AWS Whitepaper

Data Warehousing on AWS

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

Data Warehousing on AWS: AWS Whitepaper

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

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

This whitepaper is for historical reference only. Some content might be outdated and some links might not be available.

Data Warehousing on AWS

Publication date: **January 15, 2021** (*Document history and contributors*)

Enterprises across the globe want to migrate data warehousing to the cloud to improve performance and lower costs. This whitepaper discusses a modern approach to analytics and data warehousing architecture. It outlines services available on Amazon Web Services (AWS) to implement this architecture, and provides common design patterns to build data warehousing solutions using these services.

This whitepaper is aimed at data engineers, data analysts, business analysts, and developers.

Introduction

Data is an enterprise's most valuable asset. To fuel innovation, which fuels growth, an enterprise must:

- Store every relevant data point about their business
- Give data access to everyone who needs it
- Have the ability to analyze the data in different ways
- Distill the data down to insights

Most large enterprises have data warehouses for reporting and analytics purposes. They use data from a variety of sources, including their own transaction processing systems, and other databases.

In the past, building and running a data warehouse—a central repository of information coming from one or more data sources—was complicated and expensive. Data warehousing systems were complex to set up, cost millions of dollars in upfront software and hardware expenses, and took months of planning, procurement, implementation, and deployment processes. After making the initial investments and setting up the data warehouse, enterprises had to hire a team of database administrators to keep their queries running fast and protect against data loss.

Traditional data warehouse architectures and on-premises data warehousing pose many challenges:

- They are difficult to scale and have long lead times for hardware procurement and upgrades.
- They have high overhead costs for administration.

- Proprietary formats and siloed data make it costly and complex to access, refine, and join data from different sources.
- They cannot separate cold (infrequently used) and warm (frequently used) data, which results in bloated costs and wasted capacity.
- They limit the number of users and the amount of accessible data, which leads to anti-democratization of data.
- They inspire other legacy architecture patterns, such as retrofitting use cases to accommodate the wrong tools for the job, instead of using the correct tool for each use case.

In this whitepaper, we provide the information you need to take advantage of the strategic shift happening in the data warehousing space from on-premises to the cloud:

- Modern analytics architecture
- Data warehousing technology choices available within that architecture
- A deep dive on Amazon Redshift and its differentiating features
- A blueprint for building a complete data warehousing system on AWS with Amazon Redshift and other AWS services
- Practical tips for migrating from other data warehousing solutions and tapping into our partner ecosystem

Introducing Amazon Redshift

In the past, when data volumes grew or an enterprise wanted to make analytics and reports available to more users, they had to choose between accepting slow query performance or investing time and effort on an expensive upgrade process. In fact, some IT teams discourage augmenting data or adding queries to protect existing service-level agreements. Many enterprises struggled with maintaining a healthy relationship with traditional database vendors. They were often forced to either upgrade hardware for a managed system, or enter a protracted negotiation cycle for an expired term license. When they hit the scaling limit on one data warehouse engine, they were forced to migrate to another engine from the same vendor with different SQL semantics.

Cloud data warehouses like Amazon Redshift changed how enterprises think about data warehousing by dramatically lowering the cost and effort associated with deploying data warehouse systems, without compromising on features, scale, and performance.

Amazon Redshift is a fast, fully managed, petabyte-scale data warehousing solution that makes it simple and cost-effective to analyze large volumes of data using existing business intelligence (BI) tools. With Amazon Redshift, you can get the performance of columnar data warehousing engines that perform massively parallel processing (MPP) at a tenth of the cost. You can start small for $0.25 per hour, with no commitments, and scale to petabytes for $1,000 per terabyte per year. You can grow to exabyte-scale storage by storing data in an Amazon Simple Storage Service (Amazon S3) data lake and taking a lake house approach to data warehousing with the Amazon Redshift Spectrum feature. With this setup, you can query data directly from files on Amazon S3 for as low as $5 per terabyte of data scanned.

Since launching in February 2013, Amazon Redshift has been one of the fastest growing AWS Services, with tens of thousands of customers across many industries and company sizes. Enterprises such as NTT DOCOMO, FINRA, Johnson & Johnson, McDonalds, Equinox, Fannie Mae, Hearst, Amgen, and NASDAQ have migrated to Amazon Redshift.

Modern analytics and data warehousing architecture

Data typically flows into a data warehouse from transactional systems and other relational databases, and typically includes structured, semi-structured, and unstructured data. This data is processed, transformed, and ingested at a regular cadence. Users, including data scientists, business analysts, and decision-makers, access the data through BI tools, SQL clients, and other tools.

So why build a data warehouse at all? Why not just run analytics queries directly on an online transaction processing (OLTP) database, where the transactions are recorded? To answer the question, let's look at the differences between data warehouses and OLTP databases.

- **Data warehouses** are optimized for batched write operations and reading high volumes of data.
- **OLTP databases** are optimized for continuous write operations and high volumes of small read operations.

Data warehouses generally employ denormalized schemas like the Star schema and Snowflake schema because of high data throughput requirements, whereas OLTP databases employ highly normalized schemas, which are more suited for high transaction throughput requirements.

To get the benefits of using a data warehouse managed as a separate data store with your source OLTP or other source system, we recommend that you build an efficient data pipeline. Such a pipeline extracts the data from the source system, converts it into a schema suitable for data warehousing, and then loads it into the data warehouse. In the next section, we discuss the building blocks of an analytics pipeline and the different AWS services you can use to architect the pipeline.

AWS analytics services

AWS analytics services help enterprises quickly convert their data to answers by providing mature and integrated analytics services, ranging from cloud data warehouses to serverless data lakes. Getting answers quickly means less time building plumbing and configuring cloud analytics services to work together. AWS helps you do exactly that by giving you:

- An easy path to build data lakes and data warehouses, and start running diverse analytics workloads.

- A secure cloud storage, compute, and network infrastructure that meets the specific needs of analytic workloads.
- A fully integrated analytics stack with a mature set of analytics tools, covering all common use cases and leveraging open file formats, standard SQL language, open-source engines, and platforms.
- The best performance, the most scalability, and the lowest cost for analytics.

Many enterprises choose cloud data lakes and cloud data warehouses as the foundation for their data and analytics architectures. AWS is focused on helping customers build and secure data lakes and data warehouses in the cloud within days, not months. AWS Lake Formation enables secured, self-service discovery and access for users. Lake Formation provides easy, on-demand access to specific resources that fit the requirements of each analytics workload. The data is curated and cataloged, already prepared for any type of analytics. Related records are matched and de-duplicated with machine learning.

AWS provides a diverse set of analytics services that are deeply integrated with the infrastructure layers. This enables you to take advantage of features like intelligent tiering and Amazon Elastic Compute Cloud (Amazon EC2) spot instances, to reduce cost and run analytics faster. When you're ready for more advanced analytic approaches, use our broad collection of machine learning (ML) and artificial intelligence (AI) services against that same data in S3 to gain even more insight without the delays and costs of moving or transforming your data.

Analytics architecture

Analytics pipelines are designed to handle large volumes of incoming streams of data from heterogeneous sources such as databases, applications, and devices.

A typical analytics pipeline has the following stages:

1. Collect data
2. Store the data
3. Process the data
4. Analyze and visualize the data

Analytics pipeline

Data collection

At the data collection stage, consider that you probably have different types of data, such as transactional data, log data, streaming data, and Internet of Things (IoT) data. AWS provides solutions for data storage for each of these types of data.

Transactional data

Transactional data, such as e-commerce purchase transactions and financial transactions, is typically stored in relational database management systems (RDBMS) or NoSQL database systems. The choice of database solution depends on the use case and application characteristics.

- A **NoSQL** database is suitable when the data is not well-structured to fit into a defined schema, or when the schema changes often.
- An **RDBMS** solution is suitable when transactions happen across multiple table rows and the queries require complex joins.

Amazon DynamoDB is a fully managed NoSQL database service that you can use as an OLTP store for your applications. Amazon Aurora and Amazon Relational Database Service (Amazon RDS) enable you to implement an SQL-based relational database solution for your application:

- **Amazon Aurora** is a MySQL and PostgreSQL-compatible relational database built for the cloud.
- **Amazon RDS** is a service that enables you to easily set up, operate, and scale relational databases on the cloud.

For more information about the different AWS database services, see Databases on AWS.

Log data

Reliably capturing system-generated logs helps you troubleshoot issues, conduct audits, and perform analytics using the information stored in the logs. Amazon S3 is a popular storage solution for non-transactional data, such as log data, that is used for analytics. Because it provides 99.999999999 percent durability, S3 is also a popular archival solution.

Streaming data

Web applications, mobile devices, and many software applications and services can generate staggering amounts of streaming data—sometimes terabytes per hour—that need to be collected, stored, and processed continuously. Using Amazon Kinesis services, you can do that simply and at a low cost. Alternatively, you can use Amazon Managed Streaming for Apache Kafka (Amazon MSK) to run applications that use Apache Kafka to process streaming data. With Amazon MSK, you can use native Apache Kafka application programming interfaces (APIs) to populate data lakes, stream changes to and from databases, and power ML and analytics applications.

IoT data

Devices and sensors around the world send messages continuously. Enterprises today need to capture this data and derive intelligence from it. Using AWS IoT, connected devices interact easily and securely with the AWS Cloud. Use AWS IoT to leverage AWS services like AWS Lambda, Amazon Kinesis Services, Amazon S3, Amazon Machine Learning, and Amazon DynamoDB to build applications that gather, process, analyze, and act on IoT data, without having to manage any infrastructure.

Data processing

The collection process provides data that potentially has useful information. You can analyze the extracted information for intelligence that will help you grow your business. This intelligence might, for example, tell you about your user behavior and the relative popularity of your products. The best practice to gather this intelligence is to load your raw data into a data warehouse to perform further analysis.

There are two types of processing workflows to accomplish this: *batch* processing and *real-time* processing. The most common forms of processing, online analytic processing (OLAP) and OLTP, each use one of these types. OLAP processing is generally batch-based. OLTP systems are oriented toward real-time processing, and are generally not well suited for batch-based processing. If you

decouple data processing from your OLTP system, you keep the data processing from affecting your OLTP workload.

First, let's look at what is involved in batch processing.

Batch Processing

- **Extract Transform Load (ETL)** — ETL is the process of pulling data from multiple sources to load into data warehousing systems. ETL is normally a continuous, ongoing process with a well-defined workflow. During this process, data is initially extracted from one or more sources. The extracted data is then cleansed, enriched, transformed, and loaded into a data warehouse. For batch ETL, use AWS Glue or Amazon EMR. AWS Glue is a fully managed ETL service. You can create and run an ETL job with a few clicks in the AWS Management Console. Amazon EMR is for big data processing and analysis. EMR offers an expandable, low-configuration service as an easier alternative to running in-house cluster computing.
- **Extract Load Transform (ELT)** — ELT is a variant of ETL, where the extracted data is loaded into the target system first. Transformations are performed after the data is loaded into the data warehouse. ELT typically works well when your target system is powerful enough to handle transformations. Amazon Redshift is often used in ELT pipelines, because it is highly efficient in performing transformations.
- **Online Analytical Processing (OLAP)** — OLAP systems store aggregated historical data in multidimensional schemas. Used widely for query, reporting, and analytics, OLAP systems enable you to extract data and spot trends on multiple dimensions. Because it is optimized for fast joins, Amazon Redshift is often used to build OLAP systems.

Now let’s look at what’s involved in real-time processing of data.

Real-time processing

We talked about streaming data earlier, and mentioned Amazon Kinesis Services and Amazon MSK as solutions to capture and store streaming data. You can process this data sequentially and incrementally on a record-by-record basis, or over sliding time windows. Use the processed data for a wide variety of analytics, including correlations, aggregations, filtering, and sampling. This type of processing is called *real-time* processing.

Information derived from real-time processing gives companies visibility into many aspects of their business and customer activity, such as service usage (for metering or billing), server activity,

website clicks, and geolocation of devices, people, and physical goods. This enables them to respond promptly to emerging situations. Real-time processing requires a highly concurrent and scalable processing layer.

To process streaming data in real-time, use AWS Lambda. Lambda can process the data directly from AWS IoT or Amazon Kinesis Data Streams. Lambda enables you to run code without provisioning or managing servers.

Amazon Kinesis Client Library (KCL) is another way to process data from Amazon Kinesis Streams. KCL gives you more flexibility than Lambda to batch your incoming data for further processing. You can also use KCL to apply extensive transformations and customizations in your processing logic.

Amazon Data Firehose is the easiest way to load streaming data into AWS. It can capture streaming data and automatically load it into Amazon Redshift, enabling near-real-time analytics with existing BI tools, and dashboards you're already using today. Define batching rules with Firehose, and it takes care of reliably batching the data and delivering it to Amazon Redshift.

Amazon MSK is an easy way to build and run applications that use Apache Kafka to process streaming data. Apache Kafka is an open-source platform for building real-time streaming data pipelines and applications. With Amazon MSK, you can use native Apache Kafka APIs to populate data lakes, stream changes to and from databases, and power machine learning and analytics applications.

AWS Glue streaming jobs enable you to perform complex ETL on streaming data. Streaming ETL jobs in AWS Glue can consume data from streaming sources like Amazon Kinesis Data Streams and Amazon MSK, clean and transform those data streams in-flight, and continuously load the results into S3 data lakes, data warehouses, or other data stores. As you process streaming data in an AWS Glue job, you have access to the full capabilities of Spark Structured Streaming to implement data transformations, such as aggregating, partitioning, and formatting, as well as joining with other data sets to enrich or cleanse the data for easier analysis.

Data storage

You can store your data in a lake house, data warehouse, or data mart.

- **Lake house** — A lake house is an architectural pattern that combines the best elements of data warehouses and data lakes. Lake houses enable you to query data across your data warehouse, data lake, and operational databases to gain faster and deeper insights that are not possible otherwise. With a lake house architecture, you can store data in open file formats in your data

lake and query it in place while joining with data warehouse data. This enables you to make this data easily available to other analytics and machine learning tools, rather than locking it in a new silo.

- **Data warehouse** — Using data warehouses, you can run fast analytics on large volumes of data and unearth patterns hidden in your data by leveraging BI tools. Data scientists query a data warehouse to perform offline analytics and spot trends. Users across the enterprise consume the data using SQL queries, periodic reports, and dashboards as needed to make critical business decisions.
- **Data mart** — A data mart is a simple form of data warehouse focused on a specific functional area or subject matter. For example, you can have specific data marts for each division in your enterprise, or segment data marts based on regions. You can build data marts from a large data warehouse, operational stores, or a hybrid of the two. Data marts are simple to design, build, and administer. However, because data marts are focused on specific functional areas, querying across functional areas can become complex because of distribution.

You can use Amazon Redshift to build lake houses, data marts, and data warehouses. Redshift enables you to easily query data in your data lake, and write data back to your data lake in open formats. You can use familiar SQL statements to combine and process data across all your data stores, and execute queries on live data in your operational databases without requiring any data loading and ETL pipelines.

Analysis and visualization

After processing the data and making it available for further analysis, you need the right tools to analyze and visualize the processed data.

In many cases, you can perform data analysis using the same tools you use for processing data. You can use tools such as MySQL Workbench to analyze your data in Amazon Redshift with ANSI SQL. Amazon Redshift also works well with popular third-party BI solutions available on the market, such as Tableau and MicroStrategy.

Amazon Quick is a fast, cloud-powered BI service that enables you to create visualizations, perform analysis as needed, and quickly get business insights from your data. Amazon Quick offers native integration with AWS data sources such as Amazon Redshift, Amazon S3, and Amazon RDS. Amazon Redshift sources can be auto-detected by Amazon Quick, and can be queried either using a direct query or SPICE mode. SPICE is the in-memory optimized calculation engine for Amazon Quick, designed specifically for fast, as-needed data visualization. You can improve the

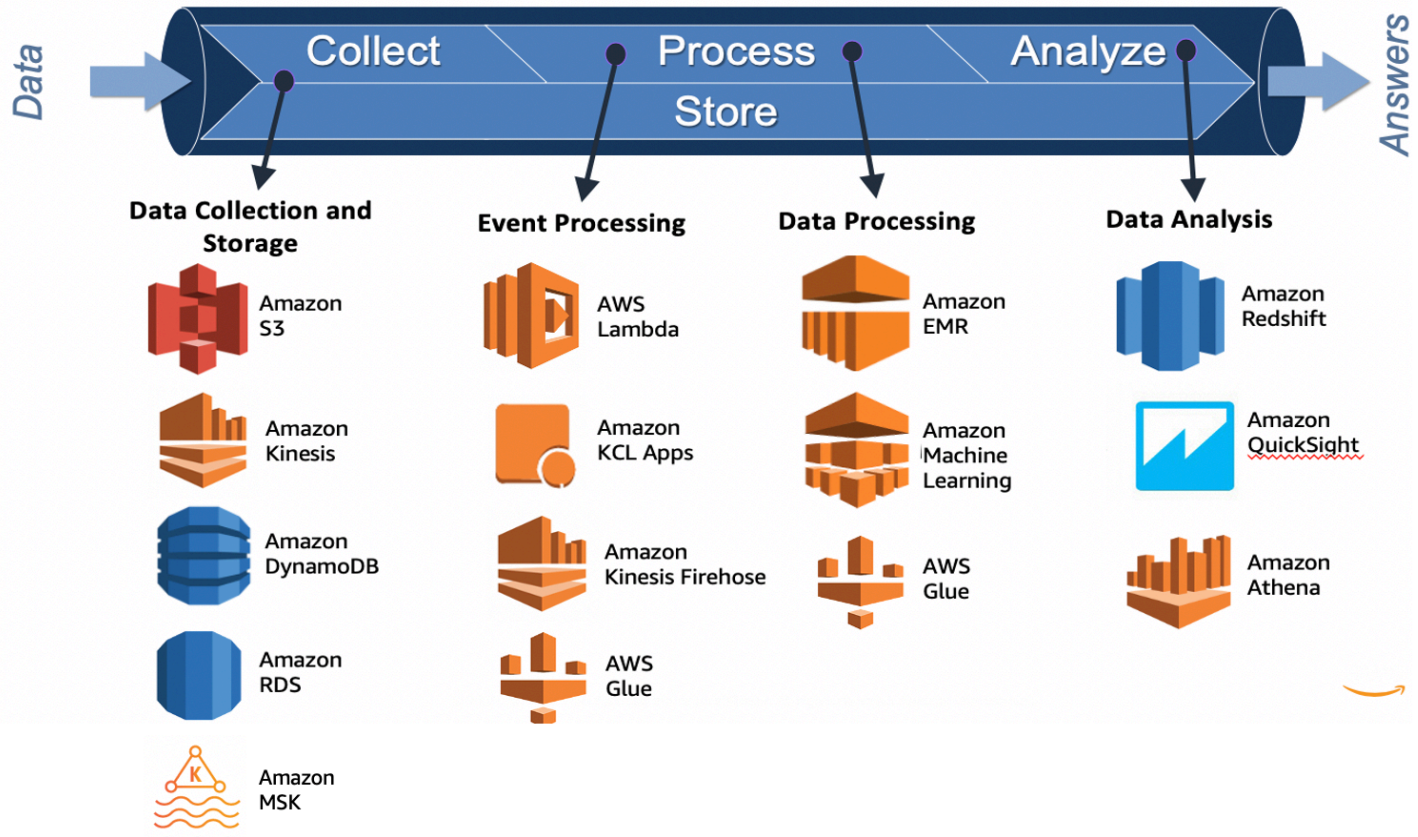
performance of database datasets by importing the data into SPICE instead of using a direct query to the database.

If you are using Amazon S3 as your primary storage, you can use Amazon Athena/QuickSight integration to perform analysis and visualization. Amazon Athena is an interactive query service that makes it easy to analyze data in S3 using standard SQL. You can run SQL queries using Athena on data stored in S3, and build business dashboards within QuickSight.

For another visualization approach, Apache Zeppelin is an open-source BI solution that you can run on Amazon EMR to visualize data in Amazon S3 using Spark SQL. You can also use Apache Zeppelin to visualize data in Amazon Redshift.

Analytics pipeline with AWS services

AWS offers a broad set of services to implement an end-to-end analytics platform. Figure 2 shows the services we discussed, and where they fit within the analytics pipeline.



Analytics pipeline with AWS services

Data warehouse technology options

In this section, we discuss options available for building a data warehouse: row-oriented databases, column-oriented databases, and massively parallel processing architectures.

Row-oriented databases

Row-oriented databases typically store whole rows in a physical block. High performance for read operations is achieved through secondary indexes. Databases such as Oracle Database Server, Microsoft SQL Server, MySQL, and PostgreSQL are row-oriented database systems. These systems have been traditionally used for data warehousing, but they are better suited for transactional processing (OLTP) than for analytics.

To optimize performance of a row-based system used as a data warehouse, developers use a number of techniques, including:

- Building materialized views
- Creating pre-aggregated rollup tables
- Building indexes on every possible predicate combination
- Implementing data partitioning to leverage partition pruning by query optimizer
- Performing index-based joins

Traditional row-based data stores are limited by the resources available on a single machine. Data marts alleviate the problem, to an extent, by using functional sharding. You can split your data warehouse into multiple data marts, each satisfying a specific functional area. However, when data marts grow large over time, data processing slows down.

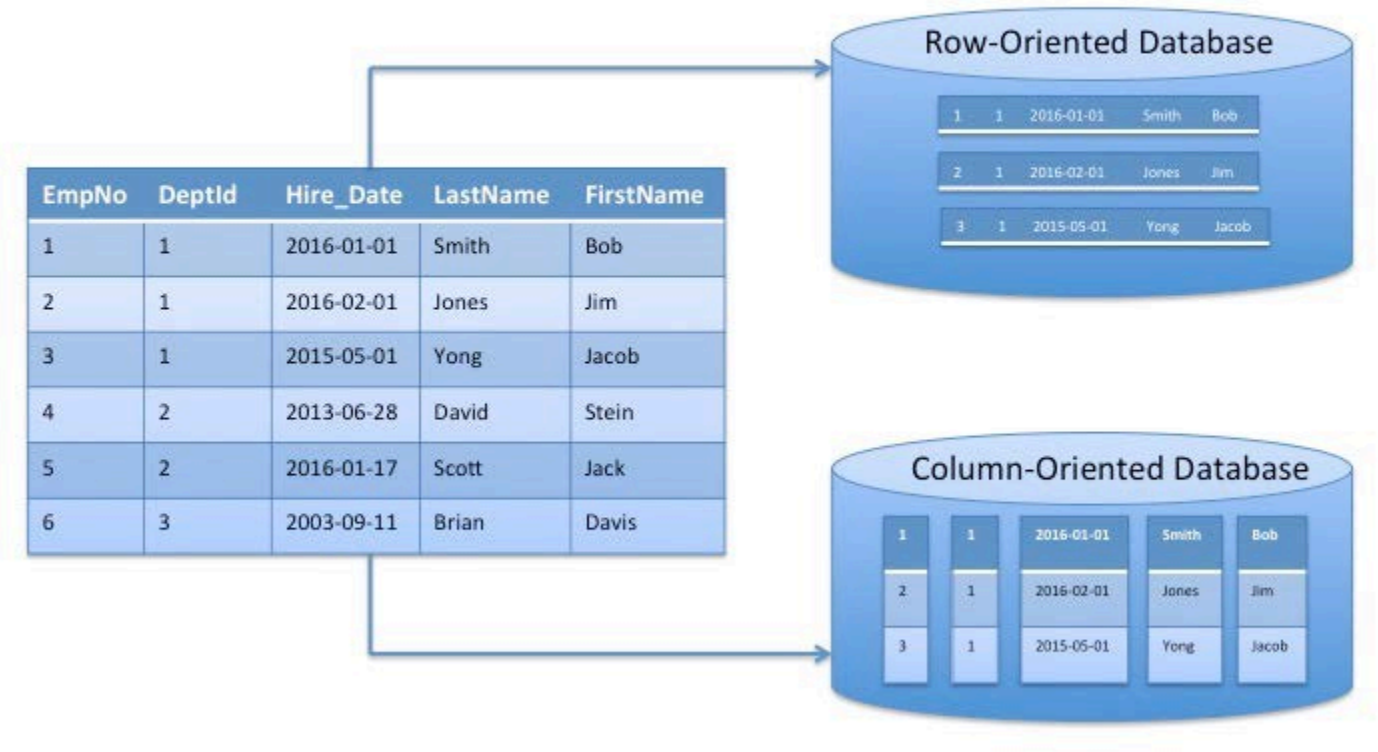
In a row-based data warehouse, every query has to read through all of the columns for all of the rows in the blocks that satisfy the query predicate, including columns you didn't choose. This approach creates a significant performance bottleneck in data warehouses, where your tables have more columns, but your queries use only a few.

Column-oriented databases

Column-oriented databases organize each column in its own set of physical blocks instead of packing the whole rows into a block. This functionality allows them to be more input/output (I/O)

efficient for read-only queries, because they have to read only those columns accessed by a query from disk (or from memory). This approach makes column-oriented databases a better choice than row-oriented databases for data warehousing.

Figure 3 illustrates the primary difference between row-oriented and column-oriented databases. Rows are packed into their own blocks in a row-oriented database, and columns are packed into their own blocks in a column-oriented database.



Row-oriented vs. column-oriented databases

After faster I/O, the next biggest benefit to using a column-oriented database is improved compression. Because every column is packed into its own set of blocks, every physical block contains the same data type. When all the data is the same data type, the database can use extremely efficient compression algorithms. As a result, you need less storage compared to a row-oriented database. This approach also results in significantly lesser I/O, because the same data is stored in fewer blocks.

Some column-oriented databases that are used for data warehousing include Amazon Redshift, Vertica, Greenplum, Teradata Aster, Netezza, and Druid.

Massively Parallel Processing (MPP) architectures

An MPP architecture enables you to use all the resources available in the cluster for processing data, which dramatically increases performance of petabyte scale data warehouses. MPP data warehouses allow you improve performance by simply adding more nodes to the cluster. Amazon Redshift, Druid, Vertica, Greenplum, and Teradata Aster are some of the data warehouses built on an MPP architecture. Open-source frameworks such as Hadoop and Spark also support MPP.

Amazon Redshift deep dive

As a columnar MPP technology, Amazon Redshift offers key benefits for performant, cost-effective data warehousing, including efficient compression, reduced I/O, and lower storage requirements. It is based on ANSI SQL, so you can run existing queries with little or no modification. As a result, it is a popular choice for enterprise data warehouses.

Amazon Redshift delivers fast query and I/O performance for virtually any data size by using columnar storage, and by parallelizing and distributing queries across multiple nodes. It automates most of the common administrative tasks associated with provisioning, configuring, monitoring, backing up, and securing a data warehouse, making it easy and inexpensive to manage. Using this automation, you can build petabyte-scale data warehouses in minutes instead of the weeks or months taken by traditional on-premises implementations. You can also run exabytes-scale queries by storing data on S3 and querying it using Amazon Redshift Spectrum. Amazon Redshift also enables you to scale compute and storage separately using Amazon Redshift RA3 nodes. RA3 nodes come with Redshift Managed Storage (RMS), which leverages your workload patterns and advanced data management techniques, such as automatic fine-grained data eviction and intelligent data pre-fetching. You can size your cluster based on your compute needs only, and pay only for the storage used.

Integration with data lake

Amazon Redshift provides a feature called Redshift Spectrum that makes it easier to both query data and write data back to your data lake in open file formats. With Spectrum, you can query open file formats such as Parquet, ORC, JSON, Avro, CSV, and more directly in S3 using familiar ANSI SQL. To export data to your data lake, you simply use the Redshift UNLOAD command in your SQL code and specify Parquet as the file format, and Redshift automatically takes care of data formatting and data movement into S3. To query data in S3, you create an external schema if the S3 object is already cataloged, or create an external table. You can write data to external tables by running CREATE EXTERNAL TABLE AS SELECT or INSERT INTO an external table. This gives you the flexibility to store highly structured, frequently accessed data in a Redshift data warehouse, while also keeping up to exabytes of structured, semi-structured, and unstructured data in S3. Exporting data from Amazon Redshift back to your data lake enables you to analyze the data further with AWS services like Amazon Athena, Amazon EMR and Amazon SageMaker AI.

Performance

Amazon Redshift offers fast, industry-leading performance with flexibility. Amazon Redshift offers multiple features to achieve this superior performance, including:

- **High performing hardware** — The Amazon Redshift Service offers multiple node types to choose from based on your requirements. The latest generation RA3 instances are built on the AWS Nitro System and feature high bandwidth networking, and performance indistinguishable from bare metal. These Amazon Redshift instances maximize speed for performance-intensive workloads that require large amounts of compute capacity, with the flexibility to pay by usage for storage, and pay separately for compute by specifying the number of instances you need.
- **AQUA (preview)** — AQUA (Advanced Query Accelerator) is a distributed and hardware-accelerated cache that enables Amazon Redshift to run up to ten times faster than any other cloud data warehouse. AQUA accelerates Amazon Redshift queries by running data intensive tasks such as filtering and aggregation closer to the storage layer. This avoids networking bandwidth limitations by eliminating unnecessary data movement between where data is stored and compute clusters. AQUA uses AWS-designed processors to accelerate queries. This includes AWS Nitro chips adapted to speed up data encryption and compression, and custom analytics processors, implemented in field-programmable gate arrays (FPGAs), to accelerate operations such as filtering and aggregation. AQUA can process large amounts of data in parallel across multiple nodes, and automatically scales out to add more capacity as your storage needs grow over time.
- **Efficient storage and high-performance query processing** — Amazon Redshift delivers fast query performance on datasets ranging in size from gigabytes to petabytes. Columnar storage, data compression, and zone maps reduce the amount of I/O needed to perform queries. Along with the industry standard encodings such as LZO and Zstandard, Amazon Redshift also offers purpose-built compression encoding, AZ64, for numeric and date/time types to provide both storage savings and optimized query performance.
- **Materialized views** — Amazon Redshift materialized views enable you to achieve significantly faster query performance for analytical workloads such as dashboarding, queries from BI tools, and ELT data processing jobs. You can use materialized views to store frequently used precomputations to speed up slow-running queries. Amazon Redshift can efficiently maintain the materialized views incrementally to speed up ELT, and provide low latency performance benefits. For more information, see Creating materialized views in Amazon Redshift.
- **Auto workload management to maximize throughput and performance** — Amazon Redshift uses machine learning to tune configuration to achieve high throughput and performance,

even with varying workloads or concurrent user activity. Amazon Redshift utilizes sophisticated algorithms to predict and classify incoming queries based on their run times and resource requirements to dynamically manage resources and concurrency while also enabling you to prioritize your business-critical workloads. Short query acceleration (SQA) sends short queries to an express queue for immediate processing rather than waiting behind long running queries. You can set the priority of your most important queries, even when hundreds of queries are being submitted.

Amazon Redshift is also a self-learning system that observes the user workload continuously, detecting opportunities to improve performance as the usage grows, applying optimizations seamlessly, and making recommendations via Redshift Advisor when an explicit user action is needed to further turbocharge Amazon Redshift performance.

- **Result caching** — Amazon Redshift uses result caching to deliver sub-second response times for repeated queries. Dashboard, visualization, and business intelligence tools that execute repeated queries experience a significant performance boost. When a query executes, Amazon Redshift searches the cache to see if there is a cached result from a prior run. If a cached result is found and the data has not changed, the cached result is returned immediately instead of re-running the query.

Durability and availability

To provide the best possible data durability and availability, Amazon Redshift automatically detects and replaces any failed node in your data warehouse cluster. It makes your replacement node available immediately, and loads your most frequently accessed data first so you can resume querying your data as quickly as possible. Amazon Redshift attempts to maintain at least three copies of data: the original and replica on the compute nodes, and a backup in S3. The cluster is in read-only mode until a replacement node is provisioned and added to the cluster, which typically takes only a few minutes.

Amazon Redshift clusters reside within one Availability Zone. However, if you want to a Multi-AZ setup for Amazon Redshift, you can create a mirror and then self-manage replication and failover.

With just a few clicks in the Amazon Redshift Management Console, you can set up a robust disaster recovery (DR) environment with Amazon Redshift. Amazon Redshift automatically takes incremental snapshots (backups) of your data every eight hours, or five gigabytes (GBs) per node of data change. You can get more information and control over a snapshot, including the ability to control the automatic snapshot's schedule.

You can keep copies of your backups in multiple AWS Regions. In case of a service interruption in one AWS Region, you can restore your cluster from the backup in a different AWS Region. You can gain read/write access to your cluster within a few minutes of initiating the restore operation.

Elasticity and scalability

With Amazon Redshift, you get the elasticity and scalability you need for your data warehousing workloads. You can scale compute and storage independently, and pay only for what you use. With the elasticity and scalability that Amazon Redshift offers, you can easily run non-uniform and unpredictable data warehousing workloads. Amazon Redshift provides two forms of compute elasticity:

- **Elastic resize** — With the elastic resize feature, you can quickly resize your Amazon cluster by adding nodes to get the resources needed for demanding workloads, and to remove nodes when the job is complete to save cost. Additional nodes are added or removed in minutes with minimal disruption to on-going read and write queries. Elastic resize can be automated using a schedule you define to accommodate changes in workload that occur on a regular basis. Resize can be scheduled with a few clicks in the console or programmatically using the AWS command line interface (AWS CLI), or an API call.
- **Concurrency Scaling** — With the Concurrency Scaling feature, you can support virtually unlimited concurrent users and up to 10 concurrent queries, with consistently fast query performance. When concurrency scaling is enabled, Amazon Redshift automatically adds additional compute capacity when you need it to process an increase in concurrent read queries. Write operations continue as normal on your main cluster. Users always see the most current data, whether the queries run on the main cluster or on a concurrency scaling cluster.

  Amazon Redshift enables you to start with as little as a single 160 GB node and scale up all the way to multiple petabytes of compressed user data using many nodes. For more information, see About Clusters and Nodes in the Amazon Redshift Cluster Management Guide.

Amazon Redshift managed storage

Amazon Redshift managed storage enables you to scale and pay for compute and storage independently so you can size your cluster based only on your compute needs. It automatically uses high-performance solid-state drive (SSD)-based local storage as tier-1 cache, and takes advantage of optimizations such as data block temperature, data block age, and workload patterns

to deliver high performance while scaling storage automatically when needed without requiring any action.

Operations

As a managed service, Amazon Redshift completely automates many operational tasks, including:

- **Cluster Performance** — Amazon Redshift performs Auto ANALYZE to maintain accurate table statistics. It also performs Auto VACUUM to ensure that the database storage is efficient and deleted data blocks are reclaimed.
- **Cost Optimization** — Amazon Redshift enables you to pause and resume the clusters that need to be available only at a specific time, enabling you to suspend on-demand billing while the cluster is not being used. Pause and resume can also be automated using a schedule you define to match your operational needs. Cost controls can be defined on Amazon Redshift clusters to monitor and control your usage and associated cost for Amazon Redshift Spectrum and Concurrency Scaling features.

Amazon Redshift Advisor

To help you improve performance and decrease the operating costs for your cluster, Amazon Redshift has a feature called Amazon Redshift Advisor. Amazon Redshift Advisor offers you specific recommendations about changes to make. Advisor develops its customized recommendations by analyzing workload and usage metrics for your cluster. These tailored recommendations relate to operations and cluster settings. To help you prioritize your optimizations, Advisor ranks recommendations by order of impact. You can view Amazon Redshift Advisor analysis results and recommendations on the AWS Management Console.

Interfaces

Amazon Redshift has custom Java Database Connectivity (JDBC) and Open Database Connectivity (ODBC) drivers you can download from the **Connect Client** tab of the console, which means you can use a wide range of familiar SQL clients. You can also use standard PostgreSQL JDBC and ODBC drivers. For more information about Amazon Redshift drivers, see Amazon Redshift and PostgreSQL in the Amazon Redshift Database Developer Guide.

Amazon Redshift provides a built-in Query Editor in the web console. The Query Editor is an in-browser interface for running SQL queries on Amazon Redshift clusters directly from AWS Management Console. It's a convenient way for a database administrator (DBA) or a user to run queries as needed, or diagnose queries.

You can also find numerous examples of validated integrations with many popular BI and ETL vendors. In these integrations, loads and unloads execute in parallel on each compute node to maximize the rate at which you can ingest or export data to and from multiple resources, including S3, Amazon EMR, and DynamoDB. You can easily load streaming data into Amazon Redshift using Amazon Data Firehose, enabling near real-time analytics with existing BI tools and dashboards. You can locate metrics for compute utilization, memory utilization, storage utilization, and read/write traffic to your Amazon Redshift data warehouse cluster by using the console, or Amazon CloudWatch API operations.

Security

To help provide data security, you can run Amazon Redshift inside a virtual private cloud based on the Amazon Virtual Private Cloud (Amazon VPC) service. You can use the software-defined networking model of the VPC to define firewall rules that restrict traffic based on the rules you configure. Amazon Redshift supports SSL-enabled connections between your client application and your Amazon Redshift data warehouse cluster, which enables data to be encrypted in transit. You can also leverage Enhanced VPC Routing to manage data flow between your Amazon Redshift cluster and other data sources. Data traffic is routed within the AWS network instead of public internet.

The Amazon Redshift compute nodes store your data, but the data can be accessed only from the cluster's leader node. This isolation provides another layer of security. Amazon Redshift integrates with AWS CloudTrail to enable you to audit all Amazon Redshift API calls. To help keep your data secure at rest, Amazon Redshift supports encryption, and can encrypt each block using hardware-accelerated Advanced Encryption Standard (AES)-256 encryption as each block is written to disk. This encryption takes place at a low level in the I/O subsystem; the I/O subsystem encrypts everything written to disk, including intermediate query results. The blocks are backed up as is, which means that backups are also encrypted. By default, Amazon Redshift takes care of key management, but you can choose to manage your keys using your own hardware security modules, or manage your keys through AWS Key Management Service (AWS KMS).

Database security management is controlled by managing user access, granting the proper privileges to tables and views to user accounts or groups, and leveraging column-level grant and revoke to meet your security and compliance needs in finer granularity.

In addition, Amazon Redshift provides multiple means of authentication to secure and simplify data warehouse access. You can use AWS Identity and Access Management (AWS IAM) within your AWS account. Use federated authentication if you already manage user identifies outside of AWS

via SAML-2.0-compatible identity providers to enable your users to access the data warehouse without managing database users and passwords. Amazon Redshift also supports multi-factor authentication (MFA) to provide additional security.

Cost model

Amazon Redshift requires no long-term commitments or upfront costs. This pricing approach frees you from the capital expense and complexity of planning and purchasing data warehouse capacity ahead of your needs. Charges are based on the size and number of nodes in your cluster. If you use Amazon Redshift-managed storage (RMS) with an RA3 instance, you pay separately for the amount of compute and RMS that you use.

If you need additional compute power to handle workload spikes, you can enable concurrency scaling. For every 24 hours that your main cluster runs, you accumulate one hour of credit to use this feature for free. Beyond that, you will be charged the per-second on-demand rate.

There is no additional charge for backup storage, up to 100 percent of your provisioned storage. For example, if you have an active cluster with two XL nodes for a total of four terabytes (TB) of storage, AWS provides up to four TB of backup storage on S3 at no additional charge. Backup storage beyond the provisioned storage size, and backups stored after your cluster is terminated, are billed at standard Amazon S3 rates. There is no data transfer charge for communication between S3 and Amazon Redshift.

If you use Redshift Spectrum to access data store in your data lake, you pay for the query cost based on how much data the query scans.

For more information, see Amazon Redshift Pricing.

Ideal usage patterns

Amazon Redshift is ideal for OLAP using your existing BI tools. Enterprises use Amazon Redshift to do the following:

- Running enterprise BI and reporting
- Analyze global sales data for multiple products
- Store historical stock trade data
- Analyze ad impressions and clicks
- Aggregate gaming data

- Analyze social trends
- Measure clinical quality, operation efficiency, and financial performance in health care

With the Amazon Redshift Spectrum feature, Amazon Redshift supports semi-structured data and extends your data warehouse to your data lake. This enables you to:

- Run as-needed analysis on large volume event data such as log analysis and social media
- Offload infrequently accessed history data out of the data warehouse
- Join the external dataset with the data warehouse directly without loading them into the data warehouse

Anti-Patterns

Amazon Redshift is not ideally suited for the following usage patterns:

- **OLTP** – Amazon Redshift is designed for data warehousing workloads delivering extremely fast and inexpensive analytic capabilities. If you require a fast transactional system, you might want to choose a relational database system such as Amazon Aurora or Amazon RDS, or a NoSQL database such as Amazon DynamoDB.
- **Unstructured data** – Data in Amazon Redshift must be structured by a defined schema. Amazon Redshift doesn't support an arbitrary schema structure for each row. If your data is unstructured, you can perform ETL on Amazon EMR to get the data ready for loading into Amazon Redshift. For JSON data, you can store key value pairs and use the native JSON functions in your queries.
- **BLOB data** – If you plan to store binary large object (BLOB) files such as digital video, images, or music, you might want to store the data in S3 and reference its location in Amazon Redshift. In this scenario, Amazon Redshift keeps track of metadata (such as item name, size, date created, owner, location, and so on) about your binary objects, but the large objects themselves are stored in S3.

Migrating to Amazon Redshift

If you decide to migrate from an existing data warehouse to Amazon Redshift, which migration strategy you should choose depends on several factors:

- The size of the database and its tables and objects
- Network bandwidth between the source server and AWS
- Whether the migration and switchover to AWS will be done in one step, or a sequence of steps over time
- The data change rate in the source system
- Transformations during migration
- The partner tool that you plan to use for migration and ETL

One-step migration

One-step migration is a good option for small databases that don't require continuous operation. Customers can extract existing databases as comma separated value (CSV) files, or columnar format like Parquet, then use services such as AWS Snowball Edge to deliver datasets to S3 for loading into Amazon Redshift. Customers then test the destination Amazon Redshift database for data consistency with the source. After all validations have passed, the database is switched over to AWS.

Two-step migration

Two-step migration is commonly used for databases of any size:

- **Initial data migration** — The data is extracted from the source database, preferably during non-peak usage to minimize the impact. The data is then migrated to Amazon Redshift by following the one-step migration approach described previously.
- **Changed data migration** — Data that changed in the source database after the initial data migration is propagated to the destination before switchover. This step synchronizes the source and destination databases.

After all the changed data is migrated, you can validate the data in the destination database, perform necessary tests, and if all tests are passed, switch over to the Amazon Redshift data warehouse.

Wave-based migration

Large-scale MPP data warehouse migration presents a challenge in terms of project complexity, and is riskier. Taking precautions to break a complex migration project into multiple logical and systematic waves can significantly reduce the complexity and risk. Starting from a workload that covers a good number of data sources and subject areas with medium complexity, then add more data sources and subject areas in each subsequent wave. See Develop an application migration methodology to modernize your data warehouse with Amazon Redshift for a description of how to migrate from the source MPP data warehouse to Amazon Redshift using the wave-based migration approach.

Tools and additional help for database migration

Several tools and technologies for data migration are available. You can use some of these tools interchangeably, or you can use other third-party or open-source tools available in the market.

- AWS Database Migration Service supports both the one-step and the two-step migration processes. To follow the two-step migration process, you enable supplemental logging to capture changes to the source system. You can enable supplemental logging at the table or database level.
- AWS Schema Conversion Tool (SCT) is a free tool that can convert the source database schema and a majority of the database code objects, including views, stored procedures, and functions, to a format compatible with the target databases. SCT can scan your application source code for embedded SQL statements and convert them as part of a database schema conversion project. After schema conversion is complete, SCT can help migrate a range of data warehouses to Amazon Redshift using built-in data migration agents.
- Additional data integration partner tools include:
  - Informatica
  - Matillion
  - SnapLogic
  - Talend
  - BryteFlow Ingest
  - SQL Server Integration Services (SSIS)

For more information on data integration and consulting partners, see Amazon Redshift Partners.

We provide technical advice, migration support, and financial assistance to help eligible customers quickly and cost-effectively migrate from legacy data warehouses to Amazon Redshift, the most popular and fastest cloud data warehouse. Qualifying customers receive advice on application architecture, migration strategies, program management, proof-of-concept, and employee training that are customized for their technology landscape and migration goals. We offer migration assistance through Amazon Database Migration Accelerator, AWS Professional Services, or our network of Partners. These teams and organizations specialize in a range of data warehouse and analytics technologies, and bring a wealth of experience acquired by migrating thousands of data warehouses and applications to AWS. We also offer service credits to minimize the financial impact of the migration. For more information, see Migrate to Amazon Redshift.

Designing data warehousing workflows

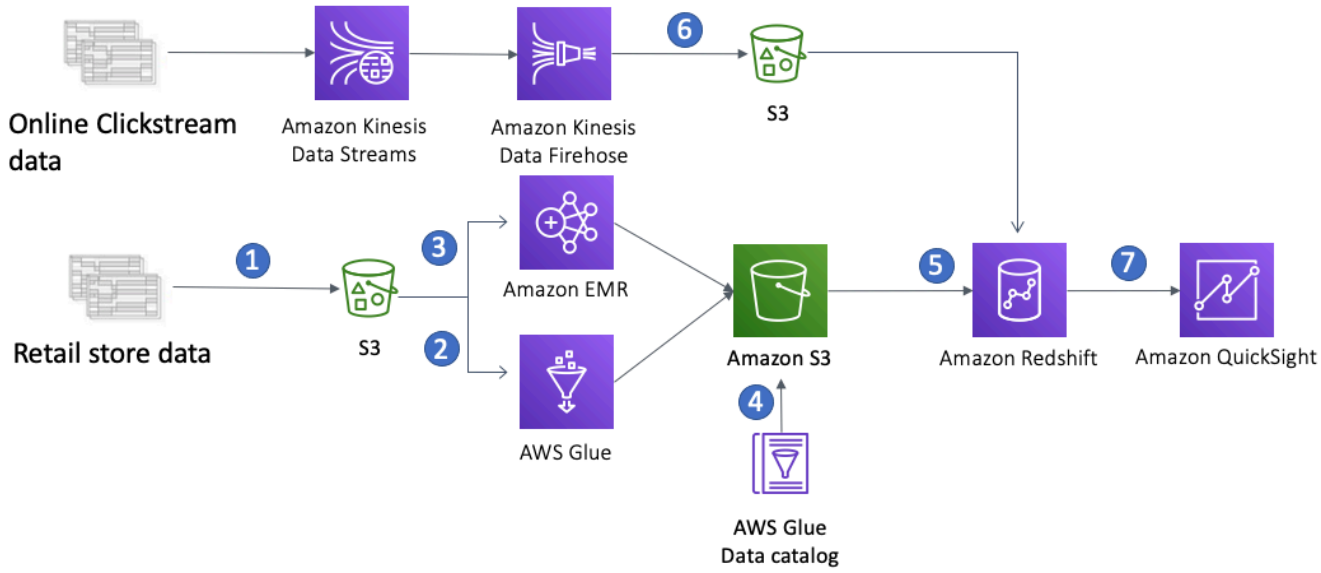
In the previous sections, we discussed the features of Amazon Redshift that make it ideally suited for data warehousing. To understand how to design data warehousing workflows with Amazon Redshift, let's look at the most common design pattern, along with an example use case.

Suppose that a multinational clothing maker has more than a thousand retail stores, sells certain clothing lines through department and discount stores, and has an online presence. From a technical standpoint, these three channels currently operate independently. They have different management, point-of-sale systems, and accounting departments. No single system merges all the related datasets together to provide the CEO with a 360-degree view across the entire business.

Suppose the CEO wants to get a company-wide picture of these channels, and perform analytics such as the following:

- What trends exist across channels?
- Which geographic regions do better across channels?
- How effective are the company's advertisements and promotions?
- What trends exist across each clothing line?
- Which external forces have impacts on the company's sales; for example, the unemployment rate and weather conditions?
- What online ads are most effective?
- How do store attributes affect sales; for example, tenure of employees and management, strip mall versus enclosed mall, location of merchandise in the store, promotion, endcaps, sales circulars, and in-store displays?

An enterprise data warehouse solves this problem. It collects data from each of the three channels' various systems, and from publicly available data such as weather and economic reports. Each data source sends data daily for consumption by the data warehouse. Clickstream data are streamed continuously and stored on S3. Because each data source might be structured differently, an ETL process is performed to reformat the data into a common structure. Then analytics can be performed across data from all sources simultaneously. To do this, we use the following data flow architecture:



Enterprise data warehouse workflow

1. The first step is getting the data from different sources into S3. S3 provides a highly durable, inexpensive, and scalable storage platform that can be written to in parallel from many different sources at a low cost.
2. For batch ETL, you can use either Amazon EMR or AWS Glue. AWS Glue is a fully managed ETL service that simplifies ETL job creation and eliminates the need to provision and manage infrastructure. You pay only for the resources used while your jobs are running. AWS Glue also provides a centralized metadata repository. Simply point AWS Glue to your data stored in AWS, and AWS Glue discovers your data and stores the associated table definition and schema in the AWSAWS Glue Data Catalog. Once cataloged, your data is immediately searchable, can be queried, and is available for ETL.
3. Amazon EMR can transform and cleanse the data from the source format to go into the destination format. Amazon EMR has built-in integration with S3, which allows parallel threads of throughput from each node in your Amazon EMR cluster to and from S3. Typically, a data warehouse gets new data on a nightly basis. Because there is usually no need for analytics in the middle of the night, the only requirement around this transformation process is that it finishes by the morning when the CEO and other business users need to access reports and dashboards.

   You can use the Amazon EC2 Spot market to further bring down the cost of ETL. A good spot strategy is to start bidding at a low price at midnight, and continually increase your price over time until capacity is granted. As you get closer to the deadline, if spot bids have not

succeeded, you can fall back to on-demand prices to ensure you still meet your completion time requirements. Each source might have a different transformation process on Amazon EMR, but with the AWS pay-as-you-go model, you can create a separate Amazon EMR cluster for each transformation. You can tune each cluster it to be exactly the right capacity to complete all data transformation jobs without contending with resources for the other jobs.

4. Each transformation job loads formatted, cleaned data into S3. We use S3 here again because Amazon Redshift can load the data in parallel from S3, using multiple threads from each cluster node. S3 also provides an historical record, and serves as the formatted source of truth between systems. Data on S3 is cataloged by AWS Glue. The metadata is stored in the AWS Glue data catalog, which allows it to be consumed by other tools for analytics or machine learning if additional requirements are introduced over time.
5. Amazon Redshift loads, sorts, distributes, and compresses the data into its tables so that analytical queries can execute efficiently and in parallel. If you leverage an RA3 instance with Amazon Redshift managed storage, Amazon Redshift can automatically scale storage as your data increases. As the business expands, you can enable Amazon Redshift concurrency scaling to handle more and more user requests, and keep near-linear performance. With new workloads are added, you can increase data warehouse capacity in minutes by adding more nodes via Amazon Redshift elastic resize.
6. Clickstream data is stored on S3 via Firehose hourly, or even more frequently. Because Amazon Redshift can query S3 external data via Spectrum without having to load them into a data warehouse, you can track the customer online journey in near real-time, and join it with sales data in your data warehouse to understand customer behavior better. This provides a more complete picture of customers, and enables business users to get insight sooner, and take action.
7. To visualize the analytics, you can use Amazon Quick or one of the many partner visualization platforms that connect to Amazon Redshift using ODBC or JDBC. This point is where the CEO and their staff view reports, dashboards, and charts. Now executives can use the data for making better decisions about company resources, which ultimately increases earnings and value for shareholders.

You can easily expand this flexible architecture when your business expands, opens new channels, launches additional customer-specific mobile applications, and brings in more data sources. It takes just a few clicks in the Amazon Redshift Management Console, or a few API calls.

Conclusion and further reading

There is a strategic shift in data warehousing as enterprises migrate their analytics databases and solutions from on-premises solutions to the cloud, to take advantage of the cloud's simplicity, performance, elasticity, and cost-effectiveness. This whitepaper offers a comprehensive account of the current state of data warehousing on AWS. AWS provides a broad set of services, and a strong partner ecosystem that enable customers to easily build and run enterprise data warehousing in the cloud. The result is a highly performant, cost-effective analytics architecture that can scale with your business on the AWS global infrastructure.

Further Reading

For additional information, see:

- Amazon Redshift FAQs
- Amazon Redshift lake house architecture
- Amazon Redshift customer success
- Amazon Redshift best practices
- Implementing workload management
- Querying external data using Amazon Redshift Spectrum
- Amazon Redshift Documentation
- Amazon Redshift system overview
- What is Amazon Redshift?
- AWS Key Management Service (KMS)
- Amazon Redshift JSON functions
- Amazon Redshift pricing
- Amazon Redshift Partners
- AWS Database Migration Service
- Develop an application migration methodology to modernize your data warehouse with Amazon Redshift (blog entry)
- What is Streaming Data?
- Column-oriented DBMS

Document history and contributors

To be notified about updates to this whitepaper, subscribe to the RSS feed.

Change	Description	Date
Whitepaper updated	Updated to include latest features and capabilities.	January 15, 2021
Initial publication	Whitepaper first published.	March 1, 2016

Note

To subscribe to RSS updates, you must have an RSS plug-in enabled for the browser that you are using.

Contributors

Contributors to this document include:

- Anusha Challa, Sr. Analytics SSA, Amazon Web Services
- Corina Radovanovich, Sr. Product Marketing Manager, Amazon Web Services
- Juan Yu, Sr. Analytics SSA, Amazon Web Services
- Lucy Friedmann, Product Marketing Manager, Amazon Web Services
- Manan Goel, Principal Product Manager, Amazon Web Services

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.