Amazon Neptune용 AWS Well-Architected 프레임워크 적용

AWS 권장 가이드

AWS 권장 가이드: Amazon Neptune용 AWS Well-Architected 프레임워크 적용

Table of Contents

Amazon Neptune용 AWS Well-Architected 프레임워크 적용

Amazon Web Services(기여자)

2026년 1월(문서 기록)

Amazon Neptune을 사용하여 Amazon Web Services(AWS)에서 그래프 기반 솔루션을 빌드할 수 있습니다. 이 가이드는 Neptune 배포를 계획할 때 AWS Well-Architected Framework 원칙을 적용하기 위한 권장 가이드를 제공합니다.

AWS Well-Architected Framework를 사용하면 다양한 애플리케이션 및 워크로드를 위한 안전하고 성능이 뛰어나며 복원력이 뛰어나고 효율적인 인프라를 구축할 수 있습니다. 또한 키텍처를 평가하고 확장 가능한 설계를 구현할 수 있는 일관된 접근 방식을 제공합니다.

AWS Well-Architected 프레임워크는 다음 6가지 원칙을 중심으로 구축되었습니다.

- 운영 우수성
- 보안
- 신뢰성
- 성능 효율성
- 비용 최적화
- 지속 가능성

이 가이드는 AWS Well-Architected Framework 설계 원칙 및 모범 사례의 정보와에 Neptune을 배포할 때 염두에 두어야 할 고려 사항을 제공합니다 AWS.

대상 독자

이 가이드는 AWS에서 그래프를 사용하는 솔루션을 설계하고 구현하는 데이터 엔지니어, 솔루션 아키텍트 및 데이터 분석가를 대상으로 합니다.

목표

이 가이드는 사용자와 조직이 다음을 수행하는 데 도움이 될 수 있습니다.

- 사용 사례 및 쿼리 패턴에 따라 지원되는 배포 옵션 및 쿼리 언어 중에서 선택합니다.

- 복원력과 보안을 개선하는 데 도움이 되는 AWS Well-Architected 설계 패턴을 따릅니다.
- 최적의 성능과 비용 절감을 위해 쿼리를 설계합니다.
- 프로덕션 환경에서 Neptune 클러스터를 관리할 때 운영 효율성을 높이는 방법에 대해 알아봅니다.

운영 우수성 요소

AWS Well-Architected Framework의 운영 우수성 원칙은 시스템을 실행 및 모니터링하고 프로세스와 절차를 지속적으로 개선하는 데 중점을 둡니다. 여기에는 효과적인 개발 및 워크로드 실행을 지원하고, 작업에 대한 인사이트를 얻으며, 지원 프로세스 및 절차를 지속적으로 개선하여 비즈니스 가치를 전달할 수 있는 능력이 포함됩니다. 사람의 개입 없이 대부분의 문제를 감지하고 해결하는 자가 복구 워크로드를 통해 운영 복잡성을 줄일 수 있습니다. 이 섹션에 설명된 모범 사례를 따르면 이 목표를 달성할 수 있습니다. Amazon Neptune 지표, API 및 메커니즘을 사용하여 워크로드가 예상 동작과 다를 때 적절하게 대응합니다.

운영 우수성 원칙에 대한 이 논의는 다음 핵심 영역에 중점을 둡니다.

- 코드형 인프라(IaC)
- 변경 관리
- 복원력 전략
- 인시던트 관리
- 규정 준수를 위한 감사 보고
- 로깅 및 모니터링

IaC 접근 방식을 사용하여 배포 자동화

IaC를 사용하여 Neptune에서 배포를 자동화하는 모범 사례로 다음이 포함됩니다.

- 가능하면 코드형 인프라(IaC)를 적용하여 Neptune 클러스터를 배포합니다. 일관된 환경 구성을 위해 AWS CloudFormation 템플릿, AWS Cloud Development Kit (AWS CDK) 또는 HashiCorp Terraform을 사용하여 클러스터에 필요한 모든 리소스를 생성합니다.
- 인스턴스 크기 조정, 읽기 복제본 추가 또는 제거, 가능하면 글로벌 테이블에서 수동 장애 조치 수행과 같은 Neptune 운영 절차를 자동화합니다.
- 연결 문자열을 클라이언트 외부에 저장합니다. 추출, 전환, 적재(ETL) 프로세스를 사용하여 블루/그린 배포 전략, 재해 복구(DR) 및 가동 중지 시간이 거의 없는 새 클러스터로의 마이그레이션을 용이하게 합니다. 연결 문자열은 AWS Secrets Manager, Amazon DynamoDB 또는 동적으로 변경할 수 있는 모든 위치에 저장할 수 있습니다.
- 태그를 사용하여 Neptune 리소스에 메타데이터를 추가하고 태그를 기반으로 사용량을 추적합니다. 자세한 내용은 Tagging Amazon Neptune Resources를 참조하세요.

되돌릴 수 있는 소규모 변경 자주 적용

다음 권장 사항은 복잡성을 최소화하고 워크로드 중단 가능성을 줄이기 위해 작고 되돌릴 수 있는 변경 사항에 중점을 둡니다.

- GitHub 또는 GitLab과 같은 소스 제어 서비스에 IaC 템플릿 및 스크립트를 저장합니다.

> **Important**
>
> 소스 제어에 AWS 자격 증명을 저장하지 마십시오.

- IaC 배포에서 AWS CodePipeline 또는 AWS CodeBuild와 같은 지속적 통합 및 지속적 전송(CI/CD) 서비스를 사용해야 합니다. 이러한 서비스는 프로덕션 Amazon Neptune 클러스터에 영향을 미치기 전에 임시 Neptune 클러스터가 포함된 비프로덕션 환경에서 코드를 컴파일, 테스트 및 배포합니다.
- 인프라 및 애플리케이션 쿼리를 프로덕션에 배포하기 전에 더 하위 환경에서 테스트합니다. 그러면 중단 가능성을 최소화하고 워크로드 및 규모에 맞게 원활하게 작동할 수 있습니다.

실패 예상

자가 복구 인프라는 장애를 예상하고 개입 없이 문제를 해결하려고 시도하며 운영 우수성을 보여줍니다. 다음 권장 사항은 Neptune을 사용하여 이러한 성숙도를 달성하는 데 도움이 됩니다.

- Amazon CloudWatch 지표를 사용하여 DB 인스턴스의 CPU 및 메모리 사용량을 모니터링하고 사용 패턴을 이해하는 모니터링 계획을 생성합니다. 애플리케이션 로그에 있는 주요 지표 및 Neptune 클라이언트 응답에 대한 CloudWatch 대시보드 및 경보를 생성합니다. CPU 사용률이 높거나 낮은 지표에 대한 자세한 내용은 Neptune 설명서의 Using CloudWatch to monitor DB instance performance in Neptune을 참조하세요.

  쿼리에 out-of-memory 예외가 자주 발생하는 경우 쿼리가 통과하는 총 노드 수를 줄이거나 RAM-to-CPU 비율이 높은 X2 패밀리의 인스턴스를 사용해 보세요.
- 알림을 설정하여 Neptune 클러스터의 상태를 모니터링합니다. 예를 들어 `BufferCacheHitRatio`는 지속적으로 높아야 하지만(99.9% 초과) `MainRequestQueuePendingRequests`는 지속적으로 낮아야 합니다(이상적으로는 0이지만 요구 사항 및 지연 시간 허용치에 따라 다름).

- 읽기 복제본을 사용하여 Neptune 내에서 고가용성을 달성하는 것이 좋습니다. 장애 조치 이벤트 중에 항상 읽기 쿼리를 지원할 수 있도록 인스턴스를 상시 가동하려면 라이터 인스턴스와 다른 가용 영역에 읽기 복제본이 두 개 이상 있어야 합니다.
- 사용률 지표를 기반으로 읽기 복제본의 크기를 자동으로 조정합니다. 자세한 내용은 Auto-scaling the number of replicas in an Amazon Neptune DB cluster를 참조하세요.
- DB 인스턴스의 장애 조치를 테스트하여 사용 사례 절차에 소요되는 시간을 파악하십시오.
- 애플리케이션에 완전한 AWS 리전 중단이 지속되어야 하는 경우 글로벌 데이터베이스를 DR 계획의 일부로 사용하는 것이 좋습니다.

모든 운영 장애로부터 학습

자가 복구 인프라는 드문 문제가 발생하거나 응답이 원하는 만큼 효과적이지 않을 때 반복으로 발전하는 장기적인 노력을 보여줍니다. 다음 사례를 채택하면 다음과 같은 목표에 집중합니다.

- 모든 장애로부터 학습하여 개선을 주도합니다.
- 조직과 여러 팀에서 학습한 내용을 공유합니다. 조직 내 여러 팀이 Neptune을 사용하는 경우 공통 채팅룸 또는 사용자 그룹을 생성하여 학습 내용과 모범 사례를 공유합니다.

로깅 기능을 사용하여 무단 또는 비정상적인 활동 모니터링

비정상적인 성능 및 활동 패턴을 관찰하려면 Amazon CloudWatch Logs에 로그를 저장합니다. 다음 모범 사례를 고려하세요.

- 느린 쿼리 로깅을 비활성화합니다. 로그를 정기적으로 검토하고 특정 쿼리가 느린 이유를 진단합니다. Gremlin, SPARQL 또는 openCypher에 대한 Neptune 설명 및 프로파일 엔드포인트를 사용하여 이러한 쿼리가 느린 이유를 파악할 수 있습니다.
- Neptune 감사 로그를 활성화하고 로그에 무단 액세스 또는 이상이 있는지 정기적으로 검토합니다.
- 느린 쿼리 로깅 또는 감사 로깅을 사용하는 경우 CloudWatch Logs에 대한 게시를 활성화합니다. 이렇게 하면 인스턴스의 디스크 공간 부족을 방지할 수 있습니다. Neptune 인스턴스는 로그 스토리지 용량을 제한하며 로그 공간이 초과되면 이전 로그 파일을 덮어씁니다. CloudWatch Logs는 로그의 장기 보존을 지원합니다. CloudWatch Logs의 향상된 모니터링 기능은 로그를 쿼리하고 문제를 진단하는 기능을 개선합니다.
- 감사 로그에 대한 더 나은 분석 도구를 용이하게 하기 위해 CloudWatch Logs의 로그 그룹에 감사 로그 데이터를 게시하도록 Neptune DB 클러스터를 구성할 수 있습니다. CloudWatch Logs로 로그 데이터에 대한 실시간 분석을 수행하고 CloudWatch를 사용하여 경보를 생성하고 지표를 보고

CloudWatch Logs를 사용하여 로그 레코드를 내구성이 높은 스토리지에 저장할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs에 Neptune Logs 게시를 참조하세요.

- Neptune은 AWS CloudTrail을 사용하여 컨트롤 플레인 작업의 로깅을 지원합니다. 자세한 내용은 를 사용하여 Amazon Neptune API 호출 로깅을 참조하세요 AWS CloudTrail.

보안 요소

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 모델은 이 사항을 클라우드 내 보안 및 클라우드의 보안으로 설명합니다.

- 클라우드 보안 - AWS 서비스 에서 실행되는 인프라를 보호할 AWS 책임이 있습니다 AWS 클라우드. AWS 또한는 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사자는 AWS 규정 준수 프로그램의 일환으로 AWS 보안의 효과를 정기적으로 테스트하고 확인합니다. Amazon Neptune에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램별 범위 내AWS 서비스를 참조하세요.
- 클라우드의 보안 - 사용자의 책임은 AWS 서비스 사용하는에 따라 결정됩니다. 또한 귀하는 데이터의 민감도, 회사 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS Shared Responsibility Model and GDPR 블로그 게시물을 참조하세요.

보안 원칙은 Neptune을 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제에서는 보안 및 규정 준수 목표를 충족하도록 Neptune을 구성하는 방법을 보여줍니다. 또한 Neptune 리소스를 모니터링하고 보호하는 데 도움이 AWS 서비스 되는 다른를 사용하는 방법을 알아봅니다.

보안 원칙에는 다음과 같은 주요 핵심 영역이 포함됩니다.

- 데이터 보안
- 네트워크 보안
- 인증 및 권한 부여

데이터 보안 구현

데이터 유출 및 위반은 고객을 위험에 빠뜨리고 회사에 상당한 부정적인 영향을 미칠 수 있습니다. 다음 모범 사례는 우발적이고 악의적인 노출로부터 고객 데이터를 보호하는 데 도움이 됩니다.

- 클러스터 이름, 태그, 파라미터 그룹, AWS Identity and Access Management (IAM) 역할 및 기타 메타데이터에는 기밀 또는 민감한 정보가 포함되어서는 안 됩니다. 해당 데이터가 결제 또는 진단 로그에 표시될 수 있기 때문입니다.
- Neptune에 데이터로 저장된 외부 서버에 대한 URI 또는 링크에는 요청을 검증하기 위한 자격 증명 정보가 포함되어서는 안 됩니다.
- Neptune 암호화 인스턴스는 기본 스토리지에 대한 무단 액세스로부터 데이터의 보안을 유지할 수 있도록 지원함으로써 데이터 보호 추가 계층을 제공합니다. 클라우드에 배포된 애플리케이션의 데이터 보안을 향상하기 위해 Neptune 암호화를 사용할 수 있습니다. 또한 이를 통해 저장 데이터의 암호화에 대한 규정 준수 요구 사항을 충족할 수 있습니다.

  새 Neptune DB 인스턴스에 대한 암호화를 활성화하려면 Neptune 콘솔의 암호화 활성화 섹션에서 예((기본적으로 선택됨)를 선택하거나 CloudFormation에서 AWS::Neptune::DBCluster::StorageEncrypted 속성을 설정합니다. 암호화가 활성화된 경우 Neptune은 기본적으로 Amazon Relational Database Service(Amazon RDS) AWS 관리형 키를 사용하거나 고객 관리형 키를 생성할 수 있습니다. Neptune DB 인스턴스 생성에 대한 자세한 내용은 Creating a new Neptune DB cluster를 참조하세요. 자세한 내용은 Encrypting Neptune Resources at Rest를 참조하세요. 자동화된 스냅샷 및 수동 스냅샷은 Neptune 클러스터에 대해 선택한 것과 동일한 암호화를 사용합니다.
- SPARQL 및 openCypher 언어를 사용하는 경우 적절한 입력 검증 및 파라미터화 기법을 사용하여 SQL 인젝션 및 기타 형태의 공격을 방지합니다. 사용자가 제공한 입력과 문자열 연결을 사용하는 쿼리를 구성하지 마세요. 파라미터화된 쿼리 또는 준비된 명령문을 사용하여 입력 파라미터를 그래프 데이터베이스에 안전하게 전달합니다. 자세한 내용은 Examples of openCypher parameterized queries 및 SPARQL Injection Defence를 참조하세요.
- Gremlin 언어의 경우 잠재적인 인젝션 문제를 방지하려면 문자열 기반 Gremlin 스크립트를 직접 전달하는 대신 Gremlin 언어 변형을 사용합니다.

네트워크 보안

Amazon Neptune DB 클러스터는 AWS의 가상 프라이빗 클라우드(VPC)에서만 생성할 수 있습니다. Neptune 1.4.6.0까지 Neptune DB 클러스터의 엔드포인트는 해당 VPC 내에서만 액세스할 수 있었습니다. Neptune 1.4.6.0 이상부터 Neptune 인스턴스는 인터넷을 통해 공개적으로 액세스할 수 있도록 구성할 수 있습니다. 개발자를 위해 Neptune에 대한 간소화된 액세스를 활성화하려면 비프로덕션 환경에서만이 기능을 사용하는 것이 가장 좋습니다(퍼블릭 액세스 가능성을 활성화하려면 항상에서 IAM 인증이 필요함). 퍼블릭 액세스가 활성화된 경우 데이터베이스 포트에 대한 인바운드 보안 그룹 규칙을 알려진 IP 주소 트래픽으로만 설정하는 것이 좋습니다. 프로덕션 환경 또는 민감한 데이터가 포함된 클러스터에서는 퍼블릭 액세스를 허용하지 않고 Neptune DB 클러스터가 위치한 VPC에 대한 액세스를

제한하여 Neptune 데이터를 보호합니다. 자세한 내용은 Connecting to your Amazon Neptune graph를 참조하세요.

전송 중 데이터를 보호하기 위해 Neptune은 보안 프로토콜 및 암호를 사용하여 HTTPS를 통해 모든 인스턴스 또는 클러스터 엔드포인트에 SSL 연결을 적용합니다. Neptune은 Neptune DB 인스턴스에 대한 SSL 인증서를 제공합니다. Neptune SSL 인증서는 클러스터 엔드포인트, 리더 엔드포인트 및 인스턴스 엔드포인트 호스트 이름만 지원합니다.

로드 밸런서 또는 프록시 서버(예: HAProxy)를 사용하는 경우 SSL 종료를 사용하고 프록시 서버에 자체 SSL 인증서가 있어야 합니다. 제공된 SSL 인증서가 프록시 서버 호스트 이름과 일치하지 않으므로 SSL 패스스루가 작동하지 않습니다. SSL을 사용하여 Neptune 엔드포인트에 연결하는 방법에 대한 자세한 내용은 Using the HTTP REST endpoint to connect to a Neptune DB instance를 참조하세요.

인증 및 권한 부여 구현

Neptune DB 클러스터 및 DB 인스턴스에서 Neptune 관리 작업을 수행할 수 있는 사용자를 제어하려면 IAM 데이터베이스 인증을 활성화하고 IAM 자격 증명을 사용합니다. IAM 자격 증명을 사용하여 AWS 에 연결할 때, IAM 역할은 Neptune 관리 작업을 수행하는 데 필요한 권한을 부여하는 IAM 정책을 보유하고 있어야 합니다. 태스크를 완료하는 데 필요한 권한만 부여하는 최소 권한 원칙을 따라야 합니다. 자세한 내용은 Using different kinds of IAM policies for controlling access to Neptune 및 IAM Authentication Using Temporary Credentials를 참조하세요.

Neptune 클러스터에 연결하고 데이터를 쿼리할 수 있는 사용자를 제어하려면 IAM을 사용하여 Neptune DB 인스턴스 또는 DB 클러스터에 인증할 수 있습니다. Neptune DB 클러스터에서 IAM 인증을 활성화하는 경우 DB 클러스터에 액세스하는 모든 사용자는 먼저 인증을 받아야 합니다. IAM 인증을 활성화하는 단계에 대한 자세한 내용은 Enabling IAM database authentication in Neptune을 참조하세요.

IAM 데이터베이스 인증이 활성화되어 있으면 AWS Signature Version 4를 사용하여 각 요청에 서명해야 합니다. IAM 인증이 활성화된 모든 Neptune 엔드포인트에 서명된 요청을 보내는 방법을 이해하려면 Connecting and Signing with AWS Signature Version 4를 참조하세요. awscurl과 같은 많은 라이브러리와 도구는 이미 AWS 서명 버전 4를 지원합니다.

다른와 상호 작용하기 위해 AWS 서비스 Amazon Neptune은 IAM 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 Neptune에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 Neptune에서 사전 정의하며 서비스에서 다른 AWS 서비스 자동으로 직접 호출하기 위해 필요한 모든 권한을 포함합니다. 자세한 내용은 Using Service-Linked Roles for Neptune을 참조하세요.

신뢰성 요소

신뢰성 원칙은 워크로드가 의도한 기능을 예상대로 올바르고 일관되게 수행할 수 있는 능력을 포함합니다. 여기에는 전체 수명 주기에 걸쳐 워크로드를 운영 및 테스트할 수 있는 기능이 포함됩니다.

신뢰할 수 있는 워크로드는 소프트웨어와 인프라에 대한 사전 설계 결정에서 시작됩니다. 아키텍처 선택은 모든 AWS Well-Architected 원칙에서 워크로드 동작에 영향을 미칩니다. 신뢰성을 달성하려면 특정 패턴을 따라야 합니다.

신뢰성 원칙은 다음 핵심 영역에 중점을 둡니다.

- 서비스 할당량 및 배포 패턴을 포함한 워크로드 아키텍처
- 변경 관리
- 장애 관리

Neptune 서비스 할당량 이해

중국 및 GovCloud(할당량이 64TiB임)를 제외한 지원되는 모든 AWS 리전 에서 Neptune 클러스터 볼륨은 최대 128테라바이트(TiB) 크기까지 늘어날 수 있습니다.

128TiB 할당량은 그래프에 약 2,000~4,000억 개의 객체를 저장하기에 충분합니다. 레이블이 지정된 속성 그래프(LPG)에서 객체는 노드, 엣지 또는 노드나 엣지의 속성입니다. 리소스 설명 프레임워크(RDF) 그래프에서 객체는 쿼드입니다.

Neptune Serverless 클러스터의 경우 Neptune 용량 단위(NCU)의 최소 및 최대 수를 모두 설정합니다. 각 NCU는 2기가바이트(GiB)의 메모리와 관련 vCPU 및 네트워킹으로 구성됩니다. 최소 및 최대 NCU 값이 클러스터의 모든 서버리스 인스턴스에 적용됩니다. 설정할 수 있는 최댓값은 128.0NCU이고, 가장 낮은 최솟값은 1.0NCU입니다. Amazon CloudWatch 지표 `ServerlessDatabaseCapacity` 및 `NCUUtilization`을 관찰하여 일반적으로 실행되는 범위를 캡처하고 해당 범위 내에서 원치 않는 동작이나 비용을 상호 연관시켜 애플리케이션에 가장 적합한 NCU 범위를 최적화합니다. 많은 워크로드에서 1.0 NCU는 시작점보다 너무 낮으며 일정 기간 동안 활동이 없으면 신뢰할 수 없는 동작이 발생합니다. 워크로드가 충분히 빠르게 조정되지 않는 경우 확장하는 동안 초기 급증에 충분한 처리를 제공하도록 최소 NCU를 늘립니다.

각 AWS 계정 에는 생성할 수 있는 데이터베이스 리소스 수에 대한 각 리전의 할당량이 있습니다. 이러한 리소스에는 DB 인스턴스 및 DB 클러스터가 포함됩니다. 리소스에 대한 제한에 도달하면 해당 리소스 생성을 위한 추가 호출이 예외와 함께 실패합니다. 일부 할당량은 요청에 따라 늘

릴 수 있는 소프트 할당량입니다. Amazon Neptune과 Amazon RDS, Amazon Aurora 및 Amazon DocumentDB(MongoDB 호환) 사이에서 공유되는 할당량 목록과 사용 가능한 경우 할당량 증가를 요청하는 링크는 Amazon RDS의 할당량을 참조하세요.

Neptune 배포 패턴 이해

Neptune DB 클러스터에는 기본 DB 인스턴스 1개와 최대 15개의 Neptune 복제본이 있습니다. 기본 DB 인스턴스는 읽기 및 쓰기 작업을 지원하고, 클러스터 볼륨에 대한 모든 데이터 수정을 수행합니다. Neptune 복제본은 기본 DB 인스턴스와 동일한 스토리지 볼륨에 연결되며 읽기 작업만 지원합니다. Neptune 복제본은 기본 DB 인스턴스에서 읽기 워크로드를 오프로드할 수 있습니다.

고가용성을 달성하려면 읽기 복제본을 사용합니다. 읽기 복제본이 기본 인스턴스의 장애 조치 대상 역할을 하기 때문에 여러 가용 영역에서 하나 이상의 읽기 복제본 인스턴스를 사용 가능하면 가용성이 향상될 수 있습니다. 즉, 라이터 인스턴스에서 장애가 발생하면 Neptune은 읽기 복제본 인스턴스를 기본 인스턴스로 승격합니다. 이 경우 승격된 인스턴스가 재부팅되는 동안 잠시 중단(보통 30초 미만)이 발생하며, 이 동안 기본 인스턴스에 대한 읽기 및 쓰기 요청이 예외적으로 실패합니다. 최고의 신뢰성을 위해 서로 다른 가용 영역에 있는 두 개의 읽기 복제본을 고려합니다. 가용 영역 1의 기본 인스턴스가 오프라인 상태가 되면 가용 영역 2의 인스턴스가 기본 인스턴스로 승격되지만 이 경우 쿼리를 처리할 수 없습니다. 따라서 전환 중에 읽기 쿼리를 처리하려면 가용 영역 3의 인스턴스가 필요합니다.

Neptune Serverless를 사용하는 경우 모든 가용 영역의 리더 및 라이터 인스턴스는 데이터베이스 로드에 따라 서로 독립적으로 스케일 업 및 스케일 다운됩니다. 리더 인스턴스의 승격 계층을 0 또는 1로 설정하여 라이터 인스턴스의 용량과 함께 스케일 업 및 스케일 다운할 수 있습니다. 이렇게 하면 언제든지 현재 워크로드를 인계받을 수 있습니다.

애플리케이션에 전 세계 공간이 있거나 다중 리전 장애 조치가 필요한 경우 Neptune 글로벌 데이터베이스를 사용하는 것이 좋습니다. Amazon Neptune 글로벌 데이터베이스는 여러 개에 걸쳐 AWS 리전 있으므로 지연 시간이 짧은 전역 읽기를 활성화하고 드물게 중단이 전체에 영향을 미치는 경우 빠른 복구를 제공합니다 AWS 리전. Neptune 글로벌 데이터베이스는 한 리전의 기본 DB 클러스터와 다른 리전의 최대 5개의 보조 DB 클러스터로 구성됩니다.

Neptune 클러스터 관리 및 규모 조정

Neptune 오토 스케일링을 사용하여 CPU 사용률 임계치에 따라 연결성 및 워크로드 요구 사항을 충족하도록 DB 클러스터의 Neptune 복제본 수를 자동으로 조정할 수 있습니다. 오토 스케일링을 사용하면 Neptune DB 클러스터가 갑작스러운 워크로드 증가를 처리할 수 있습니다. 워크로드가 감소하면 오토 스케일링은 불필요한 복제본을 제거하여 미사용 용량에 대한 비용을 지불하지 않도록 합니다. 새 인스

턴스 시작에는 최대 15분이 걸릴 수 있으므로 오토 스케일링만으로는 수요의 급격한 변화를 위한 충분한 솔루션이 아닙니다.

오토 스케일링은 이미 기본 라이터 인스턴스와 하나 이상의 읽기 복제본 인스턴스가 있는 Neptune DB 클러스터에서만 사용할 수 있습니다(see Amazon Neptune DB Clusters and Instances 참조). 또한 클러스터의 모든 읽기 복제본 인스턴스는 사용 가능한 상태여야 합니다. 읽기 복제본이 사용 가능하지 않은 상태인 경우 Neptune 오토 스케일링은 클러스터의 모든 읽기 복제본을 사용할 수 있을 때까지 아무 작업도 수행하지 않습니다.

수요가 빠르게 변하는 경우 서버리스 인스턴스를 사용하는 방법을 고려합니다. 서버리스 인스턴스는 단기간에 수직적 스케일링을 수행할 수 있는 반면, 오토 스케일링은 장기간에 수평적 스케일링을 수행합니다. 이 구성은 서버리스 인스턴스에서 수직적 스케일링을 수행하는 반면 오토 스케일링은 새 읽기 복제본을 인스턴스화하여 단일 서버리스 인스턴스의 최대 용량을 초과하는 워크로드를 처리하기 때문에 최적의 확장성을 제공합니다. Amazon Neptune Serverless의 용량 조정에 대한 자세한 내용은 Capacity scaling in a Neptune Serverless DB cluster를 참조하세요.

예측 가능한 시간에 조정 요구 사항을 변경해야 하는 경우 최소 인스턴스, 최대 인스턴스 및 임계치에 대한 변경을 예약하여 이러한 전환 요구 사항을 더 잘 처리할 수 있습니다. 필요할 때 해당 인스턴스가 온라인 상태가 될 수 있도록 스케일 아웃 이벤트를 최소 15분 전에 예약해야 합니다.

DB 파라미터 그룹에서 파라미터를 사용하여 Amazon Neptune에서 데이터베이스 구성을 관리합니다. 파라미터 그룹은 하나 이상의 DB 인스턴스에 적용되는 엔진 구성 값의 컨테이너 역할을 합니다. 파라미터 그룹에서 클러스터 파라미터를 수정할 때 정적 파라미터와 동적 파라미터의 차이와 적용 방법 및 시기를 이해합니다. 상태 엔드포인트를 사용하여 현재 적용된 구성을 확인합니다.

백업 및 장애 조치 이벤트 관리

Neptune은 클러스터 볼륨을 자동으로 백업한 후 백업 보존 기간 동안 백업된 데이터를 유지합니다. Neptune 백업은 연속적으로 또는 증분식으로 이루어지기 때문에 백업 보존 기간 내에 어떤 시점으로든 신속하게 복구가 가능합니다. DB 클러스터를 생성 또는 수정할 때 1일에서 35일 사이의 백업 보존 기간을 지정할 수 있습니다.

백업 보존 기간을 넘겨서 백업을 유지하려는 경우 클러스터 볼륨에서 데이터 스냅샷을 생성할 수도 있습니다. 스냅샷을 저장하면 Neptune 표준 스토리지 비용이 발생합니다.

DB 클러스터의 Amazon Neptune 스냅샷을 생성하면 Neptune은 개별 인스턴스가 아닌 모든 데이터를 백업하여 클러스터의 스토리지 볼륨 스냅샷을 생성합니다. 이 DB 클러스터 스냅샷에서 복원하여 추후 새로운 DB 클러스터를 생성할 수 있습니다. DB 클러스터를 복원할 때 복원 원본으로 사용할 DB 클러스터 스냅샷의 이름을 제공한 다음, 이 복원 작업에서 생성되는 새 DB 클러스터의 이름을 제공합니다.

시스템이 장애 조치 이벤트에 어떻게 응답하는지 테스트합니다. Neptune API를 사용하여 장애 조치 이벤트를 강제로 실행합니다. 장애 조치로 재부팅은 DB 인스턴스 결함을 시뮬레이션하여 테스트하거나, 장애 조치 이후 원래 가용 영역으로 작업을 복구할 때 유용한 기능입니다. 다중 AZ 배포에 대한 자세한 내용은 다중 AZ 배포 구성 및 관리를 참조하세요. DB 라이터 인스턴스를 재부팅하면 예비 복제본으로 장애 조치가 발생합니다. Neptune 복제본을 재부팅하면 장애 조치가 시작되지 않습니다.

신뢰성을 위해 클라이언트를 설계합니다. 장애 조치 이벤트 중에 동작을 테스트합니다. 지수 백오프 로직을 사용하여 클라이언트에서 재시도 로직을 구현합니다. 이 로직을 구현하는 코드 예제는 설명서의 AWS Lambda Amazon Neptune 함수 예제에서 확인할 수 있습니다.

여러 데이터베이스 엔진에 적용되는 일반적인 백업 요구 사항이 있는 경우 AWS Backup을 사용하는 방법을 고려합니다.

성능 효율성 요소

AWS Well-Architected Framework의 성능 효율성 원칙은 데이터를 수집하거나 쿼리하는 동안 성능을 최적화하는 방법에 중점을 둡니다. 성능 최적화는 다음과 같은 점진적이고 지속적인 프로세스입니다.

- 비즈니스 요구 사항 확인
- 워크로드 성능 측정
- 성능이 낮은 구성 요소 식별
- 비즈니스 요구 사항에 맞게 구성 요소 조정

성능 효율성 원칙은 사용할 올바른 그래프 데이터 모델 및 쿼리 언어를 식별하는 데 도움이 될 수 있는 사용 사례별 지침을 제공합니다. 또한 Amazon Neptune으로 데이터를 수집하고 여기에서 데이터를 소비할 때 따라야 할 모범 사례도 포함되어 있습니다.

성능 효율성 원칙은 다음 핵심 영역에 중점을 둡니다.

- 그래프 모델링
- 쿼리 최적화
- 클러스터 적정 규모 조정
- 쓰기 테이블 최적화

그래프 모델링 이해

레이블이 지정된 속성 그래프(LPG) 모델과 리소스 설명 프레임워크(RDF) 모델의 차이를 이해합니다. 대부분의 경우 이는 선호도 문제입니다. 그러나 한 모델이 다른 모델보다 더 적합한 몇 가지 사용 사례가 있습니다. 그래프에서 두 노드를 연결하는 경로에 대한 지식이 필요한 경우 LPG를 선택합니다. Neptune 클러스터 또는 기타 그래프 트리플 저장소에서 데이터를 페더레이션하려면 RDF를 선택합니다.

서비스형 소프트웨어(SaaS) 애플리케이션 또는 다중 테넌시가 필요한 애플리케이션을 빌드하는 경우 각 클러스터에 대해 하나의 테넌트를 보유하는 대신 데이터 모델에 테넌트의 논리적 분리를 통합하는 방법을 고려합니다. 이러한 유형의 설계를 달성하기 위해 고객 식별자를 레이블에 추가하거나 테넌트 식별자를 나타내는 속성 키 값 페어를 추가하는 등 SPARQL의 명명된 그래프 및 레이블 지정 전략을 사용할 수 있습니다. 클라이언트 계층이 이러한 값을 주입하여 논리적 분리를 유지하는지 확인합니다. 다중 테넌시 권장 사항에 대한 자세한 내용은 Amazon Neptune 데이터베이스를 실행하는 ISVs.

쿼리의 성능은 쿼리 처리 시 평가해야 하는 그래프 객체(노드, 엣지, 속성) 수에 따라 달라집니다. 따라서 그래프 모델은 애플리케이션의 성능에 상당한 영향을 미칠 수 있습니다. 가능하면 세분화된 레이블을 사용하고 경로 결정 또는 필터링을 달성하는 데 필요한 속성만 저장합니다. 성능을 높이려면 요약 노드 생성 또는 공통 경로를 연결하는 더 직접적인 엣지 생성과 같이 그래프의 일부를 미리 계산하는 방법을 고려합니다.

레이블이 동일한 엣지 수가 비정상적으로 많은 여러 노드에서의 탐색을 피하세요. 이러한 노드에는 종종 수천 개의 엣지가 있습니다(대부분의 노드에는 수십 개의 엣지 수가 있음). 그 결과 컴퓨팅 및 데이터 복잡성이 훨씬 높아집니다. 이러한 노드는 일부 쿼리 패턴에서 문제가 되지 않을 수 있지만, 특히 노드를 중간 단계로 탐색할 경우 데이터를 다르게 모델링하여 사용하지 않는 것이 좋습니다. 느린 쿼리 로그를 사용하여 이러한 노드에서 탐색하는 쿼리를 식별할 수 있습니다. 특히 디버그 모드를 사용하는 경우 평균 쿼리 패턴보다 지연 시간 및 데이터 액세스 지표가 훨씬 더 높을 수 있습니다.

Neptune을 사용하여 ID에 임의의 GUID 값을 할당하는 대신 사용 사례에서 지원하는 경우 노드 및 엣지에 결정적 노드 ID를 사용합니다. ID로 노드에 액세스하는 것이 가장 효율적인 방법입니다.

쿼리 최적화

openCypher 및 Gremlin 언어는 LPG 모델에서 상호 교환적으로 사용할 수 있습니다. 성능이 가장 중요한 문제인 경우 특정 쿼리 패턴에서 한 언어가 다른 언어보다 성능이 좋을 수 있으므로 두 언어를 서로 바꿔서 사용하는 방법을 고려합니다.

Neptune은 대체 쿼리 엔진(DFE)으로 전환하는 중입니다. openCypher는 DFE에서만 실행되지만 쿼리 주석을 사용하여 선택적으로 Gremlin 및 SPARQL 쿼리를 모두 DFE에서 실행하도록 설정할 수 있습니다. DFE가 활성화된 상태에서 쿼리를 테스트하고 DFE를 사용하지 않을 때 쿼리 패턴의 성능을 비교하는 것이 좋습니다.

Neptune은 전체 그래프를 평가하는 분석 쿼리가 아닌 단일 노드 또는 노드 세트에서 시작하고 여기에서 팬아웃하는 트랜잭션 유형 쿼리에 최적화되어 있습니다. 분석 쿼리 워크로드의 경우 Neptune Analytics를 사용합니다. Neptune Analytics는 데이터, 분석 및 알고리즘 처리를 위해 빠른 반복이 필요한 조사, 탐색 또는 데이터 과학 워크로드에 이상적인 선택입니다. 그래프 데이터에 대한 벡터 검색을 수행하고 Neptune 데이터베이스 인스턴스에서 직접 데이터를 로드할 수도 있습니다. Neptune Analytics가 요구 사항을 충족하지 않는 경우 AWS Pandas용 SDK 또는 AWS Glue 또는 또는 Amazon EMR과 결합된 neptune-export를 사용할 수도 있습니다.

모델 및 쿼리에서 비효율성과 병목 현상을 식별하려면 각 쿼리 언어에 대해 `profile` 및 `explain` API를 사용하여 쿼리 계획 및 쿼리 지표에 대한 자세한 설명을 얻습니다. 자세한 내용은 Gremlin profile, openCypher explain, SPARQL explain을 참조하세요.

쿼리 패턴을 이해합니다. 그래프의 명확한 엣지 수가 커질 경우 Neptune의 기본 액세스 전략이 비효율적으로 될 수 있습니다. 다음 쿼리는 매우 비효율적일 수 있습니다.

- 엣지 레이블이 지정되지 않은 경우 엣지를 역방향으로 탐색하는 쿼리.
- Gremlin의 `.both()`와 같이 이 동일한 패턴을 사용하는 절 또는 모든 언어로 노드를 삭제하는 절(레이블에 대한 지식 없이 수신 엣지를 삭제해야 함).
- 속성 레이블을 지정하지 않고 속성 값에 액세스하는 쿼리. 이러한 쿼리는 매우 비효율적일 수 있습니다. 사용 패턴과 일치하는 경우 OSGP 인덱스(객체, 제목, 그래프, 조건자)를 활성화하는 방법을 고려합니다.

느린 쿼리 로깅을 사용하여 느린 쿼리를 식별합니다. 최적화되지 않은 쿼리 계획이나 불필요하게 많은 인덱스 조회로 인해 느린 쿼리가 발생할 수 있으며, 이때 I/O 비용이 증가할 수 있습니다. Gremlin, SPARQL 또는 openCypher에 대한 Neptune 설명 및 프로파일 엔드포인트는 이러한 쿼리가 느린 이유를 이해하는 데 도움이 될 수 있습니다. 원인에 다음이 포함될 수 있습니다.

- 그래프의 평균 노드와 비교했을 때 엣지 수가 비정상적으로 많은 노드(예: 수십 개에 비해 수천 개에 해당)는 계산 복잡성을 추가하여 지연 시간을 늘리고 리소스 소비를 늘릴 수 있습니다. 이러한 노드가 올바르게 모델링되었는지 또는 통과해야 하는 엣지 수를 줄이기 위해 액세스 패턴을 개선할 수 있는지 확인합니다.
- 최적화되지 않은 쿼리에는 특정 단계가 최적화되지 않았다는 경고가 포함됩니다. 최적화된 단계를 사용하도록 이러한 쿼리를 다시 작성하면 성능이 향상될 수 있습니다.
- 중복 필터로 인해 불필요한 인덱스 조회가 발생할 수 있습니다. 마찬가지로 중복 패턴으로 인해 쿼리를 개선하여 최적화할 수 있는 중복 인덱스 조회가 나타날 수 있습니다(프로파일 출력에서 `Index Operations - Duplication ratio` 참조).
- Gremlin과 같은 일부 언어에는 강력한 형식의 숫자 값이 없으며 대신 형식 승격을 사용합니다. 예를 들어 값이 55인 경우 Neptune은 55와 동등한 정수, long, float 및 기타 숫자 유형인 값을 찾습니다. 이로 인해 추가 작업이 발생합니다. 사전에 유형 일치 항목을 알고 있는 경우 쿼리 힌트를 사용하여 이를 방지할 수 있습니다.
- 그래프 모델은 성능에 큰 영향을 미칠 수 있습니다. 보다 세분화된 레이블을 사용하거나 다중 홉 선형 경로에 대한 바로 가기를 미리 계산하여 평가해야 하는 객체 수를 줄이는 방법을 고려합니다.

쿼리 최적화만으로 성능 요구 사항에 도달할 수 없는 경우 Neptune과 함께 다양한 캐싱 기법을 사용하여 이러한 요구 사항을 충족하는 것이 좋습니다.

Neptune 성능은 각 버전마다 지속적으로 개선되고 있습니다. 릴리스 정보를 검토하여 각 릴리스의 개선 사항에 대한 세부 정보를 확인합니다. 최적의 성능을 달성하려면 Neptune DB 클러스터에 대한 정

기적인 업데이트를 계획하는 것이 좋습니다. 최신 버전은 최신 인스턴스도 지원합니다. r8g 인스턴스를 활용하려면 1.4.5.0 이상으로 업그레이드하는 것이 좋습니다. 이렇게 하면 워크로드 성능을 개선할 수 있는 방법에 대한 자세한 내용은 Amazon Neptune v1.4.5를 사용하는 AWS Graviton4 R8g 인스턴스를 사용하여 쿼리 가격 대비 성능을 4.7배 개선했습니다.

올바른 크기의 클러스터

동시성 및 처리량 요구 사항에 맞게 클러스터의 크기를 조정합니다. 클러스터의 각 인스턴스에서 처리할 수 있는 동시 쿼리 수는 해당 인스턴스의 가상 CPU(vCPU) 수의 2배와 같습니다. 모든 작업자 스레드가 점유된 동안 도착하는 추가 쿼리는 서버 측 대기열에 배치됩니다. 이러한 쿼리는 작업자 스레드가 사용 가능해지면 선입선출(FIFO) 방식으로 처리됩니다. `MainRequestQueuePendingRequests` Amazon CloudWatch 지표는 각 인스턴스의 현재 대기열 깊이를 보여줍니다. 이 값이 자주 0보다 크면 vCPU가 더 많은 인스턴스를 선택하는 것이 좋습니다. 대기열 깊이가 8,192를 초과하면 Neptune은 `ThrottlingException` 오류를 반환합니다.

각 인스턴스에 대한 RAM의 약 65%는 버퍼 캐시용으로 예약되어 있습니다. 버퍼 캐시에는 작업 데이터세트(전체 그래프가 아니라 쿼리 중인 데이터만)가 들어 있습니다. 스토리지 대신 버퍼 캐시에서 가져오는 데이터의 비율을 확인하려면 CloudWatch 지표 `BufferCacheHitRatio`를 모니터링합니다. 이 지표가 종종 99.9% 미만으로 떨어지면 메모리가 더 많은 인스턴스를 시도하여 지연 시간 및 I/O 비용을 줄이는지 확인하는 것이 좋습니다.

읽기 전용 복제본의 크기가 라이터 인스턴스와 같을 필요는 없습니다. 그러나 쓰기 워크로드가 많으면 더 작은 복제본이 지연되어 복제를 따라잡을 수 없기 때문에 재부팅될 수 있습니다. 따라서 라이터 인스턴스보다 크거나 같은 복제본을 만드는 것이 좋습니다.

읽기 복제본에 오토 스케일링을 사용하는 경우 새 읽기 복제본을 온라인 상태로 전환하는 데 최대 15분이 걸릴 수 있습니다. 클라이언트 트래픽이 빠르지만 예측 가능하게 증가하면 예약된 조정을 사용하여 해당 초기화 시간을 고려하도록 최소 읽기 복제본 수를 더 높게 설정하는 방법을 고려합니다.

서버리스 인스턴스는 여러 사용 사례와 워크로드를 지원합니다. 다음 시나리오에서는 서버리스 과다 프로비저닝된 인스턴스를 고려합니다.

- 워크로드는 하루 종일 변동되는 경우가 많습니다.
- 새 애플리케이션을 생성했는데 워크로드 크기가 어떻게 될지 잘 모릅니다.
- 개발 및 테스트를 수행하고 있습니다.

서버리스 인스턴스는 RAM의 GB당 1 USD 기준으로 동등한 프로비저닝된 인스턴스보다 비용이 더 높다는 점에 유의해야 합니다. 각 서버리스 인스턴스는 연결된 vCPU 및 네트워킹과 함께 2GB의 RAM으

로 구성됩니다. 옵션 간에 비용 분석을 수행하여 예상치 못한 비용을 방지합니다. 일반적으로 매일 몇 시간 동안만 워크로드가 매우 많고 하루의 나머지 시간에는 거의 0이거나 하루 종일 워크로드가 크게 변동하는 경우에만 서버리스를 통해 비용을 절감할 수 있습니다.

Amazon Neptune 요금 계산기를 활용하여 queries-per-second(QPS) 요구 사항과 같은 요소를 기반으로 클러스터의 올바른 구성을 평가할 수 있습니다.

쓰기 최적화

쓰기를 최적화하려면 다음을 고려합니다.

- Neptune Bulk Loader는 데이터베이스를 처음 로드하거나 기존 데이터에 추가하는 최적의 방법입니다. Neptune Loader는 트랜잭션에 기반하지 않으며 데이터를 삭제할 수 없으므로 요구 사항인 경우 사용하지 마세요.
- 트랜잭션 기반 업데이트는 지원되는 쿼리 언어를 사용하여 수행할 수 있습니다. 쓰기 I/O 작업을 최적화하려면 커밋당 50~100개의 객체 배치로 데이터를 작성합니다. 객체는 노드, 엣지 또는 LPG의 노드나 엣지에 있는 속성 또는 RDF의 트리플 저장소나 쿼드입니다.
- 모든 트랜잭션 Neptune 쓰기 작업은 각 연결에 대해 단일 스레드입니다. Neptune에 대량의 데이터를 전송할 때는 각각 데이터를 쓰는 여러 병렬 연결을 사용하는 방법을 고려합니다. Neptune에서 프로비저닝된 인스턴스를 선택하면 인스턴스 크기가 여러 vCPU에 연결됩니다. Neptune은 인스턴스의 각 vCPU에 대해 두 개의 데이터베이스 스레드를 생성하므로 최적의 병렬화를 테스트할 때 vCPU 수의 두 배에서 시작합니다. 서버리스 인스턴스는 4개의 NCU마다 약 1개의 비율로 vCPU 수를 조정합니다.

  > **Note**
  >
  > 이는 대량 로드 API에는 적용되지 않고 직접 연결에만 적용됩니다.

- 단일 연결만 언제든지 데이터를 쓰는 경우에도 모든 쓰기 프로세스 중에 ConcurrentModificationExceptions를 계획하고 효율적으로 처리합니다. `ConcurrentModificationExceptions`가 발생할 때 신뢰성을 보장하도록 클라이언트를 설계합니다.
- 모든 데이터를 삭제하려면 동시 삭제 쿼리를 실행하는 대신 빠른 재설정 API를 사용하는 방법을 고려합니다. 후자는 전자에 비해 훨씬 더 오래 걸리고 상당한 I/O 비용이 발생합니다.
- 대부분의 데이터를 삭제하려면 neptune-export를 사용하여 데이터를 새 클러스터로 로드해 유지하려는 데이터를 내보내는 것이 좋습니다. 그리고 원래 클러스터를 삭제합니다.

비용 최적화 요소

AWS Well-Architected Framework의 비용 최적화 원칙은 불필요한 비용을 방지하는 데 중점을 둡니다. 다음 권장 사항은 Amazon Neptune의 비용 최적화 설계 원칙 및 아키텍처 모범 사례를 충족하는 데 도움이 될 수 있습니다.

비용 최적화 원칙은 다음 핵심 영역에 중점을 둡니다.

- 시간 경과에 따른 지출 이해 및 자금 할당 제어
- 올바른 유형 및 수량의 리소스 선택
- 과도한 지출 없이 비즈니스 요구 사항을 충족하도록 규모 조정

필요한 사용 패턴 및 서비스 이해

Neptune은 데이터 모델에 식별 가능한 그래프 구조가 있고 쿼리가 관계를 탐색하고 여러 홉을 통과해야 하는 경우 워크로드에 적합합니다. 그래프 데이터베이스는 다음 패턴에 적합하지 않습니다.

- 주로 단일 홉 쿼리(데이터가 객체의 속성으로 더 잘 표현될 수 있는지 고려)
- 속성으로 저장된 JSON 또는 BLOB 데이터
- 많은 수의 노드에서 숫자 속성의 합계를 계산하는 등 데이터세트 전체에서 집계되는 쿼리

특정 액세스 패턴에 대해 여러 목적별 데이터베이스를 함께 사용하면 모든 요구 사항을 해결할 수 있는지 고려합니다. 예제:

- 단일 노드에 대한 높은 동시 검색 속성과 함께 복잡한 그래프에 대해 필요한 탐색 빈도가 낮은 API는 Neptune, DynamoDB 또는 Amazon DocumentDB 중 하나 이상을 사용하여 제공하는 것이 가장 좋습니다.
- 관계형 데이터베이스는 기존 기능을 유지하기 위해 Neptune과 공존할 수 있지만 관계형 데이터베이스에서 제대로 수행 및 확장되지 않는 다중 홉 순회에만 Neptune을 사용합니다.

다음을 포함하여 Neptune과 상호 작용하고 보완하는 서비스와 관련된 비용을 이해합니다.

- Neptune에 대량 로드되는 데이터 파일의 Amazon Simple Storage Service(Amazon S3) 스토리지 비용
- 삽입 또는 업서트 쿼리, 읽기 쿼리 및 Neptune 스트림 처리에 사용되는 Lambda 함수

- Amazon API Gateway 또는에서 클라이언트 애플리케이션과 상호 작용하기 위해 Neptune에 구축된 API 계층(데이터베이스에 직접 연결하는 대신) AWS AppSync
- AWS Glue Neptune과 데이터를 주고받는 데 사용되는 작업
- Neptune으로 거의 실시간에 가깝게 수집하기 위해 스트리밍 데이터를 수신하는 Amazon Kinesis 또는 Amazon Managed Streaming for Apache Kafka(Amazon MSK) 인스턴스.
- AWS Database Migration Service Neptune으로 관계형 데이터 마이그레이션
- Jupyter Notebook 및 딥 그래프 라이브러리 기계 학습 모델에 대한 Amazon SageMaker Runtime 비용

비용에 주의를 기울여 리소스 선택

Neptune 요금은 시간당 인스턴스 비용(또는 서버리스에 소비되는 Neptune 컴퓨팅 단위), 데이터 I/O 및 스토리지 사용량을 기준으로 합니다. 인스턴스는 평균적으로 전체 비용의 85%를 차지하므로 적정 규모로 조정하면 비용에 상당한 영향을 미칠 수 있습니다. 인스턴스를 적정 규모로 조정하는 가장 좋은 방법은 다양한 인스턴스에서 애플리케이션 성능을 테스트하고 다음 요소를 비교하는 것입니다.

- `MainRequestQueuePendingRequests` CloudWatch 지표가 0에 가까운 일관되게 낮은 수로 유지되나요?
- `BufferCacheHitRatio` CloudWatch 지표가 대부분의 시간 동안 99.9% 이상으로 유지되나요?
- 인스턴스 비용 및 관련 데이터 I/O 비용에 대한 비용 및 성능 곡선은 무엇인가요? 스토리지로 버퍼 캐시를 자주 교체해야 하는 크기가 작은 인스턴스의 경우 데이터 읽기 비용이 크게 증가할 수 있습니다. `BufferCacheHitRatio`는 이러한 시나리오에서 자주 감소합니다.

인스턴스 비용은 동일한 인스턴스 패밀리 내 크기에 따라 선형적으로 조정됩니다. `db.r6i.2xlarge` 인스턴스의 시간당 비용은 `db.r6i.xlarge` 인스턴스의 두 배이며 리소스 할당도 두 배입니다. `db.r6i.24xlarge` 인스턴스는 `db.r6i.xlarge` 인스턴스의 시간당 비용의 24배입니다.

지원해야 하는 동시 쿼리 수를 추정합니다. 읽기 전용 쿼리를 처리하기 위해 0~15개의 읽기 복제본을 보유할 수 있습니다. 요일, 주 또는 월에 따라 요구 사항이 다른 경우 여러 개의 작은 인스턴스를 사용하여 일정에 따라 조정할 수 있습니다. 인스턴스의 각 vCPU는 동시 쿼리를 처리하기 위한 두 개의 스레드를 제공합니다. 각각 4개의 vCPU가 있는 3개의 `db.r6i.xlarge` 읽기 복제본은 24개의 동시 쿼리를 처리할 수 있습니다.

트래픽 볼륨이 초당 쿼리(QPS)로 측정되는 경우 쿼리의 평균 지연 시간을 확인하기 위해 실험해야 합니다. Neptune 클러스터가 지원할 수 있는 초당 쿼리 수는 `vCPU × 2 × (1 second/average`

query latency)와 같습니다. 예를 들어 vCPU가 4개이고 쿼리 지연 시간이 100밀리초(0.1초)인 경우 QPS = 4 × 2 × (1s/0.1s) = 80 queries per second입니다.

프로비저닝된 인스턴스는 지속적이고 안정적이며 예측 가능한 워크로드에서 서버리스보다 저렴합니다. 서버리스는 하루에 몇 시간 동안만 사용량이 매우 많고(예: db.r6i.4xlarge) 남은 시간 동안 트래픽이 거의 없는 워크로드(예: Neptune 컴퓨팅 유닛 1개)가 있는 경우 비용을 최적화할 수 있는 기회를 제공합니다. 몇 시간 동안 스케일 업했다가 다시 스케일 다운하는 서버리스 인스턴스는 프로비저닝된 db.r6i.4xlarge 인스턴스를 하루 종일 사용하는 것보다 비용이 저렴합니다.

Neptune 1.4.5.0 이상으로 업그레이드하고 r8g 인스턴스를 활용하여 r7g 또는와 같은 이전 세대 인스턴스보다 저렴한 비용으로 더 나은 읽기 및 쓰기 처리량을 달성하는 것이 좋습니다r6g. 자세한 내용은 Amazon Neptune v1.4.5를 사용하는 AWS Graviton4 R8g 인스턴스를 사용하여 쿼리 가격 대비 성능을 4.7배 향상(블로그 게시물)을 참조하세요.AWS

Neptune 클러스터는 기본적으로 표준 스토리지로 생성됩니다(콘솔을 사용하여 생성하는 경우 기본적으로 I/O 최적화 스토리지 선택). I/O 최적화 스토리지를 사용하면 스토리지 및 인스턴스에 약간 더 높은 비용을 지불하지만 I/O 비용은 없습니다. 이로 인해 반복 비용이 더 예측 가능하게 발생하지만 I/O 사용량이 일반적으로 낮은 경우 표준 스토리지를 활용하는 것이 더 비용 효율적일 수 있습니다. 처음에 많은 데이터를 로드하려는 경우 I/O 최적화 스토리지를 선택하고 초기 데이터 로드를 수행한 다음 표준 스토리지로 전환하여 비용을 최적화할 수 있습니다. 스토리지 유형은 결제 모델에만 영향을 미치며 Neptune DB 클러스터 또는 인스턴스 구성에는 기술적 차이가 없습니다. 30일에 한 번씩 스토리지 유형을 변경할 수 있습니다. 30일 후 세부 Neptune 비용을 확인하고 Neptune 요금 페이지를 사용하여 I/O 최적화 스토리지를 사용하여 비용이 더 높았는지 여부를 계산합니다. 그렇다면 표준 스토리지를 계속 사용하고, 그렇지 않으면 I/O 최적화로 다시 전환하세요.

워크로드에 가장 적합한 Neptune 인스턴스 구성 선택

2025년 7월 15일 AWS 계정 이전에를 생성한 경우 AWS 프리 티어를 사용하여 Neptune을 사용한 엔트리 레벨 실험을 수행할 수 있습니다. 750시간의 무료 db.t3.medium 및 db.t4g.medium 인스턴스 사용만으로도 작은 규모에서 Neptune을 이해하기에 충분합니다. 무료 평가판 기간이 종료된 후에도 클러스터는 유지됩니다. 단, 이후 사용량에 대해서는 요금이 부과됩니다.

db.t3.medium 및 db.t4g.medium 인스턴스는 openCypher, Graph Explorer 또는 다양한 생성형 AI 통합을 사용하지 않는 저비용 개발 환경에 적합합니다. 이러한 인스턴스는 패밀리 인스턴스(8:1) 또는 R 패밀리 인스턴스(1X6:1)보다 RAM-to-vCPU 비율(2:1)이 작습니다. 이렇게 하면 openCypher 성능, GenAI 통합(LLM에 그래프 스키마를 알리기 위해) 및 Graph Explorer를 활성화하는 DFE 엔진 통계를 사용할 수 없습니다. T 패밀리 인스턴스를 사용할 때 특히 앞서 언급한 워크로드의 경우 성능 프로필이 크게 다를 수 있습니다. 이러한 인스턴스는 쿼리가 그래프의 상당 부분을 탐색할

`OutOfMemoryExceptions` 때의 발생을 증가시킬 수도 있습니다. 후자 조건이 영향을 받을 수 있는지 확인하려면 `BufferCacheHitRatio` CloudWatch 지표를 확인합니다.

프로덕션 환경을 나타내지 않는 일관되지 않은 결과가 발생할 수 있으므로 T 패밀리 인스턴스로 성능 또는 로드 테스트를 수행하지 않는 것이 좋습니다.

프로비저닝된 인스턴스는 워크로드가 매우 안정적이고 예측 가능한 경우 최상의 비용과 성능 조합을 제공합니다. 필요한 요청 동시성과 쿼리 복잡성에 따라 인스턴스 크기를 선택합니다. 동시성이 높을수록 vCPU가 더 많이 필요합니다. 쿼리 복잡성이 높을수록 RAM이 더 많이 필요합니다. `MainRequestQueuePendingRequests` CloudWatch 지표를 사용하여 전자의 영향을 확인합니다(0보다 크면 처리할 수 있는 것보다 동시 요청어 더 많은 상태임을 나타냄). `BufferCacheHitRatio` CloudWatch 지표를 사용하여 후자의 영향을 확인합니다. 99.9% 미만으로 자주 떨어지는 비율은 평가 중인 그래프의 작업 부분을 포함할 RAM이 부족하여 캐시 스왑이 더 자주 발생함을 나타냅니다. R 인스턴스 패밀리가 충분한 동시성을 제공하지만 RAM이 충분하지 않은 경우 인스턴스 X 패밀리를 사용해 보는 것이 좋습니다.

서버리스 인스턴스의 이상적인 사용 사례는 Neptune 설명서에 설명되어 있습니다. 프로비저닝 또는 서버리스 중 가장 적합한 옵션이 확실하지 않고 비용이 주요 관심사인 경우 서버리스에서 워크로드를 테스트하여 사용된 NCU 수를 확인하고 프로비저닝(`N hours × hourly provisioned cost`) 비용을 서버리스(`sum of NCUs × hourly cost per NCU`)와 비교합니다. 동일한 크기의 프로비저닝 인스턴스에 대해 잘 모르는 경우 NCU 하나는 약 2GB의 RAM과 관련 vCPU 및 네트워킹에 해당합니다. 프로비저닝된 인스턴스가 `r6i` 패밀리인 경우 비율은 연결된 네트워킹과 함께 RAM 8GB당 vCPU 1개 또는 NCUs 4개입니다. Amazon Neptune 요금 계산기는 최적의 비용 구성을 결정하는 데 도움이 되는 비교도 제공합니다.

기본 및 복제본 인스턴스에 서버리스를 사용하는 경우 승격 티어 0 및 1의 읽기 복제본은 라이터 인스턴스에 따라 NCU를 조정하므로 장애 조치 이벤트가 발생할 경우 적절하게 조정됩니다. 라이터 또는 리더 중 어떤 인스턴스가 가장 많은 트래픽을 수신하는지에 따라 이러한 인스턴스에 대한 NCU 제한을 설정합니다.

클러스터가 연중무휴로 필요하지 않은 환경에서는 사용하지 않을 때 Neptune 인스턴스를 끄고 사용하기 전에 다시 시작하는 스크립트를 작성하는 것이 좋습니다. Neptune 인스턴스는 필요한 유지 관리 업데이트가 적용되도록 7일마다 자동으로 다시 시작됩니다. 인스턴스를 장기간 끄려는 경우 주간 스크립트를 사용하여 인스턴스를 다시 종료합니다.

데이터 스토리지 적정 규모 조정 및 전송

보다 효율적인 쿼리(예: 그래프에서 사용해야 하는 노드, 엣지 및 속성 수가 더 적은 쿼리)는 I/O 전송을 줄이고 필요한 버퍼 캐시가 적기 때문에 더 작은 인스턴스를 사용할 수 있습니다. 쿼리 언어의 프로파

일 또는 설명 엔드포인트를 사용하여 쿼리를 최적화하고 쿼리 성능에 맞게 그래프 모델을 최적화하는 방법을 고려합니다.

Neptune은 큰 문자열에서 사전 인코딩을 사용하며, 해당 사전은 효율성이 아니라 성능에 최적화되어 있습니다. 속성에 대해 큰 BLOB, JSON 또는 빈번하게 변경되는 문자열이 있는 경우 Amazon S3, Amazon DynamoDB 또는 Amazon DocumentDB에서 이를 Neptune 외부에 저장하고 Neptune 노드 내에는 참조만 저장하는 방법을 고려합니다.

경우에 따라 더 큰 인스턴스 크기를 선택하는 방법이 더 저렴할 수 있습니다. 낮은 `BufferCacheHitRatio`로 인해 I/O 비용이 매우 높은 경우 더 큰 버퍼 캐시를 사용하면 해당 비용을 크게 줄일 수 있습니다. 스토리지에서 자주 전환되고 I/O 전송 비율을 발생시키는 대신 모든 데이터가 캐시에 맞춰지기 때문입니다.

Neptune은 쓸 때 복사 복제를 사용합니다. 그래프를 여러 샤드로 분할하기 위해 복제하는 경우 복제된 클러스터에서 원치 않는 데이터를 삭제하지 않는 방법이 더 효율적일 수 있습니다. 이 경우 새 데이터 페이지가 생성되어 스토리지 비용이 증가하기 때문입니다. 복제 이벤트 이전에 변경되지 않는 데이터는 두 클러스터에서 공유되는 단일 데이터 페이지에 존재하고, 해당 단일 사본에 대해서만 요금이 부과됩니다.

워크로드에 상당한 차이가 있는지 테스트하지 않은 경우 OSGP 인덱스를 활성화하거나 R5d 인스턴스를 사용하지 마세요. 둘 다 거의 발생하지 않는 시나리오를 위해 설계되었으며, 최소한의 이익 또는 전혀 얻는 이익 없이 비용이 증가할 수 있습니다.

지속 가능성 요소

지속 가능성 원칙은 클라우드 워크로드 실행이 환경에 미치는 영향을 최소화하는 데 중점을 둡니다. 주요 주제에는 지속 가능성에 대한 공동 책임 모델, 영향 이해, 사용 극대화로 필요한 리소스를 최소화하고 다운스트림 영향을 줄이는 것이 포함됩니다.

지속 가능성 원칙에는 다음과 같은 주요 핵심 영역이 포함됩니다.

- 사용자의 영향
- 지속 가능성 목표
- 최대 사용량
- 새롭고 보다 효율적인 하드웨어 및 소프트웨어 제품 예측 및 채택
- 관리형 서비스 사용
- 다운스트림 영향 감소

이 가이드는 사용자의 영향에 중점을 둡니다. 기타 지속 가능성 설계 원칙에 대한 자세한 내용은 AWS Well-Architected 프레임워크를 참조하세요.

사용자의 선택 사항과 요구 사항은 환경에 영향을 미칩니다. 탄소 집약도가 낮은 AWS 리전 을 선택할 수 있고 요구 사항이 가동 시간과 내구성을 극대화하는 대신 실제 워크로드 요구 사항을 반영하는 경우 워크로드의 지속 가능성이 증가합니다. 다음 섹션에서는 워크로드 설계 및 지속적 운영에 채택될 경우 환경에 긍정적인 영향을 미칠 수 있는 모범 사례와 신중한 고려 사항에 대해 설명합니다.

AWS 리전 선택

일부는 Amazon 재생 에너지 프로젝트 AWS 리전 근처에 있거나 그리드의 탄소 강도가 다른 프로젝트보다 낮은 곳에 있습니다. 워크로드에 적합할 수 있는 리전의 지속 가능성 영향을 고려하고 목록을 Neptune을 사용할 수 있는 리전과 상호 참조합니다.

사용자 행동 패턴 기반 소비

사용자의 트래픽 및 동작에 맞게 소비를 적정 규모로 조정하면 AWS 에서 환경에 미치는 서비스의 영향을 최소화하는 데 도움이 됩니다. 솔루션을 설계할 때 다음 모범 사례를 고려하세요.

- `CPUUtilization`, `MainRequestQueuePendingRequests` 및 `TotalRequestsPerSec`과 같은 Amazon CloudWatch 지표를 모니터링하여 수요가 가장 높은 시점과 가장 낮은 시점을 결정하고 해당 시간 동안 클러스터 리소스의 크기가 적절한지 확인합니다.
- 사용하지 않는 시간 동안 비프로덕션 환경의 중지를 자동화합니다. 자세한 내용은 블로그 게시물 Automate the stopping and starting of Amazon Neptune environment resources using resource tags를 참조하세요.
- 트래픽 패턴이 자주 예기치 않게 달라지는 경우 피크 트래픽에 대해 프로비저닝된 인스턴스를 사용하는 대신 수요에 따라 스케일 업 및 스케일 다운되는 Neptune Serverless 인스턴스를 사용하는 것이 좋습니다.
- 비즈니스 연속성 목표 외에도 서비스 수준 계약을 지속 가능성 목표에 맞게 조정하는 방법을 고려합니다. 다중 리전 재해 복구, 고가용성 또는 장기 백업 보존, 특히 비프로덕션 환경 또는 미션 크리티컬 이외의 워크로드와 같은 요구 사항을 완화하면 이러한 목표를 달성하는 데 필요한 리소스의 양을 줄일 수 있습니다.

소프트웨어 개발 및 아키텍처 패턴 최적화

낭비를 방지하려면 모델 및 쿼리를 최적화하고 컴퓨팅 리소스를 공유하여 Neptune 인스턴스 및 클러스터에서 사용할 수 있는 모든 리소스를 사용합니다. 구체적인 모범 사례는 다음과 같습니다.

- 개발자가 각각 고유한 인스턴스를 생성하는 대신 Neptune 인스턴스와 Jupyter Notebook 애플리케이션 인스턴스를 공유하도록 합니다. 멀티테넌시 분할 전략을 사용하여 각 개발자에게 단일 Neptune 클러스터의 자체 논리적 파티션을 제공하고 단일 Jupyter 인스턴스에서 각 개발자에 대해 별도의 노트북 폴더를 생성합니다.
- 데이터를 로드하고 레코드를 더 큰 트랜잭션으로 배치 처리하기 위한 병렬 스레드와 같이 리소스 사용을 극대화하고 유휴 시간을 최소화하는 패턴을 구현합니다.
- 결과를 계산하는 데 필요한 리소스를 최소화하도록 쿼리 및 그래프 모델을 최적화합니다.
- Gremlin 쿼리 결과의 경우 결과 캐시 기능을 사용하여 페이지가 지정되거나 자주 반복되는 쿼리를 다시 계산하는 데 소비되는 리소스를 최소화합니다.
- Neptune 환경을 최신 상태로 유지합니다. 최신 버전의 Neptune은 더 효율적인 Graviton과 같은 최신 Amazon EC2 인스턴스를 지원합니다. 또한 쿼리를 계산하는 데 필요한 리소스의 양을 줄이는 쿼리 최적화 개선 및 버그 수정도 있습니다.

리소스

참조

- AWS Well-Architected
- AWS Well-Architected Framework 설명서
- Neptune 최신 업데이트
- 모범 사례: Neptune 최대한 활용
- Amazon Neptune 요금 계산기

블로그 게시물

- Automated testing of Amazon Neptune data access with Apache TinkerPop Gremlin
- Automate the stopping and starting of Amazon Neptune environment resources using resource tags
- Fine Grained Access Control for Amazon Neptune data plane actions
- Amazon Neptune v1.4.5를 사용하여 AWS Graviton4 R8g 인스턴스의 쓰기 쿼리 가격 대비 성능이 4.7배 향상되었습니다.
- Orca Security가 Amazon Neptune 데이터베이스 성능을 최적화한 방법
- Amazon Neptune 퍼블릭 엔드포인트를 사용하여 그래프 애플리케이션을 더 빠르게 구축
- 새로운 Amazon Neptune 엔진 버전은 openCypher 쿼리 성능을 위해 최대 9배 더 빠르고 10배 더 높은 처리량을 제공합니다.

무료 AWS Skill Builder 과정

- Getting Started with Amazon Neptune
- Building Applications on Amazon Neptune
- Data Modeling for Amazon Neptune

기여자

이 가이드의 기여자는 다음과 같습니다.

- Brian O'Keefe, Principal Neptune Solutions Architect, AWS
- Abhishek Mishra, Senior Neptune Solutions Architect, AWS
- Ganesh Sawhney, 팀 리드 - 전략적 파트너 성공 솔루션 아키텍트, AWS
- Michael Havey, Senior Neptune Solutions Architect, AWS
- Kevin Phillips, Neptune 솔루션 아키텍트, AWS
- Melissa Kwok, Neptune 솔루션 아키텍트, AWS
- Sakti Mishra, Principal Solutions Architect AWS
- Javed Ali, Senior Solutions Architect, AWS

문서 기록

아래 표에 이 가이드의 주요 변경 사항이 설명되어 있습니다. 향후 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
Neptune 릴리스 업데이트	Amazon Neptune 1.4.6.0 이상에 대한 정보를 포함하도록 설명서를 업데이트했습니다.	2026년 1월 2일
최초 게시	—	2023년 9월 27일

AWS 권장 가이드 용어집

다음은 AWS 권장 가이드에서 제공하는 전략, 가이드 및 패턴에서 일반적으로 사용되는 용어입니다. 용어집 항목을 제안하려면 용어집 끝에 있는 피드백 제공 링크를 사용하십시오.

숫자

7가지 전략

애플리케이션을 클라우드로 이전하기 위한 7가지 일반적인 마이그레이션 전략 이러한 전략은 Gartner가 2011년에 파악한 5가지 전략을 기반으로 하며 다음으로 구성됩니다.

- 리팩터링/리아키텍트 - 클라우드 네이티브 기능을 최대한 활용하여 애플리케이션을 이동하고 해당 아키텍처를 수정함으로써 민첩성, 성능 및 확장성을 개선합니다. 여기에는 일반적으로 운영 체제와 데이터베이스 이식이 포함됩니다. 예: 온프레미스 Oracle 데이터베이스를 Amazon Aurora PostgreSQL 호환 에디션으로 마이그레이션합니다.
- 리플랫포밍(리프트 앤드 리셰이프) - 애플리케이션을 클라우드로 이동하고 일정 수준의 최적화를 도입하여 클라우드 기능을 활용합니다. 예: 온프레미스 Oracle 데이터베이스를 AWS 클라우드의 Amazon Relational Database Service(Amazon RDS) for Oracle로 마이그레이션합니다.
- 재구매(드롭 앤드 숍) - 일반적으로 기존 라이선스에서 SaaS 모델로 전환하여 다른 제품으로 전환합니다. 예: 고객 관계 관리(CRM) 시스템을 Salesforce.com으로 마이그레이션합니다.
- 리호스팅(리프트 앤드 시프트) - 애플리케이션을 변경하지 않고 클라우드로 이동하여 클라우드 기능을 활용합니다. 예: 온프레미스 Oracle 데이터베이스를 AWS 클라우드클라우드의 EC2 인스턴스에 있는 Oracle로 마이그레이션합니다.
- 재배치(하이퍼바이저 수준의 리프트 앤 시프트) - 새 하드웨어를 구매하거나, 애플리케이션을 다시 작성하거나, 기존 운영을 수정하지 않고도 인프라를 클라우드로 이동합니다. 온프레미스 플랫폼에서 동일한 플랫폼의 클라우드 서비스로 서버를 마이그레이션합니다. 예: Microsoft Hyper-V 애플리케이션을 로 마이그레이션합니다 AWS.
- 유지(보관) - 소스 환경에 애플리케이션을 유지합니다. 대규모 리팩터링이 필요하고 해당 작업을 나중으로 연기하려는 애플리케이션과 비즈니스 차원에서 마이그레이션할 이유가 없어 유지하려는 레거시 애플리케이션이 여기에 포함될 수 있습니다.
- 사용 중지 - 소스 환경에서 더 이상 필요하지 않은 애플리케이션을 폐기하거나 제거합니다.

A

ABAC

속성 기반 액세스 제어를 참조하세요.

추상화된 서비스

관리형 서비스를 참조하세요.

ACID

원자성, 일관성, 격리성, 내구성을 참조하세요.

능동-능동 마이그레이션

양방향 복제 도구 또는 이중 쓰기 작업을 사용하여 소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되고, 두 데이터베이스 모두 마이그레이션 중 연결 애플리케이션의 트랜잭션을 처리하는 데이터베이스 마이그레이션 방법입니다. 이 방법은 일회성 전환이 필요한 대신 소규모의 제어된 배치로 마이그레이션을 지원합니다. 더 유연하지만 액티브 패시브 마이그레이션보다 더 많은 작업이 필요합니다.

능동-수동 마이그레이션

소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되지만 소스 데이터베이스만 연결 애플리케이션의 트랜잭션을 처리하고 데이터는 대상 데이터베이스로 복제되는 데이터베이스 마이그레이션 방법입니다. 대상 데이터베이스는 마이그레이션 중 어떤 트랜잭션도 허용하지 않습니다.

집계 함수

행 그룹에서 작동하고 그룹에 대한 단일 반환 값을 계산하는 SQL 함수입니다. 집계 함수의 예로 `SUM` 및 `MAX`가 있습니다.

AI

인공 지능을 참조하세요.

AIOps

인공 지능 운영을 참조하세요.

익명화

데이터세트에서 개인 정보를 영구적으로 삭제하는 프로세스입니다. 익명화는 개인 정보 보호에 도움이 될 수 있습니다. 익명화된 데이터는 더 이상 개인 데이터로 간주되지 않습니다.

안티 패턴

솔루션이 다른 솔루션보다 비생산적이거나 비효율적이거나 덜 효과적이어서 반복되는 문제에 자주 사용되는 솔루션입니다.

애플리케이션 제어

맬웨어로부터 시스템을 보호하기 위해 승인된 애플리케이션만 사용하도록 허용하는 보안 접근 방식입니다.

애플리케이션 포트폴리오

애플리케이션 구축 및 유지 관리 비용과 애플리케이션의 비즈니스 가치를 비롯하여 조직에서 사용하는 각 애플리케이션에 대한 세부 정보 모음입니다. 이 정보는 포트폴리오 탐색 및 분석 프로세스의 핵심이며 마이그레이션, 현대화 및 최적화할 애플리케이션을 식별하고 우선순위를 정하는 데 도움이 됩니다.

인공 지능

컴퓨터 기술을 사용하여 학습, 문제 해결, 패턴 인식 등 일반적으로 인간과 관련된 인지 기능을 수행하는 것을 전문으로 하는 컴퓨터 과학 분야입니다. 자세한 내용은 What is Artificial Intelligence?를 참조하십시오.

인공 지능 운영(AIOps)

기계 학습 기법을 사용하여 운영 문제를 해결하고, 운영 인시던트 및 사용자 개입을 줄이고, 서비스 품질을 높이는 프로세스입니다. AWS 마이그레이션 전략에서 AIOps가 사용되는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

비대칭 암호화

한 쌍의 키, 즉 암호화를 위한 퍼블릭 키와 복호화를 위한 프라이빗 키를 사용하는 암호화 알고리즘입니다. 퍼블릭 키는 복호화에 사용되지 않으므로 공유할 수 있지만 프라이빗 키에 대한 액세스는 엄격히 제한되어야 합니다.

원자성, 일관성, 격리성, 내구성(ACID)

오류, 정전 또는 기타 문제가 발생한 경우에도 데이터베이스의 데이터 유효성과 운영 신뢰성을 보장하는 소프트웨어 속성 세트입니다.

ABAC(속성 기반 액세스 제어)

부서, 직무, 팀 이름 등의 사용자 속성을 기반으로 세분화된 권한을 생성하는 방식입니다. 자세한 내용은 AWS Identity and Access Management (IAM) 설명서의 용 ABAC AWS를 참조하세요.

신뢰할 수 있는 데이터 소스

가장 신뢰할 수 있는 정보 소스로 간주되는 기본 버전의 데이터를 저장하는 위치입니다. 익명화, 편집 또는 가명화와 같은 데이터 처리 또는 수정의 목적으로 신뢰할 수 있는 데이터 소스의 데이터를 다른 위치로 복사할 수 있습니다.

가용 영역

다른 가용 영역의 장애로부터 격리 AWS 리전 되고 동일한 리전의 다른 가용 영역에 저렴하고 지연 시간이 짧은 네트워크 연결을 제공하는 내의 고유한 위치입니다.

AWS 클라우드 채택 프레임워크(AWS CAF)

조직이 클라우드로 성공적으로 전환 AWS 하기 위한 효율적이고 효과적인 계획을 개발하는 데 도움이 되는의 지침 및 모범 사례 프레임워크입니다. AWS CAF는 지침을 비즈니스, 사람, 거버넌스, 플랫폼, 보안 및 운영이라는 6가지 중점 영역으로 구성합니다. 비즈니스, 사람 및 거버넌스 관점은 비즈니스 기술과 프로세스에 초점을 맞추고, 플랫폼, 보안 및 운영 관점은 전문 기술과 프로세스에 중점을 둡니다. 예를 들어, 사람 관점은 인사(HR), 직원 배치 기능 및 인력 관리를 담당하는 이해관계자를 대상으로 합니다. 이러한 관점에서 AWS CAF는 성공적인 클라우드 채택을 위해 조직을 준비하는 데 도움이 되는 인력 개발, 교육 및 커뮤니케이션에 대한 지침을 제공합니다. 자세한 내용은 AWS CAF 웹사이트와 AWS CAF 백서를 참조하세요.

AWS 워크로드 검증 프레임워크(AWS WQF)

데이터베이스 마이그레이션 워크로드를 평가하고, 마이그레이션 전략을 권장하고, 작업 견적을 제공하는 도구입니다. AWS WQF는 AWS Schema Conversion Tool (AWS SCT)에 포함되어 있습니다. 데이터베이스 스키마 및 코드 객체, 애플리케이션 코드, 종속성 및 성능 특성을 분석하고 평가 보고서를 제공합니다.

B

악성 봇

개인 또는 조직을 방해하거나 해를 입히기 위한 봇입니다.

BCP

비즈니스 연속성 계획을 참조하세요.

동작 그래프

리소스 동작과 시간 경과에 따른 상호 작용에 대한 통합된 대화형 뷰입니다. Amazon Detective에서 동작 그래프를 사용하여 실패한 로그온 시도, 의심스러운 API 직접 호출 및 유사한 작업을 검사할 수 있습니다. 자세한 내용은 Detective 설명서의 Data in a behavior graph를 참조하십시오.

빅 엔디안 시스템

가장 중요한 바이트를 먼저 저장하는 시스템입니다. 엔디안도 참조하세요.

바이너리 분류

바이너리 결과(가능한 두 클래스 중 하나)를 예측하는 프로세스입니다. 예를 들어, ML 모델이 “이 이메일이 스팸인가요, 스팸이 아닌가요?”, '이 제품은 책인가요, 자동차인가요?' 등의 문제를 예측해야 할 수 있습니다.

블룸 필터

요소가 세트의 멤버인지 여부를 테스트하는 데 사용되는 메모리 효율성이 높은 확률론적 데이터 구조입니다.

블루/그린(Blue/Green) 배포

동일하지만 별개의 두 환경을 생성하는 배포 전략입니다. 하나의 환경(파란색)에서 현재 애플리케이션 버전을 실행하고 새 애플리케이션 버전은 다른 환경(녹색)에서 실행합니다. 이 전략을 사용하면 영향을 최소화하면서 신속하게 롤백할 수 있습니다.

bot

인터넷을 통해 자동화된 태스크를 실행하고 인적 활동이나 상호 작용을 시뮬레이션하는 소프트웨어 애플리케이션입니다. 인터넷에서 정보를 인덱싱하는 웹 크롤러와 같이 유용하거나 이로운 봇도 있습니다. 악성 봇이라고 하는 다른 일부 봇은 개인 또는 조직을 방해하거나 해를 입히기 위한 봇입니다.

봇넷

맬웨어에 감염되고 봇 허더 또는 봇 운영자와 같은 단일 당사자가 제어하는 봇 네트워크입니다. 봇넷은 봇의 규모와 봇의 영향 범위를 확대하는 가장 잘 알려진 메커니즘입니다.

브랜치

코드 리포지토리의 포함된 영역입니다. 리포지토리에 생성되는 첫 번째 브랜치가 기본 브랜치입니다. 기존 브랜치에서 새 브랜치를 생성한 다음 새 브랜치에서 기능을 개발하거나 버그를 수정할 수 있습니다. 기능을 구축하기 위해 생성하는 브랜치를 일반적으로 기능 브랜치라고 합니다. 기능을 출시할 준비가 되면 기능 브랜치를 기본 브랜치에 다시 병합합니다. 자세한 내용은 About branches(GitHub 설명서)를 참조하십시오.

긴급 액세스 권한

예외적인 상황에서 승인된 프로세스를 통해 사용자가 일반적으로 액세스할 권한이 없는에 액세스할 수 AWS 계정 있는 빠른 방법입니다. 자세한 내용은 AWS Well-Architected 지침의 Implement break-glass procedures 지표를 참조하세요.

브라운필드 전략

사용자 환경의 기존 인프라 시스템 아키텍처에 브라운필드 전략을 채택할 때는 현재 시스템 및 인프라의 제약 조건을 중심으로 아키텍처를 설계합니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

버퍼 캐시

가장 자주 액세스하는 데이터가 저장되는 메모리 영역입니다.

사업 역량

기업이 가치를 창출하기 위해 하는 일(예: 영업, 고객 서비스 또는 마케팅)입니다. 마이크로서비스 아키텍처 및 개발 결정은 비즈니스 역량에 따라 이루어질 수 있습니다. 자세한 내용은 백서의 AWS에서 컨테이너화된 마이크로서비스 실행의 비즈니스 역량 중심의 구성화 섹션을 참조하십시오.

비즈니스 연속성 계획(BCP)

대규모 마이그레이션과 같은 중단 이벤트가 운영에 미치는 잠재적 영향을 해결하고 비즈니스가 신속하게 운영을 재개할 수 있도록 지원하는 계획입니다.

C

CAF

AWS Cloud Adoption Framework를 참조하세요.

카나리 배포

최종 사용자에게 제공하는 느린 증분 릴리스 버전입니다. 확신이 들면 새 버전을 배포하고 현재 버전을 완전히 교체합니다.

CCoE

클라우드 혁신 센터를 참조하세요.

CDC

데이터 캡처 변경을 참조하세요.

변경 데이터 캡처(CDC)

데이터베이스 테이블과 같은 데이터 소스의 변경 내용을 추적하고 변경 사항에 대한 메타데이터를 기록하는 프로세스입니다. 대상 시스템의 변경 내용을 감사하거나 복제하여 동기화를 유지하는 등의 다양한 용도로 CDC를 사용할 수 있습니다.

카오스 엔지니어링

시스템의 복원력을 테스트하기 위해 의도적으로 장애나 중단 이벤트를 도입합니다. AWS Fault Injection Service (AWS FIS)를 사용하여 AWS 워크로드에 스트레스를 주고 응답을 평가하는 실험을 수행할 수 있습니다.

CI/CD

지속적 통합 및 지속적 전송을 참조하세요.

분류

예측을 생성하는 데 도움이 되는 분류 프로세스입니다. 분류 문제에 대한 ML 모델은 이산 값을 예측합니다. 이산 값은 항상 서로 다릅니다. 예를 들어, 모델이 이미지에 자동차가 있는지 여부를 평가해야 할 수 있습니다.

클라이언트측 암호화

대상이 데이터를 AWS 서비스 수신하기 전에 로컬에서 데이터를 암호화합니다.

클라우드 혁신 센터(CCoE)

클라우드 모범 사례 개발, 리소스 동원, 마이그레이션 타임라인 설정, 대규모 혁신을 통한 조직 선도 등 조직 전체에서 클라우드 채택 노력을 추진하는 다분야 팀입니다. 자세한 내용은 AWS 클라우드 엔터프라이즈 전략 블로그의 CCoE 게시물을 참조하세요.

클라우드 컴퓨팅

원격 데이터 스토리지와 IoT 디바이스 관리에 일반적으로 사용되는 클라우드 기술 클라우드 컴퓨팅은 일반적으로 엣지 컴퓨팅 기술에 연결되어 있습니다.

클라우드 운영 모델

IT 조직에서 하나 이상의 클라우드 환경을 구축, 성숙화 및 최적화하는 데 사용되는 운영 모델입니다. 자세한 내용은 클라우드 운영 모델 구축을 참조하십시오.

클라우드 채택 단계

조직이 AWS 클라우드로 마이그레이션할 때 일반적으로 거치는 4단계는 다음과 같습니다.

- 프로젝트 - 개념 증명 및 학습 목적으로 몇 가지 클라우드 관련 프로젝트 실행
- 기반 – 클라우드 채택 확장을 위한 기초 투자(예: 랜딩 존 생성, CCoE 정의, 운영 모델 구축)
- 마이그레이션 - 개별 애플리케이션 마이그레이션
- Re-invention - 제품 및 서비스 최적화와 클라우드 혁신

이러한 단계는 Stephen Orban이 블로그 게시물 The Journey Toward Cloud-First and the Stages of Adoption on the AWS 클라우드 Enterprise Strategy 블로그에서 정의했습니다. AWS 마이그레이션 전략과 어떤 관련이 있는지에 대한 자세한 내용은 마이그레이션 준비 가이드를 참조하세요.

CMDB

구성 관리 데이터베이스를 참조하세요.

코드 리포지토리

소스 코드와 설명서, 샘플, 스크립트 등의 기타 자산이 버전 관리 프로세스를 통해 저장되고 업데이트되는 위치입니다. 일반적인 클라우드 리포지토리로 GitHub 또는 Bitbucket Cloud가 포함됩니다. 코드의 각 버전을 브랜치라고 합니다. 마이크로서비스 구조에서 각 리포지토리는 단일 기능 전용입니다. 단일 CI/CD 파이프라인은 여러 리포지토리를 사용할 수 있습니다.

콜드 캐시

비어 있거나, 제대로 채워지지 않았거나, 오래되었거나 관련 없는 데이터를 포함하는 버퍼 캐시입니다. 주 메모리나 디스크에서 데이터베이스 인스턴스를 읽어야 하기 때문에 성능에 영향을 미치며, 이는 버퍼 캐시에서 읽는 것보다 느립니다.

콜드 데이터

거의 액세스되지 않고 일반적으로 과거 데이터인 데이터. 이런 종류의 데이터를 쿼리할 때는 일반적으로 느린 쿼리가 허용됩니다. 이 데이터를 성능이 낮고 비용이 저렴한 스토리지 계층 또는 클래스로 옮기면 비용을 절감할 수 있습니다.

컴퓨터 비전(CV)

기계 학습을 사용하여 디지털 이미지 및 비디오와 같은 시각적 형식에서 정보를 분석하고 추출하는 AI 필드입니다. 예를 들어 Amazon SageMaker AI는 CV에 대한 이미지 처리 알고리즘을 제공합니다.

구성 드리프트

워크로드의 경우 구성이 예상되는 상태에서 변경됩니다. 이로 인해 워크로드가 규정을 준수하지 않을 수 있으며, 이는 일반적으로 점진적이고 의도되지 않은 작업입니다.

구성 관리 데이터베이스(CMDB)

하드웨어 및 소프트웨어 구성 요소와 해당 구성을 포함하여 데이터베이스와 해당 IT 환경에 대한 정보를 저장하고 관리하는 리포지토리입니다. 일반적으로 마이그레이션의 포트폴리오 탐색 및 분석 단계에서 CMDB의 데이터를 사용합니다.

규정 준수 팩

규정 준수 및 보안 검사를 사용자 지정하기 위해 조합할 수 있는 AWS Config 규칙 및 문제 해결 작업의 모음입니다. YAML 템플릿을 사용하여 적합성 팩을 AWS 계정 및 리전 또는 조직 전체에 단일 엔터티로 배포할 수 있습니다. 자세한 내용은 AWS Config 설명서의 적합성 팩을 참조하세요.

지속적 통합 및 지속적 전달(CI/CD)

소프트웨어 릴리스 프로세스의 소스, 빌드, 테스트, 스테이징 및 프로덕션 단계를 자동화하는 프로세스입니다. CI/CD는 일반적으로 파이프라인으로 설명됩니다. CI/CD를 통해 프로세스를 자동화하고, 생산성을 높이고, 코드 품질을 개선하고, 더 빠르게 제공할 수 있습니다. 자세한 내용은 지속적 전달의 이점을 참조하십시오. CD는 지속적 배포를 의미하기도 합니다. 자세한 내용은 지속적 전달(Continuous Delivery)과 지속적인 개발을 참조하십시오.

CV

컴퓨터 비전을 참조하세요.

D

저장 데이터

스토리지에 있는 데이터와 같이 네트워크에 고정되어 있는 데이터입니다.

데이터 분류

중요도와 민감도를 기준으로 네트워크의 데이터를 식별하고 분류하는 프로세스입니다. 이 프로세스는 데이터에 대한 적절한 보호 및 보존 제어를 결정하는 데 도움이 되므로 사이버 보안 위험 관리 전략의 중요한 구성 요소입니다. 데이터 분류는 AWS Well-Architected Framework에서 보안 원칙의 구성 요소입니다. 자세한 내용은 데이터 분류를 참조하십시오.

데이터 드리프트

프로덕션 데이터와 ML 모델 학습에 사용된 데이터 간의 상당한 차이 또는 시간 경과에 따른 입력 데이터의 의미 있는 변화. 데이터 드리프트는 ML 모델 예측의 전반적인 품질, 정확성 및 공정성을 저하시킬 수 있습니다.

전송 중 데이터

네트워크를 통과하고 있는 데이터입니다. 네트워크 리소스 사이를 이동 중인 데이터를 예로 들 수 있습니다.

데이터 메시

중앙 집중식 관리 및 거버넌스를 통해 분산되고 탈중앙화된 데이터 소유권을 제공하는 아키텍처 프레임워크입니다.

데이터 최소화

꼭 필요한 데이터만 수집하고 처리하는 원칙입니다. 에서 데이터를 최소화하면 개인 정보 보호 위험, 비용 및 분석 탄소 발자국을 줄일 AWS 클라우드 수 있습니다.

데이터 경계

신뢰할 수 있는 자격 증명만 예상 네트워크에서 신뢰할 수 있는 리소스에 액세스하도록 하는 데 도움이 되는 AWS 환경의 예방 가드레일 세트입니다. 자세한 내용은 데이터 경계 구축을 참조하세요 AWS.

데이터 사전 처리

원시 데이터를 ML 모델이 쉽게 구문 분석할 수 있는 형식으로 변환하는 것입니다. 데이터를 사전 처리한다는 것은 특정 열이나 행을 제거하고 누락된 값, 일관성이 없는 값 또는 중복 값을 처리함을 의미할 수 있습니다.

데이터 출처

라이프사이클 전반에 걸쳐 데이터의 출처와 기록을 추적하는 프로세스(예: 데이터 생성, 전송, 저장 방법).

데이터 주체

데이터를 수집 및 처리하는 개인입니다.

데이터 웨어하우스

분석과 같은 비즈니스 인텔리전스를 지원하는 데이터 관리 시스템입니다. 데이터 웨어하우스에는 보통 많은 양의 기록 데이터가 포함되며 일반적으로 쿼리 및 분석에 사용됩니다.

데이터 정의 언어(DDL)

데이터베이스에서 테이블 및 객체의 구조를 만들거나 수정하기 위한 명령문 또는 명령입니다.

데이터베이스 조작 언어(DML)

데이터베이스에서 정보를 수정(삽입, 업데이트 및 삭제)하기 위한 명령문 또는 명령입니다.

DDL

데이터 정의 언어를 참조하세요.

딥 앙상블

예측을 위해 여러 딥 러닝 모델을 결합하는 것입니다. 딥 앙상블을 사용하여 더 정확한 예측을 얻거나 예측의 불확실성을 추정할 수 있습니다.

딥 러닝

여러 계층의 인공 신경망을 사용하여 입력 데이터와 관심 대상 변수 간의 매핑을 식별하는 ML 하위 분야입니다.

심층 방어

네트워크와 그 안의 데이터 기밀성, 무결성 및 가용성을 보호하기 위해 컴퓨터 네트워크 전체에 일련의 보안 메커니즘과 제어를 신중하게 계층화하는 정보 보안 접근 방식입니다. 이 전략을 채택하면 AWS Organizations 구조의 여러 계층에 여러 제어를 AWS추가하여 리소스를 보호할 수 있습니다. 예를 들어, 심층 방어 접근 방식은 다단계 인증, 네트워크 세분화 및 암호화를 결합할 수 있습니다.

위임된 관리자

에서 AWS Organizations호환되는 서비스는 AWS 멤버 계정을 등록하여 조직의 계정을 관리하고 해당 서비스에 대한 권한을 관리할 수 있습니다. 이러한 계정을 해당 서비스의 위임된 관리자라고 합니다. 자세한 내용과 호환되는 서비스 목록은 AWS Organizations 설명서의 AWS Organizations와 함께 사용할 수 있는 AWS 서비스를 참조하십시오.

배포

대상 환경에서 애플리케이션, 새 기능 또는 코드 수정 사항을 사용할 수 있도록 하는 프로세스입니다. 배포에는 코드 베이스의 변경 사항을 구현한 다음 애플리케이션 환경에서 해당 코드베이스를 구축하고 실행하는 작업이 포함됩니다.

개발 환경

환경을 참조하세요.

탐지 제어

이벤트 발생 후 탐지, 기록 및 알림을 수행하도록 설계된 보안 제어입니다. 이러한 제어는 기존의 예방적 제어를 우회한 보안 이벤트를 알리는 2차 방어선입니다. 자세한 내용은 AWS에서 보안 제어 구현의 탐지 제어를 참조하세요.

개발 가치 흐름 매핑 (DVSM)

소프트웨어 개발 라이프사이클에서 속도와 품질에 부정적인 영향을 미치는 제약 조건을 식별하고 우선 순위를 지정하는 데 사용되는 프로세스입니다. DVSM은 원래 린 제조 방식을 위해 설계된 가치 흐름 매핑 프로세스를 확장합니다. 소프트웨어 개발 프로세스를 통해 가치를 창출하고 이동하는 데 필요한 단계와 팀에 중점을 둡니다.

디지털 트윈

건물, 공장, 산업 장비 또는 생산 라인과 같은 실제 시스템을 가상으로 표현한 것입니다. 디지털 트윈은 예측 유지 보수, 원격 모니터링, 생산 최적화를 지원합니다.

차원 테이블

스타 스키마에서 팩트 테이블의 정량적 데이터에 대한 데이터 속성을 포함하는 더 작은 테이블을 말합니다. 차원 테이블 속성은 일반적으로 텍스트 필드나 텍스트처럼 동작하는 개별 숫자입니다. 이러한 속성은 보통 쿼리 제약, 필터링 및 결과 세트 레이블 지정에 사용됩니다.

재해

워크로드 또는 시스템이 기본 배포 위치에서 비즈니스 목표를 달성하지 못하게 방해하는 이벤트입니다. 이러한 이벤트는 자연재해, 기술적 오류, 의도하지 않은 구성 오류 또는 멀웨어 공격과 같은 사람의 행동으로 인한 결과일 수 있습니다.

재해 복구(DR)

재해로 인한 가동 중지 시간 및 데이터 손실을 최소화하기 위해 사용하는 전략 및 프로세스입니다. 자세한 내용은 AWS Well-Architected Framework의 Disaster Recovery of Workloads on AWS: Recovery in the Cloud를 참조하세요.

DML

데이터베이스 조작 언어를 참조하세요.

도메인 기반 설계

구성 요소를 각 구성 요소가 제공하는 진화하는 도메인 또는 핵심 비즈니스 목표에 연결하여 복잡한 소프트웨어 시스템을 개발하는 접근 방식입니다. 이 개념은 에릭 에반스에 의해 그의 저서인 도메인 기반 디자인: 소프트웨어 중심의 복잡성 해결(Boston: Addison-Wesley Professional, 2003)에서 소개되었습니다. Strangler Fig 패턴과 함께 도메인 기반 설계를 사용하는 방법에 대한 자세한 내용은 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

DR

재해 복구를 참조하세요.

드리프트 감지

기준이 되는 구성과의 편차 추적을 말합니다. 예를 들어 AWS CloudFormation 를 사용하여 시스템 리소스의 드리프트를 감지하거나를 사용하여 AWS Control Tower 거버넌스 요구 사항 준수에 영향을 미칠 수 있는 랜딩 존의 변경 사항을 감지할 수 있습니다.

DVSM

개발 가치 흐름 매핑을 참조하세요.

E

EDA

탐색 데이터 분석을 참조하세요.

EDI

전자 데이터 교환을 참조하세요.

엣지 컴퓨팅

IoT 네트워크의 엣지에서 스마트 디바이스의 컴퓨팅 성능을 개선하는 기술 엣지 컴퓨팅은 클라우드 컴퓨팅에 비해 보다 통신 지연 시간을 줄이고 응답 시간을 개선할 수 있습니다.

전자 데이터 교환(EDI)

조직 간 비즈니스 문서의 자동화된 교환을 나타냅니다. 자세한 내용은 전자 데이터 교환(EDI)이란 무엇인가요?를 참조하세요.

암호화

사람이 읽을 수 있는 일반 텍스트 데이터를 사이퍼텍스트로 변환하는 컴퓨팅 프로세스입니다.

암호화 키

암호화 알고리즘에 의해 생성되는 무작위 비트의 암호화 문자열입니다. 키의 길이는 다양할 수 있으며 각 키는 예측할 수 없고 고유하게 설계되었습니다.

엔디안

컴퓨터 메모리에 바이트가 저장되는 순서입니다. 빅 엔디안 시스템은 가장 중요한 바이트를 먼저 저장합니다. 리틀 엔디안 시스템은 가장 덜 중요한 바이트를 먼저 저장합니다.

엔드포인트

서비스 엔드포인트를 참조하세요.

엔드포인트 서비스

Virtual Private Cloud(VPC)에서 호스팅하여 다른 사용자와 공유할 수 있는 서비스입니다. 를 사용하여 엔드포인트 서비스를 생성하고 다른 AWS 계정 또는 AWS Identity and Access Management (IAM) 보안 주체에 권한을 AWS PrivateLink 부여할 수 있습니다. 이러한 계정 또는 보안 주체는 인터페이스 VPC 엔드포인트를 생성하여 엔드포인트 서비스에 비공개로 연결할 수 있습니다. 자세한 내용은 Amazon Virtual Private Cloud(VPC) 설명서의 엔드포인트 서비스 생성을 참조하십시오.

엔터프라이즈 리소스 계획(ERP)

엔터프라이즈의 주요 비즈니스 프로세스(예: 회계, MES, 프로젝트 관리)를 자동화하고 관리하는 시스템입니다.

봉투 암호화

암호화 키를 다른 암호화 키로 암호화하는 프로세스입니다. 자세한 내용은 AWS Key Management Service (AWS KMS) 설명서의 봉투 암호화를 참조하세요.

환경

실행 중인 애플리케이션의 인스턴스입니다. 다음은 클라우드 컴퓨팅의 일반적인 환경 유형입니다.

- 개발 환경 - 애플리케이션 유지 관리를 담당하는 핵심 팀만 사용할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. 개발 환경은 변경 사항을 상위 환경으로 승격하기 전에 테스트하는 데 사용됩니다. 이러한 유형의 환경을 테스트 환경이라고도 합니다.
- 하위 환경 - 초기 빌드 및 테스트에 사용되는 환경을 비롯한 애플리케이션의 모든 개발 환경입니다.
- 프로덕션 환경 - 최종 사용자가 액세스할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. CI/CD 파이프라인에서 프로덕션 환경이 마지막 배포 환경입니다.
- 상위 환경 - 핵심 개발 팀 이외의 사용자가 액세스할 수 있는 모든 환경입니다. 프로덕션 환경, 프로덕션 이전 환경 및 사용자 수용 테스트를 위한 환경이 여기에 포함될 수 있습니다.

에픽

애자일 방법론에서 작업을 구성하고 우선순위를 정하는 데 도움이 되는 기능적 범주입니다. 에픽은 요구 사항 및 구현 작업에 대한 개괄적인 설명을 제공합니다. 예를 들어, AWS CAF 보안 에픽에는 ID 및 액세스 관리, 탐지 제어, 인프라 보안, 데이터 보호 및 인시던트 대응이 포함됩니다. AWS 마이그레이션 전략의 에픽에 대한 자세한 내용은 프로그램 구현 가이드를 참조하십시오.

ERP

엔터프라이즈 리소스 계획을 참조하세요.

탐색 데이터 분석(EDA)

데이터 세트를 분석하여 주요 특성을 파악하는 프로세스입니다. 데이터를 수집 또는 집계한 다음 초기 조사를 수행하여 패턴을 찾고, 이상을 탐지하고, 가정을 확인합니다. EDA는 요약 통계를 계산하고 데이터 시각화를 생성하여 수행됩니다.

F

팩트 테이블

스타 스키마의 중앙 테이블입니다. 비즈니스 운영에 대한 정량적 데이터를 저장합니다. 일반적으로 팩트 테이블은 측정값이 있는 열 및 차원 테이블에 대한 외래 키가 있는 열과 같이 두 가지 열 유형을 포함합니다.

빠른 실패

개발 수명 주기를 줄이기 위해 빈번한 증분 테스트를 사용하는 철학입니다. 애자일 접근 방식의 핵심입니다.

장애 격리 경계

에서 장애의 영향을 제한하고 워크로드의 복원력을 개선하는 데 도움이 되는 가용 영역, AWS 리전 컨트롤 플레인 또는 데이터 플레인과 같은 AWS 클라우드경계입니다. 자세한 내용은 AWS 장애 격리 경계를 참조하세요.

기능 브랜치

브랜치를 참조하세요.

기능

예측에 사용하는 입력 데이터입니다. 예를 들어, 제조 환경에서 기능은 제조 라인에서 주기적으로 캡처되는 이미지일 수 있습니다.

기능 중요도

모델의 예측에 특성이 얼마나 중요한지를 나타냅니다. 이는 일반적으로 SHAP(Shapley Additive Descriptions) 및 통합 그레디언트와 같은 다양한 기법을 통해 계산할 수 있는 수치 점수로 표현됩니다. 자세한 내용은 기계 학습 모델 해석 가능성을 참조하세요 AWS.

기능 변환

추가 소스로 데이터를 보강하거나, 값을 조정하거나, 단일 데이터 필드에서 여러 정보 세트를 추출하는 등 ML 프로세스를 위해 데이터를 최적화하는 것입니다. 이를 통해 ML 모델이 데이터를 활용

할 수 있습니다. 예를 들어, 날짜 '2021-05-27 00:15:37'을 '2021년', '5월', '목', '15일'로 분류하면 학습 알고리즘이 다양한 데이터 구성 요소와 관련된 미묘한 패턴을 학습하는 데 도움이 됩니다.

퓨샷 프롬프팅

유사한 태스크를 수행하도록 요청하기 전에 LLM에 태스크와 원하는 출력을 보여주는 몇 가지 예제를 제공합니다. 이 기법은 모델이 프롬프트에 포함된 예제(샷)에서 학습하는 컨텍스트 내 학습을 적용합니다. 퓨샷 프롬프팅은 특정 형식 지정, 추론 또는 분야별 지식이 필요한 태스크에 효과적일 수 있습니다. 제로샷 프롬프팅도 참조하세요.

FGAC

세분화된 액세스 제어를 참조하세요.

세분화된 액세스 제어(FGAC)

여러 조건을 사용하여 액세스 요청을 허용하거나 거부합니다.

플래시컷 마이그레이션

단계적 접근 방식을 사용하는 대신 변경 데이터 캡처를 통해 지속적 데이터 복제를 사용하여 최단 시간에 데이터를 마이그레이션하는 데이터베이스 마이그레이션 방법입니다. 목표는 가동 중지 시간을 최소화하는 것입니다.

FM

파운데이션 모델을 참조하세요.

파운데이션 모델(FM)

일반화되고 레이블이 지정되지 않은 데이터의 대규모 데이터세트에서 훈련된 대규모 딥 러닝 신경망입니다. FM은 언어 이해, 텍스트 및 이미지 생성, 자연어 대화와 같은 다양한 일반 태스크를 수행할 수 있습니다. 자세한 내용은 파운데이션 모델이란?을 참조하세요.

G

생성형 AI

대량의 데이터에서 훈련되었으며 간단한 텍스트 프롬프트를 사용하여 이미지, 비디오, 텍스트, 오디오와 같은 새 콘텐츠와 아티팩트를 생성할 수 있는 AI 모델의 하위 세트입니다. 자세한 내용은 생성형 AI란 무엇인가요?를 참조하세요.

지리적 차단

지리적 제한을 참조하세요.

지리적 제한(지리적 차단)

Amazon CloudFront에서 특정 국가의 사용자가 콘텐츠 배포에 액세스하지 못하도록 하는 옵션입니다. 허용 목록 또는 차단 목록을 사용하여 승인된 국가와 차단된 국가를 지정할 수 있습니다. 자세한 내용은 CloudFront 설명서의 콘텐츠의 지리적 배포 제한을 참조하십시오.

Gitflow 워크플로

하위 환경과 상위 환경이 소스 코드 리포지토리의 서로 다른 브랜치를 사용하는 방식입니다. Gitflow 워크플로는 레거시로 간주되며 트렁크 기반 워크플로는 선호되는 현대적 접근 방식입니다.

골든 이미지

시스템 또는 소프트웨어의 새 인스턴스를 배포하기 위한 템플릿으로 사용되는 해당 시스템 또는 소프트웨어의 스냅샷입니다. 예를 들어 제조 분야에서는 골든 이미지를 사용하여 여러 디바이스에서 소프트웨어를 프로비저닝할 수 있으며 이를 통해 딥이스 제조 작업의 속도, 확장성 및 생산성을 개선할 수 있습니다.

브라운필드 전략

새로운 환경에서 기존 인프라의 부재 시스템 아키텍처에 대한 그린필드 전략을 채택할 때 브라운필드라고도 하는 기존 인프라와의 호환성 제한 없이 모든 새로운 기술을 선택할 수 있습니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

가드레일

조직 단위(OU) 전체에서 리소스, 정책 및 규정 준수를 관리하는 데 도움이 되는 중요 규칙입니다. 예방 가드레일은 규정 준수 표준에 부합하도록 정책을 시행하며, 서비스 제어 정책과 IAM 권한 경계를 사용하여 구현됩니다. 탐지 가드레일은 정책 위반 및 규정 준수 문제를 감지하고 해결을 위한 알림을 생성하며, 이는 AWS Config Amazon GuardDuty AWS Security Hub CSPM, , AWS Trusted Advisor Amazon Inspector 및 사용자 지정 AWS Lambda 검사를 사용하여 구현됩니다.

H

HA

고가용성을 참조하세요.

이기종 데이터베이스 마이그레이션

다른 데이터베이스 엔진을 사용하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Oracle에서 Amazon Aurora로) 이기종 마이그레이션은 일반적으로 리아키텍트 작업의 일부이며 스

키마를 변환하는 것은 복잡한 작업일 수 있습니다.AWS 는 스키마 변환에 도움이 되는 AWS SCT를 제공합니다.

높은 가용성(HA)

문제나 재해 발생 시 개입 없이 지속적으로 운영할 수 있는 워크로드의 능력. HA 시스템은 자동으로 장애 조치되고, 지속적으로 고품질 성능을 제공하고, 성능에 미치는 영향을 최소화하면서 다양한 부하와 장애를 처리하도록 설계되었습니다.

히스토리언 현대화

제조 산업의 요구 사항을 더 잘 충족하도록 운영 기술(OT) 시스템을 현대화하고 업그레이드하는 데 사용되는 접근 방식입니다. 히스토리언은 공장의 다양한 출처에서 데이터를 수집하고 저장하는 데 사용되는 일종의 데이터베이스입니다.

홀드아웃 데이터

기계 학습 모델을 훈련하는 데 사용되는 데이터세트에서 보류되는 레이블이 지정된 기록 데이터의 일부입니다. 홀드아웃 데이터를 사용하여 모델 예측을 홀드아웃 데이터와 비교해 모델 성능을 평가할 수 있습니다.

동종 데이터베이스 마이그레이션

동일한 데이터베이스 엔진을 공유하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Microsoft SQL Server에서 Amazon RDS for SQL Server로) 동종 마이그레이션은 일반적으로 리호스팅 또는 리플랫포밍 작업의 일부입니다. 네이티브 데이터베이스 유틸리티를 사용하여 스키마를 마이그레이션할 수 있습니다.

핫 데이터

자주 액세스하는 데이터(예: 실시간 데이터 또는 최근 번역 데이터). 일반적으로 이 데이터에는 빠른 쿼리 응답을 제공하기 위한 고성능 스토리지 계층 또는 클래스가 필요합니다.

핫픽스

프로덕션 환경의 중요한 문제를 해결하기 위한 긴급 수정입니다. 핫픽스는 긴급하기 때문에 일반적인 DevOps 릴리스 워크플로 외부에서 실행됩니다.

하이퍼케어 기간

전환 직후 마이그레이션 팀이 문제를 해결하기 위해 클라우드에서 마이그레이션된 애플리케이션을 관리하고 모니터링하는 기간입니다. 일반적으로 이 기간은 1~4일입니다. 하이퍼케어 기간이 끝나면 마이그레이션 팀은 일반적으로 애플리케이션에 대한 책임을 클라우드 운영 팀에 넘깁니다.

I

IaC

코드형 인프라를 참조하세요.

자격 증명 기반 정책

AWS 클라우드 환경 내에서 권한을 정의하는 하나 이상의 IAM 보안 주체에 연결된 정책입니다.

유휴 애플리케이션

90일 동안 평균 CPU 및 메모리 사용량이 5~20%인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하거나 온프레미스에 유지하는 것이 일반적입니다.

IIoT

산업용 사물 인터넷을 참조하세요.

변경 불가능한 인프라

기존 인프라를 업데이트, 패치 또는 수정하는 대신 프로덕션 워크로드에 대한 새 인프라를 배포하는 모델입니다. 변경 불가능한 인프라는 변경 가능한 인프라보다 본질적으로 더 일관되고 안정적이며 예측 가능합니다. 자세한 내용은 AWS Well-Architected Framework의 변경 불가능한 인프라를 사용하여 배포 모범 사례를 참조하세요.

인바운드(수신) VPC

AWS 다중 계정 아키텍처에서 애플리케이션 외부에서 네트워크 연결을 수락, 검사 및 라우팅하는 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

증분 마이그레이션

한 번에 전체 전환을 수행하는 대신 애플리케이션을 조금씩 마이그레이션하는 전환 전략입니다. 예를 들어, 처음에는 소수의 마이크로서비스나 사용자만 새 시스템으로 이동할 수 있습니다. 모든 것이 제대로 작동하는지 확인한 후에는 레거시 시스템을 폐기할 수 있을 때까지 추가 마이크로서비스 또는 사용자를 점진적으로 이동할 수 있습니다. 이 전략을 사용하면 대규모 마이그레이션과 관련된 위험을 줄일 수 있습니다.

Industry 4.0

연결성, 실시간 데이터, 자동화, 분석 및 AI/ML의 발전을 통해 제조 프로세스의 현대화를 나타내기 위해 2016년에 Klaus Schwab에서 도입한 용어입니다.

인프라

애플리케이션의 환경 내에 포함된 모든 리소스와 자산입니다.

코드형 인프라(IaC)

구성 파일 세트를 통해 애플리케이션의 인프라를 프로비저닝하고 관리하는 프로세스입니다. IaC는 새로운 환경의 반복 가능성, 신뢰성 및 일관성을 위해 인프라 관리를 중앙 집중화하고, 리소스를 표준화하고, 빠르게 확장할 수 있도록 설계되었습니다.

산업용 사물 인터넷(IIoT)

제조, 에너지, 자동차, 의료, 생명과학, 농업 등의 산업 부문에서 인터넷에 연결된 센서 및 디바이스의 사용 자세한 내용은 산업용 사물 인터넷(IoT) 디지털 트랜스포메이션 전략 구축을 참조하십시오.

검사 VPC

AWS 다중 계정 아키텍처에서는 VPC(동일하거나 다른 AWS 리전), 인터넷 및 온프레미스 네트워크 간의 네트워크 트래픽 검사를 관리하는 중앙 집중식 VPCs입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

사물 인터넷(IoT)

인터넷이나 로컬 통신 네트워크를 통해 다른 디바이스 및 시스템과 통신하는 센서 또는 프로세서가 내장된 연결된 물리적 객체의 네트워크 자세한 내용은 IoT란?을 참조하십시오.

해석력

모델의 예측이 입력에 따라 어떻게 달라지는지를 사람이 이해할 수 있는 정도를 설명하는 기계 학습 모델의 특성입니다. 자세한 내용은 기계 학습 모델 해석 가능성을 참조하세요 AWS.

IoT

사물 인터넷을 참조하세요.

IT 정보 라이브러리(ITIL)

IT 서비스를 제공하고 이러한 서비스를 비즈니스 요구 사항에 맞게 조정하기 위한 일련의 모범 사례 ITIL은 ITSM의 기반을 제공합니다.

IT 서비스 관리(ITSM)

조직의 IT 서비스 설계, 구현, 관리 및 지원과 관련된 활동 클라우드 운영을 ITSM 도구와 통합하는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

ITIL

IT 정보 라이브러리를 참조하세요.

ITSM

IT 서비스 관리를 참조하세요.

L

레이블 기반 액세스 제어(LBAC)

사용자 및 데이터 자체에 각각 보안 레이블 값을 명시적으로 할당하는 필수 액세스 제어(MAC)를 구현한 것입니다. 사용자 보안 레이블과 데이터 보안 레이블 간의 교차 부분에 따라 사용자가 볼 수 있는 행과 열이 결정됩니다.

랜딩 존

랜딩 존은 확장 가능하고 안전한 잘 설계된 다중 계정 AWS 환경입니다. 조직은 여기에서부터 보안 및 인프라 환경에 대한 확신을 가지고 워크로드와 애플리케이션을 신속하게 시작하고 배포할 수 있습니다. 랜딩 존에 대한 자세한 내용은 안전하고 확장 가능한 다중 계정 AWS 환경 설정을 참조하십시오.

대규모 언어 모델(LLM)

방대한 양의 데이터에서 사전 훈련된 딥 러닝 AI 모델입니다. LLM은 질문에 대한 답변, 문서 요약, 텍스트를 다른 언어로 번역, 문장 완성과 같은 여러 태스크를 수행할 수 있습니다. 자세한 내용은 대규모 언어 모델(LLM)이란 무엇인가요?를 참조하세요.

대규모 마이그레이션

300대 이상의 서버 마이그레이션입니다.

LBAC

레이블 기반 액세스 제어를 참조하세요.

최소 권한

작업을 수행하는 데 필요한 최소 권한을 부여하는 보안 모범 사례입니다. 자세한 내용은 IAM 설명서의 최소 권한 적용을 참조하십시오.

리프트 앤드 시프트

7R을 참조하세요.

리틀 엔디안 시스템

가장 덜 중요한 바이트를 먼저 저장하는 시스템입니다. 엔디안도 참조하세요.

LLM

대규모 언어 모델을 참조하세요.

하위 환경

환경을 참조하세요.

M

기계 학습(ML)

패턴 인식 및 학습에 알고리즘과 기법을 사용하는 인공 지능의 한 유형입니다. ML은 사물 인터넷(IoT) 데이터와 같은 기록된 데이터를 분석하고 학습하여 패턴을 기반으로 통계 모델을 생성합니다. 자세한 내용은 기계 학습을 참조하십시오.

기본 브랜치

브랜치를 참조하세요.

맬웨어

컴퓨터 보안 또는 프라이버시를 위협하도록 설계된 소프트웨어입니다. 맬웨어는 컴퓨터 시스템을 방해하거나 민감한 정보를 유출하거나 무단 액세스 권한을 확보할 수 있습니다. 맬웨어의 예로 바이러스, 웜, 랜섬웨어, 트로이 목마, 스파이웨어, 키로거 등이 있습니다.

관리형 서비스

AWS 서비스 는 인프라 계층, 운영 체제 및 플랫폼을 AWS 운영하고, 사용자는 엔드포인트에 액세스하여 데이터를 저장하고 검색합니다. 관리형 서비스의 예로 Amazon Simple Storage Service(Amazon S3) 및 Amazon DynamoDB가 있습니다. 이를 추상화된 서비스라고도 합니다.

제조 실행 시스템(MES)

원자재를 생산 현장에서 완제품으로 변환하는 생산 프로세스를 추적, 모니터링, 문서화 및 제어하기 위한 소프트웨어 시스템입니다.

MAP

Migration Acceleration Program을 참조하세요.

메커니즘

도구를 생성하고 도구 채택을 유도한 다음 조정을 위해 결과를 검사하는 전체 프로세스입니다. 메커니즘은 작동 시 자체적으로 강화하고 개선하는 주기입니다. 자세한 내용은 AWS Well-Architected Framework의 메커니즘 구축을 참조하세요.

멤버 계정

조직의 일부인 관리 계정을 AWS 계정 제외한 모든 계정. AWS Organizations하나의 계정은 한 번에 하나의 조직 멤버만 될 수 있습니다.

MES

제조 실행 시스템을 참조하세요.

메시지 큐 원격 분석 전송(MQTT)

리소스 제약이 있는 IoT 디바이스에 대한 게시 및 구독 패턴을 기반으로 하는 경량 Machine-to-Machine(M2M) 통신 프로토콜입니다.

마이크로서비스

잘 정의된 API를 통해 통신하고 일반적으로 소규모 자체 팀이 소유하는 소규모 독립 서비스입니다. 예를 들어, 보험 시스템에는 영업, 마케팅 등의 비즈니스 역량이나 구매, 청구, 분석 등의 하위 영역에 매핑되는 마이크로 서비스가 포함될 수 있습니다. 마이크로서비스의 이점으로 민첩성, 유연한 확장, 손쉬운 배포, 재사용 가능한 코드, 복원력 등이 있습니다. 자세한 내용은 AWS 서버리스 서비스를 사용하여 마이크로서비스 통합을 참조하세요.

마이크로서비스 아키텍처

각 애플리케이션 프로세스를 마이크로서비스로 실행하는 독립 구성 요소를 사용하여 애플리케이션을 구축하는 접근 방식입니다. 이러한 마이크로서비스는 경량 API를 사용하여 잘 정의된 인터페이스를 통해 통신합니다. 애플리케이션의 특정 기능에 대한 수요에 맞게 이 아키텍처의 각 마이크로서비스를 업데이트, 배포 및 조정할 수 있습니다. 자세한 내용은 에서 마이크로서비스 구현을 참조하세요 AWS.

Migration Acceleration Program(MAP)

조직이 클라우드로 전환하기 위한 강력한 운영 기반을 구축하고 초기 마이그레이션 비용을 상쇄하는 데 도움이 되는 컨설팅 지원, 교육 및 서비스를 제공하는 AWS 프로그램입니다. MAP에는 레거시 마이그레이션을 체계적인 방식으로 실행하기 위한 마이그레이션 방법론과 일반적인 마이그레이션 시나리오를 자동화하고 가속화하는 도구 세트가 포함되어 있습니다.

대규모 마이그레이션

애플리케이션 포트폴리오의 대다수를 웨이브를 통해 클라우드로 이동하는 프로세스로, 각 웨이브에서 더 많은 애플리케이션이 더 빠른 속도로 이동합니다. 이 단계에서는 이전 단계에서 배운 모범 사례와 교훈을 사용하여 팀, 도구 및 프로세스의 마이그레이션 팩토리를 구현하여 자동화 및 민첩한 제공을 통해 워크로드 마이그레이션을 간소화합니다. 이것은 AWS 마이그레이션 전략의 세 번째 단계입니다.

마이그레이션 팩토리

자동화되고 민첩한 접근 방식을 통해 워크로드 마이그레이션을 간소화하는 다기능 팀입니다. 마이그레이션 팩토리 팀에는 일반적으로 스프린트에서 일하는 운영, 비즈니스 분석가 및 소유자, 마이그레이션 엔지니어, 개발자, DevOps 전문가가 포함됩니다. 엔터프라이즈 애플리케이션 포트폴리오의 20~50%는 공장 접근 방식으로 최적화할 수 있는 반복되는 패턴으로 구성되어 있습니다. 자세한 내용은 이 콘텐츠 세트의 클라우드 마이그레이션 팩토리 가이드와 마이그레이션 팩토리에 대한 설명을 참조하십시오.

마이그레이션 메타데이터

마이그레이션을 완료하는 데 필요한 애플리케이션 및 서버에 대한 정보 각 마이그레이션 패턴에는 서로 다른 마이그레이션 메타데이터 세트가 필요합니다. 마이그레이션 메타데이터의 예로는 대상 서브넷, 보안 그룹 및 AWS 계정이 있습니다.

마이그레이션 패턴

사용되는 마이그레이션 전략, 마이그레이션 대상, 마이그레이션 애플리케이션 또는 서비스를 자세히 설명하는 반복 가능한 마이그레이션 작업입니다. 예: AWS Application Migration Service를 사용하여 Amazon EC2로 마이그레이션을 리호스팅합니다.

Migration Portfolio Assessment(MPA)

AWS 클라우드로 마이그레이션하는 비즈니스 사례를 검증하기 위한 정보를 제공하는 온라인 도구입니다. MPA는 상세한 포트폴리오 평가(서버 적정 규모 조정, 가격 책정, TCO 비교, 마이그레이션 비용 분석)와 마이그레이션 계획(애플리케이션 데이터 분석 및 데이터 수집, 애플리케이션 그룹화, 마이그레이션 우선순위 지정, 웨이브 계획)을 제공합니다. MPA 도구(로그인 필요)는 모든 AWS 컨설턴트와 APN 파트너 컨설턴트가 무료로 사용할 수 있습니다.

마이그레이션 준비 상태 평가(MRA)

AWS CAF를 사용하여 조직의 클라우드 준비 상태에 대한 인사이트를 얻고, 강점과 약점을 식별하고, 식별된 격차를 해소하기 위한 행동 계획을 수립하는 프로세스입니다. 자세한 내용은 마이그레이션 준비 가이드를 참조하십시오. MRA는 AWS 마이그레이션 전략의 첫 번째 단계입니다.

마이그레이션 전략

워크로드를 AWS 클라우드로 마이그레이션하는 데 사용되는 접근 방식입니다. 자세한 내용은 이 용어집의 7R 항목과 조직을 동원하여 대규모 마이그레이션 가속화를 참조하세요.

ML

기계 학습을 참조하세요.

현대화

비용을 절감하고 효율성을 높이고 혁신을 활용하기 위해 구식(레거시 또는 모놀리식) 애플리케이션과 해당 인프라를 클라우드의 민첩하고 탄력적이고 가용성이 높은 시스템으로 전환하는 것입니다. 자세한 내용은 AWS 클라우드에서 애플리케이션을 현대화하기 위한 전략을 참조하세요.

현대화 준비 상태 평가

조직 애플리케이션의 현대화 준비 상태를 파악하고, 이점, 위험 및 종속성을 식별하고, 조직이 해당 애플리케이션의 향후 상태를 얼마나 잘 지원할 수 있는지를 확인하는 데 도움이 되는 평가입니다. 평가 결과는 대상 아키텍처의 청사진, 현대화 프로세스의 개발 단계와 마일스톤을 자세히 설명하는 로드맵 및 파악된 격차를 해소하기 위한 실행 계획입니다. 자세한 내용은 AWS 클라우드에서 애플리케이션의 현대화 준비 상태 평가를 참조하세요.

모놀리식 애플리케이션(모놀리식 유형)

긴밀하게 연결된 프로세스를 사용하여 단일 서비스로 실행되는 애플리케이션입니다. 모놀리식 애플리케이션에는 몇 가지 단점이 있습니다. 한 애플리케이션 기능에 대한 수요가 급증하면 전체 아키텍처 규모를 조정해야 합니다. 코드 베이스가 커지면 모놀리식 애플리케이션의 기능을 추가하거나 개선하는 것도 더 복잡해집니다. 이러한 문제를 해결하기 위해 마이크로서비스 아키텍처를 사용할 수 있습니다. 자세한 내용은 마이크로서비스로 모놀리식 유형 분해를 참조하십시오.

MPA

Migration Portfolio Assessment를 참조하세요.

MQTT

메시지 큐 원격 분석 전송을 참조하세요.

멀티클래스 분류

여러 클래스에 대한 예측(2개 이상의 결과 중 하나 예측)을 생성하는 데 도움이 되는 프로세스입니다. 예를 들어, ML 모델이 '이 제품은 책인가요, 자동차인가요, 휴대폰인가요?' 또는 '이 고객이 가장 관심을 갖는 제품 범주는 무엇인가요?'라고 물을 수 있습니다.

변경 가능한 인프라

프로덕션 워크로드에 대한 기존 인프라를 업데이트하고 수정하는 모델입니다. 일관성, 신뢰성 및 예측 가능성을 높이기 위해 AWS Well-Architected Framework에서는 변경 불가능한 인프라를 모범 사례로 사용할 것을 권장합니다.

O

OAC

오리진 액세스 제어를 참조하세요.

OAI

오리진 액세스 ID를 참조하세요.

OCM

조직 변경 관리를 참조하세요.

오프라인 마이그레이션

마이그레이션 프로세스 중 소스 워크로드가 중단되는 마이그레이션 방법입니다. 이 방법은 가동 중지 증가를 수반하며 일반적으로 작고 중요하지 않은 워크로드에 사용됩니다.

OI

운영 통합을 참조하세요.

OLA

운영 수준 계약을 참조하세요.

온라인 마이그레이션

소스 워크로드를 오프라인 상태로 전환하지 않고 대상 시스템에 복사하는 마이그레이션 방법입니다. 워크로드에 연결된 애플리케이션은 마이그레이션 중에도 계속 작동할 수 있습니다. 이 방법은 가동 중지 차단 또는 최소화를 수반하며 일반적으로 중요한 프로덕션 워크로드에 사용됩니다.

OPC-UA

Open Process Communications - Unified Architecture(OPC-UA)를 참조하세요.

Open Process Communications - Unified Architecture(OPC-UA)

산업 자동화를 위한 Machine-to-Machine(M2M) 통신 프로토콜입니다. OPC-UA는 데이터 암호화, 인증 및 권한 부여 체계에 관한 상호 운용성 표준을 제공합니다.

운영 수준 협약(OLA)

서비스 수준에 관한 계약(SLA)을 지원하기 위해 직무 IT 그룹이 서로에게 제공하기로 약속한 내용을 명확히 하는 계약입니다.

운영 준비 상태 검토(ORR)

인시던트 및 잠재적 장애의 범위를 이해, 평가 또는 예방하거나 줄이는 데 도움이 되는 질문 체크리스트 및 관련 모범 사례입니다. 자세한 내용은 AWS Well-Architected Framework의 운영 준비 상태 검토(ORR)를 참조하세요.

운영 기술(OT)

물리적 환경에서 작동하여 산업 운영, 장비 및 인프라를 제어하는 하드웨어 및 소프트웨어 시스템입니다. 제조 분야에서 OT 및 정보 기술(IT) 시스템의 통합은 Industry 4.0 트랜스포메이션의 주요 중점 사항입니다.

운영 통합(OI)

클라우드에서 운영을 현대화하는 프로세스로 준비 계획, 자동화 및 통합을 수반합니다. 자세한 내용은 운영 통합 가이드를 참조하십시오.

조직 트레일

조직 AWS 계정 내 모든에 대한 모든 이벤트를 로깅 AWS CloudTrail 하는에서 생성된 추적입니다 AWS Organizations. 이 트레일은 조직에 속한 각 AWS 계정 에 생성되고 각 계정의 활동을 추적합니다. 자세한 내용은 CloudTrail 설명서의 Creating a trail for an organization을 참조하십시오.

조직 변경 관리(OCM)

사람, 문화 및 리더십 관점에서 중대하고 파괴적인 비즈니스 혁신을 관리하기 위한 프레임워크입니다. OCM은 변화 채택을 가속화하고, 과도기적 문제를 해결하고, 문화 및 조직적 변화를 주도함으로써 조직이 새로운 시스템 및 전략을 준비하고 전환할 수 있도록 지원합니다. AWS 마이그레이션 전략에서는 클라우드 채택 프로젝트에 필요한 변경 속도 때문에이 프레임워크를 인력 가속화라고 합니다. 자세한 내용은 사용 가이드를 참조하십시오.

오리진 액세스 제어(OAC)

CloudFront에서 Amazon Simple Storage Service(S3) 콘텐츠를 보호하기 위해 액세스를 제한하는 고급 옵션입니다. OAC는 AWS KMS (SSE-KMS)를 사용한 모든 서버 측 암호화 AWS 리전와 S3 버킷에 대한 동적 PUT 및 DELETE 요청에서 모든 S3 버킷을 지원합니다.

오리진 액세스 ID(OAI)

CloudFront에서 Amazon S3 콘텐츠를 보호하기 위해 액세스를 제한하는 옵션입니다. OAI를 사용하면 CloudFront는 Amazon S3가 인증할 수 있는 보안 주체를 생성합니다. 인증된 보안 주체는 특

정 CloudFront 배포를 통해서만 S3 버킷의 콘텐츠에 액세스할 수 있습니다. 더 세분화되고 향상된 액세스 제어를 제공하는 OAC도 참조하십시오.

ORR

운영 준비 상태 검토를 참조하세요.

OT

운영 기술을 참조하세요.

아웃바운드(송신) VPC

AWS 다중 계정 아키텍처에서 애플리케이션 내에서 시작된 네트워크 연결을 처리하는 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

P

권한 경계

사용자나 역할이 가질 수 있는 최대 권한을 설정하기 위해 IAM 보안 주체에 연결되는 IAM 관리 정책입니다. 자세한 내용은 IAM 설명서의 권한 경계를 참조하십시오.

개인 식별 정보(PII)

직접 보거나 다른 관련 데이터와 함께 짝을 지을 때 개인의 신원을 합리적으로 추론하는 데 사용할 수 있는 정보입니다. PII의 예로는 이름, 주소, 연락처 정보 등이 있습니다.

PII

개인 식별 정보를 참조하세요.

플레이북

클라우드에서 핵심 운영 기능을 제공하는 등 마이그레이션과 관련된 작업을 캡처하는 일련의 사전 정의된 단계입니다. 플레이북은 스크립트, 자동화된 런북 또는 현대화된 환경을 운영하는 데 필요한 프로세스나 단계 요약의 형태를 취할 수 있습니다.

PLC

프로그래밍 가능 로직 컨트롤러를 참조하세요.

PLM

제품 수명 주기 관리를 참조하세요.

정책

권한 정의(ID 기반 정책 참조), 액세스 조건 지정(리소스 기반 정책 참조), AWS Organizations 내 조직의 모든 계정에 대한 최대 권한 정의(서비스 제어 정책 참조)와 같은 작업을 수행할 수 있는 객체입니다.

다국어 지속성

데이터 액세스 패턴 및 기타 요구 사항을 기반으로 독립적으로 마이크로서비스의 데이터 스토리지 기술 선택. 마이크로서비스가 동일한 데이터 스토리지 기술을 사용하는 경우 구현 문제가 발생하거나 성능이 저하될 수 있습니다. 요구 사항에 가장 적합한 데이터 저장소를 사용하면 마이크로서비스를 더 쉽게 구현하고 성능과 확장성을 높일 수 있습니다.

포트폴리오 평가

마이그레이션을 계획하기 위해 애플리케이션 포트폴리오를 검색 및 분석하고 우선순위를 정하는 프로세스입니다. 자세한 내용은 마이그레이션 준비 상태 평가를 참조하십시오.

조건자

보통 `WHERE` 절에 있는 `true` 또는 `false`를 반환하는 쿼리 조건입니다.

푸시다운 조건자

전송 전에 쿼리의 데이터를 필터링하는 데이터베이스 쿼리 최적화 기법입니다. 이렇게 하면 관계형 데이터베이스에서 검색하고 처리해야 하는 데이터의 양이 줄고 쿼리 성능이 향상됩니다.

예방적 제어

이벤트 발생을 방지하도록 설계된 보안 제어입니다. 이 제어는 네트워크에 대한 무단 액세스나 원치 않는 변경을 방지하는 데 도움이 되는 1차 방어선입니다. 자세한 내용은 Implementing security controls on AWS의 Preventative controls를 참조하십시오.

보안 주체

작업을 수행하고 리소스에 액세스할 수 AWS 있는의 엔터티입니다. 이 엔터티는 일반적으로 , AWS 계정 IAM 역할 또는 사용자의 루트 사용자입니다. 자세한 내용은 IAM 설명서의 역할 용어 및 개념의 보안 주체를 참조하십시오.

개인 정보 보호 중심 설계

전체 개발 프로세스에서 개인 정보를 고려하는 시스템 엔지니어링에서의 접근 방식입니다.

프라이빗 호스팅 영역

Amazon Route 53에서 하나 이상의 VPC 내 도메인과 하위 도메인에 대한 DNS 쿼리에 응답하는 방법에 대한 정보가 담긴 컨테이너입니다. 자세한 내용은 Route 53 설명서의 프라이빗 호스팅 영역 작업을 참조하십시오.

선제적 제어

규정 미준수 리소스의 배포를 방지하도록 설계된 보안 제어입니다. 이러한 제어는 리소스를 프로비저닝하기 전에 리소스를 스캔합니다. 리소스가 제어를 준수하지 않으면 프로비저닝되지 않습니다. 자세한 내용은 AWS Control Tower 설명서의 제어 참조 가이드를 참조하고 보안 제어 구현의 사전 예방적 제어를 참조하세요. AWS

제품 수명 주기 관리(PLM)

설계, 개발 및 출시부터 성장 및 성숙도를 거쳐 거부 및 제거에 이르기까지 전체 수명 주기 동안 제품의 데이터 및 프로세스 관리를 나타냅니다.

프로덕션 환경

환경을 참조하세요.

프로그래밍 가능 로직 컨트롤러(PLC)

제조 분야에서 기계를 모니터링하고 제조 프로세스를 자동화하는 매우 안정적이고 적응력이 뛰어난 컴퓨터입니다.

프롬프트 체이닝

한 LLM 프롬프트의 출력을 다음 프롬프트의 입력으로 사용하여 더 나은 응답을 생성합니다. 이 기법은 복잡한 태스크를 하위 태스크로 나누거나 예비 응답을 반복적으로 세부 조정하거나 확장하는 데 사용됩니다. 이를 통해 모델 응답의 정확성과 관련성을 개선하고 보다 세분화되고 개인화된 결과를 얻을 수 있습니다.

가명화

데이터세트의 개인 식별자를 자리 표시자 값으로 바꾸는 프로세스입니다. 가명화는 개인 정보를 보호하는 데 도움이 될 수 있습니다. 가명화된 데이터는 여전히 개인 데이터로 간주됩니다.

게시/구독(pub/sub)

여러 마이크로서비스에서 비동기 통신을 지원하여 확장성과 응답성을 개선하는 패턴입니다. 예를 들어 마이크로서비스 기반 MES에서 마이크로서비스는 다른 마이크로서비스가 구독할 수 있는 채널에 이벤트 메시지를 게시할 수 있습니다. 시스템은 게시 서비스를 변경하지 않고도 새 마이크로서비스를 추가할 수 있습니다.

Q

쿼리 계획

SQL 관계형 데이터베이스 시스템의 데이터에 액세스하는 데 사용되는 명령어와 같은 일련의 단계입니다.

쿼리 계획 회귀

데이터베이스 서비스 최적화 프로그램이 데이터베이스 환경을 변경하기 전보다 덜 최적의 계획을 선택하는 경우입니다. 통계, 제한 사항, 환경 설정, 쿼리 파라미터 바인딩 및 데이터베이스 엔진 업데이트의 변경으로 인해 발생할 수 있습니다.

R

RACI 매트릭스

Responsible, Accountable, Consulted, Informed(RACI)를 참조하세요.

RAG

검색 증강 생성을 참조하세요.

랜섬웨어

결제가 완료될 때까지 컴퓨터 시스템이나 데이터에 대한 액세스를 차단하도록 설계된 악성 소프트웨어입니다.

RASCI 매트릭스

Responsible, Accountable, Consulted, Informed(RACI)를 참조하세요.

RCAC

행 및 열 액세스 제어를 참조하세요.

읽기 전용 복제본

읽기 전용 용도로 사용되는 데이터베이스의 사본입니다. 쿼리를 읽기 전용 복제본으로 라우팅하여 기본 데이터베이스의 로드를 줄일 수 있습니다.

리아키텍팅

7R을 참조하세요.

Recovery Point Objective(RPO)

마지막 데이터 복구 시점 이후 허용되는 최대 시간입니다. 이에 따라 마지막 복구 시점과 서비스 중단 사이에 허용되는 데이터 손실로 간주되는 범위가 결정됩니다.

Recovery Time Objective(RTO)

서비스 중단과 서비스 복원 사이의 허용 가능한 지연 시간입니다.

리팩터링

7R을 참조하세요.

리전

지리적 영역의 AWS 리소스 모음입니다. 각 AWS 리전 는 내결함성, 안정성 및 복원력을 제공하기 위해 서로 격리되고 독립적입니다. 자세한 내용은 계정에서 사용할 수 있는 AWS 리전 지정을 참조하세요.

회귀

숫자 값을 예측하는 ML 기법입니다. 예를 들어, '이 집은 얼마에 팔릴까?'라는 문제를 풀기 위해 ML 모델은 선형 회귀 모델을 사용하여 주택에 대해 알려진 사실(예: 면적)을 기반으로 주택의 매매 가격을 예측할 수 있습니다.

리호스팅

7R을 참조하세요.

릴리스

배포 프로세스에서 변경 사항을 프로덕션 환경으로 승격시키는 행위입니다.

재배치

7R을 참조하세요.

리플랫포밍

7R을 참조하세요.

재구매

7R을 참조하세요.

복원력

중단에 저항하거나 중단을 복구할 수 있는 애플리케이션의 기능입니다. 고가용성 및 재해 복구는 AWS 클라우드에서 복원력을 계획할 때 일반적인 고려 사항입니다. 자세한 내용은 AWS 클라우드 복원력을 참조하세요.

리소스 기반 정책

Amazon S3 버킷, 엔드포인트, 암호화 키 등의 리소스에 연결된 정책입니다. 이 유형의 정책은 액세스가 허용된 보안 주체, 지원되는 작업 및 충족해야 하는 기타 조건을 지정합니다.

RACI(Responsible, Accountable, Consulted, Informed) 매트릭스

마이그레이션 활동 및 클라우드 운영에 참여하는 모든 당사자의 역할과 책임을 정의하는 매트릭스입니다. 매트릭스 이름은 매트릭스에 정의된 책임 유형에서 파생됩니다. 실무 담당자 (R), 의사 결정권자 (A), 업무 수행 조언자 (C), 결과 통보 대상자 (I). 지원자는 (S) 선택사항입니다. 지원자를 포함하면 매트릭스를 RASCI 매트릭스라고 하고, 지원자를 제외하면 RACI 매트릭스라고 합니다.

대응 제어

보안 기준에서 벗어나거나 부정적인 이벤트를 해결하도록 설계된 보안 제어입니다. 자세한 내용은 AWS에서 보안 제어 구현의 대응 제어를 참조하세요.

retain

7R을 참조하세요.

사용 중지

7R을 참조하세요.

검색 증강 세대(RAG)

응답을 생성하기 전에 LLM이 훈련 데이터 소스 외부에 있는 신뢰할 수 있는 데이터 소스를 참조하는 생성형 AI 기술입니다. 예를 들어 RAG 모델은 조직의 지식 기반 또는 사용자 지정 데이터에 대한 시맨틱 검색을 수행할 수 있습니다. 자세한 내용은 검색 증강 생성(RAG)이란 무엇인가요?를 참조하세요.

교체

공격자가 자격 증명에 액세스하는 것을 더욱 어렵게 만들기 위해 보안 암호를 주기적으로 업데이트하는 프로세스입니다.

행 및 열 액세스 제어(RCAC)

액세스 규칙이 정의된 기본적이고 유연한 SQL 표현식을 사용합니다. RCAC는 행 권한과 열 마스크로 구성됩니다.

RPO

목표 복구 시점(RPO)을 참조하세요.

RTO

목표 복구 시간(RTO)을 참조하세요.

런북

특정 작업을 수행하는 데 필요한 일련의 수동 또는 자동 절차입니다. 일반적으로 오류율이 높은 반복 작업이나 절차를 간소화하기 위해 런북을 만듭니다.

S

SAML 2.0

많은 ID 제공업체(idP)에서 사용하는 개방형 표준입니다. 이 기능을 사용하면 연동 SSO(Single Sign-On)를 AWS Management Console 사용할 수 있으므로 사용자는 조직의 모든 사용자에 대해 IAM에서 사용자를 생성하지 않고도에 로그인하거나 AWS API 작업을 호출할 수 있습니다. SAML 2.0 기반 페더레이션에 대한 자세한 내용은 IAM 설명서의 SAML 2.0 기반 페더레이션 정보를 참조하십시오.

SCADA

감독 제어 및 데이터 획득을 참조하세요.

SCP

서비스 제어 정책을 참조하세요.

보안 암호

에는 암호화된 형식으로 저장하는 암호 또는 사용자 자격 증명과 같은 AWS Secrets Manager기밀 또는 제한된 정보가 있습니다. 보안 암호 값과 메타데이터로 구성됩니다. 보안 암호 값은 바이너리, 단일 문자열 또는 여러 문자열일 수 있습니다. 자세한 내용은 AWS Secrets Manager 설명서의 Secrets Manager 보안 암호란 무엇인가요?를 참조하세요.

보안 중심 설계

전체 개발 프로세스에서 보안을 고려하는 시스템 엔지니어링에서의 접근 방식입니다.

보안 제어

위협 행위자가 보안 취약성을 악용하는 능력을 방지, 탐지 또는 감소시키는 기술적 또는 관리적 가드레일입니다. 보안 제어는 예방, 감지, 대응, 선제적과 같은 기본적인 네 가지 보안 제어 유형으로 구분됩니다.

보안 강화

공격 표면을 줄여 공격에 대한 저항력을 높이는 프로세스입니다. 더 이상 필요하지 않은 리소스 제거, 최소 권한 부여의 보안 모범 사례 구현, 구성 파일의 불필요한 기능 비활성화 등의 작업이 여기에 포함될 수 있습니다.

보안 정보 및 이벤트 관리(SIEM) 시스템

보안 정보 관리(SIM)와 보안 이벤트 관리(SEM) 시스템을 결합하는 도구 및 서비스입니다. SIEM 시스템은 서버, 네트워크, 디바이스 및 기타 소스에서 데이터를 수집, 모니터링 및 분석하여 위협과 보안 침해를 탐지하고 알림을 생성합니다.

보안 응답 자동화

보안 이벤트에 자동으로 응답하거나 이를 해결하도록 설계된 사전 정의되고 프로그래밍된 작업입니다. 이러한 자동화는 보안 모범 사례를 구현하는 데 도움이 되는 탐지 또는 대응 AWS 보안 제어 역할을 합니다. 자동화된 응답 작업의 예로 VPC 보안 그룹 수정, Amazon EC2 인스턴스 패치 적용 또는 자격 증명 교체 등이 있습니다.

서버 측 암호화

대상에서 데이터를 수신하는 AWS 서비스 에 의한 데이터 암호화.

서비스 제어 정책(SCP)

AWS Organizations에 속한 조직의 모든 계정에 대한 권한을 중앙 집중식으로 제어하는 정책입니다. SCP는 관리자가 사용자 또는 역할에 위임할 수 있는 작업에 대해 제한을 설정하거나 가드레일을 정의합니다. SCP를 허용 목록 또는 거부 목록으로 사용하여 허용하거나 금지할 서비스 또는 작업을 지정할 수 있습니다. 자세한 내용은 AWS Organizations 설명서의 서비스 제어 정책을 참조하세요.

서비스 엔드포인트

에 대한 진입점의 URL입니다 AWS 서비스. 엔드포인트를 사용하여 대상 서비스에 프로그래밍 방식으로 연결할 수 있습니다. 자세한 내용은 AWS 일반 참조의 AWS 서비스 엔드포인트를 참조하십시오.

서비스 수준에 관한 계약(SLA)

IT 팀이 고객에게 제공하기로 약속한 내용(예: 서비스 가동 시간 및 성능)을 명시한 계약입니다.

서비스 수준 지표(SLI)

오류 발생률, 가용성 또는 처리량과 같은 서비스의 성능 측면에 대한 측정값입니다.

서비스 수준 목표(SLO)

서비스 수준 지표로 측정되는 서비스의 상태를 나타내는 목표 지표입니다.

공동 책임 모델

클라우드 보안 및 규정 준수를 AWS 위해와 공유하는 책임을 설명하는 모델입니다. AWS 는 클라우드의 보안을 담당하는 반면, 사용자는 클라우드의 보안을 담당합니다. 자세한 내용은 공동 책임 모델을 참조하십시오.

SIEM

보안 정보 및 이벤트 관리 시스템을 참조하세요.

단일 장애점(SPOF)

애플리케이션을 중단시킬 수 있는 애플리케이션의 중요한 단일 구성 요소에서 발생하는 장애입니다.

SLA

서비스 수준 계약을 참조하세요.

SLI

서비스 수준 지표를 참조하세요.

SLO

서비스 수준 목표를 참조하세요.

분할 앤 시드 모델

현대화 프로젝트를 확장하고 가속화하기 위한 패턴입니다. 새로운 기능과 제품 릴리스가 정의되면 핵심 팀이 분할되어 새로운 제품 팀이 만들어집니다. 이를 통해 조직의 역량과 서비스 규모를 조정하고, 개발자 생산성을 개선하고, 신속한 혁신을 지원할 수 있습니다. 자세한 내용은 AWS 클라우드에서 애플리케이션을 현대화하기 위한 단계별 접근 방식을 참조하세요.

SPOF

단일 장애점을 참조하세요.

스타 스키마

하나의 큰 팩트 테이블을 사용하여 트랜잭션 또는 측정된 데이터를 저장하고 하나 이상의 더 작은 차원 테이블을 사용하여 데이터 속성을 저장하는 데이터베이스 조직 구조입니다. 이 구조는 데이터 웨어하우스에서 또는 비즈니스 인텔리전스 목적으로 사용하도록 설계되었습니다.

Strangler Fig 패턴

레거시 시스템을 폐기할 수 있을 때까지 시스템 기능을 점진적으로 다시 작성하고 교체하여 모놀리식 시스템을 현대화하기 위한 접근 방식. 이 패턴은 무화과 덩굴이 나무로 자라 결국 숙주를 압도하고 대체하는 것과 비슷합니다. Martin Fowler가 모놀리식 시스템을 다시 작성할 때 위험을 관리하는 방법으로 이 패턴을 도입했습니다. 이 패턴을 적용하는 방법의 예는 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

서브넷

VPC의 IP 주소 범위입니다. 서브넷은 단일 가용 영역에 상주해야 합니다.

감독 제어 및 데이터 획득(SCADA)

제조 분야에서 하드웨어와 소프트웨어를 사용하여 물리적 자산과 프로덕션 작업을 모니터링하는 시스템입니다.

대칭 암호화

동일한 키를 사용하여 데이터를 암호화하고 복호화하는 암호화 알고리즘입니다.

합성 테스트

사용자 상호 작용을 시뮬레이션하여 잠재적 문제를 감지하거나 성능을 모니터링하는 방식으로 진행되는 시스템 테스트입니다. Amazon CloudWatch Synthetics를 사용하여 이러한 테스트를 생성할 수 있습니다.

시스템 프롬프트

LLM에 컨텍스트, 명령 또는 지침을 제공하여 동작을 지시하는 기법입니다. 시스템 프롬프트는 컨텍스트를 설정하고 사용자와의 상호 작용을 위한 규칙을 설정하는 데 도움이 됩니다.

T

tags

AWS 리소스를 구성하기 위한 메타데이터 역할을 하는 키-값 페어입니다. 태그를 사용하면 리소스를 손쉽게 관리, 식별, 정리, 검색, 필터링할 수 있습니다. 자세한 내용은 AWS 리소스에 태그 지정을 참조하십시오.

대상 변수

지도 ML에서 예측하려는 값으로, 결과 변수라고도 합니다. 예를 들어, 제조 설정에서 대상 변수는 제품 결함일 수 있습니다.

작업 목록

런북을 통해 진행 상황을 추적하는 데 사용되는 도구입니다. 작업 목록에는 런북의 개요와 완료해야 할 일반 작업 목록이 포함되어 있습니다. 각 일반 작업에 대한 예상 소요 시간, 소유자 및 진행 상황이 작업 목록에 포함됩니다.

테스트 환경

환경을 참조하세요.

훈련

ML 모델이 학습할 수 있는 데이터를 제공하는 것입니다. 훈련 데이터에는 정답이 포함되어야 합니다. 학습 알고리즘은 훈련 데이터에서 대상(예측하려는 답)에 입력 데이터 속성을 매핑하는 패턴을 찾고, 이러한 패턴을 캡처하는 ML 모델을 출력합니다. 그런 다음 ML 모델을 사용하여 대상을 모르는 새 데이터에 대한 예측을 할 수 있습니다.

Transit Gateway

VPC와 온프레미스 네트워크를 상호 연결하는 데 사용할 수 있는 네트워크 전송 허브입니다. 자세한 내용은 AWS Transit Gateway 설명서의 전송 게이트웨이란 무엇입니까?를 참조하세요.

트렁크 기반 워크플로

개발자가 기능 브랜치에서 로컬로 기능을 구축하고 테스트한 다음 해당 변경 사항을 기본 브랜치에 병합하는 접근 방식입니다. 이후 기본 브랜치는 개발, 프로덕션 이전 및 프로덕션 환경에 순차적으로 구축됩니다.

신뢰할 수 있는 액세스

사용자를 대신하여 AWS Organizations 및 해당 계정에서 조직에서 작업을 수행하도록 지정하는 서비스에 대한 권한 부여. 신뢰할 수 있는 서비스는 필요할 때 각 계정에 서비스 연결 역할을 생성하여 관리 작업을 수행합니다. 자세한 내용은 설명서의 다른 AWS 서비스와 AWS Organizations 함께 사용을 참조하세요 AWS Organizations .

튜닝

ML 모델의 정확도를 높이기 위해 훈련 프로세스의 측면을 여러 변경하는 것입니다. 예를 들어, 레이블링 세트를 생성하고 레이블을 추가한 다음 다양한 설정에서 이러한 단계를 여러 번 반복하여 모델을 최적화하는 방식으로 ML 모델을 훈련할 수 있습니다.

피자 두 판 팀

피자 두 판이면 충분한 소규모 DevOps 팀. 피자 두 판 팀 규모는 소프트웨어 개발에 있어 가능한 최상의 공동 작업 기회를 보장합니다.

U

불확실성

예측 ML 모델의 신뢰성을 저해할 수 있는 부정확하거나 불완전하거나 알려지지 않은 정보를 나타내는 개념입니다. 불확실성에는 두 가지 유형이 있습니다. 인식론적 불확실성은 제한적이고 불완전한 데이터에 의해 발생하는 반면, 우연한 불확실성은 데이터에 내재된 노이즈와 무작위성에 의해 발생합니다. 자세한 내용은 Quantifying uncertainty in deep learning systems 가이드를 참조하십시오.

차별화되지 않은 작업

애플리케이션을 만들고 운영하는 데 필요하지만 최종 사용자에게 직접적인 가치를 제공하거나 경쟁 우위를 제공하지 못하는 작업을 헤비 리프팅이라고도 합니다. 차별화되지 않은 작업의 예로는 조달, 유지보수, 용량 계획 등이 있습니다.

상위 환경

환경을 참조하세요.

V

정리

스토리지를 회수하고 성능을 향상시키기 위해 증분 업데이트 후 정리 작업을 수반하는 데이터베이스 유지 관리 작업입니다.

버전 제어

리포지토리의 소스 코드 변경과 같은 변경 사항을 추적하는 프로세스 및 도구입니다.

VPC 피어링

프라이빗 IP 주소를 사용하여 트래픽을 라우팅할 수 있게 하는 두 VPC 간의 연결입니다. 자세한 내용은 Amazon VPC 설명서의 VPC 피어링이란?을 참조하십시오.

취약성

시스템 보안을 손상시키는 소프트웨어 또는 하드웨어 결함입니다.

W

웜 캐시

자주 액세스하는 최신 관련 데이터를 포함하는 버퍼 캐시입니다. 버퍼 캐시에서 데이터베이스 인스턴스를 읽을 수 있기 때문에 주 메모리나 디스크에서 읽는 것보다 빠릅니다.

웜 데이터

자주 액세스하지 않는 데이터입니다. 이런 종류의 데이터를 쿼리할 때는 일반적으로 적절히 느린 쿼리가 허용됩니다.

창 함수

현재 레코드와 어떤 식으로든 관련된 행 그룹에서 계산을 수행하는 SQL 함수입니다. 창 함수는 이동 평균을 계산하거나 현재 행의 상대적 위치를 기반으로 행 값에 액세스하는 등의 태스크를 처리하는 데 유용합니다.

워크로드

고객 대면 애플리케이션이나 백엔드 프로세스 같이 비즈니스 가치를 창출하는 리소스 및 코드 모음입니다.

워크스트림

마이그레이션 프로젝트에서 특정 작업 세트를 담당하는 직무 그룹입니다. 각 워크스트림은 독립적이지만 프로젝트의 다른 워크스트림을 지원합니다. 예를 들어, 포트폴리오 워크스트림은 애플리케이션 우선순위 지정, 웨이브 계획, 마이그레이션 메타데이터 수집을 담당합니다. 포트폴리오 워크스트림은 이러한 자산을 마이그레이션 워크스트림에 전달하고, 마이그레이션 워크스트림은 서버와 애플리케이션을 마이그레이션합니다.

WORM

Write Once, Read Many(WORM)를 참조하세요.

WQF

AWS Workload Qualification Framework를 참조하세요.

Write Once Read Many(WORM)

데이터를 한 번 쓰고 데이터가 삭제되거나 수정되지 않도록 하는 스토리지 모델입니다. 권한 있는 사용자는 필요한 만큼 여러 번 데이터를 읽을 수 있지만 데이터를 변경할 수는 없습니다. 이 데이터 스토리지 인프라는 변경 불가능한 항목으로 간주됩니다.

Z

제로데이 익스플로잇

[제로데이 취약성]을 악용하는 공격(일반적으로 맬웨어)입니다.

제로데이 취약성

프로덕션 시스템의 명백한 결함 또는 취약성입니다. 위협 행위자는 이러한 유형의 취약성을 사용하여 시스템을 공격할 수 있습니다. 개발자는 공격의 결과로 취약성을 인지하는 경우가 많습니다.

제로샷 프롬프팅

태스크를 수행하기 위해 [LLM]에 명령을 제공하지만 안내에 도움이 되는 예제(샷)는 제공하지 않습니다. LLM은 사전 훈련된 지식을 사용하여 태스크를 처리해야 합니다. 제로샷 프롬프팅의 효과는 태스크의 복잡성과 프롬프트의 품질에 따라 달라집니다. [퓨샷 프롬프팅]도 참조하세요.

좀비 애플리케이션

평균 CPU 및 메모리 사용량이 5% 미만인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하는 것이 일반적입니다.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.