Migrating Your Databases to AWS: Deep Dive on Amazon Relational Database Service and AWS Database Migration Service

Shafreen Sayyed
Solutions Architect

June 28, 2017
Agenda

The WHAT

The WHY

The HOW

The WHEN

The WHO
The What: Amazon Relational Database Service
Amazon Relational Database Service (Amazon RDS)

- No infrastructure management
- Cost-effective
- Instant provisioning
- Application compatibility
- Scale up/down
Why: Amazon Relational Database Service
Highlights Amazon RDS

- Multi-engine support: Amazon Aurora, MySQL, MariaDB, PostgreSQL, Oracle, SQL Server
- Automated provisioning, patching, scaling, backup/restore, failover
- High availability with RDS Multi-AZ
  - 99.95% SLA for Multi-AZ deployments
Security
Amazon Virtual Private Cloud (Amazon VPC)

- Securely control network configuration

Manage connectivity

- AWS Direct Connect
- VPN connection
- VPC peering
- Routing rules
- Internet gateway

10.1.0.0/16
10.1.1.0/24

Availability Zone
AWS Region
Security groups

• Database IP firewall protection

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port Range</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>3306</td>
<td>172.31.0.0/16</td>
</tr>
<tr>
<td>TCP</td>
<td>3306</td>
<td>“Application security group”</td>
</tr>
</tbody>
</table>
Encryption in Transit
Database traffic encryption with SSL/TLS

Available for all six engines
Encryption At Rest

- DB instance storage
- Automated backups
- Read Replicas
- Snapshots

- Available for all six engines
- No additional cost
- Support compliance requirements
Compliance

AICPA Service Organization Control Reports
FedRAMP
GOV.UK
FedRAMP
Hipaa
ISO
PCI Security Standards Council
ITAR

Singapore MTCS
27001/9001
27017/27018
High availability
Minimal deployment - Single AZ

VPC

10.1.0.0/16

10.1.1.0/24

Amazon Elastic Block Store volume

Availability Zone

AWS Region

PROD Env
High availability - Multi-AZ

**Availability Zone A**
- 10.1.0.0/16
- 10.1.1.0/24

**Availability Zone B**
- 10.1.2.0/24

**Same instance type as master**

**Synchronous replication**

VPC

AWS Region
High availability - Multi-AZ to DNS

dbinstancename.1234567890.us-west-2.rds.amazonaws.com:3006
Amazon Aurora - High availability

- Purpose-built log-structured distributed storage system designed for databases
- Storage volume is striped across hundreds of storage nodes distributed over 3 different availability zones
- Six copies of data, two copies in each availability zone to protect against AZ+1 failures
- Plan to apply same principles to other layers of the stack
The How: Getting onto Amazon Relational Database Service
Database Migrations ??
AWS Database Migration Service (DMS) easily and securely migrates and/or replicate your databases and data warehouses to AWS.

AWS Schema Conversion Tool (SCT) converts your commercial database and data warehouse schemas to open-source engines, Amazon Aurora and Redshift. Converts and loads data warehouse data into Amazon Redshift.

We have migrated over 26,000 unique databases using DMS. And counting…
When: AWS Database Migration Service
When to use DMS and SCT?

**Modernize**
- Modernise your database tier –
  - Commercial to open-source
  - Commercial to Amazon Aurora
- Modernise your Data Warehouse –
  - Commercial to Redshift

**Migrate**
- Migrate business-critical applications
- Migrate from Classic to VPC
- Migrate data warehouse to Redshift
- Upgrade to a minor version
- Consolidate shards into Aurora

**Replicate**
- Create cross-regions Read Replicas
- Run your analytics in the cloud
- Keep your dev/test and production environment sync
AWS SUMMIT

.... But there’s more!
Fanning-In

Source

Source

Source

Replication instance

Target
Fanning-Out

Source → Replication instance → Target

Target

Target

Target
Homogenous or heterogeneous

- Oracle
- SQL Server
- Amazon Aurora
- Amazon S3

Oracle

Oracle
Why: AWS Database Migration Service
Why use DMS and SCT?

- Remove Barriers to Entry
- Near-Zero Downtime
- Secure
- Easy to Use, but Sophisticated…
- Allow DB Freedom
- Keep a Leg in the Cloud
- Cost Effective
How: AWS Database Migration Service works
Database migration process

Step 1: Convert or Copy your Schema

Source DB or DW → Copy or Convert → Native Tool

Native Tool → Copy → Destination DB or DW

Source DB or DW

AWS SCT

Destination DB or DW

Step 2: Move your data

Source DB or DW → Database → AWS DMS

AWS DMS

Source DB or DW

AWS SCT

Destination DB or DW
Keep your apps running during the migration

- Start a replication instance
- Connect to source and target databases
- Select tables, schemas, or databases

- Let AWS DMS create tables, load data, and keep them in sync
- Switch applications over to the target at your convenience
New SCT data extractors

Extract Data from your data warehouse and migrate to Amazon Redshift

- Extracts through local migration agents
- Data is optimized for Redshift and Saved in local files
- Files are loaded to an Amazon S3 bucket (through network or Amazon Snowball) and then to Amazon Redshift
Who: Customer Use Cases
Who is saying What about DMS and SCT?

"The **SCT Assessment Report** was the key enabler to allow us to understand the scope of effort required to complete an **Oracle to PostgreSQL migration**. What was originally thought to be a largely manual task that no one was particularly excited about having to do became a **very straight-forward quick and easy process**."

"We migrated hundreds of our clients from our in-house data-center to Amazon RDS Oracle 12c using the AWS Data Migration Service (DMS). Due to this service, we could live-replicate the databases between our data-center and RDS before the migration. That kept the migration **down-time to the very minimum**. We are very happy with DMS and are **planning to use it for Oracle to MySQL migration next**."

“We are in the process of migrating some databases to Amazon Aurora. **The ease by which we can do this using the AWS Database Migration Service has simplified this process for us and enabled us to accelerate our migration efforts**. The ability to closely monitor the process, the detailed logging feature, and the support we received from AWS have given us a great deal of confidence in a successful migration.”
Our migration to AWS

A migration to Aurora using DMS
Who are Currencycloud

A complete cross border payment solution

APIs
Customize your solution with features only you and your clients need

Orchestrated Workflows

Currencycloud Direct
Get to market quickly with a fully branded off-the-shelf solution

Receive
Convert
Pay
Manage
Our old tech stack

• Ruby, JRuby, TorqueBox, RabbitMQ, Redis, MySQL, Debian
• Apps on a mix of virtual & ec2 servers
• MySQL Databases on physical servers & ec2
• Overly complex multi master mysql replication setup
Why we needed a new approach to the database

• Multi Master Replication was fragile
• Failover was manual & slow
• We needed simpler db management tools
• We were using large Physical and AWS instances
• Adding capacity was expensive (PIOPs or NVMe storage)
Our Requirements

• Data encrypted at rest and in flight
• Solution needs to be very scalable
• Must be flexible to enable fast changes to system design
• Failover should be automatic & faster
• Use a templated approach to ensure same db configuration is used on different environments
How we used DMS

• We used 2 jobs
  • A repeatable snapshot job for testing the new system
  • A continuous job for cut-over to when we migrated
• Dry run before you move Production
• We imported the schema with only minor changes
• Upscale the db source instances for migration to speed things up.
Things we learned

• Monitor DMS job for errors
• Setup cloudwatch on Aurora instances & replication
• Check table character sets
• Talk to a AWS Solutions Architect
• Leverage the experience of a partner
• Don’t try to just lift and shift
• Code changes will be required
Results

• DDL changes are significantly faster
  • E.g 30 mins to 30 secs for alter table
• We have manged to reduce expected DB instance sizes.
• DB running costs are lower than expected
• Failover time’s significantly reduced.
  • ~60 secs from ~30 mins
Next steps

• Lots more schema changes to optimize the data storage structure.
• This is expected to further improve our performance on known slow queries.
• Further usage of read replicas.
• Which should result in instance size reductions.
• Then we will move some machines to Reserved Instances.
Wrapping up
Recap

Minimal Operational Overhead

Reliable, Scalable and Secure deployments

Consistent Migrations
AWS database migration partners
Thank you!

aws.amazon.com/dms