

# AWS Database Migration

# Agenda

- Upon completing this module, you should be able to:
  - Understand AWS DB migration Challenges
  - Understand a brief Introduction to AWS DB Migration services
  - Understand AWS DB services
  - Typical use case for AWS DB Migration
  - Understand AWS DB migration services in detail
  - Understand pre, during and post DB migration task
  - Perform lab for heterogeneous DB migration as per lab guide

# DB Migration challenges

- ▶ Identifying and prioritizing DB Migration
- ▶ Large Databases (TBs ,PBs)
- ▶ Live Migration with Zero Data Loss
- ▶ Schema Migration
- ▶ Secure DB migration

# AWS Database Migration Service (DMS) – Overview

---

- AWS Database Migration Service is a web service used to migrate data from On-premises databases, Amazon Relational Database Service (Amazon RDS) DB instance or database running on an Amazon Elastic Compute Cloud (Amazon EC2) instance.
- Use the AWS Schema Conversion Tool (AWS SCT) to translate your database schema to the new platform and then use AWS DMS to migrate the data.
- AWS DMS & AWS Snowball Edge Integration enables mass Database Migrations and Migrations of large databases.
- AWS Migration Hub helps to keep track of the progress of DB Migrations and also provides key metrics .

# How is AWS DMS different ?

- AWS DMS helps to migrate databases to AWS **quickly and securely**.
- **Live Migration** : The source database remains fully operational during the migration, **minimizing downtime** to applications that rely on the source database
- DMS can migrate data to and from most widely used **commercial and open-source databases**.
- The service supports **homogenous migrations** such as Oracle to Oracle, as well as **heterogeneous DB migrations** between different database platforms, such as Oracle to Amazon Aurora or Microsoft SQL Server to MySQL
- AWS DMS can also be used for Database Consolidation , Continuous data replication with high-availability

# AWS DMS Overview



AWS  
Database Migration  
Service



ORACLE

Amazon Aurora



Start your first migration in 10 minutes or less

Keep your apps running during the migration

Replicate within, to or from Amazon EC2 or RDS

Move data to the same or different database engine

Learn more at [aws.amazon.com/dms](https://aws.amazon.com/dms)

# Introduction to AWS DB services

---

- Main Services & Tools

- AWS RDS
- AWS Redshift
- AWS Dynamo DB – A non RDS DB
- AWS Migration Hub
- AWS Schema Migration Services

- ▶ Support Services

- AWS Cloud Formation services



# AWS Services related to AWS DMS





# AWS RDS Overview

01

Amazon Relational Database Service (RDS AWS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud.

02

It provides cost-efficient, re-sizable capacity in an industry-standard relational database and manages common database administration tasks such as database setup , patching and backups.

03

Provides better performance , High Availability , Security and Compatibility for databases.

04

# AWS RDS Database Engines

- Amazon RDS is available on several database instance types - optimized for **memory, performance or I/O** - and provides with following six familiar database engines to choose from

**Amazon Aurora**  
Provides 5x faster than RDS MySQL and 2x faster than RDS PostgreSQL

**MySQL**  
Open Source Database Management System which uses SQL to access the data

**PostgreSQL**

Open Source Database Management System which uses SQL to access the data stored in its system.

**Microsoft SQL Server**

Relational Database Management System, which was developed by Microsoft in 2005 for the enterprise environment

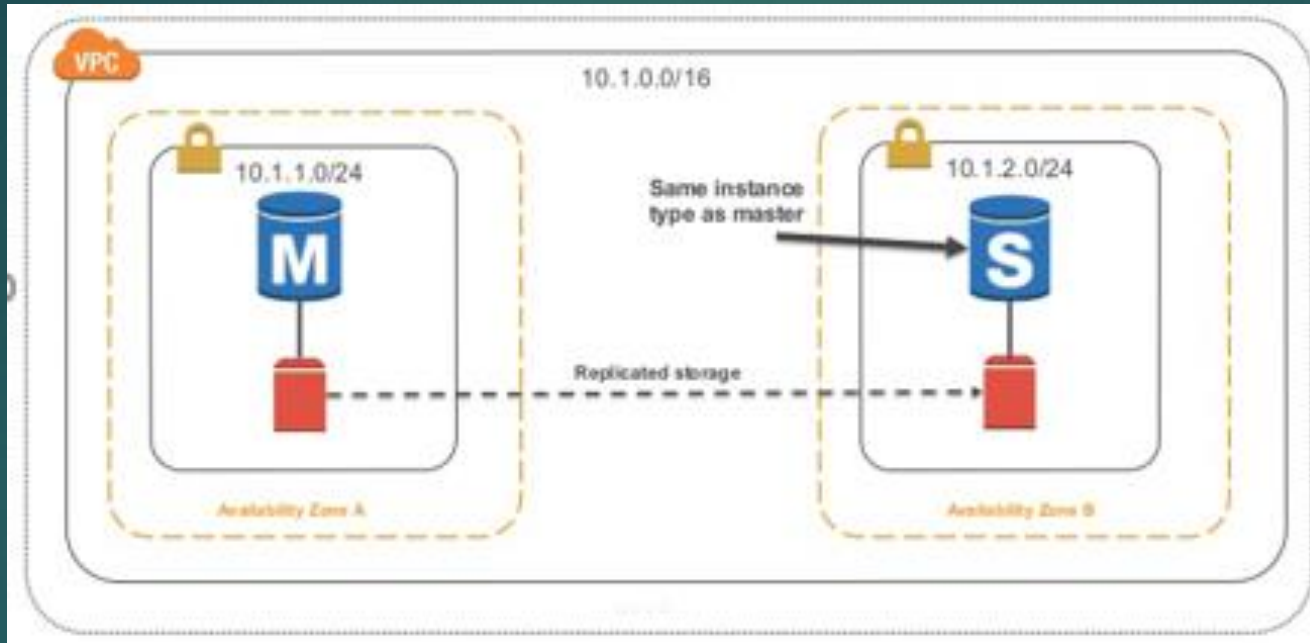
**Oracle**

Object-relational database management system which was developed by Oracle Inc

**Maria DB**

MariaDB is a community developed fork of MySQL DBMS

# AWS RDS in HA



- Amazon RDS Multi-AZ deployments provide enhanced availability and durability for Database (DB) Instances.
- When you provision a Multi-AZ DB Instance, Amazon RDS automatically creates a primary DB Instance and synchronously replicates the data to a standby instance in a different Availability Zone (AZ).
- Each AZ runs on its own physically distinct, independent infrastructure.
- Amazon RDS performs an automatic failover to the standby, in case of failure.

# AWS Dynamo DB Overview

01

Dynamo DB is a NoSQL database managed service provided by AWS

02

High-Performance databases which utilize a variety of data models, including document, graph, key-value, and columnar

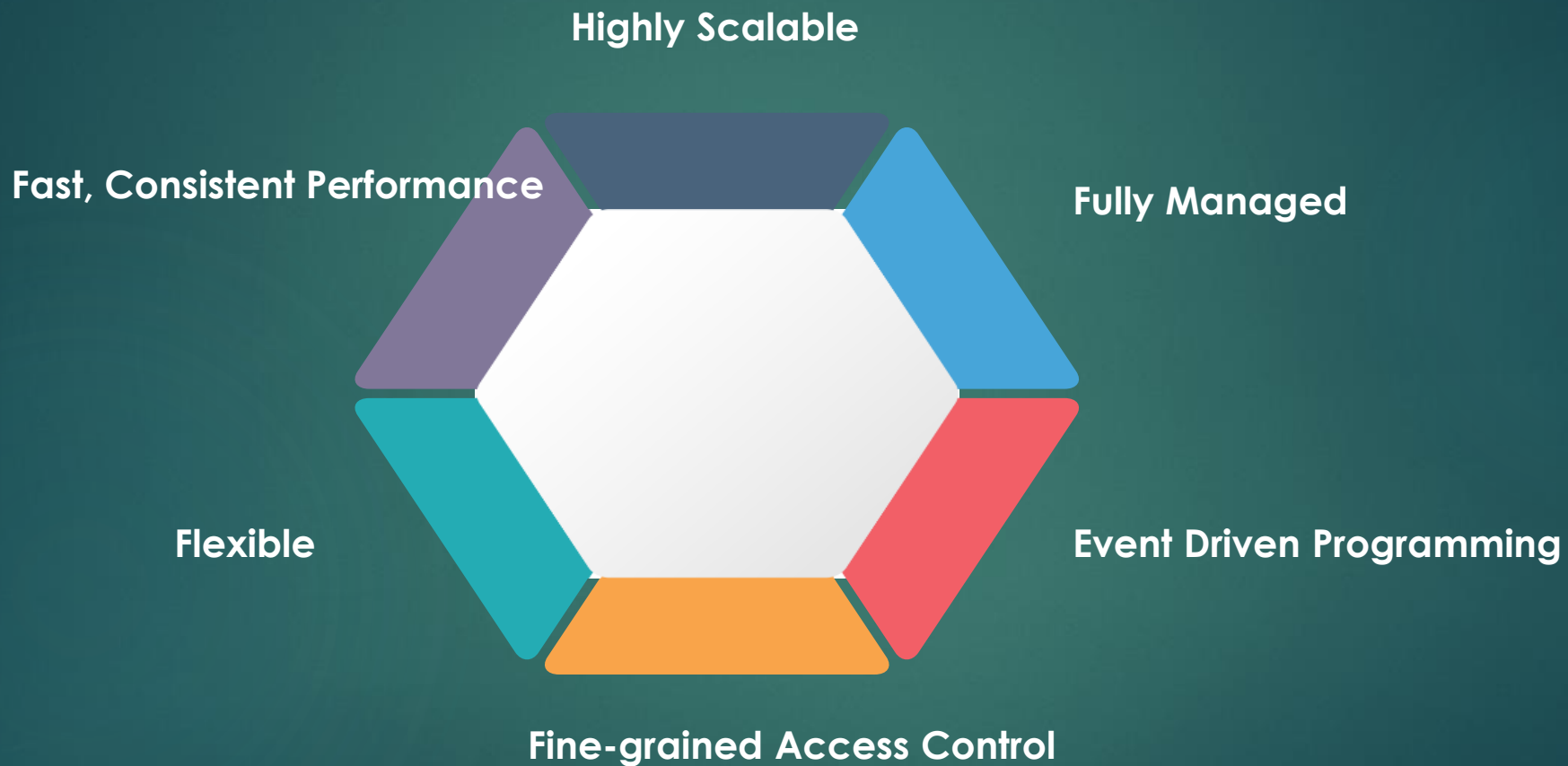
03

Recognized for ease of development, highly scalable performance, high availability, and resilience

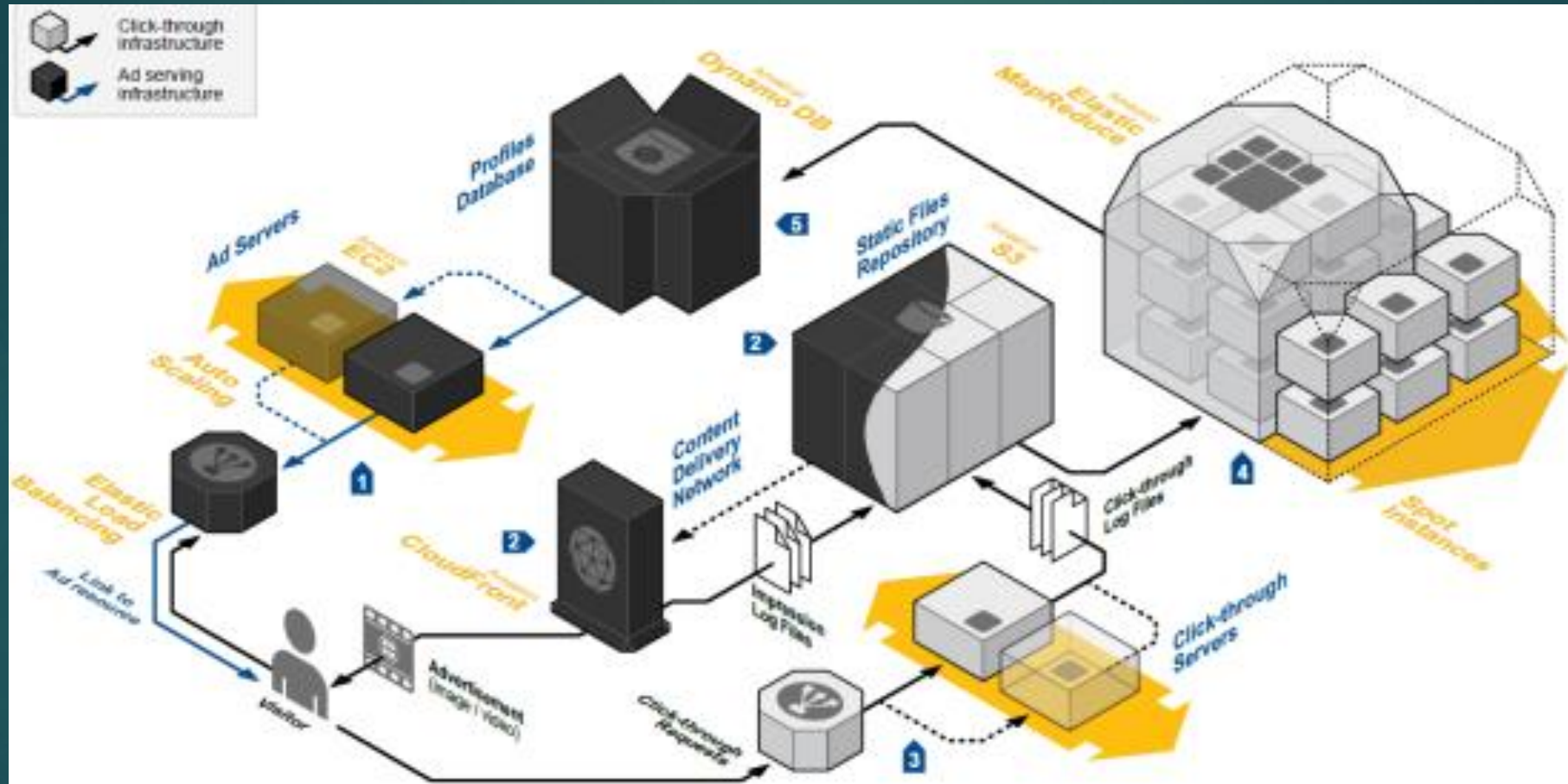
04

Database service for all applications that need consistent, single-digit millisecond latency at any scale . Eg: Mobile, Web, gaming, ad tech, IoT, and many other applications

# AWS DynamoDB Benefits

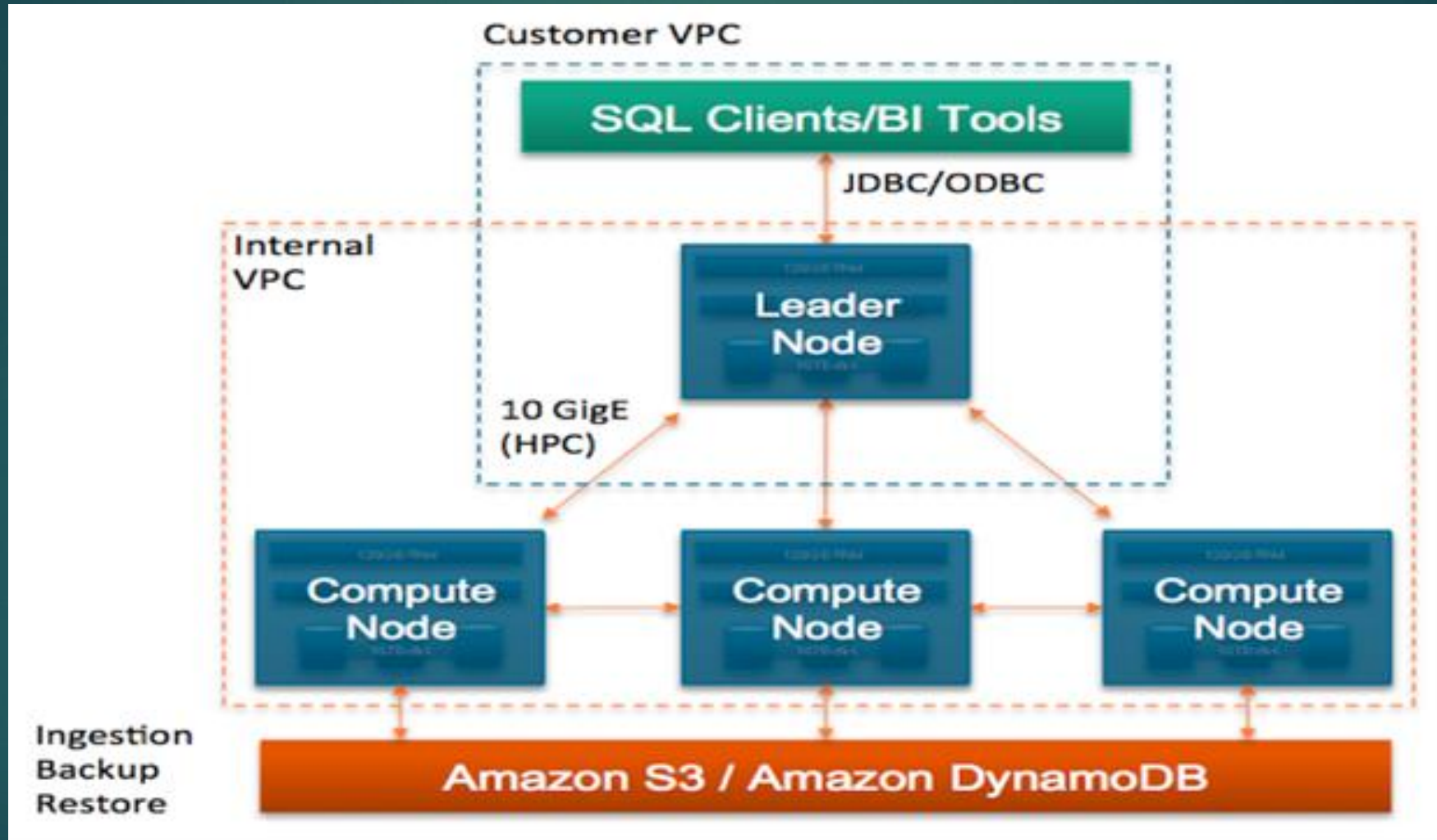


# Use Case - Ad Tech



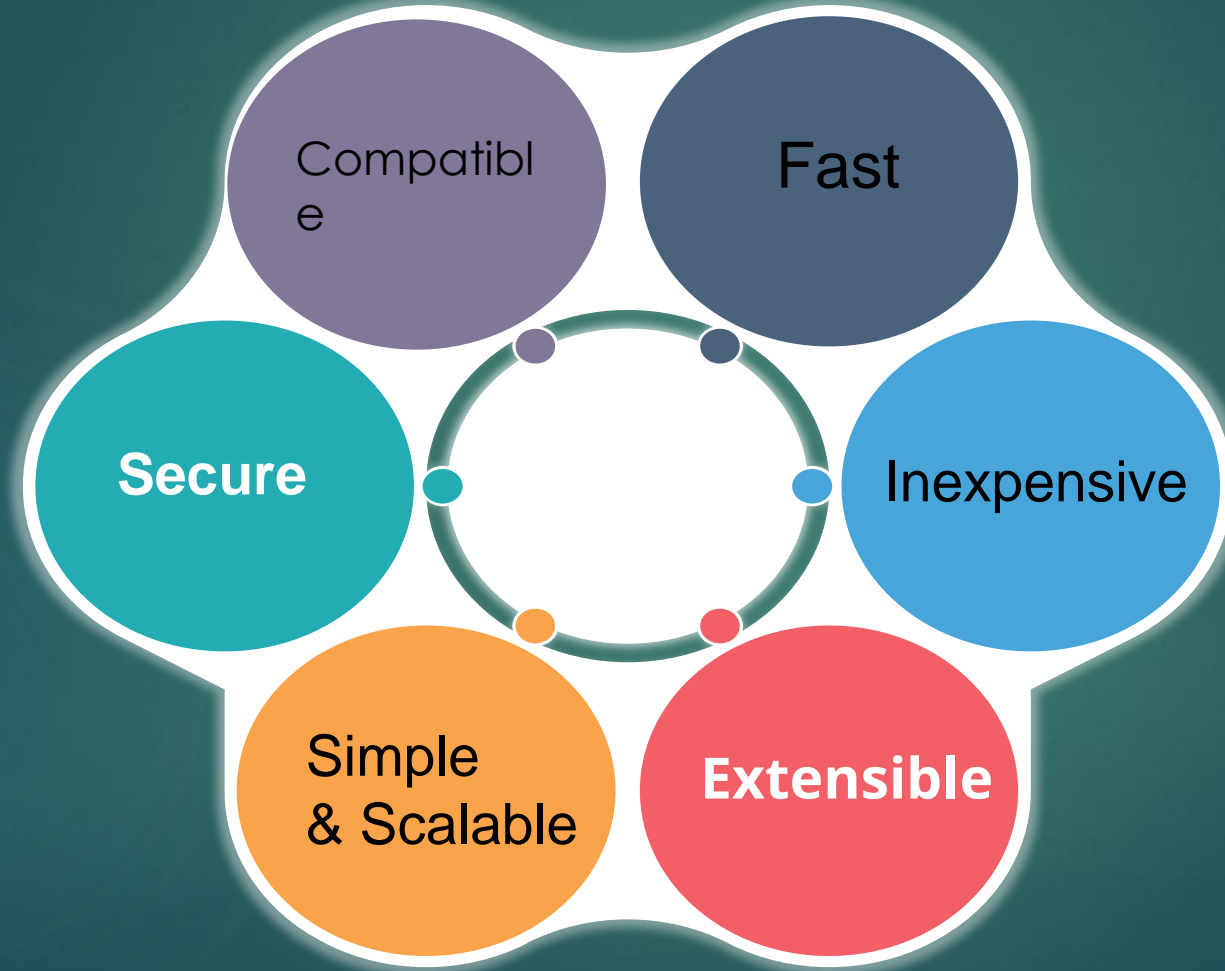


# AWS Redshift Architecture





# AWS Redshift Benefits



# AWS CloudFormation - IaasC

01 AWS CloudFormation helps customers to describe and provision all the infrastructure resources in cloud

02 Comprehensive templating language that enables you to create managed 'stacks' of AWS resources

03 AWS provides large number of ready cloud formation stack template which can be used directly or customized for creating cloud infrastructure

04 AWS CloudFormation is available at no additional charge, and you pay only for the AWS resources needed to run your applications.

05

# How CloudFormation Works



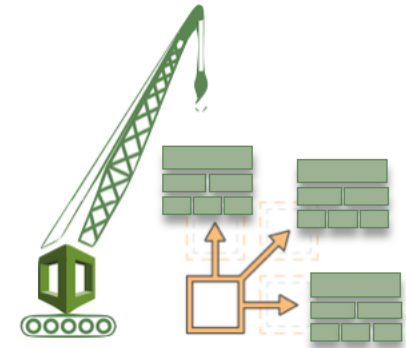
*Code your infrastructure from scratch with the CloudFormation template language, in either YAML or JSON format, or start from many available sample templates*



*Check out your template code locally, or upload it into an S3 bucket*



*Use AWS CloudFormation via the browser console, command line tools or APIs to create a stack based on your template code*



*AWS CloudFormation provisions and configures the stacks and resources you specified on your template*

# AWS Snowball Edge

01

It is a 100TB data transfer device with on-board storage and compute capabilities.

02

Used to move large amounts of data into and out of AWS and to support local workloads in remote or offline locations.

03

Connects to your existing applications and infrastructure using standard storage interfaces, streamlining the data transfer process and minimizing setup and integration.

04

Used to cluster together to form a local storage tier and process your data on-premises, helping ensure your applications continue to run even when they are not able to access the cloud.

# AWS Snowball Process



## CREATE A JOB

Create a new data transfer job in the [AWS Management Console](#). AWS will ship you one or more Snowball appliances based on the amount of data.



## CONNECT THE SNOWBALL

Connect the appliance to your network and set the [IP address](#). Download the Snowball client and job [manifest](#) from the Console, run the client to connect and identify data to transfer.



## COPY TO THE SNOWBALL

The client will encrypt and copy data to the appliance at high speed. Once complete, the E ink shipping label will automatically update.



## AWS WILL MOVE YOUR DATA TO S3

Track the job status via Amazon SNS, text messaging, or directly in the Console.



# Overview of AWS Schema Conversion Tool

## (SCT)

- 1 Used for Heterogenous Database Migrations
- 2 **Automatically converts** the source DB schema and a majority of the source DB code objects, including views, stored procedures, and functions, to a format compatible with the target database.
- 3 Marks objects which cannot be converted so that they could be manually migrated
- 4 It can also scan application source code for embedded SQL statements and convert them as part of DB Schema conversion project.
- 5 Source database can be on-premises, or in Amazon RDS or EC2 and the target database can be in either Amazon RDS or EC2

# DB Conversions supported by SCT

Source Database	Target Database on Amazon RDS
Oracle Database	Amazon Aurora, MySQL, PostgreSQL, MariaDB
Oracle Data Warehouse	Amazon Redshift
Microsoft SQL Server	Amazon Aurora, Amazon Redshift, MySQL, PostgreSQL, MariaDB
Teradata	Amazon Redshift
IBM Netezza	Amazon Redshift
Greenplum	Amazon Redshift
HPE Vertica	Amazon Redshift
MySQL and MariaDB	PostgreSQL
PostgreSQL	Amazon Aurora, MySQL, MariaDB
Amazon Aurora	PostgreSQL
IBM DB2 LUW	Amazon Aurora, MySQL, PostgreSQL



# AWS SCT Project Window

**1**

**2**

**3**

**4**

**5**

AWS Schema Conversion Tool Project 5 - AWS Schema Conversion Tool

File Actions View Settings Help

Summary Action Items

Microsoft SQL Server

- Databases (2)
- Schemas (8)
- Tables (12)

Address

- AddressType
- BusinessEntity
- BusinessEntityAddress
- BusinessEntityContact
- ContactType
- CountryRegion
- EmailAddress
- Password
- Person
- PersonPhone
- PhoneNumberType
- StateProvince

Views

- Procedures
- SQL scalar functions
- SQL table-valued functions
- SQL inline functions
- Synonyms
- Sequences
- Types
- Table types

Issue 477: MySQL doesn't support using a function or expressions as a default value. It is emulated by trigger

Recommended action: Please review generated code and modify it if necessary.

No. of occurrences: 22

Issue 678: MySQL does not support check constraints. Emulating triggers created

Recommended action: Please revise generated code and modify it if necessary.

No. of occurrences: 2

Issue 697: MySQL doesn't support an Identity column outside a primary key

Recommended action: Perform a manual conversion.

No. of occurrences: 7

Issue 700: MySQL doesn't support GEOGRAPHY type

Recommended action: To store data of this type in MySQL, use a MySQL-compatible type or use a composite type.

No. of occurrences: 1 | Documentation reference: <https://dev.mysql.com/doc/refman/5.6/en/data-types.html>

Issue 700: MySQL doesn't support UNIQUEIDENTIFIER type

Recommended action: To store data of this type in MySQL, use a MySQL-compatible type or use a composite type.

SQL Server table: Address

Property	Value
Created or last modified	
Created	2016-01-05 23:29:41.09
Last modified	2016-01-05 23:46:59.76
Object name	
Name	Address
object-id	681577509
schema-id	20
type	U
type-desc	USER_TABLE
Table extended properties	
is-memory-optimized	NO

MySQL table: Address

```
01 CREATE TABLE IF NOT EXISTS SampleDatabase_Customers
02 AddressID INT NOT NULL,
03 AddressLine1 VARCHAR(60) NOT NULL,
04 AddressLine2 VARCHAR(60) DEFAULT NULL,
05 City VARCHAR(30) NOT NULL,
06 StateProvinceID INT NOT NULL,
07 PostalCode VARCHAR(15) NOT NULL,
08 SpatialLocation VARCHAR(5000) DEFAULT NULL,
09 rowguid CHAR(38) DEFAULT NULL,
10 ModifiedDate DATETIME DEFAULT NULL,
11 ] ENGINE=InnoDB
```

Amazon RDS for MySQL

- Schema (17)
- Tables (33)

SampleDatabase\_Customers

- Address
- AddressType
- BusinessEntity
- BusinessEntityAddress
- BusinessEntityContact
- ContactType
- CountryRegion
- EmailAddress
- Password
- Person
- PersonPhone
- PhoneNumberType
- StateProvince

Used memory: 120.67 MB, Free memory: 38.93 MB, Total memory: 160.5 MB, Maximum memory: 804 MB

# AWS Schema Conversion Tool (Oracle to Aurora2)

dbmaster@ec2-107-23-147-250.cor

Schemas [47]

AAT

Tables [103]

BOOTSTRAP

T1

T100

T13

T14

T15

T16

T17

T18

T19

T2

T20

T21

T22

T23

T24

T25

T26

T27

T28

T29

T3

T30

ID	Status	Source	Target	Type	Complete %	Elapsed T...	Tables lo...	Tables q...	Tables er...
sct-booke...	load com...	pgkniewe...	sct-generi...	full-load	100%	0m	0	0	0
sct-oracle...	running	oraclemai...	sct-lambd...	full-load	79%	1h 18m	0	0	0
sct-sampl...	ready	oraclemai...	sct-generi...	full-load	0%				
select-an...	failed	oraclemai...	mysql-tar...	full-load...	0%	0m	0	1	0
ssl-aurora...	stopped	aurora-so...	mysql-tar...	full-load	66%	15m	2	1	0
ssl-aurora...	failed	aurora-so...	mysql-tar...	cdc	100%	0m	1	0	0
ssl-mysql...	failed	mysql-so...	mysql-tar...	cdc	100%	0m	1	0	0
ssl-test-s...	failed	aurora-so...	oraclemai...	full-load...	0%	0m	0	4	0
unsupport...	load com...	russomys...	mysql-tar...	full-load...	100%	0m	1	0	0
update-te...	load com...	oracle-so...	ora-targe...	full-load...	100%	0m	1	0	0

Start Stop Resume Delete Refresh Show Log

Overview Table statistics

Task name sct-sample-task-1

Task ARN arn:aws:dms:us-east-1:343299325021:task:FY6JYHULYFQGI7B5BUEBARWXJM

Status creating

Migration type full-load

Replication instance replication-instance-knievel-4

Source endpoint oraclemailtest

Target endpoint sct-generic-target-02

```
{
  "rules": [
    {
      "rule-type": "selection",

```

dbmaster@sct-generic-target-02.clust

Schemas [12]

AASC\_ERP

awsdms\_control

aws\_oracle\_ext

aws\_sqlserver\_ext

information\_schema

mysql

performance\_schema

SCT\_TEST

TEST

test\_dbo

tmp

AAT

Tables [1]

T1

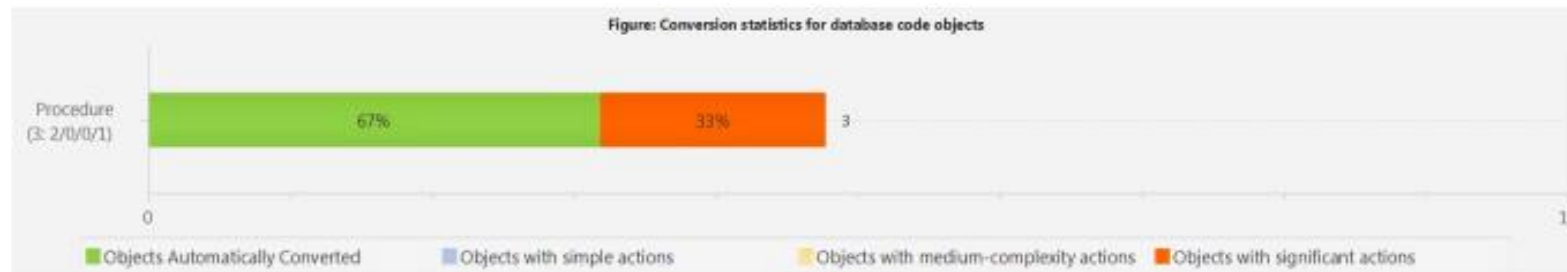
Used memory: 5.33 GB, Free memory: 5.54 GB, Total memory: 10.87 GB, Maximum memory: 42.5 GB

# Sample SCT Service Report

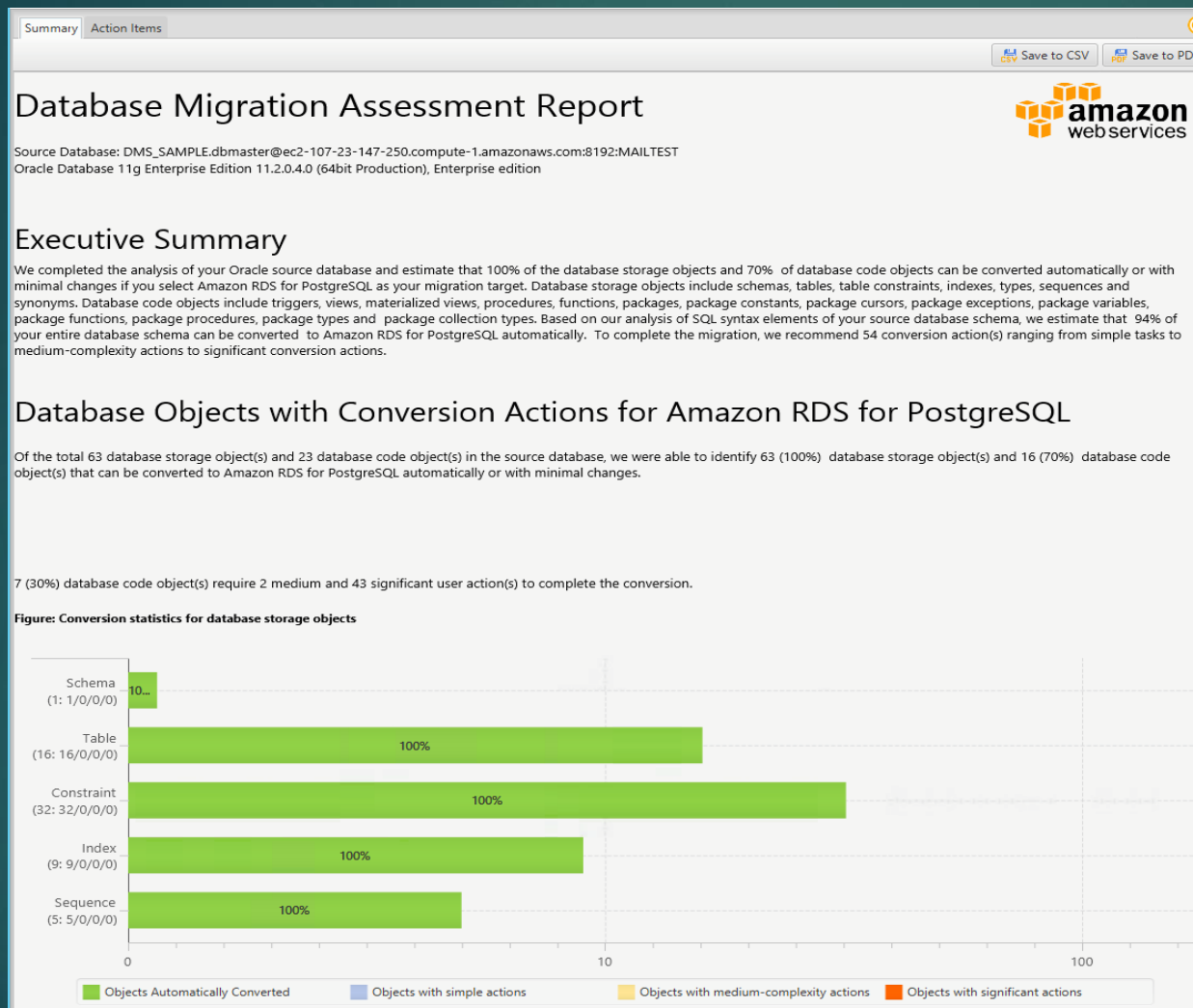
## Database Objects with Conversion Actions for Amazon RDS for PostgreSQL

Of the total 8 database storage object(s) and 3 database code object(s) in the source database, we were able to identify 8 (100%) database storage object(s) and 2 (67%) database code object(s) that can be converted to Amazon RDS for PostgreSQL automatically or with minimal changes.

1 (33%) database code object(s) require 1 medium and 4 significant user action(s) to complete the conversion.



# Sample SCT Service Report



# AWS Migration Hub - Overview

01

This AWS Migration Hub console page shows the list of migrations.

02

This page shows the details of a migration.

03

Migration Hub makes it easier for users to navigate the

04

Quickly get progress updates across all of your migrations

05



# AWS Migration Hub (Main Screen)

The screenshot displays the AWS Migration Hub main screen. At the top, there's a navigation bar with 'AWS', 'Services', and 'Edit' menus, along with region and support dropdowns. The left sidebar shows the 'Migration Hub' section with a 'Dashboard' link and sub-sections for 'Discover' (Data collection, Servers, Applications) and 'Migrate' (Applications, Updates, Tools). The main content area features the 'AWS Migration Hub' title and a description: 'Migration Hub simplifies and accelerates discovery and migration from your data centers to the AWS Cloud. [Learn more](#)'. Below this is a three-step workflow: 1. 'Discover' (orange box) with a magnifying glass icon and the step 'Deploy AWS discovery tools (Optional)'. 2. 'Migrate' (green box) with a cloud upload icon and steps 'Connect migration tools', 'Migrate using connected tools', and 'Group servers as applications'. 3. 'Track' (blue box) with a bar chart icon and the step 'Track status of migrations', including a [View example](#) link. Arrows connect the steps. Below the workflow is a section titled 'What would you like to do?' with two buttons: 'Get started with discovery' and 'Get started migrating', separated by 'OR'. The bottom of the page lists four categories of resources: 'Integrated discovery tools' (AWS Discovery Connectors, AWS Discovery Agents), 'Integrated migration tools' (AWS Database Migration Service, AWS Server Migration Service, CloudEndure Live Migration, Racemi DynaCenter), 'AWS migration programs' (Professional Services, Migration Acceleration Program, Migration Partners Solutions), and 'Documentation & support' (User Guide, Forums, Contact us). The footer contains a 'Feedback' button, 'English' language selector, copyright notice '© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.', and links to 'Privacy Policy' and 'Terms of Use'.

**Migration Hub**

**Dashboard**

Discover

- Data collection
- Servers
- Applications

Migrate

- Applications
- Updates
- Tools

## AWS Migration Hub

Migration Hub simplifies and accelerates discovery and migration from your data centers to the AWS Cloud. [Learn more](#)

Discover

- Deploy AWS discovery tools (Optional)

→

Migrate

- Connect migration tools
- Migrate using connected tools
- Group servers as applications

→

Track

- Track status of migrations [View example](#)

What would you like to do?

[Get started with discovery](#) OR [Get started migrating](#)

**Integrated discovery tools:**

- AWS Discovery Connectors
- AWS Discovery Agents

**Integrated migration tools:**

- AWS Database Migration Service
- AWS Server Migration Service
- CloudEndure Live Migration
- Racemi DynaCenter

**AWS migration programs:**

- Professional Services
- Migration Acceleration Program
- Migration Partners Solutions

**Documentation & support:**

- User Guide
- Forums
- Contact us

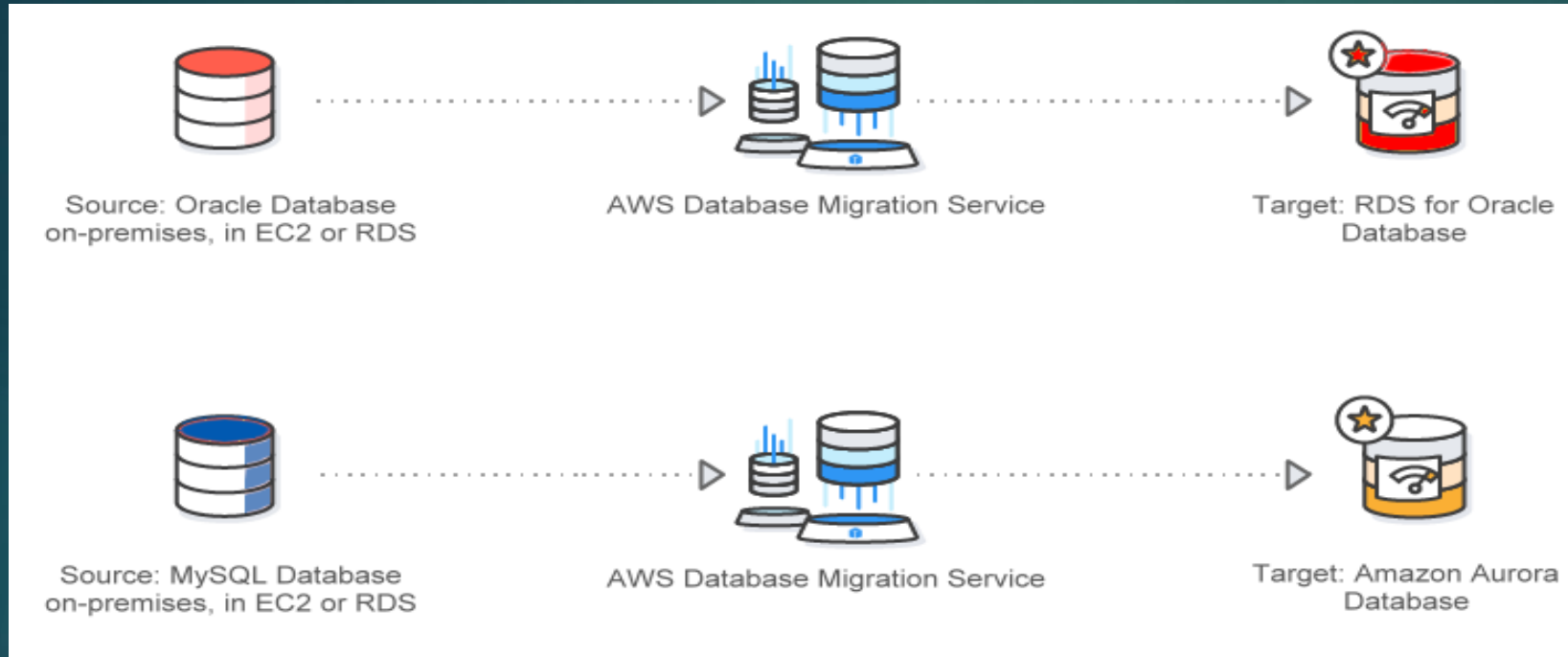
[Feedback](#) [English](#) © 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

# DB Migration Use Cases – From Source to Target





# DB Migration Use Cases – Homogeneous

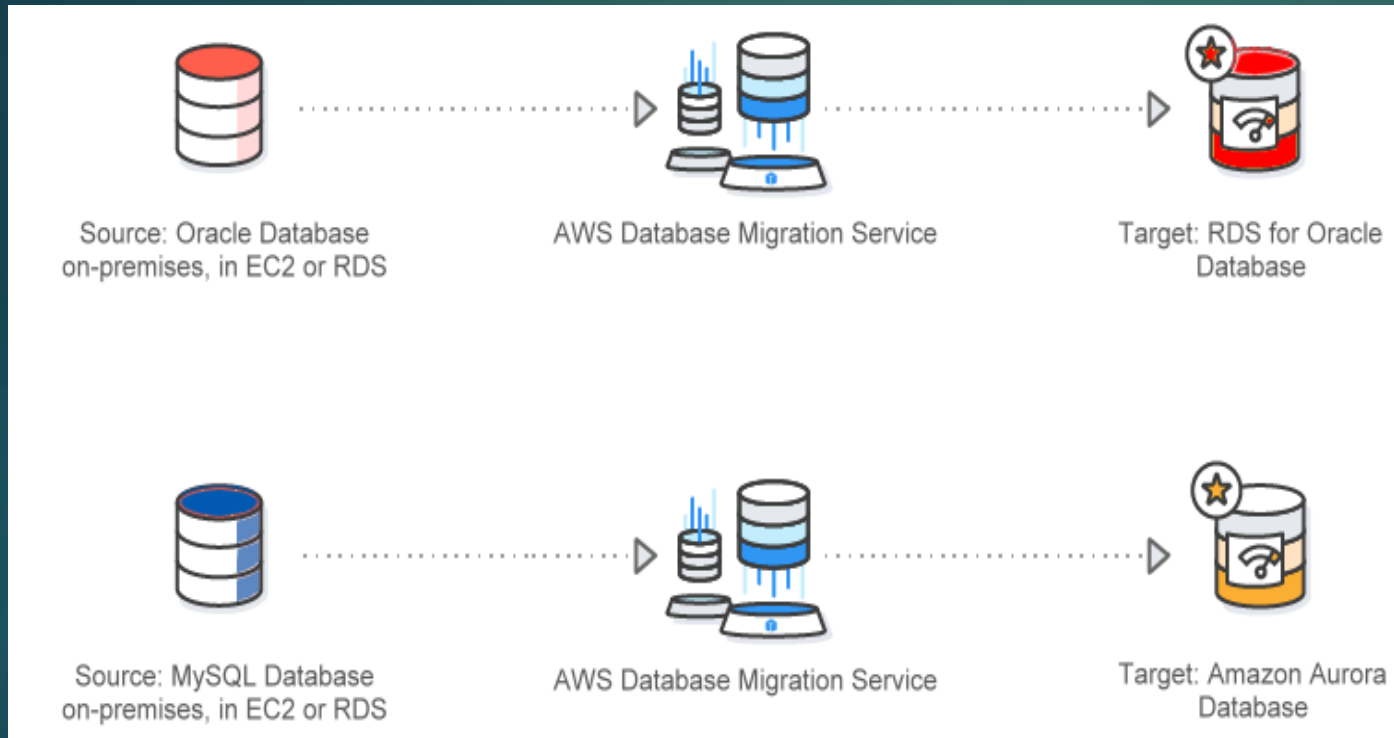


DB engine on both side are same

Amazon Aurora is a new developed DB engine which is fully managed, **MySQL-compatible**, relational database engine

Sources : <https://aws.amazon.com/dms/>

# DB Migration Use Cases – Heterogeneous



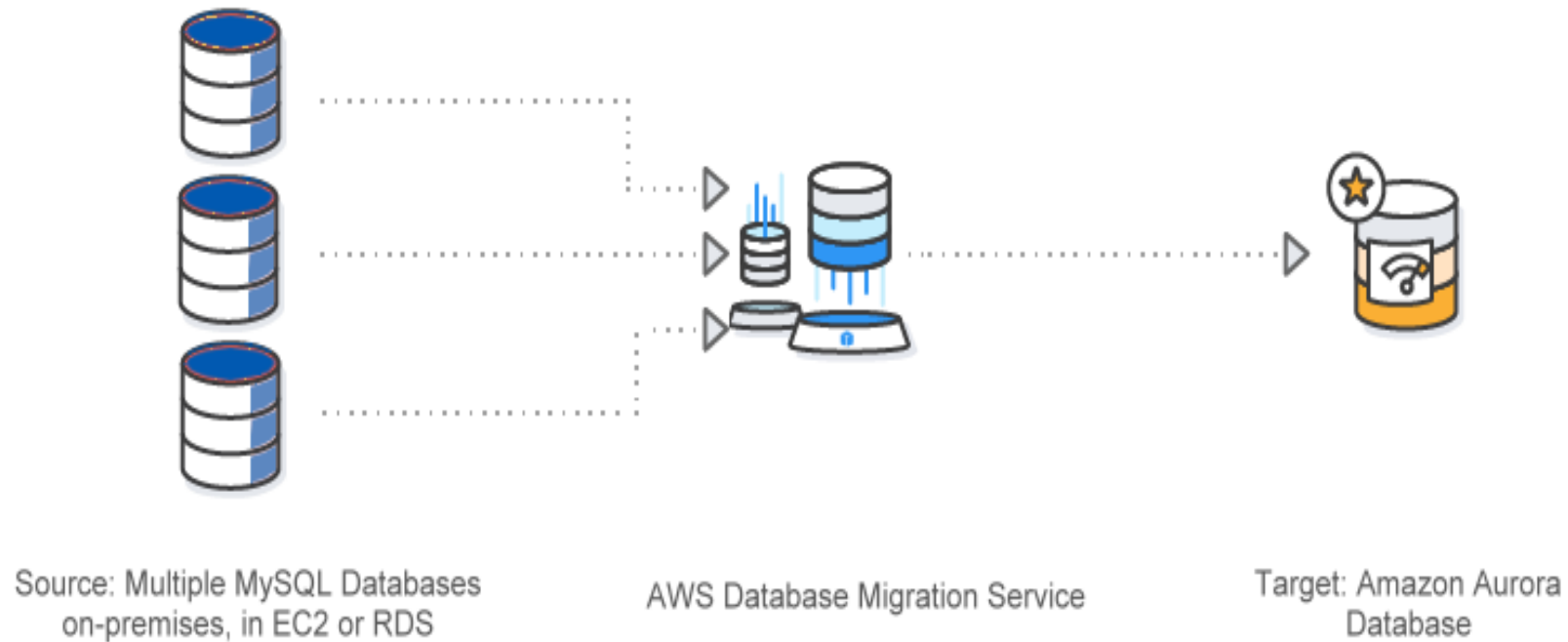
**Homogenous DB migration is a two Step Process**

**Step1:** Schema and code transformation with AWS Schema conversion(SCT) tools

**Step2:** DB migration via AWS DB Migration services

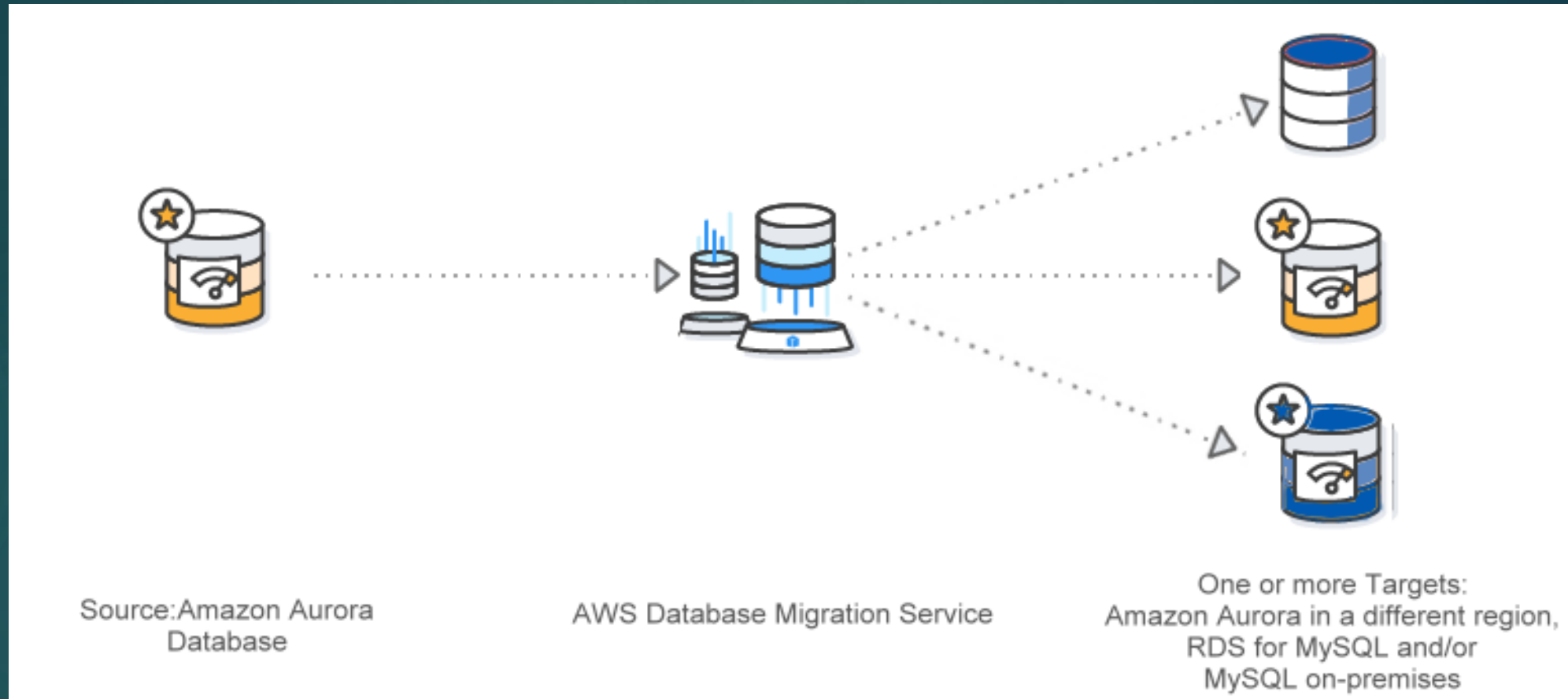
Sources : <https://aws.amazon.com/dms/>

# DB Migration Use Cases – DB Consolidation



<https://aws.amazon.com/dms/>

# DB Migration Use Cases – Continuous Data Replication



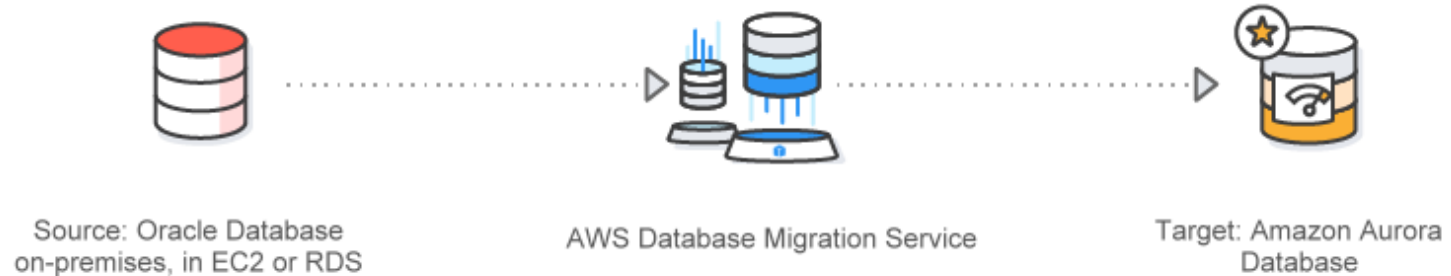
Sources : <https://aws.amazon.com/dms/>

# DMS & SCT Working together

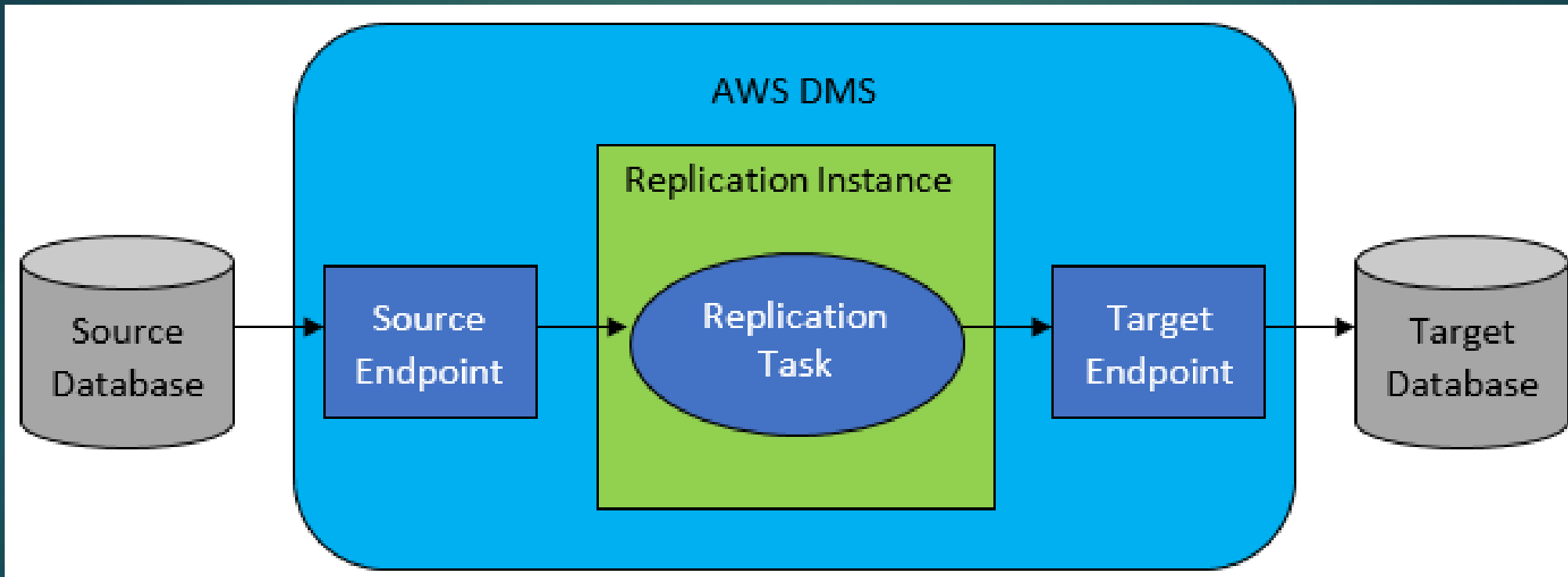
STEP 1:



STEP 2:



# How DMS Works

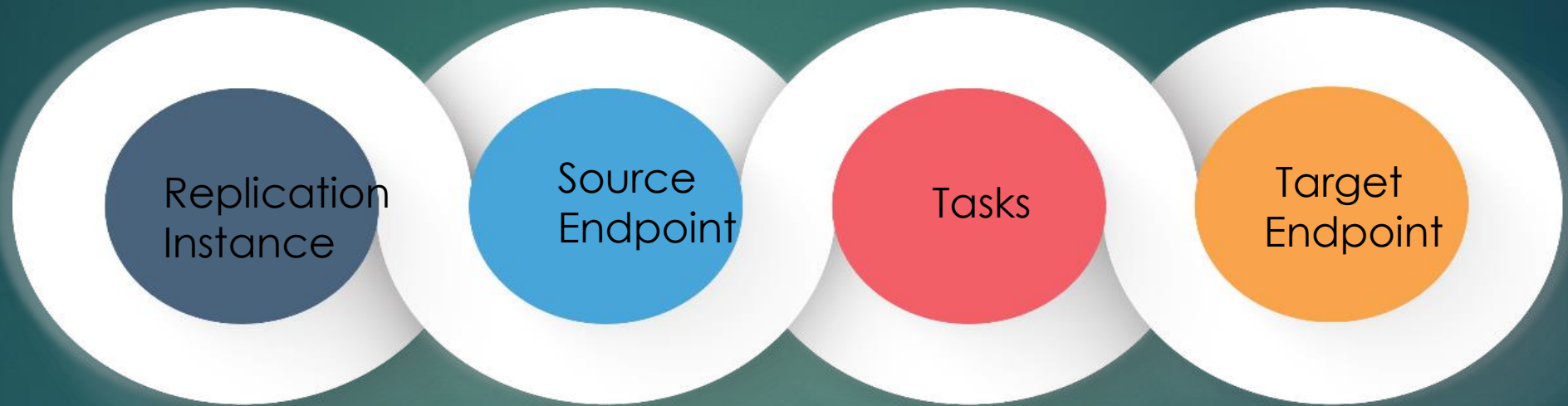


# How AWS DMS Works

- Identify your source and target data stores. These data stores can reside on any of the data engines.
- For both the source and target, configure endpoints within AWS DMS that specify the connection information to the databases. The endpoints use the appropriate ODBC drivers to communicate with your source and target.
- Provision a replication instance, which is a server that AWS DMS automatically configures with replication software.
- Create a replication task, which specifies the actual data tables to migrate and data transformation rules to apply. AWS DMS manages running the replication task and provides you status on the migration process.



# AWS DMS Components



# AWS DMS Components

- **Replication instance:** Replication instance (EC2) provides the compute resources you need for the migration. As with other AWS services, you have the freedom to choose which size fits your needs best. It also provides High Availability and fail-over capabilities.
- 
- **Endpoints:** Either called 'source' or 'target' endpoints, which specify the entry points to the database you are migrating from/to.
- **Tasks:** Specify the endpoints you want to use for the migration, task settings which can be tuned for each type of migration, and table mappings which allow you to perform some additional modifications at schema, table or column level during the migration.

# DMS Components – Replication Instance

## ➤ EC2 instance class

- **Smaller EC2 instance** classes are sufficient for testing the service or for small migrations.
- If migration involves a large number of tables and to run multiple concurrent replication tasks, consider using one of the **larger instances**.

## ➤ Storage

- Depending on the EC2 instance class storage comes with either 50 GB or 100 GB of data storage.
- Storage is used for log files and any cached changes collected during the load
- If there are multiple tasks running on the replication server, you might need to increase the amount of storage

# DMS Pricing

---

- Pay only for the compute resources and additional log storage used during migration process .
  - ❖ On-Demand Instances let you pay for database migration by the hour.
  - ❖ Currently supports the T2, C4 and R4 instance classes (R4 in limited regions)
- Data Transfer
  - ❖ All data transfer into DMS Service and between DMS and databases in Amazon RDS and Amazon EC2 Instances in the same Availability Zone is free.
  - ❖ Standard AWS data transfer rates apply when there are any migrations from source database to a target database in a different Availability Zone, Region, or outside of AWS

**For more details regarding pricing , refer the link**  
**<https://aws.amazon.com/dms/pricing/>**

# DB Migration to AWS – 3 Stages

- ▶ AWS DB Migration process can be divided into 3 stages
  - ▶ Stage 1
    - ▶ Pre migration
  - ▶ Stage 2
    - ▶ Migration
  - ▶ Stage 3
    - ▶ Post Migration
- ▶ In each stages users have to perform certain tasks which are detailed in subsequent slides

# DB Migration to AWS – Stage 1

- Learn all about all DB migration related Services
- Identify and priorities DB'S which need to be migrated
  - First migrate those DB'S which have least dependencies
- In case of heterogeneous migration , use SCT tool to get an advance report on DB Migration
- This report can help in an early understanding of DB transformation complexity
- Use **AWS CloudFormation** to define you target AWS cloud **instruction stack**
- Create your entire **target stack using CloudFormation**
- Apply **AWS well architecture framework** on AWS target cloud Architecture



# Identifying and Prioritizing DB's for Migration

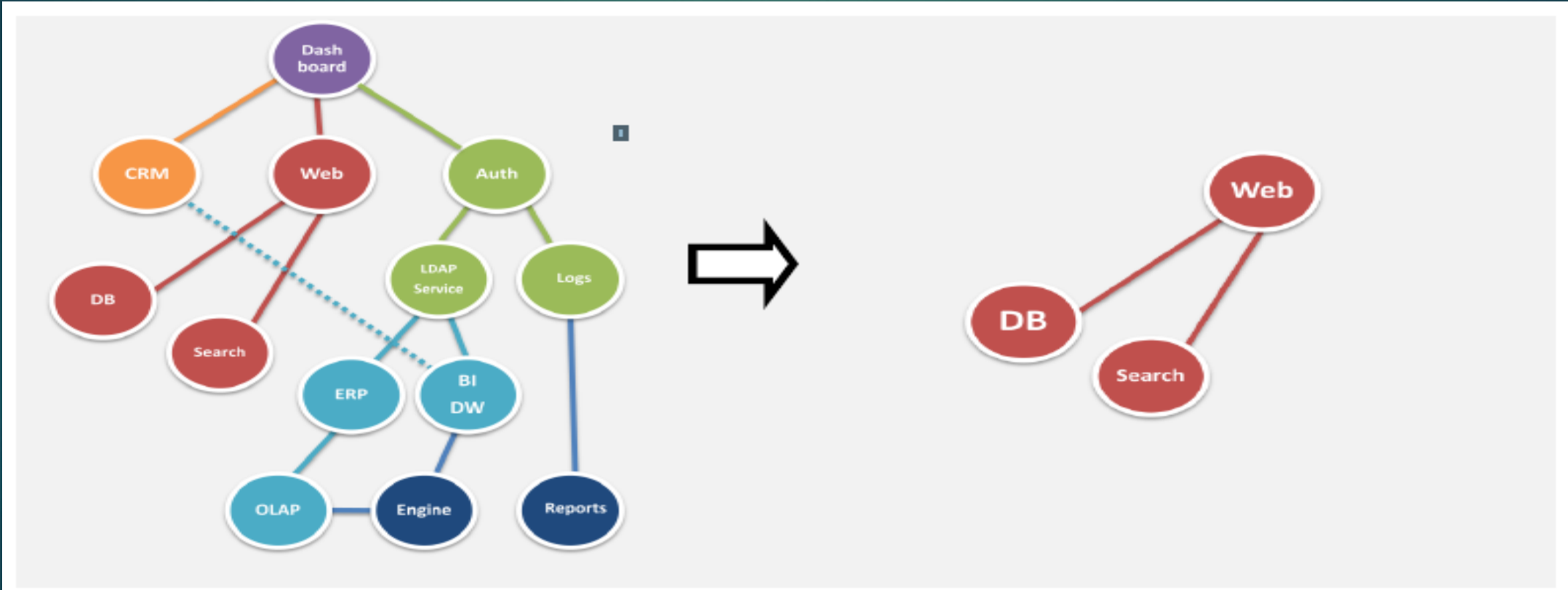
---

- ▶ Consider those DB'S which have least dependencies on other resource
- ▶ Ensure that along with the DB's relevant applications are also migrated other wise you may face latency issues
- ▶ Prioritize those DB whose role is as cloud backup or a read only DB



# Identifying and Prioritizing DB's for Migration

- For identifying a right DB migration services, create a DB dependency graph as shown below



# DB Migration to AWS – Stage 2

- Run SCT in case of heterogeneous DB migration
- Configure DMS for DB Migration
- Initiate your DB Migration task
- Use **AWS Migration Hub** to monitor all DB migration

# Running SCT tool for Heterogeneous DB Migrations

- Step 1 - Install SCT tools
  - ▶ For details Please use this [link](#)
- ▶ Step 2 – Connect to your source DB
- ▶ Step 3 – Connect to your Target DB
- ▶ Step 5 – View the schema feedback report to understand the complexity of DB migration
- ▶ Step 4 – Convert your schema from Source to Target
- ▶ Step 5 – Apply your schema on target DB

# DB Migration to AWS – Stage 3

- Validate That **all data** has been migrated
- Perform all post migration activities
- Integrate your data with other AWS cloud services
- Apply again well **AWS architecture framework** to further optimize your DB in Cloud

# DMS Configuration

- At a high level, when using AWS DMS, we should do the following for configuration:
  - Create a replication server.
  - Create source and target endpoints that have connection information about your data stores
  - Create one or more tasks to migrate data between the source and target DB'S

# Step wise Configuring DMS Service

## ▶ Step 1

- ▶ [Go to DMS home Page](#)

## ▶ Step 2

- ▶ Create DMS replication instance

## ▶ Step 3

- ▶ create aws source and target points

## ▶ Step 4

- ▶ Create and run your AWS Task

- Step 5

- Monitor your DB migration process

- Step 6

- Validate that you DB migration process has been successful

# DMS Configuration Steps

## Setting up your first replication task

### Step 1: Welcome

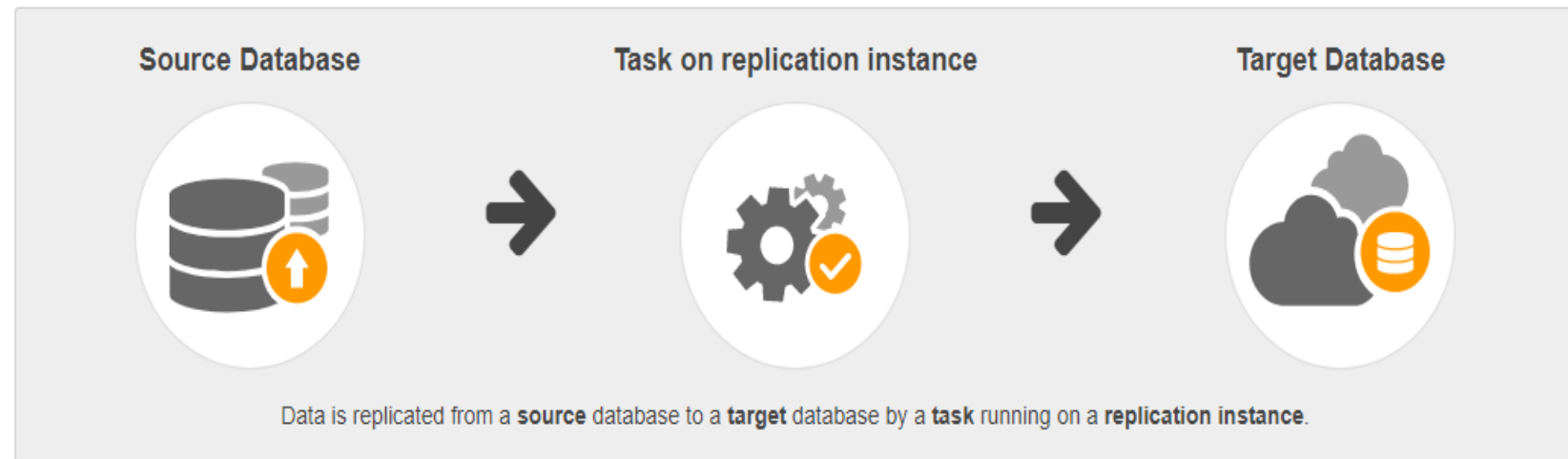
Step 2: Replication instance

Step 3: Database endpoints

Step 4: Task

## Welcome to AWS Database Migration Service

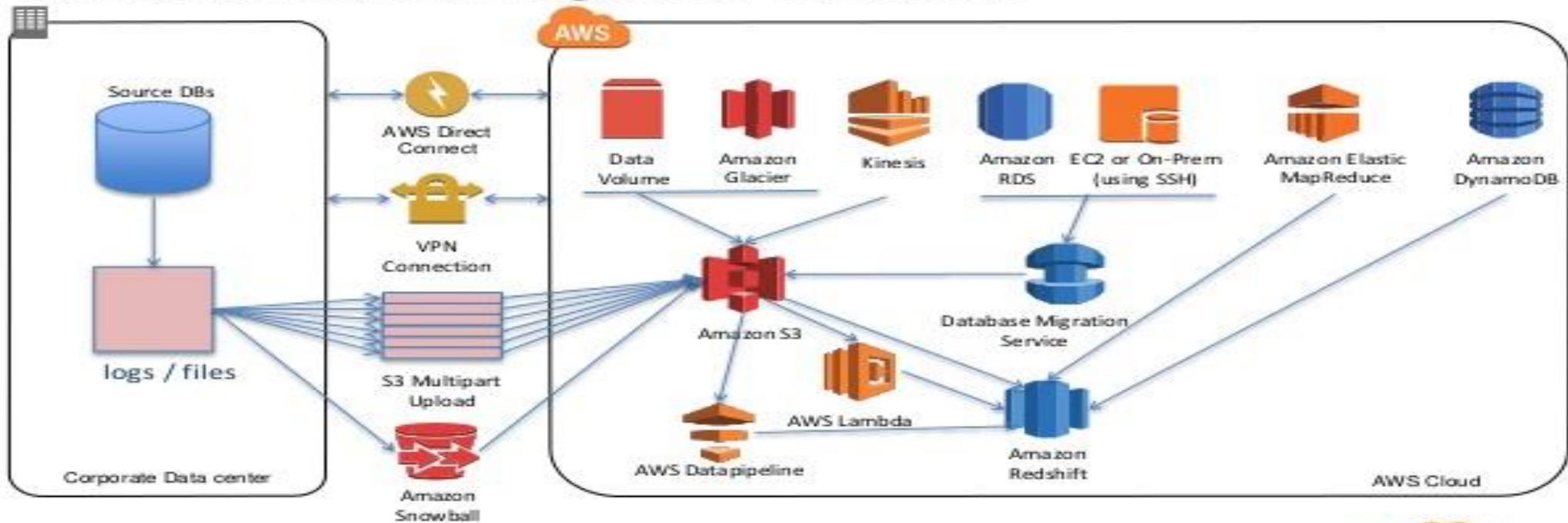
AWS Database Migration Service tasks require at least a source, a target, and a replication instance. Your source is the database you wish to move data from and the target is the database you're moving data to. The replication instance processes the migration tasks and requires access to your source and target endpoints inside your VPC. Replication instances come in different sizes depending on your performance needs. If you're migrating to a different database engine, AWS Schema Conversion Tool can generate the new schema for you. [Download AWS Schema Conversion Tool](#)





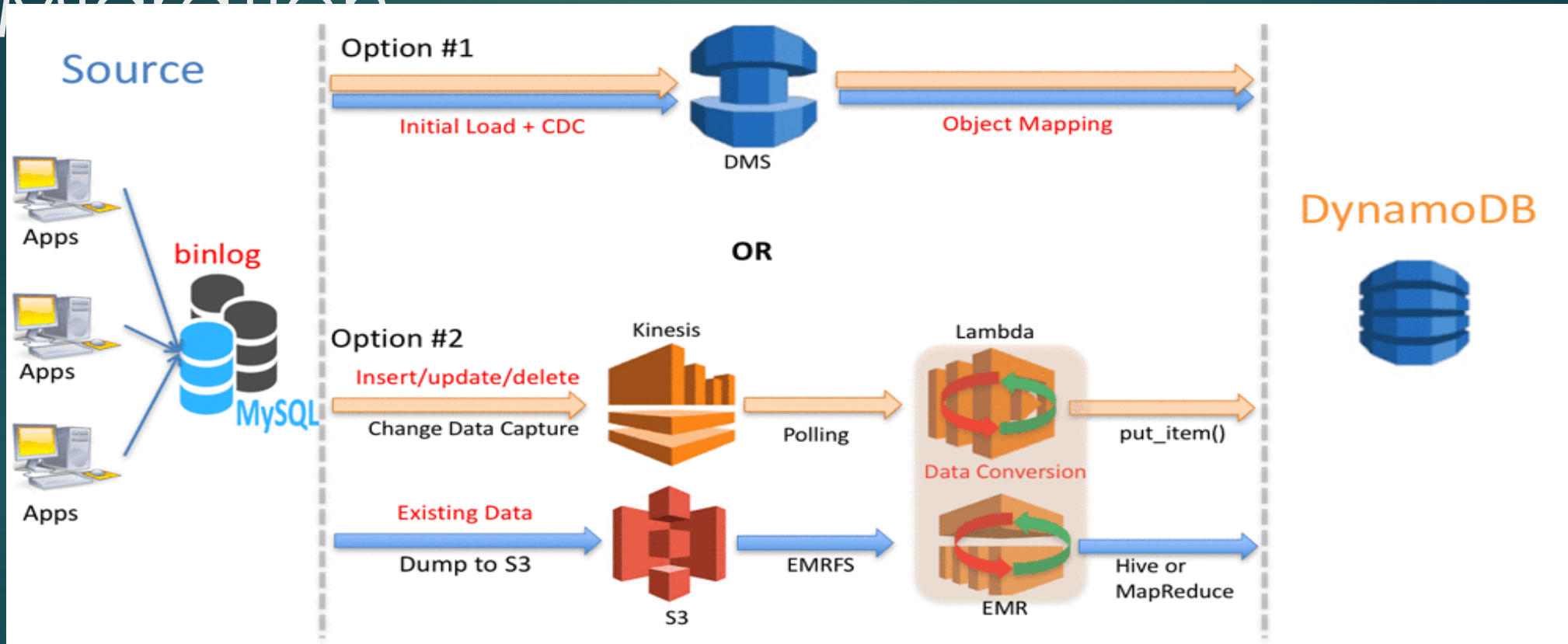
# AWS Redshift Migration Architecture

## Amazon Redshift Migration Overview



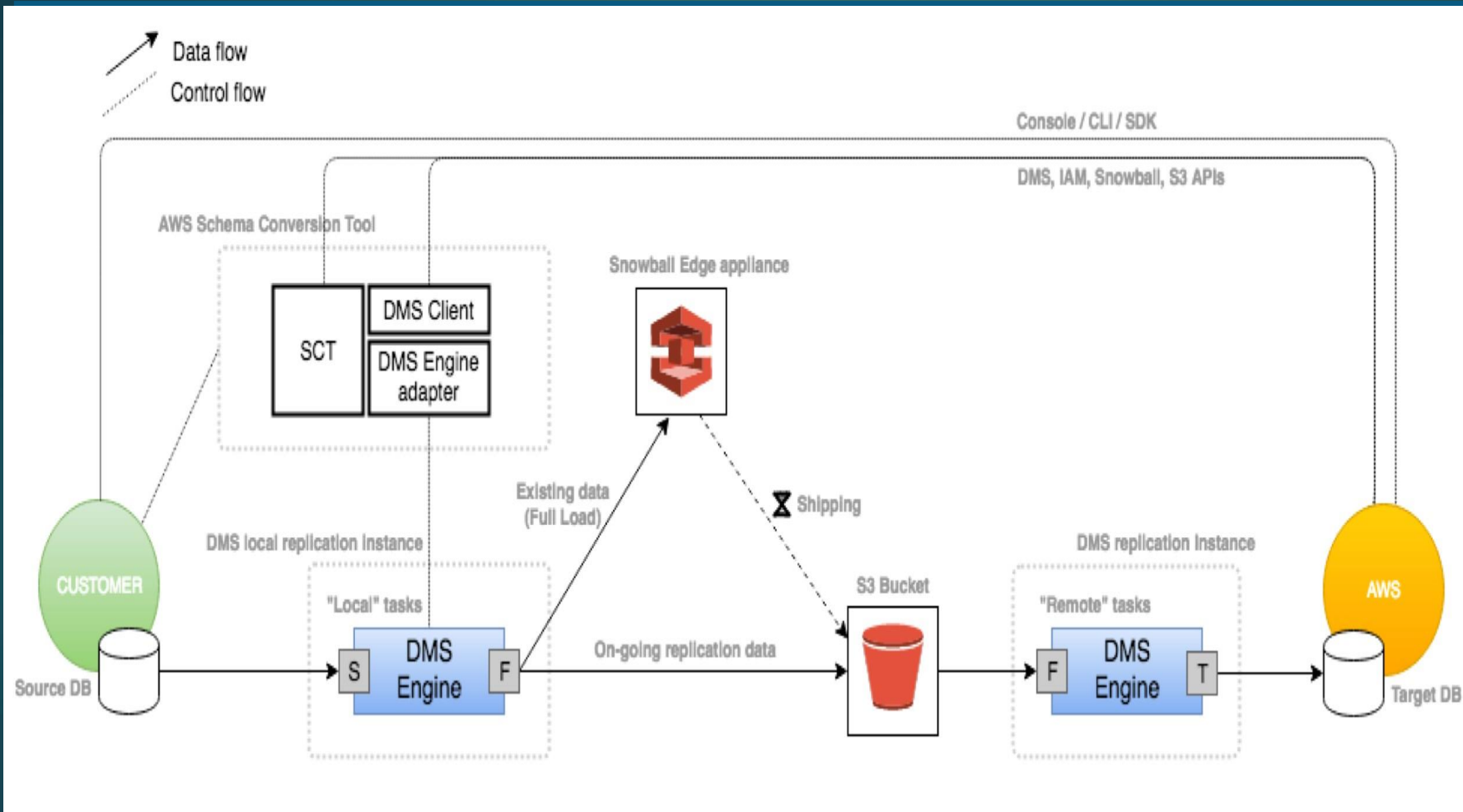
Source: <https://www.slideshare.net/AmazonWebServices/migrate-your-data-warehouse-to-amazon-redshift>

# From RDS to AWS Dynamo DB Migration



<https://aws.amazon.com/blogs/big-data/near-zero-downtime-migration-from-mysql-to-dynamodb/>

# Use Case Mass DB Transfer To AWS Using Snowball edge



AWS snowball is both a service and a secure device

AWS Snowball can be used to transfer very large DB's in the range of 100 TB via shipping logistic, with in two week

This can avoid issue like bandwidth limitation etc..

<https://aws.amazon.com/blogs/database/new-aws-dms-and-aws-snowball-integration-enables-mass-database-migrations-and-migrations-of-large-databases/>

# THANK YOU