Database Migration

Mohanraj Shanmugam

Running Databases on AWS

- Two types of Database servers
- Amazon Relational Database Service (Amazon RDS)
  - It makes it simple to set up, operate, and scale a relational database in the cloud.
  - It provides cost-efficient, resizable capacity for an industry-standard relational database, and manages common database administration tasks.
- Amazon Elastic Compute Cloud (Amazon EC2)
  - It provides scalable computing capacity in the cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so that you can develop and deploy applications faster.
  - You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage.

Data Migration Strategies

- The migration strategy you choose depends on several factors:
  - The size of the database
  - Network connectivity between the source server and AWS
  - The version and edition of your Database software
  - The database options, tools, and utilities that are available
  - The amount of time that is available for migration
  - Whether the migration and switchover to AWS will be done in one step or a sequence of steps over time

One-Step Migration

- One-step migration is a good option for small databases that can be shut down for 24 to 72 hours.
- During the shut-down period, all the data from the source database is extracted, and the extracted data is migrated to the destination database in AWS.
- The destination database in AWS is tested and validated for data consistency with the source.
- Once all validations have passed, the database is switched over to AWS.

Two-Step Migration

- Two-step migration is a commonly used method because it requires only minimal downtime and can be used for databases of any size:
- Step 1:
  - The data is extracted from the source database at a point in time (preferably during non-peak usage) and migrated while the database is still up and running.
  - Because there is no downtime at this point, the migration window can be sufficiently large.
  - After you complete the data migration, you can validate the data in the destination database for consistency with the source and test the destination database on AWS for performance, for connectivity to the applications, and for any other criteria as needed.

Two-Step Migration

- Step 2:
- Data changed in the source database after the initial data migration is propagated to the destination before switch over.
- This step synchronizes the source and destination databases.
- This should be scheduled for a time when the database can be shut down (usually over a few hours late at night on a weekend).
- During this process, there won’t be any more changes to the source database because it will be unavailable to the applications.
- Normally, the amount of data that is changed after the first step is small compared to the total size of the database, so this step will be quick and thus requires only minimal downtime.
- Once all the changed data is migrated, you can validate the data in the destination database, perform necessary tests, and, if all tests are passed, switch over to the database in AWS.

Zero-Downtime Migration

- Some business situations require database migration with no downtime.
- This requires detailed planning and the necessary data replication tools for proper execution.
- Step 1:
- you need to extract and migrate all the data from the source database at a point in time without shutting down the database.
- After the data migration, you should validate the migrated data and conduct any necessary testing.

Zero-Downtime Migration

- The next step in the process is to set up continuous data replication from the source database to the destination database in AWS.
- You can use Oracle GoldenGate, Mysql Replication to replicate to Oracle or Mysql on Amazon RDS or to Oracle or Mysql Database on Amazon EC2.
- For Amazon EC2, you can use third-party tools such as Dbvisit Replicate or Attunity Replicate as an alternative to Oracle GoldenGate.
- The replication process synchronizes the destination database with the source database and continues to replicate all data changes at the source to the destination.
- Synchronous replication could have an effect on the performance of the source database, so if a few minutes of downtime for the database is acceptable, then we recommend that you set up asynchronous replication instead.
- You can switch over to the database in AWS at any time because the source and destination databases will always be in sync

Tools Used for Oracle Database Migration

- AWS Storage Gateway
- It is a service connecting an on-premises software appliance with cloud-based storage to provide seamless and secure integration between an organization’s on-premises IT environment and the AWS storage infrastructure.
- The service allows you to securely store data in the AWS cloud for scalable and costeffective storage.
- AWS Storage Gateway supports industry-standard storage protocols that work with your existing applications.
- It provides low-latency performance by maintaining frequently accessed data on-premises while securely storing all of your data encrypted in Amazon Simple Storage Service (Amazon S3) or Amazon Glacier.

Tools Used for Oracle Database Migration

- Oracle Recovery Manager (RMAN)
  - RMAN is a tool available from Oracle for performing and managing Oracle Database backups and restorations.
  - RMAN allows full hot or cold backups plus incremental backups.
  - RMAN maintains a catalogue of the backups, making the restoration process simple and dependable.
  - RMAN can also duplicate, or clone, a database from a backup or from an active database.

Tools Used for Oracle Database Migration

- The **mysqldump** client utility performs logical backups, producing a set of SQL statements that can be executed to reproduce the original database object definitions and table data.
- It dumps one or more MySQL databases for backup or transfer to another SQL server.
- The **mysqldump** command can also generate output in CSV, other delimited text, or XML format.

Tools Used for Oracle Database Migration

- Oracle Data Pump Export is a versatile utility for exporting and importing data and metadata from or to Oracle databases.
- You can perform Data Pump Export/Import on an entire database, selective schemas, table spaces, or database objects.
- Data Pump Export/Import also has powerful data-filtering capabilities for selective export or import of data.

Tools Used for Oracle Database Migration

- The **mysqlpump** client utility performs logical backups, producing a set of SQL statements that can be executed to reproduce the original database object definitions and table data. It dumps one or more MySQL databases for backup or transfer to another SQL server.
- **mysqlpump** features include:
  - Parallel processing of databases, and of objects within databases, to speed up the dump process
  - Better control over which databases and database objects (tables, stored programs, user accounts) to dump
  - Dumping of user accounts as account-management statements (CREATE USER, GRANT) rather than as inserts into the mysql system database
  - Capability of creating compressed output
  - Progress indicator (the values are estimates)
  - For dump file reloading, faster secondary index creation for InnoDB tables by adding indexes after rows are inserted

Tools Used for Oracle Database Migration

- The **mysqlpump** client utility performs logical backups, producing a set of SQL statements that can be executed to reproduce the original database object definitions and table data. It dumps one or more MySQL databases for backup or transfer to another SQL server.
- **mysqlpump** features include:
  - Parallel processing of databases, and of objects within databases, to speed up the dump process
  - Better control over which databases and database objects (tables, stored programs, user accounts) to dump
  - Dumping of user accounts as account-management statements (CREATE USER, GRANT) rather than as inserts into the mysql system database
  - Capability of creating compressed output
  - Progress indicator (the values are estimates)
  - For dump file reloading, faster secondary index creation for InnoDB tables by adding indexes after rows are inserted

Migration Documents

- Oracle
  - https://d0.awsstatic.com/whitepapers/strategies-for-migrating-oracle-database-to-aws.pdf
- Mysql or MariaDB
  - http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/MySQL.Procedural.Importing.html

AWS Database Migration Services

- AWS Database Migration Service (AWS DMS) can migrate your data to and from most widely used commercial and open-source databases such as Oracle, PostgreSQL, Microsoft SQL Server, Amazon Redshift, Amazon Aurora, MariaDB, and MySQL.
- The service supports homogeneous migrations such as Oracle to Oracle
- And also heterogeneous migrations between different database platforms, such as Oracle to MySQL or MySQL to Amazon Aurora.
- You can begin a database migration with just a few clicks in the AWS DMS Management Console.
- When the migration has started, AWS manages all the complexities of the migration process, such as data type transformation, compression, and parallel transfer for faster data transfer.
- At the same time, AWS ensures that data changes to the source database that occur during the migration process are automatically replicated to the target.

AWS DMS is currently available in the following regions:

Region	Name
US East (N. Virginia) region	us-east-1
US West (N. California) region	us-west-1
US West (Oregon) region	us-west-2
EU (Ireland) region	eu-west-1
EU (Frankfurt) Region	eu-central-1
Asia Pacific (Tokyo) Region	ap-northeast-1
Asia Pacific (Singapore) Region	ap-southeast-1
Asia Pacific (Sydney) Region	ap-southeast-2

Migration Planning for AWS Database Migration Service

- Prerequisite
- You will need to configure a network that connects your source and target databases to a AWS DMS replication instance.
- This can be as simple as connecting two AWS resources in the same VPC as the replication instance to more complex configurations such as connecting an on-premises database to an Amazon RDS DB instance over VPN.

Source and Target Endpoints

- You will need to know what information and tables in the source database need to be migrated to the target database.
- AWS DMS supports basic schema migration, including the creation of tables and primary keys.
- However, AWS DMS doesn't automatically create secondary indexes, foreign keys, user accounts, and so on in the target database.

Schema/Code Migration

- AWS DMS doesn't perform schema or code conversion.
- You can use tools such as Oracle SQL Developer, MySQL Workbench, or pgAdmin III to convert your schema.
- If you want to convert an existing schema to a different database engine, you can use the AWS Schema Conversion Tool.
- It can create a target schema and also can generate and create an entire schema: tables, indexes, views, and so on.
- You can also use the tool to convert PL/SQL or TSQL to PgSQL and other formats.

Sources for AWS Database Migration Service

- **On-premises and EC2 instance databases**
  - Oracle versions 10g, 11g, and 12c, for the Enterprise, Standard, Standard One, and Standard Two editions
  - Microsoft SQL Server versions 2005, 2008, 2008R2, 2012, and 2014, for the Enterprise, Standard, Workgroup, and Developer editions
  - MySQL versions 5.5, 5.6, and 5.7
  - MariaDB (supported as a MySQL-compatible data source)
  - PostgreSQL 9.4 and later
  - ODBC data source (based on ODBC 3.0)

Sources for AWS Database Migration Service

- **Amazon RDS instance databases**
  - Oracle versions 11g (versions 11.2.0.3.v1 and later) and 12c, for the Enterprise, Standard, Standard One, and Standard Two editions
  - Microsoft SQL Server versions 2008R2 and 2012, for the Enterprise, Standard, Workgroup, and Developer editions. Note that CDC operations are not supported.
  - MySQL versions 5.5, 5.6, and 5.7
  - MariaDB (supported as a MySQL-compatible data source)
  - PostgreSQL 9.4. Note that CDC operations are not supported.
  - Amazon Aurora (supported as a MySQL-compatible data source)

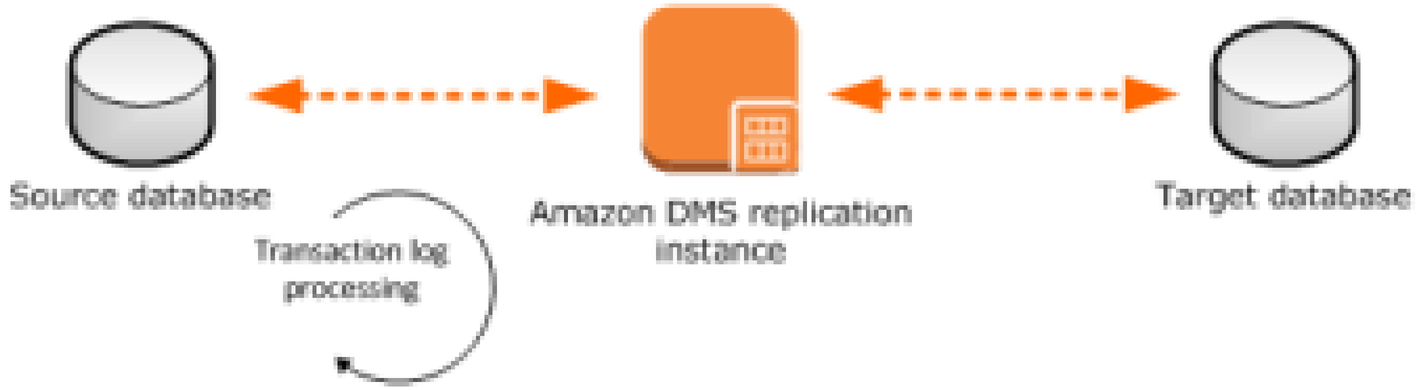
Targets for AWS Database Migration Service

- **On-premises and Amazon EC2 instance databases**
  - Oracle versions 10g, 11g, 12c, for the Enterprise, Standard, Standard One, and Standard Two editions
  - Microsoft SQL Server versions 2005, 2008, 2008R2, 2012, and 2014, for the Enterprise, Standard, Workgroup, and Developer editions
  - MySQL, versions 5.5, 5.6, and 5.7
  - MariaDB (supported as a MySQL-compatible data target)
  - PostgreSQL, versions 9.3 and later

Targets for AWS Database Migration Service

- **Amazon RDS instance databases and Amazon Redshift**
  - Oracle versions 11g (versions 11.2.0.3.v1 and later) and 12c, for the Enterprise, Standard, Standard One, and Standard Two editions
  - Microsoft SQL Server versions 2008R2 and 2012, for the Enterprise, Standard, Workgroup, and Developer editions
  - MySQL, versions 5.5, 5.6, and 5.7
  - MariaDB (supported as a MySQL-compatible data target)
  - PostgreSQL, versions 9.3 and later
  - Amazon Aurora (supported as a MySQL-compatible data target)
  - Amazon Redshift

Replication server



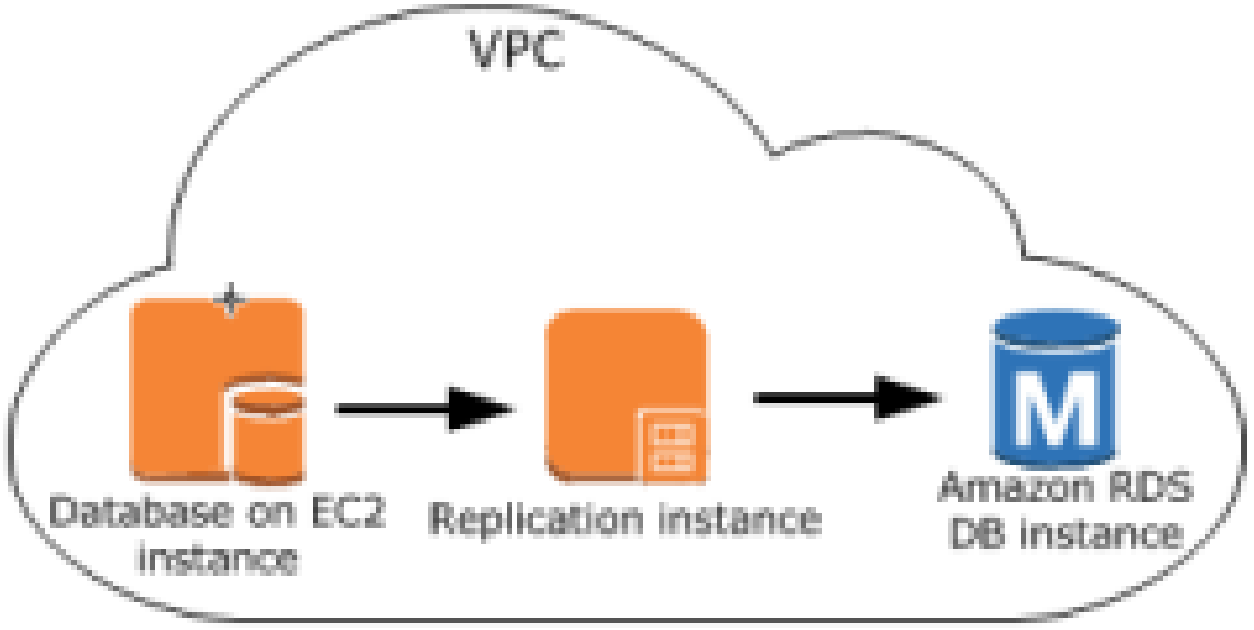
- AWS DMS runs on an Amazon Elastic Compute Cloud (Amazon EC2) instance.
- The service uses a replication server that connects to the source database, reads the source data, formats the data for consumption by the target database, and loads the data into the target database.
- Most of this processing happens in memory.
- However, large transactions might require some buffering on disk.
- Cached transactions and log files are also written to disk.

Replication server

- When creating your replication server, you should consider the following:
  - EC2 instance class — currently supports the T2 and C4 instance classes ,
  - Some of the smaller EC2 instance classes are sufficient for testing the service or for small migrations.
  - If your migration involves a large number of tables, or if you intend to run multiple concurrent replication tasks, you should consider using one of the larger instances because AWS DMS consumes a fair amount of memory and CPU.
- Storage
  - Depending on the EC2 instance class you select, your replication server comes with either 50 GB or 100 GB of data storage.
  - This storage is used for log files and any cached changes collected during the load. If your source system is busy or takes large transactions, or if you’re running multiple tasks on the replication server, you might need to increase this amount of storage. Usually the default amount is sufficient.
- Network
  - **Public and Private Replication Instances**
  - A private replication instance has a private IP address that cannot be accessed outside the replication network. A replication instance should have a private IP address when both the source and target databases are located in the same network that is connected to the replication instance's VPC by using a VPN

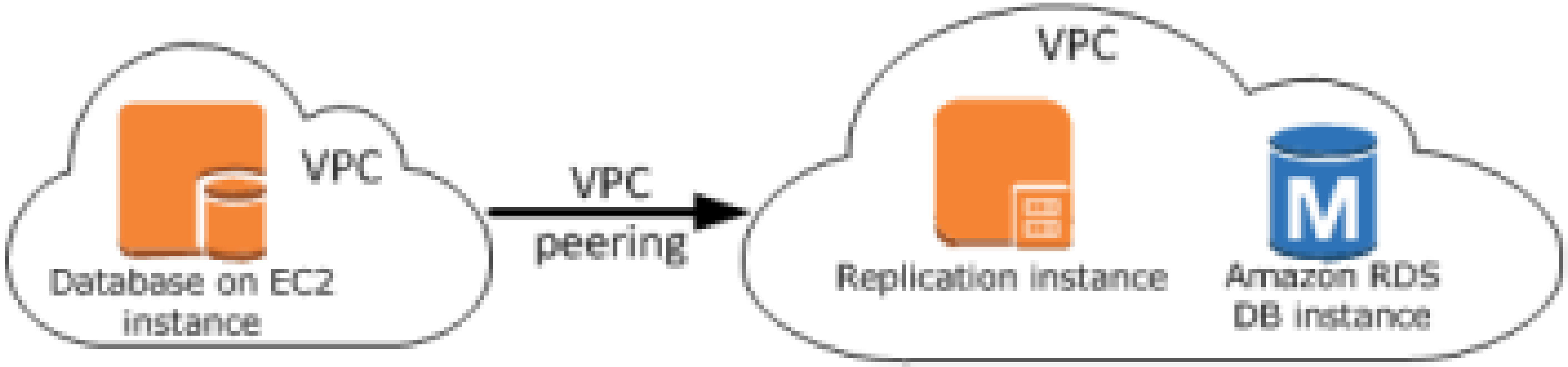
Networking for Replication Server

- **Configuration with All Database Migration Components in One VPC**

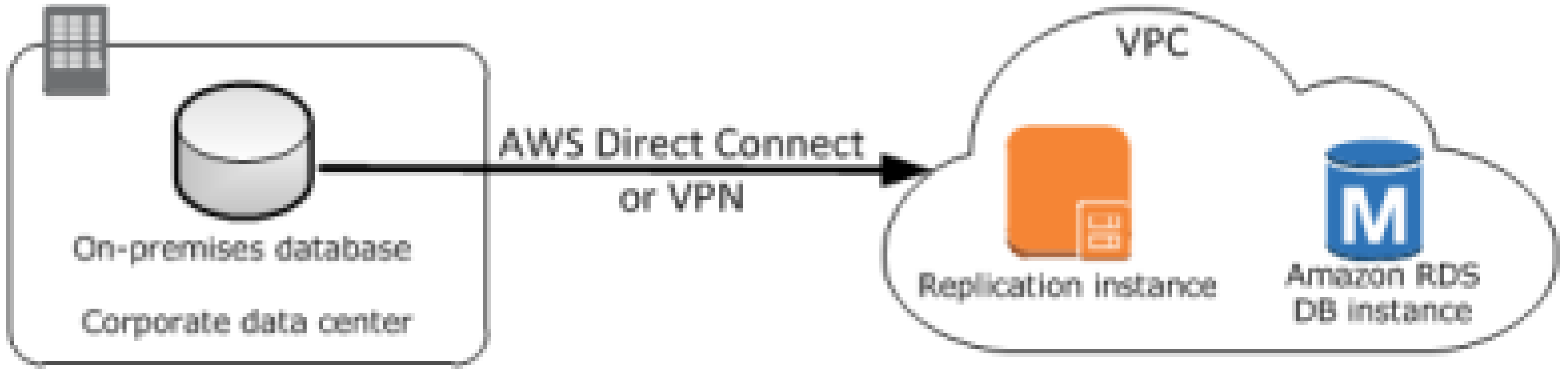


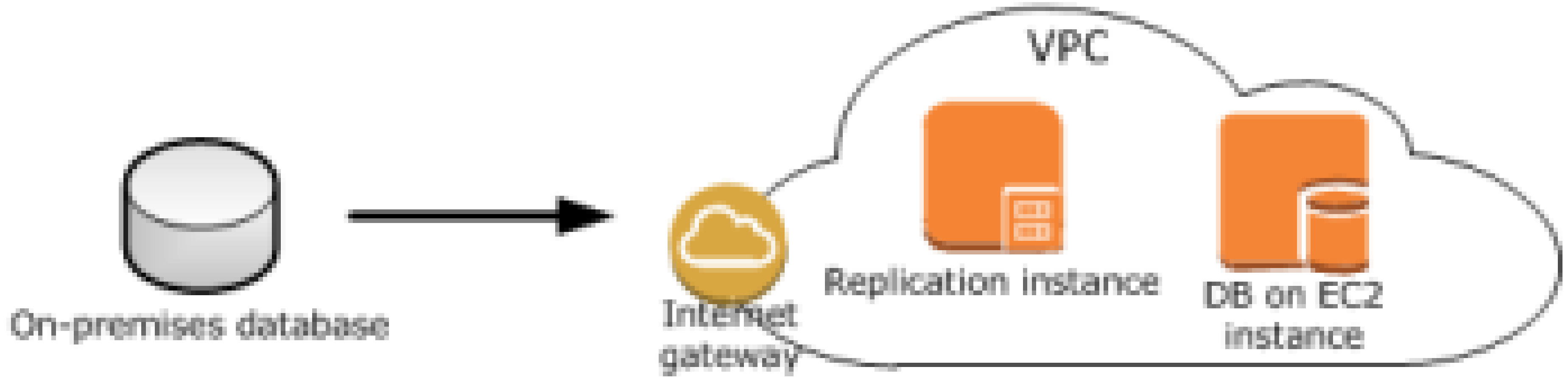
Configuration with Two VPCs

- **Configuration with All Database Migration Components in One VPC**



Configuration for a Network to a VPC Using AWS Direct Connect or a VPN





Source endpoint

- The change capture process that AWS DMS uses when replicating ongoing changes collects changes to the database logs by using the database engine's native API.
- Each engine has specific configuration requirements for exposing this change stream to a given user account.
- Most engines require some additional configuration to make the change data consumable in a meaningful way, without data loss, for the capture process.
- For example, Oracle requires the addition of supplemental logging and MySQL requires row-level bin logging.
- When using Amazon RDS as a source, we recommend ensuring that backups are enabled and that the source database is configured to retain change logs for a sufficient time (24 hours is usually enough).

Target endpoint

- Whenever possible, AWS DMS attempts to create the target schema for you.
- Sometimes, AWS DMS can't create the schema—for example, AWS DMS won't create a target Oracle schema for security reasons.
- For MySQL database targets, you can use extra connection parameters to have AWS DMS migrate all objects to the specified database and schema or create each database and schema for you as it finds the schema on the source.

Replication Tasks

- An AWS Database Migration Service task is where all the work happens. Here you specify what tables and schemas to use for your migration. A task must include the following:
  - The source and target databases
  - The schema and target tables
- You can monitor, stop, or restart replication activity using the AWS DMS console or API.

Migration Methods for AWS Database Migration Service

- AWS Database Migration Service can migrate your data in several ways:
  - **Migrating Data to the Target Database**
  - **Capturing Changes During Migration**
  - **Replicating Only Data Changes on the Source Database**

Migrating Data to the Target Database

- This process creates files or tables in the target database, automatically defines the metadata that is required at the target, and populates the tables with data from the source.
- The data from the tables is loaded in parallel for improved efficiency.
- This process is the **Migrate existing data** option in the AWS console and is called full load in the API.

Capturing Changes During Migration

- This process captures changes to the source database that occur while the data is being migrated from the source to the target.
- When the migration of the originally requested data has completed, the change data capture (CDC) process then applies the captured changes to the target database.
- Changes are captured and applied as units of single committed transactions, and several different target tables can be updated as a single source commit.
- This approach guarantees transactional integrity in the target database.
- This process is the **Migrate existing data and replicate ongoing changes** option in the AWS console and is called cdc in the API.

Replicating Only Data Changes on the Source Database

- This process reads the recovery log file of the source database management system (DBMS) and groups together the entries for each transaction.
- If AWS DMS can't apply changes to the target within a reasonable time (for example, if the target is not accessible), AWS DMS buffers the changes on the replication server for as long as necessary.
- It doesn't reread the source DBMS logs, which can take a large amount of time.
- This process is the **Replicate data changes only** option in the AWS DMS console.