aws

Integrating Amazon Aurora PostgreSQL-Compatible with heterogeneous databases and AWS services

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Integrating Amazon Aurora PostgreSQL-Compatible with heterogeneous databases and AWS services

Table of Contents

Integrating Amazon Aurora PostgreSQL-Compatible with heterogeneous databases and AWS services

Rambabu Karnena, Amazon Web Services (AWS)

August 2024 ([document history](#))

Amazon Aurora PostgreSQL-Compatible Edition offers integration with heterogeneous databases and various services on the Amazon Web Services (AWS) Cloud. You can use this integration to build scalable data architectures on AWS, unlocking new possibilities for your applications.

Overview

This guide provides a comprehensive overview of Aurora PostgreSQL-Compatible integration with AWS services and heterogeneous databases. If you currently run on Oracle Database or Microsoft SQL Server, explore Aurora PostgreSQL-Compatible features that are equivalent to linked servers, database links, and external tables. By using this guidance, you can also do the following:

- Integrate with services such as Amazon Simple Storage Service (Amazon S3), AWS Lambda, Amazon CloudWatch Logs, and Amazon Redshift.
- Avoid design mistakes that might lead to errors or performance issues.
- Make informed decisions when integrating Aurora PostgreSQL-Compatible with other databases and AWS services.

Connecting to remote and heterogeneous databases

A key strength of Aurora PostgreSQL-Compatible is its ability to connect to remote [PostgreSQL](#), [SQL Server](#), and Oracle, MySQL databases by using foreign data wrappers (FDWs). These wrappers include `postgres_fdw`, `oracle_fdw`, `tds_fdw` (for SQL Server), and `mysql_fdw`. You can use these foreign data wrappers to query data from these multiple sources directly.

Logging

Aurora PostgreSQL-Compatible also integrates with [Amazon CloudWatch Logs](#) by using the `log_fdw` extension. You can use `log_fdw` to query and analyze PostgreSQL log files directly

from the database. This integration streamlines log monitoring and analysis so that you can gain
valuable insights into your database's performance, errors, and activities.

Connecting to storage

The `aws_s3` extension integrates with [Amazon S3](#) for data imports, exports, backups, and
archiving. By supporting data movement between Aurora PostgreSQL-Compatible and Amazon S3,
the `aws_s3` extension enables efficient data ingestion, backup, and archiving processes.

Serverless compute

You can use the `aws_lambda` extension to improve performance and cost efficiency by offloading
compute-intensive tasks to serverless [AWS Lambda](#) functions. Lambda supports complex
computations, data transformations, and integrations with other AWS services directly from your
Aurora PostgreSQL-Compatible database, without the need for dedicated compute resources.

Integrations for analytics

To facilitate data migration and ETL (extract, transform, and load) processes, Aurora PostgreSQL-
Compatible can integrate with services such as [AWS Database Migration Service (AWS DMS)](#) and
AWS Glue. AWS DMS supports the migration of databases to Aurora PostgreSQL-Compatible. [AWS
Glue](#) helps you to build robust data-processing pipelines for transforming and loading data from
various sources into Aurora PostgreSQL-Compatible or other AWS services.

For a data warehousing solution, Aurora PostgreSQL-Compatible can integrate with [Amazon
Redshift](#). By acting as a data source or staging area for Amazon Redshift, Aurora PostgreSQL-
Compatible supports advanced analytics and reporting capabilities. You can use these capabilities
to derive valuable insights from your data.

Whether you're migrating existing workloads, building hybrid architectures, or developing new
data-driven applications, the integration capabilities of Aurora PostgreSQL-Compatible help you to
create scalable, high-performance, and cost-effective solutions.

Additional extensions for integrating with AWS services

PostgreSQL lists various data wrappers for connecting to files, NoSQL databases, specific database
systems. For more information, see the [PostgreSQL documentation](#).

Objectives

This guide helps new customers, especially those who are migrating from Oracle or Microsoft SQL Server, to achieve the following:

- Find equivalent features on AWS. Aurora PostgreSQL-Compatible provides functional equivalents to linked servers, database links, and external tables.
- Design systems and batch jobs that integrate with and connect to heterogeneous databases and AWS services.
- Avoid common design pitfalls and optimize infrastructure implementations.

Prerequisites

To follow along with this guide, ensure that you have access to the following:

- An active AWS account

- An Amazon Aurora PostgreSQL-Compatible Edition cluster (For instructions, see Create an Aurora PostgreSQL DB cluster.)

- Amazon Simple Storage Service (Amazon S3)

- Amazon CloudWatch Logs

- AWS Lambda

- AWS Glue

- AWS Database Migration Service (AWS DMS)

- An Amazon Elastic Compute Cloud (Amazon EC2) instance with SQL Server, Oracle, and PostgreSQL databases installed

The Aurora PostgreSQL-Compatible instance and the other databases or AWS services must in be the same virtual private cloud (VPC), or network connectivity must be established between them. Additionally, you must have the required roles and security privileges assigned.

Aurora PostgreSQL-Compatible database integration

To create connections between PostgreSQL databases and remote databases, you can use foreign data wrappers (FDWs). Foreign data wrappers offer the following advantages over SQL Server linked servers and Oracle database links:

- **Native PostgreSQL integration** – FDWs are native to PostgreSQL, and they use its SQL capabilities. This improves the integration experience.

- **Integration and optimization** – Linked servers (SQL Server) and database links (Oracle) are specific to their respective database ecosystems. By supporting queries to remote data sources and across database engines such as Oracle, SQL Server, MySQL, FDWs provide a more integrated and optimized approach for PostgreSQL.

- **Cross-database querying** – When you use FDWs, you can query data from multiple remote data sources within a single SQL statement. This supports cross-database analytics and reporting.

- **Push-down optimization** – FDWs can push operations such as filtering, projections, and sorting down to the remote data source. This reduces data transfer, and it improves query performance.

- **Parallel runs** – Foreign data wrappers support running queries that involve remote data sources in parallel, which improves performance.

By using foreign data wrapper integration, you can query and manipulate data from remote databases directly within Amazon Aurora PostgreSQL-Compatible Edition. This supports hybrid architectures and data integration scenarios.

This guide focuses on the `postgres_fdw` extension for connecting to remote PostgreSQL databases and the `tds_fdw` extension for connecting to SQL Server databases. Not covered in this guide are the following PostgreSQL extensions:

- `oracle_fdw` for accessing data from Oracle databases

- `mysql_fdw` for accessing data from MySQL databases

Aurora PostgreSQL-Compatible integration with remote PostgreSQL databases

This section discusses Amazon Aurora PostgreSQL-Compatible Edition integration with remote PostgreSQL databases using the `postgres_fdw` (foreign-data wrapper) extension or the `dblink` feature. The `postgres_fdw` module provides federated query capability for interacting with remote PostgreSQL-based databases. The remote databases can be managed or self-managed on [Amazon EC2](#) or on premises. The `postgres_fdw` extension is available in all currently supported versions of Amazon Relational Database Service (Amazon RDS) for PostgreSQL and Aurora PostgreSQL-Compatible.

Using the `postgres_fdw` extension, you can access and query data from remote PostgreSQL databases as if they were local tables. The `postgres_fdw` extension also supports the following:

- Cross-version compatibility for accessing data from external PostgreSQL servers that are running different versions.

- Transaction management, which helps to ensure data consistency and integrity when you perform operations across local and external PostgreSQL servers.

- Distributed transactions, which provide atomicity (a property of ACID transactions) and isolation guarantees when you perform operations across multiple external PostgreSQL servers. This helps ensure that either all operations in a transaction are committed or none are committed, maintaining data consistency and integrity.

Although the `dblink` module provides a way to interact with remote PostgreSQL databases, it doesn't support distributed transactions or other advanced features. If you need more advanced functionality, consider using the `postgres_fdw` extension instead. The `postgres_fdw` extension provides more integration and optimization capabilities.

postgres_fdw use cases and high-level steps

The `postgres_fdw` extension usage with Aurora PostgreSQL-Compatible supports the following use cases and scenarios:

- **Federated queries and data integration** – Querying and combining data from multiple PostgreSQL databases within a single Aurora PostgreSQL-Compatible instance

- **Offloading read workloads** – Connecting to read replicas of external PostgreSQL servers, offloading read-heavy workloads, and improving query performance

- **Cross-database operations** – Performing INSERT, UPDATE, DELETE, and COPY operations across multiple PostgreSQL databases, enabling cross-database data manipulation and maintenance tasks

To configure `postgres_fdw`, use the following high-level steps:

1. Connect to your Aurora PostgreSQL-Compatible cluster by using a PostgreSQL client, and create the `postgres_fdw` extension:

```
CREATE EXTENSION postgres_fdw;
```

This extension provides the functionality to connect to remote PostgreSQL databases.

2. Create a foreign server named `my_fdw_target` by using the `CREATE SERVER` command. This server represents the remote PostgreSQL database that you want to connect to. Specify the database name, hostname, and SSL mode as options for this server.

3. Ensure that the necessary security groups and network configurations are in place to allow Aurora PostgreSQL-Compatible to connect to the remote PostgreSQL database.

   If the remote database is hosted on premises, you might need to configure a virtual private network (VPN) or AWS Direct Connect connection.

   Run the following command:

```
CREATE SERVER my_fdw_target Foreign Data Wrapper postgres_fdw OPTIONS (DBNAME
  'postgres', HOST 'SOURCE_HOSTNAME', SSLMODE 'require');
```

4. Create a user mapping for the `dbuser` user on the `my_fdw_target` server. This mapping associates the `dbuser` user and password on the local Aurora PostgreSQL-Compatible instance with the corresponding user on the remote database.

```
CREATE USER MAPPING FOR dbuser SERVER my_fdw_target OPTIONS (user 'DBUSER', password
  'PASSWORD');
```

   This step is necessary to authenticate and provide access to remote database.

5. Create a foreign table named `customer_fdw` with the `my_fdw_target` server and user mapping that you set up previously:

```
CREATE FOREIGN TABLE customer_fdw( id int, name varchar, emailid varchar,
  projectname varchar, contactnumber bigint) server my_fdw_target OPTIONS( TABLE_NAME
  'customers');
```

The `customer_fdw` table maps to the `customers` table in the remote database specified by the `my_fdw_target` server. The foreign table has the same structure as the remote table so that you can interact with the remote data as if it were a local table.

6. You can perform various data manipulation operations on the `customer_fdw` foreign table, such as INSERT, UPDATE, and SELECT queries. The script demonstrates inserting a new row and updating an existing row, deleting a record, and truncating a table in the remote `customers` table through the `customer_fdw` foreign table:

```
INSERT INTO customer_fdw values ( 1, 'Test1', 'Test1@email.com', 'LMS1',
  '888888888');
INSERT INTO customer_fdw values ( 2, 'Test2', 'Test2@email.com', 'LMS2',
  '999999999');
INSERT INTO customer_fdw values ( 3, 'Test3', 'Test3@email.com', 'LMS3',
  '111111111');
UPDATE customer_fdw set contactnumber = '123456789' where id = 2;
DELETE FROM customer_fdw where id = 1;
TRUNCATE TABLE customer_fdw;
```

7. You can validate an SQL query plan by using the EXPLAIN statement to analyze the query plan for a SELECT query on the `customer_fdw` table:

```
EXPLAIN select * from customer_fdw where id =1;
```

This can help you understand how the query is being run and how to optimize it. For more information about using the EXPLAIN statement, see Optimizing PostgreSQL query performance in AWS Prescriptive Guidance.

8. To import multiple tables from the remote database into a local schema, use the IMPORT FOREIGN SCHEMA command:

```
CREATE SCHEMA public_fdw;
IMPORT FOREIGN SCHEMA public LIMIT TO (employees, departments)
```

```
        FROM SERVER my_fdw_target INTO public_fdw;
```

This creates local foreign tables for specified tables in the `public_fdw` schema. In this example, the specific tables are employees and departments.

9. To grant the necessary permissions to a specific database user so that they can access and use the FDW and the associated foreign server, run the following commands:

```
GRANT USAGE ON FOREIGN SERVER my_fdw_target TO targetdbuser;
GRANT USAGE ON FOREIGN DATA WRAPPER postgres_fdw TO targetdbuser;
```

This step can be beneficial when multiple users require access to the foreign tables facilitated by the foreign data wrapper.

When using foreign tables, be aware of the following limitations:

- Accessing data from a remote source can introduce data transfer costs and performance overhead caused by network latency. Performance issues can be noticeable for large data sets or queries that require significant data transfer between the Aurora PostgreSQL-Compatible instance and the remote data source.

- In complex queries that involve features such as window functions, recursive queries might not work as expected or might not be supported.

- Currently, password encryption is not supported. Implement controls to ensure that only authorized users can access the FDWs and retrieve data from remote databases.

- Primary key constraints can't be defined on foreign tables, as demonstrated by the following table-creation script attempt:

```
CREATE FOREIGN TABLE customer_fdw2( id int primary key, name varchar, emailid
 varchar, projectname varchar, contactnumber bigint) server my_fdw_target
 OPTIONS( TABLE_NAME 'customers');
Primary keys cannot be defined on Foreign table
```

- The `ON CONFLICT` clause for `INSERT` statements isn't supported on foreign tables, as shown in the following example:

```
INSERT INTO customer_fdw (id, name, emailid, projectname, contactnumber) VALUES
(1, 'test1', 'test@email.com', 'LMS', 11111111 ),
(3, 'test3', 'test3@email.com', 'LMS', 22222222 )
```

```
ON CONFLICT (id) DO UPDATE
SET name = EXCLUDED.name;
On Conflict option doesnot work.
```

Cleanup

To clean up the created objects, including dropping the `postgres_fdw` extension, the `my_fdw_target` server, user mappings, and foreign tables, run the following commands:

```
DROP FOREIGN TABLE customer_fdw;
DROP USER MAPPING for postgres;
DROP SERVER  my_fdw_target;
DROP EXTENSION postgres_fdw cascade;
```

Using dblink to create connections

The `dblink` module functions provide an alternative way to create connections and run SQL statements on remote PostgreSQL databases. The `dblink` solution is a simpler and more flexible way to run one-time queries or operations on remote databases. For more complex scenarios that involve large-scale data integration, performance optimization, and data-integrity requirements, we recommend using `postgres_fdw`.

Using `dblink` involves the following high-level steps:

1. Create the `dblink` extension:

   ```
   CREATE EXTENSION dblink;
   ```

   This extension provides the functionality to connect to remote PostgreSQL databases.

2. To establish a connection to a remote PostgreSQL database, use the `dblink_connect` function:

   ```
   SELECT dblink_connect('myconn', 'dbname=postgres port=5432 host=SOURCE_HOSTNAME
     user=postgres password=postgres');
   ```

3. After you connect to the remote PostgreSQL database, run SQL statements on the remote database by using `dblink` functions:

```
SELECT  FROM dblink('myconn', 'SELECT col1, col2 FROM remote_table') AS
  remote_data(col1 int, col2 text);
```

This query runs the `SELECT * FROM remote_table` statement on the remote database by using the `myconn` connection. The query retrieves the results into a local temporary table with columns `col1` and `col2`.

4. You can also run non-query statements, such as `INSERT`, `UPDATE`, or `DELETE`, on the remote database by using the `dblink_exec` function:

```
SELECT dblink_exec('myconn', 'INSERT INTO remote_table VALUES (1, ''value'')');
```

Aurora PostgreSQL-Compatible integration with heterogeneous databases

To integrate Aurora PostgreSQL-Compatible with remote SQL Server databases, use the Tabular Data Stream foreign data wrapper (`tds_fdw`) extension. By using the `tds_fdw` extension, you can implement federated query capability to interact with any remote SQL Server based database, both on premises and managed or self-managed on [Amazon EC2](). The `tds_fdw` extension is available in all currently supported versions of Amazon RDS for PostgreSQL and Aurora PostgreSQL-Compatible.

tds_fdw use cases and high-level steps

Integrating Aurora PostgreSQL-Compatible with heterogeneous databases such as SQL Server supports the following use cases:

- **Hybrid architectures** – Your organization might have existing SQL Server databases that must coexist and integrate with Aurora PostgreSQL-Compatible. In such cases, Aurora PostgreSQL-Compatible can be part of a hybrid architecture, where it interacts with the heterogeneous databases to exchange data or perform specific operations. With this integration, your organization can use the strengths of different database platforms while maintaining your existing investments.

- **Reporting and analytics** – You can use Aurora PostgreSQL-Compatible as a reporting or analytics database. You can consolidate data from multiple sources, including Oracle and SQL

Server databases. This use case is common in scenarios where organizations want to create specialized reporting databases or data marts tailored to specific business units or use cases.

To configure the `tds_fwd` extension in Aurora PostgreSQL-Compatible, use the following high-level steps:

1. Connect to your Aurora PostgreSQL-Compatible cluster by using a PostgreSQL client, and create the `tds_fdw` extension:

```
CREATE EXTENSION tds_fdw;
```

This extension provides the functionality to access and query data from remote SQL Server databases as if they were local tables.

2. Create a server object that represents the remote SQL Server or TDS-compatible database that you want to connect to.

3. Ensure that the necessary security groups and network configurations are in place to allow Aurora PostgreSQL-Compatible to connect to the remote SQL Server database.

   If the remote database is hosted on premises, you might need to configure a VPN or AWS Direct Connect connection.

   Run the following command:

```
CREATE SERVER my_remote_sql_server
    FOREIGN DATA WRAPPER tds_fdw
    OPTIONS (
        servername 'your_server_name',
        port '1433',
        instance 'your_instance_name'
    );
```

4. Define a user mapping that maps an Aurora PostgreSQL-Compatible user to a user on the remote SQL Server or TDS-compatible database:

```
CREATE USER MAPPING FOR postgres
    SERVER my_remote_sql_server
    OPTIONS (
        username 'your_sql_server_username',
        password 'your_sql_server_password'
```

```
    );
```

5. Create a foreign table that represents a table or view in the remote SQL Server or TDS-compatible database:

```
CREATE FOREIGN TABLE sql_server_table (
    column1 INTEGER,
    column2 VARCHAR(50)
) SERVER my_remote_sql_server
    OPTIONS (
        schema_name 'your_schema_name',
        table_name 'your_table_name'
    );
```

6. Create a foreign table based on the SQL query:

```
CREATE FOREIGN TABLE mssql_people ( empno INT NOT NULL , ename VARCHAR(10) NULL, dept
 INT) SERVER my_remote_sql_server OPTIONS (  query 'SELECT empno, ename, dept FROM
 dbo.emp');
```

7. Query the foreign table as you would query any other table in Aurora PostgreSQL-Compatible:

```
SELECT * FROM sql_server_table;
SELECT * FROM mssql_people; -- Query based on table
```

8. Import the table from SQL Server to PostgreSQL:

```
IMPORT FOREIGN SCHEMA dbo LIMIT TO (emp) FROM SERVER
                        my_remote_sql_server INTO public_fdw;
```

9. To validate the query plan, run EXPLAIN SELECT:

```
EXPLAIN SELECT * FROM mssql_people;
```

> **ⓘ Note**
>
> Data Manipulation Language (DML) operations are not available through the `tds_fdw`
> extension. The system doesn't support performing DML operations across different

database engines. INSERT, DELETE, UPDATE, and TRUNCATE   TABLE will not be successful
on the remote SQL server.

Aurora PostgreSQL-Compatible integration with CloudWatch Logs

Amazon CloudWatch Logs is a log-management service for collecting, monitoring, and analyzing logs from various AWS services. You can stream Amazon Aurora PostgreSQL-Compatible Edition logs, including error logs, slow-query logs, and audit logs, to CloudWatch Logs. You can centralize and monitor your database logs in real time, which makes identifying and troubleshooting issues easier.

You can monitor the logs by using AWS Management Console, or you can query the logs by using the `log_fdw` extension. The `log_fdw` extension supports querying and analyzing PostgreSQL log files directly from within the database. This helps you to gain insights into database performance, troubleshoot issues, and proactively identify potential problems.

To set up `log_fdw` integration with CloudWatch Logs, use the following high-level steps:

1. Connect to your Aurora PostgreSQL-Compatible cluster by using a PostgreSQL client, and create the `log_fdw` extension:

   ```
   CREATE EXTENSION log_fdw;
   ```

   This extension provides the functionality to connect to CloudWatch Logs.

2. Create a log server named `log_server` that points to the directory where PostgreSQL log files are stored. The default location for Aurora PostgreSQL-Compatible log files is `/rdsdbdata/log/`:

   ```
   CREATE SERVER log_server FOREIGN DATA WRAPPER log_fdw OPTIONS (log_directory '/
   rdsdbdata/log/');
   ```

   For more information about publishing log files from Aurora PostgreSQL-Compatible to CloudWatch Logs, see the [AWS documentation](#).

3. To list all the available log files that can be accessed through the `log_fdw` extension, run the following query:

   ```
   SELECT * FROM log_file_list('log_server');
   ```

4. To create a foreign table `log_table` that maps to the `postgres.log` file, run the following command:

```
SELECT create_foreign_table_for_log_file('log_table', 'log_server', 'postgres.log');
```

The table columns correspond to the fields that are present in the PostgreSQL log file format.

5. You can now query the log data as if it were a regular table, filtering and analyzing the log entries based on your requirements:

```
SELECT * FROM log_table
```

Cleanup

To clean up the created objects, including dropping the `log_fdw` extension, server, and foreign tables, run the following commands:

```
DROP FOREIGN TABLE log_table;
DROP SERVER log_server;
DROP EXTENSION log_fdw;
```

Aurora PostgreSQL-Compatible integration with Amazon S3

Amazon Simple Storage Service (Amazon S3) is an object storage service that provides scalable, durable, highly available, and cost-effective data storage. Amazon Aurora PostgreSQL-Compatible Edition integrates with Amazon S3 through the `aws_s3` extension, which provides direct read and write access to S3 buckets. This integration facilitates data exchange, including data ingestion, backups, and other data-related operations.

aws_s3 use cases and high-level steps

The most common high-level use cases and benefits of integrating with Amazon S3 are the following:

- **Data ingestion from Amazon S3** – Use the `aws_s3` extension to load data from comma-separated values (CSV), JSON, or other file formats stored in Amazon S3 directly into an Aurora PostgreSQL-Compatible table. This is particularly useful for batch data-ingestion processes, ETL (extract, transform, and load) workflows, or data migrations.

- **Data export to Amazon S3** – Export data from Aurora PostgreSQL-Compatible tables to CSV, JSON, or other file formats, and store the data in Amazon S3. This is useful for data archiving, backups, or sharing data with other systems or services.

- **Querying data directly from Amazon S3** – Query data stored in CSV or JSON files in Amazon S3 directly from your Aurora PostgreSQL-Compatible database without loading the data into tables. This is useful for one-time data analysis or exploratory data processing.

- **Backup and restore** – Use Amazon S3 as a backup destination for your Aurora PostgreSQL-Compatible databases. This provides an additional layer of data protection, and you can restore databases from the Amazon S3 backups if needed.

To integrate your Aurora PostgreSQL-Compatible DB cluster with an S3 bucket, use the following high-level steps:

1. Connect to your Aurora PostgreSQL-Compatible cluster by using a PostgreSQL client, and create the `aws_s3` extension:

```
create extension aws_s3
```

2. Set up access to an S3 bucket and required roles. For detailed steps, see the AWS documentation.

3. Use a psql query to import or export the data from the database:

- To import the file from Amazon S3 to an Aurora PostgreSQL-Compatible table, run the following commands:

```
SELECT aws_s3.table_import_from_s3( 'Table_Name', '',  '(format text)',

  aws_commons.create_s3_uri('S3_BUCKETNAME', 'FileName.dat','Region-Name')  );
```

- To export the file to Amazon S3 from the Aurora PostgreSQL-Compatible table, run the following command:

```
SELECT * FROM aws_s3.query_export_to_s3('TABLE_NAME',
  aws_commons.create_s3_uri('S3_BUCKETNAME', 'FileName.dat', 'Region-Name') );
```

- To export to Amazon S3 by using an SQL query, run the following command:

```
SELECT * FROM aws_s3.query_export_to_s3('SELECT * FROM data_table',
  aws_commons.create_s3_uri('S3_BUCKETNAME', 'FileName.dat', 'Region-Name') );
```

Aurora PostgreSQL-Compatible integration with Lambda

AWS Lambda is a serverless computing service for running code without provisioning or managing servers. By integrating Lambda with Amazon Aurora PostgreSQL-Compatible Edition, you can build event-driven architectures and extend the functionality of your Aurora PostgreSQL-Compatible database.

Lambda integration use cases

Common use cases for integrating Aurora PostgreSQL-Compatible with Lambda include the following:

- **Data processing and transformation** – Offload complex data processing tasks from Aurora PostgreSQL-Compatible to Lambda functions. Scenarios can be data cleansing, data enrichment, data validation, and complex calculations.

- **Event-driven workflows** – Use Lambda functions to trigger actions or workflows based on events or changes in Aurora PostgreSQL-Compatible. Scenarios include sending notifications, triggering ETL processes, or invoking other AWS services when data is inserted, updated, or deleted in Aurora PostgreSQL-Compatible.

- **Real-time analytics and reporting** – Use Lambda functions to perform real-time analytics or generate reports based on data stored in Aurora PostgreSQL-Compatible. Lambda functions can query Aurora PostgreSQL-Compatible, process the data, and generate reports or visualizations on-demand or based on a schedule.

- **Serverless APIs and microservices** – Use Lambda functions to build serverless APIs or microservices that interact with Aurora PostgreSQL-Compatible. Lambda functions can handle API requests, query or modify data in Aurora PostgreSQL-Compatible, and return the response.

- **Asynchronous processing** – Offload long-running or asynchronous tasks from Aurora PostgreSQL-Compatible to Lambda functions. Scenarios include sending email messages, generating reports, or processing large datasets without blocking the main application or database. Long-running tasks must be within the Lambda 15-minute time limit.

To set up integration between Aurora PostgreSQL-Compatible and Lambda, follow the instructions in the AWS documentation.

Aurora PostgreSQL-Compatible integration with AWS DMS

AWS Database Migration Service (AWS DMS) helps you to migrate relational databases, data warehouses, NoSQL databases, and other data stores between different engines. AWS DMS supports homogeneous migrations (for example, PostgreSQL to PostgreSQL) and heterogeneous migrations (for example, Oracle, SQL Server, or MySQL to Amazon Aurora PostgreSQL-Compatible Edition).

By integrating Aurora PostgreSQL-Compatible with AWS DMS, you minimize downtime. AWS DMS also helps to ensure data consistency during when migrating from on premises or other cloud environments to Aurora PostgreSQL-Compatible. Additionally, AWS DMS supports various source and target database engines, providing flexibility in migrating workloads to Aurora PostgreSQL-Compatible.

AWS DMS use cases and high-level steps

AWS DMS integration with Aurora PostgreSQL-Compatible supports the following use cases:

- **Migrating from on-premises databases** – Use AWS DMS to migrate existing on-premises databases (for example, Oracle, SQL Server, MySQL, or PostgreSQL) to Aurora PostgreSQL-Compatible.
- **Migrating from other cloud databases** – Use AWS DMS to facilitate migration of databases hosted on other engines or cloud platforms (for example, Amazon RDS, Azure SQL Database, or Google Cloud SQL) to Aurora PostgreSQL-Compatible.
- **Migrating between Aurora PostgreSQL-Compatible clusters** – Use AWS DMS to migrate data between Aurora PostgreSQL-Compatible clusters, either within the same AWS Region or across different Regions.
- **Continuous data replication and CDC** – Use AWS DMS for continuous data replication and change data capture (CDC) from a source database to an Aurora PostgreSQL-Compatible. This is useful for maintaining a live replica or data warehouse for analytics purposes.

To configure AWS DMS, use the following high-level steps:

1. Set up the AWS DMS replication instance in the AWS Region that you want to use.

2. Create a source endpoint in AWS DMS, specifying the details of the database from which you want to migrate data.

3. Create a target endpoint in AWS DMS, specifying the details of your Aurora PostgreSQL-Compatible cluster.

4. Configure the migration task in AWS DMS, specifying the source and target endpoints and migration type. The type can be full load, change data capture (CDC), or both. Specify any necessary mapping rules or transformations.

5. Start the migration task.

AWS DMS will handle the data transfer and replication from the source database to the target Aurora PostgreSQL-Compatible cluster.

For detailed instructions, see the AWS DMS documentation.

Aurora PostgreSQL-Compatible integration with AWS Glue

AWS Glue is a fully managed extract, transform, and load (ETL) service for preparing and loading data for analytics. You can integrate AWS Glue with Amazon Aurora PostgreSQL-Compatible Edition for any data processing and analytics workflows.

AWS Glue use cases and high-level steps

Integration of Aurora PostgreSQL-Compatible with AWS Glue supports the following use cases:

- **Data warehousing and analytics** – Use AWS Glue integration with Aurora PostgreSQL-Compatible to build data warehousing and analytics solutions. AWS Glue can extract data from Aurora PostgreSQL-Compatible databases, and transform it according to your requirements. Then AWS Glue can load the transformed data into a data warehouse such as Amazon Redshift or Amazon Athena for advanced analytics and reporting.

- **Data lake creation** – Use AWS Glue to extract data from Aurora PostgreSQL-Compatible and load it into a data lake stored in Amazon S3. You can then use this data lake for various purposes, such as machine learning, data exploration, or feeding other analytical systems.

- **ETL pipelines** – Use the AWS Glue serverless ETL service to build robust data pipelines. You can extract data from Aurora PostgreSQL-Compatible, and perform complex transformations by using Apache Spark or PySpark. You can load the processed data into a target such as Amazon S3 or Amazon Redshift, or you can load it back into Aurora PostgreSQL-Compatible.

- **Data cataloging and metadata management** – Use AWS Glue Data Catalog to automatically crawl and catalog metadata from Aurora PostgreSQL-Compatible databases and tables. AWS services such as Amazon Athena and Amazon Redshift Spectrum can use this centralized metadata repository for querying and analyzing data.

- **Data preparation for machine learning** – Use AWS Glue to prepare data from Aurora PostgreSQL-Compatible for machine learning (ML) workloads. The processed data can be loaded into Amazon SageMaker AI or other ML services for training and deploying models.

- **Data migration and replication** – While AWS Database Migration Service (AWS DMS) is the primary service for database migrations, you can also use AWS Glue. Migrate or replicate data from Aurora PostgreSQL-Compatible to other data stores, such as Amazon S3, Amazon Redshift, or even other database engines.

Your organization can use the power of AWS data integration and analytics services with the scalability, performance, and compatibility of Aurora PostgreSQL-Compatible. With these use cases, you can build robust data pipelines, perform complex data transformations, and integrate with other AWS services for advanced analytics and reporting.

To integrate Aurora PostgreSQL-Compatible with AWS Glue, use the following high-level steps:

1. Sign in to the AWS Management Console, navigate to the AWS Glue console, and create an AWS Glue Data Catalog.

   Data Catalog is a central repository that stores metadata about your data sources, including Aurora PostgreSQL-Compatible databases and tables.

2. Create an AWS Glue connection.

   Navigate to the **Connections** page, and create an AWS Glue connection. Select **Aurora PostgreSQL-Compatible** as the connection type, and provide the Aurora PostgreSQL-Compatible cluster endpoint, database name, and your database username and password.

3. Crawl the Aurora PostgreSQL-Compatible data source.

   Navigate to the **Crawlers** section, and create a crawler configured to use the connection that you created. Specify the database and table names that you want to crawl and include in the Data Catalog, and run the crawler.

4. Create and run an AWS Glue ETL job.

   Navigate to the **Jobs** section, and create an ETL job to access and query data from the Aurora PostgreSQL-Compatible database by using the Data Catalog. Choose the job type based on your requirements. In the ETL job script, perform any necessary transformations or processing, and specify the target location for the processed data. The target location can be Amazon S3, Amazon Redshift, or another Aurora PostgreSQL-Compatible database.

For detailed instructions, see the [AWS Glue documentation](#).

Aurora PostgreSQL-Compatible integration with Amazon Redshift

Amazon Redshift is a fully managed, petabyte-scale data warehousing service for large-scale data analytics and business intelligence workloads. Integration provides efficient data movement and analysis between Amazon Redshift and Amazon Aurora PostgreSQL-Compatible Edition.

AWS supports zero-ETL integration between these two services. Zero-ETL for Aurora PostgreSQL-Compatible is currently in a public preview version. For more information, see the [AWS documentation](#).

Resources

- [Foreign data wrappers](#) (PostgreSQL wiki)
- [Federated query support for Amazon Aurora PostgreSQL and Amazon RDS for PostgreSQL – Part 1](#) (AWS blog post)
- [Federated query support for Amazon Aurora PostgreSQL and Amazon RDS for PostgreSQL – Part 2](#) (AWS blog post)
- [Integrating Amazon Aurora PostgreSQL with other AWS services](#)
- [Amazon CloudWatch Logs documentation](#)
- [AWS DMS documentation](#)
- [AWS Glue documentation](#)
- [AWS Lambda documentation](#)
- [Amazon S3 documentation](#)

Document history

The following table describes significant changes to this guide. If you want to be notified about future updates, you can subscribe to an [RSS feed](#).

Change	Description	Date
[Initial publication](#)	—	August 22, 2024

AWS Prescriptive Guidance glossary

The following are commonly used terms in strategies, guides, and patterns provided by AWS Prescriptive Guidance. To suggest entries, please use the **Provide feedback** link at the end of the glossary.

Numbers

7 Rs

Seven common migration strategies for moving applications to the cloud. These strategies build upon the 5 Rs that Gartner identified in 2011 and consist of the following:

- Refactor/re-architect – Move an application and modify its architecture by taking full advantage of cloud-native features to improve agility, performance, and scalability. This typically involves porting the operating system and database. Example: Migrate your on-premises Oracle database to the Amazon Aurora PostgreSQL-Compatible Edition.

- Replatform (lift and reshape) – Move an application to the cloud, and introduce some level of optimization to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Amazon Relational Database Service (Amazon RDS) for Oracle in the AWS Cloud.

- Repurchase (drop and shop) – Switch to a different product, typically by moving from a traditional license to a SaaS model. Example: Migrate your customer relationship management (CRM) system to Salesforce.com.

- Rehost (lift and shift) – Move an application to the cloud without making any changes to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Oracle on an EC2 instance in the AWS Cloud.

- Relocate (hypervisor-level lift and shift) – Move infrastructure to the cloud without purchasing new hardware, rewriting applications, or modifying your existing operations. You migrate servers from an on-premises platform to a cloud service for the same platform. Example: Migrate a Microsoft Hyper-V application to AWS.

- Retain (revisit) – Keep applications in your source environment. These might include applications that require major refactoring, and you want to postpone that work until a later time, and legacy applications that you want to retain, because there's no business justification for migrating them.

- Retire – Decommission or remove applications that are no longer needed in your source
  environment.

A

ABAC

> See [attribute-based access control](#).

abstracted services

> See [managed services](#).

ACID

> See [atomicity, consistency, isolation, durability](#).

active-active migration

> A database migration method in which the source and target databases are kept in sync (by
> using a bidirectional replication tool or dual write operations), and both databases handle
> transactions from connecting applications during migration. This method supports migration in
> small, controlled batches instead of requiring a one-time cutover. It's more flexible but requires
> more work than [active-passive migration](#).

active-passive migration

> A database migration method in which the source and target databases are kept in sync, but
> only the source database handles transactions from connecting applications while data is
> replicated to the target database. The target database doesn't accept any transactions during
> migration.

aggregate function

> A SQL function that operates on a group of rows and calculates a single return value for the
> group. Examples of aggregate functions include SUM and MAX.

AI

> See [artificial intelligence](#).

AIOps

> See [artificial intelligence operations](#).

anonymization

The process of permanently deleting personal information in a dataset. Anonymization can help protect personal privacy. Anonymized data is no longer considered to be personal data.

anti-pattern

A frequently used solution for a recurring issue where the solution is counter-productive, ineffective, or less effective than an alternative.

application control

A security approach that allows the use of only approved applications in order to help protect a system from malware.

application portfolio

A collection of detailed information about each application used by an organization, including the cost to build and maintain the application, and its business value. This information is key to the portfolio discovery and analysis process and helps identify and prioritize the applications to be migrated, modernized, and optimized.

artificial intelligence (AI)

The field of computer science that is dedicated to using computing technologies to perform cognitive functions that are typically associated with humans, such as learning, solving problems, and recognizing patterns. For more information, see What is Artificial Intelligence?

artificial intelligence operations (AIOps)

The process of using machine learning techniques to solve operational problems, reduce operational incidents and human intervention, and increase service quality. For more information about how AIOps is used in the AWS migration strategy, see the operations integration guide.

asymmetric encryption

An encryption algorithm that uses a pair of keys, a public key for encryption and a private key for decryption. You can share the public key because it isn't used for decryption, but access to the private key should be highly restricted.

atomicity, consistency, isolation, durability (ACID)

A set of software properties that guarantee the data validity and operational reliability of a database, even in the case of errors, power failures, or other problems.

attribute-based access control (ABAC)

The practice of creating fine-grained permissions based on user attributes, such as department, job role, and team name. For more information, see ABAC for AWS in the AWS Identity and Access Management (IAM) documentation.

authoritative data source

A location where you store the primary version of data, which is considered to be the most reliable source of information. You can copy data from the authoritative data source to other locations for the purposes of processing or modifying the data, such as anonymizing, redacting, or pseudonymizing it.

Availability Zone

A distinct location within an AWS Region that is insulated from failures in other Availability Zones and provides inexpensive, low-latency network connectivity to other Availability Zones in the same Region.

AWS Cloud Adoption Framework (AWS CAF)

A framework of guidelines and best practices from AWS to help organizations develop an efficient and effective plan to move successfully to the cloud. AWS CAF organizes guidance into six focus areas called perspectives: business, people, governance, platform, security, and operations. The business, people, and governance perspectives focus on business skills and processes; the platform, security, and operations perspectives focus on technical skills and processes. For example, the people perspective targets stakeholders who handle human resources (HR), staffing functions, and people management. For this perspective, AWS CAF provides guidance for people development, training, and communications to help ready the organization for successful cloud adoption. For more information, see the AWS CAF website and the AWS CAF whitepaper.

AWS Workload Qualification Framework (AWS WQF)

A tool that evaluates database migration workloads, recommends migration strategies, and provides work estimates. AWS WQF is included with AWS Schema Conversion Tool (AWS SCT). It analyzes database schemas and code objects, application code, dependencies, and performance characteristics, and provides assessment reports.

B

bad bot

A [bot](#) that is intended to disrupt or cause harm to individuals or organizations.

BCP

See [business continuity planning](#).

behavior graph

A unified, interactive view of resource behavior and interactions over time. You can use a
behavior graph with Amazon Detective to examine failed logon attempts, suspicious API
calls, and similar actions. For more information, see [Data in a behavior graph](#) in the Detective
documentation.

big-endian system

A system that stores the most significant byte first. See also [endianness](#).

binary classification

A process that predicts a binary outcome (one of two possible classes). For example, your ML
model might need to predict problems such as "Is this email spam or not spam?" or "Is this
product a book or a car?"

bloom filter

A probabilistic, memory-efficient data structure that is used to test whether an element is a
member of a set.

blue/green deployment

A deployment strategy where you create two separate but identical environments. You run the
current application version in one environment (blue) and the new application version in the
other environment (green). This strategy helps you quickly roll back with minimal impact.

bot

A software application that runs automated tasks over the internet and simulates human
activity or interaction. Some bots are useful or beneficial, such as web crawlers that index
information on the internet. Some other bots, known as *bad bots*, are intended to disrupt or
cause harm to individuals or organizations.

botnet

Networks of bots that are infected by malware and are under the control of a single party, known as a *bot herder* or *bot operator*. Botnets are the best-known mechanism to scale bots and their impact.

branch

A contained area of a code repository. The first branch created in a repository is the *main branch*. You can create a new branch from an existing branch, and you can then develop features or fix bugs in the new branch. A branch you create to build a feature is commonly referred to as a *feature branch*. When the feature is ready for release, you merge the feature branch back into the main branch. For more information, see About branches (GitHub documentation).

break-glass access

In exceptional circumstances and through an approved process, a quick means for a user to gain access to an AWS account that they don't typically have permissions to access. For more information, see the Implement break-glass procedures indicator in the AWS Well-Architected guidance.

brownfield strategy

The existing infrastructure in your environment. When adopting a brownfield strategy for a system architecture, you design the architecture around the constraints of the current systems and infrastructure. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

buffer cache

The memory area where the most frequently accessed data is stored.

business capability

What a business does to generate value (for example, sales, customer service, or marketing). Microservices architectures and development decisions can be driven by business capabilities. For more information, see the Organized around business capabilities section of the Running containerized microservices on AWS whitepaper.

business continuity planning (BCP)

A plan that addresses the potential impact of a disruptive event, such as a large-scale migration, on operations and enables a business to resume operations quickly.

C

CAF

> See [AWS Cloud Adoption Framework](#).

canary deployment

> The slow and incremental release of a version to end users. When you are confident, you deploy
> the new version and replace the current version in its entirety.

CCoE

> See [Cloud Center of Excellence](#).

CDC

> See [change data capture](#).

change data capture (CDC)

> The process of tracking changes to a data source, such as a database table, and recording
> metadata about the change. You can use CDC for various purposes, such as auditing or
> replicating changes in a target system to maintain synchronization.

chaos engineering

> Intentionally introducing failures or disruptive events to test a system's resilience. You can use
> [AWS Fault Injection Service (AWS FIS)](#) to perform experiments that stress your AWS workloads
> and evaluate their response.

CI/CD

> See [continuous integration and continuous delivery](#).

classification

> A categorization process that helps generate predictions. ML models for classification problems
> predict a discrete value. Discrete values are always distinct from one another. For example, a
> model might need to evaluate whether or not there is a car in an image.

client-side encryption

> Encryption of data locally, before the target AWS service receives it.

Cloud Center of Excellence (CCoE)

A multi-disciplinary team that drives cloud adoption efforts across an organization, including developing cloud best practices, mobilizing resources, establishing migration timelines, and leading the organization through large-scale transformations. For more information, see the CCoE posts on the AWS Cloud Enterprise Strategy Blog.

cloud computing

The cloud technology that is typically used for remote data storage and IoT device management. Cloud computing is commonly connected to edge computing technology.

cloud operating model

In an IT organization, the operating model that is used to build, mature, and optimize one or more cloud environments. For more information, see Building your Cloud Operating Model.

cloud stages of adoption

The four phases that organizations typically go through when they migrate to the AWS Cloud:

- Project – Running a few cloud-related projects for proof of concept and learning purposes

- Foundation – Making foundational investments to scale your cloud adoption (e.g., creating a landing zone, defining a CCoE, establishing an operations model)

- Migration – Migrating individual applications

- Re-invention – Optimizing products and services, and innovating in the cloud

These stages were defined by Stephen Orban in the blog post The Journey Toward Cloud-First & the Stages of Adoption on the AWS Cloud Enterprise Strategy blog. For information about how they relate to the AWS migration strategy, see the migration readiness guide.

CMDB

See configuration management database.

code repository

A location where source code and other assets, such as documentation, samples, and scripts, are stored and updated through version control processes. Common cloud repositories include GitHub or Bitbucket Cloud. Each version of the code is called a *branch*. In a microservice structure, each repository is devoted to a single piece of functionality. A single CI/CD pipeline can use multiple repositories.

cold cache

A buffer cache that is empty, not well populated, or contains stale or irrelevant data. This affects performance because the database instance must read from the main memory or disk, which is slower than reading from the buffer cache.

cold data

Data that is rarely accessed and is typically historical. When querying this kind of data, slow queries are typically acceptable. Moving this data to lower-performing and less expensive storage tiers or classes can reduce costs.

computer vision (CV)

A field of AI that uses machine learning to analyze and extract information from visual formats such as digital images and videos. For example, Amazon SageMaker AI provides image processing algorithms for CV.

configuration drift

For a workload, a configuration change from the expected state. It might cause the workload to become noncompliant, and it's typically gradual and unintentional.

configuration management database (CMDB)

A repository that stores and manages information about a database and its IT environment, including both hardware and software components and their configurations. You typically use data from a CMDB in the portfolio discovery and analysis stage of migration.

conformance pack

A collection of AWS Config rules and remediation actions that you can assemble to customize your compliance and security checks. You can deploy a conformance pack as a single entity in an AWS account and Region, or across an organization, by using a YAML template. For more information, see Conformance packs in the AWS Config documentation.

continuous integration and continuous delivery (CI/CD)

The process of automating the source, build, test, staging, and production stages of the software release process. CI/CD is commonly described as a pipeline. CI/CD can help you automate processes, improve productivity, improve code quality, and deliver faster. For more information, see Benefits of continuous delivery. CD can also stand for *continuous deployment*. For more information, see Continuous Delivery vs. Continuous Deployment.

CV

See [computer vision](#).

D

data at rest

Data that is stationary in your network, such as data that is in storage.

data classification

A process for identifying and categorizing the data in your network based on its criticality and
sensitivity. It is a critical component of any cybersecurity risk management strategy because
it helps you determine the appropriate protection and retention controls for the data. Data
classification is a component of the security pillar in the AWS Well-Architected Framework. For
more information, see [Data classification](#).

data drift

A meaningful variation between the production data and the data that was used to train an ML
model, or a meaningful change in the input data over time. Data drift can reduce the overall
quality, accuracy, and fairness in ML model predictions.

data in transit

Data that is actively moving through your network, such as between network resources.

data mesh

An architectural framework that provides distributed, decentralized data ownership with
centralized management and governance.

data minimization

The principle of collecting and processing only the data that is strictly necessary. Practicing
data minimization in the AWS Cloud can reduce privacy risks, costs, and your analytics carbon
footprint.

data perimeter

A set of preventive guardrails in your AWS environment that help make sure that only trusted
identities are accessing trusted resources from expected networks. For more information, see
[Building a data perimeter on AWS](#).

data preprocessing

To transform raw data into a format that is easily parsed by your ML model. Preprocessing data can mean removing certain columns or rows and addressing missing, inconsistent, or duplicate values.

data provenance

The process of tracking the origin and history of data throughout its lifecycle, such as how the data was generated, transmitted, and stored.

data subject

An individual whose data is being collected and processed.

data warehouse

A data management system that supports business intelligence, such as analytics. Data warehouses commonly contain large amounts of historical data, and they are typically used for queries and analysis.

database definition language (DDL)

Statements or commands for creating or modifying the structure of tables and objects in a database.

database manipulation language (DML)

Statements or commands for modifying (inserting, updating, and deleting) information in a database.

DDL

See database definition language.

deep ensemble

To combine multiple deep learning models for prediction. You can use deep ensembles to obtain a more accurate prediction or for estimating uncertainty in predictions.

deep learning

An ML subfield that uses multiple layers of artificial neural networks to identify mapping between input data and target variables of interest.

defense-in-depth

An information security approach in which a series of security mechanisms and controls are thoughtfully layered throughout a computer network to protect the confidentiality, integrity, and availability of the network and the data within. When you adopt this strategy on AWS, you add multiple controls at different layers of the AWS Organizations structure to help secure resources. For example, a defense-in-depth approach might combine multi-factor authentication, network segmentation, and encryption.

delegated administrator

In AWS Organizations, a compatible service can register an AWS member account to administer the organization's accounts and manage permissions for that service. This account is called the *delegated administrator* for that service. For more information and a list of compatible services, see Services that work with AWS Organizations in the AWS Organizations documentation.

deployment

The process of making an application, new features, or code fixes available in the target environment. Deployment involves implementing changes in a code base and then building and running that code base in the application's environments.

development environment

See environment.

detective control

A security control that is designed to detect, log, and alert after an event has occurred. These controls are a second line of defense, alerting you to security events that bypassed the preventative controls in place. For more information, see Detective controls in *Implementing security controls on AWS*.

development value stream mapping (DVSM)

A process used to identify and prioritize constraints that adversely affect speed and quality in a software development lifecycle. DVSM extends the value stream mapping process originally designed for lean manufacturing practices. It focuses on the steps and teams required to create and move value through the software development process.

digital twin

A virtual representation of a real-world system, such as a building, factory, industrial equipment, or production line. Digital twins support predictive maintenance, remote monitoring, and production optimization.

dimension table

In a star schema, a smaller table that contains data attributes about quantitative data in a fact table. Dimension table attributes are typically text fields or discrete numbers that behave like text. These attributes are commonly used for query constraining, filtering, and result set labeling.

disaster

An event that prevents a workload or system from fulfilling its business objectives in its primary deployed location. These events can be natural disasters, technical failures, or the result of human actions, such as unintentional misconfiguration or a malware attack.

disaster recovery (DR)

The strategy and process you use to minimize downtime and data loss caused by a disaster. For more information, see Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

See database manipulation language.

domain-driven design

An approach to developing a complex software system by connecting its components to evolving domains, or core business goals, that each component serves. This concept was introduced by Eric Evans in his book, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley Professional, 2003). For information about how you can use domain-driven design with the strangler fig pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

See disaster recovery.

drift detection

Tracking deviations from a baselined configuration. For example, you can use AWS CloudFormation to [detect drift in system resources](#), or you can use AWS Control Tower to [detect changes in your landing zone](#) that might affect compliance with governance requirements.

DVSM

See [development value stream mapping](#).

E

EDA

See [exploratory data analysis](#).

EDI

See [electronic data interchange](#).

edge computing

The technology that increases the computing power for smart devices at the edges of an IoT network. When compared with [cloud computing](#), edge computing can reduce communication latency and improve response time.

electronic data interchange (EDI)

The automated exchange of business documents between organizations. For more information, see [What is Electronic Data Interchange](#).

encryption

A computing process that transforms plaintext data, which is human-readable, into ciphertext.

encryption key

A cryptographic string of randomized bits that is generated by an encryption algorithm. Keys can vary in length, and each key is designed to be unpredictable and unique.

endianness

The order in which bytes are stored in computer memory. Big-endian systems store the most significant byte first. Little-endian systems store the least significant byte first.

endpoint

See [service endpoint](#).

endpoint service

A service that you can host in a virtual private cloud (VPC) to share with other users. You can create an endpoint service with AWS PrivateLink and grant permissions to other AWS accounts or to AWS Identity and Access Management (IAM) principals. These accounts or principals can connect to your endpoint service privately by creating interface VPC endpoints. For more information, see [Create an endpoint service](#) in the Amazon Virtual Private Cloud (Amazon VPC) documentation.

enterprise resource planning (ERP)

A system that automates and manages key business processes (such as accounting, [MES](#), and project management) for an enterprise.

envelope encryption

The process of encrypting an encryption key with another encryption key. For more information, see [Envelope encryption](#) in the AWS Key Management Service (AWS KMS) documentation.

environment

An instance of a running application. The following are common types of environments in cloud computing:

- development environment – An instance of a running application that is available only to the core team responsible for maintaining the application. Development environments are used to test changes before promoting them to upper environments. This type of environment is sometimes referred to as a *test environment*.

- lower environments – All development environments for an application, such as those used for initial builds and tests.

- production environment – An instance of a running application that end users can access. In a CI/CD pipeline, the production environment is the last deployment environment.

- upper environments – All environments that can be accessed by users other than the core development team. This can include a production environment, preproduction environments, and environments for user acceptance testing.

epic

In agile methodologies, functional categories that help organize and prioritize your work. Epics provide a high-level description of requirements and implementation tasks. For example, AWS CAF security epics include identity and access management, detective controls, infrastructure security, data protection, and incident response. For more information about epics in the AWS migration strategy, see the program implementation guide.

ERP

See enterprise resource planning.

exploratory data analysis (EDA)

The process of analyzing a dataset to understand its main characteristics. You collect or aggregate data and then perform initial investigations to find patterns, detect anomalies, and check assumptions. EDA is performed by calculating summary statistics and creating data visualizations.

F

fact table

The central table in a star schema. It stores quantitative data about business operations. Typically, a fact table contains two types of columns: those that contain measures and those that contain a foreign key to a dimension table.

fail fast

A philosophy that uses frequent and incremental testing to reduce the development lifecycle. It is a critical part of an agile approach.

fault isolation boundary

In the AWS Cloud, a boundary such as an Availability Zone, AWS Region, control plane, or data plane that limits the effect of a failure and helps improve the resilience of workloads. For more information, see AWS Fault Isolation Boundaries.

feature branch

See branch.

features

The input data that you use to make a prediction. For example, in a manufacturing context,
features could be images that are periodically captured from the manufacturing line.

feature importance

How significant a feature is for a model's predictions. This is usually expressed as a numerical
score that can be calculated through various techniques, such as Shapley Additive Explanations
(SHAP) and integrated gradients. For more information, see [Machine learning model
interpretability with AWS](#).

feature transformation

To optimize data for the ML process, including enriching data with additional sources, scaling
values, or extracting multiple sets of information from a single data field. This enables the ML
model to benefit from the data. For example, if you break down the "2021-05-27 00:15:37"
date into "2021", "May", "Thu", and "15", you can help the learning algorithm learn nuanced
patterns associated with different data components.

few-shot prompting

Providing an [LLM](#) with a small number of examples that demonstrate the task and desired
output before asking it to perform a similar task. This technique is an application of in-context
learning, where models learn from examples (*shots*) that are embedded in prompts. Few-shot
prompting can be effective for tasks that require specific formatting, reasoning, or domain
knowledge. See also [zero-shot prompting](#).

FGAC

See [fine-grained access control](#).

fine-grained access control (FGAC)

The use of multiple conditions to allow or deny an access request.

flash-cut migration

A database migration method that uses continuous data replication through [change data
capture](#) to migrate data in the shortest time possible, instead of using a phased approach. The
objective is to keep downtime to a minimum.

FM

See [foundation model](#).

foundation model (FM)

A large deep-learning neural network that has been training on massive datasets of generalized and unlabeled data. FMs are capable of performing a wide variety of general tasks, such as understanding language, generating text and images, and conversing in natural language. For more information, see What are Foundation Models.

G

generative AI

A subset of AI models that have been trained on large amounts of data and that can use a simple text prompt to create new content and artifacts, such as images, videos, text, and audio. For more information, see What is Generative AI.

geo blocking

See geographic restrictions.

geographic restrictions (geo blocking)

In Amazon CloudFront, an option to prevent users in specific countries from accessing content distributions. You can use an allow list or block list to specify approved and banned countries. For more information, see Restricting the geographic distribution of your content in the CloudFront documentation.

Gitflow workflow

An approach in which lower and upper environments use different branches in a source code repository. The Gitflow workflow is considered legacy, and the trunk-based workflow is the modern, preferred approach.

golden image

A snapshot of a system or software that is used as a template to deploy new instances of that system or software. For example, in manufacturing, a golden image can be used to provision software on multiple devices and helps improve speed, scalability, and productivity in device manufacturing operations.

greenfield strategy

The absence of existing infrastructure in a new environment. When adopting a greenfield strategy for a system architecture, you can select all new technologies without the restriction

of compatibility with existing infrastructure, also known as [brownfield](). If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

guardrail

A high-level rule that helps govern resources, policies, and compliance across organizational units (OUs). *Preventive guardrails* enforce policies to ensure alignment to compliance standards. They are implemented by using service control policies and IAM permissions boundaries. *Detective guardrails* detect policy violations and compliance issues, and generate alerts for remediation. They are implemented by using AWS Config, AWS Security Hub CSPM, Amazon GuardDuty, AWS Trusted Advisor, Amazon Inspector, and custom AWS Lambda checks.

H

HA

See [high availability]().

heterogeneous database migration

Migrating your source database to a target database that uses a different database engine (for example, Oracle to Amazon Aurora). Heterogeneous migration is typically part of a re-architecting effort, and converting the schema can be a complex task. [AWS provides AWS SCT]() that helps with schema conversions.

high availability (HA)

The ability of a workload to operate continuously, without intervention, in the event of challenges or disasters. HA systems are designed to automatically fail over, consistently deliver high-quality performance, and handle different loads and failures with minimal performance impact.

historian modernization

An approach used to modernize and upgrade operational technology (OT) systems to better serve the needs of the manufacturing industry. A *historian* is a type of database that is used to collect and store data from various sources in a factory.

holdout data

A portion of historical, labeled data that is withheld from a dataset that is used to train a
machine learning model. You can use holdout data to evaluate the model performance by
comparing the model predictions against the holdout data.

homogeneous database migration

Migrating your source database to a target database that shares the same database engine
(for example, Microsoft SQL Server to Amazon RDS for SQL Server). Homogeneous migration
is typically part of a rehosting or replatforming effort. You can use native database utilities to
migrate the schema.

hot data

Data that is frequently accessed, such as real-time data or recent translational data. This data
typically requires a high-performance storage tier or class to provide fast query responses.

hotfix

An urgent fix for a critical issue in a production environment. Due to its urgency, a hotfix is
usually made outside of the typical DevOps release workflow.

hypercare period

Immediately following cutover, the period of time when a migration team manages and
monitors the migrated applications in the cloud in order to address any issues. Typically, this
period is 1–4 days in length. At the end of the hypercare period, the migration team typically
transfers responsibility for the applications to the cloud operations team.

I

IaC

See infrastructure as code.

identity-based policy

A policy attached to one or more IAM principals that defines their permissions within the AWS
Cloud environment.

idle application

An application that has an average CPU and memory usage between 5 and 20 percent over
a period of 90 days. In a migration project, it is common to retire these applications or retain
them on premises.

IIoT

See industrial Internet of Things.

immutable infrastructure

A model that deploys new infrastructure for production workloads instead of updating,
patching, or modifying the existing infrastructure. Immutable infrastructures are inherently
more consistent, reliable, and predictable than mutable infrastructure. For more information,
see the Deploy using immutable infrastructure best practice in the AWS Well-Architected
Framework.

inbound (ingress) VPC

In an AWS multi-account architecture, a VPC that accepts, inspects, and routes network
connections from outside an application. The AWS Security Reference Architecture recommends
setting up your Network account with inbound, outbound, and inspection VPCs to protect the
two-way interface between your application and the broader internet.

incremental migration

A cutover strategy in which you migrate your application in small parts instead of performing
a single, full cutover. For example, you might move only a few microservices or users to the
new system initially. After you verify that everything is working properly, you can incrementally
move additional microservices or users until you can decommission your legacy system. This
strategy reduces the risks associated with large migrations.

Industry 4.0

A term that was introduced by Klaus Schwab in 2016 to refer to the modernization of
manufacturing processes through advances in connectivity, real-time data, automation,
analytics, and AI/ML.

infrastructure

All of the resources and assets contained within an application's environment.

infrastructure as code (IaC)

The process of provisioning and managing an application's infrastructure through a set
of configuration files. IaC is designed to help you centralize infrastructure management,
standardize resources, and scale quickly so that new environments are repeatable, reliable, and
consistent.

industrial Internet of Things (IIoT)

The use of internet-connected sensors and devices in the industrial sectors, such as
manufacturing, energy, automotive, healthcare, life sciences, and agriculture. For more
information, see Building an industrial Internet of Things (IIoT) digital transformation strategy.

inspection VPC

In an AWS multi-account architecture, a centralized VPC that manages inspections of network
traffic between VPCs (in the same or different AWS Regions), the internet, and on-premises
networks. The AWS Security Reference Architecture recommends setting up your Network
account with inbound, outbound, and inspection VPCs to protect the two-way interface
between your application and the broader internet.

Internet of Things (IoT)

The network of connected physical objects with embedded sensors or processors that
communicate with other devices and systems through the internet or over a local
communication network. For more information, see What is IoT?

interpretability

A characteristic of a machine learning model that describes the degree to which a human
can understand how the model's predictions depend on its inputs. For more information, see
Machine learning model interpretability with AWS.

IoT

See Internet of Things.

IT information library (ITIL)

A set of best practices for delivering IT services and aligning these services with business
requirements. ITIL provides the foundation for ITSM.

IT service management (ITSM)

Activities associated with designing, implementing, managing, and supporting IT services for
an organization. For information about integrating cloud operations with ITSM tools, see the
operations integration guide.

ITIL

See IT information library.

ITSM

See IT service management.

L

label-based access control (LBAC)

An implementation of mandatory access control (MAC) where the users and the data itself are
each explicitly assigned a security label value. The intersection between the user security label
and data security label determines which rows and columns can be seen by the user.

landing zone

A landing zone is a well-architected, multi-account AWS environment that is scalable and
secure. This is a starting point from which your organizations can quickly launch and deploy
workloads and applications with confidence in their security and infrastructure environment.
For more information about landing zones, see Setting up a secure and scalable multi-account
AWS environment.

large language model (LLM)

A deep learning AI model that is pretrained on a vast amount of data. An LLM can perform
multiple tasks, such as answering questions, summarizing documents, translating text into
other languages, and completing sentences. For more information, see What are LLMs.

large migration

A migration of 300 or more servers.

LBAC

See label-based access control.

least privilege

The security best practice of granting the minimum permissions required to perform a task. For more information, see Apply least-privilege permissions in the IAM documentation.

lift and shift

See 7 Rs.

little-endian system

A system that stores the least significant byte first. See also endianness.

LLM

See large language model.

lower environments

See environment.

M

machine learning (ML)

A type of artificial intelligence that uses algorithms and techniques for pattern recognition and learning. ML analyzes and learns from recorded data, such as Internet of Things (IoT) data, to generate a statistical model based on patterns. For more information, see Machine Learning.

main branch

See branch.

malware

Software that is designed to compromise computer security or privacy. Malware might disrupt computer systems, leak sensitive information, or gain unauthorized access. Examples of malware include viruses, worms, ransomware, Trojan horses, spyware, and keyloggers.

managed services

AWS services for which AWS operates the infrastructure layer, the operating system, and platforms, and you access the endpoints to store and retrieve data. Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB are examples of managed services. These are also known as *abstracted services*.

manufacturing execution system (MES)

A software system for tracking, monitoring, documenting, and controlling production processes that convert raw materials to finished products on the shop floor.

MAP

See Migration Acceleration Program.

mechanism

A complete process in which you create a tool, drive adoption of the tool, and then inspect the results in order to make adjustments. A mechanism is a cycle that reinforces and improves itself as it operates. For more information, see Building mechanisms in the AWS Well-Architected Framework.

member account

All AWS accounts other than the management account that are part of an organization in AWS Organizations. An account can be a member of only one organization at a time.

MES

See manufacturing execution system.

Message Queuing Telemetry Transport (MQTT)

A lightweight, machine-to-machine (M2M) communication protocol, based on the publish/subscribe pattern, for resource-constrained IoT devices.

microservice

A small, independent service that communicates over well-defined APIs and is typically owned by small, self-contained teams. For example, an insurance system might include microservices that map to business capabilities, such as sales or marketing, or subdomains, such as purchasing, claims, or analytics. The benefits of microservices include agility, flexible scaling, easy deployment, reusable code, and resilience. For more information, see Integrating microservices by using AWS serverless services.

microservices architecture

An approach to building an application with independent components that run each application process as a microservice. These microservices communicate through a well-defined interface by using lightweight APIs. Each microservice in this architecture can be updated, deployed,

and scaled to meet demand for specific functions of an application. For more information, see [Implementing microservices on AWS](#).

Migration Acceleration Program (MAP)

An AWS program that provides consulting support, training, and services to help organizations build a strong operational foundation for moving to the cloud, and to help offset the initial cost of migrations. MAP includes a migration methodology for executing legacy migrations in a methodical way and a set of tools to automate and accelerate common migration scenarios.

migration at scale

The process of moving the majority of the application portfolio to the cloud in waves, with more applications moved at a faster rate in each wave. This phase uses the best practices and lessons learned from the earlier phases to implement a *migration factory* of teams, tools, and processes to streamline the migration of workloads through automation and agile delivery. This is the third phase of the [AWS migration strategy](#).

migration factory

Cross-functional teams that streamline the migration of workloads through automated, agile approaches. Migration factory teams typically include operations, business analysts and owners, migration engineers, developers, and DevOps professionals working in sprints. Between 20 and 50 percent of an enterprise application portfolio consists of repeated patterns that can be optimized by a factory approach. For more information, see the [discussion of migration factories](#) and the [Cloud Migration Factory guide](#) in this content set.

migration metadata

The information about the application and server that is needed to complete the migration. Each migration pattern requires a different set of migration metadata. Examples of migration metadata include the target subnet, security group, and AWS account.

migration pattern

A repeatable migration task that details the migration strategy, the migration destination, and the migration application or service used. Example: Rehost migration to Amazon EC2 with AWS Application Migration Service.

Migration Portfolio Assessment (MPA)

An online tool that provides information for validating the business case for migrating to the AWS Cloud. MPA provides detailed portfolio assessment (server right-sizing, pricing, TCO

comparisons, migration cost analysis) as well as migration planning (application data analysis and data collection, application grouping, migration prioritization, and wave planning). The MPA tool (requires login) is available free of charge to all AWS consultants and APN Partner consultants.

Migration Readiness Assessment (MRA)

The process of gaining insights about an organization's cloud readiness status, identifying strengths and weaknesses, and building an action plan to close identified gaps, using the AWS CAF. For more information, see the migration readiness guide. MRA is the first phase of the AWS migration strategy.

migration strategy

The approach used to migrate a workload to the AWS Cloud. For more information, see the 7 Rs entry in this glossary and see Mobilize your organization to accelerate large-scale migrations.

ML

See machine learning.

modernization

Transforming an outdated (legacy or monolithic) application and its infrastructure into an agile, elastic, and highly available system in the cloud to reduce costs, gain efficiencies, and take advantage of innovations. For more information, see Strategy for modernizing applications in the AWS Cloud.

modernization readiness assessment

An evaluation that helps determine the modernization readiness of an organization's applications; identifies benefits, risks, and dependencies; and determines how well the organization can support the future state of those applications. The outcome of the assessment is a blueprint of the target architecture, a roadmap that details development phases and milestones for the modernization process, and an action plan for addressing identified gaps. For more information, see Evaluating modernization readiness for applications in the AWS Cloud.

monolithic applications (monoliths)

Applications that run as a single service with tightly coupled processes. Monolithic applications have several drawbacks. If one application feature experiences a spike in demand, the entire architecture must be scaled. Adding or improving a monolithic application's features also becomes more complex when the code base grows. To address these issues, you can

use a microservices architecture. For more information, see Decomposing monoliths into microservices.

MPA

See Migration Portfolio Assessment.

MQTT

See Message Queuing Telemetry Transport.

multiclass classification

A process that helps generate predictions for multiple classes (predicting one of more than two outcomes). For example, an ML model might ask "Is this product a book, car, or phone?" or "Which product category is most interesting to this customer?"

mutable infrastructure

A model that updates and modifies the existing infrastructure for production workloads. For improved consistency, reliability, and predictability, the AWS Well-Architected Framework recommends the use of immutable infrastructure as a best practice.

O

OAC

See origin access control.

OAI

See origin access identity.

OCM

See organizational change management.

offline migration

A migration method in which the source workload is taken down during the migration process. This method involves extended downtime and is typically used for small, non-critical workloads.

OI

See operations integration.

OLA

See [operational-level agreement](#).

online migration

A migration method in which the source workload is copied to the target system without being taken offline. Applications that are connected to the workload can continue to function during the migration. This method involves zero to minimal downtime and is typically used for critical production workloads.

OPC-UA

See [Open Process Communications - Unified Architecture](#).

Open Process Communications - Unified Architecture (OPC-UA)

A machine-to-machine (M2M) communication protocol for industrial automation. OPC-UA provides an interoperability standard with data encryption, authentication, and authorization schemes.

operational-level agreement (OLA)

An agreement that clarifies what functional IT groups promise to deliver to each other, to support a service-level agreement (SLA).

operational readiness review (ORR)

A checklist of questions and associated best practices that help you understand, evaluate, prevent, or reduce the scope of incidents and possible failures. For more information, see [Operational Readiness Reviews (ORR)](#) in the AWS Well-Architected Framework.

operational technology (OT)

Hardware and software systems that work with the physical environment to control industrial operations, equipment, and infrastructure. In manufacturing, the integration of OT and information technology (IT) systems is a key focus for [Industry 4.0](#) transformations.

operations integration (OI)

The process of modernizing operations in the cloud, which involves readiness planning, automation, and integration. For more information, see the [operations integration guide](#).

organization trail

A trail that's created by AWS CloudTrail that logs all events for all AWS accounts in an organization in AWS Organizations. This trail is created in each AWS account that's part of the

organization and tracks the activity in each account. For more information, see Creating a trail
for an organization in the CloudTrail documentation.

organizational change management (OCM)

A framework for managing major, disruptive business transformations from a people, culture,
and leadership perspective. OCM helps organizations prepare for, and transition to, new
systems and strategies by accelerating change adoption, addressing transitional issues, and
driving cultural and organizational changes. In the AWS migration strategy, this framework is
called *people acceleration*, because of the speed of change required in cloud adoption projects.
For more information, see the OCM guide.

origin access control (OAC)

In CloudFront, an enhanced option for restricting access to secure your Amazon Simple Storage
Service (Amazon S3) content. OAC supports all S3 buckets in all AWS Regions, server-side
encryption with AWS KMS (SSE-KMS), and dynamic PUT and DELETE requests to the S3 bucket.

origin access identity (OAI)

In CloudFront, an option for restricting access to secure your Amazon S3 content. When you
use OAI, CloudFront creates a principal that Amazon S3 can authenticate with. Authenticated
principals can access content in an S3 bucket only through a specific CloudFront distribution.
See also OAC, which provides more granular and enhanced access control.

ORR

See operational readiness review.

OT

See operational technology.

outbound (egress) VPC

In an AWS multi-account architecture, a VPC that handles network connections that are
initiated from within an application. The AWS Security Reference Architecture recommends
setting up your Network account with inbound, outbound, and inspection VPCs to protect the
two-way interface between your application and the broader internet.

P

permissions boundary

An IAM management policy that is attached to IAM principals to set the maximum permissions
that the user or role can have. For more information, see Permissions boundaries in the IAM
documentation.

personally identifiable information (PII)

Information that, when viewed directly or paired with other related data, can be used to
reasonably infer the identity of an individual. Examples of PII include names, addresses, and
contact information.

PII

See personally identifiable information.

playbook

A set of predefined steps that capture the work associated with migrations, such as delivering
core operations functions in the cloud. A playbook can take the form of scripts, automated
runbooks, or a summary of processes or steps required to operate your modernized
environment.

PLC

See programmable logic controller.

PLM

See product lifecycle management.

policy

An object that can define permissions (see identity-based policy), specify access conditions (see
resource-based policy), or define the maximum permissions for all accounts in an organization
in AWS Organizations (see service control policy).

polyglot persistence

Independently choosing a microservice's data storage technology based on data access patterns
and other requirements. If your microservices have the same data storage technology, they can
encounter implementation challenges or experience poor performance. Microservices are more

easily implemented and achieve better performance and scalability if they use the data store best adapted to their requirements.

portfolio assessment

A process of discovering, analyzing, and prioritizing the application portfolio in order to plan the migration. For more information, see [Evaluating migration readiness](#).

predicate

A query condition that returns `true` or `false`, commonly located in a WHERE clause.

predicate pushdown

A database query optimization technique that filters the data in the query before transfer. This reduces the amount of data that must be retrieved and processed from the relational database, and it improves query performance.

preventative control

A security control that is designed to prevent an event from occurring. These controls are a first line of defense to help prevent unauthorized access or unwanted changes to your network. For more information, see [Preventative controls](#) in *Implementing security controls on AWS*.

principal

An entity in AWS that can perform actions and access resources. This entity is typically a root user for an AWS account, an IAM role, or a user. For more information, see *Principal* in [Roles terms and concepts](#) in the IAM documentation.

privacy by design

A system engineering approach that takes privacy into account through the whole development process.

private hosted zones

A container that holds information about how you want Amazon Route 53 to respond to DNS queries for a domain and its subdomains within one or more VPCs. For more information, see [Working with private hosted zones](#) in the Route 53 documentation.

proactive control

A [security control](#) designed to prevent the deployment of noncompliant resources. These controls scan resources before they are provisioned. If the resource is not compliant with the control, then it isn't provisioned. For more information, see the [Controls reference guide](#) in the

AWS Control Tower documentation and see [Proactive controls](#) in *Implementing security controls on AWS*.

product lifecycle management (PLM)

The management of data and processes for a product throughout its entire lifecycle, from design, development, and launch, through growth and maturity, to decline and removal.

production environment

See [environment](#).

programmable logic controller (PLC)

In manufacturing, a highly reliable, adaptable computer that monitors machines and automates manufacturing processes.

prompt chaining

Using the output of one [LLM](#) prompt as the input for the next prompt to generate better responses. This technique is used to break down a complex task into subtasks, or to iteratively refine or expand a preliminary response. It helps improve the accuracy and relevance of a model's responses and allows for more granular, personalized results.

pseudonymization

The process of replacing personal identifiers in a dataset with placeholder values. Pseudonymization can help protect personal privacy. Pseudonymized data is still considered to be personal data.

publish/subscribe (pub/sub)

A pattern that enables asynchronous communications among microservices to improve scalability and responsiveness. For example, in a microservices-based [MES](#), a microservice can publish event messages to a channel that other microservices can subscribe to. The system can add new microservices without changing the publishing service.

Q

query plan

A series of steps, like instructions, that are used to access the data in a SQL relational database system.

query plan regression

> When a database service optimizer chooses a less optimal plan than it did before a given
> change to the database environment. This can be caused by changes to statistics, constraints,
> environment settings, query parameter bindings, and updates to the database engine.

R

RACI matrix

> See responsible, accountable, consulted, informed (RACI).

RAG

> See Retrieval Augmented Generation.

ransomware

> A malicious software that is designed to block access to a computer system or data until a
> payment is made.

RASCI matrix

> See responsible, accountable, consulted, informed (RACI).

RCAC

> See row and column access control.

read replica

> A copy of a database that's used for read-only purposes. You can route queries to the read
> replica to reduce the load on your primary database.

re-architect

> See 7 Rs.

recovery point objective (RPO)

> The maximum acceptable amount of time since the last data recovery point. This determines
> what is considered an acceptable loss of data between the last recovery point and the
> interruption of service.

recovery time objective (RTO)

The maximum acceptable delay between the interruption of service and restoration of service.

refactor

See 7 Rs.

Region

A collection of AWS resources in a geographic area. Each AWS Region is isolated and
independent of the others to provide fault tolerance, stability, and resilience. For more
information, see Specify which AWS Regions your account can use.

regression

An ML technique that predicts a numeric value. For example, to solve the problem of "What
price will this house sell for?" an ML model could use a linear regression model to predict a
house's sale price based on known facts about the house (for example, the square footage).

rehost

See 7 Rs.

release

In a deployment process, the act of promoting changes to a production environment.

relocate

See 7 Rs.

replatform

See 7 Rs.

repurchase

See 7 Rs.

resiliency

An application's ability to resist or recover from disruptions. High availability and disaster
recovery are common considerations when planning for resiliency in the AWS Cloud. For more
information, see AWS Cloud Resilience.

resource-based policy

A policy attached to a resource, such as an Amazon S3 bucket, an endpoint, or an encryption
key. This type of policy specifies which principals are allowed access, supported actions, and any
other conditions that must be met.

responsible, accountable, consulted, informed (RACI) matrix

A matrix that defines the roles and responsibilities for all parties involved in migration activities
and cloud operations. The matrix name is derived from the responsibility types defined in the
matrix: responsible (R), accountable (A), consulted (C), and informed (I). The support (S) type
is optional. If you include support, the matrix is called a *RASCI matrix*, and if you exclude it, it's
called a *RACI matrix*.

responsive control

A security control that is designed to drive remediation of adverse events or deviations from
your security baseline. For more information, see Responsive controls in *Implementing security
controls on AWS*.

retain

See 7 Rs.

retire

See 7 Rs.

Retrieval Augmented Generation (RAG)

A generative AI technology in which an LLM references an authoritative data source that is
outside of its training data sources before generating a response. For example, a RAG model
might perform a semantic search of an organization's knowledge base or custom data. For more
information, see What is RAG.

rotation

The process of periodically updating a secret to make it more difficult for an attacker to access
the credentials.

row and column access control (RCAC)

The use of basic, flexible SQL expressions that have defined access rules. RCAC consists of row
permissions and column masks.

RPO

See [recovery point objective](#).

RTO

See [recovery time objective](#).

runbook

A set of manual or automated procedures required to perform a specific task. These are
typically built to streamline repetitive operations or procedures with high error rates.

S

SAML 2.0

An open standard that many identity providers (IdPs) use. This feature enables federated
single sign-on (SSO), so users can log into the AWS Management Console or call the AWS API
operations without you having to create user in IAM for everyone in your organization. For more
information about SAML 2.0-based federation, see [About SAML 2.0-based federation](#) in the IAM
documentation.

SCADA

See [supervisory control and data acquisition](#).

SCP

See [service control policy](#).

secret

In AWS Secrets Manager, confidential or restricted information, such as a password or user
credentials, that you store in encrypted form. It consists of the secret value and its metadata.
The secret value can be binary, a single string, or multiple strings. For more information, see
[What's in a Secrets Manager secret?](#) in the Secrets Manager documentation.

security by design

A system engineering approach that takes security into account through the whole
development process.

security control

A technical or administrative guardrail that prevents, detects, or reduces the ability of a threat actor to exploit a security vulnerability. There are four primary types of security controls: preventative, detective, responsive, and proactive.

security hardening

The process of reducing the attack surface to make it more resistant to attacks. This can include actions such as removing resources that are no longer needed, implementing the security best practice of granting least privilege, or deactivating unnecessary features in configuration files.

security information and event management (SIEM) system

Tools and services that combine security information management (SIM) and security event management (SEM) systems. A SIEM system collects, monitors, and analyzes data from servers, networks, devices, and other sources to detect threats and security breaches, and to generate alerts.

security response automation

A predefined and programmed action that is designed to automatically respond to or remediate a security event. These automations serve as detective or responsive security controls that help you implement AWS security best practices. Examples of automated response actions include modifying a VPC security group, patching an Amazon EC2 instance, or rotating credentials.

server-side encryption

Encryption of data at its destination, by the AWS service that receives it.

service control policy (SCP)

A policy that provides centralized control over permissions for all accounts in an organization in AWS Organizations. SCPs define guardrails or set limits on actions that an administrator can delegate to users or roles. You can use SCPs as allow lists or deny lists, to specify which services or actions are permitted or prohibited. For more information, see Service control policies in the AWS Organizations documentation.

service endpoint

The URL of the entry point for an AWS service. You can use the endpoint to connect programmatically to the target service. For more information, see AWS service endpoints in *AWS General Reference*.

service-level agreement (SLA)

An agreement that clarifies what an IT team promises to deliver to their customers, such as
service uptime and performance.

service-level indicator (SLI)

A measurement of a performance aspect of a service, such as its error rate, availability, or
throughput.

service-level objective (SLO)

A target metric that represents the health of a service, as measured by a service-level indicator.

shared responsibility model

A model describing the responsibility you share with AWS for cloud security and compliance.
AWS is responsible for security *of* the cloud, whereas you are responsible for security *in* the
cloud. For more information, see Shared responsibility model.

SIEM

See security information and event management system.

single point of failure (SPOF)

A failure in a single, critical component of an application that can disrupt the system.

SLA

See service-level agreement.

SLI

See service-level indicator.

SLO

See service-level objective.

split-and-seed model

A pattern for scaling and accelerating modernization projects. As new features and product
releases are defined, the core team splits up to create new product teams. This helps scale your
organization's capabilities and services, improves developer productivity, and supports rapid

innovation. For more information, see [Phased approach to modernizing applications in the AWS Cloud](#).

SPOF

See [single point of failure](#).

star schema

A database organizational structure that uses one large fact table to store transactional or measured data and uses one or more smaller dimensional tables to store data attributes. This structure is designed for use in a [data warehouse](#) or for business intelligence purposes.

strangler fig pattern

An approach to modernizing monolithic systems by incrementally rewriting and replacing system functionality until the legacy system can be decommissioned. This pattern uses the analogy of a fig vine that grows into an established tree and eventually overcomes and replaces its host. The pattern was [introduced by Martin Fowler](#) as a way to manage risk when rewriting monolithic systems. For an example of how to apply this pattern, see [Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway](#).

subnet

A range of IP addresses in your VPC. A subnet must reside in a single Availability Zone.

supervisory control and data acquisition (SCADA)

In manufacturing, a system that uses hardware and software to monitor physical assets and production operations.

symmetric encryption

An encryption algorithm that uses the same key to encrypt and decrypt the data.

synthetic testing

Testing a system in a way that simulates user interactions to detect potential issues or to monitor performance. You can use [Amazon CloudWatch Synthetics](#) to create these tests.

system prompt

A technique for providing context, instructions, or guidelines to an [LLM](#) to direct its behavior. System prompts help set context and establish rules for interactions with users.

T

tags

>   Key-value pairs that act as metadata for organizing your AWS resources. Tags can help you
>   manage, identify, organize, search for, and filter resources. For more information, see Tagging
>   your AWS resources.

target variable

>   The value that you are trying to predict in supervised ML. This is also referred to as an *outcome
>   variable*. For example, in a manufacturing setting the target variable could be a product defect.

task list

>   A tool that is used to track progress through a runbook. A task list contains an overview of
>   the runbook and a list of general tasks to be completed. For each general task, it includes the
>   estimated amount of time required, the owner, and the progress.

test environment

>   See environment.

training

>   To provide data for your ML model to learn from. The training data must contain the correct
>   answer. The learning algorithm finds patterns in the training data that map the input data
>   attributes to the target (the answer that you want to predict). It outputs an ML model that
>   captures these patterns. You can then use the ML model to make predictions on new data for
>   which you don't know the target.

transit gateway

>   A network transit hub that you can use to interconnect your VPCs and on-premises
>   networks. For more information, see What is a transit gateway in the AWS Transit Gateway
>   documentation.

trunk-based workflow

>   An approach in which developers build and test features locally in a feature branch and then
>   merge those changes into the main branch. The main branch is then built to the development,
>   preproduction, and production environments, sequentially.

trusted access

Granting permissions to a service that you specify to perform tasks in your organization in AWS Organizations and in its accounts on your behalf. The trusted service creates a service-linked role in each account, when that role is needed, to perform management tasks for you. For more information, see Using AWS Organizations with other AWS services in the AWS Organizations documentation.

tuning

To change aspects of your training process to improve the ML model's accuracy. For example, you can train the ML model by generating a labeling set, adding labels, and then repeating these steps several times under different settings to optimize the model.

two-pizza team

A small DevOps team that you can feed with two pizzas. A two-pizza team size ensures the best possible opportunity for collaboration in software development.

U

uncertainty

A concept that refers to imprecise, incomplete, or unknown information that can undermine the reliability of predictive ML models. There are two types of uncertainty: *Epistemic uncertainty* is caused by limited, incomplete data, whereas *aleatoric uncertainty* is caused by the noise and randomness inherent in the data. For more information, see the Quantifying uncertainty in deep learning systems guide.

undifferentiated tasks

Also known as *heavy lifting*, work that is necessary to create and operate an application but that doesn't provide direct value to the end user or provide competitive advantage. Examples of undifferentiated tasks include procurement, maintenance, and capacity planning.

upper environments

See environment.

V

vacuuming

A database maintenance operation that involves cleaning up after incremental updates to reclaim storage and improve performance.

version control

Processes and tools that track changes, such as changes to source code in a repository.

VPC peering

A connection between two VPCs that allows you to route traffic by using private IP addresses. For more information, see What is VPC peering in the Amazon VPC documentation.

vulnerability

A software or hardware flaw that compromises the security of the system.

W

warm cache

A buffer cache that contains current, relevant data that is frequently accessed. The database instance can read from the buffer cache, which is faster than reading from the main memory or disk.

warm data

Data that is infrequently accessed. When querying this kind of data, moderately slow queries are typically acceptable.

window function

A SQL function that performs a calculation on a group of rows that relate in some way to the current record. Window functions are useful for processing tasks, such as calculating a moving average or accessing the value of rows based on the relative position of the current row.

workload

A collection of resources and code that delivers business value, such as a customer-facing application or backend process.

workstream

Functional groups in a migration project that are responsible for a specific set of tasks. Each
workstream is independent but supports the other workstreams in the project. For example,
the portfolio workstream is responsible for prioritizing applications, wave planning, and
collecting migration metadata. The portfolio workstream delivers these assets to the migration
workstream, which then migrates the servers and applications.

WORM

See write once, read many.

WQF

See AWS Workload Qualification Framework.

write once, read many (WORM)

A storage model that writes data a single time and prevents the data from being deleted or
modified. Authorized users can read the data as many times as needed, but they cannot change
it. This data storage infrastructure is considered immutable.

Z

zero-day exploit

An attack, typically malware, that takes advantage of a zero-day vulnerability.

zero-day vulnerability

An unmitigated flaw or vulnerability in a production system. Threat actors can use this type of
vulnerability to attack the system. Developers frequently become aware of the vulnerability as a
result of the attack.

zero-shot prompting

Providing an LLM with instructions for performing a task but no examples (*shots*) that can help
guide it. The LLM must use its pre-trained knowledge to handle the task. The effectiveness of
zero-shot prompting depends on the complexity of the task and the quality of the prompt. See
also few-shot prompting.

zombie application

An application that has an average CPU and memory usage below 5 percent. In a migration
project, it is common to retire these applications.