하여 보안 암호에 대한 액세스 및 암호 해독을 활성화합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

여기서 *secret_arn*은 보안 암호의 ARN(적절하다면 SecretsManagerSecretId 또는 SecretsManagerOracleAsmSecretId에서 가져올 수 있음)이고, *kms_key_arn*은 다음 예제와 같이 보안 암호를 암호화하는 데 사용하는 AWS KMS 키의 ARN입니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
      "Resource": "arn:aws:secretsmanager:us-east-2:123456789012:secret:MySQLTestSecret-qeHamH"
    },
    {

```

```

      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:DescribeKey"
      ],
      "Resource": "arn:aws:kms:us-east-2:123456789012:key/761138dc-0542-4e58-947f-4a3a8458d0fd"
    }
  ]
}

```

Note

에서 생성한 기본 암호화 키를 사용하는 경우 AWS Secrets Manager에 대한 AWS KMS 권한을 지정할 필요가 없습니다 *kms_key_arn*. 정책에서 두 보안 암호에 대한 액세스를 제공하도록 하려면 다른 *secret_arn*에 대해 추가 JSON 리소스 객체를 지정하면 됩니다. 보안 암호가 다른 계정에 있는 경우, SecretsManagerAccessRoleArn 역할에는 계정한 보안 암호를 확인하기 위한 추가 정책이 필요합니다. 이러한 사용 사례의 경우, 정책에 작업 secretsmanager:DescribeSecret을 추가하세요. 교차 계정 보안 암호 설정에 대한 자세한 내용은 [다른 계정의 사용자에게 대한 AWS Secrets Manager 보안 암호에 대한 권한을 참조하세요](#).

- 기억하기 쉬운 이름과 설명(선택 사항)이 있는 정책을 검토하고 생성합니다.
- 역할을 선택한 다음, 역할 생성을 선택합니다.
- 신뢰할 수 있는 엔터티 유형으로서 AWS 서비스를 선택합니다.
- 서비스 목록에서 DMS를 신뢰할 수 있는 서비스로 선택하고 다음: 권한을 선택합니다.
- 4단계에서 생성한 정책을 찾아 연결한 다음, 계속 진행하여 태그를 추가하고 역할을 검토합니다. 이때 AWS DMS 리전 서비스 보안 주체를 신뢰할 수 있는 엔터티로 사용하도록 역할의 신뢰 관계를 편집합니다. 이 위탁자의 형식은 다음과 같습니다.

```
dms.region-name.amazonaws.com
```

여기서 *region-name*은 리전의 이름입니다(예: us-east-1). 따라서이 AWS DMS 리전의 리전 서비스 보안 주체는 다음과 같습니다.

```
dms.us-east-1.amazonaws.com
```

9. 역할의 신뢰할 수 있는 엔터티를 편집한 후 기억하기 쉬운 이름과 선택적 설명을 사용하여 역할을 생성합니다. 이제 IAM에서 기억하기 쉬운 이름으로 새 역할을 검색한 다음, 역할 ARN을 `SecretsManagerAccessRoleArn` 또는 `SecretsManagerOracleAsmAccessRoleArn` 값으로 검색하여 엔드포인트 데이터베이스 연결을 인증할 수 있습니다.

프라이빗 서브넷의 복제 인스턴스와 함께 Secrets Manager를 사용하려면

1. Secrets Manager VPC 엔드포인트를 생성하고 이 엔드포인트의 DNS를 기록해 둡니다. Secrets Manager VPC 엔드포인트를 생성하는 방법에 관한 자세한 내용은 AWS Secrets Manager 사용 설명서의 [VPC 엔드포인트를 통해 Secrets Manager에 연결하기](#)를 참조하세요.
2. VPC 엔드포인트 보안 그룹 수신 규칙의 경우 복제 인스턴스의 프라이빗 IP 주소 또는 복제 인스턴스에 연결된 보안 그룹의 HTTPS 트래픽을 허용합니다.
3. 복제 인스턴스 보안 그룹 송신 규칙의 경우, 대상 `0.0.0.0/0`으로 전송되는 모든 트래픽을 허용하세요.
4. 다음 예제와 같이 Secret Manager VPC 엔드포인트 DNS를 제공하려면 엔드포인트 추가 연결 속성 `secretsManagerEndpointOverride=secretsManager endpoint DNS`를 설정합니다.

```
secretsManagerEndpointOverride=vpce-1234a5678b9012c-12345678.secretsmanager.eu-west-1.vpce.amazonaws.com
```

에 대한 서비스 연결 역할 사용 AWS DMS

AWS Database Migration Service 는 AWS Identity and Access Management (IAM) [서비스 연결 역할](#)을 사용합니다. 서비스 연결 역할은 직접 연결된 고유한 유형의 IAM 역할입니다 AWS DMS. 서비스 연결 역할은에서 사전 정의 AWS DMS 하며 서비스에서 사용자를 대신하여 다른 AWS 서비스를 호출하는 데 필요한 모든 권한을 포함합니다.

필요한 권한을 수동으로 추가할 필요가 없으므로 서비스 연결 역할을 더 AWS DMS 쉽게 설정할 수 있습니다.는 서비스 연결 역할의 권한을 AWS DMS 정의하며, 달리 정의되지 않은 한 만 해당 역할을 수 임 AWS DMS 할 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며, 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제한 후에만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 AWS DMS 리소스에 대한 액세스 권한을 실수로 제거할 수 없기 때문에 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 다른 서비스에 대한 자세한 내용은 [AWS IAM으로 작업하는 서비스를](#) 참조하고 서비스 연결 역할 열에서 예인 서비스를 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예 링크를 선택합니다.

AWS DMS 기능에 대한 서비스 연결 역할

주제

- [AWS DMS Fleet Advisor의 서비스 연결 역할](#)
- [예에 대한 서비스 연결 역할 AWS DMS](#)

AWS DMS Fleet Advisor의 서비스 연결 역할

AWS DMS Fleet Advisor는 AWSServiceRoleForDMSFleetAdvisor라는 서비스 연결 역할을 사용합니다. DMS Fleet Advisor는 이 서비스 연결 역할을 사용하여 Amazon CloudWatch 지표를 관리합니다. 이 서비스 연결 역할은 관리형 정책 AWSDMSFleetAdvisorServiceRolePolicy에 연결됩니다. 이 정책에 대한 업데이트는 [AWS에 대한 관리형 정책 AWS Database Migration Service](#) 섹션을 참조하세요.

AWSServiceRoleForDMSFleetAdvisor 서비스 연결 역할은 역할을 수입하기 위해 다음 서비스를 신뢰합니다.

- `dms-fleet-advisor.amazonaws.com`

AWSDMSFleetAdvisorServiceRolePolicy라는 역할 권한 정책은 AWS DMS Fleet Advisor가 지정된 리소스에서 다음 작업을 완료하도록 허용합니다.

- 작업: `all AWS resources`에 대한 `cloudwatch:PutMetricData`

이 권한을 통해 보안 주체는 지표 데이터 포인트를 Amazon CloudWatch에 게시할 수 있습니다. AWS DMS CloudWatch의 데이터베이스 지표가 포함된 차트를 표시하려면 Fleet Advisor에 이 권한이 필요합니다.

다음 코드 예제는 AWSDMSFleetAdvisorServiceRolePolicy 역할을 생성하는 데 사용하는 AWSDMSFleetAdvisorServiceRolePolicy 정책을 보여줍니다.

JSON

```
{
```

```

    "Version": "2012-10-17",
    "Statement": {
      "Effect": "Allow",
      "Resource": "*",
      "Action": "cloudwatch:PutMetricData",
      "Condition": {
        "StringEquals": {
          "cloudwatch:namespace": "AWS/DMS/FleetAdvisor"
        }
      }
    }
  }
}

```

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 연결 역할을 생성하고 편집하거나 삭제할 수 있도록 허용하는 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 [서비스 연결 역할 권한](#)을 참조하세요.

AWS DMS Fleet Advisor에 대한 서비스 연결 역할 생성

IAM 콘솔을 사용하여 DMS – Fleet Advisor 사용 사례에서 서비스 연결 역할을 생성할 수 있습니다. AWS CLI 또는 AWS API에서 서비스 이름을 사용하여 `dms-fleet-advisor.amazonaws.com` 서비스 연결 역할을 생성합니다. 자세한 내용은 IAM 사용자 설명서의 [서비스 연결 역할 생성](#) 섹션을 참조하세요. 이 서비스 연결 역할을 삭제하면 동일한 프로세스를 사용하여 역할을 다시 생성할 수 있습니다.

데이터 수집기를 생성하기 전에 이 역할을 생성해야 합니다. DMS Fleet Advisor는 이 역할을 사용하여 AWS Management Console에서 데이터베이스 지표가 포함된 차트를 표시합니다. 자세한 내용은 [데이터 수집기 생성](#) 단원을 참조하십시오.

AWS DMS Fleet Advisor에 대한 서비스 연결 역할 편집

AWS DMS에서는 `AWSServiceRoleForDMSFleetAdvisor` 서비스 연결 역할을 편집할 수 없습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 [서비스 연결 역할 편집](#)을 참조하세요.

AWS DMS Fleet Advisor에 대한 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 특성 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 결국 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 개체가 없게 됩니다. 단, 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

Note

리소스를 삭제하려고 할 때 AWS DMS 서비스가 역할을 사용하는 경우 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForDMSFleetAdvisor에서 사용하는 AWS DMS 리소스를 삭제하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 검색 아래에 있는 데이터 수집기를 선택합니다. 데이터 수집기 페이지가 열립니다.
3. 데이터 수집기를 선택하고 삭제를 선택합니다.
4. 그런 다음, 텍스트 입력 필드에 데이터 수집기 이름을 입력합니다. 다음으로 삭제를 선택합니다.

Important

DMS 데이터 수집기를 삭제하면 DMS Fleet Advisor는 이 수집기를 사용하여 검색한 모든 데이터베이스를 인벤토리에서 삭제합니다.

데이터 수집기를 모두 삭제한 후 서비스 연결 역할을 삭제할 수 있습니다.

IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 AWSServiceRoleForDMSFleetAdvisor 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 [서비스 연결 역할 삭제](#)를 참조하십시오.

AWS DMS Fleet Advisor 서비스 연결 역할에 지원되는 리전

AWS DMS Fleet Advisor는 서비스를 사용할 수 있는 모든 리전에서 서비스 연결 역할 사용을 지원합니다. 자세한 내용은 [지원됨 AWS 리전](#) 단원을 참조하십시오.

에 대한 서비스 연결 역할 AWS DMS

AWS DMS 는 AWSServiceRoleForDMSServerless라는 서비스 연결 역할을 사용합니다.이 서비스 연결 역할을 AWS DMS 사용하여 사용자를 대신하여 AWS DMS 리소스를 생성하고 관리합니다.는 자동 인스턴스 관리를 위해이 역할을 AWS DMS 사용하므로 복제만 관리하면 됩니다.

[AWSServiceRoleForDMSServerless](#) 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- `dms.amazonaws.com`

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 연결 역할을 생성하고 편집하거나 삭제할 수 있도록 허용하는 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 [서비스 연결 역할 권한](#)을 참조하세요.

에 대한 서비스 연결 역할 생성 AWS DMS

복제 작업을 시작하거나 사전 마이그레이션 평가를 시작하면 AWS DMS AWS DMS 프로그래밍 방식으로 서비스 연결 역할을 생성합니다. 이 역할은 IAM 콘솔에서 볼 수 있습니다. 이 역할을 수동으로 생성하도록 선택할 수도 있습니다. 이 역할을 수동으로 생성하려면 IAM 콘솔을 사용하여 DMS 사용 사례로 서비스 연결 역할을 생성합니다. AWS CLI 또는 AWS API에서 서비스 이름에 `dms.amazonaws.com`를 사용하여 서비스 연결 역할을 생성합니다. 자세한 내용은 IAM 사용자 설명서의 [서비스 연결 역할 생성](#) 섹션을 참조하세요. 이 서비스 연결 역할을 삭제하면 동일한 프로세스를 사용하여 역할을 다시 생성할 수 있습니다.

Note

계정에 복제가 있는 상태에서 역할을 삭제하면 복제가 실패합니다.

에 대한 서비스 연결 역할 편집 AWS DMS

AWS DMS에서는 [AWSServiceRoleForDMSServerless](#) 서비스 연결 역할을 편집할 수 없습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 [서비스 연결 역할 편집](#)을 참조하세요.

에 대한 서비스 연결 역할 삭제 AWS DMS

서비스 연결 역할이 필요한 특성 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 결국 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 개체가 없게 됩니다. 단, 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

Note

리소스를 삭제하려고 할 때 AWS DMS 서비스가 역할을 사용하는 경우 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForDMSServerless에서 사용하는 AWS DMS 리소스를 삭제하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2> AWS DMS 콘솔을 엽니다.
2. 탐색 창의 데이터 마이그레이션에서 서버리스 복제를 선택합니다. 서버리스 페이지가 열립니다.
3. 서버리스 복제를 선택하고 삭제를 선택합니다.
4. 삭제를 확인하려면 텍스트 입력 필드에 서버리스 복제 이름을 입력합니다. 다음으로 삭제를 선택합니다.

서버리스 복제를 모두 삭제한 후 서비스 연결 역할을 삭제할 수 있습니다.

IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면 다음을 수행하세요.

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 AWSServiceRoleForDMSServerless 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 [서비스 연결 역할 삭제](#)를 참조하십시오.

AWS DMS 서비스 연결 역할에 지원되는 리전

AWS DMS 는 서비스를 사용할 수 있는 모든 리전에서 서비스 연결 역할 사용을 지원합니다.

AWS Database Migration Service 자격 증명 및 액세스 문제 해결

다음 정보를 사용하여 및 IAM으로 작업할 때 발생할 수 있는 일반적인 문제를 진단 AWS DMS 하고 수정할 수 있습니다.

주제

- [에서 작업을 수행할 권한이 없음 AWS DMS](#)
- [iam:PassRole을 수행하도록 인증되지 않음](#)
- [관리자인데 다른 사람에게 액세스하도록 허용하려고 함 AWS DMS](#)
- [내 AWS 계정 외부의 사람이 내 AWS DMS 리소스에 액세스하도록 허용하려고 함](#)

에서 작업을 수행할 권한이 없음 AWS DMS

에서 작업을 수행할 권한이 없다는 AWS Management Console 메시지가 표시되면 관리자에게 문의하여 도움을 받아야 합니다. 관리자는 사용자 이름과 암호를 제공한 사람입니다.

다음 예제 오류는 mateojackson IAM 사용자가 콘솔을 사용하여 AWS DMS 엔드포인트에 대한 세부 정보를 보려고 하지만 `dms: DescribeEndpoint` 권한이 없는 경우에 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
dms:DescribeEndpoint on resource: my-postgresql-target
```

이 경우 Mateo는 `dms:DescribeEndpoint` 작업을 사용하여 `my-postgresql-target` 엔드포인트 리소스에 액세스하도록 허용하는 정책을 업데이트할 것을 관리자에게 요청합니다.

iam:PassRole을 수행하도록 인증되지 않음

`iam:PassRole` 작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 AWS DMS에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

일부 AWS 서비스에서는 새 서비스 역할 또는 서비스 연결 역할을 생성하는 대신 기존 역할을 해당 서비스에 전달할 수 있습니다. 이렇게 하려면 역할을 서비스에 전달할 권한이 있어야 합니다.

다음 예 오류는 marymajor라는 IAM 사용자가 콘솔을 사용하여 AWS DMS에서 작업을 수행하려고 하는 경우에 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 권한이 없습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

이 경우, Mary가 `iam:PassRole` 작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

관리자인데 다른 사람 이에 액세스하도록 허용하려고 함 AWS DMS

다른 사용자에게 액세스를 허용하려면 액세스 권한이 필요한 사용자 또는 애플리케이션에 권한을 부여해야 AWS DMS합니다. AWS IAM Identity Center 를 사용하여 사용자 및 애플리케이션을 관리하는 경우 사용자 또는 그룹에 권한 세트를 할당하여 액세스 수준을 정의합니다. 권한 세트는 IAM 정책을 자동으로 생성하고 사용자 또는 애플리케이션과 연결된 IAM 역할에 할당합니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서에서 [권한 세트](#)를 참조하세요.

IAM Identity Center를 사용하지 않는 경우 액세스가 필요한 사용자 또는 애플리케이션에 대한 IAM 엔티티(사용자 또는 역할)를 생성해야 합니다. 그런 다음 AWS DMS에 대한 올바른 권한을 부여하는 정책을 엔티티에 연결해야 합니다. 권한이 부여되면 사용자 또는 애플리케이션 개발자에게 자격 증명을 제공합니다. 이들은 이 자격 증명을 사용하여 AWS에 액세스합니다. IAM 사용자, 그룹, 정책 및 권한 생성에 대해 자세히 알아보려면 IAM 사용자 설명서의 [IAM 자격 증명](#)과 [IAM의 권한 및 정책](#)을 참조하세요.

내 AWS 계정 외부의 사람이 내 AWS DMS 리소스에 액세스하도록 허용하려고 함

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACL)을 지원하는 서비스의 경우, 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세한 내용은 다음을 참조하세요.

- 에서 이러한 기능을 AWS DMS 지원하는지 여부를 알아보려면 섹션을 참조하세요 [AWS Database Migration Service](#) 에서 IAM을 사용하는 방법.
- 소유 AWS 계정 한의 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 [IAM 사용 설명서의 소유한 다른의 IAM 사용자에게 액세스 권한 제공을 참조 AWS 계정 하세요](#).
- 타사에 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 [타사가 AWS 계정 소유한에 대한 액세스 권한 제공을](#) AWS 계정참조하세요.
- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 [외부에서 인증된 사용자에게 액세스 권한 제공\(ID 페더레이션\)](#)을 참조하세요.
- 크로스 계정 액세스에 대한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 [IAM의 크로스 계정 리소스 액세스](#)를 참조하세요.

를 사용하는 데 필요한 IAM 권한 AWS DMS

AWS DMS를 사용하려면 특정 IAM 권한 및 IAM 역할을 사용합니다. IAM 사용자로 로그인하고를 사용하려면 AWS DMS계정 관리자가이 섹션에서 설명한 정책을 실행하는 데 사용하는 IAM 사용자, 그룹 또는 역할에 연결해야 합니다 AWS DMS. IAM 권한에 관한 자세한 내용은 [IAM 사용자 설명서](#)를 참조하세요.

다음 정책은 AWS KMS IAM AWS DMS, Amazon EC2 및 Amazon CloudWatch와 같은 다른 Amazon 서비스에 필요한 특정 작업에 대한 액세스 권한과 권한을 부여합니다. CloudWatch는 마이그레이션을 실시간으로 모니터링하고 AWS DMS 마이그레이션 진행 상황을 나타내는 지표를 수집하고 추적합니다. CloudWatch Logs를 사용하여 작업에 발생한 문제를 디버깅할 수 있습니다.

Note

태그 지정을 사용하여 AWS DMS 리소스에 대한 액세스를 추가로 제한할 수 있습니다. 태그 지정을 사용하여 AWS DMS 리소스에 대한 액세스를 제한하는 방법에 대한 자세한 내용은 섹션을 참조하세요 [리소스 이름 및 태그를 사용하여 세분화된 액세스 제어](#).

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "dms:*",
      "Resource": "arn:aws:dms:*:123456789012:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:ListAliases",
        "kms:DescribeKey"
      ],
      "Resource": "arn:aws:kms:*:123456789012:key/*"
    },
    {
```

```
"Effect": "Allow",
"Action": [
    "iam:GetRole",
    "iam:PassRole",
    "iam:CreateRole",
    "iam:AttachRolePolicy"
],
"Resource": "arn:aws:iam::123456789012:role/*",
"Condition": {
    "StringEquals": {
        "iam:PassedToService": "dms.amazonaws.com"
    }
}
},
{
    "Effect": "Allow",
    "Action": [
        "ec2:DescribeVpcs",
        "ec2:DescribeInternetGateways",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:ModifyNetworkInterfaceAttribute",
        "ec2:CreateNetworkInterface",
        "ec2>DeleteNetworkInterface"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "cloudwatch:Get*",
        "cloudwatch:List*"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams",
        "logs:FilterLogEvents",
        "logs:GetLogEvents"
    ],
    "Resource": "*"
}
```

```

    "Resource": "arn:aws:logs:*:123456789012:*"
  }
]
}

```

이러한 권한을 분석하면 각 권한이 필요한 이유를 더 잘 이해할 수 있습니다.

다음 섹션은 사용자가 AWS DMS API 작업을 호출하도록 허용하는 데 필요합니다.

```

{
    "Effect": "Allow",
    "Action": "dms:*",
    "Resource": "arn:aws:dms:region:account:resourcetype/id"
}

```

다음 섹션은 사용자가 콘솔에 표시할 수 있는 AWS KMS 키와 별칭을 나열할 수 있도록 허용하는 데 필요합니다. KMS 키의 Amazon 리소스 이름(ARN)을 알고 있고 AWS Command Line Interface ()만 사용하는 경우에는 이 항목이 필요하지 않습니다 AWS CLI.

```

{
    "Effect": "Allow",
    "Action": [
        "kms:ListAliases",
        "kms:DescribeKey"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}

```

엔드포인트를 사용하여 IAM 역할 ARN을 전달해야 하는 특정 엔드포인트 유형에는 다음 섹션이 필요합니다. 또한 필요한 AWS DMS 역할을 미리 생성하지 않으면 AWS DMS 콘솔에서 역할을 생성할 수 있습니다. 모든 역할이 사전에 구성된 경우, iam:GetRole 및 iam:PassRole만 있으면 됩니다. 역할에 관한 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.

```

{
    "Effect": "Allow",
    "Action": [
        "iam:GetRole",
        "iam:PassRole",
        "iam:CreateRole",

```

```

        "iam:AttachRolePolicy"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}

```

다음 섹션은 Amazon EC2 인스턴스를 생성하고 생성된 복제 인스턴스에 대한 네트워크를 구성 AWS DMS 해야 하기 때문에 필요합니다. 이러한 리소스는 고객의 계정에 있으므로 고객 대신에 이러한 작업을 수행하는 기능이 필요합니다.

```

{
    "Effect": "Allow",
    "Action": [
        "ec2:DescribeVpcs",
        "ec2:DescribeInternetGateways",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:ModifyNetworkInterfaceAttribute",
        "ec2:CreateNetworkInterface",
        "ec2>DeleteNetworkInterface"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}

```

다음 섹션에서는 사용자가 복제 인스턴스 지표를 볼 수 있도록 허용해야 합니다.

```

{
    "Effect": "Allow",
    "Action": [
        "cloudwatch:Get*",
        "cloudwatch:List*"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}

```

이 섹션에서는 사용자가 복제 로그를 볼 수 있도록 허용해야 합니다.

```

{
    "Effect": "Allow",
    "Action": [
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams",

```

```
        "logs:FilterLogEvents",
        "logs:GetLogEvents"
    ],
    "Resource": "arn:aws:service:region:account:resourcetype/id"
}
```

마이그레이션에 AWS DMS 콘솔, AWS Command Line Interface (AWS CLI) 또는 AWS DMS API를 사용하는 경우 계정에 여러 역할을 추가해야 합니다. 이러한 역할을 추가하는 방법에 관한 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.

와 함께 사용할 IAM 역할 생성 AWS DMS

데이터베이스 마이그레이션에 AWS DMS 콘솔, AWS CLI 또는 AWS DMS API를 사용하는 경우의 기능을 사용하려면 AWS 먼저 계정에 3개의 IAM 역할을 추가해야 합니다 AWS DMS. 이 중 두 가지는 `dms-vpc-role`과 `dms-cloudwatch-logs-role`입니다. Amazon Redshift를 대상 데이터베이스로 사용하는 경우 `dms-access-for-endpoint` AWS 계정에 IAM 역할도 추가해야 합니다.

관리형 정책은 자동으로 업데이트됩니다. IAM 역할에서 사용자 지정 정책을 사용하는 경우, 이 설명서의 관리형 정책 업데이트를 정기적으로 확인해야 합니다. 관리형 정책의 세부 정보를 보려면 `get-policy`와 `get-policy-version` 명령을 조합하여 사용하면 됩니다.

예를 들어 다음 `get-policy` 명령은 지정된 IAM 역할에 대한 정보를 검색합니다.

```
aws iam get-policy --policy-arn arn:aws:iam::aws:policy/service-role/AmazonDMSVPCManagementRole
```

이 명령에서 반환되는 정보는 다음과 같습니다.

```
{
  "Policy": {
    "PolicyName": "AmazonDMSVPCManagementRole",
    "PolicyId": "ANPAJHKIGMBQI4AEFFSY0",
    "Arn": "arn:aws:iam::aws:policy/service-role/AmazonDMSVPCManagementRole",
    "Path": "/service-role/",
    "DefaultVersionId": "v4",
    "AttachmentCount": 1,
    "PermissionsBoundaryUsageCount": 0,
    "IsAttachable": true,
    "Description": "Provides access to manage VPC settings for AWS managed customer configurations",
    "CreateDate": "2015-11-18T16:33:19+00:00",
    "UpdateDate": "2024-07-25T15:19:01+00:00",
    "Tags": []
  }
}
```

다음 `get-policy-version` 명령은 IAM 정책 정보를 검색합니다.


```
aws iam get-policy-version --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonDMSVPCManagementRole --version-id v4
```

이 명령에서 반환되는 정보는 다음과 같습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DeleteNetworkInterface",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeInternetGateways",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcs",
        "ec2:ModifyNetworkInterfaceAttribute"
      ],
      "Resource": "*"
    }
  ]
}
```

동일한 명령을 사용하여 AmazonDMSCloudWatchLogsRole 및 AmazonDMSRedshiftS3Role 관리형 정책에 관한 정보를 가져올 수 있습니다.

다음 절차에 따라 dms-vpc-role, dms-cloudwatch-logs-role, dms-access-for-endpoint IAM 역할을 생성합니다.

AWS CLI 또는 AWS DMS API와 함께 사용할 dms-vpc-role IAM 역할을 생성하려면

1. 다음과 같은 IAM 정책이 있는 JSON 파일을 생성합니다. JSON 파일의 이름을 `dmsAssumeRolePolicyDocument.json`으로 지정합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

다음 명령을 사용하여 AWS CLI 사용하여 역할을 생성합니다.

```
aws iam create-role --role-name dms-vpc-role --assume-role-policy-document file://
dmsAssumeRolePolicyDocument.json
```

2. 다음 명령을 사용하여 AmazonDMSVPCManagementRole 정책을 dms-vpc-role에 연결합니다.

```
aws iam attach-role-policy --role-name dms-vpc-role --policy-arn
arn:aws:iam::aws:policy/service-role/AmazonDMSVPCManagementRole
```

AWS CLI 또는 AWS DMS API와 함께 사용할 dms-cloudwatch-logs-role IAM 역할을 생성하려면

1. 다음과 같은 IAM 정책이 있는 JSON 파일을 생성합니다. JSON 파일의 이름을 dmsAssumeRolePolicyDocument2.json으로 지정합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

다음 명령을 사용하여 AWS CLI 사용하여 역할을 생성합니다.

```
aws iam create-role --role-name dms-cloudwatch-logs-role --assume-role-policy-document file://dmsAssumeRolePolicyDocument2.json
```

2. 다음 명령을 사용하여 AmazonDMSCloudWatchLogsRole 정책을 dms-cloudwatch-logs-role에 연결합니다.

```
aws iam attach-role-policy --role-name dms-cloudwatch-logs-role --policy-arn arn:aws:iam::aws:policy/service-role/AmazonDMSCloudWatchLogsRole
```

대상 데이터베이스로서 Amazon Redshift를 사용하는 경우, IAM 역할 `dms-access-for-endpoint`를 생성하여 Amazon S3에 대한 액세스 권한을 부여해야 합니다.

Amazon Redshift를 대상 데이터베이스로 사용하는 `dms-access-for-endpoint` IAM 역할을 생성하려면

1. 다음과 같은 IAM 정책이 있는 JSON 파일을 생성합니다. JSON 파일의 이름을 `dmsAssumeRolePolicyDocument3.json`으로 지정합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    },
    {
      "Sid": "2",
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. 다음 명령을 사용하여 AWS CLI 사용하여 역할을 생성합니다.

```
aws iam create-role --role-name dms-access-for-endpoint --assume-role-policy-document file://dmsAssumeRolePolicyDocument3.json
```

3. 다음 명령을 사용하여 AmazonDMSRedshiftS3Role 정책을 dms-access-for-endpoint 역할에 연결합니다.

```
aws iam attach-role-policy --role-name dms-access-for-endpoint \  
    --policy-arn arn:aws:iam::aws:policy/service-role/AmazonDMSRedshiftS3Role
```

이제 AWS CLI 또는 AWS DMS API를 사용하기 위한 IAM 정책이 마련되어 있어야 합니다.

교차 서비스 혼동된 대리인 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS교차 서비스 가장은 혼동된 대리자 문제를 초래할 수 있습니다. 교차 서비스 가장은 한 서비스(직접 호출하는 서비스)가 다른 서비스(직접 호출되는 서비스)를 직접 호출할 때 발생할 수 있습니다. 직접 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 위탁자를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

리소스 정책에서 [aws:SourceArn](#) 및 [aws:SourceAccount](#) 전역 조건 컨텍스트 키를 사용하여 리소스에 다른 서비스를 AWS Database Migration Service 제공하는 권한을 제한하는 것이 좋습니다. 만약 [aws:SourceArn](#) 값에 AWS DMS 복제 인스턴스 이름(ARN)과 같은 계정 ID가 포함되어 있지 않은 경우 권한을 제한하려면 두 전역 조건 컨텍스트 키를 모두 사용해야 합니다. 두 전역 조건 컨텍스트 키와 계정을 포함한 [aws:SourceArn](#) 값을 모두 사용하는 경우, [aws:SourceAccount](#) 값 및 [aws:SourceArn](#) 값의 계정은 동일한 정책 명령문에서 사용할 경우 반드시 동일한 계정 ID를 사용해야 합니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 [aws:SourceArn](#)을 사용하세요. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 [aws:SourceAccount](#)를 사용하세요.

AWS DMS 는 3.4.7 버전 이상부터 혼동된 대리자 옵션을 지원합니다. 자세한 내용은 [AWS Database Migration Service 3.4.7 릴리스 정보](#) 단원을 참조하십시오. 복제 인스턴스가 AWS DMS 버전 3.4.6 이하를 사용하는 경우 혼동된 대리자 옵션을 설정하기 전에 최신 버전으로 업그레이드해야 합니다.

혼동된 대리인 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 [aws:SourceArn](#)글로벌 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN을 모르거나 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드 문자(*)를 포함한 [aws:SourceArn](#) 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들어 `arn:aws:dms:*:123456789012:rep:*`입니다.

주제

- [교차 서비스 혼동된 대리자 방지를 위해 AWS DMS API와 함께 사용할 IAM 역할](#)
- [교차 서비스 혼동된 대리자 방지를 위해 Amazon S3에 사전 평가를 저장하기 위한 IAM 정책](#)
- [교차 서비스 혼동된 대리자 방지를 AWS DMS 위해에서 Amazon DynamoDB를 대상 엔드포인트로 사용](#)

교차 서비스 혼동된 대리자 방지를 위해 AWS DMS API와 함께 사용할 IAM 역할

데이터베이스 마이그레이션에 AWS CLI 또는 AWS DMS API를 사용하려면의 기능을 사용하기 전에 AWS 계정에 `dms-vpc-role` 및 `dms-cloudwatch-logs-role` IAM 역할을 추가해야 합니다 AWS DMS. 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 단원을 참조하십시오.

다음 예는 `my-replication-instance` 복제 인스턴스에서 `dms-vpc-role` 역할을 사용하기 위한 정책을 보여줍니다. 혼동된 대리자 문제를 방지하려면 이들 정책을 사용하세요.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDMSAssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "111122223333"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:dms:*:123456789012:*"
        }
      }
    }
  ]
}
```

교차 서비스 혼동된 대리자 방지를 위해 Amazon S3에 사전 평가를 저장하기 위한 IAM 정책

사전 평가 결과를 S3 버킷에 저장하려면 AWS DMS가 Amazon S3의 객체를 관리하도록 허용하는 IAM 정책을 생성합니다. 자세한 내용은 [IAM 리소스 생성](#) 단원을 참조하십시오.

다음 예제는 지정된 사용자 계정에서 모든 작업 및 평가 실행에 AWS DMS 액세스할 수 있도록 허용하는 IAM 역할에 설정된 혼동된 대리자 조건이 있는 신뢰 정책을 보여줍니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDMSAssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "111122223333"
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:dms:*:123456789012:assessment-run:*",
            "arn:aws:dms:*:123456789012:task:*"
          ]
        }
      }
    }
  ]
}
```

교차 서비스 혼동된 대리자 방지를 AWS DMS 위해에서 Amazon DynamoDB를 대상 엔드포인트로 사용

Amazon DynamoDB를 데이터베이스 마이그레이션의 대상 엔드포인트로 사용하려면 DynamoDB 테이블에 대한 액세스 권한을 AWS DMS 수임하고 부여하도록 허용하는 IAM 역할을 생성해야 합니다. 그런 다음 이 역할을 사용하여 AWS DMS에서 대상 DynamoDB 엔드포인트를 생성합니다. 자세한 내용은 [대상으로 Amazon DynamoDB 사용](#) 단원을 참조하십시오.

다음 예제는 모든 AWS DMS 엔드포인트가 DynamoDB 테이블에 액세스하도록 허용하는 IAM 역할에 설정된 혼동된 대리자 조건이 있는 신뢰 정책을 보여줍니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDMSAssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "111122223333"
        }
      },
      "ArnLike": {
        "aws:SourceArn": [
          "arn:aws:dms:*:123456789012:assessment-run:*",
          "arn:aws:dms:*:123456789012:task:*"
        ]
      }
    }
  ]
}
```

AWS 에 대한 관리형 정책 AWS Database Migration Service

주제

- [AWS 관리형 정책: AmazonDMSVPCManagementRole](#)
- [AWS 관리형 정책: AWSDMSServerlessServiceRolePolicy](#)
- [AWS 관리형 정책: AmazonDMSCloudWatchLogsRole](#)
- [AWS 관리형 정책: AWSDMSFleetAdvisorServiceRolePolicy](#)
- [AWS 관리형 정책: AmazonDMSRedshiftS3Role](#)
- [AWS DMS AWS 관리형 정책에 대한 업데이트](#)

AWS 관리형 정책: AmazonDMSVPCManagementRole

이 정책은 `dms-vpc-role` 역할에 연결되므로가 사용자를 대신하여 작업을 AWS DMS 수행할 수 있습니다.

이 정책은 AWS DMS 가 네트워크 리소스를 관리할 수 있는 권한을 기여자에게 부여합니다.

권한 세부 정보

이 정책에는 다음 옵션이 포함됩니다.

- `ec2:CreateNetworkInterface` - 네트워크 인터페이스를 생성하려면이 권한이 AWS DMS 필요합니다. 이러한 인터페이스는 AWS DMS 복제 인스턴스가 소스 및 대상 데이터베이스에 연결하는데 필수적입니다.
- `ec2:DeleteNetworkInterface` - 더 이상 필요하지 않은 네트워크 인터페이스를 정리하려면이 권한이 AWS DMS 필요합니다. 이렇게 하면 리소스 관리와 불필요한 비용 방지에 도움이 됩니다.
- `ec2:DescribeAvailabilityZones` - 이 권한을 통해 AWS DMS 는 리전의 가용 영역에 대한 정보를 검색할 수 있습니다. AWS DMS 는 이 정보를 사용하여 중복 및 가용성을 위해 올바른 영역에 리소스를 프로비저닝합니다.
- `ec2:DescribeDhcpOptions` - 지정된 VPC에 대한 DHCP 옵션 세트 세부 정보를 AWS DMS 검색합니다. 이 정보는 복제 인스턴스에 대해 네트워킹을 올바르게 구성하는 데 필요합니다.
- `ec2:DescribeInternetGateways` - VPC에 구성된 인터넷 게이트웨이를 이해하려면이 권한이 AWS DMS 필요할 수 있습니다. 복제 인스턴스 또는 데이터베이스에 인터넷 액세스가 필요한 경우 이 정보가 중요합니다.
- `ec2:DescribeNetworkInterfaces` - VPC 내의 기존 네트워크 인터페이스에 대한 정보를 AWS DMS 검색합니다. 이 정보는가 네트워크 인터페이스를 올바르게 구성하고 마이그레이션 프로세스에 적절한 네트워크 연결을 보장하는 AWS DMS 데 필요합니다.
- `ec2:DescribeSecurityGroups` - 보안 그룹은 인스턴스 및 리소스에 대한 인바운드 및 아웃바운드 트래픽을 제어 AWS DMS 합니다. 네트워크 인터페이스를 올바르게 구성하고 복제 인스턴스와 데이터베이스 간의 적절한 통신을 보장하기 위해 보안 그룹을 설명해야 합니다.
- `ec2:DescribeSubnets` -이 권한을 통해서는 VPC의 서브넷을 나열 AWS DMS 할 수 있습니다.이 정보를 AWS DMS 사용하여 적절한 서브넷에서 복제 인스턴스를 시작하여 필요한 네트워크 연결을 보장합니다.
- `ec2:DescribeVpcs` - VPC를 설명하는 것은 AWS DMS 가 복제 인스턴스와 데이터베이스가 있는 네트워크 환경을 이해하는 데 필수적입니다. 여기에는 CIDR 블록 및 기타 VPC별 구성에 대한 지식이 포함됩니다.

- `ec2:ModifyNetworkInterfaceAttribute` -이 권한은가 관리하는 네트워크 인터페이스의 속성을 수정하는 AWS DMS 데 필요합니다. 여기에는 연결 및 보안을 보장하기 위한 설정 조정이 포함될 수 있습니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Statement1",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2>DeleteNetworkInterface",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeInternetGateways",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcs",
        "ec2:ModifyNetworkInterfaceAttribute"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS 관리형 정책: AWSDMSServerlessServiceRolePolicy

이 정책은 `AWSServiceRoleForDMSServerless` 역할에 연결되므로가 사용자를 대신하여 작업을 AWS DMS 수행할 수 있습니다. 자세한 내용은 [에 대한 서비스 연결 역할 AWS DMS](#) 단원을 참조하십시오.

이 정책은가 복제 리소스를 관리할 수 있는 권한을 기여 AWS DMS 자에게 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- AWS DMS - 보안 주체가 AWS DMS 리소스와 상호 작용할 수 있도록 허용합니다.
- Amazon S3 - DMS가 S3 버킷을 생성하여 마이그레이션 전 평가를 저장할 수 있도록 허용합니다. S3 버킷은 리전당 한 명의 사용자에게 대해 생성되며 해당 버킷 정책은 서비스의 서비스 역할로만 액세스를 제한합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "id0",
      "Effect": "Allow",
      "Action": [
        "dms:CreateReplicationInstance",
        "dms:CreateReplicationTask"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "dms:req-tag/ResourceCreatedBy": "DMSServerless"
        }
      }
    },
    {
      "Sid": "id1",
      "Effect": "Allow",
      "Action": [
        "dms:DescribeReplicationInstances",
        "dms:DescribeReplicationTasks"
      ],
      "Resource": "*"
    },
    {
      "Sid": "id2",
      "Effect": "Allow",
      "Action": [
        "dms:StartReplicationTask",
        "dms:StopReplicationTask",
        "dms:ModifyReplicationTask",
        "dms>DeleteReplicationTask",
        "dms:ModifyReplicationInstance",
        "dms>DeleteReplicationInstance"
      ],
    },
  ]
}
```

```

        "Resource": [
            "arn:aws:dms:*:*:rep:*",
            "arn:aws:dms:*:*:task:*"
        ],
        "Condition": {
            "StringEqualsIgnoreCase": {
                "aws:ResourceTag/ResourceCreatedBy": "DMSServerless"
            }
        }
    },
    {
        "Sid": "id3",
        "Effect": "Allow",
        "Action": [
            "dms:TestConnection",
            "dms>DeleteConnection"
        ],
        "Resource": [
            "arn:aws:dms:*:*:rep:*",
            "arn:aws:dms:*:*:endpoint:*"
        ]
    },
    {
        "Sid": "id4",
        "Effect": "Allow",
        "Action": [
            "s3:PutObject",
            "s3>DeleteObject",
            "s3:GetObject",
            "s3:PutObjectTagging"
        ],
        "Resource": [
            "arn:aws:s3:::dms-serverless-premigration-results-*",
            "arn:aws:s3:::dms-premigration-results-*"
        ],
        "Condition": {
            "StringEquals": {
                "s3:ResourceAccount": "${aws:PrincipalAccount}"
            }
        }
    },
    {
        "Sid": "id5",
        "Effect": "Allow",

```

```

    "Action": [
      "s3:PutBucketPolicy",
      "s3:ListBucket",
      "s3:GetBucketLocation",
      "s3:CreateBucket"
    ],
    "Resource": [
      "arn:aws:s3:::dms-serverless-premigration-results-*",
      "arn:aws:s3:::dms-premigration-results-*"
    ],
    "Condition": {
      "StringEquals": {
        "s3:ResourceAccount": "${aws:PrincipalAccount}"
      }
    }
  },
  {
    "Sid": "id6",
    "Effect": "Allow",
    "Action": [
      "dms:StartReplicationTaskAssessmentRun"
    ],
    "Resource": [
      "arn:aws:dms:*:*:task:*"
    ],
    "Condition": {
      "StringEquals": {
        "aws:ResourceAccount": "${aws:PrincipalAccount}"
      }
    }
  }
}
]
}

```

AWS 관리형 정책: AmazonDMSCloudWatchLogsRole

이 정책은 dms-cloudwatch-logs-role 역할에 연결되므로가 사용자를 대신하여 작업을 AWS DMS 수행할 수 있습니다. 자세한 내용은 [에 대한 서비스 연결 역할 사용 AWS DMS](#) 단원을 참조하십시오.

이 정책은가 복제 로그 AWS DMS 를 CloudWatch 로그에 게시할 수 있는 권한을 기여자에게 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- logs – 위탁자가 로그를 CloudWatch Logs에 게시하도록 허용합니다. 가 CloudWatch를 AWS DMS 사용하여 복제 로그를 표시하려면 이 권한이 필요합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDescribeOnAllLogGroups",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogGroups"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "AllowDescribeOfAllLogStreamsOnDmsTasksLogGroup",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogStreams"
      ],
      "Resource": [
        "arn:aws:logs:*:*:log-group:dms-tasks-*",
        "arn:aws:logs:*:*:log-group:dms-serverless-replication-*"
      ]
    },
    {
      "Sid": "AllowCreationOfDmsLogGroups",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup"
      ],
      "Resource": [
```

```

        "arn:aws:logs:*:*:log-group:dms-tasks-*",
        "arn:aws:logs:*:*:log-group:dms-serverless-replication-*:log-
stream:"
    ],
    },
    {
        "Sid": "AllowCreationOfDmsLogStream",
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream"
        ],
        "Resource": [
            "arn:aws:logs:*:*:log-group:dms-tasks-*:log-stream:dms-task-*",
            "arn:aws:logs:*:*:log-group:dms-serverless-replication-*:log-
stream:dms-serverless-*"
        ]
    },
    {
        "Sid": "AllowUploadOfLogEventsToDmsLogStream",
        "Effect": "Allow",
        "Action": [
            "logs:PutLogEvents"
        ],
        "Resource": [
            "arn:aws:logs:*:*:log-group:dms-tasks-*:log-stream:dms-task-*",
            "arn:aws:logs:*:*:log-group:dms-serverless-replication-*:log-
stream:dms-serverless-*"
        ]
    }
]
}

```

AWS 관리형 정책: AWS DMS Fleet Advisor Service Role Policy

AWS DMS Fleet Advisor Service Role Policy를 IAM 엔터티에 연결할 수 없습니다. 이 정책은 AWS DMS Fleet Advisor가 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 [에 대한 서비스 연결 역할 사용 AWS DMS](#) 단원을 참조하십시오.

이 정책은 AWS DMS Fleet Advisor가 Amazon CloudWatch 지표를 게시할 수 있는 기고자 권한을 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `cloudwatch` – 위탁자가 Amazon CloudWatch에 지표 데이터 포인트를 게시하도록 허용합니다. 이 권한은 AWS DMS Fleet Advisor가 CloudWatch를 사용하여 데이터베이스 지표가 포함된 차트를 표시할 수 있도록 하기 위해 필요합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Resource": "*",
    "Action": "cloudwatch:PutMetricData",
    "Condition": {
      "StringEquals": {
        "cloudwatch:namespace": "AWS/DMS/FleetAdvisor"
      }
    }
  }
}
```

AWS 관리형 정책: AmazonDMSRedshiftS3Role

이 정책은 Redshift 엔드포인트 AWS DMS에 대한 S3 설정을 관리할 수 있는 권한을 제공합니다.

권한 세부 정보

이 정책에는 다음 옵션이 포함됩니다.

- `s3:CreateBucket` - DMS가 "dms-" 접두사가 있는 S3 버킷을 생성할 수 있도록 허용
- `s3:ListBucket` - DMS가 "dms-" 접두사가 있는 S3 버킷의 콘텐츠를 나열할 수 있도록 허용
- `s3:DeleteBucket` - DMS가 "dms-" 접두사가 있는 S3 버킷을 삭제하도록 허용
- `s3:GetBucketLocation` - DMS가 S3 버킷이 위치한 리전을 검색할 수 있도록 허용

- s3:GetObject - DMS가 "dms-" 접두사가 있는 S3 버킷에서 객체를 검색할 수 있도록 허용
- s3:PutObject - DMS가 "dms-" 접두사를 사용하여 S3 버킷에 객체를 추가할 수 있도록 허용
- s3:DeleteObject - DMS가 "dms-" 접두사가 있는 S3 버킷에서 객체를 삭제할 수 있도록 허용
- s3:GetObjectVersion - DMS가 버전이 지정된 버킷에 있는 객체의 특정 버전을 검색할 수 있도록 허용
- s3:GetBucketPolicy - DMS가 버킷 정책을 검색할 수 있도록 허용
- s3:PutBucketPolicy - DMS가 버킷 정책을 생성하거나 업데이트하도록 허용
- s3:GetBucketAcl - DMS가 버킷 액세스 제어 목록(ACL) 허용
- s3:PutBucketVersioning - DMS가 버킷에서 버전 관리를 활성화하거나 일시 중지하도록 허용
- s3:GetBucketVersioning - DMS가 버킷의 버전 관리 상태를 검색할 수 있도록 허용
- s3:PutLifecycleConfiguration - DMS가 버킷에 대한 수명 주기 규칙을 생성하거나 업데이트하도록 허용
- s3:GetLifecycleConfiguration - DMS가 버킷에 대해 구성된 수명 주기 규칙을 검색할 수 있도록 허용
- s3:DeleteBucketPolicy - DMS가 버킷 정책을 삭제하도록 허용

이러한 모든 권한은 `arn:aws:s3:::dms-*` ARN 패턴이 있는 리소스에만 적용됩니다

JSON 정책 문서

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:ListBucket",
        "s3:DeleteBucket",
        "s3:GetBucketLocation",
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject",
        "s3:GetObjectVersion",
        "s3:GetBucketPolicy",

```

```

        "s3:PutBucketPolicy",
        "s3:GetBucketAcl",
        "s3:PutBucketVersioning",
        "s3:GetBucketVersioning",
        "s3:PutLifecycleConfiguration",
        "s3:GetLifecycleConfiguration",
        "s3:DeleteBucketPolicy"
    ],
    "Resource": "arn:aws:s3:::dms-*"
}
]
}

```

AWS DMS AWS 관리형 정책에 대한 업데이트

이 서비스가 이러한 변경 사항을 추적하기 시작한 AWS DMS 이후부터의 AWS 관리형 정책 업데이트에 대한 세부 정보를 봅니다. 이 페이지의 변경 사항에 대한 자동 알림을 받으려면 AWS DMS 문서 기록 페이지에서 RSS 피드를 구독하세요.

변경	설명	Date
AWS DMS Serverless Service Role Policy – 변경	AWS DMS DMSAWS DMS Serverless Service Role Policy 가 S3 버킷을 생성하고 DMS 서버리스와 관련이 없는 복제 작업을 위해 마이그레이션 전 평가 결과를 해당 버킷에 넣을 수 있도록 업데이트되었습니다.	2025년 11월 5일
AWS DMS 서버리스의 서비스 연결 역할 - 변경	AWS DMS dms:Start ReplicationTaskAssessmentRun 가 마이그레이션 전 평가 실행을 지원AWS DMS Serverless Service Role Policy 하도	2025년 2월 14일

변경	설명	Date
	록 업데이트되었습니다. AWS DMS 또한 S3 버킷을 생성하고 마이그레이션 전 평가 결과를 해당 버킷에 넣도록 서버리스 서비스 연결 역할을 업데이트했습니다.	
AWSDMSServerlessServiceRolePolicy – 변경	AWS DMS 복제 ModifyReplicationTask 작업을 수정하기 위해 작업을 호출하는 데 AWS DMS Serverless에 dms:ModifyReplicationTask 필요한가 추가되었습니다. 복제 인스턴스를 수정하기 위해 ModifyReplicationInstance 작업을 호출dms:ModifyReplicationInstance 하는 데 AWS DMS Serverless에 필요한가 AWS DMS 추가되었습니다.	2025년 1월 17일
AmazonDMSVPCManagementRole – 변경	AWS DMS AWS DMS 가 사용자를 대신하여 네트워크 설정을 관리할 수 있도록 ec2:DescribeDhcpOptions 및 ec2:DescribeNetworkInterfaces 작업을 추가했습니다.	2024년 6월 17일

변경	설명	Date
AWSDMSServerlessServiceRolePolicy – 새 정책	AWS DMS 에서 Amazon CloudWatch 지표 게시와 같이 AWS DMS 가 사용자를 대신하여 서비스를 생성하고 관리할 수 있도록 허용하는 AWSDMSServerlessServiceRolePolicy 역할을 추가했습니다.	2023년 5월 22일
AmazonDMSCloudWatchLogsRole – 변경 사항	AWS DMS 는 서버리스 복제 구성에서 CloudWatch Logs로 AWS DMS 복제 로그를 업로드할 수 있도록 서버리스 리소스에 대한 ARN을 부여된 각 권한에 추가했습니다.	2023년 5월 22일
AWSDMSFleetAdvisorServiceRolePolicy – 새 정책	AWS DMS Fleet Advisor는 지표 데이터 포인트를 Amazon CloudWatch에 게시할 수 있도록 허용하는 새 정책을 추가했습니다.	2023년 3월 6일
AWS DMS 에서 변경 내용 추적 시작	AWS DMS 가 AWS 관리형 정책에 대한 변경 내용 추적을 시작했습니다.	2023년 3월 6일

에 대한 규정 준수 검증 AWS Database Migration Service

타사 감사자는 여러 규정 준수 프로그램의 AWS Database Migration Service 일환으로의 보안 및 AWS 규정 준수를 평가합니다. 여기에는 다음 프로그램이 포함됩니다.

- SOC
- PCI
- ISO
- FedRAMP
- DoD CC SRG
- HIPAA BAA
- MTCS
- CS
- K-ISMS
- ENS High
- OSPAR
- HITRUST CSF

특정 규정 준수 프로그램 범위의 AWS 서비스 목록은 규정 준수 프로그램 [AWS 범위의 서비스 규정 준수 프로그램](#). 일반 정보는 [AWS 규정 준수 프로그램](#) 참조하세요.

를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다 AWS Artifact. 자세한 내용은 [artifact에서 AWS 보고서 다운로드](#).

사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 AWS DMS 결정됩니다.는 규정 준수를 지원하기 위해 다음 리소스를 AWS 제공합니다.

- [보안 및 규정 준수 빠른 시작 안내서](#) – 이 배포 안내서에서는 아키텍처 고려 사항에 관해 설명하고 AWS에서 보안 및 규정 준수에 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.
- [Amazon Web Services의 HIPAA 보안 및 규정 준수를 위한 설계 백서](#) -이 백서에서는 기업이 AWS를 사용하여 HIPAA 준수 애플리케이션을 생성하는 방법을 설명합니다.
- [AWS 규정 준수 리소스](#) -이 워크북 및 가이드 모음은 업계 및 위치에 적용될 수 있습니다.
- [AWS Config](#) -이 AWS 서비스는 리소스 구성이 내부 관행, 업계 지침 및 규정을 얼마나 잘 준수하는지 평가합니다.

- [AWS Security Hub CSPM](#) -이 AWS 서비스는 보안 업계 표준 및 모범 사례 준수를 확인하는 데 도움이
이 AWS 되는 내 보안 상태에 대한 포괄적인 보기를 제공합니다.

의 복원력 AWS Database Migration Service

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 기반으로 구축됩니다. AWS 리전은 물리적으로 분리되고 격리된 여러 가용 영역을 제공하며, 이는 지연 시간이 짧고 처리량이 높으며 중복성이 높은 네트워크와 연결됩니다. 가용 영역을 사용하면 중단 없이 가용 영역 간에 자동으로 장애 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 복수 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 [AWS 글로벌 인프라](#)를 참조하세요.

AWS 글로벌 인프라 외에도는 다중 AZ 옵션을 선택할 때 다중 AZ 배포를 사용하여 복제 인스턴스에 대한고가용성 및 장애 조치 지원을 AWS DMS 제공합니다.

다중 AZ 배포에서 AWS DMS는 다른 가용 영역에 복제 인스턴스의 대기 복제본을 자동으로 프로비저닝하고 유지합니다. 기본 복제 인스턴스는 대기 복제본에 동기식으로 복제됩니다. 기본 복제 인스턴스에 장애가 발생하거나 무응답 상태가 되면 대기 복제본은 중단을 최소화하면서 실행 중이던 작업을 다시 시작합니다. 기본 복제본은 자신의 상태를 끊임없이 대기 복제본으로 복제하기 때문에 다중 AZ 배포는 약간의 성능 오버헤드를 발생시킵니다.

다중 AZ 배포 사용에 관한 자세한 내용은 [AWS DMS 복제 인스턴스 작업](#) 단원을 참조하십시오.

의 인프라 보안 AWS Database Migration Service

관리형 서비스인 AWS 글로벌 네트워크 보안으로 보호 AWS Database Migration Service 됩니다. AWS 보안 서비스 및가 인프라를 AWS 보호하는 방법에 대한 자세한 내용은 [AWS 클라우드 보안을](#) 참조하세요. 인프라 보안 모범 사례를 사용하여 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 [인프라 보호](#)를 참조하세요 AWS .

AWS 에서 게시한 API 호출을 사용하여 네트워크를 AWS DMS 통해 액세스합니다. 클라이언트는 다음을 지원해야 합니다.

- Transport Layer Security(TLS). TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

모든 네트워크 위치에서 이러한 API 작업을 호출할 수 있습니다.는 예를 들어 소스 IP 주소를 기반으로 작업 및 리소스에 대한 제한을 지정할 수 있는 리소스 기반 액세스 정책 AWS DMS 도 지원합니다. 또한 AWS DMS 정책을 사용하여 특정 Amazon VPC 엔드포인트 또는 특정 Virtual Private Cloud(VPCs). 이렇게 하면 네트워크 내의 특정 VPC에서만 지정된 AWS DMS 리소스에 대한 AWS 네트워크 액세스가 효과적으로 격리됩니다. 예제를 포함하여 AWS DMS에서 리소스 기반 액세스 정책을 사용하는 방법에 대한 자세한 내용은 섹션을 참조하세요 [리소스 이름 및 태그를 사용하여 세분화된 액세스 제어](#).

단일 VPC AWS DMS 내에서와의 통신을 제한하기 위해를 AWS DMS 통해 연결할 수 있는 VPC 인터페이스 엔드포인트를 생성할 수 있습니다 AWS PrivateLink. AWS PrivateLink 는 AWS DMS 및 관련 결과에 대한 모든 호출이 인터페이스 엔드포인트가 생성된 특정 VPC로 제한되도록 합니다. 그런 다음 AWS CLI 또는 SDK를 사용하여 실행하는 모든 AWS DMS 명령에서이 인터페이스 엔드포인트의 URL을 옵션으로 지정할 수 있습니다. 이렇게 하면와의 전체 통신이 VPC로 제한된 AWS DMS 상태를 유지하고 퍼블릭 인터넷에서 볼 수 없도록 할 수 있습니다.

단일 VPC에서 DMS에 액세스할 수 있도록 인터페이스 엔드포인트를 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/vpc/> Amazon VPC 콘솔을 엽니다.
2. 탐색 창에서 엔드포인트를 선택합니다. 그러면 VPC에서 인터페이스 엔드포인트를 생성할 수 있는 엔드포인트 생성 페이지가 열립니다 AWS DMS.
3. AWS 서비스를 선택한 다음 서비스 이름의 값을 검색하고 선택합니다.이 경우 다음 형식으로 AWS DMS 표시됩니다.

```
com.amazonaws.region.dms
```

여기서와 같이가 AWS DMS 실행되는 AWS 리전을 *region* 지정합니다 `com.amazonaws.us-west-2.dms`.

4. VPC의 경우, 인터페이스 엔드포인트를 생성할 VPC를 선택합니다(예를 들면 `vpc-12abcd34`).
5. 가용 영역 및 서브넷 ID에서 값을 하나씩 선택합니다. 이 값은 선택한 AWS DMS 엔드포인트가 실행될 수 있는 위치를 나타내야 합니다(예: `us-west-2a (usw2-az1)` 및 `subnet-ab123cd4`).
6. DNS 이름을 사용하여 엔드포인트를 생성하려면 DNS 이름 활성화를 선택합니다. 이 DNS 이름은 임의의 문자열(`ab12dc34`)과 함께 하이픈으로 구분된 엔드포인트 ID(`vpce-12abcd34efg567hij`)로 구성됩니다. 이는 점을 통해 역순으로 서비스 이름과 구분되며 `vpce(dms.us-west-2.vpce.amazonaws.com)`가 추가된 것이 특징입니다.

예를 들면, `vpce-12abcd34efg567hij-ab12dc34.dms.us-west-2.vpce.amazonaws.com`입니다.

7. 보안 그룹의 경우, 엔드포인트에 사용할 그룹을 선택합니다.

보안 그룹을 설정할 때는 해당 그룹 내에서 아웃바운드 HTTPS 호출을 허용해야 합니다. 자세한 내용을 알아보려면 Amazon VPC 사용 설명서의 [보안 그룹 생성](#)을 참조하세요.

8. 정책에서 전체 액세스 또는 사용자 지정 값을 선택합니다. 예를 들어, 다음과 비슷한 사용자 지정 정책을 선택하여 특정 작업 및 리소스에 대한 엔드포인트의 액세스를 제한할 수 있습니다.

```
{
  "Statement": [
    {
      "Action": "dms:*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": "*"
    },
    {
      "Action": [
        "dms:ModifyReplicationInstance",
        "dms>DeleteReplicationInstance"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:dms:us-west-2:<account-id>:rep:<replication-instance-id>",
      "Principal": "*"
    }
  ]
}
```

```
}  
]  
}
```

여기서 샘플 정책은 특정 복제 인스턴스를 삭제하거나 수정하는 것을 제외하고 모든 AWS DMS API 호출을 허용합니다.

이제 6단계에서 생성된 DNS 이름을 사용하여 구성된 URL을 옵션으로 지정할 수 있습니다. 생성된 인터페이스 엔드포인트를 사용하여 서비스 인스턴스에 액세스하는 모든 AWS DMS CLI 명령 또는 API 작업에 대해 이 값을 지정합니다. 예를 들어, 이 VPC에서 다음과 같이 DMS CLI 명령 `DescribeEndpoints`를 실행할 수 있습니다.

```
$ aws dms describe-endpoints --endpoint-url https://vpce-12abcd34efg567hij-ab12dc34.dms.us-west-2.vpce.amazonaws.com
```

프라이빗 DNS 옵션을 활성화한 경우, 요청에 엔드포인트 URL을 지정할 필요는 없습니다.

VPC 인터페이스 엔드포인트 생성 및 사용(프라이빗 DNS 옵션 활성화 포함)에 관한 자세한 내용은 Amazon VPC 사용 설명서의 [인터페이스 VPC 엔드포인트\(AWS PrivateLink\)](#)를 참조하십시오.

리소스 이름 및 태그를 사용하여 세분화된 액세스 제어

Amazon 리소스 이름(ARNs)을 기반으로 리소스 이름과 리소스 태그를 사용하여 AWS DMS 리소스에 대한 액세스를 관리할 수 있습니다. IAM 정책에 조건문을 포함하거나 허용 작업을 정의하여 이 작업을 수행할 수 있습니다.

리소스 이름을 사용하여 액세스 제어

IAM 사용자 계정을 생성하고 AWS DMS 리소스의 ARN에 따라 정책을 할당할 수 있습니다.

다음 정책은 ARN `arn:aws:dms:us-east-1:152683116:rep:DOH67ZTOXGLIXMIHKITV`를 사용하여 AWS DMS 복제 인스턴스에 대한 액세스를 거부합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "dms:*"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:dms:us-east-1:111122223333:rep:DOH67ZTOXGLIXMIHKITV"
    }
  ]
}
```

예를 들면, 정책이 적용 중일 때는 다음 명령이 실패합니다.

```
$ aws dms delete-replication-instance
--replication-instance-arn "arn:aws:dms:us-east-1:152683116:rep:DOH67ZTOXGLIXMIHKITV"
```

```
A client error (AccessDeniedException) occurred when calling the
DeleteReplicationInstance
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
```

```
dms:DeleteReplicationInstance on resource: arn:aws:dms:us-east-1:152683116:rep:D0H67ZT0XGLIXMIHKITV
```

```
$ aws dms modify-replication-instance
  --replication-instance-arn "arn:aws:dms:us-east-1:152683116:rep:D0H67ZT0XGLIXMIHKITV"
```

A client error (AccessDeniedException) occurred when calling the ModifyReplicationInstance operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform: dms:ModifyReplicationInstance on resource: arn:aws:dms:us-east-1:152683116:rep:D0H67ZT0XGLIXMIHKITV

AWS DMS 엔드포인트 및 복제 작업에 대한 액세스를 제한하는 IAM 정책을 지정할 수도 있습니다.

다음 정책은 AWS DMS 엔드포인트의 ARN을 사용하여 엔드포인트에 대한 액세스를 제한합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "dms:*"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:dms:us-east-1:111122223333:endpoint:D6E37YBXTNH0A6XRQSZCUGX"
    }
  ]
}
```

예를 들면, 엔드포인트의 ARN을 사용하는 정책이 적용 중일 때는 다음 명령이 실패합니다.

```
$ aws dms delete-endpoint
  --endpoint-arn "arn:aws:dms:us-east-1:152683116:endpoint:D6E37YBXTNH0A6XRQSZCUGX"
```

A client error (AccessDeniedException) occurred when calling the DeleteEndpoint operation:

User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:DeleteEndpoint
on resource: arn:aws:dms:us-east-1:152683116:endpoint:D6E37YBXTNH0A6XRQSZCUGX

```
$ aws dms modify-endpoint
  --endpoint-arn "arn:aws:dms:us-east-1:152683116:endpoint:D6E37YBXTNH0A6XRQSZCUGX"
```

A client error (AccessDeniedException) occurred when calling the ModifyEndpoint operation:

User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:ModifyEndpoint
on resource: arn:aws:dms:us-east-1:152683116:endpoint:D6E37YBXTNH0A6XRQSZCUGX

다음 정책은 AWS DMS 작업의 ARN을 사용하여 작업에 대한 액세스를 제한합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "dms:*"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:dms:us-east-1:111122223333:task:U03YR4N47DXH3ATT4YMW0IT"
    }
  ]
}
```

예를 들면, 작업의 ARN을 사용하는 정책이 적용 중일 때는 다음 명령이 실패합니다.

```
$ aws dms delete-replication-task
  --replication-task-arn "arn:aws:dms:us-east-1:152683116:task:U03YR4N47DXH3ATT4YMW0IT"
```

```
A client error (AccessDeniedException) occurred when calling the DeleteReplicationTask
operation:
```

```
User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
```

```
dms:DeleteReplicationTask
```

```
on resource: arn:aws:dms:us-east-1:152683116:task:U03YR4N47DXH3ATT4YMW0IT
```

태그를 사용하여 액세스 제어

AWS DMS 는 추가 태그 지정 요구 사항 없이 고객 정의 정책에서 사용할 수 있는 공통 키-값 페어 세트를 정의합니다. AWS DMS 리소스 태그 지정에 대한 자세한 내용은 섹션을 참조하세요 [AWS Database Migration Service의 리소스에 태그 지정](#).

다음은와 함께 사용할 수 있는 표준 태그입니다 AWS DMS.

- `aws:CurrentTime` – 임시 기준에 따라 액세스 제한을 허용하는 요청 날짜 및 시간을 나타냅니다.
- `aws:EpochTime` – 이 태그는 위의 `aws:CurrentTime` 태그와 비슷하지만, 현재 시간이 Unix 에포크 이후에 경과된 초 시간으로 표시됩니다.
- `aws:MultiFactorAuthPresent` – 요청이 멀티 팩터 인증을 통해 서명되었는지 여부를 나타내는 부울 태그입니다.
- `aws:MultiFactorAuthAge` – 멀티 팩터 인증 토큰의 수명(초)에 액세스할 수 있습니다.
- `aws:principaltype` - 현재 요청에 대한 위탁자 유형(사용자, 계정, 페더레이션 사용자 등)에 액세스할 수 있습니다.
- `aws:SourceIp` - 요청을 실행하는 사용자의 원본 IP 주소를 나타냅니다.
- `aws:UserAgent` – 리소스를 요청하는 클라이언트 애플리케이션에 대한 정보를 제공합니다.
- `aws:userid` – 요청을 실행하는 사용자의 ID에 액세스할 수 있습니다.
- `aws:username` – 요청을 실행하는 사용자의 이름에 액세스할 수 있습니다.
- `dms:InstanceClass` – 복제 인스턴스 호스트의 컴퓨팅 크기에 액세스할 수 있습니다.
- `dms:StorageSize` - 스토리지 볼륨 크기(GB)에 액세스할 수 있습니다.

고유한 태그를 정의할 수도 있습니다. 고객 정의 태그는 AWS 태그 지정 서비스에 유지되는 간단한 키-값 페어입니다. 이러한 키-값 페어를 복제 인스턴스, 엔드포인트 및 작업을 비롯한 AWS DMS 리소스에 추가할 수 있습니다. 이러한 태그는 정책의 IAM "조건"문을 사용하여 일치되며 특정 조건 태그를 사용하여 참조됩니다. 태그 키에는 "dms" 접두사, 리소스 유형 및 "tag" 접두사가 붙습니다. 태그 형식은 다음과 같습니다.

```
dms:{resource type}-tag/{tag key}={tag value}
```

예를 들어 "stage=production" 태그를 포함하는 복제 인스턴스에 대해 API 직접 호출 성공만 허용하는 정책을 정의하려 한다고 가정해 보겠습니다. 다음 조건문은 리소스와 지정된 태그를 일치시킵니다.

```
"Condition":
{
  "streq":
  {
    "dms:rep-tag/stage":"production"
  }
}
```

이 정책 조건과 일치하는 복제 인스턴스에 다음 태그를 추가합니다.

```
stage production
```

AWS DMS 리소스에 이미 할당된 태그 외에도 정책을 작성하여 지정된 리소스에 적용할 수 있는 태그 키와 값을 제한할 수도 있습니다. 이 경우 태그 접두사는 "req"입니다.

예를 들면, 다음 정책문은 사용자가 지정된 리소스에 할당할 수 있는 태그를 특정 허용 값 목록으로 제한합니다.

```
"Condition":
{
  "streq":
  {
    "dms:rep-tag/stage": [ "production", "development", "testing" ]
  }
}
```

다음 정책 예제에서는 AWS DMS 리소스 태그를 기반으로 리소스에 대한 액세스를 제한합니다.

다음 정책은 태그 값이 "Desktop"이고 태그 키가 "Env"인 복제 인스턴스에 대한 액세스를 제한합니다.

JSON


```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "dms:*"
      ],
      "Effect": "Deny",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "dms:rep-tag/Env": [
            "Desktop"
          ]
        }
      }
    }
  ]
}
```

다음 명령은 태그 값이 "Desktop"이고 태그 키가 "Env"일 때 액세스를 제한하는 IAM 정책에 따라 성공하거나 실패합니다.

```
$ aws dms list-tags-for-resource
--resource-name arn:aws:dms:us-east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN
--endpoint-url http://localhost:8000
{
  "TagList": [
    {
      "Value": "Desktop",
      "Key": "Env"
    }
  ]
}

$ aws dms delete-replication-instance
--replication-instance-arn "arn:aws:dms:us-east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN"
A client error (AccessDeniedException) occurred when calling the
DeleteReplicationInstance
```

```
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:DeleteReplicationInstance on resource: arn:aws:dms:us-
east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN
```

```
$ aws dms modify-replication-instance
  --replication-instance-arn "arn:aws:dms:us-
east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN"
```

A client error (AccessDeniedException) occurred when calling the
ModifyReplicationInstance

```
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:ModifyReplicationInstance on resource: arn:aws:dms:us-
east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN
```

```
$ aws dms add-tags-to-resource
  --resource-name arn:aws:dms:us-east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN
  --tags Key=CostCenter,Value=1234
```

A client error (AccessDeniedException) occurred when calling the AddTagsToResource
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:AddTagsToResource on resource: arn:aws:dms:us-
east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN

```
$ aws dms remove-tags-from-resource
  --resource-name arn:aws:dms:us-east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN
  --tag-keys Env
```

A client error (AccessDeniedException) occurred when calling the
RemoveTagsFromResource

```
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:RemoveTagsFromResource on resource: arn:aws:dms:us-
east-1:152683116:rep:46DH0U7J0JY0JXWD0ZNFEN
```

다음 정책은 태그 값이 "Desktop"이고 태그 키가 "Env"인 AWS DMS 엔드포인트에 대한 액세스를 제한합니다.

JSON

```
{
  "Version": "2012-10-17",
```

```

    "Statement": [
      {
        "Action": [
          "dms:*"
        ],
        "Effect": "Deny",
        "Resource": "*",
        "Condition": {
          "StringEquals": {
            "dms:endpoint-tag/Env": [
              "Desktop"
            ]
          }
        }
      }
    ]
  }
}

```

다음 명령은 태그 값이 "Desktop"이고 태그 키가 "Env"일 때 액세스를 제한하는 IAM 정책에 따라 성공하거나 실패합니다.

```

$ aws dms list-tags-for-resource
--resource-name arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLF52344IZWA6I
{
  "TagList": [
    {
      "Value": "Desktop",
      "Key": "Env"
    }
  ]
}

```

```

$ aws dms delete-endpoint
--endpoint-arn "arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLF52344IZWA6I"

```

A client error (AccessDeniedException) occurred when calling the DeleteEndpoint operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform: dms:DeleteEndpoint on resource: arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLF52344IZWA6I

```
$ aws dms modify-endpoint
  --endpoint-arn "arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I"
```

A client error (AccessDeniedException) occurred when calling the ModifyEndpoint operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform: dms:ModifyEndpoint on resource: arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I

```
$ aws dms add-tags-to-resource
  --resource-name arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I
  --tags Key=CostCenter,Value=1234
```

A client error (AccessDeniedException) occurred when calling the AddTagsToResource operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform: dms:AddTagsToResource on resource: arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I

```
$ aws dms remove-tags-from-resource
  --resource-name arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I
  --tag-keys Env
```

A client error (AccessDeniedException) occurred when calling the RemoveTagsFromResource operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform: dms:RemoveTagsFromResource on resource: arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I

다음 정책은 태그 값이 "Desktop"이고 태그 키가 "Env"인 복제 작업에 대한 액세스를 제한합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "dms:*"
      ],
      "Effect": "Deny",
      "Resource": "*"
    }
  ]
}
```

```

        "Condition": {
            "StringEquals": {
                "dms:task-tag/Env": [
                    "Desktop"
                ]
            }
        }
    ]
}

```

다음 명령은 태그 값이 "Desktop"이고 태그 키가 "Env"일 때 액세스를 제한하는 IAM 정책에 따라 성공하거나 실패합니다.

```

$ aws dms list-tags-for-resource
  --resource-name arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
{
  "TagList": [
    {
      "Value": "Desktop",
      "Key": "Env"
    }
  ]
}

$ aws dms delete-replication-task
  --replication-task-arn "arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3"

```

A client error (AccessDeniedException) occurred when calling the DeleteReplicationTask operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform: dms:DeleteReplicationTask on resource: arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3

```

$ aws dms add-tags-to-resource
  --resource-name arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
  --tags Key=CostCenter,Value=1234

```

A client error (AccessDeniedException) occurred when calling the AddTagsToResource operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:

```
dms:AddTagsToResource on resource: arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
```

```
$ aws dms remove-tags-from-resource
  --resource-name arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
  --tag-keys Env
```

```
A client error (AccessDeniedException) occurred when calling the
RemoveTagsFromResource
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:RemoveTagsFromResource on resource: arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
```

AWS DMS 동종 마이그레이션을 위한 암호화

AWS DMS 동종 마이그레이션은 스토리지 및 기타 구성 요소를 포함하여 데이터 마이그레이션에 사용되는 리소스도 암호화합니다. 고객 관리형 키를 지정하지 않으면 AWS DMS 동종 마이그레이션은 AWS 소유 키를 자동으로 사용하여 리소스를 암호화합니다.

고객 관리형 키를 사용하는 경우 AWS DMS 동종 마이그레이션은 키에 대한 권한을 생성하여 서비스가 필요에 따라 리소스를 암호화하고 해독할 수 있도록 합니다. 이러한 권한 부여는 마이그레이션 수명 주기의 일부로 자동으로 관리됩니다.

동종 마이그레이션에서 고객 관리형 키 사용

동종 마이그레이션에 고객 관리형 키를 사용하려면 AWS DMS 동종 마이그레이션을 사용하려면 IAM 사용자 계정에 부여해야 하는 IAM 권한에 다음 권한을 추가합니다.

```
{
  "Effect": "Allow",
  "Action": [
    "kms:CreateGrant",
    "kms:DescribeKey",
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:GenerateDataKeyWithoutPlaintext"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
```

```

        "kms:ViaService": [
            "dms.us-west-2.amazonaws.com",
            "elasticfilesystem.us-west-2.amazonaws.com"
        ]
    }
}

```

DMS는 마이그레이션 프로세스 중에 암호화 키에 액세스하고 관리하기 위한 내부 권한을 생성합니다. CreateGrant 작업의 범위를 더 좁히기 위해 고객 관리형 키 정책에 다음 제한 사항을 적용할 수 있습니다.

```

{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
  },
  "Action": "kms:CreateGrant",
  "Resource": "*",
  "Condition": {
    "ForAllValues:StringEquals": {
      "kms:GrantOperations": [
        "CreateGrant",
        "DescribeKey",
        "Encrypt",
        "Decrypt",
        "RetireGrant",
        "GenerateDataKeyWithoutPlaintext"
      ]
    },
    "StringEquals": {
      "kms:ViaService": [
        "dms.us-west-2.amazonaws.com",
        "elasticfilesystem.us-west-2.amazonaws.com"
      ]
    }
  }
}

```

⚠ Important

현재 추가 암호화 컨텍스트 제약 조건은 지원되지 않습니다. 이러한 제약 조건을 포함하면 마이그레이션이 실패합니다.

암호화 키 설정 및 AWS KMS 권한 지정

AWS DMS 는 복제 인스턴스에서 사용하는 스토리지와 엔드포인트 연결 정보를 암호화합니다. 복제 인스턴스에서 사용하는 스토리지를 암호화하려면 AWS 계정에 고유한 AWS Key Management Service (AWS KMS) 키를 AWS DMS 사용합니다. 를 사용하여이 키를 보고 관리할 수 있습니다 AWS KMS. 계정의 기본 KMS 키(aws/dms)를 사용하거나 사용자 지정 KMS 키를 생성할 수 있습니다. 기존 KMS 키가 있는 경우, 암호화에 이 키를 사용할 수도 있습니다.

i Note

암호화 AWS KMS 키로 사용하는 사용자 지정 키 또는 기존 키는 대칭 키여야 합니다. AWS DMS 는 비대칭 암호화 키 사용을 지원하지 않습니다. 대칭 및 비대칭 암호화 KMS 키에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 <https://docs.aws.amazon.com/kms/latest/developerguide/symmetric-asymmetric.html> 섹션을 참조하세요.

복제 인스턴스 생성 페이지의 고급 섹션에서 사용자 지정 KMS 키를 선택하지 않은 경우, 복제 인스턴스를 처음 시작할 때 기본 KMS 키(aws/dms)가 생성됩니다. 기본 KMS 키를 사용할 경우, 마이그레이션을 위해 사용 중인 IAM 사용자 계정에 부여해야 하는 유일한 권한은 kms:ListAliases와 kms:DescribeKey입니다. 기본 KMS 키 사용에 대한 자세한 정보는 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

사용자 지정 KMS 키를 사용하려면 다음 옵션 중 하나를 사용하여 사용자 지정 KMS 키에 대한 권한을 할당해야 합니다.

- 마이그레이션에 사용되는 IAM 사용자 계정을 사용자 AWS KMS 지정 키의 키 관리자 또는 키 사용자로 추가합니다. 이렇게 하면 필요한 AWS KMS 권한이 IAM 사용자 계정에 부여됩니다. 이 작업은 AWS DMS를 사용하기 위해 사용자가 IAM 사용자 계정에 부여하는 IAM 권한 외의 것입니다. 키 사용자에게 권한을 부여하는 방법에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 [키 사용자가 KMS 키를 사용하도록 허용](#)을 참조하세요.

- 사용자 지정 KMS 키에서 IAM 사용자 계정을 키 관리자/키 사용자로서 추가하지 않을 경우, AWS DMS를 사용할 수 있도록 IAM 사용자 계정에 부여해야 하는 IAM 권한에 다음 추가 권한을 부여해야 합니다.

```
{
    "Effect": "Allow",
    "Action": [
        "kms:ListAliases",
        "kms:DescribeKey",
        "kms:CreateGrant",
        "kms:Encrypt",
        "kms:ReEncrypt*"
    ],
    "Resource": "*"
},
```

AWS DMS 는 KMS 키 별칭에서도 작동합니다. 자체 AWS KMS 키를 생성하고 사용자에게 KMS 키에 대한 액세스 권한을 부여하는 방법에 대한 자세한 내용은 [AWS KMS 개발자 안내서](#)를 참조하세요.

KMS 키 식별자를 지정하지 않으면 기본 암호화 키를 AWS DMS 사용합니다.는 AWS 계정의에 AWS DMS 대한 기본 암호화 키를 AWS KMS 생성합니다. 계정에 AWS 는 AWS 리전마다 다른 기본 암호화 키가 있습니다.

AWS DMS 리소스를 암호화하는 데 사용되는 AWS KMS 키를 관리하려면를 사용합니다 AWS Key Management Service.는 안전하고 가용성이 높은 하드웨어와 소프트웨어를 AWS KMS 결합하여 클라우드에 맞게 조정된 키 관리 시스템을 제공합니다. 를 사용하여 암호화 키를 AWS KMS생성하고 이러한 키의 사용 방법을 제어하는 정책을 정의할 수 있습니다.

AWS KMS 에서 찾을 수 있습니다. AWS Management Console

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/kms> AWS Key Management Service (AWS KMS) 콘솔을 엽니다.
2. 를 변경하려면 페이지 오른쪽 상단의 리전 선택기를 AWS 리전사용합니다.
3. 다음 옵션 중 하나를 선택하여 AWS KMS 키로 작업합니다.
 - 가 AWS 자동으로 생성하고 관리하는 계정의 키를 보려면 탐색 창에서 AWS 관리형 키를 선택합니다.

- 해당 계정에서 직접 생성하고 관리하는 키를 보려면 탐색 창에서 고객 관리형 키를 선택합니다.

AWS KMS 는 AWS CloudTrail를 지원하므로 키 사용을 감사하여 키가 적절하게 사용되고 있는지 확인할 수 있습니다. AWS KMS 키는 AWS DMS 및 Amazon RDS, Amazon S3, Amazon Redshift, Amazon EBS와 같은 지원되는 AWS 서비스와 함께 사용할 수 있습니다.

또한 사용자 지정 AWS KMS 키를 생성하여 다음 AWS DMS 엔드포인트의 대상 데이터를 암호화할 수도 있습니다.

- Amazon Redshift – 자세한 내용은 [AWS KMS 키를 사용하여 Amazon Redshift 대상 데이터 암호화](#) 섹션을 참조하세요.
- Amazon S3 – 자세한 내용은 [Amazon S3 대상 객체를 암호화하는 AWS KMS 키 생성](#) 섹션을 참조하세요.

KMS 키로 AWS DMS 리소스를 생성한 후에는 해당 리소스의 암호화 키를 변경할 수 없습니다. AWS DMS 리소스를 생성하기 전에 암호화 키 요구 사항을 결정해야 합니다.

에 대한 네트워크 보안 AWS Database Migration Service

를 사용할 때 생성하는 네트워크의 보안 요구 사항은 네트워크를 구성하는 방법에 AWS Database Migration Service 따라 다릅니다. 의 네트워크 보안에 대한 일반 규칙 AWS DMS 은 다음과 같습니다.

- 복제 인스턴스에는 리소스 및 대상 엔드포인트에 대한 액세스 권한이 있어야 합니다. 복제 인스턴스 용 보안 그룹에는 데이터베이스 포트에서 밖으로 데이터베이스 엔드포인트까지 인스턴스를 내보내도록 허용하는 네트워크 ACL 또는 규칙이 있어야 합니다.
- 데이터베이스 엔드포인트에는 복제 인스턴스에서 수신되는 액세스를 허용하는 네트워크 ACL과 보안 그룹 규칙이 포함되어야 합니다. 이 작업을 완료하기 위해 사용자의 구성에 따라 복제 인스턴스의 보안 그룹, 프라이빗 IP 주소, 퍼블릭 IP 주소 또는 NAT 게이트웨이의 퍼블릭 주소를 사용합니다.
- 네트워크에서 VPN 터널을 사용하는 경우, NAT 게이트웨이 역할을 하는 Amazon EC2 인스턴스는 복제 인스턴스가 트래픽을 전송하도록 허용하는 규칙을 가진 보안 그룹을 사용해야 합니다.

기본적으로 AWS DMS 복제 인스턴스에서 사용하는 VPC 보안 그룹에는 모든 포트에서 0.0.0.0/0으로 송신을 허용하는 규칙이 있습니다. 이 보안 그룹을 수정하거나 자체 보안 그룹을 사용하는 경우, 각 데이터베이스 포트에서 최소한 발신이 원본과 대상 엔드포인트에 허용되어야 합니다.

각 데이터베이스 마이그레이션에서 사용할 수 있는 네트워크를 구성할 때에는 다음 구체적인 보안 사항을 고려해야 합니다.

- [하나의 VPC에 모든 데이터베이스 마이그레이션 구성 요소가 있는 구성](#) - 엔드포인트에서 사용되는 보안 그룹은 복제 인스턴스의 데이터베이스 포트에서 수신을 허용해야 합니다. 복제 인스턴스에서 사용하는 보안 그룹이 엔드포인트에 대한 수신이 있는지 확인하고, 그렇지 않으면 복제 인스턴스 액세스의 프라이빗 IP 주소를 허용하고 엔드포인트가 사용하는 보안 그룹에 규칙을 생성할 수 있습니다.
- [여러 VPC를 사용한 구성](#) - 복제 인스턴스에서 사용하는 보안 그룹에는 데이터베이스의 VPC 범위와 DB 포트에 대한 규칙이 있어야 합니다.
- [Direct Connect 또는 VPN을 사용하여 네트워크를 VPC로 구성](#) - VPC에서 온프레미스 VPN으로 트래픽 전송을 허용하는 VPN 터널입니다. 이 구성에서 VPC에는 특정 IP 주소나 범위로 지정된 트래픽을 호스트로 전송하는 라우팅 규칙이 포함되어 있어 트래픽을 VPC에서 온프레미스 VPN으로 연결할 수 있습니다. 이 경우에 트래픽을 복제 인스턴스의 프라이빗 IP 주소나 보안 그룹에서 NAT 인스턴스로 전송하도록 허용해야 하는 자체 보안 그룹 설정이 NAT 호스트에 적용되어 있습니다.
- [인터넷을 사용한 네트워크와 VPC 연결 구성](#) - VPC 보안 그룹에는 트래픽을 VPC로 전달하지 않고 인터넷 게이트웨이로 전송하는 라우팅 규칙이 적용되어 있어야 합니다. 이 구성에서 엔드포인트와의 연결이 복제 인스턴스의 퍼블릭 IP 주소에서 오는 것으로 나타납니다.

- [ClassicLink를 사용하여 VPC에 없는 RDS DB 인스턴스를 VPC에 있는 DB 인스턴스와 연결하여 구성](#) - 원본 또는 대상 Amazon RDS DB 인스턴스가 VPC에 없고 보안 그룹을 VPC(복제 인스턴스가 있음)와 공유하지 않을 경우, 프록시 서버를 설정하고 ClassicLink를 사용하여 원본과 대상 데이터베이스를 연결할 수 있습니다.
- 소스 엔드포인트가 복제 인스턴스에서 사용하는 VPC 외부에 있고 NAT 게이트웨이를 사용 - 단일 탄력적 네트워크 인터페이스에 연결된 단일 탄력적 IP 주소를 사용하여 네트워크 주소 변환(NAT) 게이트웨이를 구성할 수 있습니다. 그런 다음 이 탄력적 네트워크 인터페이스는 NAT 식별자(nat-#####)를 수신합니다. VPC에 인터넷 게이트웨이가 아닌 해당 NAT 게이트웨이로 기본 경로가 포함된 경우, 대신 복제 인스턴스가 인터넷 게이트웨이의 퍼블릭 IP 주소를 사용하여 데이터베이스 엔드포인트에 접속하는 것으로 나타납니다. 이 경우에 VPC 외부에 있는 데이터베이스 엔드포인트로의 수신은 복제 인스턴스의 퍼블릭 IP 주소가 아닌 NAT 주소의 수신을 허용해야 합니다.
- 비 RDBMS 엔진용 VPC 엔드포인트 - AWS DMS 는 비 RDBMS 엔진용 VPC 엔드포인트를 지원하지 않습니다.

에서 SSL 사용 AWS Database Migration Service

Secure Sockets Layer(SSL)을 사용하여 소스와 대상 엔드포인트 연결을 암호화할 수 있습니다. 이렇게 하려면 AWS DMS Management Console 또는 AWS DMS API를 사용하여 엔드포인트에 인증서를 할당할 수 있습니다. AWS DMS 콘솔을 사용하여 인증서를 관리할 수도 있습니다.

모든 데이터베이스가 동일한 방식으로 SSL을 사용하는 것은 아닙니다. Amazon Aurora MySQL 호환 에디션은 클러스터에 있는 기본 인스턴스의 엔드포인트인 서버 이름을 SSL용 엔드포인트로 사용합니다. Amazon Redshift 엔드포인트는 이미 SSL 연결을 사용하므로 AWS DMS에서 설정한 SSL 연결이 필요하지 않습니다. Oracle 엔드포인트에서는 추가 단계가 필요합니다. 자세한 내용은 [Oracle 엔드포인트용 SSL 지원](#) 섹션을 참조하세요.

주제

- [에서 SSL 사용 시 제한 사항 AWS DMS](#)
- [인증서 관리](#)
- [MySQL 호환, PostgreSQL 또는 SQL Server 엔드포인트에서 SSL 활성화](#)

보안 연결을 설정하려면 엔드포인트에서 서버의 SSL 인증서를 서명하는 데 사용된 루트 인증서 또는 루트까지 이어지는 중간 CA 인증서 체인(인증서 번들 형태)을 제공해야 합니다. 인증서는 PEM 형식 X509 파일로만 허용됩니다. 인증서를 가져올 때에는 엔드포인트에서 인증서를 지정하는 데 사용할 수 있는 Amazon 리소스 이름(ARN)을 수신합니다. Amazon RDS를 사용하는 경우 Amazon RDS에서 호스팅하는 rds-combined-ca-bundle.pem 파일에 제공된 루트 CA 및 인증서 번들을 다운로드할 수 있습니다. 이 파일을 다운로드하는 방법에 자세한 내용은 Amazon RDS 사용 설명서에서 [SSL을 사용하여 DB 인스턴스에 대한 연결 암호화](#)를 참조하세요.

여러 SSL 모드에서 선택하여 SSL 인증서 확인에 사용할 수 있습니다.

- none - 연결이 암호화되지 않습니다. 이 옵션은 보안이 적용되지 않지만, 오버헤드를 줄여야 합니다.
- require - CA 검증 없이 SSL(TLS)를 사용하여 연결이 암호화됩니다. 이 옵션은 보안이 강화되어 오버헤드가 증가합니다.
- verify-ca - 연결이 암호화됩니다. 이 옵션은 보안이 강화되어 오버헤드가 증가합니다. 이 옵션은 서버 인증서를 확인합니다.
- verify-full - 연결이 암호화됩니다. 이 옵션은 보안이 강화되어 오버헤드가 증가합니다. 이 옵션은 서버 인증서를 확인하고 서버 호스트 이름이 인증서의 호스트 이름 속성과 일치하는지 확인합니다.

데이터베이스 엔드포인트에서 작동하지 않는 SSL 모드도 있습니다. 다음 표에는 각 데이터베이스 엔진별로 지원되는 SSL 모드가 나와 있습니다.

DB 엔진	none	[require]	[verify-ca]	[verify-full]
MySQL/MariaDB/ Amazon Aurora MySQL	기본값	지원되지 않음	지원	지원됨
Microsoft SQL Server	기본값	지원됨	지원되지 않음	지원됨
PostgreSQL	기본값	지원됨	지원됨	지원됨
Amazon Redshift	기본값	SSL이 활성화 되지 않음	SSL이 활성화 되지 않음	SSL이 활성화 되지 않음
Oracle	기본값	지원되지 않음	지원됨	지원되지 않음
SAP ASE	기본값	SSL이 활성화 되지 않음	SSL이 활성화 되지 않음	지원됨
MongoDB	기본값	지원됨	지원되지 않음	지원됨
Db2 LUW	기본값	지원되지 않음	지원됨	지원되지 않음
Db2 for z/OS	기본값	지원되지 않음	지원됨	지원되지 않음

Note

DMS 콘솔 또는 API의 SSL 모드 옵션은 Kinesis, DynamoDB와 같은 일부 데이터 스트리밍 및 NoSQL 서비스에는 적용되지 않습니다. 이러한 스트리밍 및 서비스는 기본적으로 안전하므로 DMS에서는 SSL 모드 설정이 없음으로 표시됩니다(SSL 모드=None). 엔드포인트에서 SSL을 사용하기 위한 추가 구성을 제공할 필요는 없습니다. 예를 들어 Kinesis를 대상 엔드포인트로 사용하는 경우, 기본적으로 안전합니다. Kinesis에 대한 모든 API 직접 호출은 SSL을 사용하므로 DMS 엔드포인트에서는 추가 SSL 옵션이 필요하지 않습니다. Kinesis 데이터 스트림에 연결할 때 기본적으로 DMS가 사용하는 HTTPS 프로토콜을 사용하여 SSL 엔드포인트를 통해 안전하게 데이터를 넣고 검색할 수 있습니다.

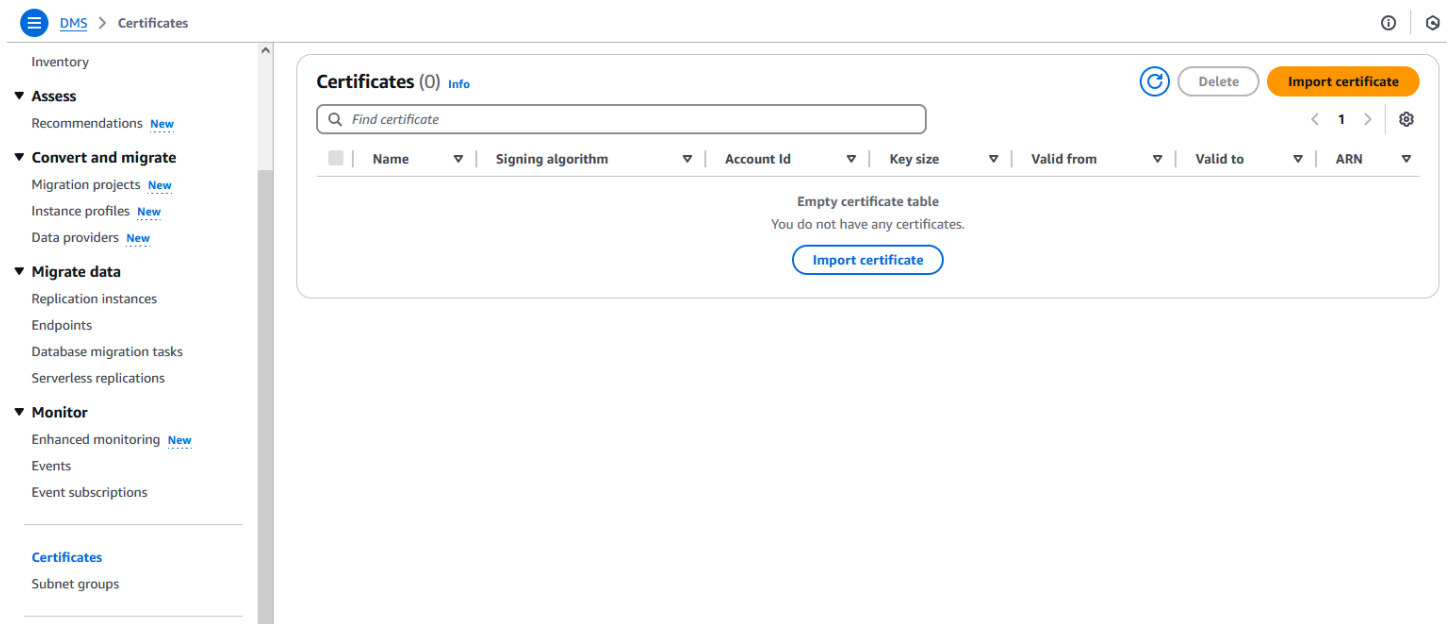
에서 SSL 사용 시 제한 사항 AWS DMS

다음은에서 SSL을 사용할 때 적용되는 제한 사항입니다 AWS DMS.

- Amazon Redshift 대상 엔드포인트에 대한 SSL 연결은 지원되지 않습니다.는 Amazon S3 버킷을 AWS DMS 사용하여 Amazon Redshift 데이터베이스로 데이터를 전송합니다. 이 전송은 기본적으로 Amazon Redshift로 암호화됩니다.
- SSL이 활성화된 Oracle 엔드포인트를 통해 CDC(변경 데이터 캡처) 작업을 수행할 때 SQL 시간 초과가 발생할 수 있습니다. CDC 카운터가 예상 수치를 반영하지 않는 문제가 발생할 경우, 작업 설정의 ChangeProcessingTuning 섹션에서 MinimumTransactionSize 파라미터를 더 낮은 값으로 설정합니다. 최저 100부터 시작할 수 있습니다. MinimumTransactionSize 파라미터에 대한 자세한 내용은 [변경 처리 튜닝 설정](#)을 참조하세요.
- 인증서는 .pem 및 .sso(Oracle wallet) 형식으로만 가져올 수 있습니다.
- 경우에 따라 중간 인증 기관(CA)에서 서버 SSL 인증서를 서명할 수 있습니다. 이 경우 중간 CA에서 루트 CA까지 전체 인증서 체인을 하나의 .pem 파일로 가져와야 합니다.
- 서버에서 자체 서명 인증서를 사용하는 경우, SSL 모드로서 필수를 선택합니다. 필수 SSL 모드는 서버의 SSL 인증서를 명시적으로 신뢰하고 이 인증서가 CA에서 서명되었는지 여부를 확인하려고 하지 않습니다.
- AWS DMS 는 MySQL 및 MariaDb 엔드포인트에 대해 TLS 버전 1.3을 지원하지 않습니다.

인증서 관리

DMS 콘솔을 사용하여 SSL 인증서를 보고 관리할 수 있습니다. 또한 DMS 콘솔을 사용하여 인증서를 가져올 수도 있습니다.



MySQL 호환, PostgreSQL 또는 SQL Server 엔드포인트에서 SSL 활성화

새로 생성한 엔드포인트 또는 기존 엔드포인트에 SSL 연결을 추가할 수 있습니다.

SSL을 사용하여 AWS DMS 엔드포인트를 생성하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

AWS Identity and Access Management (IAM) 사용자로 로그인한 경우에 액세스할 수 있는 적절한 권한이 있는지 확인합니다 AWS DMS. 데이터베이스 마이그레이션에 필요한 권한에 대한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

2. 탐색 창에서 인증서를 선택합니다.
3. 인증서 가져오기를 선택합니다.
4. 엔드포인트와의 연결을 암호화하는 데 사용할 인증서를 업로드합니다.

Note

데이터베이스 엔드포인트 생성 페이지에서 새 CA 인증서 추가를 선택하여 엔드포인트를 생성하거나 수정할 때 AWS DMS 콘솔을 사용하여 인증서를 업로드할 수도 있습니다. Aurora Serverless를 대상으로 사용하려면 [Aurora Serverless에서 TLS/SSL 사용](#)에 설명된 인증서를 받으세요.

5. [2단계: 소스 및 대상 엔드포인트 지정](#) 섹션의 설명과 같이 엔드포인트를 생성합니다.

SSL을 사용하도록 기존 AWS DMS 엔드포인트를 수정하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

IAM 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다. 데이터베이스 마이그레이션에 필요한 권한에 대한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

2. 탐색 창에서 인증서를 선택합니다.
3. 인증서 가져오기를 선택합니다.
4. 엔드포인트와의 연결을 암호화하는 데 사용할 인증서를 업로드합니다.

Note

데이터베이스 엔드포인트 생성 페이지에서 새 CA 인증서 추가를 선택하여 엔드포인트를 생성하거나 수정할 때 AWS DMS 콘솔을 사용하여 인증서를 업로드할 수도 있습니다.

5. 탐색 창에서 엔드포인트를 선택하고, 수정할 엔드포인트를 선택하고, 수정을 선택합니다.
6. SSL 모드의 값을 선택합니다.

verify-ca 또는 verify-full 모드를 선택할 경우 다음과 같이 CA 인증서에 사용할 인증서를 지정합니다.

Secure Socket Layer (SSL) mode | Info

The type of Secure Socket Layer enforcement

verify-full ▼

CA certificate

 ▼

Add new CA certificate

7. 수정을 선택합니다.
8. 엔드포인트를 수정할 때 엔드포인트를 선택한 뒤 연결 테스트를 선택하여 SSL 연결이 잘 작동하는지 여부를 확인합니다.

소스와 대상 엔드포인트를 생성한 후 이 엔드포인트를 사용하는 작업을 생성합니다. 작업 생성에 대한 자세한 내용은 [3단계: 태스크 생성 및 데이터 마이그레이션](#) 섹션을 참조하세요.

데이터베이스 암호 변경

대다수의 경우에 소스 또는 대상 엔드포인트에서 데이터베이스 암호를 쉽게 변경할 수 있습니다. 마이그레이션이나 복제 작업에 현재 사용하고 있는 엔드포인트의 데이터베이스 암호를 변경해야 하는 경우, 이 과정에는 추가 단계가 필요합니다. 다음 절차에서는 이 작업을 수행하는 방법을 보여 줍니다.

마이그레이션이나 복제 작업에 현재 사용하고 있는 엔드포인트의 데이터베이스 암호를 변경하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.

IAM 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다. 필요한 권한에 관한 자세한 내용은 [를 사용하는 데 필요한 IAM 권한 AWS DMS](#) 섹션을 참조하세요.

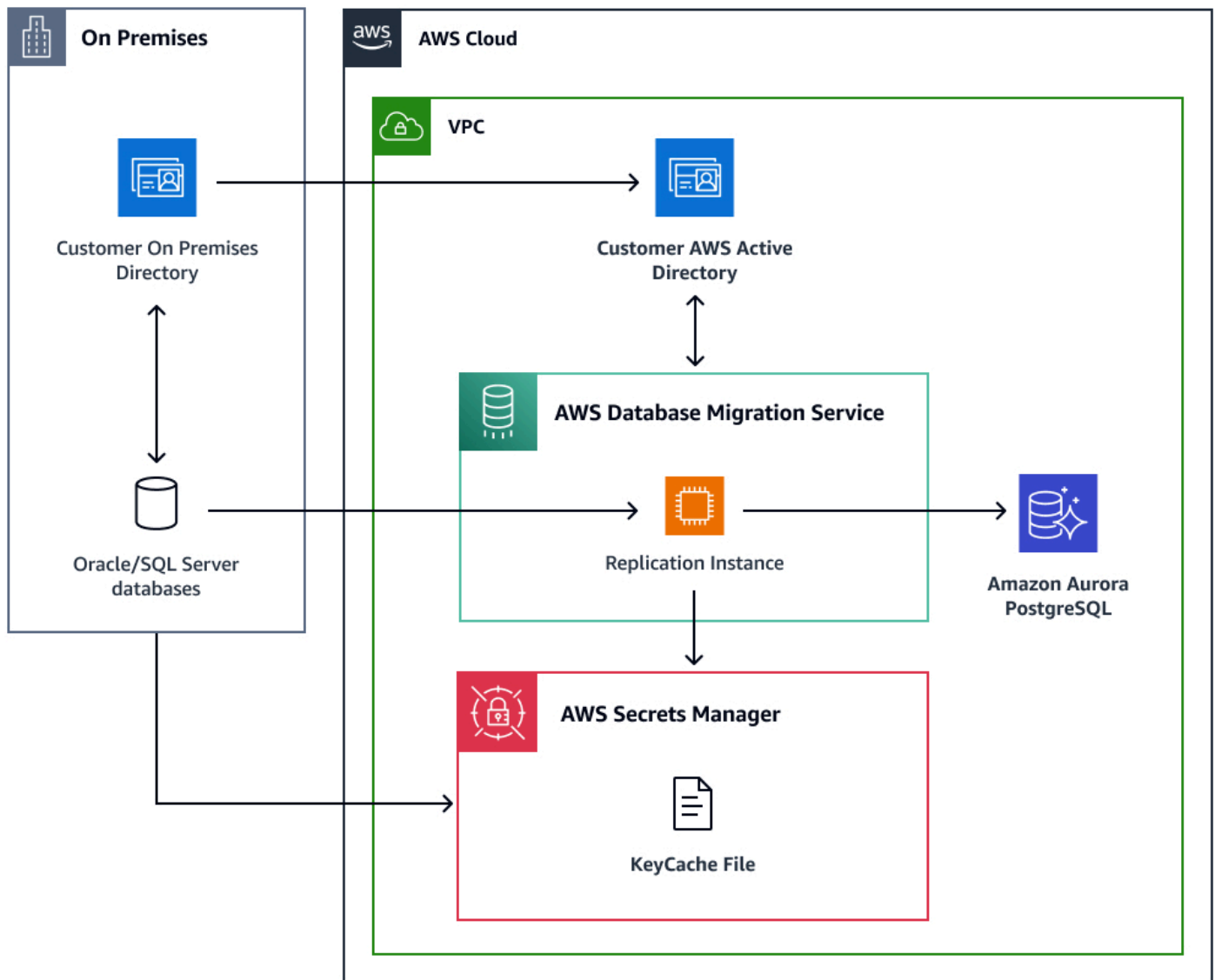
2. 탐색 창에서 데이터베이스 마이그레이션 작업을 선택합니다.
3. 변경하려는 엔드포인트를 사용하는 작업을 선택하여 데이터베이스 암호를 변경한 후 중지를 선택합니다.
4. 작업을 중지한 동안 데이터베이스에서 사용하는 기본 도구를 사용하여 엔드포인트의 데이터베이스 암호를 변경할 수 있습니다.
5. DMS 관리 콘솔로 돌아가 탐색 창에서 엔드포인트를 선택합니다.
6. 암호를 변경할 데이터베이스의 엔드포인트를 선택하고 나서 수정을 선택합니다.
7. 암호 상자에 새 암호를 입력하고 저장을 선택합니다.
8. 탐색 창에서 데이터베이스 마이그레이션 작업을 선택합니다.
9. 이전에 중지한 작업을 선택하고 시작/재개를 선택합니다.
10. 작업을 계속할 방법에 따라 다시 시작 또는 재개를 선택하고 작업 시작을 선택합니다.

에서 Kerberos 인증 사용 AWS Database Migration Service

DMS v3.5.3부터 Kerberos 인증을 사용하여 데이터베이스 인스턴스에 연결하도록 Oracle 또는 SQL Server 소스 엔드포인트를 구성할 수 있습니다. DMS는 Directory Service Microsoft Active Directory 및 Kerberos 인증을 지원합니다. Microsoft Active Directory 서비스에 대한 AWS관리형 액세스에 대한 자세한 내용은 [란 무엇입니까 Directory Service?](#)를 참조하세요.

AWS DMS Kerberos 인증 아키텍처 개요

다음 다이어그램은 AWS DMS Kerberos 인증 워크플로에 대한 개략적인 개요를 제공합니다.



에서 Kerberos 인증 사용에 대한 제한 사항 AWS DMS

에서 Kerberos 인증을 사용할 때는 AWS DMS 다음과 같은 제한 사항이 적용됩니다.

- DMS 복제 인스턴스는 Kerberos `krb5.conf` 파일 하나와 키 캐시 파일 하나를 지원합니다.
- 티켓이 만료되기 최소 30분 전에 Secrets Manager에서 Kerberos 키 캐시 파일을 업데이트해야 합니다.
- Kerberos 지원 DMS 엔드포인트는 Kerberos 지원 DMS 복제 인스턴스에서만 작동합니다.

사전 조건

시작하려면 기존 Active Directory 또는 Kerberos 인증 호스트에서 다음 사전 조건을 완료해야 합니다.

- 온프레미스 AD와 Active Directory 신뢰 관계를 설정합니다. 자세한 내용은 [자습서: AWS 관리형 Microsoft AD와 자체 관리형 Active Directory 도메인 간의 신뢰 관계 생성을 참조하세요](#).
- 간소화된 버전의 Kerberos krb5.conf 구성 파일을 준비합니다. 영역, 도메인 관리자 서버의 위치, Kerberos 영역에 대한 호스트 이름 매핑에 대한 정보를 포함합니다. krb5.conf 콘텐츠의 형식이 영역 및 도메인 영역 이름에 대한 올바른 대/소문자 혼합 형식인지 확인해야 합니다. 예제:

```
[libdefaults]
    dns_lookup_realm = true
    dns_lookup_kdc = true
    forwardable = true
    default_realm = MYDOMAIN.ORG
[realms]
MYDOMAIN.ORG = {
    kdc = mydomain.org
    admin_server = mydomain.org
}
[domain_realm]
.mydomain.org = MYDOMAIN.ORG
mydomain.org = MYDOMAIN.ORG
```

- Kerberos 키 캐시 파일을 준비합니다. 파일에는 클라이언트 위탁자 정보의 임시 Kerberos 자격 증명 이 포함되어 있습니다. 파일은 클라이언트의 암호를 저장하지 않습니다. DMS 태스크는 이 캐시 티켓 정보를 사용하여 암호 없이 추가 자격 증명을 가져옵니다. 기존 Active Directory 또는 Kerberos 인증 호스트에서 다음 단계를 실행하여 키 캐시 파일을 생성합니다.
- Kerberos 키탭 파일을 생성합니다. kutil 또는 ktpass 유틸리티를 사용하여 키탭 파일을 생성할 수 있습니다.

Microsoft ktpass 유틸리티에 대한 자세한 내용은 Windows Server 설명서의 [ktpass](#)를 참조하세요.

MIT kutil 유틸리티에 대한 자세한 내용은 MIT Kerberos 설명서의 [kutil](#)을 참조하세요.

- kinit 유틸리티를 사용하여 keytab 파일에서 Kerberos 키 캐시 파일을 생성합니다. kinit 유틸리티에 대한 자세한 내용은 MIT Kerberos 설명서의 [kinit](#)를 참조하세요.
- SecretBinary 파라미터를 사용하여 Secrets Manager에 Kerberos 키 캐시 파일을 저장합니다. Secrets Manager에 키 캐시 파일을 업로드하면 DMS가 해당 파일을 검색한 다음 약 30분마다 로컬 캐시 파일을 업데이트합니다. 로컬 키 캐시 파일이 미리 정의된 만료 타임스탬프를 초과하면 DMS는

작업을 정상적으로 중지합니다. 지속적 복제 작업 중에 인증 실패를 방지하려면 티켓 만료 최소 30분 전에 Secrets Manager에서 키 캐시 파일을 업데이트합니다. 자세한 내용은 Systems Manager API 참조의 [createsecret](#)을 참조하세요. 다음 AWS CLI 샘플은 Secrets Manager에 키 캐시 파일을 이진 형식으로 저장하는 방법을 보여줍니다.

```
aws secretsmanager create-secret --name keycache --secret-binary fileb://keycachefile
```

- IAM 역할에 Secrets Manager에서 키 캐시 파일을 가져올 수 있는 GetSecretValue 및 DescribeSecret 권한을 부여합니다. IAM 역할에 dms-vpc-role 신뢰 정책이 포함되어 있는지 확인합니다. dms-vpc-role 신뢰 정책에 대한 자세한 내용은 [와 함께 사용할 IAM 역할 생성 AWS DMS](#) 섹션을 참조하세요.

다음 예에서는 Secrets Manager GetSecretValue 및 DescribeSecret 권한을 사용하여 IAM 역할 정책을 보여줍니다. *<keycache_secretsmanager_arn>* 값은 이전 단계에서 생성한 Keycache Secrets Manager ARN입니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "secretsmanager:DescribeSecret"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS DMS 복제 인스턴스에서 Kerberos 지원 활성화

Kerberos 영역은 Windows의 도메인과 동일합니다. 원칙 영역을 해결하기 위해 Kerberos는 도메인 이름 서비스(DNS)를 사용합니다. dns-name-servers 파라미터를 설정하면 복제 인스턴스는 미리 정

의된 사용자 지정 DNS 서버 세트를 사용하여 Kerberos 도메인 영역을 확인합니다. Kerberos 영역 쿼리를 해결하는 또 다른 대안은 복제 인스턴스 가상 프라이빗 클라우드(VPC)에서 Amazon Route 53를 구성하는 것입니다. 자세한 내용은 [Route 53](#)를 참조하세요.

를 사용하여 DMS 복제 인스턴스에서 Kerberos 지원 활성화 AWS Management Console

콘솔을 사용하여 Kerberos 지원을 활성화하려면 복제 인스턴스 생성 또는 복제 인스턴스 수정 페이지의 Kerberos 인증 섹션에 다음 정보를 입력합니다.

- krb5.conf 파일의 콘텐츠
- 키 캐시 파일이 포함된 Secrets Manager 보안 암호의 ARN
- 보안 암호 관리자 ARN에 액세스할 수 있는 IAM 역할의 ARN 및 키 캐시 파일을 검색할 수 있는 권한

를 사용하여 DMS 복제 인스턴스에서 Kerberos 지원 활성화 AWS CLI

다음 AWS CLI 샘플 호출은 Kerberos를 지원하는 프라이빗 DMS 복제 인스턴스를 생성합니다. 복제 인스턴스는 사용자 지정 DNS를 사용하여 Kerberos 영역을 확인합니다. 자세한 내용은 [create-replication-instance](#) 섹션을 참조하세요.

```
aws dms create-replication-instance
--replication-instance-identifier my-replication-instance
--replication-instance-class dms.t2.micro
--allocated-storage 50
--vpc-security-group-ids sg-12345678
--engine-version 3.5.4
--no-auto-minor-version-upgrade
--kerberos-authentication-settings '{"KeyCacheSecretId":<secret-id>,"KeyCacheSecretIamArn":<secret-iam-role-arn>,"Krb5FileContents":<krb5.conf file contents>}'
--dns-name-servers <custom dns server>
--no-publicly-accessible
```

소스 엔드포인트에서 Kerberos 지원 활성화

DMS Oracle 또는 SQL Server 소스 엔드포인트에서 Kerberos 인증을 활성화하기 전에 클라이언트 머신의 Kerberos 프로토콜을 사용하여 소스 데이터베이스에 인증할 수 있는지 확인합니다. AWS DMS 진단 AMI를 사용하여 복제 인스턴스와 동일한 VPC에서 Amazon EC2 인스턴스를 시작한 다음

kerberos 인증을 테스트할 수 있습니다. API에 대한 자세한 내용은 [AWS DMS 진단 지원 AMI 작업](#) 섹션을 참조하세요.

AWS DMS 콘솔 사용

엔드포인트 데이터베이스에 대한 액세스에서 Kerberos 인증을 선택합니다.

사용 AWS CLI

엔드포인트 설정 파라미터를 지정하고 AuthenticationMethod 옵션을 kerberos로 설정합니다. 예제:

Oracle

```
aws dms create-endpoint
--endpoint-identifier my-endpoint
--endpoint-type source
--engine-name oracle
--username dmsuser@MYDOMAIN.ORG
--server-name mydatabaseserver
--port 1521
--database-name mydatabase
--oracle-settings "{\"AuthenticationMethod\": \"kerberos\"}"
```

SQL Server

```
aws dms create-endpoint
--endpoint-identifier my-endpoint
--endpoint-type source
--engine-name sqlserver
--username dmsuser@MYDOMAIN.ORG
--server-name mydatabaseserver
--port 1433
--database-name mydatabase
--microsoft-sql-server-settings "{\"AuthenticationMethod\": \"kerberos\"}"
```

소스 엔드포인트 테스트

Kerberos 지원 복제 인스턴스에 대해 Kerberos 지원 엔드포인트를 테스트해야 합니다. Kerberos 인증을 위해 복제 인스턴스 또는 소스 엔드포인트를 제대로 구성하지 않으면 엔드포인트 test-connection 작업이 실패하고가 Kerberos 관련 오류를 반환할 수 있습니다. 자세한 내용은 [테스트 연결](#)을 참조하세요.

에 대한 할당량AWS Database Migration Service

다음에서 AWS Database Migration Service(AWS DMS)의 리소스 할당량 및 명명 제약 조건을 확인할 수 있습니다.

마이그레이션할 AWS DMS수 있는 데이터베이스의 최대 크기는 여러 요인에 따라 달라집니다. 예를 들어 소스 환경, 소스 데이터베이스에서 데이터 배포, 소스 시스템의 사용 빈도에 따라 다를 수 있습니다.

특정 시스템이 후보인지 확인하는 가장 좋은 방법은 테스트AWS DMS하는 것입니다. 구성이 작동하도록 천천히 시작한 후 복합 객체를 추가합니다. 마지막으로 전체 로드를 테스트로 시도해 봅니다.

에 대한 리소스 할당량AWS Database Migration Service

각 AWS계정에는 생성할 수 있는 AWS DMS리소스 수에 대한 각 AWS리전의 할당량이 있습니다. 리소스 할당량에 도달하면 해당 리소스 생성을 위한 추가 호출이 예외와 함께 실패합니다.

다음 표에는 각AWS 리전의 AWS DMS리소스와 할당량이 나열되어 있습니다.

Resource	기본 할당량
API 요청 제한	초당 최대 요청 200개
API 요청 새로 고침 빈도	초당 요청 8개
사용자 계정당 복제 인스턴스 수	60
복제 인스턴스에 대한 총 스토리지 양	30,000GB
사용자 계정당 이벤트 구독 수	60
사용자 계정당 복제 서브넷 그룹 수	60
복제 서브넷 그룹당 서브넷	60
사용자 계정당 엔드포인트 수	1,000
복제 인스턴스당 엔드포인트 수	100
사용자 계정당 태스크 수	600

Resource	기본 할당량
복제 인스턴스당 태스크 수	200
사용자 계정당 인증서 수	100
사용자 계정당 데이터 공급자 수	1,000
사용자 계정당 인스턴스 프로파일 수	60
사용자 계정당 마이그레이션 프로젝트 수	50
사용자 계정당 DMS 데이터 수집기 수	10
한 번에 생성되는 대상 권장 사항 수	100
DMS 데이터 수집기가 시간당 업로드할 수 있는 파일 수	500
사용자 계정당 동종 데이터 마이그레이션 수	600
한 번에 실행되는 동종 데이터 마이그레이션 수	100
마이그레이션 프로젝트당 동종 데이터 마이그레이션 수	10
서버리스 복제	100

API 요청 제한 할당량 및 새로 고침 빈도에 대한 자세한 내용은 [API 요청 제한에 대한 이해](#) 섹션을 참조하세요.

스토리지에 대한 30,000GB 할당량은 지정된 AWS리전의 모든 AWS DMS복제 인스턴스에 적용됩니다. 이 스토리지는 대상이 소스를 유지하고 로그 정보를 저장할 수 없는 경우에 변경 사항을 캐시하는데 사용됩니다.

API 요청 제한에 대한 이해

AWS DMS는 초당 200개의 API 호출로 구성된 다양하지만 최대 API 요청 할당량을 지원합니다. 즉, API 요청이 이 속도를 초과하면 병목 현상이 발생합니다. 또한 다른 API 요청을 하기 전에 할당량을 새로 고치는 AWS DMS에 걸리는 시간에 따라 초당 더 적은 API 호출로 제한될 수 있습니다. 이 할당량은 API를 직접 호출하는 경우와 AWS DMS 관리 콘솔을 사용하여 사용자 대신 호출하는 경우 모두에 적용됩니다.

API 요청 제한이 작동하는 방식을 이해하려면 AWS DMS가 API 요청을 추적하는 토큰 버킷을 유지 관리한다고 상상해 보는 것이 좋습니다. 이 시나리오에서는 버킷의 각 토큰을 사용하여 단일 API 직접 호출을 실행할 수 있습니다. 버킷에서 한 번에 200개 이상의 토큰을 보유할 수 없습니다. API 호출 시 버킷에서 토큰 하나를 AWS DMS 제거합니다. 1초 이내에 200개의 API 직접 호출을 실행하면 버킷이 비어 있고 다른 API 직접 호출을 시도해도 실패합니다. API 호출을 수행하지 않는 매초마다 최대 200개의 토큰까지 8개의 토큰을 버킷에 AWS DMS 추가합니다. 이것이 AWS DMS API 요청 새로고침 빈도입니다. 제한 후 언제든지 버킷에 토큰을 추가하면 호출이 다시 제한될 때까지 사용 가능한 토큰만큼 추가 API 직접 호출을 실행할 수 있습니다.

AWS CLI를 사용하여 제한된 API 호출을 실행하는 경우는 다음과 같은 오류를 AWS DMS 반환합니다.

```
An error occurred (ThrottlingException) when calling the AwsDmsApiCall operation
(reached max retries: 2): Rate exceeded
```

여기에서 *AwsDmsApiCall*은 제한이 발생한 AWS DMS API 작업의 이름입니다(예: DescribeTableStatistics). 그러면 제한을 방지하도록 충분한 시간이 경과한 후 다시 시도하거나 다른 호출을 시도할 수 있습니다.

 Note

Amazon EC2와 같은 몇몇 다른 서비스에서 관리하는 API 요청 제한과 달리 AWS DMS에서 관리하는 API 요청 제한 할당량은 증가를 요청할 수 없습니다.

에서 마이그레이션 작업 문제 해결 AWS Database Migration Service

아래에서 AWS Database Migration Service() 관련 문제 해결에 대한 주제를 확인할 수 있습니다AWS DMS. 이러한 주제는 AWS DMS 및 선택한 엔드포인트 데이터베이스를 모두 사용하여 일반적인 문제를 해결하는 데 도움이 될 수 있습니다.

AWS 지원 사례를 연 경우 지원 엔지니어가 엔드포인트 데이터베이스 구성 중 하나와 관련된 잠재적 문제를 식별할 수 있습니다. 엔지니어가 데이터베이스에 대한 진단 정보를 반환하는 지원 스크립트를 실행할 것을 요청할 수도 있습니다. 이러한 유형의 지원 스크립트에서 진단 정보를 다운로드, 실행, 업로드하는 방법에 대한 자세한 내용은 [에서 진단 지원 스크립트 작업 AWS DMS](#) 섹션을 참조하세요.

문제 해결을 위해서는 복제 인스턴스에서 추적 및 덤프 파일을 AWS DMS 수집합니다. 문제 해결이 필요한 문제가 발생하는 경우 AWS Support에 이러한 파일을 제공할 수 있습니다. DMS는 30일 이상 지난 추적 및 덤프 파일은 제거하도록 기본 설정되어 있습니다. 추적 및 덤프 파일 수집을 옵트아웃하려면 AWS Support에서 사례를 엽니다.

주제

- [마이그레이션 태스크가 느리게 실행됨](#)
- [태스크 상태 표시줄이 움직이지 않음](#)
- [태스크가 완료되었지만 아무것도 마이그레이션되지 않음](#)
- [외부 키와 보조 인덱스 누락](#)
- [AWS DMS 는 CloudWatch 로그를 생성하지 않습니다.](#)
- [Amazon RDS에 연결할 때 문제가 발생함](#)
- [네트워킹 문제 발생](#)
- [전체 로드 후 CDC 중단](#)
- [태스크 재시작 시 프라이머리 키 위반 오류 발생](#)
- [스키마 초기 로드 실패](#)
- [알 수 없는 오류로 인해 태스크 실패](#)
- [시작부터 작업 재시작 시 테이블 로드](#)
- [태스크당 테이블 수로 인해 문제 발생](#)
- [LOB 열에 프라이머리 키가 생성되면 태스크가 실패함](#)

- [프라이머리 키가 없는 대상 테이블에서 중복 레코드 발생](#)
- [소스 엔드포인트가 예약된 IP 범위에 속함](#)
- [Amazon Athena 쿼리에서 타임스탬프가 깨져 보임](#)
- [Oracle 관련 문제 해결](#)
- [MySQL 관련 문제 해결](#)
- [PostgreSQL 관련 문제 해결](#)
- [Microsoft SQL Server 관련 문제 해결](#)
- [Amazon Redshift 관련 문제 해결](#)
- [Amazon Aurora MySQL 관련 문제 해결](#)
- [SAP ASE 관련 문제 해결](#)
- [IBM DB2 관련 문제 해결](#)
- [테이블이 "Failed to build 'where' 문" 오류와 함께 테이블을 일시 중지했습니다.](#)
- [의 지연 시간 문제 해결 AWS Database Migration Service](#)
- [에서 진단 지원 스크립트 작업 AWS DMS](#)
- [AWS DMS 진단 지원 AMI 작업](#)

마이그레이션 태스크가 느리게 실행됨

마이그레이션 작업이 느리게 실행되거나 연속 작업이 초기 작업보다 느리게 실행되도록 만들 수 있는 여러 문제가 있습니다.

마이그레이션 작업이 느리게 실행되는 가장 일반적인 이유는 AWS DMS 복제 인스턴스에 할당된 리소스가 부족하기 때문입니다. 복제 인스턴스의 CPU, 메모리, 스왑 파일, IOPS 사용량을 확인하여 실행 중인 태스크를 지원하는 리소스가 인스턴스에 충분히 있는지 확인합니다. 예를 들어, 엔드포인트로서 Amazon Redshift에서 여러 태스크를 실행하면 I/O가 집중적으로 사용됩니다. 복제 인스턴스용 IOPS를 늘리거나 여러 복제 인스턴스에서 작업을 분할하면 보다 효율적인 마이그레이션이 가능합니다.

복제 인스턴스 크기 결정에 대한 자세한 내용은 [복제 인스턴스에 가장 적합한 크기 선택](#) 섹션을 참조하세요.

다음 작업을 수행하여 초기 마이그레이션 로드 속도를 촉진할 수 있습니다.

- 대상이 Amazon RDS DB 인스턴스인 경우, 다중 AZ가 대상 DB 인스턴스에서 활성화되지 않도록 해야 합니다.

- 로드 중에 대상 데이터베이스에서 자동 백업 또는 로깅 기능을 끄고, 마이그레이션이 완료된 후 이러한 기능을 다시 켭니다.
- 이 기능을 대상에서 사용할 수 있는 경우, 프로비저닝된 IOPS를 사용합니다.
- 마이그레이션 데이터에 LOB가 포함된 경우, 작업이 LOB 마이그레이션에 최적화되어 있는지 확인합니다. LOB를 위한 최적화에 대한 자세한 내용은 [대상 메타데이터 작업 설정](#) 섹션을 참조하세요.

태스크 상태 표시줄이 움직이지 않음

작업 상태 표시줄은 작업 진행률의 추정치를 나타냅니다. 이 추정치의 품질은 소스 데이터베이스의 테이블 통계 품질에 따라 달라지며, 테이블 통계 품질이 좋을수록 추정 정확도가 높아집니다.

예상 행 통계가 없는 테이블이 하나뿐인 작업의 경우는 어떤 종류의 백분율 전체 추정치도 제공할 AWS DMS 수 없습니다. 이 경우 태스크 상태와 로드되는 행 표시를 사용하여 태스크가 실제로 실행 및 진행되고 있다는 것을 확인할 수 있습니다.

태스크가 완료되었지만 아무것도 마이그레이션되지 않음

태스크가 완료된 후 아무것도 마이그레이션되지 않은 경우 다음을 수행합니다.

- 엔드포인트를 생성한 사용자에게 마이그레이션하려는 테이블에 대한 읽기 권한이 있는지 확인합니다.
- 마이그레이션하려는 객체가 테이블인지 확인합니다. 보기인 경우, 테이블 매핑을 업데이트하고 객체 로케이터를 'view' 또는 'all'로 지정합니다. 자세한 내용은 [콘솔에서 테이블 선택 및 변환 규칙 지정](#) 단원을 참조하십시오.

외부 키와 보조 인덱스 누락

AWS DMS 는 테이블, 기본 키 및 경우에 따라 고유한 인덱스를 생성하지만 소스에서 데이터를 효율적으로 마이그레이션하는 데 필요하지 않은 다른 객체는 생성하지 않습니다. 예를 들어 보조 인덱스, 기본이 아닌 키 제약 조건 또는 데이터 기본값을 생성하지 않습니다.

데이터베이스에서 보조 객체를 마이그레이션하려면, 원본 데이터베이스와 동일한 데이터베이스 엔진으로 이동할 경우 데이터베이스의 기본 도구를 사용합니다. 보조 객체를 마이그레이션하기 위해 소스 데이터베이스가 사용하는 것과 다른 데이터베이스 엔진으로 마이그레이션할 경우 AWS Schema Conversion Tool (AWS SCT)을 사용합니다.

AWS DMS 는 CloudWatch 로그를 생성하지 않습니다.

복제 태스크에서 CloudWatch 로그가 생성되지 않을 경우, 계정에 `dms-cloudwatch-logs-role` 역할이 있는지 확인합니다. 이 역할이 없는 경우 다음을 수행하여 생성합니다.

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/iam/> IAM 콘솔을 엽니다.
2. 역할 탭을 선택합니다. 역할 생성을 선택합니다.
3. 신뢰할 수 있는 유형의 엔터티 선택 섹션에서 AWS 서비스를 선택합니다.
4. 사용 사례 선택 섹션에서 DMS를 선택합니다.
5. 다음: 권한을 선택합니다.
6. 검색 표시줄에 **AmazonDMSCloudWatchLogsRole**을 입력하고 AmazonDMSCloudWatchLogsRole 옆에 있는 확인란을 선택합니다. 이렇게 하면 CloudWatch에 액세스할 수 있는 AWS DMS 권한이 부여됩니다.
7. 다음: 태그를 선택합니다.
8. 다음: 검토를 선택합니다.
9. 역할 이름에 **dms-cloudwatch-logs-role**을 입력합니다. 이 이름은 대/소문자를 구분합니다.
10. 역할 생성을 선택합니다.

Note

계정이 AWS Organizations의 일부인 경우 서비스 제어 정책(SCPs)이 IAM 역할 권한을 제한하지 않는지 확인합니다. SCPs는 제대로 구성된 경우에도 IAM 역할 권한을 재정의하고 제한할 수 있습니다.

Amazon RDS에 연결할 때 문제가 발생함

소스 또는 대상으로 설정한 Amazon RDS DB 인스턴트에 연결할 수 없는 이유에는 여러 가지가 있습니다. 확인해야 할 몇 가지 항목은 다음과 같습니다.

- 사용자 이름과 암호 조합이 올바른지 확인합니다.
- 인스턴스의 Amazon RDS 콘솔에 표시된 엔드포인트 값이 AWS DMS 엔드포인트를 생성하는 데 사용된 엔드포인트 식별자와 동일한지 확인합니다.

- 인스턴스의 Amazon RDS 콘솔에 표시된 포트 값이 AWS DMS 엔드포인트에 할당된 포트와 동일한지 확인합니다.
- Amazon RDS DB 인스턴스에 할당된 보안 그룹이 AWS DMS 복제 인스턴스의 연결을 허용하는지 확인합니다.
- AWS DMS 복제 인스턴스와 Amazon RDS DB 인스턴스가 동일한 Virtual Private Cloud(VPC)에 없는 경우 DB 인스턴스에 공개적으로 액세스할 수 있는지 확인합니다.

오류 메시지: Incorrect thread connection string: incorrect thread value 0

이 오류는 흔히 엔드포인트와의 연결을 테스트할 때 발생할 수 있습니다. 이 오류는 연결 문자열에 오류가 있음을 나타냅니다. 호스트 IP 주소 뒤의 공백을 예로 들 수 있습니다. 또 다른 예로, 연결 문자열에 잘못된 문자를 복사하는 경우도 있습니다.

네트워킹 문제 발생

가장 일반적인 네트워킹 문제에는 AWS DMS 복제 인스턴스에서 사용하는 VPC 보안 그룹이 포함됩니다. 기본적으로 이 보안 그룹에는 모든 포트에서 0.0.0.0/0로의 발신을 허용하는 규칙이 있습니다. 대부분의 경우 이 보안 그룹을 수정하거나 자체 보안 그룹을 사용합니다. 이렇게 할 경우, 최소한 각 데이터베이스 포트의 소스 및 대상 엔드포인트에 송신을 제공해야 합니다.

그 밖의 구성 관련 문제에는 다음이 포함될 수 있습니다.

- 동일한 VPC의 복제 인스턴스와 소스 및 대상 엔드포인트 – 엔드포인트에서 사용하는 보안 그룹은 복제 인스턴스의 데이터베이스 포트에서 수신을 허용해야 합니다. 복제 인스턴스에서 사용하는 보안 그룹에 엔드포인트로의 수신에 있는지 확인합니다. 그렇지 않을 경우, 복제 인스턴스 액세스의 프라이빗 IP 주소를 허용하고 엔드포인트가 사용하는 보안 그룹에 규칙을 생성할 수 있습니다.
- 소스 엔드포인트가 복제 인스턴스에서 사용하는 VPC 외부에 있음(인터넷 게이트웨이 사용) – VPC 보안 그룹에는 VPC로 전달되지 않고 인터넷 게이트웨이로 트래픽을 전송하는 라우팅 규칙이 포함되어야 합니다. 이 구성에서 엔드포인트와의 연결이 복제 인스턴스의 퍼블릭 IP 주소에서 오는 것으로 나타납니다.
- 소스 엔드포인트가 복제 인스턴스에서 사용하는 VPC 외부에 있음(NAT 게이트웨이 사용) - 단일 탄력적 네트워크 인터페이스에 연결된 단일 탄력적 IP 주소를 사용하여 네트워크 주소 변환(NAT) 게이트웨이를 구성할 수 있습니다. 이 NAT 게이트웨이는 NAT 식별자(nat-####)를 수신합니다.

경우에 따라 VPC에는 인터넷 게이트웨이 대신, 이러한 NAT 게이트웨이에 대한 기본 경로가 포함됩니다. 이러한 경우 복제 인스턴스는 NAT 게이트웨이의 퍼블릭 IP 주소를 사용하여 데이터베이스 엔

드포인트에 접속하는 것으로 보입니다. 이런 상황에서 VPC 외부에 있는 데이터베이스 엔드포인트로의 수신은 복제 인스턴스의 퍼블릭 IP 주소가 아닌 NAT 주소의 수신을 허용해야 합니다.

자체 온프레미스 네임 서버 사용에 대한 자세한 내용은 [자체 온프레미스 이름 서버 사용](#) 섹션을 참조하세요.

전체 로드 후 CDC 중단

느리거나 중단된 복제 변경은 여러 AWS DMS 설정이 서로 충돌할 때 전체 로드 마이그레이션 이후에 나타날 수 있습니다.

예를 들어 대상 테이블 준비 모드 파라미터가 아무 작업 안 함 또는 자르기로 설정되어 있다고 가정해 보겠습니다. 이 경우 기본 및 고유 인덱스 생성을 포함하여 대상 테이블에 대한 설정을 수행하지 AWS DMS 않도록 지시했습니다. 대상 테이블에 기본 또는 고유 키를 생성하지 않은 경우는 각 업데이트에 대해 전체 테이블 스캔을 AWS DMS 수행합니다. 이런 접근 방식은 성능에 상당한 영향을 미칠 수 있습니다.

태스크 재시작 시 프라이머리 키 위반 오류 발생

이 오류는 데이터가 이전 마이그레이션 작업의 대상 데이터베이스에 있을 때 발생할 수 있습니다. 대상 테이블 준비 모드 옵션이 아무 작업 안 함으로 설정된 경우 AWS DMS 는 이전 작업에서 삽입된 데이터 정리를 포함하여 대상 테이블에 대한 준비를 수행하지 않습니다.

작업을 다시 시작하고 이러한 오류를 방지하려면, 이전 태스크 실행에서 대상 테이블에 삽입된 행을 제거합니다.

스키마 초기 로드 실패

경우에 따라 `Operation:getSchemaListDetails:errType=, status=0, errMessage=, errDetails=` 오류가 발생하여 스키마의 초기 로드가 실패할 수 있습니다.

이러한 경우에서 소스 엔드포인트에 연결하는 AWS DMS 데 사용하는 사용자 계정에 필요한 권한이 없습니다.

알 수 없는 오류로 인해 태스크 실패

알 수 없는 유형의 오류가 발생하는 원인은 다양할 수 있습니다. 하지만 문제가 AWS DMS 복제 인스턴스에 할당된 리소스 부족과 관련된 경우가 많습니다.

복제 인스턴스의 CPU, 메모리, 스왑 파일, IOPS 사용량을 확인하여 마이그레이션을 수행하는 리소스가 인스턴스에 충분히 있는지 확인합니다. 모니터링에 대한 자세한 내용은 [AWS Database Migration Service 지표](#) 섹션을 참조하세요.

시작부터 작업 재시작 시 테이블 로드

AWS DMS 는 테이블의 초기 로드를 완료하지 않은 경우 처음부터 테이블 로드를 다시 시작합니다. 작업이 다시 시작되면 초기 로드가 완료되지 않았을 때부터 테이블을 AWS DMS 다시 로드합니다.

태스크당 테이블 수로 인해 문제 발생

복제 태스크당 테이블 수에 대한 제한은 설정되어 있지 않습니다. 그러나 한 태스크의 테이블 수는 60,000개 미만으로 제한하는 것이 좋습니다. 단일 작업에서 60,000개 이상의 테이블을 사용할 때 리소스에 흔히 병목 현상이 발생할 수 있습니다.

LOB 열에 프라이머리 키가 생성되면 태스크가 실패함

FULL LOB 또는 LIMITED LOB 모드에서는 LOB 데이터 형식인 기본 키의 복제를 지원하지 AWS DMS 않습니다.

DMS는 처음에 LOB 열이 null인 행을 마이그레이션한 다음 나중에 LOB 열을 업데이트합니다. 따라서 LOB 열에 프라이머리 키가 생성되면 프라이머리 키가 null이 될 수 없으므로 초기 삽입이 실패합니다. 차선책으로서 다른 열을 프라이머리 키로 추가하고 LOB 열에서 프라이머리 키를 제거합니다.

프라이머리 키가 없는 대상 테이블에서 중복 레코드 발생

전체 로드 및 CDC 태스크를 실행하면 프라이머리 키 또는 고유 인덱스가 없는 대상 테이블에 중복 레코드를 생성할 수 있습니다. 전체 로드 및 CDC 태스크 중에 대상 테이블에 레코드가 중복되지 않도록 하려면 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있어야 합니다.

소스 엔드포인트가 예약된 IP 범위에 속함

AWS DMS 소스 데이터베이스가 예약된 IP 범위 192.168.0.0/24 내의 IP 주소를 사용하는 경우 소스 엔드포인트 연결 테스트가 실패합니다. 다음 단계에서는 가능한 해결 방법을 제공합니다.

1. 192.168.0.0/24에서 소스 데이터베이스와 통신할 수 있는 예약된 범위에 없는 Amazon EC2 인스턴스를 찾습니다.

2. socat 프록시를 설치하고 실행합니다. 다음은 그 한 예입니다.

```
yum install socat

socat -d -d -lmlocal2 tcp4-listen:database_port,bind=0.0.0.0,reuseaddr,fork
tcp4:source_database_ip_address:database_port
&
```

AWS DMS 엔드포인트에 대해 위에서 지정한 Amazon EC2 인스턴스 IP 주소와 데이터베이스 포트를 사용합니다. 엔드포인트가 데이터베이스 포트에 액세스할 AWS DMS 수 있도록 허용하는 보안 그룹이 있는지 확인합니다. 단, 프록시는 DMS 태스크 실행 기간에 실행해야 합니다. 사용 사례에 따라, 프록시 설정을 자동화해야 할 수도 있습니다.

Amazon Athena 쿼리에서 타임스탬프가 깨져 보임

Athena 쿼리에서 타임스탬프가 왜곡된 경우 AWS Management Console 또는 [ModifyEndpoint](#) 작업을 사용하여 Amazon S3 엔드포인트의 `parquetTimestampInMillisecond` 값을 `true`로 설정합니다. 자세한 내용은 [S3Settings](#) 섹션을 참조하세요.

Oracle 관련 문제 해결

아래에서 Oracle 데이터베이스와 AWS DMS 함께를 사용하는 것과 관련된 문제 해결에 대해 알아볼 수 있습니다.

주제

- [보기에서 데이터 가져오기](#)
- [12c에서 LOB 마이그레이션](#)
- [Oracle LogMiner와 Binary Reader 간 전환](#)
- [오류: Oracle CDC stopped 122301 Oracle CDC maximum retry counter exceeded.](#)
- [보충 로깅을 Oracle 소스 엔드포인트에 자동으로 추가합니다.](#)
- [LOB 변경 사항이 캡처되지 않음](#)
- [오류: ORA-12899: 값이 column-name 열에 대해 너무 큼](#)
- [NUMBER 데이터 형식 오해](#)
- [전체 로드 중 레코드 누락](#)
- [테이블 오류](#)

- [오류: Cannot retrieve Oracle archived Redo log destination ids](#)
- [Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능 평가](#)
- [LOB 데이터를 가져오지 못했습니다](#)

보기에서 데이터 가져오기

보기에서 데이터를 끌어와서 지속적인 복제에 사용할 수 없습니다. 보기에서 데이터를 추출하려면 Oracle 소스 엔드포인트 페이지의 엔드포인트 설정 섹션에 아래와 같은 코드를 추가해야 합니다. 보기에서 데이터를 추출하면 보기가 대상 스키마에서 테이블로 표시됩니다.

```
"ExposeViews": true
```

12c에서 LOB 마이그레이션

AWS DMS 는 두 가지 방법을 사용하여 Oracle 데이터베이스인 Binary Reader와 Oracle LogMiner에 대한 변경 사항을 캡처할 수 있습니다. 기본적으로는 Oracle LogMiner를 AWS DMS 사용하여 변경 사항을 캡처합니다. 그렇지만 Oracle 12c에서 Oracle LogMiner는 LOB 열을 지원하지 않습니다. Oracle 12c에서 LOB 열의 변경 사항을 캡처하려면, Binary Reader를 사용합니다.

Oracle LogMiner와 Binary Reader 간 전환

AWS DMS 는 두 가지 방법을 사용하여 소스 Oracle 데이터베이스인 Binary Reader와 Oracle LogMiner의 변경 사항을 캡처할 수 있습니다. Oracle LogMiner가 기본값입니다. 변경 캡처를 위해 Binary Reader 사용으로 전환하려면, 다음 작업을 수행합니다.

변경 캡처를 위해 Binary Reader를 사용하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 엔드포인트를 선택합니다.
3. Binary Reader를 사용할 Oracle 소스 엔드포인트를 선택합니다.
4. 수정을 선택합니다.
5. 고급을 선택한 다음, 추가 연결 속성에 대해 아래의 코드를 추가합니다.

```
useLogminerReader=N
```

6. SQL-Plus와 같은 Oracle 개발자 도구를 사용하여 Oracle 엔드포인트에 연결하는 데 사용되는 AWS DMS 사용자 계정에 다음과 같은 추가 권한을 부여합니다.

```
SELECT ON V_$TRANSPORTABLE_PLATFORM
```

오류: Oracle CDC stopped 122301 Oracle CDC maximum retry counter exceeded.

이 오류는 AWS DMS가 필요한 Oracle 아카이브 로그를 사용하여 변경 사항을 캡처하기 전에 서버에서 제거되었을 때 발생합니다. 데이터 서버에서 로그 보존 기간 정책을 증가시킵니다. Amazon RDS 데이터베이스에서는 다음 절차를 실행하여 로그 보존 기간을 증가시킵니다. 예를 들어, 다음 코드는 Amazon RDS DB 인스턴스에서 로그 보존 기간을 24시간으로 증가시킵니다.

```
exec rdsadmin.rdsadmin_util.set_configuration('archivelog retention hours',24);
```

보충 로깅을 Oracle 소스 엔드포인트에 자동으로 추가합니다.

기본적으로에는 보충 로깅 AWS DMS이 꺼져 있습니다. 원본 Oracle 엔드포인트에서 보충 로깅을 자동으로 켜려면, 다음을 수행합니다.

보충 로깅을 소스 Oracle 엔드포인트에 추가하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 엔드포인트를 선택합니다.
3. 보충 로깅을 추가할 Oracle 원본 엔드포인트를 선택합니다.
4. 수정을 선택합니다.
5. 고급을 선택하고 나서 추가 연결 속성 텍스트 상자에 다음 코드를 추가합니다.

```
addSupplementalLogging=Y
```

6. 수정을 선택합니다.

LOB 변경 사항이 캡처되지 않음

현재가 LOB 변경 사항을 캡처 AWS DMS 하려면 테이블에 기본 키가 있어야 합니다. LOB를 포함하는 테이블에 프라이머리 키가 없는 경우, LOB 변경 사항을 캡처하기 위해 취할 수 있는 몇 가지 조치가 있습니다.

- 테이블에 프라이머리 키를 추가합니다. 이 작업은 트리거를 사용하여 ID 열을 추가하고 이 열을 시퀀스로 채우는 것만큼이나 간편합니다.
- 프라이머리 키로서 시스템에서 생성한 ID를 포함하는 테이블의 구체화된 뷰를 생성하고 테이블 이외에 구체화된 뷰를 마이그레이션합니다.
- 논리 대기를 생성하고, 테이블에 프라이머리 키를 추가하고, 논리 대기에서 마이그레이션합니다.

오류: ORA-12899: 값이 *column-name* 열에 대해 너무 큼

"ORA-12899: *column-name* 열의 값이 너무 큼" 오류는 주로 몇 가지 문제로 인해 발생합니다.

이러한 문제 중 하나는 소스 데이터베이스와 대상 데이터베이스에서 사용하는 문자 집합이 일치하지 않는 것입니다.

또 다른 문제는 두 데이터베이스의 National Language Support(NLS) 설정이 다르다는 점입니다. 이 원인의 일반적인 원인은 원본 데이터베이스 NLS_LENGTH_SEMANTICS 파라미터가 CHAR로 설정되고 대상 데이터베이스 NLS_LENGTH_SEMANTICS 파라미터가 BYTE로 설정되어 있기 때문입니다.

NUMBER 데이터 형식 오해

Oracle NUMBER 데이터 형식은 NUMBER의 정밀도와 규모에 따라 다양한 AWS DMS 데이터 형식으로 변환됩니다. 이 변환은 [Oracle용 소스 데이터 형식](#)에 소개되어 있습니다. 소스 Oracle 엔드포인트에 엔드포인트 설정을 사용할 경우 NUMBER 형식의 변환 방식에 영향을 미칠 수도 있습니다. 이러한 엔드포인트 설정은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#)에 설명되어 있습니다.

전체 로드 중 레코드 누락

전체 로드를 수행할 때는 데이터베이스 수준에서 열린 트랜잭션을 AWS DMS 찾아 트랜잭션이 커밋될 때까지 기다립니다. 예를 들어 작업 설정에 따라서는 열린 트랜잭션이 테이블 매핑에 포함되지 않은 테이블에 있더라도 10분 동안 TransactionConsistencyTimeout=600 AWS DMS 기다립니다. 그러나

열린 트랜잭션이 테이블 매핑에 포함된 테이블에 있고, 트랜잭션이 제시간에 커밋되지 않으면 대상 테이블 결과에서 레코드가 누락됩니다.

열린 트랜잭션을 커밋하는 데 시간이 오래 걸린다는 점을 알게 된 경우, TransactionConsistencyTimeout 태스크 설정을 수정하고 대기 시간을 늘릴 수 있습니다.

또한 FailOnTransactionConsistencyBreached 태스크 설정의 기본값은 false입니다. 즉, 다른 트랜잭션은 AWS DMS 계속 적용되지만 미결 트랜잭션은 누락됩니다. 열린 트랜잭션이 제 시간에 종료되지 않은 경우 태스크가 실패하도록 설정하려면 FailOnTransactionConsistencyBreached를 true로 설정하면 됩니다.

테이블 오류

Table Error 절이 프라이머리 키 열을 참조하지 않고, 모든 열에 보충 로깅이 사용되지 않을 경우 복제 중에 테이블 통계에 WHERE가 표시됩니다.

이 문제를 해결하려면 참조된 테이블의 모든 열에 대해 보충 로깅을 설정합니다. 자세한 내용은 [보충 로깅 설정](#) 단원을 참조하십시오.

오류: Cannot retrieve Oracle archived Redo log destination ids

이 오류는 Oracle 소스에 생성된 아카이브 로그가 없거나, V\$ARCHIVED_LOG가 비어 있는 경우 발생합니다. 로그를 수동으로 전환하여 오류를 해결할 수 있습니다.

Amazon RDS 데이터베이스에서는 다음 절차를 실행하여 로그 파일을 전환합니다. switch_logfile 절차에는 파라미터가 없습니다.

```
exec rdsadmin.rdsadmin_util.switch_logfile;
```

자체 관리형 Oracle 소스 데이터베이스에서는 아래의 명령을 사용하여 로그 전환을 적용합니다.

```
ALTER SYSTEM SWITCH LOGFILE ;
```

Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능 평가

Oracle 소스에서 성능 문제가 발생할 경우, Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능을 평가하여 성능을 개선할 방법을 찾을 수 있습니다. 재실행 또는 아카이브 로그의 읽기 성능을 테스트하려면 [AWS DMS 진단 Amazon Machine Image\(AMI\)](#)를 사용합니다.

AWS DMS 진단 AMI를 사용하여 다음을 수행할 수 있습니다.

- bFile 메서드를 사용하여 재실행 로그 파일의 성능을 평가합니다.
- LogMiner 메서드를 사용하여 재실행 로그 파일의 성능을 평가합니다.
- PL/SQL(dbms_lob.read) 메서드를 사용하여 재실행 로그 파일의 성능을 평가합니다.
- 단일 스레드를 사용하여 ASMFile의 읽기 성능을 평가합니다.
- 다중 스레드를 사용하여 ASMFile의 읽기 성능을 평가합니다.
- Direct OS Readfile() Windows 또는 Pread64 Linux 함수를 사용하여 재실행 로그 파일을 평가합니다.

그런 다음, 결과에 따라 수정 단계를 진행할 수 있습니다.

Oracle 재실행 또는 아카이브 로그 파일의 읽기 성능을 테스트하려면

1. AWS DMS 진단 AMI Amazon EC2 인스턴스를 생성하고 연결합니다.

자세한 내용은 [AWS DMS 진단 AMI 작업을 참조하세요](#).

2. awsreplperf 명령을 실행합니다.

```
$ awsreplperf
```

명령은 AWS DMS Oracle 읽기 성능 유틸리티 옵션을 표시합니다.

```
0. Quit
1. Read using Bfile
2. Read using LogMiner
3. Read file PL/SQL (dms_lob.read)
4. Read ASMFile Single Thread
5. Read ASMFile Multi Thread
6. Readfile() function
```

3. 목록에서 옵션을 선택합니다.
4. 아래의 데이터베이스 연결 및 아카이브 로그 정보를 입력합니다.

```
Oracle user name [system]:
Oracle password:

Oracle connection name [orcllx]:
Connection format hostname:port/instance
```



```
Oracle event trace? [N]:
Default N = No or Y = Yes
```

```
Path to redo or archive log file []:
```

5. 표시된 출력에서 관련 읽기 성능 정보를 확인합니다. 예를 들어, 아래에는 LogMiner를 사용하여 읽기라는 옵션 번호 2를 선택했을 때 표시될 수 있는 출력이 나와 있습니다.

```
Enter your choice>>2
Oracle user name: [system] >> * * *
Oracle password :
Oracle connection name : [orcl1x] >> * * *
Oracle event trace ? : [N] >>n
Full path to redo or archive log file: [] >>+EBSFRA/PORCL/ONLINELOG/group_11.1380.1101828345
Elapsed Time : 7044.83973 sec
Read speed in : 0.088575 MB/sec
LogMinerRead: counted 1198389 redo log rows, total undo / redo size : 655073562
```

6. 유틸리티를 종료하려면 0을 입력합니다.

다음 단계

- 결과에 읽기 속도가 허용 가능한 임계값 미만인 것으로 표시될 경우, 엔드포인트에서 [Oracle 진단 지원 스크립트](#)를 실행하고 대기 시간, 로드 프로파일, IO 프로파일 섹션을 검토합니다. 그런 다음, 읽기 성능을 개선할 수 있는 비정상적인 구성을 조정합니다. 예를 들어, 재실행 로그 파일이 최대 2GB인 경우 LOG_BUFFER를 200MB로 늘리면 성능을 개선하는 데 도움이 될 수 있습니다.
- [AWS DMS 모범 사례](#)를 검토하여 DMS 복제 인스턴스, 태스크, 엔드포인트가 최적으로 구성되어 있는지 확인합니다.

LOB 데이터를 가져오지 못했습니다

의 LOB(Large Object) 조회 실패는 데이터 마이그레이션 프로세스 중에 특정 상황에서 AWS DMS 발생합니다. 전체 로드 단계에서는 작업이 전체 LOB 모드로 구성된 경우 LOB 데이터 마이그레이션을 위한 조회 방법을 AWS DMS 적용합니다. 특히 CDC(데이터 캡처 변경) 단계에서는 LOB 설정에 관계없이 조회 메서드를 AWS DMS 일관되게 사용합니다.

AWS DMS 는 먼저 LOB 열이 없는 행을 복제하고, SELECT 명령을 사용하여 LOB 데이터를 검색하고, UPDATE 명령을 실행하여 대상에 LOB 필드를 복제합니다. 이 순차적 INSERT 및 UPDATE 작업은 작업 동작을 특성화합니다. CDC 단계 중 LOB 조회는 모든 데이터베이스 엔진에 보편적으로 적용할 수 있는 것은 아니며 데이터 크기에 따라 작업이 열 데이터와 함께 인라인 행을 복제할 수 있습니다.

LOB 조회 프로세스 실패는 마이그레이션 중에 "lob 데이터를 가져오지 못하여 null로 설정" 오류 메시지가 표시되는 일반적인 문제입니다. 이 실패 시 대상의 부분 테이블 데이터, 특히 LOB 열은 NULL 값으로 표시됩니다. 다음과 같은 다양한 요인으로 인해 이러한 실패가 발생할 수 있습니다.

- DMS가 조회 프로세스를 완료하기 전에 발생하는 소스 행 삭제
- 조회 스레드를 방해하는 간헐적인 연결 문제
- 소스 테이블 잠금 시나리오로 인해 대기 상태로 전환되는 DMS 조회 쿼리

이러한 LOB 조회 실패를 해결하기 위해 다음을 수행할 수 있습니다.

- 전체 로드 단계에서 제한된 LOB 설정을 구현하여 조회 동작을 제거하고 성능을 개선합니다.
- 대상에서 조회 실패 메시지 및 부분 데이터가 발생할 때 영향을 받는 테이블을 다시 로드합니다.
- 간헐적인 네트워크 또는 소스 데이터베이스 가용성 문제로 인해 발생하는 문제의 경우 작업을 다시 시작하여 모든 테이블 데이터 불일치를 해결합니다.

LOB 조회 실패를 처리하는 이러한 단계는 보다 안정적인 데이터 마이그레이션을 보장하고 프로세스 전반에 걸쳐 데이터 무결성을 유지하는 데 도움이 됩니다.

MySQL 관련 문제 해결

아래에서 MySQL 데이터베이스와 AWS DMS 함께를 사용하는 것과 관련된 문제 해결에 대해 알아볼 수 있습니다.

주제

- [이진 로깅 비활성화로 인해 Amazon RDS DB 인스턴스 엔드포인트에서 CDC 작업 실패](#)
- [작업 중에 대상 MySQL 인스턴스 연결이 끊김](#)
- [MySQL 호환 엔드포인트에 자동 커밋 추가](#)
- [대상 MySQL 호환 엔드포인트에서 외래 키 비활성화](#)
- [물음표로 대체된 문자](#)
- ['Bad event' 로그 항목](#)
- [MySQL 5.5로 변경 데이터 캡처](#)
- [Amazon RDS DB 인스턴스를 위한 이진 로그 보존 기간 증가](#)
- [로그 메시지: 소스 데이터베이스의 일부 변경 사항은 대상 데이터베이스에 적용될 때 영향이 전혀 없었습니다.](#)

- [오류: Identifier too long](#)
- [오류: 지원되지 않는 문자 집합으로 인해 필드 데이터 변환 실패](#)
- [오류: Codepage 1252 to UTF8 \[120112\] A field data conversion failed](#)
- [인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음](#)

이진 로깅 비활성화로 인해 Amazon RDS DB 인스턴스 엔드포인트에서 CDC 작업 실패

이 문제는 자동 백업이 비활성화되어 있기 때문에 Amazon RDS DB 인스턴스에서 발생합니다. 백업 보존 기간을 0이 아닌 값으로 설정하여 자동 백업을 활성화합니다.

작업 중에 대상 MySQL 인스턴스 연결이 끊김

MySQL 대상과 연결이 끊긴 LOB가 포함된 태스크가 있을 경우, 태스크 로그에 다음과 같은 유형의 오류가 표시될 수 있습니다.

```
[TARGET_LOAD ]E: RetCode: SQL_ERROR SqlState: 08S01 NativeError:
2013 Message: [MySQL][ODBC 5.3(w) Driver][mysqld-5.7.16-log]Lost connection
to MySQL server during query [122502] ODBC general error.
```

```
[TARGET_LOAD ]E: RetCode: SQL_ERROR SqlState: HY000 NativeError:
2006 Message: [MySQL][ODBC 5.3(w) Driver]MySQL server has gone away
[122502] ODBC general error.
```

이 경우 일부 태스크 설정을 조정해야 할 수도 있습니다.

작업이 MySQL 대상과 연결이 끊기는 문제를 해결하려면, 다음 작업을 수행합니다.

- 데이터베이스 변수 `max_allowed_packet`이 가장 큰 LOB를 보유할 수 있을 정도로 충분히 크게 설정되어 있는지 확인합니다.
- 다음 변수가 큰 시간 초과 값을 갖도록 설정했는지 확인합니다. 이 변수 각각에서 최소 5분 이상의 값을 사용할 것을 제안합니다.
 - `net_read_timeout`
 - `net_write_timeout`
 - `wait_timeout`

MySQL 시스템 변수 설정에 대한 자세한 내용은 [MySQL 설명서](#)의 [Server System Variables](#)를 참조하세요.

MySQL 호환 엔드포인트에 자동 커밋 추가

대상 MySQL 호환 엔드포인트에 자동 커밋을 추가하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 엔드포인트를 선택합니다.
3. 자동 커밋을 추가할 MySQL과 호환되는 대상 엔드포인트를 선택합니다.
4. 수정을 선택합니다.
5. 고급을 선택하고 나서 추가 연결 속성 텍스트 상자에 다음 코드를 추가합니다.

```
Initstmt= SET AUTOCOMMIT=1
```

6. 수정을 선택합니다.

대상 MySQL 호환 엔드포인트에서 외래 키 비활성화

대상 MySQL, Amazon Aurora MySQL 호환 에디션 또는 MariaDB 엔드포인트의 고급 섹션에 있는 추가 연결 속성에 아래의 코드를 추가하여 MySQL에서 외부 키 검사를 비활성화할 수 있습니다.

대상 MySQL 호환 엔드포인트에서 외래 키를 비활성화하려면

1. 에 로그인 AWS Management Console 하고 <https://console.aws.amazon.com/dms/v2/> AWS DMS 콘솔을 엽니다.
2. 엔드포인트를 선택합니다.
3. 외부 키를 비활성화할 MySQL, Aurora MySQL 또는 MariaDB 대상 엔드포인트를 선택합니다.
4. 수정을 선택합니다.
5. 고급을 선택하고 나서 추가 연결 속성 텍스트 상자에 다음 코드를 추가합니다.

```
Initstmt=SET FOREIGN_KEY_CHECKS=0
```

6. 수정을 선택합니다.

물음표로 대체된 문자

이 문제를 일으키는 가장 일반적인 상황은 소스 엔드포인트 문자가 지원되지 않는 AWS DMS 인스턴스 문자 집합으로 인코딩된 경우입니다.

'Bad event' 로그 항목

마이그레이션 로그에서 'Bad event' 항목은 대개 소스 데이터베이스 엔드포인트에서 지원되지 않는 출력 데이터 정의 언어(DDL) 작업이 시도되었음을 나타냅니다. 지원되지 않는 DDL 작업은 복제 인스턴스가 건너뛸 수 없는 이벤트를 유발하므로 잘못된 이벤트가 로깅됩니다.

이 문제를 해결하려면 태스크를 처음부터 다시 시작합니다. 이렇게 하면 지원되지 않는 DDL 작업이 실행된 후 테이블이 다시 로드되고 변경 사항 캡처가 시작됩니다.

MySQL 5.5로 변경 데이터 캡처

AWS DMS Amazon RDS MySQL 호환 데이터베이스의 변경 데이터 캡처(CDC)에는 MySQL 버전 5.5 이하에서 지원되지 않는 전체 이미지 행 기반 바이너리 로깅이 필요합니다. AWS DMS CDC를 사용하려면 Amazon RDS DB 인스턴스를 MySQL 버전 5.6으로 업그레이드해야 합니다.

Amazon RDS DB 인스턴스를 위한 이진 로그 보존 기간 증가

AWS DMS에서는 변경 데이터 캡처를 위해 이진 로그 파일을 보존해야 합니다. Amazon RDS 인스턴스에서 로그 보존 기간을 연장하려면, 다음 절차를 실행합니다. 다음은 이진 로그 보존 기간을 24시간으로 연장하는 예제입니다.

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

로그 메시지: 소스 데이터베이스의 일부 변경 사항은 대상 데이터베이스에 적용될 때 영향이 전혀 없었습니다.

가 MySQL 데이터베이스 열의 값을 기존 값으로 AWS DMS 업데이트하면 MySQL에서의 메시지가 반환 `zero rows affected`됩니다. 이러한 동작은 Oracle 및 SQL Server와 같은 다른 데이터베이스 엔진과 다릅니다. 이러한 엔진은 대체 값이 현재 값과 동일한 경우에도 하나의 행을 업데이트합니다.

오류: Identifier too long

다음 오류는 식별자가 너무 길 때 발생합니다.

```
TARGET_LOAD E: RetCode: SQL_ERROR SqlState: HY000 NativeError:
1059 Message: MySQLhttp://ODBC 5.3(w) Driverhttp://mysqld-5.6.10Identifier
name 'name' is too long 122502 ODBC general error. (ar_odbc_stmt.c:4054)
```

경우에 따라 대상 데이터베이스에 테이블과 기본 키를 생성 AWS DMS 하도록을 설정합니다. 이러한 경우 DMS는 현재 소스 데이터베이스에서 사용된 프라이머리 키에 동일한 이름을 사용하지 않습니다. 대신, DMS는 테이블 이름을 기준으로 프라이머리 키 이름을 생성합니다. 테이블 이름이 길면, 자동 생성된 식별자는 MySQL에 허용된 한도보다 길 수 있습니다.

이 문제를 해결하기 위한 현재의 접근 방식은 먼저 대상 데이터베이스에 테이블과 프라이머리 키를 미리 생성하는 것입니다. 그런 다음, 태스크 설정 대상 테이블 준비 모드가 아무 작업 안 함 또는 자르기로 설정된 태스크를 사용하여 대상 테이블을 채웁니다.

오류: 지원되지 않는 문자 집합으로 인해 필드 데이터 변환 실패

다음 오류는 지원되지 않는 문자 집합으로 인해 필드 데이터 변환이 실패할 때 발생합니다.

```
"[SOURCE_CAPTURE ]E: Column 'column-name' uses an unsupported character set [120112]
A field data conversion failed. (mysql_endpoint_capture.c:2154)
```

연결과 관련된 데이터베이스의 파라미터를 확인합니다. 다음 명령은 이 파라미터를 설정하는 데 사용할 수 있습니다.

```
SHOW VARIABLES LIKE '%char%';
```

오류: Codepage 1252 to UTF8 [120112] A field data conversion failed

원본 MySQL 데이터베이스에 codepage-1252 문자가 없을 경우 마이그레이션 중에 다음과 같은 오류가 발생할 수 있습니다.

```
[SOURCE_CAPTURE ]E: Error converting column 'column_xyz' in table
'table_xyz with codepage 1252 to UTF8 [120112] A field data conversion failed.
(mysql_endpoint_capture.c:2248)
```

차선책으로 CharsetMapping 추가 연결 속성을 원본 MySQL 엔드포인트와 함께 사용하여 문자 세트 매핑을 지정할 수 있습니다. 이 엔드포인트 설정을 추가하는 경우 마이그레이션 AWS DMS 작업을 처음부터 다시 시작해야 할 수 있습니다.

예를 들어, 소스 문자 집합이 Utf8 또는 latin1인 MySQL 소스 엔드포인트에 아래와 같은 엔드포인트 설정을 사용할 수 있습니다. 65001은 UTF8 코드 페이지 식별자입니다.

```
CharsetMapping=utf8,65001
CharsetMapping=latin1,65001
```

인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음

AWS DMS는 인덱스 및 외래 키와 같은 보조 객체 마이그레이션을 지원하지 않습니다. 하위 항목 업데이트 또는 삭제 작업에서 하위 테이블의 변경 사항을 복제하려면 트리거하는 외부 키 제약 조건이 대상 테이블에서 활성화된 상태여야 합니다. 이러한 제한을 해결하려면 대상 테이블에서 외부 키를 수동으로 생성합니다. 그런 다음, 아래에 설명된 대로 전체 로드 및 CDC를 위한 단일 태스크를 생성하거나, 전체 로드 및 CDC를 위한 별도의 태스크 두 개를 생성합니다.

전체 로드 및 CDC를 지원하는 단일 태스크 생성

이 절차에서는 전체 로드 및 CDC를 위한 단일 태스크를 사용하여 외부 키와 인덱스를 마이그레이션하는 방법을 설명합니다.

전체 로드 및 CDC 태스크 생성

1. 대상에서 외부 키와 인덱스가 있는 테이블을 수동으로 생성하여 소스 테이블과 일치하도록 합니다.
2. 대상 AWS DMS 엔드포인트에 다음 ECA를 추가합니다.

```
Initstmt=SET FOREIGN_KEY_CHECKS=0;
```

3. 가 로 TargetTablePrepMode 설정된 AWS DMS 태스크를 생성합니다 DO_NOTHING.
4. Stop task after full load completes 설정값을 StopTaskCachedChangesApplied로 설정합니다.
5. task.Start는 전체 로드가 완료된 후 작업을 자동으로 AWS DMS 중지하고 캐시된 변경 사항을 적용합니다.
6. 이전에 추가한 SET FOREIGN_KEY_CHECKS ECA를 제거합니다.
7. 태스크를 재개합니다. 태스크는 CDC 단계에 진입하고, 소스 데이터베이스의 지속적인 변경 사항을 대상에 적용합니다.

전체 로드 태스크 및 CDC 태스크를 별도로 생성

이 절차에서는 전체 로드 및 CDC를 위한 별도의 태스크를 사용하여 외부 키와 인덱스를 마이그레이션하는 방법을 설명합니다.

전체 로드 태스크 생성

1. 대상에서 외부 키와 인덱스가 있는 테이블을 수동으로 생성하여 소스 테이블과 일치하도록 합니다.
2. 대상 AWS DMS 엔드포인트에 다음 ECA를 추가합니다.

```
Initstmt=SET FOREIGN_KEY_CHECKS=0;
```

3. TargetTablePrepMode 파라미터를 로 설정하고 DO_NOTHING EnableValidation 로 설정하여 AWS DMS 작업을 생성합니다 FALSE.
4. task.Start는 전체 로드가 완료된 후 작업을 자동으로 AWS DMS 중지하고 캐시된 변경 사항을 적용합니다.
5. 태스크가 완료되면 전체 로드 태스크 시작 시간(UTC) 또는 바이너리 로그 파일의 이름과 위치를 기록하여 CDC 전용 작업을 시작합니다. 로그를 참조하여 초기 전체 로드 시작 시간에서 타임스탬프(UTC)를 가져옵니다.

CDC 전용 태스크 생성

1. 이전에 설정한 SET FOREIGN_KEY_CHECKS ECA를 제거합니다.
2. 시작 위치를 이전 단계에서 기록한 전체 로드 시작 시간으로 설정하여 CDC 전용 태스크를 생성합니다. 또는 이전 단계에서 기록된 바이너리 로그 위치를 사용할 수도 있습니다. TargetTablePrepMode 설정값을 DO_NOTHING로 설정합니다. 필요한 경우 EnableValidation 설정을 TRUE로 설정하여 데이터 검증을 활성화합니다.
3. CDC 전용 태스크를 시작하고, 로그에 오류가 있는지 모니터링합니다.

Note

이 해결 방법은 MySQL에서 MySQL로 마이그레이션하는 경우에만 적용됩니다. 일괄 적용 기능을 사용하려면 대상 테이블에 활성화된 외부 키가 없어야 하므로, 이 방법은 일괄 적용 기능과 함께 사용할 수 없습니다.

PostgreSQL 관련 문제 해결

아래에서 PostgreSQL 데이터베이스와 AWS DMS 함께를 사용하는 것과 관련된 문제 해결에 대해 알아볼 수 있습니다.

주제

- [잘리는 JSON 데이터 형식](#)
- [사용자 정의 데이터 형식 열이 올바르게 마이그레이션되지 않음](#)
- [오류: 생성하도록 선택된 스키마가 없음](#)
- [테이블 삭제와 업데이트가 CDC를 사용하여 복제되지 않음](#)
- [자르기 문이 전파되지 않고 있음](#)
- [PostgreSQL에서 DDL이 캡처되지 않음](#)
- [DDL 캡처를 위해 데이터베이스 객체가 생성되는 스키마 선택](#)
- [PostgreSQL로 마이그레이션한 후 Oracle 테이블 누락](#)
- [ETL 워크로드와 같은 긴 트랜잭션 중에 ReplicationSlotDiskUsage가 증가하고 restart_lsn의 진행이 중지됨](#)
- [소스로 보기를 사용한 작업에서 복사된 행이 없음](#)
- ["UTF8" 인코딩을 위한 잘못된 바이트 시퀀스](#)

잘리는 JSON 데이터 형식

AWS DMS 는 PostgreSQL의 JSON 데이터 형식을 LOB 데이터 유형 열로 처리합니다. 다시 말해서 제한된 LOB 모드를 사용할 때 LOB 크기 제한이 JSON 데이터에 적용됨을 의미합니다.

예를 들어, 제한된 LOB 모드가 4,096KB로 설정되어 있다고 가정해 보겠습니다. 이 경우 4,096KB를 초과하는 모든 JSON 데이터는 4,096KB 한도에서 잘리며 PostgreSQL에서 검증 테스트에 실패합니다.

다음 로그 정보에는 제한된 LOB 모드 설정 및 실패한 검증으로 인해 잘린 JSON 데이터가 나와 있습니다.

```
03:00:49
2017-09-19T03:00:49 [TARGET_APPLY ]E: Failed to execute statement:
'UPDATE "public"."delivery_options_quotes" SET "id"=? , "enabled"=? ,
"new_cart_id"=? , "order_id"=? , "user_id"=? , "zone_id"=? , "quotes"=? ,
"start_at"=? , "end_at"=? , "last_quoted_at"=? , "created_at"=? ,
"updated_at"=? WHERE "id"=? ' [1022502] (ar_odbc_stmt
2017-09-19T03:00:49 [TARGET_APPLY ]E: Failed to execute statement:
'UPDATE "public"."delivery_options_quotes" SET "id"=? , "enabled"=? ,
"new_cart_id"=? , "order_id"=? , "user_id"=? , "zone_id"=? , "quotes"=? ,
"start_at"=? , "end_at"=? , "last_quoted_at"=? , "created_at"=? ,
"updated_at"=? WHERE "id"=? ' [1022502] (ar_odbc_stmt.c:2415)
#
03:00:49
2017-09-19T03:00:49 [TARGET_APPLY ]E: RetCode: SQL_ERROR SqlState:
22P02 NativeError: 1 Message: ERROR: invalid input syntax for type json;,
Error while executing the query [1022502] (ar_odbc_stmt.c:2421)
2017-09-19T03:00:49 [TARGET_APPLY ]E: RetCode: SQL_ERROR SqlState:
22P02 NativeError: 1 Message: ERROR: invalid input syntax for type json;,
Error while executing the query [1022502] (ar_odbc_stmt.c:2421)
```

사용자 정의 데이터 형식 열이 올바르게 마이그레이션되지 않음

PostgreSQL 소스에서 복제할 때는 사용자 정의 데이터 형식이 있는 열을 제외하고 모든 열에 대해 동일한 데이터 형식으로 대상 테이블을 AWS DMS 생성합니다. 그러한 경우에 데이터 형식은 대상에서 'character varying'으로 생성됩니다.

오류: 생성하도록 선택된 스키마가 없음

경우에 따라 "SQL_ERROR SqlState: 3F000 NativeError: 7 Message: ERROR: no schema has been selected to create in"이라는 오류가 표시될 수 있습니다.

JSON 테이블 매핑에 스키마에 대한 와일드카드 값이 포함되어 있지만 소스 데이터베이스가 해당 값을 지원하지 않는 경우, 이 오류가 발생할 수 있습니다.

테이블 삭제와 업데이트가 CDC를 사용하여 복제되지 않음

원본 테이블에 기본 키가 없는 경우 변경 데이터 캡처(CDC) 중 삭제 및 업데이트 작업은 무시됩니다. 기본 키가 있는 PostgreSQL 테이블에 대한 변경 데이터 캡처(CDC)를 AWS DMS 지원합니다.

테이블에 프라이머리 키가 없는 경우, Write Ahead Log(WAL)에 데이터베이스 행의 이전 이미지가 포함되지 않습니다. 이 경우는 테이블을 업데이트할 AWS DMS 수 없습니다. 복제할 작업을 삭제하려면 소스 테이블에서 프라이머리 키를 생성합니다.

자르기 문이 전파되지 않고 있음

변경 데이터 캡처(CDC)를 사용하는 경우 TRUNCATE 작업은에서 지원되지 않습니다 AWS DMS.

PostgreSQL에서 DDL이 캡처되지 않음

아래의 엔드포인트 설정 문을 추가하여 PostgreSQL 대상 엔드포인트가 DDL 문을 캡처하지 않도록 할 수 있습니다.

```
"CaptureDDLs": "N"
```

DDL 캡처를 위해 데이터베이스 객체가 생성되는 스키마 선택

DDL 캡처와 관련된 데이터베이스 객체가 생성되는 스키마를 제어할 수 있습니다. 아래의 엔드포인트 설정 문을 추가합니다. 엔드포인트 설정 파라미터는 소스 엔드포인트의 탭에서 제공됩니다.

```
"DdlArtifactsSchema": "xyzddlschema"
```

PostgreSQL로 마이그레이션한 후 Oracle 테이블 누락

이 경우 일반적으로 테이블과 데이터에는 계속 액세스할 수 있습니다.

Oracle 기본값을 대문자 테이블 이름으로 설정하고, PostgreSQL 기본값을 소문자 테이블 이름으로 설정합니다. Oracle에서 PostgreSQL로 마이그레이션할 경우, 태스크의 테이블 매핑 섹션 아래에 특정 변환 규칙을 제공하는 것이 좋습니다. 이는 테이블 이름의 대소문자를 변환하는 변환 규칙입니다.

변환 규칙을 사용하지 않고 테이블을 마이그레이션하여 테이블 이름의 대소문자를 변환할 경우, 테이블 이름을 참조할 때 테이블 이름을 따옴표로 묶어야 합니다.

ETL 워크로드와 같은 긴 트랜잭션 중에 ReplicationSlotDiskUsage가 증가하고 restart_lsn의 진행이 중지됨

논리적 복제가 활성화된 경우 트랜잭션당 메모리에 보관되는 최대 변경 사항 수는 4MB입니다. 그 후에는 변경 사항이 디스크로 유출됩니다. 따라서 트랜잭션이 완료/중지되고 롤백이 완료될 때까지 ReplicationSlotDiskUsage가 증가하며 restart_lsn이 진행되지 않습니다. 트랜잭션이 길기 때문에 롤백하는 데 시간이 오래 걸릴 수 있습니다.

따라서 논리적 복제가 활성화된 경우, 장기 실행 트랜잭션은 피하세요. 대신 트랜잭션을 여러 개의 작은 트랜잭션으로 나눕니다.

소스로 보기를 사용한 작업에서 복사된 행이 없음

보기를 마이그레이션하려면 table-type을 all 또는 view로 설정합니다. 자세한 내용은 [콘솔에서 테이블 선택 및 변환 규칙 지정](#) 단원을 참조하십시오.

보기를 지원하는 소스는 다음과 같습니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- IBM DB2 LUW
- SAP Adaptive Server Enterprise(ASE)

"UTF8" 인코딩을 위한 잘못된 바이트 시퀀스

를 사용하여 Oracle에서 PostgreSQL로 데이터를 마이그레이션하면 두 데이터베이스 간의 문자 집합 인코딩 차이로 인해 고유한 문제가 AWS DMS 발생합니다. 4바이트 문자를 완전히 지원하는 Oracle의 AL32UTF8 문자 세트에서 중요한 문제가 발생하는 반면 PostgreSQL의 UTF8 문자 세트 구현에는 이 기능이 없습니다. 이러한 차이는 특히 4바이트 문자가 포함된 Oracle 소스의 테이블 또는 열을 처리할 때 마이그레이션 실패로 이어지는 경우가 많습니다.

마이그레이션을 시도하는 동안 DMS 작업 로그와 PostgreSQL 대상 데이터베이스 로그 모두에서 잘못된 UTF8 바이트 시퀀스 문제를 나타내는 오류 메시지가 표시될 수 있습니다. 일반적인 오류 메시지

"ERROR: invalid byte sequence for encoding "UTF8": 0xed 0xb0 0x86"이 표시됩니다. 이 문제를 해결하기 위해서는 "ReplaceChars" 설정을 통해 솔루션을 AWS DMS 제공합니다. 마이그레이션 프로세스 중에 잘못된 문자를 자동으로 대체하거나 제거합니다. 이 접근 방식은 소스 데이터를 수정할 필요 없이 인코딩 관련 오류를 효과적으로 방지합니다.

자세한 내용은 [캐릭터 대체 작업 설정 주제](#)의 캐릭터 세트 검증 및 대체 글머리 기호를 참조하세요.

Microsoft SQL Server 관련 문제 해결

아래에서 Microsoft SQL Server 데이터베이스와 AWS DMS 함께를 사용하는 것과 관련된 문제 해결에 대해 알아볼 수 있습니다.

주제

- [RDS for SQL Server가 보조로 장애 조치된 후 지속적인 복제 실패](#)
- [SQL Server 데이터베이스에서 변경 사항을 캡처하는 동안 발생하는 오류](#)
- [자격 증명 열 누락](#)
- [오류: SQL Server doesn't support publications](#)
- [대상에 변경 사항이 표시되지 않음](#)
- [파티션 간에 매핑된 비균일 테이블](#)
- [오류: 트랜잭션을 처리하는 동안 잘못된 봉투, 잘못된 데이터 컨텍스트/LCX 코드로 인해 CDC 작업이 실패했습니다](#)

RDS for SQL Server가 보조로 장애 조치된 후 지속적인 복제 실패

소스 SQL Server 인스턴스가 보조 인스턴스로 장애 조치되는 경우 AWS DMS 지속적 복제는 연결을 계속 시도하고 소스가 다시 온라인 상태가 되면 계속 복제합니다. 그러나 RDS for SQL Server MAZ 인스턴스의 경우 특정 상황에서 보조 데이터베이스 소유자를 NT AUTHORITY\SYSTEM으로 설정할 수 있습니다. 장애 조치 후 다음 오류와 함께 DMS 태스크가 실패합니다.

```
[SOURCE_CAPTURE ]E: RetCode: SQL_ERROR SqlState: 42000 NativeError: 33009 Message:
[Microsoft][ODBC Driver 17 for SQL Server][SQL Server]The database
owner SID recorded in the master
database differs from the database owner SID recorded in database
'rdsadmin'. You should correct
this situation by resetting the owner of database 'rdsadmin' using the
ALTER AUTHORIZATION statement.
```

```
Line: 1 Column: -1 [1022502] (ar_odbc_stmt.c:5035)
```

이 문제를 해결하려면 [데이터베이스의 rdsa 계정으로 db_owner 변경](#)의 단계를 수행한 다음 DMS 태스크를 재개합니다.

SQL Server 데이터베이스에서 변경 사항을 캡처하는 동안 발생하는 오류

변경 데이터 캡처(CDC) 중에 발생하는 오류는 흔히 사전 요구 사항 중 하나에 부합하지 않음을 나타낼 수 있습니다. 예를 들어, 자주 간과되는 사전 요구 사항은 전체 데이터베이스 백업입니다. 이 작업 로그에는 다음 오류와 함께 이러한 누락이 나타납니다.

```
SOURCE_CAPTURE E: No FULL database backup found (under the 'FULL' recovery model).  
To enable all changes to be captured, you must perform a full database backup.  
120438 Changes may be missed. (sqlserver_log_queries.c:2623)
```

[Microsoft SQL Server 데이터베이스를의 소스로 사용 AWS DMS](#)에서 원본으로서 SQL Server를 위해 나열된 사전 요구 사항을 검토합니다.

자격 증명 열 누락

AWS DMS 는 대상 스키마를 생성할 때 자격 증명 열을 지원하지 않습니다. 초기 로드가 완료된 후 자격 증명 열을 추가해야 합니다.

오류: SQL Server doesn't support publications

SQL Server Express를 원본 엔드포인트로 사용할 때 다음 오류가 생성됩니다.

```
RetCode: SQL_ERROR SqlState: HY000 NativeError: 21106  
Message: This edition of SQL Server does not support publications.
```

AWS DMS 는 현재 SQL Server Express를 소스 또는 대상으로 지원하지 않습니다.

대상에 변경 사항이 표시되지 않음

AWS DMS에서는 변경 사항을 일관되게 캡처하기 위해 소스 SQL Server 데이터베이스가 'FULL' 또는 'BULK LOGGED' 데이터 복구 모델에 있어야 합니다. 'SIMPLE' 모델은 지원되지 않습니다.

SIMPLE 복구 모델은 사용자가 데이터베이스를 복구하도록 허용하는 데 필요한 최소한의 정보를 로깅합니다. 체크포인트가 발생하면 모든 비활성 로그 항목은 자동으로 잘립니다.

모든 작업은 여전히 기록됩니다. 하지만 체크포인트가 발생하는 즉시 로그는 자동으로 잘립니다. 이렇게 잘리면 로그를 재사용할 수 있게 되고, 이전 로그 항목을 덮어쓸 수 있습니다. 로그 항목을 덮어쓰면 변경 사항을 캡처할 수 없습니다. 이 문제는가 SIMPLE 데이터 복구 모델을 지원하지 AWS DMS 않는 이유입니다. SQL Server를 소스로 사용할 때 다른 필요한 사전 요구 사항에 대한 정보는 [Microsoft SQL Server 데이터베이스를의 소스로 사용 AWS DMS](#) 섹션을 참조하세요.

파티션 간에 매핑된 비균일 테이블

변경 데이터 캡처(CDC) 중애가 테이블에서 CDC를 제대로 수행할 AWS DMS 수 없는 경우 특수 구조의 테이블 마이그레이션이 일시 중지됩니다. 다음과 같은 메시지가 표시됩니다.

```
[SOURCE_CAPTURE ]W: Table is not uniformly mapped across partitions. Therefore - it is
excluded from CDC (sqlserver_log_metadata.c:1415)
[SOURCE_CAPTURE ]I: Table has been mapped and registered for CDC.
(sqlserver_log_metadata.c:835)
```

SQL Server 테이블에서 CDC를 실행할 때는 SQL Server tlog를 AWS DMS 구문 분석합니다. 각 tlog 레코드에서는 변경 중에 삽입, 업데이트 또는 삭제된 열에 대한 데이터가 포함된 16진수 값을 AWS DMS 구문 분석합니다.

16진수 레코드를 구문 분석하기 위해서는 SQL Server 시스템 테이블에서 테이블 메타데이터를 AWS DMS 읽습니다. 이러한 시스템 테이블은 특수 구조화된 테이블 열이 무엇인지 식별하고 “xoffset” 및 “null bit position”과 같은 내부 속성 중 일부를 나타냅니다.

AWS DMS 는 테이블의 모든 원시 파티션에 대해 메타데이터가 동일할 것으로 예상합니다. 하지만 특수하게 구성된 테이블의 모든 파티션의 경우에는 메타데이터가 동일하지 않을 때도 있습니다. 이러한 경우는 변경 사항을 잘못 구문 분석하고 대상에 잘못된 데이터를 제공하지 않도록 해당 테이블에서 CDC를 일시 중지할 AWS DMS 수 있습니다. 해결 방법은 다음과 같습니다.

- 테이블에 클러스터링된 인덱스가 있는 경우 인덱스를 재구축합니다.
- 테이블에 클러스터링된 인덱스가 없으면 테이블에 클러스터링된 인덱스를 추가합니다(원하는 경우 나중에 삭제할 수 있음).

오류: 트랜잭션을 처리하는 동안 잘못된 봉투, 잘못된 데이터 컨텍스트/LCX 코드로 인해 CDC 작업이 실패했습니다

'잘못된 봉투' 오류는 AWS DMS가 검증 프로세스 중에 CDC 단계 복제에서 특정 이벤트 유형을 검증할 수 없을 때 발생합니다. 이 오류는 일반적으로 트랜잭션 중간에 있는 특정 타임스탬프에서 작업을 재개할 때 발생할 수 있습니다. 이러한 경우 작업은 해당 '트랜잭션 시작' 이벤트를 찾지 못하고 커밋 이벤트를 읽어 잘못된 트랜잭션 컨텍스트로 이어지고 '잘못된 봉투' 오류를 트리거할 수 있습니다.

이 문제를 해결하려면 작업을 재개하기 전에 추가 연결 속성 섹션에서 `ignoreTxnCtxValidityCheck` 파라미터를 `true`로 설정하여 SQL Server 소스 엔드포인트 구성을 수정해야 합니다. 이 솔루션을 구현한 후에도 오류가 지속되면 AWS [지원 티켓](#)을 제출하세요.

Amazon Redshift 관련 문제 해결

아래에서 Amazon Redshift 데이터베이스와 AWS DMS 함께를 사용하는 것과 관련된 문제 해결에 대해 알아볼 수 있습니다.

주제

- [다른 AWS 리전의 Amazon Redshift 클러스터에 로드](#)
- [오류: 관계 "awsdms_apply_exceptions"가 이미 존재합니다](#)
- [awsdms_changes"로 시작하는 테이블 이름이 있는 오류](#)
- [dms.awsdms_changes000000000XXXX와 같은 이름과 함께 클러스터에 테이블 표시](#)
- [Amazon Redshift 사용에 필요한 권한](#)

다른 AWS 리전의 Amazon Redshift 클러스터에 로드

복제 인스턴스와 다른 AWS 리전의 Amazon Redshift 클러스터에는를 로드할 수 AWS DMS 없습니다. DMS를 사용하려면 복제 인스턴스와 Amazon Redshift 클러스터가 동일한 리전에 있어야 합니다.

오류: 관계 "awsdms_apply_exceptions"가 이미 존재합니다

오류 "관계 'awsdms_apply_exceptions'가 이미 존재합니다"는 흔히 Redshift 엔드포인트가 PostgreSQL 엔드포인트로써 지정될 때 발생합니다. 이 문제를 해결하려면, 엔드포인트를 수정하고 대상 엔진을 "redshift"로 변경합니다.

awsdms_changes"로 시작하는 테이블 이름이 있는 오류

'awsdms_changes'로 시작하는 이름과 관련된 테이블 오류 메시지는 데이터를 동일한 Amazon Redshift 클러스터로 로드하려고 하는 태스크 2개를 동시에 실행할 때 발생합니다. 임시 테이블 이름을 지정하는 방식으로 인해 동일한 테이블을 업데이트할 때 동시 작업이 충돌할 수 있습니다.

dms.awsdms_changes000000000XXXX와 같은 이름과 함께 클러스터에 테이블 표시

AWS DMS 는 Amazon S3에 저장된 파일에서 데이터가 로드될 때 임시 테이블을 생성합니다. 이 임시 테이블의 이름에는 접두사 dms.awsdms_changes가 있습니다. 가 데이터를 처음 로드할 때와 최종 대상 테이블에 배치하기 전에 데이터를 AWS DMS 저장하려면 이러한 테이블이 필요합니다.

Amazon Redshift 사용에 필요한 권한

Amazon Redshift AWS DMS 에서를 사용하려면 Amazon Redshift에 액세스하는 데 사용하는 사용자 계정에 다음 권한이 있어야 합니다.

- CRUD(선택, 삽입, 업데이트, 삭제)
- 대량 로드
- 생성, 변경, 삭제(필요한 경우 태스크의 정의에 따라)

대상으로서 Amazon Redshift를 사용할 때 필요한 모든 사전 요구 사항을 보려면, [Amazon Redshift 데이터베이스를의 대상으로 사용 AWS Database Migration Service](#) 섹션을 참조하세요.

Amazon Aurora MySQL 관련 문제 해결

아래에서 Amazon Aurora MySQL 데이터베이스와 AWS DMS 함께를 사용하는 것과 관련된 문제 해결에 대해 알아볼 수 있습니다.

주제

- [오류: CHARACTER SET UTF8 fields terminated by ',' enclosed by '"' lines terminated by '\n'](#)

오류: CHARACTER SET UTF8 fields terminated by ',' enclosed by '"' lines terminated by '\n'

Amazon Aurora MySQL을 대상으로 사용 중인 경우, 로그에 다음과 같은 오류가 표시될 수 있습니다. 이러한 유형의 오류는 일반적으로 ANSI_QUOTES가 SQL_MODE 파라미터의 일부로 포함되어 있음을 나타냅니다. SQL_MODE 파라미터의 일부로서 ANSI_QUOTES가 포함되어 있으면 큰따옴표가 일반 따옴표로 처리되어 태스크를 실행할 때 문제가 발생할 수 있습니다.

이 오류를 수정하려면, SQL_MODE 파라미터에서 ANSI_QUOTES를 제거합니다.

```
2016-11-02T14:23:48 [TARGET_LOAD ]E: Load data sql statement. load data local infile
"/rdsdbdata/data/tasks/7X04FJHCV0N7TYTLQ6RX3CQH DU/data_files/4/LOAD000001DF.csv" into
table
`VOSPUSER`.`SANDBOX_SRC_FILE` CHARACTER SET UTF8 fields terminated by ','
enclosed by '"' lines terminated by '\n'(`SANDBOX_SRC_FILE_ID`,`SANDBOX_ID`,
`FILENAME`,`LOCAL_PATH`,`LINES_OF_CODE`,`INSERT_TS`,`MODIFIED_TS`,`MODIFIED_BY`,
`RECORD_VER`,`REF_GUID`,`PLATFORM_GENERATED`,`ANALYSIS_TYPE`,`SANITIZED`,`DYN_TYPE`,
`CRAWL_STATUS`,`ORIG_EXEC_UNIT_VER_ID` ) ; (provider_syntax_manager.c:2561)
```

SAP ASE 관련 문제 해결

아래에서 SAP ASE 데이터베이스와 AWS DMS 함께를 사용하는 것과 관련된 문제 해결에 대해 알아볼 수 있습니다.

오류: LOB columns have NULL values when source has a composite unique index with NULL values

NULL 값을 허용하는 복합 고유 인덱스로 구성된 테이블이 있는 소스로서 SAP ASE를 사용할 경우, 복제가 진행되는 동안 LOB 값이 마이그레이션되지 않을 수 있습니다. 일반적으로 이러한 동작은 DMS 복제 인스턴스 클라이언트에서 ANSI_NULL의 기본값이 1로 설정되었을 때 발생합니다.

LOB 필드가 올바르게 마이그레이션되도록 하려면 엔드포인트 설정을 작업 'AnsiNull=0'의 AWS DMS 소스 엔드포인트에 포함합니다.

IBM DB2 관련 문제 해결

아래에서 IBM Db2 데이터베이스와 AWS DMS 함께를 사용하는 것과 관련된 문제 해결에 대해 알아볼 수 있습니다.

오류: Resume from timestamp is not supported Task

지속적 복제(CDC)의 경우 특정 타임스탬프에서 복제를 시작하려는 경우 추가 연결 속성을 필요한 타임스탬프 StartFromContext로 설정합니다. 자세한 내용은 [Db2 LUW 사용 시 추가 연결 속성 \(ECAs\)을 참조하세요](#). StartFromContext를 필수 타임스탬프로 설정하면 아래와 같은 문제를 방지할 수 있습니다.

```
Last Error Resume from timestamp is not supported Task error notification received
from
subtask 0, thread 0 [reptask/replicationtask.c:2822] [1020455] 'Start from timestamp'
was blocked to prevent Replicate from
scanning the log (to find the timestamp). When using IBM DB2 for LUW, 'Start from
timestamp' is only supported if an actual
change was captured by this Replicate task earlier to the specified timestamp.
```

테이블이 "Failed to build 'where' 문" 오류와 함께 테이블을 일시 중지했습니다.

DMS에서 프라이머리 키가 없는 테이블의 레코드를 업데이트하려고 하면 시스템에서 WHERE 조건을 구성할 수 없으며 다음 오류가 표시됩니다.

```
[TARGET_APPLY ]E: Failed to build 'where' statement
```

이는 여러 알려진 문제 또는 제한 사항으로 인해 발생할 수 있으며 다음과 같습니다.

- 프라이머리 키 열은 remove-column 변환 규칙을 사용하여 제거됩니다.
- 테이블 구조가 소스와 대상 데이터베이스 간에 일치하지 않습니다. 즉, 소스에는 존재하지만 대상에는 존재하지 않거나 제거되었을 수 있는 프라이머리 키 열입니다.
- 알려진 제한 사항 또는 누락된 사전 조건:
 - Oracle 테이블에서 보충 로깅이 제대로 활성화되지 않았습니다.
 - 긴 객체 이름(30바이트 이상)으로 생성된 Oracle 테이블이므로 객체 이름은 테이블 또는 열 이름일 수 있습니다.

- Oracle 애플리케이션 컨테이너 PDB에서 복제합니다.

의 지연 시간 문제 해결 AWS Database Migration Service

이 섹션에서는 지속적 복제 단계(CDC) 중 AWS DMS 작업 지연 시간의 일반적인 원인에 대한 개요를 제공합니다. 데이터는 비동기적으로 AWS DMS 복제합니다. 지연 시간은 변경 사항이 소스에서 커밋된 시간과 변경 사항이 대상에 복제된 시간 사이에 지연이 발생하는 것입니다. 다음과 같은 복제 구성 요소의 잘못된 구성으로 인해 지연 시간이 발생할 수 있습니다.

- 소스 엔드포인트 또는 데이터 소스
- 대상 엔드포인트 또는 데이터 소스
- 복제 인스턴스
- 이러한 구성 요소 간의 네트워크

테스트 마이그레이션을 개념 증명으로 사용하여 복제에 대한 정보를 수집하는 것이 좋습니다. 그런 다음, 이 정보를 사용해 복제 구성을 튜닝하여 지연 시간을 최소화할 수 있습니다. 개념 증명 마이그레이션 실행에 대한 자세한 내용은 [개념 증명 실행](#) 섹션을 참조하세요.

주제

- [CDC 지연 시간 유형](#)
- [CDC 지연 시간이 발생하는 일반적인 원인](#)
- [지연 시간 문제 해결](#)

CDC 지연 시간 유형

이 섹션에는 CDC 중에 발생할 수 있는 복제 지연 시간 유형이 나와 있습니다.

소스 지연 시간

소스 엔드포인트에서 캡처한 마지막 이벤트의 커밋 시간과 복제 인스턴스의 현재 시스템 타임스탬프 간의 지연 시간(초)입니다. CDCLatencySource CloudWatch 지표를 사용하여 데이터 소스와 복제 인스턴스 간의 지연 시간을 모니터링할 수 있습니다. CDCLatencySource 지표가 높으면 소스에서 변경 사항을 캡처하는 프로세스가 지연되었음을 나타냅니다. 예를 들어 애플리케이션이 10:00에 소스에 삽입을 커밋하고 10:02에 변경 사항을 AWS DMS 사용하는 경우 CDCLatencySource 지표는 120초입니다.

의 CloudWatch 지표에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [복제 작업 지표](#).

대상 지연 시간

대상에 커밋되기를 대기하는 첫 이벤트의 소스에 대한 커밋 시간과 DMS 복제 인스턴스의 현재 타임스탬프 사이의 지연 시간(초)입니다. CDCLatencyTarget CloudWatch 지표를 사용하여 데이터 소스와 대상 데이터에 대한 커밋 사이의 지연 시간을 모니터링할 수 있습니다. 즉, 소스에서 데이터를 읽는 데 걸리는 모든 지연도 CDCLatencyTarget에 포함됩니다. 따라서 CDCLatencyTarget은 항상 CDCLatencySource보다 크거나 같습니다.

예를 들어 애플리케이션이 10:00에 소스에 삽입을 커밋하고 10:02에 이를 AWS DMS 사용하고 10:05에 대상에 쓰는 경우 CDCLatencyTarget 지표는 300초입니다.

CDC 지연 시간이 발생하는 일반적인 원인

이 섹션에는 CDC 중에 복제 작업 시 발생할 수 있는 지연 시간 유형이 나와 있습니다.

주제

- [엔드포인트 리소스](#)
- [복제 인스턴스 리소스](#)
- [네트워크 속도 및 대역폭](#)
- [DMS 구성](#)
- [복제 시나리오](#)

엔드포인트 리소스

복제 성능 및 지연 시간에 큰 영향을 미치는 요인은 다음과 같습니다.

- 소스 및 대상 데이터베이스 구성
- 인스턴스 크기
- 과소 프로비저닝되었거나 잘못 구성된 소스 또는 대상 데이터 저장소

AWS호스팅된 소스 및 대상의 엔드포인트 문제로 인한 지연 시간의 원인을 식별하려면 다음 CloudWatch 지표를 모니터링하세요.

- FreeMemory

- CPUUtilization
- 처리량 및 I/O 지표(예: WriteIOPS, WriteThroughput 또는 ReadLatency)
- 트랜잭션 볼륨 지표(예CDCIncomingChanges:)

CloudWatch 지표 모니터링에 대한 자세한 내용은 [AWS Database Migration Service 지표](#) 섹션을 참조하세요.

복제 인스턴스 리소스

복제 인스턴스 리소스는 복제에 매우 중요합니다. 이러한 리소스로 인해 소스 및 대상 지연 시간이 모두 발생할 수 있으므로, 리소스 병목 현상이 없는지 확인해야 합니다.

복제 인스턴스에 리소스 병목 현상이 있는지 알아보려면 다음 사항을 확인합니다.

- CPU, 메모리, 초당 I/O, 스토리지와 같은 중요한 CloudWatch 지표에 급격한 증가나 지속적으로 높은 값이 발생하지 않습니다.
- 복제 인스턴스의 크기가 워크로드에 알맞게 적정합니다. 복제 인스턴스의 알맞은 크기 결정에 대한 자세한 내용은 [복제 인스턴스에 가장 적합한 크기 선택](#) 섹션을 참조하세요.

네트워크 속도 및 대역폭

네트워크 대역폭은 데이터 전송에 영향을 미치는 요인입니다. 복제의 네트워크 성능을 분석하려면 다음 작업 중 하나를 수행하세요.

- 인스턴스 수준에서 ReadThroughput 및 WriteThroughput 지표를 확인합니다. CloudWatch 지표 모니터링에 대한 자세한 내용은 [AWS Database Migration Service 지표](#) 섹션을 참조하세요.
- AWS DMS 진단 지원 AMI를 사용합니다. 현재 리전에서 진단 지원 AMI를 사용할 수 없는 경우, 지원되는 임의의 지역에서 이를 다운로드한 후 해당 리전에 복사하여 네트워크 분석을 수행할 수 있습니다. 진단 지원 AMI에 대한 자세한 내용은 [AWS DMS 진단 지원 AMI 작업](#) 섹션을 참조하세요.

의 CDC AWS DMS 는 데이터 일관성을 보장하기 위해 단일 스레드입니다. 따라서 단일 스레드 데이터 전송 속도를 계산하여 네트워크에서 지원 가능한 데이터 볼륨을 결정할 수 있습니다. 예를 들어, 100Mbps(초당 메가비트) 네트워크를 사용하여 태스크를 소스에 연결할 경우 복제에 할당되는 이론상의 최대 대역폭은 12.5Mbps(초당 메가바이트)입니다. 이는 시간당 45기가비트입니다. 소스의 트랜잭션 로그 생성 속도가 시간당 45기가비트보다 크면 태스크에 CDC 지연 시간이 발생했다는 것을 의미합니다. 100Mbps 네트워크의 경우 이러한 속도는 이론상 최대값입니다. 소스 및 대상의 네트워크 트래픽과 리소스 오버헤드 같은 다른 요인으로 인해 실제 사용 가능한 대역폭이 감소합니다.

DMS 구성

이 섹션에는 지연 시간을 줄이는 데 도움이 되는 권장 복제 구성이 나와 있습니다.

- **엔드포인트 설정:** 소스 및 대상 엔드포인트 설정으로 인해 복제 인스턴스의 성능이 저하될 수 있습니다. 리소스를 많이 사용하는 기능을 활성화하는 엔드포인트 설정은 성능에 영향을 미칩니다. 예를 들어, Oracle 엔드포인트의 경우 LogMiner를 비활성화하고 Binary Reader를 사용하면 LogMiner가 리소스를 많이 사용하므로 성능이 향상됩니다. 다음과 같은 엔드포인트 설정은 Oracle 엔드포인트의 성능을 향상합니다.

```
useLogminerReader=N;useBfile=Y
```

엔드포인트 설정에 대한 자세한 내용은 [AWS DMS 엔드포인트 작업](#) 주제의 소스 및 대상 엔드포인트 엔진에 대한 설명서를 참조하세요.

- **태스크 설정:** 특정 복제 시나리오의 일부 태스크 설정으로 인해 복제 인스턴스의 성능이 저하될 수 있습니다. 예를 들어, AWS DMS는 Amazon Redshift를 제외한 모든 엔드포인트의 경우 CDC에 기본적으로 트랜잭션 적용 모드(BatchApplyEnabled=false)를 사용합니다. 하지만 변경 횟수가 많은 소스의 경우 BatchApplyEnabled를 true로 설정하면 성능이 향상될 수 있습니다.

작업 설정에 관한 자세한 내용은 [AWS Database Migration Service 작업에 대한 작업 설정 지정](#) 섹션을 참조하세요.

- **CDC 전용 태스크의 시작 위치:** 과거의 위치 또는 타임스탬프에서 CDC 전용 태스크를 시작하면 CDC 소스 지연 시간이 증가한 상태에서 작업이 시작됩니다. 소스의 변경 사항 볼륨에 따라 태스크 지연 시간이 감소하는 데 시간이 걸릴 수 있습니다.
- **LOB 설정:** 대용량 객체 데이터 유형은 대용량 바이너리 데이터를 복제하는 방식으로 인해 AWS DMS 복제 성능을 저해할 수 있습니다. 자세한 내용은 다음 항목을 참조하세요.
 - [AWS DMS 작업의 소스 데이터베이스에 대한 LOB 지원 설정](#)
 - [대용량 이진 객체\(LOB\) 마이그레이션.](#)

복제 시나리오

이 섹션에서는 특정 복제 시나리오를 살펴보고 이러한 시나리오가 지연 시간에 어떤 영향을 미칠 수 있는지 설명합니다.

주제

- [장기간 태스크 중지](#)

- [캐시된 변경 사항](#)
- [교차 리전 복제](#)

장기간 태스크 중지

작업을 중지하면 소스에서 읽은 마지막 트랜잭션 로그의 위치를 AWS DMS 저장합니다. 작업을 재개하면 DMS는 동일한 트랜잭션 로그 위치에서 읽기 작업을 계속 수행하려고 합니다. 몇 시간 또는 며칠 후에 작업을 재개하면 DMS가 트랜잭션 백로그 사용을 마칠 때까지 CDC 소스 지연 시간이 증가합니다.

캐시된 변경 사항

캐시된 변경 사항은 애플리케이션이 전체 로드 복제 단계를 AWS DMS 실행하는 동안 데이터 소스에 기록하는 변경 사항입니다. DMS는 전체 로드 단계가 완료되고 CDC 단계가 시작될 때까지 이러한 변경 사항을 적용하지 않습니다. 트랜잭션 수가 많은 소스의 경우, 캐시된 변경 사항을 적용하는 데 시간이 오래 걸리므로 CDC 단계가 시작되면 소스 지연 시간이 증가합니다. 캐시된 변경 사항 수를 최소화하려면 트랜잭션 볼륨이 적을 때 전체 로드 단계를 실행하는 것이 좋습니다.

교차 리전 복제

DMS 엔드포인트 또는 복제 인스턴스를 서로 다른 AWS 리전에 배치하면 네트워크 지연 시간이 늘어납니다. 이 경우 복제 지연 시간도 증가합니다. 최상의 성능을 위해 동일한 AWS 리전에서 소스 엔드포인트, 대상 엔드포인트 및 복제 인스턴스를 찾습니다.

지연 시간 문제 해결

이 섹션에는 복제 지연 시간에 대한 문제 해결 단계가 나와 있습니다.

지연 시간 문제를 해결하려면 다음을 수행합니다.

- 먼저, 태스크의 지연 시간 유형과 양을 확인합니다. DMS 콘솔 또는 CLI에서 태스크의 테이블 통계 섹션을 확인합니다. 카운터가 변경되면 데이터 전송이 진행 중인 것입니다. CDCLatencySource 및 CDCLatencyTarget 지표를 함께 확인하여 CDC 중에 병목 현상이 발생했는지 확인합니다.
- 높은 CDCLatencySource 또는 CDCLatencyTarget 지표가 복제에서 병목 현상이 발생했음을 나타내는 경우 다음 사항을 확인하세요.
 - CDCLatencySource가 높고 CDCLatencyTarget와 같으면 소스 엔드포인트에 병목 현상이 있고 대상에 데이터를 원활하게 쓰 AWS DMS 고 있음을 CDCLatencySource나타냅니다. 다음을 [소스 지연 시간 문제 해결](#) 섹션을 참조하세요.

- CDCLatencySource가 낮고 CDCLatencyTarget가 높으면 대상 엔드포인트에 병목 현상이 있고 소스에서 데이터를 원활하게 AWS DMS 읽고 있음을 나타냅니다. 아래의 [대상 지연 시간 문제 해결](#) 섹션을 참조하세요.
- CDCLatencySource가 높고 CDCLatencyTarget가 CDCLatencySource보다 훨씬 더 높은 경우, 소스 읽기와 대상 쓰기 양쪽 모두에 병목 현상이 발생했음을 나타냅니다. 먼저 소스 지연 시간을 검사한 다음, 대상 지연 시간을 검사합니다.

DMS 태스크 지표의 모니터링에 대한 자세한 내용은 [AWS DMS 작업 모니터링](#) 섹션을 참조하세요.

소스 지연 시간 문제 해결

아래의 주제에서는 소스 엔드포인트 유형별 복제 시나리오를 설명합니다.

주제

- [Oracle 엔드포인트 문제 해결](#)
- [MySQL 엔드포인트 문제 해결](#)
- [PostgreSQL 엔드포인트 문제 해결](#)
- [SQL Server 엔드포인트 문제 해결](#)

Oracle 엔드포인트 문제 해결

이 섹션에는 Oracle과 관련된 복제 시나리오가 나와 있습니다.

소스 읽기 일시 중지

AWS DMS 는 다음 시나리오에서 Oracle 소스에서 읽기를 일시 중지합니다. 이 동작은 설계에 따른 것입니다. 태스크 로그를 사용하여 이 문제의 원인을 조사할 수 있습니다. 태스크 로그에서 다음과 비슷한 메시지를 찾아보세요. 태스크 로그에 대한 자세한 내용은 [AWS DMS 작업 로그 보기 및 관리](#) 섹션을 참조하세요.

- SORTER 메시지: DMS가 복제 인스턴스에서 트랜잭션을 캐싱하고 있음을 나타냅니다. 자세한 내용은 [태스크 로그의 SORTER 메시지](#) 섹션을 참조하세요.
- 디버그 태스크 로그: DMS가 읽기 프로세스를 중단할 경우, 태스크는 컨텍스트 필드 또는 타임스탬프를 변경하지 않고 다음과 같은 메시지를 디버그 태스크 로그에 반복적으로 기록합니다.
- Binary reader:

```
[SOURCE_CAPTURE ]T: Produce CTI event:
```

```
context '00000020.f23ec6e5.00000002.000a.00.0000:190805.3477731.16'
xid [00000000001e0018] timestamp '2021-07-19 06:57:55'
thread 2 (oradcdc_oralog.c:817)
```

- Logminer:

```
[SOURCE_CAPTURE ]T: Produce INSERT event:
object id 1309826 context
'000000000F2CECAA010000010005A8F500000275016C0000000000000F2CEC58'
xid [000014e06411d996] timestamp '2021-08-12 09:20:32' thread 1
(oradcdc_reader.c:2269)
```

- AWS DMS 는 모든 새 다시 실행 또는 아카이브된 로그 작업에 대해 다음 메시지를 기록합니다.

```
00007298: 2021-08-13T22:00:34 [SOURCE_CAPTURE ]I: Start processing archived
Redo log sequence 14850 thread 2 name XXXXX/XXXXX/ARCHIVELOG/2021_08_14/
thread_2_seq_14850.22977.1080547209 (oradcdc_redo.c:754)
```

소스에 새 다시 실행 또는 아카이브된 로그 작업이 있고 AWS DMS 가 이러한 메시지를 로그에 쓰지 않는 경우 이는 작업이 이벤트를 처리하지 않음을 의미합니다.

높은 재실행 생성률

태스크가 재실행 로그 또는 아카이브된 로그를 처리하는 중인데도 소스 지연 시간이 여전히 높은 경우, 재실행 로그 생성률 및 생성 패턴을 파악해 보세요. 재실행 로그 생성 수준이 높을 경우, 복제된 테이블과 관련된 변경 사항을 가져오기 위해 태스크가 모든 재실행 로그 및 아카이브 로그를 읽으므로 소스 지연 시간이 증가합니다.

재실행 생성률을 확인하려면 아래의 쿼리를 사용합니다.

- 일별 재실행 생성률:

```
select trunc(COMPLETION_TIME,'DD') Day, thread#,
round(sum(BLOCKS*BLOCK_SIZE)/1024/1024/1024) GB,
count(*) Archives_Generated from v$archived_log
where completion_time > sysdate- 1
group by trunc(COMPLETION_TIME,'DD'),thread# order by 1;
```

- 시간당 재실행 생성률:

```
Alter session set nls_date_format = 'DD-MON-YYYY HH24:MI:SS';
```

```
select trunc(COMPLETION_TIME,'HH') Hour,thread# ,
round(sum(BLOCKS*BLOCK_SIZE)/1024/1024) "REDO PER HOUR (MB)",
count(*) Archives from v$archived_log
where completion_time > sysdate- 1
group by trunc(COMPLETION_TIME,'HH'),thread# order by 1 ;
```

이 시나리오의 지연 시간 문제를 해결하려면 다음 사항을 확인하세요.

- 복제의 네트워크 대역폭과 단일 스레드 성능을 확인하여 기본 네트워크가 소스 재실행 생성물을 지원할 수 있는지 확인합니다. 네트워크 대역폭이 복제 성능에 어떤 영향을 미치는지에 대한 자세한 내용은 이전의 [네트워크 속도 및 대역폭](#) 섹션을 참조하세요.
- 보충 로깅을 올바르게 설정했는지 확인합니다. 소스에 대한 추가 로깅(예: 테이블의 모든 열에 대한 로깅 활성화)을 사용하지 않아야 합니다. 보충 로깅 설정에 대한 자세한 내용은 [보충 로깅 설정](#) 섹션을 참조하세요.
- 올바른 API를 사용하여 재실행 로그 또는 아카이빙된 로그를 읽고 있는지 확인합니다. Oracle LogMiner 또는 AWS DMS Binary Reader를 사용할 수 있습니다. LogMiner는 온라인 재실행 로그와 아카이브된 재실행 로그 파일을 읽는 반면, Binary Reader는 원시 재실행 로그 파일을 직접 읽고 구문 분석합니다. 따라서 Binary Reader의 성능이 더 높습니다. 재실행 로그 생성물이 시간당 10GB를 초과할 경우 Binary Reader를 사용하는 것이 좋습니다. 자세한 내용은 [CDC용 Oracle LogMiner 또는 AWS DMS Binary Reader 사용](#) 단원을 참조하십시오.
- ArchivedLogsOnly를 Y로 설정했는지 확인합니다. 이 엔드포인트 설정이 설정되어 있으면 AWS DMS 는 아카이브된 재실행 로그에서 읽기를 수행합니다. 이렇게 하면가 읽기 전에 온라인 다시 실행 로그가 보관될 AWS DMS 때까지 기다리기 때문에 소스 지연 시간이 증가합니다. 자세한 내용은 [ArchivedLogsOnly](#) 섹션을 참조하세요.
- Oracle 소스가 Automatic Storage Management(ASM)를 사용하는 경우, 데이터 저장소를 올바르게 구성하는 방법에 대한 자세한 내용은 [Oracle을의 소스로 사용할 때 Oracle ASM에 REDO 저장 AWS DMS](#) 섹션을 참조하세요. 추가 연결 속성(ECA) asmUsePLSQLArray를 사용하여 읽기 성능을 더욱 최적화할 수도 있습니다. asmUsePLSQLArray 사용에 대한 자세한 내용은 [Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#)을 참조하세요.

MySQL 엔드포인트 문제 해결

이 섹션에는 MySQL과 관련된 복제 시나리오가 포함되어 있습니다.는 MySQL 바이너리 로그를 주기적으로 AWS DMS 스캔하여 변경 사항을 복제합니다. 다음과 같은 시나리오에서는 이러한 프로세스 때문에 지연 시간이 증가할 수 있습니다.

주제

- [소스에서 장시간 실행 중인 트랜잭션](#)
- [소스의 높은 워크로드](#)
- [바이너리 로그 경합](#)

소스에서 장시간 실행 중인 트랜잭션

MySQL은 커밋된 트랜잭션만 바이너리 로그에 작성하므로, 트랜잭션이 장시간 실행되면 쿼리 실행 시간에 비례하여 지연 시간이 급증합니다.

장시간 실행 중인 트랜잭션을 파악하려면 아래의 쿼리를 사용하거나 느린 쿼리 로그를 사용합니다.

```
SHOW FULL PROCESSLIST;
```

느린 쿼리 로그 사용에 대한 자세한 내용은 [MySQL 설명서](#)의 [The Slow Query Log](#)를 참조하세요.

장시간 실행 중인 트랜잭션으로 인한 지연 시간 급증을 방지하려면 소스 트랜잭션을 재구성하여 쿼리 실행 시간을 줄이거나 커밋 빈도를 늘립니다.

소스의 높은 워크로드

DMS CDC는 단일 스레드 방식이므로, 트랜잭션 수가 많으면 소스 지연 시간이 증가할 수 있습니다. 소스 지연 시간이 과중한 워크로드로 인한 것인지 확인하려면, 지연 시간 동안 생성된 바이너리 로그의 수와 크기를 지연 시간 이전에 생성된 로그와 비교합니다. 바이너리 로그와 DMS CDC 스레드 상태를 확인하려면 아래의 쿼리를 사용합니다.

```
SHOW BINARY LOGS;  
SHOW PROCESSLIST;
```

CDC 바이너리 로그 덤프 스레드 상태에 대한 자세한 내용은 [Replication Source Thread States](#)를 참조하세요.

소스에서 생성된 최신 바이너리 로그 위치를 DMS가 현재 처리 중인 이벤트와 비교하여 지연 시간을 확인할 수 있습니다. 소스 최신 바이너리 로그를 확인하려면 다음을 수행합니다.

- SOURCE_CAPTURE 구성 요소에 대한 디버그 로그를 활성화합니다.
- 태스크 디버그 로그에서 DMS 처리 바이너리 로그와 위치 세부 정보를 검색합니다.
- 소스에 대한 최신 바이너리 로그를 확인하려면 아래의 쿼리를 사용합니다.

```
SHOW MASTER STATUS;
```

성능을 더 최적화하려면 `EventsPollInterval`을 튜닝합니다. 기본적으로 DMS는 5초마다 바이너리 로그를 폴링하지만, 이 값을 줄여 성능을 개선할 수 있습니다. `EventsPollInterval` 설정에 관한 자세한 내용은 [MySQL을의 소스로 사용할 때 엔드포인트 설정 AWS DMS](#)를 참조하세요.

바이너리 로그 경합

데이터의 양이 많은 여러 테이블을 마이그레이션할 경우, MySQL 5.7.2 이상에서는 테이블을 별도의 태스크로 분할하는 것이 좋습니다. MySQL 버전 5.7.2 이상에서는 마스터 덤프 스레드가 잠금 경합을 줄이고 처리량을 개선합니다. 따라서 덤프 스레드는 이벤트를 읽을 때마다 더 이상 바이너리 로그를 잠그지 않습니다. 이렇게 되면 여러 덤프 스레드가 바이너리 로그 파일을 동시에 읽을 수 있습니다. 이는 클라이언트가 바이너리 로그에 쓰는 동안 덤프 스레드가 바이너리 로그를 읽을 수 있다는 의미이기도 합니다. 덤프 스레드에 대한 자세한 내용은 [Replication Threads](#) 및 [MySQL 5.7.2 릴리스 정보](#)를 참조하세요.

5.7.2 이전 버전의 MySQL 소스의 복제 성능을 개선하려면 CDC 구성 요소로 작업을 통합해 보시기 바랍니다.

PostgreSQL 엔드포인트 문제 해결

이 섹션에는 PostgreSQL과 관련된 복제 시나리오가 나와 있습니다.

주제

- [소스에서 장시간 실행 중인 트랜잭션](#)
- [소스의 높은 워크로드](#)
- [높은 네트워크 처리량](#)
- [Aurora PostgreSQL에서 파일 유출](#)

소스에서 장시간 실행 중인 트랜잭션

소스 데이터베이스에 장시간 실행 중인 트랜잭션이 있는 경우(예: 단일 트랜잭션에 수천 개의 삽입이 있는 경우), 트랜잭션이 완료될 때까지 DMS CDC 이벤트 및 트랜잭션 카운터가 증가하지 않습니다. 이러한 지연으로 인해 `CDCLatencyTarget` 지표로 측정 가능한 지연 시간 문제가 발생할 수 있습니다.

장시간 실행 중인 트랜잭션을 검토하려면 다음 중 하나를 수행합니다.

- `pg_replication_slots` 보기를 사용합니다. `restart_lsn` 값이 업데이트되지 않을 경우, 장시간 실행 중인 활성 트랜잭션으로 인해 PostgreSQL이 Write Ahead Log(WAL)를 릴리스하지 못할 수 있습니다. `pg_replication_slots` 보기에 대한 자세한 내용은 [PostgreSQL 15.4 설명서의 `pg\_replication\_slots` 섹션](#)을 참조하세요.
- 아래의 쿼리를 사용하여 데이터베이스의 모든 활성 쿼리 목록을 관련 정보와 함께 반환합니다.

```
SELECT pid, age(clock_timestamp(), query_start), username, query
FROM pg_stat_activity WHERE query != '<IDLE>'
AND query NOT ILIKE '%pg_stat_activity%'
ORDER BY query_start desc;
```

쿼리 결과의 `age` 필드에는 각 쿼리의 활성 기간이 표시되므로, 장시간 실행 중인 쿼리를 확인하는 데 사용할 수 있습니다.

소스의 높은 워크로드

소스에 PostgreSQL의 워크로드가 높은 경우, 다음 사항을 확인하여 지연 시간을 줄입니다.

- 초당 트랜잭션(TPS) 값이 높은 소스 데이터베이스에서 테이블의 하위 집합을 마이그레이션하는 동안 `test_decoding` 플러그인을 사용하면 지연 시간이 길어질 수 있습니다. 그 이유는 `test_decoding` 플러그인이 모든 데이터베이스 변경 사항을 복제 인스턴스로 보낸 다음, DMS가 태스크의 테이블 매핑을 기반으로 필터링하기 때문입니다. 태스크의 테이블 매핑에 포함되지 않은 테이블의 이벤트는 소스 지연 시간을 증가시킬 수 있습니다.
- 다음 방법 중 하나를 사용하여 TPS 처리량을 확인합니다.
 - Aurora PostgreSQL 소스의 경우 CommitThroughput CloudWatch 지표를 사용합니다.
 - Amazon RDS 또는 온프레미스에서 실행되는 PostgreSQL의 경우, PSQL 클라이언트 버전 11 이상을 활용하여 아래의 쿼리를 사용합니다(결과를 진행하려면 쿼리 중에 **enter**를 누릅니다).

```
SELECT SUM(xact_commit)::numeric as temp_num_tx_ini FROM pg_stat_database; \gset
select pg_sleep(60);
SELECT SUM(xact_commit)::numeric as temp_num_tx_final FROM pg_stat_database; \gset
select (:temp_num_tx_final - :temp_num_tx_ini)/ 60.0 as "Transactions Per Second";
```

- `test_decoding` 플러그인 사용 시 지연 시간을 줄으려면 `pglogical` 플러그인을 대신 사용하는 것이 좋습니다. `test_decoding` 플러그인과 달리, `pglogical` 플러그인은 원본에서 Write Ahead Log(WAL) 변경 사항을 필터링하고 관련 변경 사항만 복제 인스턴스로 전송합니다. 여기서 `pglogical` 플러그인을 사용하는 방법에 대한 자세한 내용은 [섹션을 AWS DMS참조하세요](#) [pglogical 플러그인 구성](#).

높은 네트워크 처리량

test_decoding 플러그인을 사용할 경우, 특히 대용량 트랜잭션 시 복제를 수행할 때 네트워크 대역폭 사용량이 높을 수 있습니다. 그 이유는 test_decoding 플러그인은 변경 사항을 처리한 후 이를 사람이 읽을 수 있는 형식으로 변환하는데, 원래 바이너리 형식보다 용량이 크기 때문입니다.

성능을 개선하려면 바이너리 플러그인인 pglogical 플러그인을 대신 사용하는 것이 좋습니다. test_decoding 플러그인과 달리 pglogical 플러그인은 바이너리 형식의 출력을 생성하므로, 압축된 Write Ahead Log(WAL) 스트림 변경 사항이 발생합니다.

Aurora PostgreSQL에서 파일 유출

PostgreSQL 버전 13 이상에서 logical_decoding_work_mem 파라미터는 디코딩 및 스트리밍을 위한 메모리 할당을 결정합니다. logical_decoding_work_mem 파라미터에 대한 자세한 내용은 [PostgreSQL 13.13 설명서](#)에서 [PostgreSQL의 리소스 소비](#)를 참조하세요.

논리적 복제는 트랜잭션이 커밋될 때까지 메모리의 모든 트랜잭션에 대한 변경 사항을 누적합니다. 모든 트랜잭션에 저장된 데이터의 양이 데이터베이스 파라미터 logical_decoding_work_mem에 지정된 양을 초과하는 경우 DMS는 트랜잭션 데이터를 디스크에 유출하여 새 디코딩 데이터에 대한 메모리를 릴리스합니다.

오래 실행되는 트랜잭션 또는 많은 하위 트랜잭션으로 인해 DMS에서 논리적 디코딩 메모리를 사용할 수 있습니다. 이렇게 메모리 사용량이 증가하면 DMS가 디스크에 유출 파일을 생성하여 복제 중에 소스 지연 시간이 높아집니다.

소스 워크로드 증가의 영향을 줄이려면 다음을 수행합니다.

- 장기 실행 트랜잭션을 줄입니다.
- 하위 트랜잭션 수를 줄입니다.
- 단일 트랜잭션에서 전체 테이블을 삭제하거나 업데이트하는 등 대량의 로그 레코드 버스트를 생성하는 작업을 수행하지 마세요. 대신 더 작은 배치로 작업을 수행합니다.

다음 CloudWatch 지표를 사용하여 소스의 워크로드를 모니터링할 수 있습니다.

- TransactionLogsDiskUsage: 논리적 WAL이 현재 점유하는 바이트 수입니다. 논리적 복제 슬롯이 새 쓰기 속도를 따라갈 수 없거나 오래 실행되는 트랜잭션으로 인해 오래된 파일의 폐영역 회수가 불가능한 경우 이 값은 단조롭게 증가합니다.
- ReplicationSlotDiskUsage: 논리적 복제 슬롯이 현재 사용하는 디스크 공간의 양입니다.

logical_decoding_work_mem 파라미터를 조정하여 소스 지연 시간을 줄일 수 있습니다. 이 파라미터의 기본값은 64MB입니다. 이 파라미터는 각 논리적 스트리밍 복제 연결에 사용되는 메모리의 양을 제한합니다. DMS가 디스크에 쓰는 디코딩된 변경의 양을 줄이려면 logical_decoding_work_mem 값을 work_mem 값보다 크게 설정하는 것이 좋습니다.

특히 마이그레이션 활동이나 지연 시간이 많은 기간에는 유출 파일을 정기적으로 확인하는 것이 좋습니다. DMS가 상당한 수의 유출 파일을 생성하는 경우 논리적 디코딩이 효율적으로 작동하지 않아 지연 시간이 늘어날 수 있습니다. 이를 완화하려면 logical_decoding_work_mem 파라미터 값을 늘리세요.

aurora_stat_file 함수를 사용하여 현재 트랜잭션 오버플로를 확인할 수 있습니다. 자세한 내용은 Amazon Relational Database Service 개발자 안내서의 [논리적 디코딩을 위한 작업 메모리 조정](#)을 참조하세요.

SQL Server 엔드포인트 문제 해결

이 섹션에는 SQL Server와 관련된 복제 시나리오가 나와 있습니다. SQL 서버에서 복제할 변경 사항을 확인하려면 트랜잭션 로그를 AWS DMS 읽고 소스 데이터베이스에서 주기적 스캔을 실행합니다. 일반적으로 복제 지연 시간이 발생하는 이유는 리소스 제약으로 인해 SQL Server에서 이러한 스캔을 스로틀링해서 그렇습니다. 짧은 시간 내에 트랜잭션 로그에 작성되는 이벤트 수가 크게 증가하기 때문일 수도 있습니다.

주제

- [인덱스 재구축](#)
- [대규모 트랜잭션](#)
- [Amazon RDS SQL 서버의 MS-CDC 폴링 간격이 잘못 구성된 경우](#)
- [동일한 소스 데이터베이스에서 복제되는 여러 CDC 태스크](#)
- [RDS for SQL Server에 대한 트랜잭션 로그 백업 처리](#)

인덱스 재구축

SQL Server는 큰 인덱스를 재구축할 때 단일 트랜잭션을 사용합니다. 이로 인해 많은 이벤트가 생성되며, SQL Server가 한 번에 여러 인덱스를 재구축할 경우 많은 양의 로그 공간을 모두 사용하게 될 수 있습니다. 이렇게 되면 일시적인 복제 급증이 발생할 수 있습니다. SQL Server 소스에서 지속적으로 로그 급증이 발생할 경우 다음 사항을 확인하세요.

- 먼저, CDCLatencySource 및 CDCLatencySource CloudWatch 지표를 사용하거나 태스크 로그의 처리량 모니터링 메시지를 확인하여 지연 시간이 급증하는 기간을 확인합니다. 에 대한 CloudWatch 지표에 대한 자세한 내용은 섹션을 AWS DMS참조하세요 [복제 작업 지표](#).
- 지연 시간이 급증하는 동안 활성 트랜잭션 로그 또는 로그 백업의 크기가 증가했는지 확인합니다. 해당 기간에 유지 관리 작업이나 재구축이 실행되었는지도 확인합니다. 트랜잭션 로그 크기 확인에 대한 자세한 내용은 [SQL Server 기술 설명서](#)의 [Monitor log space use](#) 섹션을 참조하세요.
- 유지 관리 계획이 SQL Server 모범 사례를 준수하는지 확인합니다. SQL Server 유지 관리 모범 사례에 대한 자세한 내용은 [SQL Server 기술 설명서](#)의 [Index maintenance strategy](#) 섹션을 참조하세요.

인덱스 재구축 중 지연 시간 문제를 해결하려면 다음을 수행합니다.

- 오프라인 재구축에 BULK_LOGGED 복구 모델을 사용하면 태스크가 처리해야 하는 이벤트를 줄일 수 있습니다.
- 가능한 경우, 인덱스 재구축 중에는 태스크를 중지합니다. 또는 사용량이 많지 않은 시간에 인덱스 재구축을 예약하여 지연 시간 급증으로 인한 영향을 완화하시기 바랍니다.
- 디스크 지연 시간 또는 I/O 처리량처럼 DMS 읽기 속도를 저하시키는 리소스 병목 현상을 확인하고 이를 해결합니다.

대규모 트랜잭션

이벤트가 많은 트랜잭션 또는 장시간 실행 중인 트랜잭션으로 인해 트랜잭션 로그가 커집니다. 이 경우 DMS 읽기 시간이 길어져 지연 시간이 발생합니다. 이는 인덱스 재구축이 복제 성능에 미치는 영향과 유사합니다.

소스 데이터베이스의 일반적인 워크로드에 익숙하지 않다면 이 문제를 확인하기 어려울 수 있습니다. 이 문제를 해결하려면 다음을 수행합니다.

- 먼저, ReadThroughput 및 WriteThroughput CloudWatch 지표를 사용하거나 태스크 로그의 처리량 모니터링 메시지를 확인하여 지연 시간이 급증한 시간을 파악합니다.
- 지연 시간이 급증했을 때 소스 데이터베이스에 장시간 실행 중인 쿼리가 있는지 확인합니다. 장시간 실행 중인 쿼리에 대한 자세한 내용은 [SQL Server 기술 설명서](#)의 [Troubleshoot slow-running queries in SQL Server](#) 섹션을 참조하세요.
- 활성 트랜잭션 로그 또는 로그 백업의 크기가 증가했는지 확인합니다. 자세한 내용은 [SQL Server 기술 설명서](#)의 [Monitor log space use](#) 섹션을 참조하세요.

이 문제를 해결하려면 다음 중 하나를 수행합니다.

- 가장 좋은 해결 방법은 애플리케이션 측에서 트랜잭션을 재구성하여 트랜잭션이 빠르게 완료되도록 하는 것입니다.
- 트랜잭션을 재구성할 수 없는 경우, 단기적인 해결 방법은 디스크 대기 또는 CPU 경합과 같은 리소스 병목 현상이 있는지 확인하는 것입니다. 소스 데이터베이스에서 병목 현상이 발견되면 소스 데이터베이스의 디스크, CPU, 메모리 리소스를 늘려 지연 시간을 줄일 수 있습니다. 이렇게 하면 시스템 리소스 경합이 감소하여 DMS 쿼리를 더 빨리 완료할 수 있습니다.

Amazon RDS SQL 서버의 MS-CDC 폴링 간격이 잘못 구성된 경우

Amazon RDS 인스턴스에서 폴링 간격 설정이 잘못 구성되면 트랜잭션 로그가 증가할 수 있습니다. 복제로 인해 로그 잘림이 방지되기 때문입니다. 실행 중인 태스크를 지연 시간을 최소화하여 계속 복제할 수 있긴 하지만, 태스크를 중지 및 재개하거나 CDC 전용 작업을 시작할 경우 태스크가 실패할 수 있습니다. 대용량 트랜잭션 로그를 스캔하는 동안 시간 초과가 발생하기 때문입니다.

잘못 구성된 폴링 간격 문제를 해결하려면 다음을 수행합니다.

- 활성 트랜잭션 로그 크기가 증가하고 있는지, 그리고 로그 사용량이 100%에 가까운지 확인합니다. 자세한 내용은 [SQL Server 기술 설명서](#)의 [Monitor log space use](#) 섹션을 참조하세요.
- log_reuse_wait_desc value의 원인이 REPLICATION이기 때문에 로그 잘림이 지연되고 있는 것인지 확인합니다. 자세한 내용은 [SQL Server 기술 설명서](#)의 [The Transaction Log \(SQL Server\)](#) 섹션을 참조하세요.

이전 목록의 주제에서 문제가 발견될 경우, MS-CDC 폴링 간격을 튜닝합니다. 폴링 간격 튜닝에 대한 자세한 내용은 [RDS for SQL Server를 소스로 사용할 때 권장되는 설정 AWS DMS](#) 섹션을 참조하세요.

동일한 소스 데이터베이스에서 복제되는 여러 CDC 태스크

전체 로드 단계에서는 태스크 전체에서 테이블을 분할하여 성능을 개선하고, 종속 테이블을 논리적으로 분리하고, 태스크 실패로 인한 영향을 완화하는 것이 좋습니다. 하지만 CDC 단계에서는 태스크를 통합하여 DMS 스캔을 최소화하는 것이 바람직합니다. CDC 단계에서 각 DMS 태스크는 1분에 여러 번 트랜잭션 로그를 스캔하여 새 이벤트가 있는지 확인합니다. 각 태스크는 독립적으로 실행되므로 모든 태스크는 각각의 트랜잭션 로그를 개별적으로 스캔합니다. 이 경우 소스 SQL Server 데이터베이스의 디스크 및 CPU 사용량이 증가합니다. 따라서 많은 태스크를 동시에 실행하면 SQL Server가 DMS 읽기 속도를 제한할 수 있으므로, 지연 시간이 증가하게 됩니다.

여러 태스크가 점진적으로 시작될 경우, 이 문제를 확인하기 어려울 수 있습니다. 이 문제의 가장 일반적인 증상은 대부분의 태스크 스캔에 소요되는 시간이 더 길어지기 시작한다는 것입니다. 이 경우 이러한 스캔의 지연 시간이 길어집니다. SQL Server는 몇 가지 태스크 스캔의 우선 순위를 지정하므로, 일부 태스크에서는 정상적인 지연 시간이 표시됩니다. 이 문제를 해결하려면 모든 태스크에 대한 CDCLatencySource 지표를 확인합니다. 일부 태스크는 CDCLatencySource가 증가하는 반면, 일부 태스크는 CDCLatencySource가 낮을 경우 SQL Server가 일부 태스크의 DMS 읽기 속도를 스로틀링하고 있을 가능성이 높습니다.

CDC 중에 SQL Server가 태스크 읽기 속도를 스로틀링할 경우, 태스크를 통합하여 DMS 스캔 수를 최소화합니다. 경합을 생성하지 않고 소스 데이터베이스에 연결할 수 있는 최대 태스크 수는 소스 데이터베이스 용량, 트랜잭션 로그 증가율, 테이블 수 등과 같은 여러 요인에 따라 달라집니다. 복제 시나리오에 적합한 태스크 수를 확인하려면 프로덕션 환경과 비슷한 테스트 환경에서 복제를 테스트하세요.

RDS for SQL Server에 대한 트랜잭션 로그 백업 처리

AWS DMS 3.5.3 이상에서는 RDS for SQL Server 로그 백업에서 복제를 지원합니다. RDS 인스턴스의 백업 로그에서 이벤트를 복제하는 것은 활성 트랜잭션 로그에서 이벤트를 복제하는 것보다 느립니다. 이는 DMS가 트랜잭션 시퀀스를 유지하고 Amazon RDS 인스턴스 스토리지 채우기 위험을 최소화하기 위해 백업에 대한 액세스를 직렬로 요청하기 때문입니다. 또한 Amazon RDS 종료 시 DMS에 백업을 제공하는 데 걸리는 시간은 로그 백업 크기와 RDS for SQL Server 인스턴스의 로드와 따라 달라집니다.

이러한 제약으로 인해 ECA ActivateSafeguard를 true로 설정하는 것이 좋습니다. 이렇게 하면 DMS 작업이 활성 트랜잭션 로그에서 읽는 동안 트랜잭션이 백업되지 않습니다. 또한 이 설정은 DMS가 백업에서 트랜잭션을 읽을 때 활성 로그의 Amazon RDS 아카이빙 트랜잭션을 방지하여 DMS가 활성 로그를 따라잡을 수 없는 가능성을 제거합니다. 하지만 이렇게 하면 태스크를 따라잡는 동안 활성 로그 크기가 증가할 수 있습니다. 인스턴스에 공간이 부족하지 않도록 인스턴스에 충분한 스토리지가 있는지 확인합니다.

RDS for SQL Server 소스에서 복제하는 CDC 전용 작업의 경우 가능하면 네이티브 CDC 시작 시간 동안 네이티브 CDC 시작 위치를 사용합니다. 이는 DMS가 기본 시작 시간을 지정할 때 개별 로그 백업을 스캔하는 대신 시스템 테이블에 의존하여 기본 시작 위치의 시작점을 식별하기 때문입니다.

대상 지연 시간 문제 해결

이 섹션에는 대상 지연 시간에 영향을 미칠 수 있는 시나리오가 나와 있습니다.

주제

- [인덱싱 문제](#)

- [태스크 로그의 SORTER 메시지](#)
- [데이터베이스 잠금](#)
- [느린 LOB 조회](#)
- [다중 AZ, 감사 로깅, 백업](#)

인덱싱 문제

CDC 단계에서는 대상에서 DML 문(삽입, 업데이트 및 삭제)을 실행하여 소스의 변경 사항을 AWS DMS 복제합니다. DMS를 사용하는 이기종 마이그레이션의 경우, 소스와 대상의 인덱스 최적화 차이로 인해 대상에 쓰기 작업을 수행하는 데 더 오랜 시간이 걸릴 수 있습니다. 이 경우 대상 지연 시간 및 성능 문제가 발생합니다.

인덱싱 문제를 해결하려면 다음을 수행합니다. 이러한 단계의 절차는 데이터베이스 엔진마다 다릅니다.

- 대상 데이터베이스의 쿼리 시간을 모니터링합니다. 대상과 소스의 쿼리 실행 시간을 비교하면 최적화가 필요한 인덱스를 나타낼 수 있습니다.
- 느리게 실행 중인 쿼리에 대한 로깅을 활성화합니다.

장시간 실행 중인 복제의 인덱싱 문제를 해결하려면 다음을 수행합니다.

- 소스 및 대상 데이터베이스의 인덱스를 튜닝하여 소스와 대상의 쿼리 실행 시간이 비슷해지도록 조정합니다.
- 소스와 대상의 DML 쿼리에 사용된 보조 인덱스를 비교합니다. 대상의 DML 성능이 소스 DML 성능과 비슷하거나 더 나은지 확인합니다.

참고로, 인덱스를 최적화하는 절차는 데이터베이스 엔진에 따라 다릅니다. 소스 및 대상 인덱스를 튜닝할 수 있는 DMS 기능은 없습니다.

태스크 로그의 SORTER 메시지

대상 엔드포인트가 AWS DMS 에서 기록하는 변경 볼륨을 따라잡을 수 없는 경우 작업은 복제 인스턴스에 변경 사항을 캐싱합니다. 캐시가 내부 임계값보다 커지면 태스크는 소스에서 추가 변경 사항을 읽는 것을 중단합니다. DMS는 복제 인스턴스의 스토리지가 부족해지거나, 대기 중인 대량의 이벤트를 읽는 동안 태스크가 중단되는 것을 방지하기 위해 이러한 조치를 수행합니다.

이 문제를 해결하려면 CloudWatch 로그에서 다음 중 하나와 유사한 메시지가 있는지 확인합니다.

```
[SORTER ]I: Reading from source is paused. Total disk usage exceeded the limit 90%
(sorter_transaction.c:110)
[SORTER ]I: Reading from source is paused. Total storage used by swap files exceeded
the limit 1048576000 bytes (sorter_transaction.c:110)
```

로그에 첫 번째 메시지와 유사한 메시지가 있는 경우, 태스크에 대한 모든 추적 로깅을 비활성화하고 복제 인스턴스 스토리지를 늘립니다. 복제 인스턴스 스토리지 증가에 대한 자세한 내용은 [복제 인스턴스 수정](#) 섹션을 참조하세요.

두 번째 메시지와 유사한 메시지가 로그에 있는 경우 다음을 수행합니다.

- 트랜잭션이 많거나 장시간 실행 중인 DML 작업이 있는 테이블이 태스크의 다른 테이블에 종속되지 않을 경우, 해당 테이블을 별도의 태스크로 이동합니다.
- MemoryLimitTotal 및 MemoryKeepTime 설정을 늘려 트랜잭션을 메모리에 보관하는 기간을 늘립니다. 이렇게 하면 지연 시간이 지속될 경우에는 도움이 되지 않지만, 트랜잭션 볼륨이 단기간에 급증하는 동안에는 지연 시간을 줄이는 데 도움이 될 수 있습니다. 이러한 태스크 설정에 대한 내용은 [변경 처리 튜닝 설정](#) 섹션을 참조하세요.
- BatchApplyEnabled를 true로 설정하여 트랜잭션에 일괄 적용을 사용할 수 있는지 평가합니다. BatchApplyEnabled 설정에 대한 내용은 [대상 메타데이터 작업 설정](#) 섹션을 참조하세요.

데이터베이스 잠금

애플리케이션이 복제 대상으로 사용하는 데이터베이스에 액세스하는 경우 애플리케이션 AWS DMS 은 DMS가 액세스하려는 테이블을 잠글 수 있습니다. 이 경우 잠금 경합이 발생합니다. DMS는 소스에서 발생한 순서대로 대상 데이터베이스에 변경 사항을 작성하므로, 잠금 경합으로 인해 한 테이블에 대한 쓰기 지연이 발생할 경우 모든 테이블에 대한 쓰기 작업이 지연됩니다.

이 문제를 해결하려면 대상 데이터베이스를 쿼리하여 잠금 경합 때문에 DMS 쓰기 트랜잭션이 차단되고 있는 것인지 확인합니다. 대상 데이터베이스가 DMS 쓰기 트랜잭션을 차단하고 있는 경우, 다음 중 하나 이상의 작업을 수행합니다.

- 변경 사항을 더 자주 커밋하도록 쿼리를 재구성합니다.
- 잠금 제한 시간 설정을 수정합니다.
- 테이블을 분할하여 잠금 경합을 최소화합니다.

참고로, 잠금 경합을 최적화하는 절차는 데이터베이스 엔진에 따라 다릅니다. 잠금 경합을 튜닝할 수 있는 DMS 기능은 없습니다.

느린 LOB 조회

가 대형 객체(LOB) 열을 AWS DMS 복제하면 대상에 변경 사항을 쓰기 직전에 소스에 대한 조회를 수행합니다. 보통 이러한 조회로 인해 대상에 지연 시간이 발생하지는 않지만, 잠금으로 인해 소스 데이터베이스에서 조회가 지연되면 대상 지연 시간이 급증할 수 있습니다.

이 문제는 일반적으로 진단하기 어렵습니다. 이 문제를 해결하려면 태스크 로그에서 세부 디버깅을 활성화하고, DMS LOB 조회 호출의 타임스탬프를 비교합니다. 세부 디버깅 활성화에 대한 자세한 내용은 [AWS DMS 작업 로그 보기 및 관리](#) 섹션을 참조하세요.

이 문제를 해결하려면 다음 작업을 수행합니다.

- 소스 데이터베이스에 대한 SELECT 쿼리 성능을 개선합니다.
- DMS LOB 설정을 튜닝합니다. LOB 설정에 대한 자세한 내용은 [대용량 이진 객체\(LOB\) 마이그레이션](#) 섹션을 참조하세요.

다중 AZ, 감사 로깅, 백업

Amazon RDS 대상의 경우, 다음 작업을 수행하는 동안 대상 지연 시간이 증가할 수 있습니다.

- 백업
- 다중 가용 영역(다중 AZ)을 활성화한 후
- 감사 또는 느린 쿼리 로그 등의 데이터베이스 로깅을 활성화한 후.

이러한 문제는 일반적으로 진단하기 어렵습니다. 이러한 문제를 해결하려면 Amazon RDS 유지 관리 기간 또는 데이터베이스 부하가 과중한 기간에 주기적인 급증이 발생하는 지연 시간을 모니터링합니다.

이러한 문제를 해결하려면 다음 작업을 수행합니다.

- 가능하다면 단기 마이그레이션 중에는 다중 AZ, 백업 또는 로깅을 비활성화합니다.
- 유지 관리 기간을 활동이 적은 기간으로 다시 예약합니다.

에서 진단 지원 스크립트 작업 AWS DMS

작업 시 문제가 발생하면 AWS DMS 지원 엔지니어가 소스 또는 대상 데이터베이스에 대한 추가 정보가 필요할 수 있습니다. AWS Support는 가능한 한 짧은 시간 내에 필요한 정보를 최대한 많이 확보하

고자 합니다. 따라서 여러 주요 관계형 데이터베이스 엔진에서 이러한 정보를 쿼리하는 스크립트를 개발했습니다.

지원 스크립트를 데이터베이스에 사용할 수 있는 경우, 아래에 설명된 해당 스크립트 주제의 링크를 사용하여 이를 다운로드할 수 있습니다. 스크립트를 확인하고 검토한 후(아래 설명 참조), 스크립트 주제에 설명된 프로시저에 따라 스크립트를 실행할 수 있습니다. 스크립트 실행이 완료되면 출력을 AWS 지원 사례(다시 설명)에 업로드할 수 있습니다.

스크립트를 실행하기 전에, 지원 스크립트를 다운로드하거나 저장할 때 발생했을 가능성이 있는 오류를 감지할 수 있습니다. 이렇게 하려면 스크립트 파일의 체크섬을에서 제공한 값과 비교합니다 AWS. 는 체크섬에 SHA256 알고리즘을 AWS 사용합니다.

체크섬을 사용하여 지원 스크립트 파일을 확인하려면

1. 제공된 최신 체크섬 파일을 열어 <https://d2pwp9zz55emqw.cloudfront.net/sha256Check.txt>에서 이러한 지원 스크립트를 확인합니다. 예를 들어, 파일에 아래와 같은 내용이 포함되어 있을 수 있습니다.

```
MYSQL dfafd0d511477c699f96c64693ad0b1547d47e74d5c5f2f2025b790b1422e3c8
ORACLE 6c41ebcf9c99518cfa8a10cb2ce8943b153b2cc7049117183d0b5de3d551bc312
POSTGRES 6ccd274863d14f6f3146fbdabbba43f2d8d4c6a4c25380d7b41c71883aa4f9790
SQL_SERVER 971a6f2c46aec8d083d2b3b6549b1e9990af3a15fe4b922e319f4fdd358debe7
```

2. 지원 파일이 포함된 디렉터리에서 운영 체제의 SHA256 검증 명령을 실행합니다. 예를 들어, macOS 운영 체제에서는 이 주제의 뒷부분에 설명된 Oracle 지원 스크립트에서 아래의 명령을 실행할 수 있습니다.

```
shasum -a 256 awsdms_support_collector_oracle.sql
```

3. 명령 결과를 최근에 열어본 sha256Check.txt 파일에 표시된 값과 비교합니다. 두 값이 일치해야 합니다. 일치하지 않을 경우, 지원 엔지니어에게 불일치하는 부분을 이야기하고 깨끗한 지원 스크립트 파일을 얻을 수 있는 방법을 문의합니다.

깨끗한 지원 스크립트 파일을 보유한 경우, 스크립트를 실행하기 전에 성능 및 보안 측면에서 SQL을 읽고 이해해야 합니다. 이 스크립트에서 SQL을 실행하는 것이 마음에 걸릴 경우 문제의 SQL을 주석 처리하거나 제거할 수 있습니다. 지원 엔지니어에게 적절한 해결 방법을 문의할 수도 있습니다.

스크립트가 성공적으로 완료되고 달리 명시되지 않는 한, 스크립트는 읽기 가능한 HTML 형식으로 출력을 반환합니다. 스크립트는 업무에 영향을 미칠 수 있는 모든 데이터 또는 보안 세부 정보를 이 HTML에서 제외하도록 설계되었습니다. 또한 스크립트는 데이터베이스나 해당 환경을 수정하지 않습

니다. 하지만 HTML에서 공유하기에 부적합한 정보를 발견한 경우, HTML을 업로드하기 전에 문제의 정보를 얼마든지 제거해도 됩니다. HTML이 허용되는 경우, 지원 사례의 사례 세부 정보에 있는 첨부 파일을 사용하여 업로드합니다.

다음 각 주제에서는 지원되는 AWS DMS 데이터베이스에 사용할 수 있는 스크립트와 이를 실행하는 방법을 설명합니다. 지원 엔지니어가 아래에 설명된 특정 스크립트를 안내합니다.

주제

- [Oracle 진단 지원 스크립트](#)
- [SQL Server 진단 지원 스크립트](#)
- [MySQL 호환 데이터베이스용 진단 지원 스크립트](#)
- [PostgreSQL 진단 지원 스크립트](#)

Oracle 진단 지원 스크립트

다음은 AWS DMS 마이그레이션 구성에서 온프레미스 또는 Amazon RDS for Oracle 데이터베이스를 분석하는 데 사용할 수 있는 진단 지원 스크립트입니다. 이러한 스크립트는 소스 또는 대상 엔드포인트에서 작동합니다. 스크립트는 모두 SQL*Plus 명령줄 유틸리티에서 실행하도록 작성되었습니다. 이 유틸리티 사용에 대한 자세한 내용은 Oracle 설명서의 [A Using SQL Command Line](#)을 참조하세요.

스크립트를 실행하기 전에, 사용하려는 사용자 계정에 Oracle 데이터베이스에 액세스하는 데 필요한 권한이 있는지 확인합니다. 표시된 권한 설정에서는 다음과 같이 사용자를 생성한 것으로 가정합니다.

```
CREATE USER script_user IDENTIFIED BY password;
```

온프레미스 데이터베이스의 경우, 다음과 같이 *script_user*에 대한 최소 권한을 설정합니다.

```
GRANT CREATE SESSION TO script_user;  
GRANT SELECT on V$DATABASE to script_user;  
GRANT SELECT on V$VERSION to script_user;  
GRANT SELECT on GV$SGA to script_user;  
GRANT SELECT on GV$INSTANCE to script_user;  
GRANT SELECT on GV$DATAGUARD_CONFIG to script_user;  
GRANT SELECT on GV$LOG to script_user;  
GRANT SELECT on DBA_TABLESPACES to script_user;  
GRANT SELECT on DBA_DATA_FILES to script_user;  
GRANT SELECT on DBA_SEGMENTS to script_user;  
GRANT SELECT on DBA_LOBS to script_user;  
GRANT SELECT on V$ARCHIVED_LOG to script_user;
```



```

GRANT SELECT on DBA_TAB_MODIFICATIONS to script_user;
GRANT SELECT on DBA_TABLES to script_user;
GRANT SELECT on DBA_TAB_PARTITIONS to script_user;
GRANT SELECT on DBA_MVIEWS to script_user;
GRANT SELECT on DBA_OBJECTS to script_user;
GRANT SELECT on DBA_TAB_COLUMNS to script_user;
GRANT SELECT on DBA_LOG_GROUPS to script_user;
GRANT SELECT on DBA_LOG_GROUP_COLUMNS to script_user;
GRANT SELECT on V$ARCHIVE_DEST to script_user;
GRANT SELECT on DBA_SYS_PRIVS to script_user;
GRANT SELECT on DBA_TAB_PRIVS to script_user;
GRANT SELECT on DBA_TYPES to script_user;
GRANT SELECT on DBA_CONSTRAINTS to script_user;
GRANT SELECT on V$TRANSACTION to script_user;
GRANT SELECT on GV$ASM_DISK_STAT to script_user;
GRANT SELECT on GV$SESSION to script_user;
GRANT SELECT on GV$SQL to script_user;
GRANT SELECT on DBA_ENCRYPTED_COLUMNS to script_user;
GRANT SELECT on DBA_PDBS to script_user;

GRANT EXECUTE on dbms_utility to script_user;

```

Amazon RDS 데이터베이스의 경우, 다음과 같이 최소 권한을 설정합니다.

```

GRANT CREATE SESSION TO script_user;
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$DATABASE', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$VERSION', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$SGA', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$INSTANCE', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_
$DATAGUARD_CONFIG', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$LOG', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TABLESPACES', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_DATA_FILES', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_SEGMENTS', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_LOBS', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$ARCHIVED_LOG', 'script_user', 'SELECT');
exec
  rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_MODIFICATIONS', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TABLES', 'script_user', 'SELECT');
exec
  rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_PARTITIONS', 'script_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_MVIEWS', 'script_user', 'SELECT');

```

```

exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_OBJECTS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_COLUMNS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_LOG_GROUPS','script_user','SELECT');
exec
    rdsadmin.rdsadmin_util.grant_sys_object('DBA_LOG_GROUP_COLUMNS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$ARCHIVE_DEST','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_SYS_PRIVS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_PRIVS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TYPES','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_CONSTRAINTS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$TRANSACTION','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_
$ASM_DISK_STAT','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$SESSION','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$SQL','script_user','SELECT');
exec
    rdsadmin.rdsadmin_util.grant_sys_object('DBA_ENCRYPTED_COLUMNS','script_user','SELECT');

exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_PDBS','script_user','SELECT');

exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_UTILITY','script_user','EXECUTE');

```

아래에는 Oracle에서 사용할 수 있는 각 SQL*Plus 지원 스크립트를 다운로드, 검토, 실행하는 방법에 대한 설명이 나와 있습니다. 출력을 검토하고 이를 AWS Support 사례에 업로드하는 방법도 참조할 수 있습니다.

주제

- [awsdms_support_collector_oracle.sql 스크립트](#)

awsdms_support_collector_oracle.sql 스크립트

[awsdms_support_collector_oracle.sql](#) 스크립트를 다운로드합니다.

이 스크립트는 Oracle 데이터베이스 구성에 대한 정보를 수집합니다. 스크립트의 체크섬을 반드시 확인하고, 체크섬이 확인되면 스크립트의 SQL 코드를 검토하여 실행하기에 부적합한 코드는 모두 주석 처리합니다. 스크립트의 무결성 및 내용에 만족한다면 스크립트를 실행해도 됩니다.

스크립트를 실행하고 결과를 지원 사례에 업로드하려면

1. 아래의 SQL*Plus 명령줄을 사용하여 데이터베이스 환경에서 스크립트를 실행합니다.

```
SQL> @awsdms_support_collector_oracle.sql
```

<result>

스크립트에 간단한 설명이 표시되며, 실행을 계속할지 중단할지 묻는 프롬프트가 표시됩니다. [Enter] 키를 눌러 계속 진행합니다.

</result>

2. 다음에 나오는 프롬프트에서 마이그레이션할 스키마 중 하나의 이름만 입력합니다.
3. 다음에 나오는 프롬프트에서 데이터베이스에 연결하기 위해 정의한 사용자 이름(*script_user*)을 입력합니다.
4. 다음에 나오는 프롬프트에서 검사할 데이터의 일수를 입력하거나, 기본값을 수락합니다. 이렇게 하면 스크립트가 데이터베이스에서 지정된 데이터를 수집합니다.

<result>

스크립트가 완료되면 출력 HTML 파일의 이름(예: `dms_support_oracle-2020-06-22-13-20-39-ORCL.html`)이 표시됩니다. 스크립트는 이 파일을 작업 디렉터리에 저장합니다.

</result>

5. 이 HTML 파일을 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. HTML을 공유할 수 있는 경우 AWS 지원 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 [에서 진단 지원 스크립트 작업 AWS DMS](#) 섹션을 참조하세요.

SQL Server 진단 지원 스크립트

다음은 AWS DMS 마이그레이션 구성에서 온프레미스 또는 Amazon RDS for SQL Server 데이터베이스를 분석하는 데 사용할 수 있는 진단 지원 스크립트에 대한 설명입니다. 이러한 스크립트는 소스 또는 대상 엔드포인트에서 작동합니다. 온프레미스 데이터베이스의 경우, sqlcmd 명령줄 유틸리티에서 이러한 스크립트를 실행합니다. 이 유틸리티 사용에 대한 자세한 내용은 Microsoft 설명서의 [sqlcmd - Use the utility](#) 섹션을 참조하세요.

Amazon RDS 데이터베이스의 경우에는 sqlcmd 명령줄 유틸리티를 사용하여 연결할 수 없습니다. 대신 Amazon RDS SQL Server에 연결되는 클라이언트 도구를 사용하여 이러한 스크립트를 실행합니다.

스크립트를 실행하기 전에, 사용하려는 사용자 계정에 SQL Server 데이터베이스에 액세스하는 데 필요한 권한이 있는지 확인합니다. 온프레미스 및 Amazon RDS SQL Server 데이터베이스의 경우, SysAdmin 역할 없이도 SQL Server 데이터베이스에 액세스할 때 사용하는 것과 동일한 권한을 사용할 수 있습니다.

주제

- [온프레미스 SQL Server 데이터베이스에 대한 최소 권한 설정](#)
- [Amazon RDS SQL Server 데이터베이스에 대한 최소 권한 설정](#)
- [SQL Server 지원 스크립트](#)

온프레미스 SQL Server 데이터베이스에 대한 최소 권한 설정

온프레미스 SQL Server 데이터베이스를 실행하기 위한 최소 권한을 설정하려면

1. SQL Server Management Studio(SSMS)를 사용하여 암호 인증이 있는 새 SQL Server 계정을 생성합니다(예: *on-prem-user*).
2. SSMS의 사용자 매핑 섹션에서 MSDB 및 MASTER 데이터베이스(공용 권한 제공)를 선택하고, 스크립트를 실행할 데이터베이스에 DB_OWNER 역할을 할당합니다.
3. 새 계정의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고, 보안을 선택한 후 명시적으로 Connect SQL 권한을 부여합니다.
4. 아래의 grant 명령을 실행합니다.

```
GRANT VIEW SERVER STATE TO on-prem-user;  
USE MSDB;  
GRANT SELECT ON MSDB.DBO.BACKUPSET TO on-prem-user;  
GRANT SELECT ON MSDB.DBO.BACKUPMEDIAFAMILY TO on-prem-user;  
GRANT SELECT ON MSDB.DBO.BACKUPFILE TO on-prem-user;
```

Amazon RDS SQL Server 데이터베이스에 대한 최소 권한 설정

Amazon RDS SQL Server 데이터베이스에 대한 최소 권한으로 실행하려면

1. SQL Server Management Studio(SSMS)를 사용하여 암호 인증이 있는 새 SQL Server 계정을 생성합니다(예: *rds-user*).
2. SSMS의 사용자 매핑 섹션에서 MSDB 데이터베이스(공용 권한 제공)를 선택하고, 스크립트를 실행할 데이터베이스에 DB_OWNER 역할을 할당합니다.

3. 새 계정의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고, 보안을 선택한 후 명시적으로 Connect SQL 권한을 부여합니다.
4. 아래의 grant 명령을 실행합니다.

```
GRANT VIEW SERVER STATE TO rds-user;
USE MSDB;
GRANT SELECT ON MSDB.DBO.BACKUPSET TO rds-user;
GRANT SELECT ON MSDB.DBO.BACKUPMEDIAFAMILY TO rds-user;
GRANT SELECT ON MSDB.DBO.BACKUPFILE TO rds-user;
```

SQL Server 지원 스크립트

다음 주제에서는 SQL Server에 사용할 수 있는 각 지원 스크립트를 다운로드, 검토, 실행하는 방법을 설명합니다. 스크립트 출력을 검토하고 이를 AWS Support 사례에 업로드하는 방법도 설명합니다.

주제

- [awsdms_support_collector_sql_server.sql 스크립트](#)

awsdms_support_collector_sql_server.sql 스크립트

[awsdms_support_collector_sql_server.sql](#) 스크립트를 다운로드합니다.

Note

이 SQL Server 진단 지원 스크립트는 SQL Server 2014 이상 버전에서만 실행합니다.

이 스크립트는 SQL Server 데이터베이스 구성에 대한 정보를 수집합니다. 스크립트의 체크섬을 반드시 확인하고, 체크섬이 확인되면 스크립트의 SQL 코드를 검토하여 실행하기에 부적합한 코드는 모두 주석 처리합니다. 스크립트의 무결성 및 내용에 만족한다면 스크립트를 실행해도 됩니다.

온프레미스 SQL Server 데이터베이스용 스크립트를 실행하려면

1. 아래의 sqlcmd 명령줄을 사용하여 스크립트를 실행합니다.

```
sqlcmd -Uon-prem-user -Ppassword -SDMS-SQL17AG-N1 -y 0
-iC:\Users\admin\awsdms_support_collector_sql_server.sql -oC:\Users\admin
\DMS_Support_Report_SQLServer.html -dsqlserverdb01
```

지정된 sqlcmd 명령 파라미터에는 다음 내용이 포함됩니다.

- -U – 데이터베이스 사용자 이름.
 - -P – 데이터베이스 사용자 암호.
 - -S – SQL Server 데이터베이스 서버 이름.
 - -y – sqlcmd 유틸리티의 최대 출력 열 너비. 값이 0이면 너비가 무제한인 열이 지정됩니다.
 - -i – 실행할 지원 스크립트의 경로(이 경우 `awsdms_support_collector_sql_server.sql`).
 - -o – 수집된 데이터베이스 구성 정보가 포함된 출력 HTML 파일의 경로(지정한 파일 이름 포함).
 - -d – SQL Server 데이터베이스 이름.
2. 스크립트가 완료되면 출력 HTML 파일을 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. HTML을 공유할 수 있는 경우 AWS 지원 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 [에서 진단 지원 스크립트 작업 AWS DMS](#) 섹션을 참조하세요.

Amazon RDS for SQL Server에서는 sqlcmd 명령줄 유틸리티를 사용하여 연결할 수 없으므로, 다음 프로시저를 사용합니다.

온프레미스 RDS SQL Server 데이터베이스용 스크립트를 실행하려면

1. RDS SQL Server에 Master 사용자로 연결하고 출력을 HTML 파일로 저장할 수 있는 클라이언트 도구를 사용하여 스크립트를 실행합니다.
2. 출력 HTML 파일을 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. HTML을 공유할 수 있는 경우 AWS 지원 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 [에서 진단 지원 스크립트 작업 AWS DMS](#) 섹션을 참조하세요.

MySQL 호환 데이터베이스용 진단 지원 스크립트

다음은 AWS DMS 마이그레이션 구성에서 온프레미스 또는 Amazon RDS for MySQL 호환 데이터베이스를 분석하는 데 사용할 수 있는 진단 지원 스크립트입니다. 이러한 스크립트는 소스 또는 대상 엔드포인트에서 작동합니다. 스크립트는 모두 MySQL SQL 명령줄 유틸리티에서 실행하도록 작성되었습니다.

MySQL 클라이언트 설치에 대한 자세한 내용은 MySQL 설명서의 [Installing MySQL Shell](#) 섹션을 참조하세요. MySQL 클라이언트 사용에 대한 자세한 내용은 MySQL 설명서의 [Using MySQL Shell Commands](#) 섹션을 참조하세요.

스크립트를 실행하기 전에, 사용하려는 사용자 계정에 MySQL 호환 데이터베이스에 액세스하는 데 필요한 권한이 있는지 확인합니다. 아래의 절차를 사용하여 사용자 계정을 생성하고, 이 스크립트를 실행하는 데 필요한 최소 권한을 제공합니다.

이러한 스크립트를 실행할 수 있는 최소 권한으로 사용자 계정을 설정하려면

1. 스크립트를 실행할 사용자를 생성합니다.

```
create user 'username'@'hostname' identified by password;
```

2. 데이터베이스에서 select 명령을 부여하여 데이터베이스를 분석합니다.

```
grant select on database-name.* to username;  
grant replication client on *.* to username;
```

3.

```
grant execute on procedure mysql.rds_show_configuration to username;
```

다음 주제에서는 MySQL 호환 데이터베이스에 사용할 수 있는 각 지원 스크립트를 다운로드, 검토, 실행하는 방법을 설명합니다. 또한 스크립트 출력을 검토하고 AWS 지원 사례에 업로드하는 방법도 설명합니다.

주제

- [awsdms_support_collector_MySQL.sql 스크립트](#)

awsdms_support_collector_MySQL.sql 스크립트

[awsdms_support_collector_MySQL.sql](#) 스크립트를 다운로드합니다.

이 스크립트는 MySQL 호환 데이터베이스 구성에 대한 정보를 수집합니다. 스크립트의 체크섬을 반드시 확인하고, 체크섬이 확인되면 스크립트의 SQL 코드를 검토하여 실행하기에 부적합한 코드는 모두 주석 처리합니다. 스크립트의 무결성 및 내용에 만족한다면 스크립트를 실행해도 됩니다.

명령줄을 사용하여 데이터베이스 환경에 연결한 후 스크립트를 실행합니다.

이 스크립트를 실행하고 결과를 지원 사례에 업로드하려면

1. 아래의 mysql 명령을 사용하여 데이터베이스에 연결합니다.

```
mysql -p -h hostname -P port -u username database-name
```

2. 아래의 mysql source 명령을 사용하여 스크립트를 실행합니다.

```
source awsdms_support_collector_MySQL.sql
```

생성된 보고서를 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. 내용을 공유해도 괜찮다면 AWS Support 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 [에서 진단 지원 스크립트 작업 AWS DMS](#) 섹션을 참조하세요.

Note

- [MySQL 호환 데이터베이스용 진단 지원 스크립트](#)에 설명된 필수 권한을 보유한 사용자 계정이 이미 있는 경우, 기존 사용자 계정을 사용하여 스크립트를 실행할 수도 있습니다.
- 스크립트를 실행하기 전에 데이터베이스에 연결해야 합니다.
- 스크립트는 텍스트 형식으로 출력을 생성합니다.
- 보안 모범 사례를 고려한다면, 이 MySQL 진단 지원 스크립트를 실행하기 위한 용도로만 새 사용자 계정을 생성할 경우 스크립트를 실행한 후에는 이 사용자 계정을 삭제하는 것이 좋습니다.

PostgreSQL 진단 지원 스크립트

다음은 AWS DMS 마이그레이션 구성에서 PostgreSQL RDBMS(온프레미스, Amazon RDS 또는 Aurora PostgreSQL)를 분석하는 데 사용할 수 있는 진단 지원 스크립트입니다. 이러한 스크립트는 소스 또는 대상 엔드포인트에서 작동합니다. 스크립트는 모두 psql 명령줄 유틸리티에서 실행하도록 작성되었습니다.

이러한 스크립트를 실행하기 전에, 사용하려는 사용자 계정에 PostgreSQL RDBMS에 액세스하는 데 필요한 다음과 같은 권한이 있는지 확인합니다.

- PostgreSQL 10.x 이상 버전 – pg_catalog.pg_ls_waldir 함수에 대한 실행 권한이 있는 사용자 계정.
- PostgreSQL 9.x 이전 버전 – 기본 권한이 있는 사용자 계정.

이러한 스크립트를 실행하려면 적절한 권한이 있는 기존 계정을 사용하는 것이 좋습니다.

새 사용자 계정을 생성하거나 기존 계정에 이러한 스크립트를 실행할 수 있는 권한을 부여해야 하는 경우, PostgreSQL 버전을 기반으로 하는 모든 PostgreSQL RDBMS에 대해 다음과 같은 SQL 명령을 실행할 수 있습니다.

PostgreSQL 데이터베이스 버전 10.x 이상에서 이러한 스크립트를 실행할 수 있는 권한을 계정에 부여하려면

- 다음 중 하나를 수행하세요.
- 새 사용자 계정의 경우 다음을 실행합니다.

```
CREATE USER script_user WITH PASSWORD 'password';  
GRANT EXECUTE ON FUNCTION pg_catalog.pg_ls_waldir TO script_user;
```

- 새 사용자 계정의 경우 다음을 실행합니다.

```
GRANT EXECUTE ON FUNCTION pg_catalog.pg_ls_waldir TO script_user;
```

PostgreSQL 9.x 이하 버전의 데이터베이스에서 이러한 스크립트를 실행할 수 있는 권한을 계정에 부여하려면

- 다음 중 하나를 수행하세요.
- 새 사용자 계정의 경우 기본 권한으로 다음을 실행합니다.

```
CREATE USER script_user WITH PASSWORD password;
```

- 기존 사용자 계정의 경우 기존 권한을 사용합니다.

Note

이러한 스크립트는 PostgreSQL 9.x 이하 버전 데이터베이스의 WAL 크기 확인과 관련된 특정 기능을 지원하지 않습니다. 자세한 내용은 AWS Support에 문의하세요.

다음 주제에서는 PostgreSQL에 사용할 수 있는 각 지원 스크립트를 다운로드, 검토, 실행하는 방법을 설명합니다. 스크립트 출력을 검토하고 이를 AWS Support 사례에 업로드하는 방법도 설명합니다.

주제

- [awsdms_support_collector_postgres.sql 스크립트](#)

awsdms_support_collector_postgres.sql 스크립트

[awsdms_support_collector_postgres.sql](#) 스크립트를 다운로드합니다.

이 스크립트는 PostgreSQL 데이터베이스 구성에 대한 정보를 수집합니다. 스크립트에서 체크섬을 반드시 확인해야 합니다. 체크섬이 확인되면 스크립트의 SQL 코드를 검토하여 실행하기에 부적합한 코드는 모두 주석 처리합니다. 스크립트의 무결성 및 내용에 만족한다면 스크립트를 실행해도 됩니다.

Note

psql 클라이언트 버전 10 이상에서 이 스크립트를 실행할 수 있습니다.

다음 절차에 따라 데이터베이스 환경 또는 명령줄에서 이 스크립트를 실행할 수 있습니다. 둘 중 어떤 경우든 나중에 AWS Support에 파일을 업로드할 수 있습니다.

이 스크립트를 실행하고 결과를 지원 사례에 업로드하려면

1. 다음 중 하나를 수행하세요.

- 아래의 psql 명령줄을 사용하여 데이터베이스 환경에서 스크립트를 실행합니다.

```
dbname=# \i awsdms_support_collector_postgres.sql
```

다음에 나오는 프롬프트에서 마이그레이션할 스키마 중 하나의 이름만 입력합니다.

다음에 나오는 프롬프트에서 데이터베이스에 연결하기 위해 정의한 사용자 이름 (*script_user*)을 입력합니다.

- 명령줄에서 직접 아래의 스크립트를 실행합니다. 이 옵션을 사용하면 스크립트 실행 전에 프롬프트가 표시되지 않습니다.

```
psql -h database-hostname -p port -U script_user -d database-name -f  
awsdms_support_collector_postgres.sql
```

2. 출력 HTML 파일을 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. HTML을 공유해도 괜찮다면 AWS Support 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 [에서 진단 지원 스크립트 작업 AWS DMS](#) 섹션을 참조하세요.

AWS DMS 진단 지원 AMI 작업

작업 시 네트워크 관련 문제가 발생하면 지원 엔지니어 AWS DMS가 네트워크 구성에 대한 추가 정보를 필요로 할 수 있습니다. AWS Support는 최대한 짧은 시간 내에 필요한 정보를 최대한 많이 얻고자 합니다. 따라서 AWS DMS 네트워킹 환경을 테스트하기 위한 진단 도구를 사용하여 사전 구축된 Amazon EC2 AMI를 개발했습니다.

Amazon Machine Image(AMI)에 설치된 진단 테스트에는 다음 사항이 포함됩니다.

- 가상 프라이빗 클라우드(VPC)
- 네트워크 패킷 손실
- 네트워크 지연 시간
- 최대 전송 단위(MTU) 크기

주제

- [새 AWS DMS 진단 Amazon EC2 인스턴스 시작](#)
- [IAM 역할 생성](#)
- [진단 테스트 실행](#)
- [다음 단계](#)
- [리전별 AMI ID](#)
- [AWS 시스템 패치 관리자](#)

Note

Oracle 소스에서 성능 문제가 발생할 경우, Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능을 평가하여 성능을 개선할 방법을 찾을 수 있습니다. 자세한 내용은 [Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능 평가](#) 섹션을 참조하세요.

새 AWS DMS 진단 Amazon EC2 인스턴스 시작

이 섹션에서는 새 Amazon EC2 인스턴스를 시작합니다. Amazon EC2 인스턴스를 시작하는 방법에 대한 내용은 [Amazon EC2 사용 설명서](#)의 [Amazon EC2 Linux 인스턴스 시작](#) 섹션을 참조하세요.

아래의 설정을 사용하여 Amazon EC2 인스턴스를 시작합니다.

- 애플리케이션 및 OS 이미지(Amazon Machine Image)의 경우, DMS-DIAG-AMI를 검색합니다. 콘솔에 로그인한 경우 [이 쿼리](#)를 사용하여 AMI를 검색할 수 있습니다. 해당 리전에서 AWS 진단 AMI의 AMI ID는 [리전별 AMI ID](#) 다음을 참조하세요.
- 인스턴스 유형으로는 t2.micro를 선택하는 것을 권장합니다.
- 네트워크 설정에서 복제 인스턴스가 사용하는 것과 동일한 VPC를 선택합니다.

인스턴스가 활성화되면 인스턴스에 연결합니다. Amazon EC2 Linux 인스턴스에 연결하는 방법에 대한 내용은 [Linux 인스턴스에 연결합니다](#) 섹션을 참조하세요.

IAM 역할 생성

필요한 최소한의 권한을 사용하여 복제 인스턴스에서 진단 테스트를 실행하려면 다음과 같은 권한 정책을 사용하는 IAM 역할을 생성합니다.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "dms:DescribeEndpoints",
        "dms:DescribeTableStatistics",
        "dms:DescribeReplicationInstances",
        "dms:DescribeReplicationTasks",
        "secretsmanager:GetSecretValue"
      ],
      "Resource": "*"
    }
  ]
}
```

```
}
```

새 IAM 사용자에게 역할을 연결합니다. IAM 역할, 정책, 사용자 생성에 대한 자세한 내용은 [IAM 사용자 설명서](#)의 다음 섹션을 참조하세요.

- [IAM 시작하기](#)
- [IAM 역할 생성](#)
- [IAM 정책 생성](#)

진단 테스트 실행

Amazon EC2 인스턴스를 생성하고 연결한 후, 다음을 수행하여 복제 인스턴스에서 진단 테스트를 실행합니다.

1. AWS CLI를 구성합니다.

```
$ aws configure
```

진단 테스트를 실행하는 데 사용할 AWS 사용자 계정의 액세스 자격 증명을 제공합니다. VPC 및 복제 인스턴스의 리전을 입력합니다.

2. 리전에서 사용 가능한 AWS DMS 작업을 표시합니다. 샘플 리전을 현재 리전으로 바꿉니다.

```
$ dms-report -r us-east-1 -l
```

이 명령은 사용자의 태스크 상태를 표시합니다.



3. **태스크 엔드포인트 및 설정을 표시합니다.** *<DMS-Task-ARN>*을 Amazon 리소스 이름(ARN) 태스크로 바꿉니다.

```
$ dms-report -t <DMS-Task-ARN>
```

이 명령은 태스크의 엔드포인트와 설정을 표시합니다.

```
#####  
#  
#  
#      AWS DMS Diagnostic  
#      Date: 07-13-2022  
#  
#  
#      aws region: us-east-2  
#  
#####  
  
===== DMS Task Check =====  
  
[DMS Replication ID]    aws:dms:us-east-2:pglogical-task  
[DMS Replication Task]  arn:aws:dms:us-east-2:123456789012:task:QJZVDFGQKCTEDTLLNDCXWDDUWYRWDGACHTHEI  
[Replication Instance]  arn:aws:dms:us-east-2:123456789012:rep:MALCOWBOLUJLHTEPGLNGEPLKATSLUGMREKIDWQWQA  
  
[RepInstanceClass]      dms.t3.small  
[RepAvailabilityZone]    us-east-2c  
[RepEngineVersion]      3.4.6  
[RepPrivateIP]          172.30.2.230  
[RepPublicIP]           172.30.172.172  
[RepCreateDate]         2021-12-15T21:37:19.904000+00:00  
[RepVPCId]              vpc-08ba020355d8a952e  
  
===== SOURCE =====  
[Endpoints]             arn:aws:dms:us-east-2:123456789012:endpoint:41DQCFB3A3CXGSEVRSXWMTOTQYJWWYABSDUHQ  
[ID]                    end-dms-s3-postgres-dev-source  
[Server]                end-dms-s3-postgres-dev-instance-1.cucdvzaur7nk.us-east-2.rds.amazonaws.com  
[TCP-Port]              5432  
[IP-Test]               172.30.1.204 << IP and TCP Port are GOOD  
[Network-Check] Host  
[Engine]                aurora-postgresql  
[DB]                    testdb  
[Status]                active,  
  
===== TARGET =====  
[Endpoints]             arn:aws:dms:us-east-2:123456789012:endpoint:IXKJWISXCCSBQFTHPUYNLSYBUCJDSXGCCSXTNY  
[ID]                    rds-postgres-instance-1  
[Server]                rds-postgres-instance-1.cucdvzaur7nk.us-east-2.rds.amazonaws.com  
[TCP-Port]              5432  
[IP-Test]               172.30.2.18 << IP and TCP Port are Good  
[Network-Check] Host  
[Engine]                aurora-postgresql  
[DB]                    testdb  
[Status]                active,  
  
===== Full Load =====  


| [Table]              | [Duration] | [Rows] | [LastUpdate]                     | [Status]  |
|----------------------|------------|--------|----------------------------------|-----------|
| public.test_orders   | 00:00:01   | 2      | 2022-06-21T08:48:12.172000+00:00 | completed |
| public.test_details  | 00:00:01   | 8      | 2022-06-21T08:48:12.172000+00:00 | completed |
| public.test_products | 00:00:00   | 2      | 2022-06-21T08:48:12.172000+00:00 | completed |
| public.delete_log    | 00:00:00   | 2      | 2022-06-21T08:48:12.172000+00:00 | completed |
| public.address       | 00:00:01   | 1999   | 2022-06-21T08:48:12.172000+00:00 | completed |
| public.bugfix_test   | 00:00:00   | 6      | 2022-06-21T08:48:12.172000+00:00 | completed |


```

4. 진단 테스트를 실행합니다. *<DMS-Task-ARN>*을 태스크 ARN으로 바꿉니다.

```
$ dms-report -t <DMS-Task-ARN> -n y
```

이 명령은 복제 인스턴스의 VPC, 네트워크 패킷 전송, 네트워크 지연 시간, 네트워크 최대 전송 단위(MTU) 크기에 대한 진단 데이터를 표시합니다.


```

#####
#
#
#   AWS DMS Diagnostic
#   Date: 07-13-2022
#
#
#   aws region: us-east-2
#
#
#####
===== DMS DIAG Info =====
Public IP: 3.12.101.10
Private IP: 172.30.0.240
Instance ID: i-04829b2beb8214602
Instance MAC: 02:58:04:b5:52:28
Instance Type: t2.micro
Instance Sec Group: DMS-Sub-sec-group
Instance AWS Region: us-east-2
Instance VPC Id: vpc-08ba020355d8a952e

===== Network Packet Check =====
1.) Check DMS EC2 MetaData Service
>>>>Result: 10 packets transmitted, 10 packets received, 0% packet loss
    Looks good with no issue. <<<<

2.) Check Source endpoint (dms-0d0lms-postgres-dev-instance-1.cucdvzaur7nk.us-east-2.rds.amazonaws.com:5432)
>>>>Result: 10 packets transmitted, 10 packets received, 0% packet loss
    Looks good with no issue. <<<<

3.) Check Target endpoint (rds-postgres-instance-1.cucdvzaur7nk.us-east-2.rds.amazonaws.com:5432)
>>>>Result: 10 packets transmitted, 10 packets received, 0% packet loss
    Looks good with no issue. <<<<

===== End network packet check =====

===== Network Latency Check =====
1.) Check DMS MetaData Service
>>>>Result: round-trip min/avg/max = 0.4/0.4/0.5 ms
    Looks good with no issue. <<<<

2.) Check Source endpoint (dms-0d0lms-postgres-dev-instance-1.cucdvzaur7nk.us-east-2.rds.amazonaws.com:5432)
>>>>Result: round-trip min/avg/max = 1.0/1.1/1.2 ms
    Looks good with no issue. <<<<

3.) Check Target endpoint (rds-postgres-instance-1.cucdvzaur7nk.us-east-2.rds.amazonaws.com:5432)
>>>>Result: round-trip min/avg/max = 1.4/1.4/1.5 ms
    Looks good with no issue. <<<<

===== End network latency check =====

===== Network MTU Check =====
1.) Check DMS MetaData Service
>>>>Result: MTU setting looks good. Local MTU (9001) matches remote MTU (9001) <<<<

2.) Check Source endpoint (dms-0d0lms-postgres-dev-instance-1.cucdvzaur7nk.us-east-2.rds.amazonaws.com:5432)
>>>>Result: MTU setting looks good. Local MTU (9001) matches remote MTU (9001) <<<<

3.) Check Target endpoint (rds-postgres-instance-1.cucdvzaur7nk.us-east-2.rds.amazonaws.com:5432)
>>>>Result: MTU setting looks good. Local MTU (9001) matches remote MTU (9001) <<<<

===== End network MTU check =====

```

Perform AMI Diag EC2 VPC Check

Perform Network Packet Test

Returns Test Results and Recommendation

Perform Network Latency Test

Perform Network Maximum Transmission Unit (MTU) Check

다음 단계

다음 섹션에서는 네트워크 진단 테스트 결과에 기반한 문제 해결 정보를 설명합니다.

VPC 테스트

이 테스트는 진단 Amazon EC2 인스턴스가 복제 인스턴스와 동일한 VPC에 있는지 확인합니다. 진단 Amazon EC2 인스턴스가 복제 인스턴스와 동일한 VPC에 있지 않을 경우, 해당 인스턴스를 종료하고 올바른 VPC에서 다시 생성합니다. 생성한 후에는 Amazon EC2 인스턴스의 VPC를 변경할 수 없습니다.

네트워크 패킷 손실 테스트

이 테스트는 다음과 같은 엔드포인트로 패킷 10개를 전송한 후 패킷 손실 여부를 확인합니다.

- 포트 80의 AWS DMS Amazon EC2 메타데이터 서비스
- 소스 엔드포인트
- 대상 엔드포인트

모든 패킷이 도착해야 합니다. 패킷이 손실된 경우, 네트워크 엔지니어에게 문의하여 문제를 확인하고 해결 방법을 찾습니다.

네트워크 지연 시간 테스트

이 테스트는 이전 테스트와 동일한 엔드포인트로 패킷 10개를 전송한 후 패킷 지연 시간이 발생했는지 확인합니다. 모든 패킷의 지연 시간은 100밀리초 미만이어야 합니다. 패킷에 100밀리초 이상의 지연 시간이 있을 경우, 네트워크 엔지니어에게 문의하여 문제를 확인하고 해결 방법을 찾습니다.

최대 전송 단위(MTU) 크기 테스트

이 테스트는 이전 테스트와 동일한 엔드포인트에서 경로 추적 도구를 사용하여 MTU 크기를 감지합니다. 테스트의 모든 패킷은 MTU 크기가 동일해야 합니다. MTU 크기가 다른 패킷이 있을 경우, 시스템 전문가에게 문의하여 문제를 확인하고 해결 방법을 찾습니다.

리전별 AMI ID

AWS 리전에서 사용 가능한 DMS 진단 AMIs 목록을 보려면 다음 AWS CLI 샘플을 실행합니다.

```
aws ec2 describe-images --owners 343299325021 --filters "Name=name, Values=DMS-DIAG*"
--query "sort_by(Images, &CreationDate)[-1].[Name, ImageId, CreationDate]" --output
text
```

출력에 결과가 표시되지 않으면 해당 AWS 리전에서 DMS 진단 AMI를 사용할 수 없음을 의미합니다. 해결 방법은 다음 단계에 따라 다른 리전에서 진단 AMI를 복사하는 것입니다. 자세한 내용은 [AMI 복사](#)를 참조하세요.

- 사용 가능한 리전에서 인스턴스를 시작합니다.
- 이미지를 생성합니다. 이미지는 사용자가 소유합니다.
- AMI를 사용자 리전(예: 중동(UAE) 리전)에 복사합니다.
- 사용자 로컬 리전에서 인스턴스를 시작합니다.

AWS 시스템 패치 관리자

AWS Systems Patch Manager는 EC2 인스턴스의 운영 체제 및 애플리케이션에 대한 패치 적용을 자동화합니다.

패치 매니저를 구성하려면 다음 단계를 수행합니다.

1. 시스템 매니저 활성화:
 - a. AmazonSSMManagedInstanceCore IAM 정책을 EC2 인스턴스 역할에 연결합니다.
 - b. SSM 에이전트가 설치되어 있는지 확인합니다(Amazon Linux 2, Ubuntu 20.04+ AMIs 경우 기본값).
2. 패치 애플리케이션 일정을 포함하여 중요/보안 업데이트에 대한 규칙을 정의하여 패치 기준을 생성합니다.
3. SSM의 유지 관리 기간을 사용하여 정의된 시간에 패치를 적용하기 위한 업데이트를 예약합니다.
4. Systems Manager에서 패치 상태 및 규정 준수 보고서를 확인하여 규정 준수를 확인합니다.

AWS DMS 참조

이 참조 섹션에서 데이터 형식 변환 정보 등 AWS Database Migration Service(AWS DMS)를 사용할 때 필요한 추가 정보를 찾을 수 있습니다.

AWS DMS는 소스 및 대상 모두에서 동일한 엔진 유형을 사용하는 동종 데이터베이스 마이그레이션을 수행할 때 데이터 유형을 그대로 유지합니다. 한 가지 데이터베이스 엔진 유형에서 다른 데이터베이스 엔진으로 마이그레이션하는 이종 마이그레이션을 수행할 때는 데이터 형식이 중간 데이터 형식으로 변환됩니다. 데이터 형식이 대상 데이터베이스에서 표시되는 방식은 원본 및 대상 데이터베이스 엔진의 데이터 형식 테이블을 참조하세요.

데이터베이스를 마이그레이션할 때 데이터 형식에 대해 몇 가지 중요한 사실을 알아 두세요.

- FLOAT 데이터 형식은 본래 근사치입니다. FLOAT에 특정한 값을 삽입하면 데이터베이스에서 다르게 나타날 수 있습니다. FLOAT는 NUMBER, NUMBER(p,s) 등의 10진수 데이터 형식과 같이 정확한 데이터 형식이 아니므로 이 차이가 생깁니다. 따라서 데이터베이스에 저장된 FLOAT의 내부 값이 삽입한 값과 다를 수 있습니다. 따라서 FLOAT의 마이그레이션된 값이 원본 데이터베이스의 값과 정확히 일치하지 않을 수 있습니다.

이 문제에 대한 자세한 내용은 다음 문서를 참조하세요.

- Wikipedia의 [IEEE 부동 소수점](#)
- Microsoft Learn의 [IEEE 부동 소수점 표현](#)
- Microsoft Learn의 [부동 소수점 숫자의 정밀도가 떨어지는 이유](#)

주제

- [AWS Database Migration Service에서 사용되는 데이터 형식](#)

AWS Database Migration Service에서 사용되는 데이터 형식

AWS Database Migration Service는 내장 데이터 형식을 사용하여 소스 데이터베이스 엔진 유형에서 대상 데이터베이스 엔진 유형으로 데이터를 마이그레이션합니다. 다음 테이블에는 내장 데이터 형식과 그에 대한 설명이 나와 있습니다.

AWS DMS 데이터 형식	설명
STRING	문자열.

AWS DMS 데이터 형식	설명
WSTRING	2바이트 문자열.
BOOLEAN	부울 값.
BYTE	이진 데이터 값.
날짜	날짜 값: 연도, 월, 일.
TIME	시간 값: 시간, 분, 초.
DATETIME	타임스탬프 값: 연도, 월, 일, 시간, 분, 초, 소수부 초. 소수부 초의 최대 크기는 9자리입니다. 지원되는 형식은 다음과 같습니다. YYYY:MM:DD HH:MM:SS.F(9). Amazon S3 Select 및 Amazon S3 Glacier Select에서는 DATETIME 데이터 형식이 다릅니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 지원되는 데이터 형식 에서 timestamp 기본 데이터 형식에 대한 설명을 참조하세요.
INT1	1바이트, 부호 있는 정수.
INT2	2바이트, 부호 있는 정수.
INT4	4바이트, 부호 있는 정수.
INT8	8바이트, 부호 있는 정수.
NUMERIC	고정 정밀도 및 크기를 사용한 정확한 숫자 값.
REAL4	단일 정밀도 부동 소수점 값.
REAL8	배정밀도 부동 소수점 값.
UINT1	1바이트, 부호 없는 정수.
UINT2	2바이트, 부호 없는 정수.
UINT4	4바이트, 부호 없는 정수.

AWS DMS 데이터 형식	설명
UINT8	8바이트, 부호 없는 정수.
BLOB	이진 대용량 객체.
CLOB	문자 대용량 객체.
NCLOB	고유 문자 대용량 객체.

 Note

AWS DMS는 어떤 LOB 데이터 형식도 Apache Kafka 엔드포인트로 마이그레이션 할 수 없습니다.

AWS DMS 릴리스 정보

아래에서 AWS Database Migration Service()의 현재 및 이전 버전에 대한 릴리스 정보를 확인할 수 있습니다AWS DMS.

AWS DMS 는 복제 인스턴스에 대해 자동 버전 업그레이드를 활성화할 때 메이저 버전과 마이너 버전을 구분하지 않습니다. DMS는 유지 관리 기간 동안 더 이상 사용되지 않는 복제 인스턴스 버전을 자동으로 업그레이드합니다.

복제 인스턴스의 버전을 버전 3.4.x에서 3.5.x로 수동으로 업그레이드하려면(API 또는 CLI 사용) AllowMajorVersionUpgrade 파라미터를 true로 설정해야 합니다. AllowMajorVersionUpgrade 파라미터에 대한 자세한 내용은 DMS API 설명서의 [ModifyReplicationInstance](#)를 참조하세요.

Note

의 현재 기본 엔진 버전은 3.5.4 AWS DMS 입니다.

다음 표에는 활성 DMS 버전의 다음 날짜가 나와 있습니다.

- 버전 릴리스 날짜
- 버전으로 새 인스턴스를 생성할 수 없는 날짜
- DMS가 해당 버전의 인스턴스를 자동으로 업데이트하는 날짜(EOL 날짜)

버전	릴리스 날짜	새 인스턴스 없음 날짜	EOL 날짜
3.6.1	2025년 5월 15일	2027년 4월 30일	2027년 6월 30일
3.6.0	2024년 12월 27일	2026년 4월 27일	2026년 6월 27일
3.5.4	2024년 11월 15일	2026년 9월 15일	2026년 11월 15일
3.5.3	2024년 5월 17일	2026년 2월 28일	2026년 6월 30일
3.5.2	2023년 10월 29일	2025년 3월 30일	2025년 4월 29일
3.5.1	2023년 6월 30일	2024년 11월 30일	2025년 3월 15일

버전	릴리스 날짜	새 인스턴스 없음 날짜	EOL 날짜
3.4.7	2022년 5월 31일	2024년 9월 30일	2024년 10월 31일
3.4.6	2021년 11월 30일	2024년 9월 30일	2024년 10월 31일

Note

AWS DMS 버전 3.6.0은 현재 아시아 태평양(시드니): ap-southeast-2 리전에서 사용할 수 있습니다.

AWS Database Migration Service 3.6.1 릴리스 정보

AWS DMS 3.6.1의 새로운 기능

새로운 기능 또는 향상된 기능	설명
DMS 데이터 재동기화	AWS DMS 소스 데이터베이스와 대상 데이터베이스 간의 데이터 검증을 통해 식별된 데이터 불일치를 자동으로 수정합니다. 데이터 재동기화 기능은 Oracle 및 SQL Server를 소스로, Postgre SQL을 대상 데이터베이스로 지원합니다.
MariaDB, MySQL 및 PostgreSQL IAM 데이터베이스 인증	AWS Identity and Access Management(IAM) 데이터베이스 인증 을 통해 Amazon RDS 및 Aurora MariaDB, MySQL 및 PostgreSQL 엔드포인트에 연결하는 지원을 도입했습니다. 이 기능을 통해 각 데이터베이스 엔드포인트에서 개별적으로 액세스를 관리하는 대신 IAM을 사용하여 데이터베이스 리소스에 대한 액세스를 중앙에서 관리할 수 있습니다.
변경 데이터 캡처(CDC) 복제에 대한 PostgreSQL 읽기 전용 복제본 지원	AWS DMS는 PostgreSQL AWS DMS 버전 3.6.1부터 PostgreSQL 버전 16.x 이상에서 사용할 수 있는 변경 데이터 캡처(CDC) 복제를 위한 소스 엔드포인트로 PostgreSQL 읽기 전용 복제본 사용을 지원합니다. 이 기능을 사용하면 CDC 작업에 읽기 전용 복제본을 활용할 수 있습니다. 자세한 내용은 PostgreSQL 데이터베이스를 AWS DMS 원본으로 사용 을 참조하세요.

새로운 기능 또는 향상된 기능	설명
누락된 대상 열에 대한 향상된 로깅	기본 로깅 수준에서 누락된 대상 테이블 열을 고객에게 알리도록 로깅을 개선했습니다. 이 개선 사항은 로깅 수준을 VERBOSE에서 WARNING으로 변경하므로 세부 디버그 설정 없이 열 불일치 알림을 더 잘 볼 수 있습니다.
SQL Server Binary(16)에서 PostgreSQL UUID로 마이그레이션	SQL Server Binary(16) 데이터 형식을 PostgreSQL UUID 형식으로 마이그레이션하기 위한 지원이 추가되었습니다. 이를 통해 바이너리 GUID 데이터를 네이티브 UUID 유형으로 원활하게 변환하여 SQL Server와 PostgreSQL 엔드포인트 간의 데이터 유형 호환성을 개선할 수 있습니다.

AWS DMS 버전 3.6.1에는 다음과 같은 해결된 문제가 포함되어 있습니다.

DMS 3.6.1에서 해결된 문제

해결된 문제	설명
데이터 마스킹 문제를 사용한 선택 규칙 필터링	이제 테이블 선택 규칙 필터가 데이터 마스킹 변환 규칙에서 올바르게 작동합니다. 수정 사항은 데이터 마스킹 변환이 적용될 때 적절한 필터링을 보장합니다.
PostgreSQL 대상의 무한 숫자 처리 문제	PostgreSQL 대상 엔드포인트는 최적화된 메모리 할당을 통해 무한 숫자 데이터 유형을 처리할 수 있습니다. 이렇게 하면 숫자 값이 큰 테이블을 마이그레이션할 때 작업 실패를 방지할 수 있습니다.
Amazon Redshift 대상 LOB 처리 문제	Redshift 대상 엔드포인트는 이제 대용량 병렬 마이그레이션 중에 LOB 데이터 무결성을 유지합니다. 향상된 LOB 처리는 병렬 적용 스레드를 사용할 때 데이터 손상을 방지합니다.
변환 규칙 메타데이터 문제	변환 규칙에서 표현식이 잘못 적용된 문제를 수정했습니다.

해결된 문제	설명
메타데이터 테이블 삭제 및 다시 생성 문제	이제 작업 중지 중에 소스 테이블이 삭제되고 다시 생성되면 복제 작업이 성공적으로 재개됩니다. 재생성된 소스 테이블의 메타데이터 처리 및 상태 동기화를 개선했습니다.
SQL Server AlwaysOn 기본 복제본 문제	대소문자 구분 변형을 처리하여 AlwaysOn 구성에서 SQL Server 소스에 대한 기본 복제본 감지를 개선했습니다.
MySQL 소스 TIME 데이터 유형 처리 문제	loadUsingCSV 가 비활성화된 경우 분수와 24 시간을 초과하는 시간이 포함된 TIME 값의 마이그레이션 문제를 해결했습니다.
데이터 검증 NUL 문자 처리 문제	이제 데이터 검증은 NUL(0x00) 문자를 실패 없이 올바르게 처리합니다. NUL 문자가 포함된 데이터에 대한 검증 처리를 개선했습니다.
SQL Server 데이터 검증 문제	SQL Server 소스의 UNIQUEIDENTIFIER 열과 관련된 검증 비교 문제를 해결했습니다.
테이블 사전 로드 검증 문제	AWS DMS 이제 작업이 CDC 복제 중에 테이블이 사전 로드 단계에 있을 때 검증을 성공적으로 완료합니다.
Oracle 소스 로깅 문제	Oracle 소스 오류 로그는 이제 시간대 쿼리 실행 중에 올바른 SQL 문을 표시합니다. 로깅 정확도를 개선하여 문제 해결 기능을 개선했습니다.
SQL Server 드라이버 업데이트	SQL Server 드라이버 지원이 ODBC 17에서 ODBC 18로 업데이트되었습니다.
DB2 빈 문자열 CDC 마이그레이션 문제	변경 데이터 캡처(CDC) 작업 중에 IBM DB2 LUW 소스 데이터베이스의 빈 VARCHAR 값이 빈 문자열 대신 NULL 값으로 잘못 캡처되는 문제를 해결했습니다.

해결된 문제	설명
Oracle LOB 조회 메모리 누수 문제	전체 LOB 모드가 활성화된 경우 LOB 조회 중에 Oracle 소스 엔드포인트의 메모리 누수가 수정되었습니다. 이렇게 하면 LOB 데이터를 사용한 복제 작업 중에 메모리가 지속적으로 증가하거나 out-of-memory 오류가 발생하는 것을 방지할 수 있습니다.
Redshift CDC 부울 열 매핑 문제	이제 Redshift 대상 엔드포인트는 MapBooleanAsBoolean이 활성화된 경우 CDC 중에 추가된 새 부울 열을 올바르게 매핑합니다. 이렇게 하면 varchar 열을 생성하는 대신 PostgreSQL 소스와 데이터 형식의 일관성을 유지할 수 있습니다.
PostgreSQL UUID 배열 인라인 LOB 마이그레이션 문제	이제 인라인 LOB 모드가 활성화된 경우 PostgreSQL 소스 엔드포인트가 UUID 배열 데이터 형식을 올바르게 마이그레이션합니다.
데이터 검증 nullable column diff 보고 문제	이제 데이터 검증은 null 가능한 열에 대한 불일치 세부 정보를 올바르게 보고합니다.
SQL Server에서 Oracle로의 검증 필터링 문제	이제 데이터 검증은 SQL Server 소스를 사용할 때 Oracle 대상 엔드포인트에 대한 날짜 필터 조건의 형식을 올바르게 지정합니다.
DO_NOTHING 모드 테이블 생성 경고 메시지	가 DO_NOTHING 모드에서 대상 테이블을 AWS DMS 자동으로 생성할 때 경고 로그 메시지가 추가되었습니다. 이렇게 하면 자동 테이블 생성이 없음을 암시하는 모드를 선택하더라도 가시성이 향상되고 사용자가 테이블 생성 시기를 식별할 수 있습니다.
SQL Server BIT에서 PostgreSQL로의 SMALLINT 변환 문제	CDC 작업 중에 SQL Server BIT에서 PostgreSQL SMALLINT로의 변환을 수정했습니다.

해결된 문제	설명
S3 교차 계정 버킷 액세스 검증 문제	S3를 마이그레이션 대상 또는 소스로 사용할 때 보안을 개선하기 위해 S3 버킷 소유권 검증을 개선했습니다.
Redshift 엔드포인트 재연결 및 초기화 문제	이제 Redshift 엔드포인트가 연결 종료 후 올바르게 다시 연결되고 초기화 중에 자격 증명을 올바르게 검증합니다. 이렇게 하면 복구 중에 "서버 이름을 입력해야 함" 오류 및 작업 실패가 방지됩니다.
데이터 검증 메모리 누수 문제	데이터 검증에서 메모리 누수를 수정했습니다. 이렇게 하면 장기 실행 검증 작업 중에 out-of-memory 오류가 방지됩니다.
SQL Server CHAR 기본 키 검증 거짓 긍정 문제	후행 공백이 포함된 CHAR/NCHAR 기본 키를 사용하여 SQL Server에서 PostgreSQL로의 마이그레이션을 검증할 때 발생하는 오탐지 MISSING_TARGET 오류를 수정했습니다.
S3 검증 Athena 데이터베이스 충돌 문제	여러 태스크가 태스크당 Athena 데이터베이스를 격리하여 S3 엔드포인트를 공유하는 경합 조건을 수정했습니다.

AWS Database Migration Service 3.6.0 릴리스 정보

AWS DMS 3.6.0의 새로운 기능

새로운 기능 또는 향상된 기능	설명
변환을 위한 새로운 메타데이터 변수	변환 규칙에 대한 두 가지 새로운 메타데이터 변수인 \$AR_M_MODIFIED_SCHEMA 및 \$AR_M_MODIFIED_TABLE_NAME 을 도입했습니다. 이러한 변수를 사용하면 마이그레이션 프로세스 중에 스키마 또는 테이블 이름 변경에 맞게 조정할 수 있는 보다 역동적이고 유연한 변환을 생성할 수 있습니다.

새로운 기능 또는 향상된 기능	설명
UPSERT 오류 처리 모드에서 LOB 열 복제 지원	"UPDATE: Insert the missing target record" 오류 처리 옵션을 사용할 때 대형 객체(LOB) 열 복제에 대한 지원을 도입했습니다. 이 향상된 기능을 통해 이제 LOB 열을 정확하게 복제하여 완전하고 정확한 데이터 복제를 보장할 수 있습니다.

AWS DMS 버전 3.6.0에는 다음과 같은 해결된 문제가 포함되어 있습니다.

DMS 3.6.0에서 해결된 문제

해결된 문제	설명
PostgreSQL 메모리 문제	PostgreSQL을 소스로 사용할 때 과도한 메모리 소비를 해결했습니다. 미리 쓰기 로그(WAL) 슬롯이 지속적으로 증가하여 성능이 저하되는 문제를 해결했습니다. 이 업데이트는 특히 대규모 데이터베이스 또는 장기 실행 작업의 경우 PostgreSQL 마이그레이션의 안정성과 효율성을 향상시킵니다.
Oracle Character Large Object(CLOB) 및 Character(CHAR) 문제	ASCII가 아닌 문자(예: 특수 기호 또는 국제 문자)를 포함하는 넓은 문자 대형 객체(CLOB) 및 문자(CHAR) 값이 올바르게 복제되지 않는 문제를 해결했습니다. 이 수정 사항을 통해 대용량 텍스트 데이터를 정확하게 복제하여 오류를 줄이고 데이터 일관성을 유지할 수 있습니다.
Microsoft SQL Server 잘못된 지연 시간 문제	Microsoft SQL Server 소스 엔드포인트가 매우 긴 지연 시간을 잘못 보고하는 문제를 해결했습니다. 이 수정 사항은 보다 정확한 성능 지표를 제공하므로 SQL Server 마이그레이션 작업을 더 잘 모니터링하고 최적화할 수 있습니다.
Microsoft SQL Server 메타데이터 문제	다른 테이블의 메타데이터를 검색하기 위한 중복 호출 문제를 수정하여 테이블의 메타데이터가 변경된 후 성능과 지연 시간이 저하되었습니다. 이 최적화는 특히 스키마 변경이 빈번한 데이터베이스의 경우 전반적인 작업 성능을 개선합니다.
Microsoft SQL Server 업데이트 문제	비표준 UPDATE 작업으로 인해 오류 메시지를 생성하지 않고 작업이 예기치 않게 종료되는 문제를 수정했습니다. 이 수정을 통해 이러

해결된 문제	설명
	한 작업이 올바르게 처리되므로 복잡한 업데이트 시나리오 중에 작업이 실패하지 않습니다.
대형 객체 일괄 적용(LOB) 문제	DELETE 작업과 INSERT 작업을 단일 UPDATE 작업으로 결합할 때 LOB(대형 객체) 조회가 레코드를 찾지 못하는 배치 최적화 적용 모드 문제를 수정했습니다. 이 수정 사항은 LOB 데이터 및 복잡한 데이터 조작 언어(DML) 작업과 관련된 마이그레이션 중에 데이터 일관성과 무결성을 개선합니다.
변환 작업 시작 문제	시작 중에 수많은 변환과 관련된 작업이 충돌하는 문제를 해결했습니다. 이 수정 사항은 복잡한 변환 로직이 있는 작업의 안정성과 신뢰성을 보장하므로 데이터 처리 작업을 자신 있게 실행할 수 있습니다.
MySQL 데이터 정의 언어 (DDL) 문제	변경 데이터 캡처(CDC) 단계에서 특수 형식 데이터 정의 언어(DDL) 변경 사항을 캡처하지 못하는 문제를 수정했습니다. 이 수정 사항을 통해 모든 스키마 변경 사항이 올바르게 복제되어 마이그레이션 프로세스 전반에 걸쳐 소스 데이터베이스와 대상 데이터베이스 간의 스키마 일관성을 유지할 수 있습니다.
S3 교차 계정 버킷 액세스 검증 문제	S3를 마이그레이션 대상 또는 소스로 사용할 때 보안을 개선하기 위해 S3 버킷 소유권 검증을 개선했습니다.
데이터 검증 메모리 누수 문제	데이터 검증에서 메모리 누수를 수정했습니다. 이렇게 하면 장기 실행 검증 작업 중에 out-of-memory 오류가 방지됩니다.

AWS Database Migration Service 3.5.4 릴리스 정보

AWS DMS 3.5.4의 새로운 기능

새로운 기능 또는 향상된 기능	설명
데이터 마스킹	열 수준에서 숫자 무작위화, 마스킹 또는 해싱 옵션으로 민감한 데이터를 변환할 수 있도록 데이터 마스킹을 도입했습니다.

새로운 기능 또는 향상된 기능	설명
향상된 데이터 검증 성능	일부 마이그레이션 경로에서 전체 로드 및 CDC 마이그레이션 작업 중에 대규모 데이터셋을 더 빠르게 처리할 수 있도록 향상된 데이터 검증 성능을 도입했습니다.
PostgreSQL 소스 유니코드 문제	필터링을 사용하는 동안 마이그레이션 성능이 저하되는 PostgreSQL 소스 문제를 수정했습니다. 이 동작을 제어하기 위해 disableUnicodeSourceFilter ECA를 도입했습니다.
향상된 처리량 기능을 위한 변환 지원	향상된 처리량 기능을 위한 모든 변환 규칙에 대한 지원을 소개합니다.
누락된 대상 열에 대한 향상된 로깅	기본 로깅 수준에서 누락된 대상 테이블 열을 고객에게 알리도록 로깅을 개선했습니다. 이 개선 사항은 로깅 수준을 VERBOSE에서 WARNING으로 변경하므로 세부 디버그 설정 없이 열 불일치 알림을 더 잘 볼 수 있습니다.
SQL Server Binary(16)에서 PostgreSQL UUID로 마이그레이션	SQL Server Binary(16) 데이터 형식을 PostgreSQL UUID 형식으로 마이그레이션하기 위한 지원이 추가되었습니다. 이를 통해 바이너리 GUID 데이터를 네이티브 UUID 유형으로 원활하게 변환하여 SQL Server와 PostgreSQL 엔드포인트 간의 데이터 유형 호환성을 개선할 수 있습니다.
PostgreSQL 17 지원	PostgreSQL 버전 17에 대한 지원이 도입되었습니다. 자세한 내용은 다음을 참조하세요. • PostgreSQL 데이터베이스를 AWS DMS 소스로 사용 • PostgreSQL 데이터베이스를 AWS Database Migration Service의 대상으로 사용

AWS DMS 버전 3.5.4에는 다음과 같은 해결된 문제가 포함되어 있습니다.

DMS 3.5.4에서 해결된 문제

해결된 문제	설명
PostgreSQL, test_decoding 문제	test_decoding 플러그인을 사용하는 동안 특정 이벤트가 복제되지 않는 PostgreSQL 소스 문제를 수정했습니다.
MySQL에서 Redshift로의 타임스탬프 문제	대상에 타임스탬프 열이 올바르게 정의되지 않는 MySQL에서 Redshift로의 마이그레이션 문제를 수정했습니다.
Oracle 2024년 7월 PSU 문제	Oracle 2024년 7월 PSU를 적용한 후 DMS 작업이 충돌하는 바이너리 리더를 사용하는 Oracle 소스 문제를 수정했습니다.
MySQL 보안 암호 관리자 문제	보안 암호 관리자를 사용할 때 자격 증명이 손상되는 MySQL 엔드포인트 문제를 수정했습니다.
Amazon DocumentDB/MongoDB 데이터 레코드 처리 문제	Amazon DocumentDB/MongoDB 엔드포인트에서 일부 레코드가 대상으로 두 번 전송되어 중복 키 예외 및 AWS DMS 작업 실패가 발생하는 문제를 수정했습니다.
noSQL 마이그레이션 문제와 관련	잘못된 PK 처리로 인해 문서 구조가 잘못된 경우 RDS for SQL Server에서 noSQL 마이그레이션 문제를 수정했습니다.
Oracle 엔드포인트의 데이터 검증 문제	Oracle 소스에서 데이터 검증이 null 또는 빈 LOB.
Babelfish 대상의 고유 식별자 PK 문제입니다.	PK가 고유 식별자로 정의된 테이블을 복제하는 동안 AWS DMS 작업이 실패하는 PG - Babelfish 대상 문제를 수정했습니다.
MAZ와 관련된 PostgreSQL 소스 문제입니다.	AWS DMS MAZ 장애 조치로 인해 치명적인 AWS DMS 작업 실패가 발생하는 PostgreSQL 소스 문제를 수정했습니다.
열 순서 문제	열 순서가 소스와 대상 간에 다를 때 LOB 데이터가 올바르게 복제되지 않는 문제를 수정했습니다.
내부 AWS DMS 데이터베이스 경합 문제	내부 데이터베이스 동시성 문제로 인해 AWS DMS 작업이 실패하는 내부 DMS AWS DMS 데이터베이스 문제를 수정했습니다.
내부 AWS DMS 데이터베이스 구조 문제	특정 내부 AWS DMS 데이터베이스 객체가 없어 AWS DMS 작업이 실패하는 내부 데이터베이스 문제를 수정했습니다.

해결된 문제	설명
Oracle 소스 데이터 검증 문제	드문 유형의 복제된 특정 이벤트에 대해 데이터 검증에서 오탐지가 반환되는 Oracle 소스 문제를 수정했습니다.
유니코드 데이터 형식에 대한 데이터 검증 문제	ceratin 유니코드 데이터 형식이 올바르게 비교되지 않아 오탐지가 발생하는 데이터 검증 기능의 문제를 수정했습니다.
Parquet 대상 타임스탬프 문제	제로 타임스탬프가 null로 복제되는 parquet 대상 문제를 수정했습니다.
Babelfish 대상 GeoSpatial 데이터 유형 문제입니다.	Babelfish 대상에서 GeoSpatial 데이터 형식이 지원되지 않는 문제를 수정했습니다.
CDC 중에 열이 추가되는 Amazon S3 대상 문제	이미지 이전 설정이 활성화되었을 때 새 열 추가가 제대로 처리되지 않는 Amazon S3 대상 문제를 수정했습니다.
SQL Server 2022 CU12 문제	CU12 이상에서 SQL Server 2022를 사용하는 소스에서 MS Replication 사전 조건을 자동으로 구현 AWS DMS 할 수 없는 SQL Server 소스 문제를 수정했습니다.
PostgreSQL 부울 문제	MapBooleanAsBoolean 이 true로 설정되고 pglogical 플러그인이 사용된 동안 부울 데이터 유형이 올바르게 마이그레이션되지 않는 PostgreSQL 소스 문제를 수정했습니다.
TaskrecoveryTableEnabled 문제 설정	true로 TaskrecoveryTableEnabled 설정하면 AWS DMS 작업 중지 시 작업이 실패하는 설정 문제를 수정했습니다.
TaskrecoveryTableEnabled 설정을 사용한 데이터 복제	TaskrecoveryTableEnabled 설정이 활성화되었을 때 일부 트랜잭션이 두 번 복제되는 문제를 수정했습니다.
MySQL 5.5 소스 문제	MySQL v5.5BINLOG에서 읽을 수 없어 AWS DMS 작업이 실패하는 MySQL 소스 문제를 수정했습니다.
손상된 데이터 유형 이름의 데이터 검증 파티션 문제	이제 데이터 검증은 데이터 유형 처리 중 메모리 손상을 방지하여 영향을 받는 파티션에 대한 row-by-row 검증으로의 폴백을 제거합니다.

해결된 문제	설명
Amazon S3 소스에서 대상 마이그레이션 문제	S3 소스-대상 복제는 이제 전체 로드 및 지속적 복제 중에 외부 테이블 객체를 올바르게 처리합니다.
Amazon S3 대상 CDC 복제 문제	S3 대상 CDC 복제는 지속적 복제 단계에서 CSV 형식 데이터를 올바르게 처리합니다.
S3 복제 문제	향상된 S3 디렉터리 정리 프로세스는 마이그레이션 중에 작업이 중단되는 것을 방지합니다.
데이터 검증 성능 문제	이제 데이터 검증 작업은 검증 단계 간 전환을 최적화하여 불필요한 지연을 줄입니다.
특정 데이터 유형의 데이터 검증 문제	이제 데이터 검증은 결합되지 않은 문자와 TEXT 데이터 유형을 정확하게 처리하여 올바른 검증 결과를 보장합니다.
MAZ와 관련된 PostgreSQL 소스 문제	PostgreSQL 소스 복제는 다중 AZ 장애 조치 이벤트 중에 연결을 유지하여 작업 실패를 방지합니다.
Babelfish 날짜/시간 검증 문제	이제 Babelfish를 대상으로 사용할 때 데이터 검증에서 날짜/시간 값을 올바르게 비교합니다.
MySQL 소스 열 복제 문제	MySQL 소스 복제는 이제 테이블 중간 중복 열 추가를 올바르게 처리하여 작업 종단을 방지합니다.
MySQL 소스 열 수정 문제	MySQL 소스 복제는 CDC 작업 중에 여러 열이 추가될 때 열 시퀀스 무결성을 유지합니다.
DynamoDB LOB 복제 문제	DynamoDB 대상 복제는 이제 CDC 중에 LOB 데이터를 올바르게 처리하여 완전한 데이터 전송을 보장합니다.
PostgreSQL 부울 데이터 검증 문제	이제 PostgreSQL 소스 데이터 검증은 부울 데이터 형식 매핑을 올바르게 해석하여 정확한 비교 결과를 생성합니다.
DocumentDB 연결 복구 문제	DocumentDB 복제는 클러스터 연결 중단 및 복구 중에 데이터 일관성을 유지합니다.
Oracle 확장 VARCHAR2 데이터 잘림 문제	Oracle 소스 복제는 확장 데이터 유형 지원이 활성화된 경우 VARCHAR2(4000) 열의 후행 공간을 보존합니다.

해결된 문제	설명
SQL Server 보조 복제본 DDL 처리 문제	SQL Server 소스 복제는 보조 복제본에서 DDL 작업 중에 연결을 유지하여 작업 중단을 방지합니다.
Secret Manager 특수 문자 처리 문제	AWS Secrets Manager 연결 문자열은 이제 보안 프로토콜을 유지하면서 특수 문자를 지원합니다.
MongoDB/Amazon DocumentDB 중복 키 문제	MongoDB 및 Amazon DocumentDB 복제는 이전에 키 제약 조건 오류를 트리거한 레코드 중복을 방지합니다.
Oracle 타임스탬프 처리 문제	Oracle 소스 복제는 다양한 세션 시간대 구성에서 타임스탬프 값을 정확하게 처리합니다.
Oracle 데이터 변환 문제	Oracle 소스 복제는 이제 데이터 유형 변환을 더 강력하게 처리하여 ORA-01460 오류를 방지합니다.
SQL Server 2022 CDC 복제 문제	SQL Server 소스(CU12 이상)는 이제 AWS DMS에서 MS Replication 사전 조건을 자동으로 구현할 수 있습니다.
DDL 처리 디코딩 테스트	AWS DMS 는 테이블 매핑에 구성되지 않은 테이블에 대한 DDL 문을 무시하므로 매핑되지 않은 테이블에 대한 소스 데이터베이스 스키마 변경 사항이 불필요하게 처리되지 않습니다.
소스 SQL Server 뷰에서 열 잘림	소스 SQL Server 뷰에서 대상 엔드포인트로 데이터를 마이그레이션할 때 열 길이가 올바르게 보존되고 열이 잘리지 않습니다.
향상된 데이터 검증을 위한 지표 기록	AWS DMS 이제는 향상된 데이터 검증을 사용할 때 검증된 레코드 지표를 정확하게 보고합니다.
업그레이드 후 DMS 작업 재개 문제	AWS DMS 버전 업그레이드 후 작업을 재개할 때 내부 파일 형식이 호환되지 않아 DMS 작업이 재개되지 않습니다.
SQL Server Always On 기본 복제본 문제	AWS DMS 이제 Microsoft SQL Server Always On 가용성 그룹에서 기본 복제본을 올바르게 식별하여 이전 대/소문자 구분 감지 오류를 해결하는 문제를 수정했습니다.
PostgreSQL 대상 숫자 데이터 처리 문제	무한 숫자 데이터 형식을 PostgreSQL 대상 엔드포인트로 마이그레이션할 때 AWS DMS 작업이 실패하는 문제를 수정했습니다.

해결된 문제	설명
데이터 검증 문자 처리 문제	AWS DMS 이제는 마이그레이션 중에 CHAR 및 VARCHAR 데이터를 올바르게 검증하여 검증 작업에서 거짓 긍정 보고서를 제거합니다.
Amazon Redshift LOB 병렬 적용 문제	AWS DMS Amazon Redshift를 대상 엔드포인트로 사용할 때 병렬 배치 작업 중에 대형 객체(LOB) 데이터의 데이터 손상을 방지합니다.
열 필터 문제가 있는 데이터 검증	VARCHAR, CHAR, DATE 및 DATETIME 데이터 유형에 소스 열 필터를 적용하면 검증 실패가 방지되고 Amazon S3 대상에서 쿼리가 성공적으로 실행됩니다.
S3 검증 데이터 지속성 문제	Amazon S3 대상을 사용하는 경우 변경 데이터 캡처(CDC) 작업 전체에서 일관된 데이터 검증 상태가 유지됩니다.
문자열 형식 확인 문제	레코드 비교 작업 중 문자열 형식 오류로 인해 데이터 검증이 실패하는 문제를 수정했습니다.
DB2 빈 문자열 CDC 마이그레이션 문제	변경 데이터 캡처(CDC) 작업 중에 IBM DB2 LUW 소스 데이터베이스의 빈 VARCHAR 값이 빈 문자열 대신 NULL 값으로 잘못 캡처되는 문제를 해결했습니다.
Oracle LOB 조회 메모리 누수 문제	전체 LOB 모드가 활성화된 경우 LOB 조회 중에 Oracle 소스 엔드포인트의 메모리 누수가 수정되었습니다. 이렇게 하면 LOB 데이터를 사용한 복제 작업 중에 메모리가 지속적으로 증가하고 out-of-memory 오류가 발생하는 것을 방지할 수 있습니다.
Redshift CDC 부울 열 매핑 문제	이제 Redshift 대상 엔드포인트는 MapBooleanAsBoolean이 활성화된 CDC 중에 추가된 새 부울 열을 올바르게 매핑합니다. 이렇게 하면 varchar 열을 생성하는 대신 PostgreSQL 소스와 데이터 형식의 일관성을 유지할 수 있습니다.
PostgreSQL UUID 배열 인라인 LOB 마이그레이션 문제	이제 인라인 LOB 모드가 활성화된 경우 PostgreSQL 소스 엔드포인트가 UUID 배열 데이터 형식을 올바르게 마이그레이션합니다.
데이터 검증 nullable column diff 보고 문제	이제 데이터 검증은 null 가능한 열에 대한 불일치 세부 정보를 올바르게 보고합니다.

해결된 문제	설명
SQL Server에서 Oracle로의 검증 필터링 문제	이제 데이터 검증은 SQL Server 소스를 사용할 때 Oracle 대상 엔드 포인트에 대한 날짜 필터 조건의 형식을 올바르게 지정합니다.
DO_NOTHING 모드 테이블 생성 경고 메시지	가 DO_NOTHING 모드에서 대상 테이블을 AWS DMS 자동으로 생성할 때 경고 로그 메시지가 추가되었습니다. 이렇게 하면 자동 테이블 생성이 없음을 암시하는 모드를 선택하더라도 가시성이 향상되고 사용자가 테이블 생성 시기를 식별할 수 있습니다.
SQL Server BIT에서 PostgreSQL로의 SMALLINT 변환 문제	CDC 작업 중에 SQL Server BIT에서 PostgreSQL SMALLINT로의 변환을 수정했습니다.
S3 교차 계정 버킷 액세스 검증 문제	S3를 마이그레이션 대상 또는 소스로 사용할 때 보안을 개선하기 위해 S3 버킷 소유권 검증을 개선했습니다.
Redshift 엔드포인트 재연결 및 초기화 문제	이제 Redshift 엔드포인트가 연결 종료 후 올바르게 다시 연결되고 초기화 중에 자격 증명을 올바르게 검증합니다. 이렇게 하면 복구 중에 "서버 이름을 입력해야 함" 오류 및 작업 실패가 방지됩니다.
데이터 검증 메모리 누수 문제	데이터 검증에서 메모리 누수를 수정했습니다. 이렇게 하면 장기 실행 검증 작업 중에 out-of-memory 오류가 방지됩니다.
SQL Server CHAR 기본 키 검증 거짓 긍정 문제	후행 공백이 포함된 CHAR/NCHAR 기본 키를 사용하여 SQL Server에서 PostgreSQL로의 마이그레이션을 검증할 때 발생하는 오탐지 MISSING_TARGET 오류를 수정했습니다.
S3 검증 Athena 데이터베이스 충돌 문제	여러 태스크가 태스크당 Athena 데이터베이스를 격리하여 S3 엔드포인트를 공유하는 경합 조건을 수정했습니다.

AWS Database Migration Service 3.5.3 릴리스 정보

AWS DMS 3.5.3의 새로운 기능

새로운 기능 또는 향상된 기능	설명
Babelfish 지원을 위한 향상된 PostgreSQL 소스 엔드포인트	AWS DMS 는 Babelfish 데이터 유형을 지원하도록 PostgreSQL 소스 엔드포인트를 개선했습니다. 자세한 내용은 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용 단원을 참조하십시오.
소스로 S3 Parquet 지원	AWS DMS 는 S3 Parquet을 소스로 지원합니다. 자세한 내용은 Amazon S3를의 소스로 사용 AWS DMS 섹션을 참조하세요.
PostgreSQL 16.x 지원	AWS DMS 는 PostgreSQL 버전 16.x를 지원합니다. 자세한 내용은 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용 및 PostgreSQL 데이터베이스를의 대상으로 사용 AWS Database Migration Service 섹션을 참조하세요.
Oracle에서 Amazon Redshift로의 전체 로드 마이그레이션을 위한 향상된 처리량	AWS DMS 서버리스는 Oracle에서 Amazon Redshift로의 전체 로드 마이그레이션을 위해 처리량 성능을 크게 개선합니다. 자세한 내용은 Oracle에서 Amazon Redshift 및 Amazon S3로의 전체 로드 마이그레이션을 위한 향상된 처리량 단원을 참조하십시오.
RDS for SQL Server 로그 백업에 대한 지원	AWS DMS 는 RDS for SQL Server 소스에 대한 트랜잭션 로그 백업에서 지속적인 복제를 지원합니다. 이 기능은 RDS for SQL Server에서 읽는 신규 및 수정된 AWS DMS 소스 엔드포인트에서만 사용할 수 있습니다. 자세한 내용은 클라우드 SQL Server DB 인스턴스에서 지속적 복제 설정 단원을 참조하십시오.
누락된 대상 열에 대한 향상된 로깅	기본 로깅 수준에서 누락된 대상 테이블 열을 고객에게 알리도록 로깅을 개선했습니다. 이 개선 사항은 로깅 수준을 VERBOSE에서 WARNING으로 변경하므로 세부 디버그 설정 없이 열 불일치 알림을 더 잘 볼 수 있습니다.

AWS DMS 버전 3.5.3에는 다음과 같은 해결된 문제가 포함되어 있습니다.

2024년 5월 17일에 발표된 DMS 3.5.3에서 해결된 문제

해결된 문제	설명
데이터 검증 재정의 함수	<code>override-validation-function</code> 테이블 매핑에서 규칙 작업이 로 설정된 경우가 소스 필터링을 준수하지 AWS DMS 애플리케이션 데이터 검증 기능의 문제를 수정했습니다.
MySQL 소스 CDC 오류	UTF16 인코딩으로 CDC 마이그레이션이 실패하는 소스로 MySQL에 대한 문제를 수정했습니다.
데이터 검증 데이터 정렬 차이	열 필터링을 사용할 때가 <code>HandleCollationDiff</code> 작업 설정을 제대로 적용 AWS DMS 하지 않는 데이터 검증 기능의 문제를 수정했습니다.
데이터 검증 태스크 중단.	가 null인 오류와 함께 AWS DMS 작업이 중단되는 데이터 검증 기능의 문제를 수정했습니다.
PostgreSQL에서 PostgreSQL로의 복제 태스크 실패.	CDC 복제 중에 대상에 LOB 데이터를 삽입하는 동안 AWS DMS 작업이 실패하는 PostgreSQL에서 PostgreSQL로의 마이그레이션 문제를 수정했습니다.
PostgreSQL을 소스로 사용하는 데이터 손실	특정 엣지 사례 시나리오에서 데이터 손실이 발생하는 소스로서 PostgreSQL에 대한 문제를 수정했습니다.
MySQL 5.5 소스 CDC 오류	MySQL 버전 5.5에서 CDC 복제가 실패하는 소스로 MySQL에 대한 문제를 수정했습니다.
Oracle 소스 IOT 테이블 문제.	모든 열에서 보충 로깅이 활성화된 IOT 테이블에 대해가 UPDATE 문을 올바르게 복제하지 AWS DMS 못하는 소스로서의 Oracle 문제를 수정했습니다.
MySQL 소스 LOB	LOBs 초과하여 AWS DMS 작업이 실패하는 MySQL에서 Amazon Redshift로의 마이그레이션 문제를 수정했습니다.
SkipLobColumns 의 검증 문제	기본 키가 소스 테이블의 마지막 열에 있을 SkipLobColumns = true 때 AWS DMS 작업이 실패하는 데이터 검증 기능의 문제를 수정했습니다.

해결된 문제	설명
고유 키가 null인 검증 건너뛰기	가 null 고유 키가 있는 행을 제대로 건너뛰지 않는 데이터 검증 기능의 문제를 수정 AWS DMS 했습니다.
Oracle COLLATE 연산자에 대한 데이터 검증 개선 사항.	12.2 이전 Oracle 버전에서 구문 오류와 함께 검증이 실패하는 데이터 검증 기능에 대한 문제를 수정했습니다.
전체 로드 중 오류 처리	잘못된 데이터로 인해 테이블 오류가 발생한 후 전체 로드 단계에서 태스크가 중단되는 대상으로서 PostgreSQL의 문제를 해결했습니다.
CDC 검증 전용 태스크의 재검증	CDC 검증 전용 태스크에서 재검증을 허용하도록 데이터 검증 기능을 개선했습니다.
대상으로서 S3 CdcMaxBatchInterval Out of Memory 문제	CdcMaxBatchInterval이 설정된 메모리 부족 조건에서 AWS DMS 작업이 실패하는 S3를 대상으로 사용하는 문제를 수정했습니다.
Oracle 소스 드라이버	AWS DMS Oracle 소스 드라이버를 v12.2에서 v19.18로 업그레이드했습니다.
SQL Server 소스를 사용한 LOB 잘림 경고	CDC 중 LOB 잘림에 대한 경고를 표시하기 위한 소스로서 SQL Server에 대한 로깅을 개선했습니다.
Oracle 바이너리 리더 개선 사항	다음을 지원하도록 Oracle 소스 바이너리 리더를 개선했습니다. • Big Endian 플랫폼 • HCC 압축을 사용한 병렬 DML 힌트 • Golden Gate가 활성화된 고급 Oracle 압축
중단 후 캐시된 변경 사항 건너뛰기 문제	변경 사항을 적용하는 동안 예기치 않은 중단이 발생할 경우 캐시된 변경 사항을 건너뛰는 문제를 수정했습니다.
MongoDB AWS DMS 스왑 파일 문제	AWS DMS 스왑 파일이 있을 때 AWS DMS 업그레이드 후 AWS DMS 작업이 재개되지 않는 MongoDB 소스 문제를 수정했습니다.

해결된 문제	설명
MySQL에서 Amazon Redshift JSON으로 마이그레이션 문제.	배치 적용이 활성화된 상태에서 JSON 데이터 형식이 고립되지 않는 MySQL 소스 문제를 수정했습니다.
SQL Server에서 APG로의 마이그레이션에 대한 데이터 검증 문제입니다.	SQL Server에서 PostgreSQL로 마이그레이션할 때 데이터 유효성이 특정 상황에서 거짓 긍정을 보고하는 문제를 수정했습니다.
Amazon Redshift 대상 CDC에 대한 EmptyAsNull ECA 문제	Amazon Redshift 대상에서 EmptyAsNull ECA가 제대로 작동하지 않는 문제를 수정했습니다.
SQL Server 소스에 대한 로깅 개선 사항	스토리지 단위 값을 제외하지 않도록 SQL Server 소스에 대한 로깅을 개선했습니다.
메모리 누수 문제	CSV 파일을 사용하여 데이터 AWS DMS 작업을 로드하는 대상에서 메모리 누수가 표시되는 문제를 수정했습니다.
Amazon S3 대상 문제	cdcInsertAndUpdateOnly 설정이 활성화되었을 때 CdcMaxBatchInterval 및 CdcMinFileSize를 준수하지 않는 Amazon S3 대상 문제를 수정했습니다.
Kafka 대상에 대한 로깅 개선 사항	Kafka 대상에 대한 향상된 로깅
Oracle 소스에 대한 로깅 개선 사항입니다.	프라이머리 키 누락으로 인해 테이블을 건너뛰고 있음을 올바르게 나타내기 위해 이진 리더를 사용하는 Oracle 소스에 대한 로깅을 개선했습니다.
SQL Server 소스에 대한 로깅 개선 사항	누락된 권한을 올바르게 나타내도록 AlwaysOn 구성에서 SQL Server 소스에 대한 로깅을 개선했습니다.
DDL이 비활성화된 마이그레이션에 대한 로깅 개선 사항	외부에서 수정된 후 예기치 않은 대상 테이블 구조를 나타내도록 DDL 복제가 비활성화된 마이그레이션에 대한 로깅을 개선했습니다 AWS DMS.

해결된 문제	설명
Db2 대상 문제	AWS DMS 상태 테이블이 활성화된 경우 작업이 실패하는 Db2 대상 문제를 수정했습니다.
MongoDB/Amazon DocumentDB Secret Manger 문제입니다.	Secret Manager에서 자격 증명을 검색할 수 없어 오류가 발생한 MongoDB/Amazon DocumentDB 엔드포인트 문제를 수정했습니다.
MongoDB/Amazon DocumentDB 엔드포인트 문제	특정 이벤트 시퀀스를 복제하는 동안 ParallelApply가 활성화된 상태에서 작업이 실패하는 MongoDB/Amazon DocumentDB 문제를 수정했습니다.
Amazon Redshift 대상에 대한 로깅 개선 사항	기본 로깅 수준에 자세한 정보를 포함하도록 Amazon Redshift 대상에 대한 로깅을 개선했습니다.
Amazon S3 대상 작업 충돌 문제	GlueCatalogGeneration 이 활성화된 경우 테이블 DDL 변경을 수신한 후 DMS 작업이 충돌하는 Amazon S3 대상 문제를 수정했습니다.
데이터 유효성 검사 문제	NUL (0x00) 문자에 대한 검증이 실패하는 데이터 검증 기능의 문제를 수정했습니다.
Babelsfh 엔드포인트 문제	대소문자가 혼합된 테이블 이름이 일시 중지되는 babelfish 엔드포인트 문제를 수정했습니다.
Amazon S3 소스 문제	파일 이름 검증 문제로 인해 파일이 처리되지 않는 Amazon S3 소스 문제를 수정했습니다.
선택 규칙과 관련된 Db2 LUC 소스 문제입니다.	선택 규칙의 "table-type" 옵션이 무시되는 Db2 LUW 소스 문제를 수정했습니다.
Amazon Redshift 대상 데이터 누락 문제.	Amazon Redshift 대상에서 특정 조건에서 ParallelLoadThreads 가 >0일 때 데이터 손실이 발생하는 문제를 수정했습니다.
Amazon Redshift 대상에 대한 데이터 검증 개선 사항	HandleCollationDiff 설정을 지원하도록 Amazon Redshift 대상에 대한 데이터 검증 기능을 개선했습니다.

해결된 문제	설명
Amazon S3 대상 데이터 검증 문제	테이블에 PK 이외의 열이 없을 때 검증이 실패하는 Amazon S3 대상 데이터 검증 문제를 수정했습니다.
데이터 검증 CloudWatch 지표	검증을 위해 CloudWatch 지표가 누락되어 완료하는 데 짧은 시간이 걸리는 데이터 검증 기능의 문제를 수정했습니다.
데이터 검증 재검증 옵션 사용 불가	특정 상황에서 재검증 옵션을 사용할 수 없는 데이터 검증 기능의 문제를 수정했습니다.
대규모 Transaction 복제 문제	특정 조건에서 트랜잭션당 최대 이벤트 수가 201,326,592개로 제한되는 문제를 수정했습니다.
MySQL 소스 데이터 손실 문제	MySQL에서 Amazon S3로 마이그레이션할 때 "열 추가" DDL 이후에 실행된 첫 번째 DML이 누락되어 데이터가 손실되는 문제를 수정했습니다.
일괄 메모리 누수 문제 적용	특정 조건에서 발생하는 배치 적용 기능의 메모리 누수 문제를 수정했습니다.
DMS 작업 시작 문제	AWS DMS 태스크 시작이 매우 오래 걸리고 완료되지 않는 문제를 수정했습니다.
MySQL 소스 데이터 검증 문제입니다.	자동 증가된 열이 제대로 처리되지 않는 MySQL 소스 데이터 검증 문제를 수정했습니다.
Oracle 엔드포인트 Kerberos 인증 문제	Kerberos 인증에서 연결이 작동하지 않는 Oracle 엔드포인트 문제를 수정했습니다.
Babelfish 대상 고유 식별자 문제	프라이머리 키가 고유 식별자로 정의된 테이블에 대한 복제가 실패하는 Babelfish 대상 문제를 수정했습니다.
PostgreSQL Source 누락된 데이터 문제	복제 슬롯의 알 수 없는 이벤트로 인해 데이터 손실이 발생하는 PostgreSQL 소스 문제를 수정했습니다.
유니코드 데이터 형식에 대한 데이터 검증 문제입니다.	ceratin 유니코드 데이터 형식이 올바르게 비교되지 않아 오탐지가 발생하는 데이터 검증 기능의 문제를 수정했습니다.

해결된 문제	설명
Redshift 대상 타임스탬프 열	타임스탬프 열 유형이 올바르게 설정되지 않은 ColumnType 정의 관련 문제를 수정했습니다
S3 대상 타임스탬프 변환	타임스탬프 값 0개를 처리하여 S3 대상 Parquet 파일 형식 관련 문제를 수정했습니다(이 값은 대상에서 NULL로 변환해야 함)
MySQL 소스 및 대상 LOB 열 순서	MySQL 소스 및 대상 마이그레이션 Lob 열 관련 문제를 수정했습니다. 이제 DMS는 LOB 데이터를 작성해야 하는 열을 결정할 때 소스 테이블 대신 대상 테이블의 열 ID를 사용합니다
Babelfish 대상 날짜/시간 데이터 형식	Babelfish 대상에서 반올림 오류를 해결하기 위해 날짜/시간 유형에 대한 소수 초 정밀도를 제한하는 문제를 수정했습니다.
진행 중인 복제(CDC) 중에 병로그를 읽는 동안 MySQL 5.5 작업이 실패합니다.	MySQL 5.5 소스 관련 문제를 수정했으며, DMS가 진행 중인 복제(CDC) 중에 이진 로그 이벤트를 읽지 못할 때 작업 실패를 방지하는 재시도 메커니즘을 추가했습니다.
테스트 디코딩 플러그인을 사용하는 동안 PostgreSQL 소스 Segfault 및 메모리 누수 문제.	Postgres용 test_decoding 플러그인을 사용할 때 특정 온징 복제(CDC) 이벤트를 올바르게 구문 분석하지 못하는 PostgreSQL 소스 문제를 수정했습니다.
DocumentDB 대상 병렬 적용 설정	이 기능을 사용하는 동안 여러 스레드를 사용하지 못하게 하는 병렬 적용 설정이 있는 DocumentDB 대상 문제를 수정했습니다.
Oracle HCC 압축 DIRECT INSERT	누락 및 중복 데이터를 유발하는 병렬 DML 힌트가 있는 Oracle HCC 압축 DIRECT INSERT 문제를 수정했습니다.
Oracle 2024년 7월 CPU 관련 Oracle Binary Reader 문제	Oracle 소스 관련 문제를 수정했습니다. Oracle 2024년 7월 CPU로 인해 바이너리 리더를 사용하는 DMS 작업이 실패했습니다
Babelfish 대상 UUID 프라이머리 키	프라이머리 키가 UUID로 정의된 테이블을 복제하는 동안 DMS 작업이 실패하는 Babelfish 대상 문제를 수정했습니다.

해결된 문제	설명
TaskRecoveryTableEnabled로 실패한 DMS ongoing 복제(CDC) 작업	대상 연결이 종료된 후 DMS가 대상 시스템 테이블 awsdms_txn_state 를 업데이트하려고 시도하는 TaskRecoveryTableEnabled 이 활성화된 문제를 수정했습니다.
PostgreSQL 소스 중복 레코드	TaskrecoveryTableEnabled 설정이 활성화된 경우 일부 트랜잭션이 두 번 복제되는 PostgreSQL 소스 문제를 수정했습니다.
DMS S3 소스/대상 복제 문제	S3 소스에서 S3 대상으로의 문제에서 DMS 작업이 전체 로드 중 및 복제 진행 중 데이터를 복제하지 않는 문제를 수정했습니다.
진행 중인 복제 중 DMS S3 소스 작업 실패	DMS version S3.5.3에 대한 진행 중 복제 중에 DMS 작업이 Seg Faulting되는 S3 소스 문제를 수정했습니다
DMS DB2 소스 CcsidMapping ECA	CcsidMapping에서 DB2 source 관련 문제를 수정했습니다. 이제 코드 페이지가 0이고 데이터가 제대로 마이그레이션되면 CCSID mappin ECA가 작업에 올바르게 적용됩니다.
DMS Aurora PostgreSQL에서 Redshift Serverless 부울 값 복제.	Aurora PostgreSQL에서 Redshift Serverless로의 DMS 마이그레이션에서 부울 값에 문제가 발생하는 문제를 수정했습니다.
특정 데이터 유형의 데이터 검증 문제	이제 데이터 검증 작업은 결합되지 않은 문자와 TEXT 데이터 유형을 정확하게 처리하여 올바른 검증 결과를 보장할 수 있습니다.
MAZ와 관련된 PostgreSQL 소스 문제	PostgreSQL 소스 복제는 다중 AZ 장애 조치 이벤트 중에 연결을 유지하여 작업 실패를 방지합니다.
Babelfish 날짜/시간 검증 문제	이제 데이터 검증은 Babelfish를 대상으로 사용할 때 날짜/시간 값을 올바르게 비교하여 교차 플랫폼 호환성을 개선합니다.
MySQL 소스 열 복제 문제	MySQL 소스 복제는 이제 테이블 중간 중복 열 추가를 올바르게 처리하여 작업 중단을 방지합니다.
MySQL 소스 열 수정 문제	MySQL 소스 복제는 CDC 작업 중에 여러 열이 추가될 때 열 시퀀스 무결성을 유지합니다.

해결된 문제	설명
DynamoDB LOB 복제 문제	DynamoDB 대상 복제는 이제 CDC 중에 LOB 데이터를 올바르게 처리하여 완전한 데이터 전송을 보장합니다.
PostgreSQL 부울 데이터 검증 문제	이제 PostgreSQL 소스 데이터 검증은 부울 데이터 형식 매핑을 올바르게 해석하여 정확한 비교 결과를 생성합니다.
Oracle 확장 VARCHAR2 데이터 잘림 문제	Oracle 소스 복제는 확장 데이터 유형 지원이 활성화된 경우 VARCHAR2(4000) 열의 후행 공간을 보존합니다.
SQL Server 보조 복제본 DDL 처리 문제	SQL Server 소스 복제는 보조 복제본에서 DDL 작업 중에 연결을 유지하여 작업 중단을 방지합니다.
Secret Manager 특수 문자 처리 문제	AWS 이제 Secrets Manager 연결 문자열은 보안 프로토콜을 유지하면서 특수 문자를 지원합니다.
MongoDB/Amazon DocumentDB 중복 키 문제	MongoDB 및 Amazon DocumentDB 복제는 이전에 키 제약 조건 오류를 트리거한 레코드 중복을 방지합니다.
Oracle 타임스탬프 처리 문제	Oracle 소스 복제는 다양한 세션 시간대 구성에서 타임스탬프 값을 정확하게 처리합니다.
Oracle 데이터 변환 문제	Oracle 소스 복제는 이제 데이터 유형 변환을 더 강력하게 처리하여 ORA-01460 오류 및 관련 작업 실패를 방지합니다.
DDL 처리 디코딩 테스트	AWS DMS 는 테이블 매핑에 구성되지 않은 테이블에 대한 DDL 문을 무시하므로 매핑되지 않은 테이블에 대한 소스 데이터베이스 스키마 변경 사항이 불필요하게 처리되지 않습니다.
소스 SQL Server 뷰에서 열 잘림	소스 SQL Server 뷰에서 대상 엔드포인트로 데이터를 마이그레이션할 때 열 길이가 올바르게 보존되고 열이 잘리지 않습니다.
SQL Server Always On 기본 복제본 문제	AWS DMS 이제 Microsoft SQL Server Always On 가용성 그룹에서 기본 복제본을 올바르게 식별하여 이전 대/소문자 구분 감지 오류를 해결하는 문제를 수정했습니다.
PostgreSQL 대상 숫자 데이터 처리 문제	무한 숫자 데이터 형식을 PostgreSQL 대상 엔드포인트로 마이그레이션할 때 AWS DMS 작업이 실패하는 문제를 수정했습니다.

해결된 문제	설명
데이터 검증 문자 처리 문제	AWS DMS 이제는 마이그레이션 중에 CHAR 및 VARCHAR 데이터를 올바르게 검증하여 검증 작업에서 거짓 긍정 보고서를 제거합니다.
Amazon Redshift LOB 병렬 적용 문제	AWS DMS Amazon Redshift를 대상 엔드포인트로 사용할 때 병렬 배치 작업 중에 대형 객체(LOB) 데이터의 데이터 손상을 방지합니다.
열 필터 문제가 있는 데이터 검증	VARCHAR, CHAR, DATE 및 DATETIME 데이터 유형에 소스 열 필터를 적용하면 검증 실패가 방지되고 Amazon S3 대상에서 쿼리가 성공적으로 실행됩니다.
S3 검증 데이터 지속성 문제	Amazon S3 대상을 사용하는 경우 변경 데이터 캡처(CDC) 작업 전체에서 일관된 데이터 검증 상태가 유지됩니다.
문자열 형식 확인 문제	레코드 비교 작업 중 문자열 형식 오류로 인해 데이터 검증이 실패하는 문제를 수정했습니다.
DB2 빈 문자열 CDC 마이그레이션 문제	변경 데이터 캡처(CDC) 작업 중에 IBM DB2 LUW 소스 데이터베이스의 빈 VARCHAR 값이 빈 문자열 대신 NULL 값으로 잘못 캡처되는 문제를 해결했습니다.
Oracle LOB 조회 메모리 누수 문제	전체 LOB 모드가 활성화된 경우 LOB 조회 중에 Oracle 소스 엔드포인트의 메모리 누수가 수정되었습니다. 이렇게 하면 LOB 데이터를 사용한 복제 작업 중에 메모리가 지속적으로 증가하고 out-of-memory 오류가 발생하는 것을 방지할 수 있습니다.
S3 교차 계정 버킷 액세스 검증 문제	S3를 마이그레이션 대상 또는 소스로 사용할 때 보안을 개선하기 위해 S3 버킷 소유권 검증을 개선했습니다.
데이터 검증 메모리 누수 문제	데이터 검증에서 메모리 누수를 수정했습니다. 이렇게 하면 장기 실행 검증 작업 중에 out-of-memory 오류가 방지됩니다.

AWS Database Migration Service 3.5.2 릴리스 정보

AWS DMS 3.5.2의 새로운 기능

새로운 기능 또는 향상된 기능	설명
Amazon Redshift 데이터 검증	AWS DMS 는 이제 Amazon Redshift 대상의 데이터 검증을 지원합니다.
Microsoft SQL Server 버전 2022를 소스와 대상으로 지원.	AWS DMS 는 이제 Microsoft SQL Server 버전 2022를 소스 및 대상으로 사용할 수 있도록 지원합니다.
IBM DB2 LUW를 대상으로 사용	AWS DMS 는 이제 IBM Db2 LUW를 대상으로 지원합니다. 이제 AWS DMS를 사용하여 IBM Db2 LUW에서 IBM Db2 LUW로 라이브 마이그레이션을 수행할 수 있습니다.

AWS DMS 버전 3.5.2에는 다음과 같은 해결된 문제가 포함되어 있습니다.

2024년 4월 29일에 발표된 DMS 3.5.2 유지 관리 릴리스에서 해결된 문제

해결된 문제	설명
IBM DB2 대상 세그먼트화된 전체 로드	IBM DB2를 대상으로 하는 세그먼트화된 전체 로드에 대한 지원이 추가되었습니다.
Amazon Timestream을 대상으로 사용 설정	Timestream을 대상으로 사용할 때 유효하지 않은 타임스탬프 설정 및 지원되지 않는 테이블 작업의 처리를 개선했습니다.
열 필터를 사용한 태스크 충돌	DMS가 변환 규칙을 사용하여 동적으로 추가한 열에서 필터를 사용하는 동안 태스크가 충돌하는 문제를 수정했습니다.
트랜잭션 스왑 파일 읽기 로깅	DMS가 트랜잭션 스왑 파일에서 읽는 시기를 보여주는 로깅을 추가했습니다.
CdcInsertsAndUpdates 를 사용한 대상으로서 S3	CdcInsertsAndUpdates 가 true이고 PreserveTransactions 이 true인 경우 태스크가 충돌하는 대상으로서 S3에 대한 문제를 수정했습니다.

해결된 문제	설명
소스 필터 음수 연산자	동일한 열에 변환 규칙이 정의된 경우 소스 필터 연산자가 음수 연산자로 설정된 경우 잘못된 동작이 발생하는 문제를 수정했습니다.
DMS가 소스에서 읽기를 일시 중지할 경우의 로깅 추가	DMS가 성능을 개선하기 위해 소스에서 읽기를 일시적으로 일시 중지하는 경우를 보여주는 로깅이 향상되었습니다.
이스케이프된 문자가 있는 소스 필터	DMS가 CDC 중에 새로 생성된 테이블에 이스케이프된 문자를 적용하는 소스 필터 문제를 수정했습니다.
대상으로서 PostgreSQL, 잘못 복제된 삭제	DMS 복제가 null 값으로 삭제되는 대상으로서 PostgreSQL에 대한 문제를 수정했습니다.
소스로서 Oracle 로깅 개선 사항	외부 오류 코드를 제거하기 위한 소스로서 Oracle에 대한 로깅이 향상되었습니다.
XMLTYPE 제한 사항의 로깅 개선	XMLTYPE 데이터 유형의 전체 LOB 모드에 대한 DMS의 지원 부족을 보여주기 위해 소스로서 Oracle에 대한 로깅을 개선했습니다.
MySQL 데이터 손실	손상된 열 메타데이터로 인해 태스크 충돌 또는 데이터 손실이 발생할 수 있는 대상으로서 MySQL 문제를 수정했습니다.
새 열에 적용된 필터	전체 로드 중에 변환 규칙이 새 열에 추가하는 필터를 DMS가 무시하는 문제를 수정했습니다.
대상으로서 S3: 검증 문제	검증 파티셔닝 정의가 다른 여러 테이블을 마이그레이션하는 동안 데이터 검증이 실패하는 대상으로서 S3의 문제를 수정했습니다.
CDC 전용 태스크 충돌	TaskRecoveryTableEnabled 가 true일 때 태스크가 충돌하는 CDC 전용 태스크의 문제를 해결했습니다.
MySQL에서 MariaDB로 호환되지 않는 데이터 정렬	DMS가 tf8mb4_0900_ai_ci 데이터 정렬로 MySQL v8 테이블을 마이그레이션하지 않는 MySQL에서 MariaDB로 마이그레이션하는 문제를 수정했습니다.
BatchApplyEnabled 로 인해 태스크 충돌	특정 조건에서 작업이 실패하는 배치 적용 기능의 문제를 수정했습니다.

해결된 문제	설명
Amazon DocumentDB의 비 UTF-8 문자	Amazon DocumentDB 엔드포인트의 비 UTF-8 문자에 대한 지원이 추가되었습니다.
배치 적용 태스크 충돌	대량 트랜잭션을 복제하는 동안 DMS 태스크가 충돌하는 배치 적용 기능에 대한 문제를 수정했습니다.
Db2 트랜잭션 롤백 처리	소스에서 롤백됨에도 불구하고 DMS가 대상에 INSERT를 복제하는 소스로서 Db2에 대한 문제를 수정했습니다.
소스 필터를 사용한 검증	검증이 소스 필터를 준수하지 않는 문제를 해결했습니다.
MongoDB AWS DMS 스왑 파일 문제	AWS DMS 스왑 파일이 있을 때 AWS DMS 업그레이드 후 DMS 작업이 재개되지 않는 MongoDB 소스 문제를 수정했습니다.
MySQL에서 Amazon Redshift JSON으로 마이그레이션 문제.	배치 적용이 활성화된 상태에서 JSON 데이터 형식이 고립되지 않는 MySQL 소스 문제를 수정했습니다.
MySQL 5.5 소스 CDC 오류	MySQL 소스에서 CDC 복제가 MySQL 버전 5.5에서 실패하던 문제를 수정했습니다.
LOB 마이그레이션 문제	특정 이벤트 유형을 처리하는 동안 AWS DMS 작업이 충돌하는 LOB 마이그레이션 문제를 수정했습니다.
데이터 유효성 검사 문제	특정 DDL 이벤트에서 검증 전용 작업이 중단되는 데이터 검증 기능의 문제를 수정했습니다.
필터 관련 데이터 검증 문제입니다.	필터가 있을 때 HandleCollationDiff 설정이 적용되지 않는 데이터 검증 기능의 문제를 수정했습니다.
MySQL SourceEnum 인코딩 문제.	MySQL 소스에서 UTF-16 인코딩 열거형 값이 올바르게 마이그레이션되지 않는 문제를 수정했습니다.
SQL Server 소스를 사용한 LOB 잘림 경고	SQL Server 소스에 대한 로깅을 강화하여 CDC 수행 중 LOB 잘림 발생 시 경고를 표시하도록 개선했습니다.

해결된 문제	설명
SQL Server에서 APG로의 마이그레이션에 대한 데이터 검증 문제입니다.	SQL Server에서 PostgreSQL로 마이그레이션할 때 데이터 유효성이 특정 상황에서 거짓 긍정을 보고하는 문제를 수정했습니다.
Amazon Redshift 대상 CDC에 대한 EmptyAsNull ECA 문제	Amazon Redshift 대상에서 EmptyAsNull ECA가 제대로 작동하지 않는 문제를 수정했습니다.
SQL Server 소스에 대한 로깅 개선 사항	스토리지 단위 값을 제외하지 않도록 SQL Server 소스에 대한 로깅을 개선했습니다.
메모리 누수 문제	CSV 파일을 사용하여 데이터 AWS DMS 작업을 로드하는 대상에서 메모리 누수가 표시되는 문제를 수정했습니다.
Amazon S3 대상 문제	cdcInsertAndUpdateOnly 설정이 활성화되었을 때 CdcMaxBatchInterval 및 CdcMinFileSize 가 준수되지 않는 Amazon S3 대상 문제를 수정했습니다.
Kafka 대상에 대한 로깅 개선 사항	Kafka 대상에 대한 향상된 로깅
Oracle 소스에 대한 로깅 개선 사항입니다.	프라이머리 키 누락으로 인해 테이블을 건너뛰고 있음을 올바르게 나타내기 위해 이진 리더를 사용하는 Oracle 소스에 대한 로깅을 개선했습니다.
SQL Server 소스에 대한 로깅 개선 사항	누락된 권한을 올바르게 나타내도록 AlwaysOn 구성에서 SQL Server 소스에 대한 로깅을 개선했습니다.
DDL이 비활성화된 마이그레이션에 대한 로깅 개선 사항	외부에서 수정된 후 예기치 않은 대상 테이블 구조를 나타내도록 DDL 복제가 비활성화된 마이그레이션에 대한 로깅을 개선했습니다 AWS DMS.
Db2 대상 문제	DMS 상태 테이블이 활성화된 경우 작업이 실패하는 Db2 대상 문제를 수정했습니다.

해결된 문제	설명
MongoDB/Amazon DocumentDB Secret Manger 문제입니다.	Secret Manager에서 자격 증명을 검색할 수 없어 오류가 발생한 MongoDB/Amazon DocumentDB 엔드포인트 문제를 수정했습니다.
MongoDB/Amazon DocumentDB 엔드포인트 문제	특정 이벤트 시퀀스를 복제하는 동안 ParallelApply가 활성화된 상태에서 작업이 실패하는 MongoDB/Amazon DocumentDB 문제를 수정했습니다.
MongoDB/Amazon DocumentDB 데이터 검증 문제	MongoDB/Amazon DocumentDB 데이터 검증에서 데이터 차이를 감지한 후 오류가 발생하는 문제를 수정했습니다.
Amazon Redshift 대상에 대한 로깅 개선 사항	기본 로깅 수준에 자세한 정보를 포함하도록 Amazon Redshift 대상에 대한 로깅을 개선했습니다.
Amazon S3 대상 작업 충돌 문제	가 GlueCatalogGeneration 활성화되면 변경 테이블 DDL을 수신한 후 AWS DMS 작업이 충돌하는 Amazon S3 대상에 대한 문제를 Fixed.
데이터 유효성 검사 문제	NUL (0x00) 문자에 대한 검증이 실패하는 데이터 검증 기능의 문제를 수정했습니다.
Babelsifh 엔드포인트 문제	대소문자가 혼합된 테이블 이름이 일시 중지되는 babelfish 엔드포인트 문제를 수정했습니다.
선택 규칙과 관련된 Db2 LUC 소스 문제입니다.	선택 규칙의 "table-type" 옵션이 무시되는 Db2 LUW 소스 문제를 수정했습니다.
Amazon Redshift 대상 데이터 누락 문제.	Amazon Redshift 대상에서 특정 조건에서 ParallelLoadThreads 가 >0일 때 데이터 손실이 발생하는 문제를 수정했습니다.
Amazon Redshift 대상에 대한 데이터 검증 개선 사항	HandleCollationDiff 설정을 지원하도록 Amazon Redshift 대상에 대한 데이터 검증 기능을 개선했습니다.
Amazon S3 대상 데이터 검증 문제	테이블에 PK 이외의 열이 없을 때 검증이 실패하는 Amazon S3 대상 데이터 검증 문제를 수정했습니다.

해결된 문제	설명
데이터 검증 CloudWatch 지표	검증을 위해 CloudWatch 지표가 누락되어 완료하는 데 짧은 시간이 걸리는 데이터 검증 기능의 문제를 수정했습니다.
데이터 검증 재검증 옵션 사용 불가	특정 상황에서 재검증 옵션을 사용할 수 없는 데이터 검증 기능의 문제를 수정했습니다.
대규모 Transaction 복제 문제	특정 조건에서 트랜잭션당 최대 이벤트 수가 201,326,592개로 제한되는 문제를 수정했습니다.
테이블 다시 로드 문제	테이블 중 하나 이상이 유효하지 않을 때 여러 테이블의 재로드가 취소되는 문제를 수정했습니다.
MySQL 소스 데이터 손실 문제	MySQL에서 S3로 마이그레이션할 때 "열 추가" DDL 이후에 실행된 첫 번째 DML이 누락되어 데이터가 손실되는 문제를 수정했습니다.
일괄 메모리 누수 문제 적용	특정 조건에서 발생하는 배치 적용 기능의 메모리 누수 문제를 수정했습니다.
AWS DMS 작업 시작 문제	AWS DMS 태스크 시작이 매우 오래 걸리고 완료되지 않는 문제를 수정했습니다.
PostgreSQL Source 누락된 데이터 문제	복제 슬롯의 알 수 없는 이벤트로 인해 데이터 손실이 발생하는 PostgreSQL 소스 문제를 수정했습니다.
MySQL 소스 및 대상 LOB 열 순서	MySQL 소스 및 대상 마이그레이션 LOB 열 관련 문제를 수정했습니다. 이제 DMS는 LOB 데이터를 작성해야 하는 열을 결정할 때 소스 테이블 대신 대상 테이블의 열 ID를 사용합니다.
진행 중인 복제(CDC) 중에 병로그를 읽는 동안 MySQL 5.5 작업이 실패합니다.	MySQL 5.5 소스 관련 문제를 수정하고, DMS가 지속적 복제(CDC) 중에 이진 로그 이벤트를 읽지 못할 때 작업 실패를 방지하는 재시도 메커니즘을 추가했습니다.
테스트 디코딩 플러그인을 사용하는 동안 PostgreSQL 소스 Segfault 및 메모리 누수 문제.	Postgres용 test_decoding 플러그인을 사용할 때 특정 온징 복제(CDC) 이벤트를 올바르게 구문 분석하지 못하는 PostgreSQL 소스 문제를 수정했습니다.

해결된 문제	설명
DocumentDB 대상 병렬 적용 설정	이 기능을 사용하는 동안 여러 스레드를 사용하지 못하게 하는 병렬 적용 설정이 있는 DocumentDB 대상 문제를 수정했습니다.
Oracle HCC 압축 DIRECT INSERT	누락 및 중복 데이터를 유발하는 병렬 DML 힌트가 있는 Oracle HCC 압축 DIRECT INSERT 문제를 수정했습니다.
Oracle 2024년 7월 CPU 관련 Oracle Binary Reader 문제	Oracle 소스 관련 문제를 수정했습니다. Oracle 2024년 7월 CPU로 인해 바이너리 리더를 사용하는 DMS 작업이 실패했습니다.
TaskRecoveryTableEnabled로 실패한 DMS ongoing 복제(CDC) 작업	대상 연결이 종료된 후 DMS가 대상 시스템 테이블 awsdms_txn_state 를 업데이트하려고 시도하는 TaskRecoveryTableEnabled 이 활성화된 문제를 수정했습니다.
PostgreSQL 소스 중복 레코드	TaskrecoveryTableEnabled 설정이 활성화된 경우 일부 트랜잭션이 두 번 복제되는 PostgreSQL 소스 문제를 수정했습니다.
DMS S3 소스/대상 복제 문제	S3 소스에서 S3 대상으로의 문제에서 DMS 작업이 전체 로드 중 및 복제 진행 중 데이터를 복제하지 않는 문제를 수정했습니다.
DMS DB2 소스 CcsidMapping ECA	CcsidMapping에서 DB2 source 관련 문제를 수정했습니다. 이제 코드 페이지가 0이고 데이터가 제대로 마이그레이션되면 CCSID mappin ECA가 작업에 올바르게 적용됩니다.
DMS Aurora PostgreSQL에서 Redshift Serverless 부울 값 복제.	Aurora PostgreSQL에서 Redshift Serverless로의 DMS 마이그레이션에서 부울 값에 문제가 발생하는 문제를 수정했습니다.
MAZ와 관련된 PostgreSQL 소스 문제	이제 PostgreSQL은 소스 엔드포인트로서 다중 AZ 장애 조치 이벤트 중에 중단 없는 복제를 유지하여 작업 실패를 방지합니다.
MySQL 소스 열 복제 문제	소스 복제로서의 MySQL은 테이블 중간 중복 열 추가를 처리하여 예기치 않은 작업 중단을 방지합니다.
MySQL 소스 열 수정 문제	소스 복제로서의 MySQL은 CDC 작업 중에 여러 열이 추가될 때 열 시퀀스 무결성을 유지합니다.

해결된 문제	설명
PostgreSQL 부울 데이터 검증 문제	PostgreSQL은 이제 소스 데이터 검증으로서 부울 데이터 형식 매핑을 올바르게 해석하여 정확한 비교 결과를 생성합니다.
Oracle 확장 VARCHAR2 데이터 잘림 문제	확장 데이터 유형 지원을 사용할 때 VARCHAR2(4000) 열에서 후행 공백이 잘리는 소스로서의 Oracle 문제를 수정했습니다.
SQL Server 보조 복제본 DDL 처리 문제	SQL Server 소스 복제는 보조 복제본에서 DDL 작업 중에 연결을 유지하여 작업 종단을 방지합니다.
Secret Manager 특수 문자 처리 문제	AWS 이제 Secrets Manager 연결 문자열은 보안 프로토콜을 유지하면서 특수 문자를 지원합니다.
MongoDB/Amazon DocumentDB 중복 키 문제	MongoDB 및 Amazon DocumentDB 복제는 이전에 키 제약 조건 오류를 트리거한 레코드 중복을 방지합니다.
Oracle 타임스탬프 처리 문제	Oracle 소스 복제는 이제 다양한 세션 시간대 구성에서 타임스탬프 값을 정확하게 처리합니다.
Oracle 데이터 변환 문제	Oracle 소스로 인해 데이터 유형 변환 중에 ORA-01460 오류와 함께 AWS DMS 작업이 실패했습니다. 이 업데이트는 작업 실패를 방지하기 위해 데이터 유형 변환 처리를 개선합니다.

AWS Database Migration Service 3.5.1 릴리스 정보

스트리밍 대상에 대한 큰 숫자 값 처리 변경

AWS DMS 버전 3.5.1에서는 Kafka 및 Kinesis와 같은 대상으로 데이터를 스트리밍할 때 정수 및 고정밀 숫자 값이 처리되는 방식이 변경되었습니다. 특히 내부 데이터 형식 표현을 AWS DMS 변경하여 이러한 값을 INT64 대신 로 처리했습니다. 이러한 전환으로 인해 특히 값이 INT8의 제한을 초과하는 경우 스트리밍 엔드포인트에서 데이터 형식이 다를 수 있습니다. 따라서 이러한 숫자 유형의 표현은 Kafka 및 Kinesis와 같은 대상으로 스트리밍할 때 이전 형식과 다를 수 있으며, 이는 이러한 대상의 데이터를 사용하는 다운스트림 시스템 및 프로세스에 영향을 미칠 수 있습니다.

변경 사항 요약:

- 이전 버전(예: 3.4.7/3.4.6)에서는 큰 정수 값이 정수로 표시되었습니다.

- 버전 3.5.1부터 이러한 값은 과학적 표기법(예: 7.88129934789981E15)에 나타날 수 있으며, 이로 인해 정밀도와 형식 차이가 발생할 수 있습니다.

영향을 받는 데이터 유형

최근 변경 사항은 Kafka 및 Kinesis와 같은 엔드포인트로 스트리밍될 때 여러 숫자 유형의 표현에 영향을 미칩니다. 영향을 받는 유형은 다음과 같습니다.

- 큰 정수 유형(예: bigint)
- 부동 소수점 형식(FLOAT, DOUBLE)
- 고정밀 십진수 유형(DECIMAL, NUMERIC)

영향을 받는 시나리오:

- 스트리밍 대상으로의 전체 로드 마이그레이션
- 데이터 캡처(CDC)를 스트리밍 대상으로 변경

이 변경은 특히 Kafka 및 Kinesis와 같은 스트리밍 엔드포인트에 영향을 미치는 반면, 비스트리밍 대상은 영향을 받지 않습니다.

이 변경을 완화하기 위해 큰 숫자를 정수로 나타내는 이전 형식으로 되돌리는 데이터 형식 변환을 구현할 수 있습니다. 그러나 이 해결 방법은 제한 또는 호환성 문제가 발생할 수 있으므로 모든 시나리오에 적합하지 않을 수 있다는 점에 유의해야 합니다.

권장 사항:

- AWS DMS 버전 3.5.1 이상을 배포하기 전에 비프로덕션 환경에서 특정 사용 사례를 테스트하여 이 변경의 영향을 식별하고 해결합니다.
- 영향을 받는 고객은 해당하는 경우 change-data-type 변환 해결 방법을 구현하여 큰 숫자를 정수로 하여 이전 형식으로 되돌릴 수 있습니다. 그러나 이 접근 방식이 모든 시나리오에 적합하지는 않을 수 있습니다.

향후 릴리스에서 엔드포인트 간에 일관된 데이터 유형 처리를 보장하기 위해 이 동작을 검토하고 있습니다.

다음 표에서는 AWS Database Migration Service (AWS DMS) 버전 3.5.1에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
PostgreSQL 15.x 지원	AWS DMS 버전 3.5.1은 PostgreSQL 버전 15.x를 지원합니다. 자세한 내용은 PostgreSQL을 소스로 사용 및 PostgreSQL을 대상으로 사용 섹션을 참조하세요.
Amazon DocumentDB Elastic Clusters(샤딩된 컬렉션 포함) 지원	AWS DMS 버전 3.5.1은 샤딩된 컬렉션이 있는 Amazon DocumentDB Elastic Clusters를 지원합니다. 자세한 내용은 Amazon DocumentDB를 AWS Database Migration Service의 대상으로 사용 단원을 참조하십시오.
Amazon Redshift Serverless 대상	대상 엔드포인트로 Amazon Redshift Serverless를 사용하는 기능 지원. 자세한 내용은 Amazon Redshift 데이터베이스를의 대상으로 사용 AWS Database Migration Service 섹션을 참조하세요.
Babelfish 엔드포인트 설정	Babelfish 지원을 제공하기 위한 PostgreSQL 대상 엔드포인트 설정을 개선했습니다. 자세한 내용은 PostgreSQL 데이터베이스를의 대상으로 사용 AWS Database Migration Service 섹션을 참조하세요.
Oracle 소스 미결 트랜잭션	AWS DMS 3.5.1은 Oracle 소스의 시작 위치에서 CDC 전용 작업을 시작할 때 미결 트랜잭션을 처리하는 방법을 개선합니다. 자세한 내용은 Oracle을의 소스로 사용할 때 엔드포인트 설정 AWS DMS 섹션의 OpenTransactionWindow 섹션을 참조하세요.
대상으로서 Amazon Timestream	Amazon Timestream을 대상 엔드포인트로 사용 지원. 자세한 내용은 Amazon Timestream을의 대상으로 사용 AWS Database Migration Service 단원을 참조하십시오.

AWS DMS 버전 3.5.1에는 다음과 같은 해결된 문제가 포함되어 있습니다.

해결된 문제	설명
큰 숫자 값 처리 변경	스트리밍 대상의 큰 숫자 값 표현이 업데이트되었습니다. 잠재적 영향에 대한 자세한 내용은 '스트리밍 대상에서 큰 숫자 값 처리' 설명서를 검토하세요.

해결된 문제	설명
비활성 세션이 증가하는 소스로서 Oracle	CDC 전용 태스크에서 비활성 세션이 지속적으로 증가하여 ORA-00020: maximum number of processes exceeded on the source database 예외가 발생하는 Oracle 소스에 대한 문제를 수정했습니다.
DocumentDB에 업데이트 변경 사항 복제	일부 시나리오에서 UPDATE 문이 제대로 복제되지 않는 대상으로서 DocumentDB에 대한 문제를 수정했습니다.
검증 전용 태스크	검증 전용 태스크에 대해 데이터 검증이 비활성화된 경우 태스크를 제대로 실패하도록 하는 데이터 검증 기능의 오류 처리를 개선했습니다.
연결 종료 후 Amazon Redshift 복제	연결 종료 후 대상에 0보다 크도록 설정된 ParallelApplyThreads 가 있는 경우 DMS 태스크가 대상에 변경 사항을 다시 적용하지 않아 데이터가 손실되는 Amazon Redshift 대상 문제를 수정했습니다.
MySQL 텍스트에서 중간 텍스트로 복제	전체 LOB 모드를 사용한 중간 텍스트 데이터 유형의 MySQL에서 MySQL로의 복제 문제를 수정했습니다.
CDC 태스크가 교체된 보안 암호로 복제되지 않음	Secrets Manager가 암호를 교체한 후 DMS가 데이터 복제를 중지하는 BatchApplyEnabled 가 true로 설정된 DMS 태스크에 대한 문제를 수정했습니다.
MongoDB/DocumentDB 세분화 문제	프라이머리 키 열에 큰 값이 포함되어 있을 때 범위 분할이 제대로 작동하지 않는 MongoDB/DocDB 소스에 대한 문제를 수정했습니다.
비결합 숫자 값의 Oracle 데이터 검증	DMS가 데이터 검증 중에 비결합 데이터 유형의 값 NUMERIC을 STRING으로 인식하는 Oracle 대상에 대한 문제를 수정했습니다.
SQL Server 데이터 검증	DMS 데이터 검증이 잘못된 SQL 문을 구성한 SQL Server 엔드포인트에 대한 문제를 수정했습니다.
MongoDB 자동 세그먼트화	MongoDB 대상에서 문서를 병렬로 마이그레이션할 때 데이터 자동 파티셔닝 기능을 개선했습니다.

해결된 문제	설명
Amazon S3 Apache Parquet 형식	S3 대상에 쓴 Apache Parquet 파일을 Apache Arrow C++가 설치된 Python을 사용하여 볼 수 있도록 문제를 수정했습니다.
PostgreSQL 소스 DDL 처리	PostgreSQL 소스에서 지원되지 않는 DDL 작업이 올바르게 무시되지 않는 문제를 수정했습니다.
PostgreSQL timestamp tz 데이터 오류	PostgreSQL에서 PostgreSQL로 마이그레이션할 때 CDC 중에 일괄 적용이 활성화된 상태에서 시간대 데이터가 포함된 타임스탬프가 제대로 마이그레이션되지 않는 문제를 수정했습니다.
Oracle에서 PostgreSQL로의 유효성 검사 실패	Oracle에서 PostgreSQL로 마이그레이션할 때 NUMERIC(38,30) 데이터 형식에 대한 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
Oracle 확장 데이터 형식 오류	Oracle 소스에서 확장된 varchar 데이터 형식이 잘리는 문제를 수정했습니다.
필터 연산자 결합	열 필터링 기능에서 null 열 연산자를 다른 유형의 연산자와 결합할 수 없는 문제를 수정했습니다.
과도한 로깅으로 인한 CDC 지연.	PostgreSQL 소스에서 pglogical 플러그인 경고가 과도하게 로깅되어 소스 CDC 지연이 발생하는 문제를 수정했습니다.
Create Table DDL의 양방향 복제 처리	PostgreSQL에서 PostgreSQL로의 양방향 복제 시 Create Table DDL 변경 사항이 제대로 복제되지 않는 문제를 수정했습니다.
필터 사용 중 CDC 오류	필터링 기능에서 CDC 복제가 실패하는 문제를 수정했습니다.
Kafka 엔드포인트에 대한 인증 기관 호스트 이름 유효성 검사	인증 기관의 호스트 이름 유효성 검사를 비활성화하는 옵션 (SslEndpointIdentificationAlgorithm)을 추가하여 Kafka 엔드포인트의 기능을 개선했습니다.
IBM DB2 LUW 유효성 검사	데이터 유효성 검사 중에 Db2 LUW 소스 날짜, 타임스탬프 및 시간 데이터 형식이 제대로 처리되지 않는 문제를 수정했습니다.

해결된 문제	설명
S3 유효성 검사	Db2 LUW에서 S3로의 마이그레이션 시 유효성 검사 기능이 timestamp(0) 데이터 유식을 제대로 처리하지 못하는 문제를 수정했습니다.
DMS 태스크 재시작 실패	PostgreSQL 소스에서 AWS DMS 작업이 다시 시작되지 않아 pglogical 플러그인을 사용할 때 관계형 이벤트를 사용할 수 없는 문제를 수정했습니다.
HIERARCHY 데이터 형식에 대한 SQL Server 유효성 검사	SQL Server 소스에서 HIERARCHY 데이터 형식 유효성 검사가 실패하는 문제를 수정했습니다.
제어 문자가 포함된 SQL Server 문자열	SQL Server 소스에서 제어 문자가 포함된 문자열이 제대로 복제되지 않는 문제를 수정했습니다.
Amazon Redshift 및 Secrets Manager	Amazon Redshift 대상에서 Secrets Manager를 사용할 때 엔드포인트 테스트가 실패하는 문제를 수정했습니다.
MySQL ParallelLoadThreads 설정 불일치	MySQL 대상에서 태스크 설정이 변경된 후 ParallelLoadThreads 설정이 제대로 유지되지 않는 문제를 수정했습니다.
PostgreSQL에서 Oracle로 데이터 형식 매핑 중 오류	PostgreSQL에서 Oracle로의 마이그레이션에서 데이터 형식 TEXT에서 데이터 형식 VARCHAR2(2000)로 복제할 때 태스크가 실패하는 문제를 수정했습니다.
Oracle에서 PostgreSQL로의 데이터 유효성 검사	Oracle에서 PostgreSQL로의 마이그레이션에서 NULL 문자가 공백 문자로 복제될 때 데이터 유효성 검사가 거짓 공정을 보고하는 문제를 수정했습니다.
AlwaysOn 구성의 SQL Server 소스	복제본 이름이 실제 서버 이름과 정확히 일치하지 않을 때 AWS DMS 작업이 실패하는 AlwaysOn 구성의 SQL Server 소스 문제를 수정했습니다.
Oracle 소스 엔드포인트 테스트 실패	Oracle 세션 ID(SID)를 검색하는 동안 권한이 부족하여 AWS DMS 엔드포인트 연결 테스트가 실패하는 Oracle 소스 문제를 수정했습니다.

해결된 문제	설명
CDC가 새 테이블을 픽업하지 않음	CDC 전용 태스크에서 태스크가 시작된 후 소스에서 생성된 테이블이 일부 경우에서 복제되지 않는 문제를 수정했습니다.
Oracle을 소스로 사용 시 미결 트랜잭션	Oracle 소스의 시작 위치에서 CDC 전용 태스크를 시작할 때 미결 트랜잭션을 처리하는 방법을 개선했습니다.
누락 데이터 문제	캐시된 변경 사항을 적용한 후(<code>StopTaskCachedChangesApplied</code> 옵션이 <code>true</code> 로 설정됨) 태스크를 중지하면 태스크를 재개할 때 데이터가 누락되는 문제를 수정했습니다. 이 문제는 소스의 많은 변경 사항으로 인해서 AWS DMS 복제 인스턴스 디스크에 캐시된 변경 사항을 유지하는 경우 AWS DMS 거의 발생하지 않을 수 있습니다.
확장 데이터 형식에 대한 데이터 유효성 검사 문제	PostgreSQL에서 Oracle로의 데이터 유효성 검사에서 확장된 데이터 형식에 대한 유효성 검사가 실패하는 문제를 수정했습니다.
일치하지 않은 문자 인코딩에 대한 데이터 유효성 검사 문제	소스와 대상 간에 문자 인코딩이 일치하지 않을 때 SQL Server에서 PostgreSQL로의 데이터 유효성 검사에 실패하는 문제를 수정했습니다.
데이터 유효성 검사 문제 ORA-01455	PostgreSQL <code>integer</code> 이 Oracle <code>number(10)</code> 에 매핑될 때 유효성 검사 도중 ORA-01455 오류가 발생하는 문제를 수정했습니다.
SQL Server IDENTITY 지원	SQL Server에서 SQL Server로의 데이터 복제에서 대상 열에 IDENTITY 속성이 있는 경우 ID 열 마이그레이션이 실패하는 문제를 수정했습니다.
ALTER 문의 문자 세트 문제	CDC 중에 ALTER 문을 마이그레이션할 때가 문자 집합을 UTF16으로 AWS DMS 변경하는 MySQL에서 MySQL로의 복제 문제를 수정했습니다.
PostgreSQL에서 Amazon Redshift로의 공간 데이터 형식 지원	PostgreSQL에서 Amazon Redshift로 마이그레이션할 때 <code>spatial</code> 데이터 형식에 대한 지원을 추가했습니다.
.parquet 파일의 GZIP 압축	가 S3를 대상으로 하는 GZIP 압축을 사용하여 .parquet 파일을 생성 AWS DMS 하지 못하는 문제를 수정했습니다.

해결된 문제	설명
MongoDB/DocDB 소스 마이그레이션	AWS DMS 가 MongoDB 소스에서 일부 파티션을 마이그레이션하지 않는 문제를 수정했습니다.
테이블 통계 문제	복제 인스턴스의 태스크 중 하나 이상에 1,001개가 넘는 테이블이 포함된 경우 테이블 통계가 표시되지 않는 문제를 수정했습니다.
IBM DB2 LUW 버전 10.1.0 이하에서 테이블 일시 중지	소스 데이터베이스 버전이 10.1.0 이하일 때 TYPESTRINGUNITS is not valid 오류와 함께 테이블 마이그레이션이 일시 중단되는 Db2 LUW 소스 문제를 수정했습니다.
MongoDB 파티셔닝 문제	MongoDB/DocDB에서 소스 파티션의 세그먼트가 하나 이상 누락되는 문제를 수정했습니다.
MongoDB 파티셔닝 문제	NumberLong() 형식의 열을 기반으로 하는 세그먼트화가 형식 변환 버그로 인해 실패하는 문제를 수정했습니다.
MongoDB 파티셔닝 문제	MongoDB를 소스로 사용할 때 대규모 데이터세트에 대한 자동 세그먼트화 성능을 개선했습니다.
MongoDB 드라이버 버전	MongoDB 버전 3.6 이하를 계속 지원하기 위해 MongoDB 드라이버를 1.20.0으로 다운그레이드했습니다.
Amazon S3 Apache Parquet 타임스탬프 데이터 형식	Amazon S3 parquet target. AWS DMS now에서 이전 버전의 동작과 일치하도록 형식 파라미터를 isAdjustedToUTC true로 설정하는 문제를 수정했습니다 AWS DMS.
Amazon Redshift 대상 복사 명령	Amazon Redshift를 대상으로 사용할 때 Amazon S3에서 Amazon Redshift로 데이터를 복사하는 경우 대용량 테이블에 대해 복사 명령이 실패하는 문제를 수정했습니다.
PostgreSQL 지오메트리 데이터 형식	PostgreSQL에서 PostgreSQL로의 마이그레이션에서 대규모 지오메트리 데이터 형식에서 마이그레이션이 실패하는 문제를 수정했습니다.
Oracle에서 PostgreSQL로의 XML	Oracle에서 PostgreSQL로 복제할 때 마이그레이션으로 인해 XML에 공백 하나가 추가되는 문제를 해결했습니다.

해결된 문제	설명
지원되는 엔진에서 대상 체크포인트 업데이트	AWS DMS 는 이제 대상 데이터베이스의 <code>awsdms_txn_state</code> 테이블에서 대상 체크포인트를 업데이트합니다.
MongoDB/DocDB 레코드가 잘못된 컬렉션으로 전송됨	MongoDB/DocDB에서 데이터가 잘못된 대상 컬렉션으로 전송되는 문제를 수정했습니다.
EscapeCharacter 엔드포인트 설정을 사용한 Oracle 소스 새 테이블 선택	EscapeCharacter 엔드포인트 설정이 설정된 동안 작업이 중지되고 재개될 때 AWS DMS 가 복제를 위해 새 테이블만 선택하는 Oracle 소스 문제를 수정했습니다.
CDC 복구 체크포인트	대상 데이터 저장소와 AWS DMS 콘솔 간에 관찰된 CDC 복구 체크포인트의 불일치를 수정했습니다.
CDC 검증 전용 작업	작업의 모든 테이블에 오류가 발생해도 작업이 실패하지 않는 CDC 검증 전용 작업 관련 문제를 해결했습니다.
소스 또는 대상 연결 문제의 유효성 검사 동작	연결이 끊 AWS DMS 어질 때가 소스 또는 대상의 테이블을 일시 중지하는 데이터 검증 문제를 수정했습니다.
Oracle에서 PostgreSQL로의 데이터 유효성 검사 거짓 긍정	가 거짓 긍정을 AWS DMS 보고한 Oracle에서 PostgreSQL로의 데이터 검증 문제를 수정했습니다. 이 문제는 대상의 소스 NULL 문자 표현의 차이가 VARCHAR 이외의 텍스트 기반 데이터 형식에서 반영되지 않기 때문에 발생합니다.
Oracle에서 PostgreSQL로 마이그레이션 시 데이터 잘림	Oracle을 소스로, PostgreSQL을 대상으로 사용할 때 Oracle NLS_NCHAR_CHARACTERSET 설정이 AL16UTF16 으로 설정된 상태에서 AWS DMS 가 NVARCHAR 열의 데이터를 자르는 문제를 수정했습니다.
데이터 유효성 검사 오류	소스 필터링과 열 추가 변환 규칙을 모두 사용할 때 <code>unable to create where filter clause</code> 오류가 발생하는 데이터 유효성 검사 문제를 수정했습니다.
Amazon Redshift 대상 오류 처리	Amazon Redshift를 대상으로 사용할 때 CDC 태스크에서 <code>ParallelApplyThreads</code> 태스크 설정이 0보다 큰 값으로 설정되면 오류 처리가 구성된 대로 작동하지 않는 문제를 수정했습니다.

해결된 문제	설명
Oracle 소스 통신 실패	Oracle을 소스로 사용할 때 태스크가 RUNNING 상태로 유지되지만 통신 실패 후 데이터를 마이그레이션할 수 없는 문제를 수정했습니다.
열 필터로 인한 CDC 테이블 일시 중단	열 필터가 적용된 CDC 단계에서 테이블이 일시 중단되는 전체 로드 + CDC 태스크 관련 문제를 수정했습니다.
S3 대상의 특수 문자에 대한 데이터 유효성 검사 실패	테이블 이름에 밑줄 이외의 특수 문자가 포함된 경우 태스크가 실패하는 S3 대상 데이터 유효성 검사 문제를 수정했습니다.
MongoDB 소스 전체 로드 및 CDC 실패	MongoDB를 소스로 사용할 때 대규모 컬렉션을 마이그레이션하면서 캐시 이벤트를 처리하는 동안 전체 로드 + CDC 태스크가 실패하는 문제를 수정했습니다.
BatchApplyEnabled 를 true로 설정 시 업그레이드 문제	경우에 따라 작업 설정이 true로 설정된 BatchApplyEnabled 작업이 AWS DMS 버전 3.4.6에서 3.5.1로 마이그레이션된 후 실패하는 문제를 수정했습니다.
SQL Server AlwaysOn 소스의 대/소문자 구분 데이터 정렬	SQL Server AlwaysOn을 소스로 사용할 때 대/소문자를 구분하는 데이터 정렬이 실패하는 문제를 수정했습니다.
MySQL 소스 태스크 중단	MySQL을 소스로 사용할 때 소스가 제대로 구성되지 않으면 태스크가 실패하는 대신 중단되는 문제를 수정했습니다.
S3 소스 전체 로드 태스크 실패	S3를 소스로 사용할 때 AWS DMS 버전 3.4.6 또는 3.4.7에서 버전 3.5.1로 업그레이드한 후 재개 시 작업이 실패하는 문제를 수정했습니다.
CaptureDDLs가 false로 설정된 PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 CaptureDDLs 엔드포인트 설정이 false로 설정되면 DDL이 제대로 처리되지 않는 문제를 수정했습니다.
재개 시 Oracle 소스 태스크 충돌	Oracle을 소스로 사용할 때 재개 시 열 이름의 잘못된 데이터로 인해 태스크가 중단되는 문제를 수정했습니다.

해결된 문제	설명
MySQL 소스 LOB 조회 실패	MySQL을 소스로 사용할 때 <code>ParallelApplyThreads</code> 태스크 설정이 0보다 큰 값으로 설정되면 LOB 조회가 실패하는 문제를 수정했습니다.
SQL Server 소스 비논리적 LSN 오류	SQL Server를 소스로 사용할 때 AWS DMS 버전 3.4.7에서 버전 3.5.1로 업그레이드한 후 <code>illogical LSN sequencing state error</code> 오류가 발생하여 작업이 실패하는 문제를 수정했습니다.
pglogical을 사용하는 PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 태스크를 중지하고, 테이블을 선택 규칙에서 제거하고, 태스크를 재개하고, 제거된 테이블을 변경하면 pglogical 플러그인을 사용하는 태스크가 실패하는 문제를 수정했습니다.
Aurora MySQL의 잘못된 복구 체크포인트.	Aurora MySQL을 소스로 사용할 때 Aurora 장애 조치 또는 Aurora 소스 중지 및 시작의 결과로 잘못된 복구 체크포인트가 저장되는 문제를 해결했습니다.
SQL Server를 소스로 사용할 시 작업 중단.	SQL Server를 소스로 사용할 때 <code>SafeguardPolicy</code> 가 <code>RELY_ON_SQL_SERVER_REPLICATION_AGENT</code> 로 설정되면 작업이 중단되는 문제를 수정했습니다.
MySQL을 대상으로 사용할 시 잘못된 데이터 유형 캐스팅	MySQL을 대상으로 사용할 때 배치 적용 단계에서 잘못된 데이터 유형 변환으로 인해 CDC 복제가 실패하는 문제를 해결했습니다.
PostgreSQL을 소스로 사용할 시 CaptureDDL이 false로 설정되면 작업 실패.	PostgreSQL을 소스로 사용할 때 <code>CaptureDDLs</code> 엔드포인트 설정이 <code>false</code> 로 설정된 경우 DDL이 DML로 처리되어 작업이 실패하는 문제를 해결했습니다.
MongoDB 빈 컬렉션 중단	MongoDB를 소스로 사용할 때 빈 컬렉션으로 인해 작업이 중단되는 문제를 해결했습니다.
Amazon Redshift를 대상으로 사용할 시 전체 로드 작업 중단	Amazon Redshift를 대상으로 사용할 때 복구 체크포인트 컨트롤 테이블이 활성화된 경우 전체 로드 단계에서 작업이 중단되는 문제를 해결했습니다.
S3에서 S3로 데이터 이동 없음.	가 지정되지 않은 경우 AWS DMS 가 데이터를 복제하지 않는 S3에서 S3로의 복제 문제를 수정bucketFolder 했습니다.

해결된 문제	설명
GlueCatalogGeneration 이 true로 설정 시 CDC 지연 시간	S3를 대상으로 사용할 때 GlueCatalogGeneration 가 true로 설정된 경우 과도한 지연 시간이 발생하는 문제를 해결했습니다.
Oracle을 대상으로 사용 시 데이터 잘림	Oracle을 대상으로 사용할 때가 VARCHAR2 열의 데이터를 AWS DMS 잘라내는 문제를 수정했습니다.
PostgreSQL 언더스코어 와 일드카드 동작	PostgreSQL을 소스로 사용할 때 선택 규칙의 '_' 와일드카드 동작이 문서에 설명된 대로 작동하지 않는 문제를 해결했습니다.
PostgreSQL을 소스로 사용 시 빈 WAL 헤더 문제.	PostgreSQL을 소스로 사용할 때 복제 슬롯에서 수신한 빈 WAL 헤더로 인해 작업이 실패하는 문제를 해결했습니다.
MySQL 또는 MariaDB를 소스로 사용 시 압축된 바이너리 로그 관련 문제	MySQL 및 MariaDB를 소스로 사용할 때에서 BINLOG 압축을 AWS DMS 감지했을 때 적절한 오류 메시지가 발생하지 않는 문제를 수정했습니다.
S3 데이터 검증 특수 문자	프라이머리 키와 비프라이머리 키 열의 특수 문자를 처리하도록 S3 데이터 검증을 개선했습니다.
Amazon Redshift를 대상으로 사용 시 잘못된 작업 로그 항목.	Amazon Redshift를 대상으로 사용할 때 UPDATES 및 DELETES에서 batch-apply 문 실패를 보고하는 잘못된 항목이 작업 로그에 표시되는 문제를 해결했습니다.
SQL Server에서 S3로의 마이그레이션 작업 중단.	SQL Server에서 S3로 마이그레이션할 때 캐시된 변경 사항을 적용하는 동안 작업이 중단되는 문제를 해결했습니다.
일괄 적용 오류 시 데이터 누락.	배치 적용 시 오류가 발생하면 데이터가 누락되는 배치 적용 기능 관련 문제를 해결했습니다.
SQL Server 소스에 대한 로깅 개선 사항	스토리지 단위 값을 제외하지 않도록 SQL Server 소스에 대한 로깅을 개선했습니다. 누락된 권한을 올바르게 나타내도록 AlwaysOn 구성에서 SQL Server 소스에 대한 로깅을 개선했습니다.
Kafka 대상에 대한 로깅 개선 사항	Kafka 대상에 대한 더 나은 가시성과 문제 해결 용량을 제공하기 위해 몇 가지 로그인 개선 사항이 도입되었습니다.

해결된 문제	설명
Oracle 소스에 대한 로깅 개선 사항입니다.	프라이머리 키 누락으로 인해 테이블을 건너뛰고 있음을 올바르게 나타내기 위해 이진 리더를 사용하는 Oracle 소스에 대한 로깅을 개선했습니다.
DDL이 비활성화된 마이그레이션에 대한 로깅 개선 사항	외부에서 수정된 후 예기치 않은 대상 테이블 구조를 나타내도록 DDL 복제가 비활성화된 마이그레이션에 대한 로깅을 개선했습니다 AWS DMS.
일시 중지된 소스 캡처에 대한 로깅 개선 사항입니다.	일시 중지된 소스 캡처 상황을 더 잘 설명하기 위해 로깅을 개선했습니다.
AWS DMS 스왑을 위한 로깅 개선 사항	AWS DMS 가 내부 스왑 파일에서 읽는 시기를 나타내도록 로깅을 개선했습니다.
Amazon Redshift 대상에 대한 로깅 개선 사항	기본 로깅 수준에 자세한 정보를 포함하도록 Amazon Redshift 대상에 대한 로깅을 개선했습니다.
메타데이터 관리자에 대한 로깅 개선 사항	정보 로깅 수준에서 테이블 metadata 관련 문제를 보고하도록 로깅을 개선하여 문제 해결을 간소화했습니다.
Amazon Redshift 대상에 대한 데이터 검증 개선 사항	HandleCollationDiff 설정을 지원하도록 Amazon Redshift 대상에 대한 데이터 검증 기능을 개선했습니다.
데이터 검증 재검증 옵션 사용 불가	특정 상황에서 재검증 옵션을 사용할 수 없는 데이터 검증 기능의 문제를 수정했습니다.
테이블 다시 로드 문제	테이블 중 하나 이상이 유효하지 않을 때 여러 테이블의 재로드가 취소되는 문제를 수정했습니다.
MySQL에서 Amazon Redshift JSON으로 마이그레이션 문제.	배치 적용이 활성화된 상태에서 JSON 데이터 형식이 고립되지 않는 MySQL 소스 문제를 수정했습니다.
열 필터링 문제	FL 중에 새로 추가된 열에 필터가 올바르게 적용되지 않는 열 필터링 기능의 문제를 수정했습니다.

해결된 문제	설명
선택 규칙과 관련된 Db2 LUC 소스 문제입니다.	선택 규칙의 "table-type" 옵션이 무시되는 Db2 LUW 소스 문제를 수정했습니다.
필터링된 작업의 데이터 검증 문제입니다.	데이터를 검증하는 동안 필터를 준수하지 않는 데이터 검증 기능의 문제를 수정했습니다.
LOB 마이그레이션 문제	특정 이벤트 유형을 처리하는 동안 AWS DMS 작업이 충돌하는 LOB 마이그레이션 문제를 수정했습니다.
데이터 유효성 검사 문제	특정 DDL 이벤트에서 검증 전용 작업이 중단되는 데이터 검증 기능의 문제를 수정했습니다.
필터 관련 데이터 검증 문제	필터가 있을 때 HandleCollationDiff 설정이 적용되지 않는 데이터 검증 기능의 문제를 수정했습니다.
MySQL SourceEnum 인코딩 문제.	MySQL 소스에서 UTF-16 인코딩 열거형 값이 올바르게 마이그레이션되지 않는 문제를 수정했습니다.
SQL Server에서 APG로의 마이그레이션에 대한 데이터 검증 문제입니다.	SQL Server에서 PostgreSQL로 마이그레이션할 때 데이터 유효성이 특정 상황에서 거짓 긍정을 보고하는 문제를 수정했습니다.
Amazon Redshift 대상 CDC에 대한 EmptyAsNull ECA 문제	Amazon Redshift 대상에서 EmptyAsNull ECA가 제대로 작동하지 않는 문제를 수정했습니다.
메모리 누수 문제	CSV 파일을 사용하여 데이터를 로드하는 대상 DMS 작업에서 메모리 누수가 표시되는 문제를 수정했습니다.
데이터 검증을 통한 변환 규칙	재정의 규칙이 존재할 때 변환 규칙이 무시되는 데이터 검증 기능의 문제를 수정했습니다.
S3 대상 문제	cdcInsertAndUpdateOnly 설정이 활성화되었을 때 CdcMaxBatchInterval 및 CdcMinFileSize 가 적용되지 않는 Amazon S3 대상 문제를 수정했습니다.

해결된 문제	설명
MySQL 대상 문제	손상된 열 메타데이터로 인해 AWS DMS 작업 충돌 및/또는 데이터 손실이 발생할 수 있는 MySQL 대상 문제를 수정했습니다.
데이터 유효성 검사 문제	모든 테이블 근거에서 검증 프로세스가 조기에 종료되는 데이터 검증 기능의 문제를 수정했습니다.
Oracle 대상 배치 적용 문제	배치 적용이 활성화된 상태에서 AWS DMS 작업이 충돌하는 Oracle 대상 문제를 수정했습니다.
Amazon S3 대상 데이터 검증 문제.	Amazon S3 대상 데이터 검증에서 Athena가 테이블 이름을 올바르게 저장하지 않아 태스크가 실패하던 문제를 수정했습니다.
MongoDB 및 Amazon DocumentDB Secret Manger 문제.	Secret Manager에서 자격 증명을 검색할 수 없어 오류가 발생하는 MongoDB 및 Amazon DocumentDB 엔드포인트 문제를 수정했습니다.
Oracle 데이터 검증은 문제를 완료하지 않습니다.	Oracle 데이터 검증에서 특정 테이블의 검증이 완료되지 않는 문제를 수정했습니다.
데이터 검증 중단 문제	메모리 할당 부족으로 인해 특정 테이블의 검증이 중단되는 데이터 검증 기능의 문제를 수정했습니다.
Amazon S3 대상 작업 충돌 문제	이 활성화되면 변경 테이블 DDL을 수신한 후 AWS DMS 작업이 충돌하는 Amazon S3 대상 문제를 수정GlueCatalogGeneration 했습니다.
데이터 유효성 검사 문제	NUL (0x00) 문자에 대한 검증이 실패하는 데이터 검증 기능의 문제를 수정했습니다.
Babelsifh 엔드포인트 문제	대소문자가 혼합된 테이블 이름이 일시 중지되는 babelfish 엔드포인트 문제를 수정했습니다.
Amazon Redshift 대상 누락 데이터 문제	Amazon Redshift 대상에서 특정 조건에서 ParallelLoadThreads 가 >0일 때 데이터 손실이 발생하는 문제를 수정했습니다.
Amazon S3 대상 데이터 검증 문제	테이블에 PK 이외의 열이 없을 때 검증이 실패하는 Amazon S3 대상 데이터 검증 문제를 수정했습니다.

해결된 문제	설명
데이터 검증 CloudWatch 지표	검증을 위해 CloudWatch 지표가 누락되어 완료하는 데 짧은 시간이 걸리는 데이터 검증 기능의 문제를 수정했습니다.
일괄 메모리 누수 문제 적용	특정 조건에서 발생하는 배치 적용 기능의 메모리 누수 문제를 수정했습니다.
AWS DMS 작업 시작 문제	AWS DMS 태스크 시작이 매우 오래 걸리고 완료되지 않는 문제를 수정했습니다.
PostgreSQL Souce 누락된 데이터 문제	복제 슬롯의 알 수 없는 이벤트로 인해 데이터 손실이 발생하는 PostgreSQL 소스 문제를 수정했습니다.
Amazon S3 대상 데이터 손실 문제	Amazon S3 대상에서 실패한 LOB 조회로 인해 데이터가 손실되는 문제를 수정했습니다.

AWS Database Migration Service 3.5.0 베타 릴리스 정보

Important

AWS DMS 3.5.0은 복제 인스턴스 엔진의 베타 버전입니다.는 이전 릴리스와 동일한이 버전을 AWS DMS 지원합니다. 그러나 프로덕션 목적으로 사용하기 전에 AWS DMS 3.5.0 베타를 테스트하는 것이 좋습니다.

다음 표에는 AWS Database Migration Service(AWS DMS) 버전 3.5.0 베타에 도입된 새로운 기능과 향상된 기능이 나와 있습니다.

새로운 기능 또는 향상된 기능	설명
Oracle 및 Microsoft SQL Server용 Time Travel	이제 DMS 지원 Oracle, Microsoft SQL Server 및 PostgreSQL 소스 엔드포인트와 DMS 지원 PostgreSQL 및 MySQL 대상 엔드포인트가 있는 모든 AWS 리전에서 Time Travel을 사용할 수 있습니다.

새로운 기능 또는 향상된 기능	설명
S3 유효성 검사	AWS DMS 는 이제 Amazon S3 대상 엔드포인트에서 복제된 데이터의 검증을 지원합니다. Amazon S3 대상 데이터 유효성 검사에 대한 자세한 내용은 Amazon S3 대상 데이터 검증 섹션을 참조하세요.
Glue Catalog 통합	AWS Glue 는 데이터를 분류하는 간단한 방법을 제공하는 서비스로, 라고 하는 메타데이터 리포지토리로 구성됩니다 AWS Glue Data Catalog. 이제를 Amazon S3 대상 엔드포인트 AWS Glue Data Catalog 와 통합하고 Amazon Athena와 같은 다른 AWS 서비스를 통해 Amazon S3 데이터를 쿼리할 수 있습니다. Amazon Athena 자세한 내용은 에 Amazon S3 대상과 AWS Glue Data Catalog 함께 사용 AWS DMS 단원을 참조하십시오.
DocumentDB 대상에 대한 병렬 적용	DocumentDB를 새 ParallelApply* 작업 설정과 함께 대상으로 사용하면 AWS DMS 이제는 CDC 복제 중에 초당 최대 5,000개의 레코드를 지원합니다. 자세한 내용은 Amazon DocumentDB를 AWS Database Migration Service의 대상으로 사용 단원을 참조하십시오.
고객 중심 로깅	이제 AWS DMS 버전 3.5.0을 사용하여 작업 로그를 더 효과적으로 검사하고 관리할 수 있습니다. AWS DMS 작업 로그 보기 및 관리에 대한 자세한 내용은 섹션을 참조하세요 AWS DMS 작업 로그 보기 및 관리 .
Kafka 대상 엔드포인트에 대한 SASL_PLAIN 메커니즘	이제 SASL_PLAIN 인증을 사용하여 Kafka MSK 대상 엔드포인트를 지원할 수 있습니다.
MySQL에서 XA 트랜잭션 복제	이제 MySQL DMS 소스에서 XA 트랜잭션을 사용할 수 있습니다. DMS 3.5.0 이전에는 XA 트랜잭션의 일부로 적용된 DML 변경 사항이 제대로 복제되지 않았습니다.
Oracle 확장 데이터 형식	AWS DMS 는 이제 Oracle 버전 12.2 이상에서 확장 데이터 유형의 복제를 지원합니다.

새로운 기능 또는 향상된 기능	설명
Db2 LUW PureScale 환경	AWS DMS 는 이제 Db2 LUW PureScale 환경에서 복제를 지원합니다. 이 기능은 소스 변경 위치에서 변경 사항 처리 시작 옵션을 사용해야만 지원됩니다.
READ_COMMITTED_SNAPSHOT 옵션이 설정된 SQL Server 소스	READ_COMMITTED_SNAPSHOT 옵션이 TRUE로 설정된 Microsoft SQL Server 소스 데이터베이스를 사용하는 경우 forceData RowLookup 연결 속성을 설정하여 DML 변경 사항을 올바르게 복제할 수 있습니다.

AWS DMS 3.5.0에는 다음과 같은 해결된 문제가 포함되어 있습니다.

202 AWS DMS 3년 3월 17일에 시작된 3.5.0에서 해결된 문제 17-March-2023

Topic	해결 방법	
Oracle - 숫자에서 변환된 문자열의 특별 사례 비교	Oracle 소스에서 문자열로의 데이터 형식 변환이 있는 숫자 열에 대해 필터링 규칙이 예상대로 작동하지 않는 문제를 수정했습니다.	
온프레미스 SQL Server AG의 향상된 기능	AlwaysOn 구성에서 DMS가 사용하지 않는 복제본에 대한 불필요한 연결을 제거하여 SQL Server 소스와의 연결 처리 효율성을 개선했습니다.	
SQL Server HIERARCHYID 내부 변환	SQL Server 소스에서 HIERARCHYID 데이터 형식이 SQL Server 대상에 HIERARCHYID가 아닌 VARCHAR(250)로 복제되는 문제를 수정했습니다.	
S3 대상 이동 태스크 수정	S3 대상으로의 이동 태스크가 시간이 매우 오래 걸리거나 중단된 것처럼 보이거나 완료되지 않는 문제를 수정했습니다.	
Kafka의 SASL Plain 메커니즘	Kafka MSK 대상 엔드포인트에 대한 SASL Plain 인증 방법 지원을 도입했습니다.	

Topic	해결 방법	
Opensearch 2.x의 _type 파라미터로 인해 병렬 로드 및 병렬 적용 실패	Opensearch 2.x 대상에서 _type 파라미터가 지원되지 않아 병렬 로드 또는 병렬 적용이 실패하는 문제를 수정했습니다.	
혼합 연산자를 사용하는 테이블 매핑 필터 지원	한 열에 하나의 필터만 적용할 수 있는 제한을 제거했습니다.	
S3, Kinesis, Kafka 엔드포인트 - CDC 단계의 변경 기반 로그 열 마이그레이션	Kinesis, Kafka 및 S3 대상에서 CDC 중에 추가된 LOB 열의 데이터가 복제되지 않는 문제를 수정했습니다.	
MongoDB 드라이버 업그레이드	MongoDB 드라이버를 v1.23.2로 업그레이드했습니다.	
Kafka 드라이버 업데이트	Kafka 드라이버를 1.5.3에서 1.9.2로 업그레이드했습니다.	
S3 엔드포인트 설정이 제대로 작동하지 않음	S3 대상에서 데이터에 S3 대상용 구분 기호로 지정된 문자가 포함된 경우 AddTrailingPadding Character 엔드포인트 설정이 작동하지 않는 문제를 수정했습니다.	
Kinesis 대상 태스크가 충돌함	Kinesis 대상에서 PK 값이 비어 있고 상세 디버그가 활성화되어 있을 때 태스크가 중단되는 문제를 수정했습니다.	
S3 대상의 열 이름이 한 자리씩 이동된 경우	S3 대상에서 AddColumnName 이 true로 설정되고 TimestampColumnName 이 ""로 설정되어 있을 때 열 이름이 한 자리씩 이동하는 문제를 수정했습니다.	
로깅 LOB 자르기 경고 개선	LOB를 검색하는 데 사용되는 select 문을 포함하도록 SQL Server 소스의 LOB 자르기에 대한 경고 로깅을 개선했습니다.	

Topic	해결 방법	
TDE 암호가 잘못된 경우 DMS 태스크 충돌이 발생하지 않도록 치명적 오류 추가.	Oracle 소스에서 잘못된 TDE 암호로 인한 오류 메시지 없이 DMS 태스크가 실패하는 상황에서 의미 있는 오류 메시지를 도입하고 태스크 충돌 문제를 없앴습니다.	
CDC 중에 PostgreSQL CTAS(create table as selected) DDL의 마이그레이션 허용.	CDC 중에 DMS에서 PostgreSQL CTAS(create table as selected) DDL을 복제할 수 없는 제한을 제거했습니다.	
CDC에서 테이블 열이 삭제될 때 발생하는 pg_logical 태스크 충돌을 수정했습니다.	S3 대상을 사용하는 PostgreSQL 소스에서 LOB 지원이 비활성화되었지만 LOB가 있을 때 대상에서 열이 잘못 정렬되는 문제를 수정했습니다.	
MySQL 연결 처리에서 메모리 누수 수정	MySQL 소스에서 태스크 메모리 사용량이 지속적으로 증가하는 문제를 수정했습니다.	
Oracle 소스 엔드포인트 설정 - ConvertTimestampWithZoneToUTC	'TIMESTAMP WITH TIME ZONE' 및 'TIMESTAMP WITH LOCAL TIME ZONE' 열의 타임스탬프 값을 UTC로 변환하려면 이 속성을 true로 설정합니다. 기본적으로 이 속성의 값은 'false'이며, 데이터는 소스 데이터베이스 시간대를 사용하여 복제됩니다.	
Oracle 소스 - SUSPEND_TABLE을 위한 DataTruncationErrorPolicy가 작동하지 않음	S3 대상을 사용하는 Oracle 소스에서 DataTruncationErrorPolicy 태스크 설정이 SUSPEND_TABLE로 설정되어 있는 동안 테이블이 일시 중단되지 않는 문제를 수정했습니다.	
쿼리 절을 작성하는 동안 긴 스키마 및 테이블에서 SQL Server 실패	SQL Server 소스에서 선택 규칙에 쉼표로 구분된 테이블 목록이 포함되면 태스크가 실패하거나 응답하지 않는 문제를 수정했습니다.	

Topic	해결 방법	
MongoDB 엔드포인트를 사용하는 Secret Manager 인증	MongoDB 및 DocumentDB 엔드포인트에서 Secret Manager 기반 인증이 작동하지 않는 문제를 수정했습니다.	
NLS_NCHAR_CHARACTERSET이 UTF8로 설정된 경우 DMS가 CDC 중에 멀티바이트 varchar 열의 데이터를 자름	Oracle 대상을 사용하는 Oracle 소스에서 NLS_NCHAR_CHARACTERSET이 UTF8로 설정된 멀티바이트 VARCHAR 열의 데이터가 잘리는 문제를 수정했습니다.	
Oracle LogMiner용 filterTransactionsOfUser ECA	LogMiner를 사용하여 Oracle에서 데이터를 복제할 때 DMS가 지정된 사용자의 트랜잭션을 무시할 수 있도록 추가 연결 속성(ECA) filterTransactionsOfUser 를 추가했습니다.	
백업에서 lsn이 누락된 경우 SQL Server가 복구 가능 오류 설정	SQL Server에서 LSN이 누락되어도 태스크가 실패하지 않는 문제를 수정했습니다.	

AWS Database Migration Service 3.4.7 릴리스 정보

다음 표에는 AWS Database Migration Service(AWS DMS) 버전 3.4.7에 도입된 새로운 기능과 향상된 기능이 나와 있습니다.

새로운 기능 또는 향상된 기능	설명
Babelfish를 대상으로 지원	AWS DMS 는 이제 Babelfish를 대상으로 지원합니다. AWS DMS를 사용하면 가동 중지 시간을 최소화하면서 AWS DMS 지원되는 모든 소스에서 Babelfish로 라이브 데이터를 마이그레이션할 수 있습니다. 자세한 내용은 Babelfish를의 대상으로 사용 AWS Database Migration Service 단원을 참조하십시오.

새로운 기능 또는 향상된 기능	설명
전체 로드 전용 소스로서 IBM DB2 z/OS 데이터베이스 지원	AWS DMS 는 이제 IBM Db2 z/OS 데이터베이스를 소스로 지원합니다. 이제 AWS DMS를 사용하여 Db2 메인프레임에서 AWS DMS 지원되는 모든 대상으로 라이브 마이그레이션을 수행할 수 있습니다. 자세한 내용은 IBM Db2 for z/OS 데이터베이스를의 소스로 사용 AWS DMS 단원을 참조하십시오.
SQL Server 읽기 전용 복제본을 소스로 지원	AWS DMS 는 이제 SQL Server 읽기 전용 복제본을 소스로 지원합니다. 이제 AWS DMS를 사용하여 SQL Server 읽기 전용 복제본에서 AWS DMS 지원되는 대상으로 실시간 마이그레이션을 수행할 수 있습니다. 자세한 내용은 Microsoft SQL Server 데이터베이스를의 소스로 사용 AWS DMS 단원을 참조하십시오.
EventBridge DMS 이벤트 지원	AWS DMS 는 DMS 이벤트에 EventBridge를 사용하여 이벤트 구독 관리를 지원합니다. 자세한 내용은 에서 Amazon EventBridge 이벤트 및 알림 작업 AWS Database Migration Service 단원을 참조하십시오.

새로운 기능 또는 향상된 기능	설명
VPC 소스 및 대상 엔드포인트 지원	AWS DMS 는 이제 Amazon Virtual Private Cloud(VPC) 엔드포인트를 소스 및 대상으로 지원합니다. 이제 AWS 서비스에 대해 명시적으로 정의된 경로가 VPC에 정의된 경우는 AWS DMS VPC 엔드포인트가 있는 모든 서비스에 연결할 AWS DMS 수 있습니다.  Note AWS DMS 버전 3.4.7 이상으로 업그레이드하려면 먼저 VPC 엔드포인트를 AWS DMS 사용하거나 퍼블릭 경로를 사용하도록 구성해야 합니다. 이 요구 사항은 Amazon S3, Amazon Kinesis Data Streams, Amazon DynamoDB AWS Secrets Manager, Amazon Redshift 및 Amazon OpenSearch Service의 소스 및 대상 엔드포인트에 적용됩니다. 자세한 내용은 에 대한 VPC 엔드포인트 구성 AWS DMS 단원을 참조하십시오.
새 PostgreSQL 버전	PostgreSQL 버전 14.x가 이제 소스 및 대상으로 지원됩니다.
Aurora Serverless v2를 대상으로 지원	AWS DMS 는 이제 Aurora Serverless v2를 대상으로 지원합니다. 이제 AWS DMS를 사용하여 Aurora Serverless v2로 라이브 마이그레이션을 수행할 수 있습니다. 지원되는 AWS DMS 대상에 대한 자세한 내용은 섹션을 참조하십시오 요마이그레이션에 적합한 대상.

새로운 기능 또는 향상된 기능	설명
새 IBM DB2 for LUW 버전	AWS DMS 는 이제 IBM Db2 for LUW 버전 11.5.6 및 11.5.7을 소스로 지원합니다. 이제 이를 사용하여 최신 버전의 IBM DB2 for LUW에서 라이브 마이그레이션을 수행할 AWS DMS 수 있습니다. AWS DMS 소스에 대한 자세한 내용은 섹션을 참조하세요 데이터 마이그레이션용 소스. 지원되는 AWS DMS 대상에 대한 자세한 내용은 섹션을 참조하세요 마이그레이션에 적합한 대상.

AWS DMS 3.4.7에는 다음과 같은 새로운 동작 또는 변경된 동작과 해결된 문제가 포함되어 있습니다.

- 이제 Amazon S3를 소스로 사용할 때 테이블 정의의 날짜 형식을 사용하여 데이터 문자열을 날짜 객체로 파싱할 수 있습니다.
- 이제 새 테이블 통계 카운터(AppliedInserts, AppliedDdls, AppliedDeletes, AppliedUpdates.)를 사용할 수 있습니다
- 이제 OpenSearch를 대상으로 사용할 때 기본 매핑 유형을 선택할 수 있습니다.
- Oracle, PostgreSQL 및 SQLServer 소스에 대한 새로운 TrimSpaceInChar 엔드포인트 설정을 통해 CHAR 및 NCHAR 데이터 형식의 데이터를 트리밍할지 여부를 지정할 수 있습니다.
- Amazon S3에 대한 새로운 ExpectedBucketOwner 엔드포인트 설정이 S3를 소스 또는 대상으로 사용할 때 스나이핑을 방지합니다.
- RDS SQL Server, Azure SQL Server 및 자체 관리형 SQL Server의 경우 - 이제 DMS는 프라이머리 키가 있거나 없는 마이그레이션 태스크에 선택된 모든 테이블에서 또는 프라이머리 키가 있는 자체 관리형 SQL Server 테이블에서 MS-REPLICATION의 활성화 우선 순위를 고려하는 고유 인덱스를 사용하여 MS-CDC 자동 설정을 제공합니다.
- Oracle 동종 마이그레이션 중에 Oracle 파티션 및 하위 파티션 DDL 작업의 복제에 대한 지원을 추가했습니다.
- Oracle을 소스 및 대상으로 사용할 때 복합 프라이머리 키로 인해 데이터 유효성 검사 태스크가 충돌하는 문제를 수정했습니다.
- Amazon Redshift를 대상으로 사용할 때 대상 열이 부울로 미리 생성된 상태에서 CHARACTER VARYING 형식을 부울로 올바르게 캐스팅하는 문제를 수정했습니다.

- PostgreSQL을 대상으로 사용할 때 알려진 ODBC 문제로 인해 마이그레이션된 varchar 데이터 형식의 데이터가 잘리는 문제를 수정했습니다.
- Oracle을 대상으로 사용할 때 BatchApplyEnabled가 true로 설정되고 BatchApplyPreserveTransaction이 false로 설정되면 DELETE 작업에 대한 병렬 힌트가 준수되지 않는 문제를 수정했습니다.
- Amazon S3의 새 AddTrailingPaddingCharacter 엔드포인트 설정은 S3를 대상으로 사용할 때 문자열 데이터에 패딩을 추가합니다.
- 새 max_statement_timeout_seconds 태스크 설정은 엔드포인트 쿼리의 기본 제한 시간을 연장합니다. 이 설정은 현재 MySQL 엔드포인트 메타데이터 쿼리에 사용됩니다.
- PostgreSQL을 대상으로 사용할 때 CDC 태스크가 오류 처리 태스크 설정을 제대로 활용하지 않는 문제를 수정했습니다.
- DMS가 Redis OSS Enterprise 인스턴스의 Redis OSS 모드를 올바르게 식별하지 못하는 문제를 수정했습니다.
- S3 대상 Parquet 형식에 대한 includeOpForFullLoad 추가 연결 속성(ECA) 지원을 확대했습니다.
- 새로운 PostgreSQL 엔드포인트 설정 migrateBooleanAsBoolean을 도입했습니다. PostgreSQL에서 Amazon Redshift로 마이그레이션할 때 이 설정을 true로 설정하면 부울이 varchar(1)로 마이그레이션됩니다. false로 설정하면 부울이 varchar(15)로 마이그레이션되며 이는 기본 동작입니다.
- SQL Server 소스를 사용할 때 datetime 데이터 형식과 관련된 마이그레이션 문제를 수정했습니다. 이 수정 사항은 정밀도가 밀리초 단위인 경우 Null 삽입 문제를 해결합니다.
- PGLOGICAL을 사용하는 PostgreSQL 소스의 경우, pglogical을 사용하고 CDC 단계 중에 소스 테이블에서 필드를 제거할 때 제거된 필드 이후의 값이 대상 테이블로 마이그레이션되지 않는 마이그레이션 문제를 수정했습니다.
- 양방향 복제가 반복된 레코드를 가져오는 SQL Server 루프백 마이그레이션 문제를 수정했습니다.
- PostgreSQL 소스를 위한 새 ECA mapBooleanAsBoolean을 추가했습니다. 이 추가 연결 속성을 사용하여 PostgreSQL 부울의 기본 데이터 형식 매핑을 RedShift 부울 데이터 형식으로 재정의할 수 있습니다.
- SQL Server를 소스로 사용할 때 ALTER DECIMAL/NUMERIC SCALE이 대상에 복제되지 않는 마이그레이션 문제를 수정했습니다.
- 2022년 10월 17일부터 DMS 3.4.7은 복제 인스턴스에 대한 6세대 Amazon EC2 인스턴스 클래스를 지원합니다.
- 2022년 11월 25일부터 DMS 3.4.7에서 DMS 스키마 변환을 사용하여 데이터베이스 스키마와 코드 객체를 변환하고, DMS Fleet Advisor를 사용하여 네트워크 환경에서 마이그레이션에 적합한 데이터베이스를 검색할 수 있습니다.

- 2022년 11월 25일부터 DMS 스튜디오는 사용 중지됩니다.
- 2023년 1월 31일부터 DMS 스키마 변환은 Aurora MySQL 및 Aurora PostgreSQL을 대상 데이터 공급자로 지원합니다.
- 2023년 3월 6일부터 DMS Fleet Advisor를 사용하여 소스 데이터베이스에 적합한 크기의 대상 권장 사항을 생성할 수 있습니다.
- 2023년 3월 6일부터는 지표 데이터 포인트를 Amazon CloudWatch에 게시할 수 있는 AWS 관리형 정책을 AWS DMS 지원합니다.

2023년 5월 5일에 발표된 DMS 3.4.7 유지 관리 릴리스에서 해결된 문제

Topic	해결 방법	
PostgreSQL 소스 태스크 실패	PostgreSQL 소스에서 단일 이벤트에 허용되는 최대 DDL 태스크를 초과할 경우 태스크가 실패하는 문제를 수정했습니다.	
PostgreSQL 소스 데이터 유효성 검사 거짓 긍정	Oracle 대상을 사용하는 PostgreSQL 소스에서 타임스탬프 필드를 잘못 캐스팅하면 거짓 긍정 데이터 유효성 검사 오류가 발생하는 문제를 수정했습니다.	
MySQL 소스 오류 처리	MySQL 소스에서 다음 BIN 로그를 사용할 수 없을 때 DMS 태스크가 실패하지 않는 문제를 수정했습니다.	
MySQL 소스 ROTATE_EVENT 로깅	MySQL 소스에서 ROTATE_EVENT와 관련된 로깅을 개선했습니다. 읽는 BIN 로그 이름을 포함시켰습니다.	
데이터 유효성 검사 시간 초과 문제	데이터 유효성 검사와 관련된 쿼리에 대해 <code>executeTimeout</code> 엔드포인트 설정이 준수되지 않는 데이터 유효성 검사 기능 문제를 수정했습니다.	
PostgreSQL 대상 병렬 전체 로드 문제	PostgreSQL 대상에서 '연결 중단' 오류로 인해 세그먼트 (병렬) 전체 로드가 실패하는 문제를 수정했습니다.	

Topic	해결 방법	
DMS 태스크 이동 문제	S3 대상에서 DMS 태스크 이동 작업이 매우 오래 걸리거나 완료되지 않는 문제를 수정했습니다.	
PostgreSQL 소스 중복 레코드 문제	PostgreSQL 소스에서 태스크가 중지 후 재개되면 DMS 태스크가 대상에서 중복 관련 오류를 throw 하는 문제를 수정했습니다.	
Oracle 대상 데이터 유효성 검사 거짓 긍정	Oracle 대상에서 타임스탬프 필드의 시간대가 잘못 복제되어 데이터 유효성 검사가 거짓 긍정 오류를 보고하는 문제를 수정했습니다.	

2023년 2월 22일에 발표된 DMS 3.4.7 유지 관리 릴리스에서 해결된 문제

Topic	해결 방법	
소스로서의 SQL Server AG 복제본	리스너 TCP 포트가 복제본 TCP 포트와 다른 AlwaysOn 구성에서 SQL Server 소스에 대한 지원을 추가했습니다.	
Amazon Redshift 대상에서 데이터 손실	Amazon Redshift 대상에서 드문 경우지만 예상치 못한 Amazon Redshift 재시작으로 인해 대상에서 데이터가 누락될 수 있었던 문제를 수정했습니다.	
SQL Server 소스 세이프가드 지원	SQL Server 소스에서 엔드포인트 설정 "SafeguardPolicy": "EXCLUSIVE_AUTOMATIC_TRUNCATION" 이 지정되면 DMS 태스크가 실패하고 트랜잭션 로그 백업을 읽을 수 없다는 오류가 발생하는 문제를 수정했습니다.	
Oracle 소스에 대한 데이터 유효성 검사 태스크 실패	Oracle 소스에서 잘못 식별된 프라이머리 키 값으로 인해 DMS 태스크가 데이터 유효성 검사에서 실패할 수 있는 문제를 수정했습니다.	
Kinesis 이미지 전 데이터 문제	스트리밍 대상(Kinesis, Kafka) 에서 "EnableBeforeImage" 태스크 설정이 문자 데이터 형식에만 작동하는 문제를 수정했습니다.	

Topic	해결 방법	
Time Travel 로그 파일	Time Travel 기능에서 소스가 유휴 상태일 때 DMS가 0 바이트 Time Travel 로그 파일을 생성하는 문제를 수정했습니다.	

2022년 12월 16일에 발표된 DMS 3.4.7 유지 관리 릴리스에서 해결된 문제

Topic	해결 방법	
BatchApplyEnabled	BatchApplyEnabled가 True로 설정되었을 때 과도한 로깅 문제를 수정했습니다.	
새 MongoDB 엔드포인트 설정 - FullLoadNoCursorTimeout	MongoDB 엔드포인트 설정 FullLoadNoCursorTimeout은 전체 로드 커서에 대해 NoCursorTimeout을 지정합니다. NoCursorTimeout은 서버가 유휴 상태인 경우 커서를 닫지 못하도록 하는 MongoDB 연결 설정입니다.	
MongoDB - 단일 열 세그먼트화를 위한 필터 함수	새로운 필터 함수는 세그먼트화에 단일 열을 사용하여 MongoDB 데이터베이스 마이그레이션 성능을 개선합니다.	
MongoDB에서 Amazon Redshift로	MongoDB에서 Amazon Redshift로 마이그레이션할 때 MongoDB 컬렉션에 이진 데이터 형식이 있는 경우 DMS가 Amazon Redshift에서 대상 테이블을 생성하지 않는 문제를 수정했습니다.	
새 MongoDB SocketTimeoutMS 연결 속성	새 MongoDB SocketTimeoutMS 추가 연결 속성은 MongoDB 클라이언트의 연결 제한 시간을 밀리초 단위로 구성합니다. 값이 0보다 작거나 같으면 MongoDB 클라이언트 기본값이 사용됩니다.	
Amazon Kinesis 태스크가 충돌을 일으키는 문제를 수정했습니다.	Amazon Kinesis Data Streams 대상으로 마이그레이션할 때 테이블에 프라이머리 키가 없는 경우 null 값을 처리하는 문제를 수정했습니다.	

Topic	해결 방법	
Oracle NULL PK/UK 데이터 유효성 검사 지원	NULL PK/UK 값의 데이터 유효성 검사가 지원되지 않는 제한을 제거했습니다.	
Oracle에서 Amazon S3로	Oracle에서 Amazon S3로 마이그레이션할 때 일부 레코드가 NULL로 잘못 마이그레이션되는 문제를 수정했습니다.	
Oracle Standby	Oracle Standby를 소스로 사용할 때 DMS가 미결 트랜잭션을 처리할 수 있는 기능을 추가했습니다.	
SDO_GEOMETRY 공간 데이터 형식을 사용하는 Oracle에서 Oracle로 마이그레이션	Oracle에서 Oracle로 마이그레이션할 때 테이블의 DDL에 SDO_GEOMETRY 열이 있는 경우 태스크가 실패하는 문제를 수정했습니다.	
Oracle 소스	Oracle을 소스로 사용할 때 DMS가 Oracle 다시 실행 로그 시퀀스 번호를 가끔 건너뛰는 문제를 수정했습니다.	
Oracle 소스 - 아카이브/온라인 다시 실행 로그 누락	Oracle을 소스로 사용할 때 아카이브 로그가 누락되면 DMS 태스크가 실패하는 문제를 수정했습니다.	
수정 - DMS가 가끔 Oracle Standby 다시 실행 로그를 건너뛰는 문제	Oracle을 소스로 사용할 때 DMS가 Oracle 다시 실행 로그 시퀀스 번호를 가끔 건너뛰는 문제를 수정했습니다.	
수정 - CDC 중에 Oracle에서 Oracle로 공간 데이터 형식이 복제되지 않는 문제	Oracle에서 Oracle로 복제할 때 CDC 중에 공간 데이터 형식이 복제되지 않는 문제를 수정했습니다.	
Oracle 대상	Oracle을 대상으로 사용할 때 ORA-01747 오류로 인해 대상 적용이 실패하는 문제를 수정했습니다.	

Topic	해결 방법	
Amazon S3 - 테이블 다시 로드 데이터 손실 문제 수정	Amazon S3를 대상으로 사용할 때 테이블 다시 로드 작업에서 CDC 파일이 생성되지 않는 문제를 수정했습니다.	
수정 - 프라이머리 서버를 소스로 사용할 때 SQL Server Always On 컨텍스트 초기화	SQL Server Always On을 소스로 사용할 때 소스가 프라이머리이고 AlwaysOnSharedSyncedBackupsEnabled가 true로 설정된 경우 가용성 그룹(AG)이 초기화되지 않는 문제를 수정했습니다.	
SQL Server 엔드포인트 설정 업데이트	소스 엔드포인트가 SQL Server Always On 가용성 그룹이고 보조 복제본인 경우 AlwaysOnSharedSyncedBackupsEnabled를 True로 설정하면 복제 태스크가 실패하는 문제를 수정했습니다.	
PostgreSQL 소스	CDC가 PostgreSQL 소스에서 삭제/업데이트 작업을 마이그레이션하지 못하는 문제를 수정했습니다. 이 수정 사항은 3.4.7에서 mapBooleanAsBoolean을 지원하면서 도입되었습니다.	

AWS Database Migration Service 3.4.6 릴리스 정보

다음 표에는 AWS Database Migration Service(AWS DMS) 버전 3.4.6에 도입된 새로운 기능과 향상된 기능이 나와 있습니다.

새로운 기능 또는 향상된 기능	설명
AWS DMS 시간 이동	AWS DMS 는 고객에게 로깅 기능에 대한 유연성을 부여하고 문제 해결 경험을 개선하는 기능인 Time Travel 을 도입했습니다. Time Travel을 사용하면 Amazon S3를 사용하여 AWS DMS 로그를 저장 및 암호화하고 특정 기간 내에 로그를 보고, 다운로드하고, 난독화할 수 있습니다.
Microsoft Azure SQL 관리형 인스턴스를 소스로 지원	AWS DMS 는 이제 Microsoft Azure SQL 관리형 인스턴스를 소스로 지원합니다. 이제 AWS DMS를 사용하여 Microsoft Azure SQL 관리

새로운 기능 또는 향상된 기능	설명
	형 인스턴스에서 AWS DMS 지원되는 대상으로 라이브 마이그레이션을 수행할 수 있습니다. AWS DMS 소스에 대한 자세한 내용은 섹션을 참조하세요 데이터 마이그레이션용 소스. 지원되는 AWS DMS 대상에 대한 자세한 내용은 섹션을 참조하세요 마이그레이션에 적합한 대상.
Google Cloud SQL for MySQL을 소스로 지원	AWS DMS 는 이제 MySQL용 Google Cloud SQL을 소스로 지원합니다. 이제 AWS DMS를 사용하여 Google Cloud SQL for MySQL에서 AWS DMS 지원되는 모든 대상으로 라이브 마이그레이션을 수행할 수 있습니다. AWS DMS 소스에 대한 자세한 내용은 섹션을 참조하세요 데이터 마이그레이션용 소스. 지원되는 AWS DMS 대상에 대한 자세한 내용은 섹션을 참조하세요 마이그레이션에 적합한 대상.
파티션된 데이터의 S3로 병렬 로드 지원	AWS DMS 는 이제 파티셔닝된 데이터에 대해 Amazon S3로 병렬 로드를 지원하므로 파티셔닝된 데이터를 지원되는 데이터베이스 엔진 소스 데이터에서 Amazon S3로 마이그레이션하기 위한 로드 시간이 단축됩니다. 이 기능은 데이터베이스 소스에 있는 테이블의 각 파티션에 대해 Amazon S3 하위 폴더를 생성하여 AWS DMS 가 병렬 프로세스를 실행하여 각 하위 폴더를 채울 수 있도록 합니다.
단일 태스크에서 여러 Apache Kafka 대상 주제 지원	AWS DMS 는 이제 단일 작업으로 Apache Kafka 다중 주제 대상을 지원합니다. 이제 AWS DMS를 사용하면 동일한 태스크를 사용하여 단일 데이터베이스의 여러 스키마를 다른 Apache Kafka 대상 주제로 복제할 수 있습니다. 따라서 동일한 소스 데이터베이스의 여러 테이블을 서로 다른 Kafka 대상 주제로 마이그레이션해야 하는 상황에서 여러 태스크를 따로 생성할 필요가 없습니다.

AWS DMS 3.4.6에서 해결된 문제는 다음과 같습니다.

- Amazon S3를 CSV 형식의 대상으로 사용할 때 프라이머리 키 열이 첫 번째 열이 아닌 경우 UPDATE 문의 열이 잘못된 열로 채워지는 문제를 수정했습니다.
- PostgreSQL로 사용할 때 제한된 LOB 모드의 BYTEA 열에 NULL 값이 있는 pglogical 플러그인을 사용할 때 AWS DMS 작업이 충돌할 수 있는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 많은 수의 소스 테이블이 삭제될 때 AWS DMS 작업이 충돌할 수 있는 문제를 수정했습니다.
- UTC가 아닌 날짜에도 파티셔닝을 허용하는 새로운 Amazon S3 설정 DatePartitionTimezone을 도입하여 Amazon S3 날짜 기반 폴더 파티셔닝을 개선했습니다.
- Amazon Redshift를 대상으로 사용할 때 소스의 TIMESTAMP WITH TIME ZONE에서 TIMESTAMPTZ로 데이터 형식 간 매핑을 지원했습니다.
- MongoDB 또는 Amazon DocumentDB를 소스로 사용할 때 와일드카드 선택 규칙이 없는 태스크에 대한 CDC의 성능을 개선했습니다.
- Db2 LUW를 소스로 사용할 때 밀줄 와일드카드 및 길이가 8 미만인 스키마 이름이 AWS DMS 태스크에서 캡처되지 않는 문제를 수정했습니다.
- OpenSearch Service를 대상으로 사용할 때 대용량 데이터 볼륨에서 AWS DMS 인스턴스의 메모리가 부족해지는 문제를 수정했습니다.
- 전체 로드 유효성 검사만 지원하여 데이터 유효성 검사 성능을 개선했습니다.
- Sybase를 소스로 사용할 때 강제 장애 조치 후 AWS DMS 태스크가 재개되지 않는 문제를 수정했습니다.
- 가 경고를 Invalid BC timestamp was encountered in column 잘못 AWS DMS 보내는 문제를 수정했습니다.

DMS 3.4.6 유지 관리 릴리스에서 해결된 문제는 다음과 같습니다.

- Oracle을 소스 및 대상으로 사용할 때 대량 적용 모드를 활성화하면 태스크가 충돌하는 문제를 수정했습니다.
- PostgreSQL이 소스일 때 전체 로드 태스크가 ExecuteTimeout 엔드포인트 설정을 제대로 사용하도록 문제를 수정했습니다.
- PostgreSQL을 소스로 사용하는 동안 태스크가 제한적 LOB 모드로 설정되면 Array 데이터 형식 열을 마이그레이션할 때 발생하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 1970년 1월 1일 이전의 시간대를 사용하는 타임스탬프를 마이그레이션할 때 발생하는 문제를 수정했습니다.
- SQL Server를 소스 및 대상으로 사용할 때 DMS가 복제 중에 빈 문자열을 null로 처리하는 문제를 수정했습니다.

- MySQL 소스 및 타겟을 사용할 때 세션 읽기 및 쓰기 제한 시간 엔드포인트 설정을 준수하도록 문제를 수정했습니다.
- Amazon S3를 소스로 사용할 때 DMS CDC 태스크가 전체 로드 관련 파일을 다운로드하는 문제를 수정했습니다.
- Amazon S3를 대상으로 사용할 때 CdcInsertsAndUpdates 및 PreserveTransactions 둘 다 true로 설정된 경우 로그 충돌 문제를 수정했습니다.
- Amazon Kinesis Data Streams를 소스로 사용할 때 ParallelApply* 기능을 활성화했지만 일부 테이블에 기본 프라이머리 키가 없을 때 태스크가 충돌하는 문제를 수정했습니다.
- Amazon Kinesis Data Streams를 소스로 사용할 때 잘못된 StreamArn에 대해 오류가 발생하지 않는 문제를 수정했습니다.
- OpenSearch를 대상으로 사용할 때 프라이머리 키 값을 빈 문자열로 지정하면 태스크가 중단되는 문제를 수정했습니다.
- 데이터 유효성 검사에 너무 많은 디스크 공간이 사용되는 문제를 수정했습니다.

2022년 12월 13일에 발표된 DMS 3.4.6 유지 관리 릴리스에서 해결된 문제

Topic	해결 방법	
SAP ASE odbc 드라이버	SAP ASE를 소스로 사용할 때 ODBC 드라이버가 문자 세트를 지원할 수 있도록 문제를 수정했습니다.	
로그 조회를 위한 Sqlserver datetime 프라이머리 키 버그	SQL Server 소스에서 프라이머리 키에 밀리초 단위 정밀도의 datetime 데이터 형식이 있으면 LOB 조회가 제대로 작동하지 않는 문제를 수정했습니다.	
SQL Server에서 Amazon Redshift로 - 'datetimeoffset'를 'timestampz'로 매핑	SQL Server에서 Amazon Redshift로 마이그레이션할 때 SQL Server의 'datetimeoffset' 형식이 Amazon Redshift 'timestampz' 형식에 매핑되도록 매핑을 개선했습니다.	
데이터 유효성 검사 - SkipLobColumns가 True	SkipLobColumns가 True이고, 소스에 LOB가 있고, 프라이머리 키가 마지막 열에 있고, 유효성 검사 결과 데이터 차이가 감지되면 DMS 태스크가 충돌하는 문제를 수정했습니다.	

Topic	해결 방법	
MySQL 소스에서의 데이터 유효성 검사	데이터 유효성 검사가 활성화된 MySQL 소스에서 null 값이 포함된 복합 고유 키가 있는 테이블을 사용하면 DMS 태스크 충돌이 발생하는 문제를 수정했습니다.	
MySQL 소스	MySQL을 소스로 사용할 때 정밀도를 높이기 위해 열을 변경하면 오버플로 오류와 함께 테이블이 일시 중단되는 문제를 수정했습니다.	
MySQL ODBC 드라이버를 8.0.23으로 업그레이드	MySQL을 소스로 사용할 때 데이터 정렬 'utf8mb4_0900_bin'이 DMS에서 사용하는 mysql 드라이버와 호환되지 않는 문제를 수정했습니다.	
MySQL - 파티션된 테이블에 대한 DDL 변경 지원	사용자가 CDC 중에 파티션 DDL 변경으로 인한 테이블 일시 중지를 건너뛸 수 있도록 새로운 MySQL 엔드포인트 설정인 skipTableSuspensionForPartitionDdl을 도입했습니다. 따라서 이제 DMS는 파티션된 MySQL 테이블에 대한 DDL 변경을 지원할 수 있습니다.	
MongoDB에서 Amazon Redshift로 마이그레이션	MongoDB에서 Amazon Redshift로 마이그레이션할 때 MongoDB 컬렉션에 이진 데이터 형식이 있으면 DMS가 Amazon Redshift에서 대상 테이블을 생성하지 못하는 문제를 수정했습니다.	
Amazon Redshift 대상 - 일괄 적용에서 Time Travel Segfault	Amazon Redshift를 대상으로 사용할 때 BatchApplyEnabled가 true로 설정된 상태에서 DMS 태스크가 충돌하는 문제를 수정했습니다.	
대상으로서의 Amazon Redshift	Amazon Redshift를 대상으로 사용할 때 parallel-load가 type=partitions-auto로 설정되면 병렬 세그먼트가 동일한 테이블 디렉터리에 대량 CSV 파일을 쓰면서 서로 간섭하는 문제를 수정했습니다.	
대상으로서의 Amazon Redshift	Amazon Redshift를 대상으로 사용할 때 CDC 중에 대상 열은 부울 형식이고 소스는 CHARACTER VARYING 형식인 문제를 수정했습니다.	

Topic	해결 방법	
대상으로서의 Amazon Redshift	Amazon Redshift 대상으로 복제되지 않는 DDL 변경 사항을 식별하도록 태스크 로그를 개선했습니다.	
PostgreSQL의 데이터 유효성 검사	PostgreSQL의 유효성 검사에서 부울 데이터 형식이 있으면 유효성 검사가 실패하는 문제를 수정했습니다.	
PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 전체 로드 추가 연결 속성의 ExecuteTimeout 필드를 사용하도록 문제를 수정했습니다.	
PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 요청된 태스크 재개 LSN보다 큰 LSN을 읽으면 60분 이상 동안 태스크가 실패하여 사용 중인 복제 슬롯에 문제가 있음을 나타내도록 문제를 수정했습니다.	
PostgreSQL 소스 - 1970년 1월 1일 이전의 timestamptz	PostgreSQL을 소스로 사용할 때 CDC 중에 1970년 1월 1일 이전의 timestamptz가 올바르게 마이그레이션되지 않는 문제를 수정했습니다.	
PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 CDC 중에 DMS가 CHARACTER VARYING 데이터 형식 값을 자르는 문제를 수정했습니다.	
PostgreSQL 소스 - 중지된 태스크 재개	PostgreSQL 소스에서 이전에 중지된 태스크 재생을 재개할 때 CDC 중에 하나 이상의 트랜잭션이 누락되는 문제를 수정했습니다.	
Amazon S3 대상	S3를 대상으로 사용할 때 AddColumnName이 true이고 TimestampColumnName이 ""이면 결과 CSV 파일 헤더가 한 열 차이로 표시되는 문제를 수정했습니다.	
Amazon S3 소스 - 태스크의 전체 로드 단계에서 메모리 사용 동작	S3를 소스로 사용할 때 전체 로드 DMS 태스크가 전체 테이블을 대상 데이터베이스에 로드한 후에만 사용된 메모리를 해제하는 문제를 수정했습니다.	
Amazon S3 대상 - 테이블 다시 로드 작업	S3를 대상으로 사용할 때 테이블 다시 로드 작업에서 CDC 파일이 생성되지 않는 문제를 수정했습니다.	

AWS Database Migration Service 3.4.5 릴리스 정보

다음 표에는 AWS Database Migration Service(AWS DMS) 버전 3.4.5에 도입된 새로운 기능과 향상된 기능이 나와 있습니다.

새로운 기능 또는 향상된 기능	설명
Redis OSS를 대상으로 지원	AWS DMS 는 이제 Redis OSS를 대상으로 지원합니다. 이제 가동 중지 시간을 최소화하면서 AWS DMS를 사용하여 AWS DMS 지원되는 모든 소스에서 Redis OSS 데이터 스토어로 라이브 데이터를 마이그레이션할 수 있습니다. AWS DMS 대상에 대한 자세한 내용은 섹션을 참조하세요 마이그레이션에 적합한 대상 .
MongoDB 4.2 및 4.4를 소스로 지원	AWS DMS 는 이제 MongoDB 4.2 및 4.4를 소스로 지원합니다. 이제 를 사용하여 MongoDB 4.2 및 4.4 클러스터의 데이터를 가동 중지 시간을 최소화하면서 Amazon DocumentDB(MongoDB 호환)를 포함한 AWS DMS 지원되는 대상으로 마이그레이션 AWS DMS할 수 있습니다. AWS DMS 소스에 대한 자세한 내용은 섹션을 참조하세요 데이터 마이그레이션용 소스 .
MongoDB를 소스로 사용 시 여러 데이터베이스 지원	AWS DMS 이제는 MongoDB를 소스로 사용하여 하나의 작업에서 여러 데이터베이스를 마이그레이션할 수 있도록 지원합니다. 이제 AWS DMS를 사용하여 MongoDB 클러스터의 여러 데이터베이스를 그룹화하고 하나의 데이터베이스 마이그레이션 작업을 사용하여 마이그레이션할 수 있습니다. 가동 중지 시간을 최소화하면서 Amazon DocumentDB(MongoDB 호환)를 포함하여 AWS DMS 지원되는 모든 대상으로 마이그레이션할 수 있습니다.
MongoDB 또는 Amazon DocumentDB(MongoDB 호환)를 소스로 사용하는 자동 세그먼트화 지원	AWS DMS 는 이제 MongoDB 또는 Amazon DocumentDB를 소스로 사용하여 자동 분할을 지원합니다. 를 사용하면 MongoDB 또는 DocumentDB 클러스터 컬렉션을 자동으로 분할하도록 데이터베이스 마이그레이션 작업을 구성할 AWS DMS수 있습니다. 그런 다음 다운타임을 최소화하면서 Amazon DocumentDB를 포함하여 AWS DMS 지원되는 모든 대상에 세그먼트를 병렬로 마이그레이션할 수 있습니다.

새로운 기능 또는 향상된 기능	설명
Amazon Redshift 전체 로드 성능 개선	AWS DMS 이제는 전체 로드 중에 Amazon Redshift를 대상으로 사용할 때 병렬 스레드 사용을 지원합니다. 멀티스레드 전체 로드 작업 설정을 활용하면 AWS DMS 지원되는 모든 소스에서 Amazon Redshift로 초기 마이그레이션의 성능을 개선할 수 있습니다. AWS DMS 대상에 대한 자세한 내용은 섹션을 참조하세요 마이그레이션에 적합한 대상 .

AWS DMS 3.4.5에서 해결된 문제는 다음과 같습니다.

- PostgreSQL을 트랜잭션 동시성이 높은 소스로 사용할 때 재개 후 데이터가 누락되거나 중복될 수 있는 문제를 수정했습니다.
- pglogical 플러그인이 활성화된 상태에서 PostgreSQL을 소스로 사용할 때 관계 ID를 찾을 수 없습니다...라는 오류와 함께 데이터베이스 마이그레이션 태스크가 실패하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용하고 Oracle을 대상으로 사용할 때 VARCHAR 열이 제대로 복제되지 않는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 프라이머리 키가 테이블 정의의 첫 번째 열이 아닌 경우 삭제 작업이 제대로 캡처되지 않는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 데이터베이스 마이그레이션 태스크가 특수 메타데이터 설정에서 LOB 업데이트를 누락하는 문제를 수정했습니다.
- MySQL 버전 8을 소스로 사용할 때 전체 LOB 모드에서 TIMESTAMP 열이 DATETIME으로 처리되는 문제를 수정했습니다.
- MySQL 5.6.4 이상을 소스로 사용할 때 NULL DATETIME 레코드를 파싱하면 데이터베이스 마이그레이션 태스크가 실패하는 문제를 수정했습니다.
- Amazon Redshift를 병렬 적용의 대상으로 사용할 때 스레드 종료 중 오류가 발생한 후 데이터베이스 마이그레이션 태스크가 중단되는 문제를 수정했습니다.
- CDC 일괄 적용 중에 데이터베이스 마이그레이션 태스크와 Amazon Redshift 대상 엔드포인트의 연결이 끊길 때 데이터가 손실될 수 있는 문제를 수정했습니다.
- Amazon Redshift를 대상으로 사용할 때 ACCEPTINVCHARS를 호출하여 전체 로드 성능을 개선했습니다.
- Amazon Redshift를 대상으로 사용할 때 일대일 모드에서 병렬 적용 모드로 되돌리면 중복된 레코드가 복제되는 문제를 수정했습니다.

- Amazon S3를 대상으로 사용할 때 데이터베이스 마이그레이션 태스크가 Amazon S3 객체 소유권을 `cannedAclForObjects=bucket_owner_full_control`을 보유한 버킷 소유자로 전환하지 않는 문제를 수정했습니다.
- Oracle을 소스로 사용할 `additionalArchivedLogDestId` 때 ECA에서 여러 아카이브 대상을 지원 AWS DMS 하여 개선되었습니다.
- 전체 LOB 모드에서 LOB 열을 업데이트하는 동안 데이터베이스 마이그레이션 태스크가 `OCI_INVALID_HANDLE` 오류와 함께 실패하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 CDC 중에 `NVARCHAR2` 열이 제대로 마이그레이션되지 않는 문제를 수정했습니다.
- RDS for SQL Server를 소스로 사용할 `SafeguardPolicy` 때를 활성화 AWS DMS 하여 개선되었습니다.
- 비 RDS SQL Server 소스를 사용할 때 데이터베이스 마이그레이션 태스크가 `rdsadmin`에 대한 오류를 보고하는 문제를 수정했습니다.
- SQL Server를 소스로 사용할 때 파티션 설정에서 UUID를 프라이머리 키로 사용하면 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
- Db2 LUW를 소스로 사용할 때 데이터베이스 로그에서 필요한 LSN을 찾을 수 없으면 전체 로드 및 CDC 태스크가 실패할 수 있는 문제를 수정했습니다.
- MongoDB를 소스로 사용할 때 사용자 지정 CDC 타임스탬프를 지원 AWS DMS 하여 개선되었습니다.
- MongoDB를 소스로 사용할 때 `endSessions`에서 MongoDB 드라이버 오류가 발생하면 데이터베이스 마이그레이션 태스크가 중단되는 문제를 수정했습니다.
- DynamoDB를 대상으로 사용할 때가 기본이 아닌 필드를 업데이트 AWS DMS 하지 못하는 문제를 해결했습니다.
- 데이터 유효성 검사가 `CLOB` 및 `NCLOB` 열에 대해 거짓 긍정 불일치를 보고하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 공백 전용 레코드에서 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
- 파티션된 테이블을 자를 때 데이터베이스 마이그레이션 태스크가 충돌하는 문제를 수정했습니다.
- `awsdms_apply_exceptions` 제어 테이블을 생성할 때 데이터베이스 마이그레이션 태스크가 실패하는 문제를 수정했습니다.
- MySQL 버전 8을 사용할 때 `caching_sha2_password` 인증 플러그인 지원을 확대했습니다.

AWS Database Migration Service 3.4.4 릴리스 정보

다음 표에서는 AWS DMS 버전 3.4.4에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
Kafka를 대상으로 사용할 때 TLS 암호화 및 TLS 또는 SASL 인증 지원	AWS DMS 는 이제 Amazon MSK 및 온프레미스 Kafka 클러스터를 대상으로 사용하여 TLS 암호화와 TLS 또는 SASL 인증을 지원합니다. Kafka 엔드포인트에서 암호화 및 인증 사용에 대한 자세한 내용은 Transport Layer Security(TLS)을 사용하여 Kafka에 연결 섹션을 참조하세요.

AWS DMS 3.4.4에서 해결된 문제는 다음과 같습니다.

- Oracle 엔드포인트를 사용할 때 작업 실패에 대한 AWS DMS 로깅을 개선했습니다.
- Oracle Data Guard 장애 조치 후 Oracle 소스 엔드포인트가 역할을 전환할 때 AWS DMS 작업 실행이 계속 처리됩니다.
- Oracle 엔드포인트를 사용할 때 오류 처리가 ORA—12561을 복구 가능한 오류로 취급하도록 개선했습니다.
- Oracle을 소스로 사용할 때 EMPTY_BLOB() 및 EMPTY_CLOB() 열이 null로 마이그레이션되는 문제를 수정했습니다.
- SQL Server를 소스로 사용할 때 열 DDL 변경 추가 후 AWS DMS 작업이 레코드를 업데이트하지 못하는 문제를 수정했습니다.
- TIMESTAMP WITH TIME ZONE 데이터 형식을 지원하여 PostgreSQL 소스 마이그레이션을 개선했습니다.
- PostgreSQL을 대상으로 사용할 때 전체 로드 중에 afterConnectScript 설정이 작동하지 않는 문제를 수정했습니다.
- PostgreSQL 엔드포인트를 사용할 때 정밀도 및 소수점 자릿수가 없는 NUMERIC 날짜 형식을 더 잘 처리할 수 있는 새 mapUnboundedNumericAsString 설정을 도입했습니다.
- PostgreSQL을 소스로 사용할 때 AWS DMS 작업을 중지하고 재개한 후 “영향을 받는 행 0개”로 인해 작업이 실패하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 BC가 접미사를 사용하여 TIMESTAMP 데이터 형식을 마이그레이션 AWS DMS 하지 못하는 문제를 수정했습니다.

- PostgreSQL을 소스로 사용할 때가 ""infinity" TIMESTAMP 값을 마이그레이션 AWS DMS 하지 못하는 문제를 수정했습니다.
- csvNullValue 설정이 다른 값으로 설정된 상태에서 S3를 소스로 사용할 때 빈 문자열이 NULL로 처리되는 문제를 수정했습니다.
- S3를 대상으로 사용할 때 CDC를 사용한 전체 로드에서 timestampColumnName 추가 연결 속성을 CDC 중에 정렬할 수 있도록 개선했습니다.
- S3를 소스로 사용할 때 16진수 형식(예: BYTE, BINARY, BLOB)의 이진 데이터 형식 처리를 개선했습니다.
- S3를 대상으로 사용할 때 삭제된 레코드가 특수 문자로 마이그레이션되는 문제를 수정했습니다.
- Amazon DocumentDB(MongoDB 호환)를 대상으로 사용할 때 빈 키 값을 처리하는 문제를 수정했습니다.
- MongoDB 또는 Amazon DocumentDB(MongoDB 호환)를 소스로 사용할 때가 NumberDecimal 또는 Decimal128 열을 복제 AWS DMS 하지 못하는 문제를 해결했습니다.
- MongoDB 또는 Amazon DocumentDB(MongoDB 호환)를 소스로 사용할 때 장애 조치가 발생하면 CDC 태스크를 재시도할 수 있도록 문제를 수정했습니다.
- Kinesis, Kafka 또는를 대상으로 사용할 때 RAW 데이터 유형 값에 16진수 "0x" 접두사를 제거하는 옵션을 추가 OpenSearch 했습니다.
- Db2 LUW를 소스로 사용할 때 고정 길이 문자 열에서 유효성 검사가 실패하는 문제를 수정했습니다.
- 소스 데이터 형식 또는 대상 데이터 형식만 FLOAT 또는 DOUBLE인 경우 유효성 검사가 실패하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 NULL 문자에서 유효성 검사가 실패하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 XML 열에서 유효성 검사가 실패하는 문제를 수정했습니다.
- MySQL을 소스로 사용하여 복합 키에 null이 가능한 열이 있을 때 AWS DMS 작업이 충돌하는 문제를 수정했습니다.
- 가 SQL Server 소스 엔드포인트의 두 UNIQUEIDENTIFIER 열과 PostgreSQL 대상 엔드포인트의 UUID 열을 모두 검증 AWS DMS 하지 못하는 문제를 수정했습니다.
- CDC 태스크가 수정된 후 업데이트된 소스 테이블 정의를 사용하지 않는 문제를 수정했습니다.
- 잘못된 사용자 이름 또는 암호로 인한 작업 실패를 복구 가능한 오류로 처리하도록 AWS DMS 장애 조치를 개선했습니다.
- RDS for SQL Server를 소스로 사용할 때 LSNs 누락으로 인해 AWS DMS 작업이 실패하는 문제를 수정했습니다.

AWS Database Migration Service 3.4.3 릴리스 정보

다음 표에서는 AWS DMS 버전 3.4.3에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
새 Amazon DocumentDB 버전	이제 Amazon DocumentDB 버전 4.0이 소스로 지원됩니다.
새 MariaDB 버전	이제 MariaDB 버전 10.4가 소스 및 대상으로 지원됩니다.
AWS Secrets Manager 통합 지원	지원되는 엔드포인트의 데이터베이스 연결 세부 정보(사용자 자격 증명 정보)를 AWS Secrets Manager에 안전하게 저장할 수 있습니다. 그런 다음 엔드포인트를 생성하거나 수정할 AWS DMS 때 일반 텍스트 자격 증명 대신 해당 보안 암호를 제출할 수 있습니다. AWS DMS 그런 다음 보안 암호를 사용하여 엔드포인트 데이터베이스에 연결합니다. AWS DMS 엔드포인트의 보안 암호 생성에 대한 자세한 내용은 섹션을 참조하세요 보안 암호를 사용하여 AWS Database Migration Service 엔드포인트 액세스 .
C5 및 R5 복제 인스턴스용 대용량 옵션	이제 다음과 같이 더 큰 복제 인스턴스 크기를 생성할 수 있습니다. C5 크기는 최대 96개의 vCPU 및 192GiB 메모리까지, R5 크기는 최대 96개의 vCPU 및 768GiB 메모리까지 생성할 수 있습니다.
Amazon Redshift 성능 개선	AWS DMS 는 이제 Amazon Redshift를 대상으로 사용하여 지속적 복제의 성능을 개선할 때 병렬 적용을 지원합니다. 자세한 내용은 Amazon Redshift에 대한 멀티스레드 작업 설정 단원을 참조하십시오.

AWS DMS 3.4.3에서 해결된 문제는 다음과 같습니다.

- Db2 LUW를 소스로 사용할 때 지연된 이벤트의 커밋 타임스탬프가 '1970-01-01 00:00:00'이 되는 문제를 수정했습니다.
- SQL Server를 전체 LOB 모드의 소스로 사용할 때 NVARCHAR 열이 기본 키인 AWS DMS 태스크가 실패하는 문제를 수정했습니다.

- SQL Server를 소스로 사용할 때 캐시된 변경 사항 단계에서 레코드가 누락되는 문제를 수정했습니다.
- RDS for SQL Server를 소스로 사용할 때 AWS DMS 작업이 재개된 후 레코드를 건너뛰는 문제를 수정했습니다.
- AWS DMS ASSERTION 로깅 구성 요소가 SQL Server에 대한 대규모 로그를 생성하는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 CDC 단계에서 열 파싱 오버플로로 인해 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
- PostgreSQL을 대상으로 사용할 때 데이터 검증 중에 세분화 오류로 인해 AWS DMS 작업이 중단되는 문제를 수정했습니다.
- PostgreSQL을 소스 및 대상으로 사용할 때 CDC 중에 DOUBLE 데이터 형식에서 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용하고 Amazon Redshift를 대상으로 사용할 때 복사 명령으로 삽입한 레코드가 제대로 복제되지 않는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 캐시된 변경 사항 단계 중에 발생하는 데이터 손실 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 잠재적으로 데이터 손실 또는 레코드 중복을 일으킬 수 있는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 대/소문자가 혼합된 스키마가 pglogical을 사용하여 마이그레이션되지 못하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 마지막 실패 메시지에 ORA 오류가 포함되지 않는 문제를 수정했습니다.
- Oracle을 대상으로 사용할 때 AWS DMS 작업이 UPDATE 문을 빌드하지 못하는 문제를 수정했습니다.
- Oracle 12.2를 ASM 및 플러그형 데이터베이스 구성의 소스로 사용할 때 AWS DMS 작업이 데이터를 복제하지 않는 문제를 수정했습니다.
- S3를 소스로 사용할 때 RFC 4180을 준수하도록 따옴표를 보존하여 레코드 파싱을 개선했습니다.
- 전체 로드의 열을 CDC의 열로 정렬할 수 있도록 `timestampColumnName` 처리를 개선했습니다.
- 새 엔드포인트 설정을 도입하여 1MB보다 큰 LOB 요소가 있을 때 AWS DMS 작업이 실패하는 문제를 `MessageMaxBytes` 수정했습니다.
- Amazon Redshift를 대상으로 사용할 때 세분화 장애로 인해 AWS DMS 작업이 중단되는 문제를 수정했습니다.
- Amazon Redshift 테스트 연결에 대한 오류 로깅을 개선했습니다.

- 가 전체 로드 중에 MongoDB에서 DocumentDB로 모든 문서를 전송하지 않는 문제를 수정 AWS DMS 했습니다.
- 테이블 매핑 규칙에 포함된 테이블이 없을 때 AWS DMS 작업이 치명적인 오류를 보고하는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 AWS DMS 태스크를 다시 시작하기 전에 생성한 스키마 및 테이블이 대상에 복제되지 않는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 제외 규칙에서 와일드카드 이스케이프 []가 와일드카드 “_”를 이스케이프할 수 없는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 UNSIGNED BIGINT 데이터 형식의 열이 제대로 복제되지 않는 문제를 수정했습니다.

AWS Database Migration Service 3.4.2 릴리스 정보

다음 표에서는 AWS DMS 버전 3.4.2에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
인터넷 게이트웨이, NAT 디바이스, VPN 연결 또는 연결 없이 Amazon Virtual Private Cloud(Amazon VPC)를 AWS Database Migration Service(DMS)에 비공개로 AWS Direct Connect 연결할 수 있도록 지원합니다.	이제 생성한 VPC 인터페이스 엔드포인트를 통해 Amazon VPC에 연결하고 Amazon VPC AWS DMS 에서 액세스할 수 있습니다. 이 인터페이스 엔드포인트를 사용하면 Amazon 네트워크 인프라 내에서 AWS DMS 복제 인스턴스의 모든 네트워크 활동을 격리할 수 있습니다. AWS CLI 또는 SDK를 AWS DMS 사용하여에 대한 모든 API 호출에이 인터페이스 엔드포인트에 대한 참조를 포함하면 모든 AWS DMS 활동이 퍼블릭 인터넷에 표시되지 않습니다. 자세한 내용은 인프라 보안 AWS Database Migration Service 단원을 참조하십시오.
	Note 이 기능은 지원되는 모든 AWS DMS 엔진 버전을 사용하여 사용할 수 있습니다.

새로운 기능 또는 향상된 기능	설명
Amazon S3를 대상으로 사용하는 CDC 날짜 기반 폴더 파티셔닝	AWS DMS 는 이제 S3를 대상으로 사용하여 데이터를 복제할 때 날짜 기반 폴더 파티셔닝을 지원합니다. 자세한 내용은 날짜 기반 폴더 파티셔닝 사용 단원을 참조하십시오.

AWS DMS 3.4.2에서 해결된 문제는 다음과 같습니다.

- Amazon Redshift를 대상으로 사용하여 마이그레이션을 수행할 때 STATUPDATE 옵션을 추가했습니다.
- 새 설정을 도입하여 유효성 검사 태스크를 개선했습니다. ValidQueryCdcDelaySecond는 소스 및 대상 엔드포인트 모두에서 첫 번째 유효성 검사 쿼리를 지연시켜 마이그레이션 지연 시간이 높을 때 리소스 경합을 줄일 수 있습니다.
- 검증 작업을 시작하는 데 시간이 오래 AWS DMS 걸리는 문제를 수정했습니다.
- S3를 대상으로 사용하여 복제 태스크를 시작 또는 중지할 때 빈 레코드가 생성되는 문제를 수정했습니다.
- 전체 로드가 완료된 후 태스크가 중단되는 문제를 수정했습니다.
- S3를 소스로 사용할 때 소스 테이블에 데이터 오류가 있으면 태스크가 중단되는 문제를 수정했습니다.
- 시작하는 동안 소스 엔드포인트의 사용자 계정이 비활성화되어 있으면 태스크가 중단되는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 REPLICA IDENTITY FULL을 설정하면 태스크가 충돌하는 문제를 수정했습니다.
- PostgreSQL 소스에서 pglogical 플러그인을 사용하면 태스크가 트랜잭션을 누락하는 문제를 수정했습니다.
- Amazon Redshift를 대상으로 사용할 때가 압축된 소스 파일을 삭제하지 않는 문제를 수정 AWS DMS 했습니다.
- MySQL을 BIGINT UNSIGNED 데이터 형식의 소스 및 대상으로 사용할 때 유효성 검사 태스크가 거짓 부정을 보고하는 문제를 수정했습니다.
- SQL Server를 프라이머리 키 열이 CHAR 형식인 소스로 사용할 때 유효성 검사 태스크가 거짓 긍정을 보고하는 문제를 수정했습니다.
- AWS DMS 를 사용하여 S3를 대상으로 사용하는 복제 작업을 start-replication 시작할 때가 대상 객체를 지우지 않는 문제를 수정했습니다.

- Db2를 소스로 사용할 때 발생하는 데이터 유효성 검사와 관련된 몇 가지 문제를 수정했습니다.
- SQL Server를 VARCHAR 열이 프라이머리 키인 소스로 사용할 때 유효성 검사 태스크가 중단되는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 데이터 형식 TIMESTAMP WITH TIMEZONE에 대한 지원을 추가했습니다.

AWS Database Migration Service 3.4.1 베타 릴리스 정보

다음 표에서는 AWS DMS 버전 3.4.1 베타에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
새 MongoDB 버전	이제 MongoDB 버전 4.0이 소스로 지원됩니다.
SQL Server에 대한 TLS 1.2 지원	AWS DMS 는 이제 SQL Server 엔드포인트용 TLS 1.2를 지원합니다.

AWS DMS 3.4.1 베타에서 해결된 문제는 다음과 같습니다.

- Oracle 19c TDE 지원을 개선했습니다.
- Amazon Redshift를 대상으로 사용할 때 utf8mb4 문자 세트 및 ID 데이터 형식에 대한 지원을 개선했습니다.
- MySQL을 소스로 사용하고 바이너리 로그가 없는 경우 복제 태스크 실패 처리가 개선되었습니다.
- 다양한 데이터 형식 및 문자 세트에 대한 데이터 유효성 검사 지원을 개선했습니다.
- Kinesis 및 Kafka를 대상으로 사용할 때 새로운 엔드포인트 설정 IncludeNullAndEmpty를 통해 null 값 처리를 개선했습니다.
- Kafka를 대상으로 사용할 때 오류 로깅 및 처리를 개선했습니다.
- SQL Server를 소스로 사용할 때 DST 시간 오프셋을 개선했습니다.
- 복제 태스크가 Oracle용 기존 테이블을 대상으로 생성하려고 시도하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 데이터베이스 연결이 끊긴 후 복제 태스크가 중단되는 문제를 수정했습니다.
- AlwaysON 설정에서 SQL Server를 소스로 사용할 때 복제 태스크가 새 프라이머리 서버를 검색하고 다시 연결하지 못하는 문제를 수정했습니다.

- S3를 대상으로 사용할 때 특정 조건에서 복제 태스크가 "OP" 열의 "D"를 추가하지 않는 문제를 수정했습니다.

AWS Database Migration Service 3.4.0 베타 릴리스 정보

다음 표에서는 AWS DMS 버전 3.4.0에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
새 MySQL 엔진 버전	AWS DMS 이제는 트랜잭션 페이로드가 압축된 경우를 제외하고 MySQL 버전 8.0을 소스로 지원합니다.
MySQL에 대한 TLS 1.2 지원	AWS DMS 는 이제 MySQL 엔드포인트용 TLS 1.2를 지원합니다.
새 MariaDB 버전	AWS DMS 는 이제 MariaDB 버전 10.3.13을 소스로 지원합니다.
자체 관리형 Microsoft SQL Server 소스에 대한 비 SysAdmin 액세스	AWS DMS 는 이제 SysAdmin 사용자가 아닌 사용자가 온프레미스 및 EC2-hosted SQL Server 소스 엔드포인트에 액세스할 수 있도록 지원합니다.  Note 이 기능은 현재 베타 상태입니다. 사용해 보려면 AWS 지원 팀에 자세한 내용을 문의하세요.
CREATE TABLE AS을 사용하여 생성된 CDC 태스크 및 Oracle 소스 테이블	AWS DMS 는 이제 CREATE TABLE AS 문을 사용하여 생성된 Oracle 소스 테이블에 대해 실행되는 전체 로드 및 CDC 및 CDC 전용 작업을 모두 지원합니다.

AWS DMS 3.4.0에서 해결된 문제는 다음과 같습니다.

- 업그레이드 태스크 평가를 개선했습니다. 자세한 내용은 [작업에 대한 마이그레이션 전 평가 활성화 및 활용](#) 섹션을 참조하세요.
- float, real 및 double 데이터 형식에 대한 데이터 유효성 검사를 개선했습니다.

- “지정된 키가 존재하지 않습니다.” 오류를 더 잘 처리하여 대상으로서의 Amazon Redshift를 개선했습니다.
- Amazon OpenSearch Service(OpenSearch Service) 대상에 대해 ParallelApplyThreads, ParallelApplyBufferSize, ParallelApplyQueuesPerThread 등의 멀티스레드 CDC 로드 태스크 설정을 지원합니다.
- 복합 프라이머리 키 사용을 지원하여 대상으로서의 OpenSearch Service를 개선했습니다.
- PostgreSQL을 소스로 사용할 때 암호에 특수 문자가 있는 경우 테스트 연결이 실패하는 문제를 수정했습니다.
- SQL Server를 소스로 사용할 때 일부 VARCHAR 열이 잘리는 경우 발생하는 문제를 수정했습니다.
- Amazon RDS SQL Server를 소스로 사용할 때가 열린 트랜잭션을 닫지 않는 문제를 수정 AWS DMS 했습니다. 폴링 간격 파라미터를 잘못 설정하면 데이터가 손실될 수 있습니다. 권장 폴링 간격 값을 설정하는 방법에 대한 자세한 내용은 [Microsoft SQL Server 데이터베이스의 소스로 사용 AWS DMS](#) 섹션을 참조하세요.
- Oracle Standby를 소스로 사용할 때 Binary Reader를 사용하면 CDC 태스크가 예기치 않게 중지되는 문제를 수정했습니다.
- IBM DB2 for LUW에서 “숫자 리터럴 0은 해당 값이 범위를 벗어났으므로 유효하지 않습니다.”라는 메시지와 함께 태스크가 실패하는 문제를 수정했습니다.
- PostgreSQL에서 PostgreSQL로 마이그레이션할 때 PostgreSQL 소스에 새 열이 추가되고 해당 열이 소스에서 원래 생성된 데이터 형식과 다른 데이터 형식으로 생성되는 경우 발생하는 문제를 수정했습니다.
- MySQL 소스에서 binlog를 가져올 수 없을 때 마이그레이션 태스크가 예기치 않게 중단되는 문제를 수정했습니다.
- Oracle 대상에서 BatchApply 사용과 관련된 문제를 수정했습니다.
- MySQL 및 MariaDB에서 TIME 데이터 형식을 마이그레이션할 때 발생하는 문제를 수정했습니다.
- IBM DB2 LUW 소스에서 테이블에 프라이머리 키 또는 고유 키가 없는 경우 LOB가 포함된 테이블 마이그레이션이 실패하는 문제를 수정했습니다.

AWS Database Migration Service 3.3.4 릴리스 정보

AWS DMS 3.3.4에서 해결된 문제는 다음과 같습니다.

- PostgreSQL을 소스로 사용할 때 트랜잭션이 삭제되거나 중복되는 문제를 수정했습니다.
- 스키마 이름에 달러 기호(\$) 사용에 대한 지원을 개선했습니다.

- RDS SQL Server를 소스로 사용할 때 복제 인스턴스가 미결 트랜잭션을 닫지 않는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 암호에 특수 문자가 있는 경우 테스트 연결이 실패하는 문제를 수정했습니다.
- “지정된 키가 존재하지 않습니다.” 오류를 더 잘 처리하여 대상으로서의 Amazon Amazon Redshift를 개선했습니다.
- 다양한 데이터 형식 및 문자 세트에 대한 데이터 유효성 검사 지원을 개선했습니다.
- 복제 태스크가 Oracle용 기존 테이블을 대상으로 생성하려고 시도하는 문제를 수정했습니다.
- Amazon S3를 대상으로 사용할 때 특정 조건에서 복제 태스크가 "OP" 열의 "D"를 추가하지 않는 문제를 수정했습니다.

AWS Database Migration Service 3.3.3 릴리스 정보

다음 표에서는 AWS DMS 버전 3.3.3에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
새 PostgreSQL 버전	PostgreSQL 버전 12는 이제 소스 및 대상으로 지원됩니다.
Amazon OpenSearch Service를 대상으로 사용하는 복합 프라이머리 키 지원	AWS DMS 3.3.3부터 OpenSearch Service 대상에서 복합 기본 키 사용을 지원합니다.
Oracle 확장 데이터 형식 지원	이제 Oracle 소스 및 대상 모두에 대한 Oracle 확장 데이터 형식이 지원됩니다.
계정당 AWS DMS 리소스 수 증가	생성할 수 있는 AWS DMS 리소스 수에 대한 제한이 증가했습니다. 자세한 내용은 에 대한 할당량 AWS Database Migration Service 단원을 참조하십시오.

AWS DMS 3.3.3에서 해결된 문제는 다음과 같습니다.

- Amazon Kinesis에서 병렬 적용을 사용하는 특정 업데이트 문을 사용하여 태스크가 충돌하는 문제를 수정했습니다.
- Amazon S3를 대상으로 사용할 때 ALTER TABLE 문에서 태스크가 충돌하는 문제를 수정했습니다.

- Microsoft SQL Server를 소스로 사용할 때 다각형 열의 값이 잘리는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 JA16SJISTILDE 및 JA16EUCTILDE의 유니코드 변환기 관련 문제를 수정했습니다.
- MEDIUMTEXT 및 LONGTEXT 열이 MySQL에서 S3의 심표로 구분된 값(CSV) 형식으로 마이그레이션되지 않는 문제를 수정했습니다.
- Apache Parquet 출력을 사용할 때 부울 열이 잘못된 형식으로 변환되는 문제를 수정했습니다.
- Oracle에서 확장된 varchar 열 관련 문제를 수정했습니다.
- 특정 타임스탬프 조합으로 인해 데이터 유효성 검사 작업이 실패하는 문제를 수정했습니다.
- Sybase 데이터 정의 언어(DDL) 복제 관련 문제를 수정했습니다.
- Oracle RAC(실제 애플리케이션 클러스터) 소스가 Oracle Binary Reader와 충돌하는 문제를 수정했습니다.
- 스키마 이름의 대/소문자를 사용하는 Oracle 대상에 대한 유효성 검사 관련 문제를 수정했습니다.
- IBM DB2 버전 9.7 및 10의 유효성 검사 관련 문제를 수정했습니다.
- StopTaskCachedChangesApplied 및 StopTaskCachedChangesNotApplied가 활성화된 작업이 두 번 중지되지 않는 문제를 수정했습니다.

문서 기록

다음 표에서는 2018년 1월 이후 AWSDatabase Migration Service 사용 설명서의 중요한 변경 사항을 설명합니다.

RSS 피드를 구독하면 이 설명서에 대한 업데이트 알림을 받을 수 있습니다. AWS DMS버전 릴리스에 대한 자세한 내용은 섹션을 참조하세요 [AWS DMS 릴리스 정보](#).

변경 사항	설명	날짜
AWS DMS에 새 AWS관리형 역할에 대한 지원 추가	AWS DMSDMS가 S3 버킷을 생성하고 DMS 서버리스와 관련이 없는 복제 작업을 위해 마이그레이션 전 평가 결과를 해당 버킷에 넣을 수 AWSDMSServerlessServiceRolePolicy 있도록 업데이트되었습니다.	2025년 11월 5일
지원 종료 알림	지원 종료 공지: 2026년 5월 20일에 AWS는 AWS Database Migration ServiceFleet Advisor에 대한 지원을 종료합니다. 2026년 5월 20일 이후에는 더 이상 AWS DMSFleet Advisor 콘솔 또는 AWS DMSFleet Advisor 리소스에 액세스할 수 없습니다. 자세한 내용은 AWS DMS Fleet Advisor 지원 종료 를 참조하세요.	2025년 5월 20일
AWS DMSAWS서비스 연결 역할에 대한 지원 업데이트	AWS DMSdms:StartReplicationTaskAssessmentRun 가 마이그레이션 전 평가 실행을 지원AWSDMSServerlessServiceRolePolicy 하도	2025년 2월 14일

록 업데이트되었습니다. AWS DMS 또한 S3 버킷을 생성하고 마이그레이션 전 평가 결과를 해당 버킷에 넣도록 서버리스 서비스 연결 역할을 업데이트했습니다.

[AWS DMS에 새 AWS관리형 정책에 대한 지원 추가](#)

AWS DMS 복제 ModifyReplicationTask 작업을 수정하기 위해 작업을 호출하는 데 Serverless에 dms:ModifyReplicationTask AWS DMS 필요한가 추가되었습니다. 복제 인스턴스를 수정하기 위해 ModifyReplicationInstance 작업을 호출 dms:ModifyReplicationInstance 하는 데 AWS DMSServerless에 필요한가 AWS DMS 추가되었습니다.

2025년 1월 17일

[AWS DMS 향상된 모니터링 대시보드에 대한 지원 추가](#)

AWS DMS에는 이제 데이터 마이그레이션 작업 및 복제 인스턴스의 중요 지표에 대한 포괄적인 가시성을 제공하는 향상된 모니터링 대시보드가 포함되어 있습니다.

2024년 9월 19일

[AWS DMS에 RDS IBM DB2에 대한 지원이 대상으로 추가됨](#)

AWS DMS는 이제 Amazon RDS IBM DB2를 대상으로 사용할 수 있도록 지원합니다.

2023년 12월 4일

[AWS DMS는 Timestream에 대한 지원을 대상으로 추가했습니다.](#)

AWS DMS는 이제 Timestream을 대상으로 지원합니다.

2023년 11월 17일

[AWS DMS Redshift 대상 데이터 검증에 대한 지원 추가](#)

AWS DMS이제에서 Redshift 대상의 데이터 검증을 지원합니다.

2023년 11월 14일

[AWS DMS에 네 가지 새로운 엔드포인트 유형에 대한 지원이 추가되었습니다.](#)

AWS DMS는 이제 Microsoft Azure Database for PostgreSQL, Microsoft Azure Database for MySQL, OCI MySQL Heatwave 및 Google Cloud for PostgreSQL을 소스로 사용할 수 있도록 지원합니다.

2023년 10월 26일

[AWS DMS에 새 AWS서비스 연결 역할에 대한 지원 추가](#)

AWS DMS는 이제 Amazon CloudWatch에 지표 데이터 포인트를 게시AWSServiceRoleForDMSServerless 하는 등 AWS DMS가 사용자를 대신하여 리소스를 생성하고 관리할 수 있는 AWS서비스 연결 역할을 지원합니다.

2023년 5월 22일

[AWS DMS에 새 AWS관리형 정책에 대한 지원 추가](#)

AWS DMS는 이제 CloudWatch Logs에 서버리스 복제 로그를 게시할 수 있는 AWS관리형 정책을 지원합니다.

2023년 5월 22일

[AWS DMS에 새 AWS관리형 정책에 대한 지원 추가](#)

AWS DMS는 이제 지표 데이터 포인트를 Amazon CloudWatch에 게시할 수 있는 AWS관리형 정책을 지원합니다. 또한는 AWS관리형 정책에 대한 변경 내용 추적을 AWS DMS시작했습니다.

2023년 3월 6일

VPC 소스 및 대상 엔드포인트 지원

AWS DMS는 이제 Virtual Private Cloud(VPC) 엔드포인트를 소스 및 대상으로 지원합니다. AWS DMS이제 AWS 서비스에 대한 명시적으로 정의된 경로가 VPC에 정의된 경우 AWS DMSVPC 엔드포인트를 사용하여 모든 서비스에 연결할 수 있습니다.

2022년 6월 30일

SQL Server 읽기 전용 복제본을 소스로 지원

AWS DMS는 이제 SQL Server 읽기 전용 복제본을 소스로 지원합니다. 이제 AWS DMS를 사용하여 SQL Server 읽기 전용 복제본에서 AWS DMS지원되는 대상으로 실시간 마이그레이션을 수행할 수 있습니다.

2022년 6월 30일

전체 로드 전용 소스로서 IBM Db2 z/OS 데이터베이스 지원

AWS DMS는 이제 IBM Db2 z/OS 데이터베이스를 소스로 지원합니다. 이제 AWS DMS를 사용하여 Db2 메인프레임에서 AWS DMS지원되는 모든 대상으로 라이브 마이그레이션을 수행할 수 있습니다.

2022년 6월 30일

EventBridge DMS 이벤트 지원

AWS DMS는 DMS 이벤트에 EventBridge를 사용하여 이벤트 구독 관리를 지원합니다.

2022년 6월 30일

Babelfish를 대상으로 지원

AWS DMS는 이제 Babelfish를 대상으로 지원합니다. 이제 AWS DMS를 사용하여 가동 중지 시간을 최소화하면서 AWS DMS지원되는 모든 소스에서 Babelfish로 라이브 데이터를 마이그레이션할 수 있습니다.

2022년 6월 30일

[Aurora Serverless v2를 대상으로 지원](#)

AWS DMS는 이제 Aurora Serverless v2를 대상으로 지원합니다. 이제 AWS DMS를 사용하여 Aurora Serverless v2로 라이브 마이그레이션을 수행할 수 있습니다.

2022년 6월 30일

[시작하기 자습서](#)

AWS DMS 시작하기 자습서의 업데이트입니다. 이 자습서에서는 MySQL 데이터베이스를 소스로 사용하고 PostgreSQL 데이터베이스를 대상으로 사용합니다.

2021년 5월 20일

[Amazon Neptune을 대상으로 지원](#)

데이터 마이그레이션 대상으로 Amazon Neptune에 대한 지원을 추가했습니다.

2020년 6월 1일

[대상으로서 Apache Kafka에 대한 지원](#)

데이터 마이그레이션 대상으로 Apache Kafka에 대한 지원을 추가했습니다.

2020년 3월 20일

[보안 콘텐츠 업데이트됨](#)

고객 요청에 대한 응답으로 보안 콘텐츠를 업데이트하고 표준화했습니다.

2019년 12월 20일

[를 사용하여 마이그레이션 AWS Snowball Edge](#)

AWS Snowball Edge를 사용하여 대규모 데이터베이스를 마이그레이션하기 위한 지원이 추가되었습니다.

2019년 1월 24일

[대상으로서 Amazon DocumentDB\(MongoDB 호환\)에 대한 지원](#)

데이터 마이그레이션의 대상으로서 Amazon DocumentDB(MongoDB 호환)에 대한 지원이 추가되었습니다.

2019년 1월 9일

대상으로서 Amazon OpenSearch Service 및 Amazon Kinesis Data Streams에 대한 지원	데이터 마이그레이션의 대상으로서 OpenSearch Service 및 Kinesis Data Streams에 대한 지원이 추가되었습니다.	2018년 11월 15일
CDC 기본 시작 지원	변경 데이터 캡처(CDC) 사용 시 기본 시작점에 대한 지원이 추가되었습니다.	2018년 6월 28일
Db2 LUW 지원	데이터 마이그레이션 소스로서 IBM DB2 LUW에 대한 지원이 추가되었습니다.	2018년 4월 26일
대상으로서 SQL Server 지원	소스로서 Amazon RDS for Microsoft SQL Server에 대한 지원이 추가되었습니다.	2018년 2월 6일

AWS 용어집

최신 AWS 용어는 AWS 용어집 참조서의 [AWS 용어집](#)을 참조하십시오.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.