aws certified

Solutions Architect

Professional

DOLFINed

AWS RELATIONAL DATABASE SERVICE (RDS)

AMAZON RELATIONAL DATABASE SERVICE (RDS)

AWS RDS

Relational Database Service (RDS) - Introduction

Review Topic : Relational Database Service

AWS RDS

Is an AWS fully managed Relational DB Engine service where AWS is responsible for:

- Security and patching of the DB instances
- Automated backup for your DB instances (default setting)
- Software updates for the DB engine
- Easy scaling for storage and compute as required
- If selected, Multi-AZ with Synchronous replication between the active and standby DB instances
- Automatic failover if Multi-AZ option was selected
- Providing the ability to create DB read replicas for DB read scaling (intensive read deployments)

- AWS is NOT responsible for:
  - Managing DB Settings
  - Building a relational DB schema
  - DB performance tuning

Review Topic : Relational Database Service

Supported Relational DB engines

- MS SQL Server
- ORACLE (Two licensing models – bring your own license [BYOL] or License included)
- PostgreSQL
- MariaDB
- MySQL

- AWS Aurora (Explained in a separate section)

Review Topic : Relational Database Service

RDS instance storage

- Amazon RDS use EBS volumes (not Instance-Store) for DB and Logs storage

- General Purpose (gp2) RDS Storage:
  - Used for DB workloads with moderate I/O requirements

- Provisioned IOPS (io1) RDS Storage:
  - Used for High performance OLTP workloads
    - Provisioned IOPS storage is optimized for I/O intensive workloads that require low I/O latency and consistent throughput
      - Ex. Online Transaction Processing (OLTP) workloads that have consistent performance requirements.

- Magnetic RDS Storage:
  - Use for small DB workloads

Review Topic : Relational Database Service

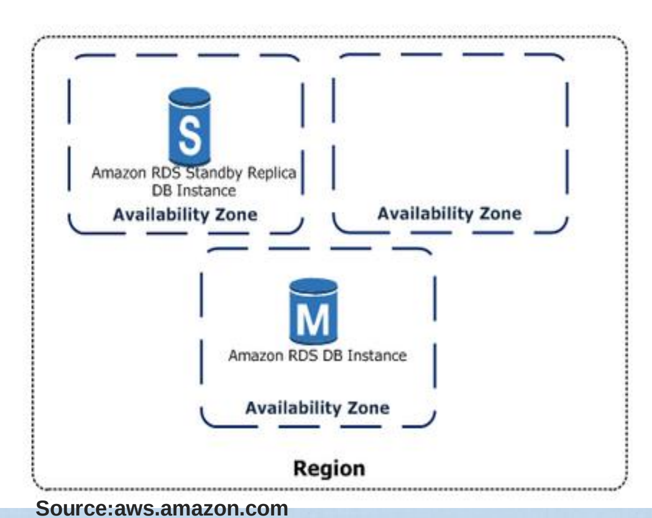
DB Subnet Group

- Is a collection of subnets in a VPC that you allocate for DB instances launched in the VPC
- Each DB subnet group must have at least one subnet in each AZ in a region
- AWS recommends, even if you are starting with standalone RDS instance, configure the subnet group with a subnet in each AZ in the region.
  - This will facilitate launching your standby instance in the subnet group when you opt for the multi-AZ deployment
- During creating your RDS instance you can select a preferred AZ, and specify which Subnet group, and subnet of that group, for your RDS DB instance
  - Then RDS service will allocate an IP address in that subnet to your RDS instance
  - And then RDS service will create an ENI, attach it to the RDS instance, and assign the above IP address to it

Review Topic : Relational Database Service

Multi-AZ RDS option

- Multi-AZ for RDS provides high availability, data durability, and fault tolerance for DB instances
- Oracle, MySQL, PostgreSQL, Maria DB use AWS failover technology
- MS SQL Server uses SQL Server DB Mirroring (DBM)
- You can select the Multi-AZ option during RDS DB instance launch or modify an existing standalone RDS instance
- RDS service creates a standby instance in a different AZ in the same region, and configures "SYNCHRONOUS" replication between the primary and standby
- You can NOT read/write to the Standby RDS DB instance



Source:aws.amazon.com

Review Topic : Relational Database Service

Multi-AZ RDS – Failover trigger

Failover may be triggered when:

- Loss of primary AZ or primary DB instance failure
- Loss of network connectivity to primary
- Compute (EC2) unit failure on primary
- Storage (EBS) unit failure on primary
- The primary DB instance is changed
- Patching the OS of the DB instance
- Manual failover (reboot with failover on primary)

- Failover Logistics:
  - During failover, the CNAME of the RDS DB instance is updated to map to the standby IP address (use the endpoint to reference your DB instances and not its IP address)
  - The CNAME itself does not change, because the RDS endpoint does not change

Review Topic : Relational Database Service

Processed that happen on Standby first

- The following procedures are done on standby first, then on primary
  - OS Patching
  - System upgrades
  - DB Scaling
- In Multi-AZ, Snapshots and Automated backups are done on Standby instance to avoid I/O suspension on Primary instance
- DB engine version upgrades happen on both primary and standby at the same time (causes an Outage)
- Maintenance sequence of events in Multi-AZ:
  - Maintenance on Standby is performed
  - Standby promoted to Primary
  - Maintenance performed on old primary (Current Standby)

Review Topic : Relational Database Service

DB Automated Backups or Manual Snapshots

- There are the two methods to backup your RDS DB instances
  - AWS RDS automated backups
  - User initiated manual backups
- Either one creates a storage volume snapshot of your entire DB instance that gets saved in S3
  - Not just the individual databases
- You can make copies of automated backups and of manual snapshots. The resulting copy is considered a manual snapshot
- You can share manual snapshots, but not the automated ones
  - If you need to share an automated backup, make a copy first, then share the copy
- Retention period: AWS RDS keeps the automated backup for 7 days by default (retention period of 0 means no retention. It can be configured up to 35 days)

Review Topic : Relational Database Service

DB Automated Backups

- Automated backups are used for point-in-time DB instance recovery
- By default, RDS automatically backs up the DB instances daily, by creating a storage volume snapshot of your DB instance (full daily snapshot), including **the DB transaction logs (modifications),**
  - You can choose when during the day this is done (Backup window)
  - No additional charge for RDS backing up your DB instances
  - Enabled by default, you can disable it by setting retention period to zero (0)
- The first snapshot is a full one, and then subsequent snapshots are incremental
- It can restore the DB up to 5 minutes in time using the DB transaction logs and the Automated snapshot

Review Topic : Relational Database Service

DB Manual Snapshots

- Are not used for point-in-time recovery
- Stored in Amazon S3
- They are not deleted automatically when you delete your RDS instance. Rather, they will stay on S3 until you go ahead and delete them
- It is recommended to take a final snapshot before deleting your RDS DB instance
- Can be shared with other AWS accounts directly

Review Topic : Relational Database Service

DB Automated Backups – Restore/Recovery

- You can specify a point-in-time restore to any given second during the retention period
- When you initiate a point-in-time recovery, transaction logs are applied to the most appropriate daily backup to restore your DB to that point-in-time
- You can not restore into an existing DB instance, it has to be a new DB instance with a new Instance/DB endpoint
- When you restore a DB instance, only the default DB parameters & Security groups are associated with the restored instance
  - Once the restore is complete, you need to associate/apply the customer DB parameters and security group settings
- You can change the Storage type (magnetic, Provisioned IOPS, General purpose) during a restore process

AWS RDS

Relational Database Service (RDS) – DB Read Replicas

Review Topic : Relational Database Service

Read Replicas and Scaling DB read operations

- Remember that the standby DB instance in a Multi-AZ deployment **CAN NOT** be used for Read or Write.
- When the required read I/O capacity is reached but still more I/O capacity is required for heavy/intensive read applications, **RDS read replicas** can be helpful
- A read replica is a replica of the primary RDS DB instance that can only be used for read actions
- Amazon RDS uses the built-in replication functionality within MySQL, PostgreSQL, Oracle, an d MariaDB to create the read replicas.
- Automatic backups must be enabled and remain enabled for read replicas to work
- This can be used in the following use cases:
  - Shifting read intensive applications such as Business (or Sales) reporting, or Data Warehousing to read from read replicas as opposed to overload the primary DB
  - Scaling beyond the I/O capacity of your main DB instance for read-heavy workloads
  - Service read traffic while the source is unavailable

Review Topic : Relational Database Service

Read Replicas

- Read replicas can be created in the same region as the master DB
  - In this case they must also be in the same VPC, not in a peered VPC in the same region
- They can also be created in a different region
  - If the read replica is in a different AWS region, Amazon RDS establishes a secure channel for replication between the master DB and the read replica
- Amazon RDS does not support a read replica on an EC2 instance (self managed) or on premises
- When initiated, RDS takes a snapshot from the master DB and creates the replica from that snapshot
- Primary DB instance becomes the source of the replication to the read replica
  - Using **asynchronous replication** (a time lag exists) data gets replicated to the read replica
  - If in Muli-AZ setup, the replication is done from the standby instance instead

Review Topic : Relational Database Service

Read Replicas

- Automatic backups must be enabled and remain enabled for read replicas to work
- RDS Multi-AZ can be combined with read replicas in one deployment
- For MySQL, MariaDB you can create a read replica from a read replica.
- Multi-AZ for read replicas:
  - It is also possible to create a standby read replica DB instances for your read replica instances. This is supported by MariaDB, PostgreSQL, and MySQL.
    - This is independent from whether the primary (source) DB instance was a standalone or Multi-AZ

Review Topic : Relational Database Service

RDS Replication – Replication to/from Non RDS DBs

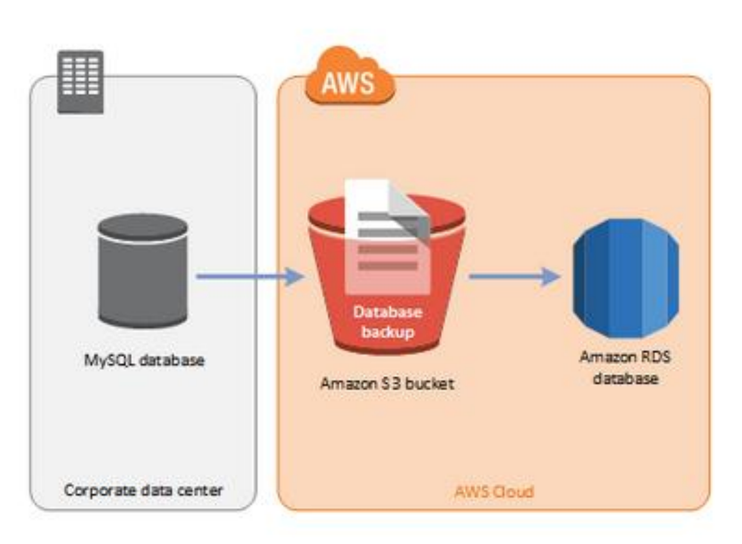
- RDS replication can be used to export data from an RDS MySQL (5.6 or later) to a MySQL external to RDS on EC2 instance or on-premise
  - This is just to migrate the data, can not be used as an ongoing replication.

- Replication can be configured between an external DB (MySQL or MariaDB) as the master (source) DB to an existing MySQL or MariaDB replica on RDS.
  - This can be used on-going, or to migrate the external DB to Amazon RDS (by failing over to the RDS replica once done)

Review Topic : Relational Database Service

Restoring Backup into an Amazon RDS MySQL DB

- Amazon RDS supports importing MySQL databases by using backup files.
- Backup your on-premise or on EC2 instance MySQL DB to S3
- Use the backup file to restore into a new Amazon RDS MySQL DB



Review Topic : Relational Database Service

Importing Data from Non RDS DBs

- You can **import data** from an existing MySQL or MariaDB into an existing RDS MySQL or MariaDB instance

- If the DB external to RDS (on premise) is large or is supporting live applications then:
  - Create a backup copy of the DB on premise
  - Compress the copied data
  - Create an EC2 instance and copy the compressed DB to it
  - Create an Amazon MySQL or MariaDB instance in the same region as the EC2 instance (install MySQL or MariaDB client tools on the instance)
  - Import the data from the EC2 instance
  - Use MySQL or MariaDB replication to bring the new RDS DB up-to-date with the on-premise source DB (changes have happened to the source during the above steps)

Review Topic : Relational Database Service

Promoting Read replicas to Standalone DB instance

- You can promote a read replica into a standalone/single AZ database instance
  - This is true for MySQL, MariaDB, PostgreSQL, and Oracle
  - Read replica will be rebooted before the standalone DB instance becomes available

- The promoted replica into a standalone DB instance will retain:
  - Backup retention period
  - Backup window
  - Option Group
  - DB parameter group of the formed read replica source (primary instance)

AWS RDS

RDS DB Security and Encryption

Review Topic : Relational Database Service

DB Instance encryption

- RDS Service supports **encryption at rest (**i.e data that is on the DB instance**)** for all DB engines using AWS KMS
- If you enable your RDS DB instance encryption at rest, underlying DB Storage, Logs, Snapshots, Read replicas, and automated backups will all be encrypted
- Amazon RDS uses AWS KMS for encryption keys
- RDS supports SSL encryption for communication between the DB clients and the RDS DB instances
  - RDS generates a certificate for the instance which is used to encrypt this communication
- You can NOT enable encryption for an existing, un-encrypted database instance, alternatively to do that, you need to create an encrypted copy of that DB, this is how:
  - Create a snapshot of that DB
  - Copy the snapshot and choose to encrypt it during the copy process
  - Restore the encrypted copy into a **New DB**
- The resulting DB is an encrypted copy of your original, un-encrypted, DB
- You can't disable encryption on an encrypted DB

Review Topic : Relational Database Service

Transparent Data Encryption (TDE)

- Amazon RDS supports TDE for Oracle (Enterprise Edition) and MS SQL Server
- In TDE, data is automatically encrypted before it is written to DB storage, and automatically decrypted when it is read from storage.
- This can come in handy in scenarios where compliance requirements or the need to protect sensitive data is required
- Once TDE is enabled in an RDS option group it can’t be disabled or turned off.
- TDE for SQL Server and Oracle can be simultaneously used with RDS encryption at rest, but this can impact the DB performance slightly

Review Topic : Relational Database Service

Amazon RDS encryption and Read Replicas

- A read replica of an Encrypted RDS instance is also encrypted
  - With the same RDS encryption key if the read replica is in the same region, or
  - With the read replica region’s encryption key if in another region than the DB instance
- The read replica encryption status is like that of the original/Master DB,
  - Hence, you can't have an encrypted Read Replica of an unencrypted DB instance
  - Also, you can’t have an unencrypted Read Replica of an encrypted DB instance.
- You can't restore an unencrypted backup or snapshot to an encrypted DB instance.
  - As discussed, to do this, make a copy, choose to encrypt it, then restore from the encrypted copy
- To copy an encrypted snapshot from one AWS Region to another, you must specify the KMS key identifier of the destination AWS Region. This is because KMS encryption keys are specific to the AWS Region that they are created in.
- The source snapshot remains encrypted throughout the copy process.

Review Topic : Relational Database Service

IAM DB Authentication for MySQL and PostgreSQL

- Is a mechanism that allows authentication to access the RDS DB instance

- IAM database authentication works with MySQL and PostgreSQL.

- With this authentication method, you don't need to use a password when you connect to a DB instance. Instead, you use an authentication token.
  - An *authentication token* is a unique string of characters that Amazon RDS generates on request.
  - Authentication tokens are generated using AWS Signature Version 4.
  - Each token has a lifetime of 15 minutes.
  - You don't need to store user credentials in the database, because authentication is managed externally using IAM.

- You can also still use standard database authentication.

Review Topic : Relational Database Service

IAM DB Authentication for MySQL and PostgreSQL - Benefits

- Network traffic to and from the database is encrypted using Secure Sockets Layer (SSL).
- You can use IAM to centrally manage access to your database resources, instead of managing access individually on each DB instance.
- For applications running on Amazon EC2, you can use profile credentials specific to your EC2 instance to access your database instead of a password, for greater security.

- Use this feature with care as currently it does not scale well for large number of DB connection requests per second

Review Topic : Relational Database Service

RDS DB Instances – Security Best Practices

- Use AWS IAM accounts to control access to RDS API actions
- Assign an individual IAM account to each person who manages RDS resources
- Grant the least permissions required by each user to perform the assigned duties
- Use IAM groups to manage/grant permission to multiple users at one time
- Rotate your IAM credentials regularly

Review Topic : Relational Database Service

Scaling

- You can scale (Up only, not down) the compute and storage capacity of your existing RDS DB instances
  - Scaling storage can happen while the RDS instance is still running (some performance impact)
  - Scaling compute will cause a downtime to your DB instance

- If you hit the largest RDS DB instance, and you still need to scale, you can:
  - Use partitioning and split your RDS DB over multiple RDS instances

AWS RDS

Relational Database Service (RDS) – Billing and DB Instance Purchasing

RDS Billing – Standalone DB Instance

- No upfront costs

- You pay for:
  - DB instance hours (partial hours charged as full hours)
  - Storage GB/mo.
  - I/O requests/mo.   - for Magnetic RDS storage Instance only
  - Provisioned IOPS/mo. – For RDS Provisioned IOPS SSD instance
  - Data transfer in and out of the DB instance from/to the internet or other AWS regions
  - Backup Storage (DB backups, and Active manual Snapshots)
    - This increases by increasing DB backups retention period
    - Backup storage for automated RDS backups (not the manual snapshots) up to the Provisioned RDS instance's EBS volume size (EBS volume) is free of charge

Review Topic : Relational Database Service

RDS Billing – Multi-AZ deployments

AWS will charge for the following (in addition to the single AZ DB instance charges)

- Multi-AZ DB hours
- Provisioned Storage (Multi-AZ)
- Double write I/Os (writing to the Active/Primary, and Replication from Primary to Standby)
- You are not charged for DB data transfer during replication from primary to standby
- Your DB storage does not change between Standalone and Multi-AZ deployments (same DB and same AWS Storage volume for that DB in multiple AZs for durability)

Review Topic : Relational Database Service

Size-Flexible Reserved DB Instances (RIs)

- Similar to EC2 Reserved Instances
  - One or three year term options
- DB RIs are "region" specific
- Each reservation must be specific in:
  - DB Engine
  - DB Instance class
  - Region
    - For RDS RI pricing to apply, an Exact RDS instance must be created on-demand, exact on all above (DB Engine, Instance class, **and** region)

- You can NOT move RDS RIs between regions
- You can move RDS RIs between AZs in the same region
- You can NOT cancel an RDS RI's reservation

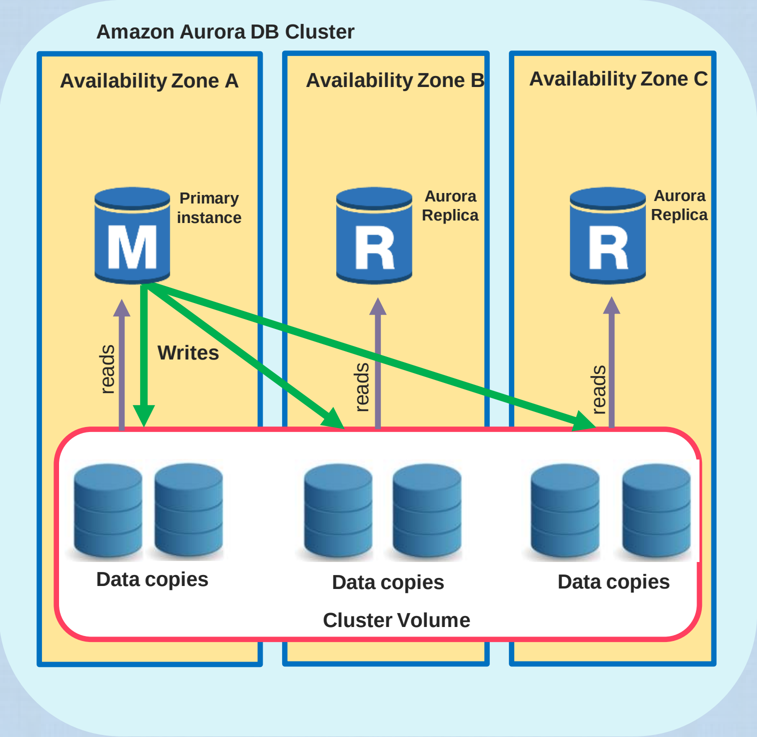
AMAZON RDS - AURORA

AWS AURORA

Introduction & Aurora Clusters

Amazon Aurora

- It is a fully managed, AWS proprietary, relational database engine that's compatible with MySQL and PostgreSQL.
  - The code, tools, and applications used today with existing MySQL and PostgreSQL databases can be used with Aurora.
- Aurora can deliver up to 5 times the throughput of MySQL and up to 3 times the throughput of PostgreSQL without requiring changes to most of your existing applications.
- Its MySQL and PostgreSQL compatible database engines are customized to take advantage of that fast-distributed storage.
- The underlying storage grows automatically as needed, up to 64 TiB.
- An Amazon Aurora DB cluster consists of one or more DB instances and a cluster volume that manages the data for those DB instances.
- The Aurora cluster volume is a virtual database storage volume that spans multiple AZ's in one region.
  - Aurora maintains multiple (at least 2) copies of your data in three Availability Zones, in a single AWS region

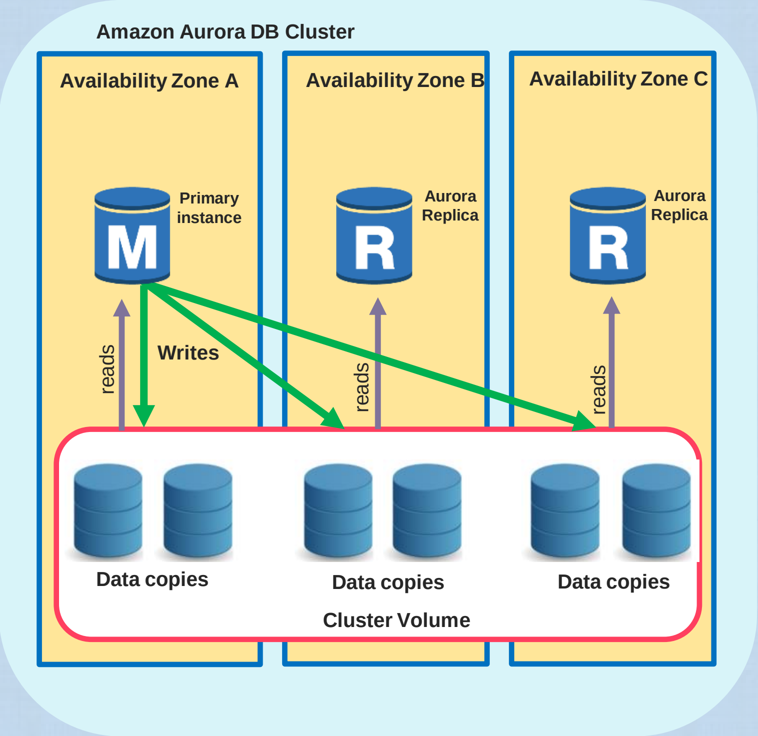


Amazon Aurora DB Clusters

- The Aurora cluster illustrates the separation of compute capacity and storage.
  - An Aurora configuration with only a single DB instance is still a cluster,
    - Since the underlying storage volume involves multiple storage nodes distributed across multiple Availability Zones (AZs).

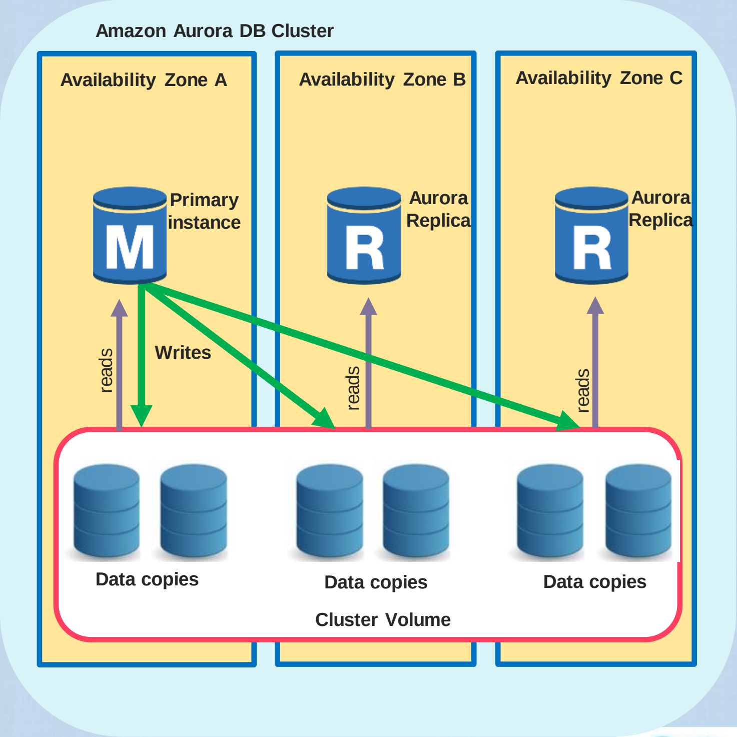
Two types of DB instances make up an Aurora single Master DB cluster:

- **Primary DB instance**
  - *Supports read and write* operations and performs all of the data modifications to the cluster volume.
  - Each Aurora DB cluster has one primary DB instance.
- **Aurora Replica**
  - Connects to the same storage volume as the primary DB instance and *supports only read operations (scaling read operations for the cluster).*



Amazon Aurora DB Clusters

- Each Aurora DB cluster can have up to 15 Aurora Replicas in addition to the primary DB instance.
- You can Maintain high availability by locating Aurora Replicas in separate Availability Zones.
- Aurora automatically fails over to an Aurora Replica in case the primary DB instance becomes unavailable.
  - ○ You can specify the failover priority for Aurora Replicas.

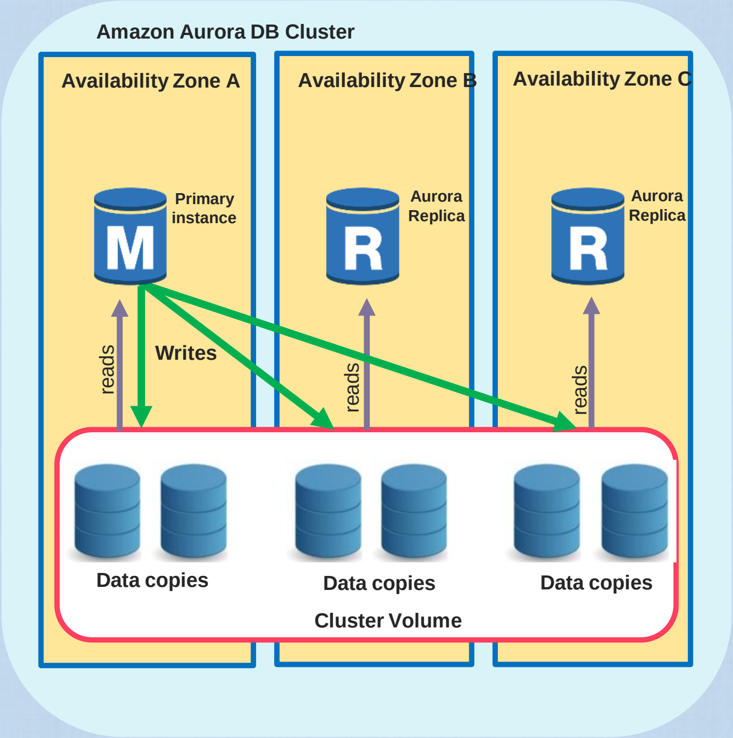


Amazon Aurora

- Aurora features like automatic clustering, replication, and storage allocation make it cost-effective and simple to set up, operate, and scale large MySQL and PostgreSQL deployments.

- Data can be migrated from Amazon RDS for MySQL and Amazon RDS for PostgreSQL into Aurora, to do this
  - Create RDS snapshot and restore it to Aurora, or by setting up one-way replication.

- Push-button migration tools can be used to convert existing Amazon RDS for MySQL and Amazon RDS for PostgreSQL applications to Aurora.



Amazon Aurora – Connection Management & EndPoints

- When connecting to an Aurora cluster, you use URLs or hostnames known as Endpoints
  - Endpoints ensure that hardcoding all the hostnames or writing own logic for load-balancing and rerouting connections when some DB instances aren't available is not required.

Types of Aurora Endpoints:

- **A cluster endpoint** for an Aurora DB cluster that connects to the current primary DB instance for that DB cluster.
    - This endpoint is the only one that can perform write operations.
  - Example: mydbcluster.**cluster**-123456789012.us-east-1.rds.amazonaws.com:3306

- **A reader endpoint** for an Aurora DB cluster connects to one of the available Aurora Replicas for that DB cluster (or to the primary DB instance if no read replicas). Only one exists per cluster
  - Used for read only operations (can’t be used for writes)
  - The reader endpoint provides load-balancing support for read-only connections to the DB cluster.
  - Example: mydbcluster.cluster-**ro**-123456789012.us-east-1.rds.amazonaws.com:3306

Amazon Aurora – Types of Aurora Endpoints (cont.)

- **An instance endpoint** connects to a specific DB instance within an Aurora cluster.
  - Each DB instance in a DB cluster has its own unique instance endpoint.
  - The instance endpoint provides direct control over connections to the DB cluster.
  - Example use case:
    - A client application might require more fine-grained load balancing based on workload type.
      - In this case, multiple clients can be configured to connect to different Aurora Replicas in a DB cluster to distribute read workloads.
  - Example: mydbinstance.123456789012.us-east-1.rds.amazonaws.com:3306

- **A custom endpoint** for an Aurora cluster represents a **set of DB instances** that you choose.
  - An Aurora DB cluster has no custom endpoints until you create one.
  - When you connect to the endpoint, Aurora performs load balancing and chooses one of the instances in the group to handle the connection.
  - Example: myendpoint.cluster-**custom-**123456789012.us-east-1.rds.amazonaws.com:3306

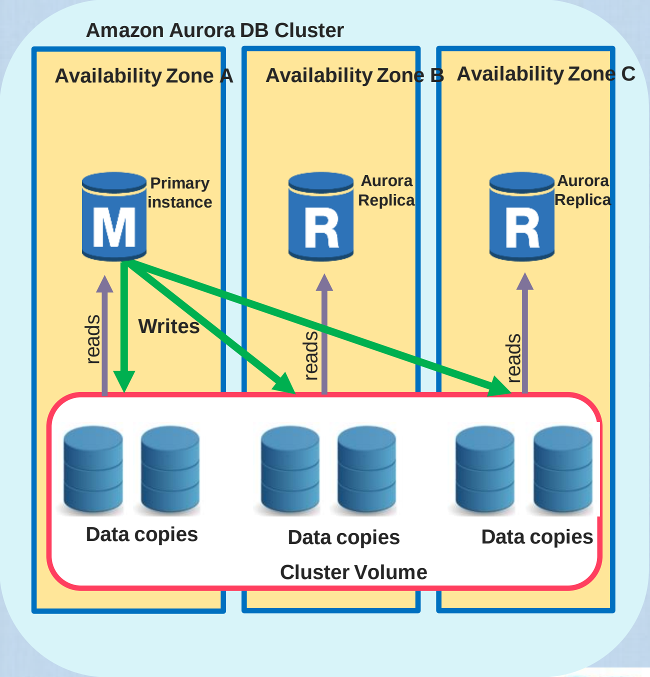
AWS AURORA

- Aurora Replicas
- Autoscaling
- Storage and Reliability
- Primary Failover
- High Availability

Amazon Aurora – Aurora Read Replicas

- Due to this cluster volumes architecture, Aurora Replicas can return the same data for query results with minimal replica lag
    - ○ Usually much less than 100 milliseconds after the primary instance has written an update.
    - ○ Because the cluster volume is shared among all instances, no additional work is required to replicate a copy of the data for each Aurora Replica.

- In contrast to Amazon RDS MySQL, Read Replicas must replay, on a single thread, all write operations from the master DB instance to their local data store.
    - ○ This affects the ability of the Read Replicas to support large volumes of reads.



Auto Scaling with Aurora Replicas

- Aurora Auto Scaling dynamically adjusts the number of Aurora Replicas provisioned for an Aurora DB cluster using single-master replication.
- Aurora Auto Scaling enables Aurora DB clusters to handle sudden increases in connectivity or workload.
- When the connectivity or workload decreases, Aurora Auto Scaling removes unnecessary Aurora Replicas so that no change will be incurred for unused provisioned DB instances.
- Aurora Auto Scaling is available for both Aurora MySQL and Aurora PostgreSQL.
- Although Aurora Auto Scaling manages Aurora Replicas, the Aurora DB cluster must start with at least one Aurora Replica

Amazon Aurora – Primary Failover

- If the primary DB instance of a DB cluster fails, Aurora **automatically** fails over to a new primary DB instance.

It does so by either

- Promoting an Aurora Replica if one exists to be the primary instance.
- If there are no Aurora Replicas in the Aurora Cluster, then the cluster will be unavailable for the duration it takes the DB instance to recover or new DB instance gets created.

Notes

- Promoting an Aurora Replica is much faster than recreating the primary instance.
  - For high-availability scenarios, AWS recommends creating one or more Aurora Replicas of the same DB instance class as the primary instance and in different Availability Zones for your Aurora DB cluster.

AWS AURORA

- Aurora Security
- Aurora Encryption
- Aurora – Global DB
- Aurora with other AWS services

Amazon Aurora - Security

Security for Amazon Aurora is managed at different levels:

- IAM is used to control who can perform Amazon RDS management actions on Aurora DB clusters and DB instances.
  - With IAM database authentication, you authenticate to your Aurora MySQL DB cluster by using an IAM user or IAM role and an authentication token.

- Aurora DB clusters must be created in an Amazon Virtual Private Cloud (VPC).
  - An Amazon VPC endpoint for Amazon Aurora is a logical entity within a VPC that allows connectivity only to Amazon Aurora.
  - The Amazon VPC routes intra VPC requests to Amazon Aurora and routes responses back to the VPC through the VPC endpoint.

- To control which devices and Amazon EC2 instances can open connections to the endpoint and port of the DB instance for Aurora DB clusters in a VPC, a VPC security group is used.

Amazon Aurora - Encryption

- SSL can be used from the client application to encrypt a connection to a DB cluster running Aurora MySQL or Aurora PostgreSQL.

- Encrypt Amazon Aurora DB clusters and snapshots at rest by enabling the encryption option for Aurora DB clusters.
  - Data that is encrypted at rest includes the underlying storage for DB clusters, its automated backups, Read Replicas, and snapshots.
  - Database clients need not be modified to use encryption.

- Amazon Aurora encrypted DB clusters use the industry standard AES-256 encryption algorithm.

- You can't convert an unencrypted DB cluster to an encrypted one.
  - You can, however, restore an unencrypted Aurora DB cluster snapshot to an encrypted Aurora DB cluster.
  - DB clusters that are encrypted can't be modified to disable encryption.

- As of now, an encrypted Aurora Replica can't be created from an unencrypted Aurora DB cluster, and
  - An unencrypted Aurora Replica can't be created from an encrypted Aurora DB cluster.

Amazon Aurora Global Data Base

- An Aurora global database consists of **one primary AWS Region** where data is mastered, and **one read-only, secondary AWS Region.**
  - The Aurora cluster in the primary AWS Region performs both read and write operations.
  - The cluster in the secondary region can scale up to 16 Aurora replicas. It enables low-latency reads only.

- Aurora replicates data to the secondary AWS Region **with typical latency of under a second** using a dedicated infrastructure to do the replication.

- If you have an existing Aurora cluster, you can take a snapshot and restore it to a new Aurora global database

- You can manually activate the failover mechanism if a cluster in a different AWS Region is a better choice to be the primary cluster.

- This is different from Aurora Replication that will be explained later, so do not confuse the two.

Amazon Aurora with other AWS Services

- You can use Aurora MySQL DB native function or stored procedure to invoke Lambda functions

- Loading data into a table from text files in an Amazon S3 bucket is available for Amazon Aurora MySQL

- It is possible to query data from an Amazon Aurora MySQL DB cluster and save it directly into text files stored in an Amazon S3 bucket.
  - This functionality can be used to skip bringing the data down to the client first, and then copying it from the client to Amazon S3.

AWS AURORA

- Aurora Replication
- Automatic backups and manual snapshots
- Sharing Aurora snapshots
- Aurora Backtrack feature
- Monitoring & Logging
- Multi-Master Clusters

Amazon Aurora MySQL Replication - Across AWS Regions

- An Amazon Aurora MySQL DB cluster can be created as a Read Replica in a different AWS Region than the source DB cluster.
  - This approach can:
    - Improve your disaster recovery capabilities,
    - Allows for scaling read operations into an AWS Region that is closer to your users, and
    - Can make it easier to migrate from one AWS Region to another.
- You can create Read Replicas of both encrypted and unencrypted DB clusters.
  - The Read Replica must be encrypted if the source DB cluster is encrypted.
- For each source DB cluster, you can have up to five cross-region DB clusters that are Read Replicas.
- When creating the Read Replica, Amazon RDS takes a snapshot of the source cluster and transfers the snapshot to the Read Replica region.
- As a DR recovery mechanism, you can promote a read replica in another region to be come a standalone Aurora DB Cluster

Amazon Aurora – Automatic Backup and Manual Snapshots

- Aurora backs up your cluster **volume automatically and retains restore** data for the length of the backup retention period.
  - A backup retention period, from 1 to 35 days, can be specified when a DB cluster is created or modified.
- Aurora backups are **continuous and incremental** so you can quickly restore to any point within the backup retention period.
  - Recover your data to any given time during the retention period by creating **a new Aurora DB cluster** from the backup data that Aurora retains.
- No performance impact or interruption of database service occurs as backup data is being written.
- **Manual snapshots** of the data can also be created for cluster volume to retain a backup beyond the backup retention period,
  - A new DB cluster can be created from these snapshots

Amazon Aurora – Sharing DB Cluster Snapshots

- A **manual** DB cluster snapshot can be shared
- To share an **automated** DB cluster snapshot, create a **manual** DB cluster snapshot by copying the automated snapshot, and then share that copy.
- You can share a manual snapshot with up to 20 other AWS accounts.
- An unencrypted manual snapshot can be shared as public, it will be available to all AWS accounts.
- DB cluster snapshots that have been encrypted "at rest" can be shared,
  - The account this is shared with need to be given access (by sharing) the KMS encryption key used to encrypt the snapshot

Amazon Aurora - Backtrack

- Backtracking "rewinds" the DB cluster to the time specified.
- Backtracking is not a replacement for backing up the DB cluster which allows for restoring it to a point in time.
- It does not rely on restoring from a backup or snapshot
- Backtracking provides the following advantages over traditional backup and restore:
  - You can easily undo mistakes. If you mistakenly perform a destructive action, you can backtrack the DB cluster to a time before the destructive action with minimal interruption of service.
  - You can backtrack a DB cluster quickly.
  - Unlike restoring the DB cluster to a point in time, Backtracking a DB cluster doesn't require a new DB cluster and rewinds the DB cluster in minutes.
  - It allows for exploring earlier data changes.
    - DB cluster can be backtracked back and forth in time to help determine when a particular data change occurred.

Amazon Aurora – Monitoring and Logging

- **Amazon CloudWatch Alarms**
- **AWS CloudTrail Logs**
  - CloudTrail provides a record of actions taken by a user, role, or an AWS service in Amazon Aurora.
- **Enhanced Monitoring**
  - Amazon Aurora provides metrics in real time for the operating system (OS) that the DB cluster runs on.
- **Amazon RDS Performance Insights**
  - Performance Insights expands on existing Amazon Aurora monitoring features to illustrate the database's performance and help analyze any issues that affect it.
- **Database Logs**
  - Database logs can be viewed, downloaded, and watched
- **Amazon Aurora Event Notification**
  - Amazon Aurora uses the Amazon SNS to provide notification when an Amazon Aurora event occurs.

Amazon Aurora DB Clusters – Multi Master Clusters

- You can create Multiple Read/Write nodes in different availability zones
  - This will ensure continuous write operations in case of a primary node or AZ failure
  - For Aurora multi-master clusters, all DB instances have read-write capability.
  - There isn't any failover when a writer DB instance becomes unavailable, because another writer DB instance is immediately available to take over the work of the failed instance.
    - This is referred to as continuous availability, to distinguish it from the high availability (with brief downtime during failover) offered by a single master cluster.
- Multi-master clusters are best suited for segmented workloads, such as for multitenant applications.
- As of AUG 2019, Aurora Multi Master Clusters are generally available in some regions (Expected in Exam by end of 2019)

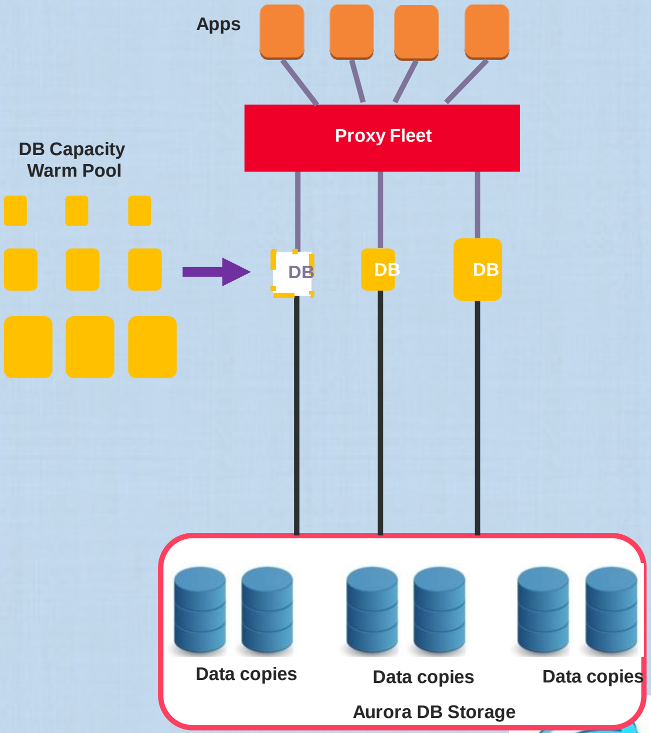
AMAZON AURORA SERVERLESS

AWS AURORA

- Aurora Serverless

Amazon Aurora Serverless – What it is..

- It is an on-demand, autoscaling configuration for Amazon Aurora.
- Aurora Serverless provides a relatively simple, cost effective option for infrequent, intermittent, or unpredictable workloads.
  - ○ It can provide this because it automatically starts up, scales compute capacity to match your application's usage, and shuts down when it's not in use.
- A non-Serverless DB cluster for Aurora is called a **provisioned DB cluster.**
- It has the same kind of high-capacity, distributed, and highly available storage volume as in RDS Aurora.
- You can connect to Aurora Serverless clusters using the TLS/SSL protocol
- Aurora Serverless manages the warm pool of resources in an AWS Region to minimize scaling time.
- When new resources are added to the Aurora DB cluster, Aurora Serverless uses the proxy fleet to switch active client connections to the new resources.
- At any specific time, charges are only for the ACUs that are being used.
- It scales to zero capacity when there are no connections for a 5-minute period (default for Pause feature) actively used in your Aurora DB cluster.



Amazon Aurora Serverless – Use cases

- **Infrequently used applications**
  - Applications, such as **low-volume blog sites**, that are only used for a few minutes several times per day or week.
- **New applications**
  - When a **new application** is being deployed and required instance size is unknown
- **Variable workloads**
  - A lightly used application, with peaks of 30 minutes to several hours a few times each day, or several times per year.
    - Such as **human resources, budgeting, and operational reporting** applications.
- **Unpredictable workloads**
  - When running workloads where there is database usage throughout the day, but also peaks of activity that are hard to predict.
- **Development and test databases**
  - When developers use databases during work hours but don't need them on nights or weekends.
- **Multi-tenant applications**
  - With Aurora Serverless, individually managing database capacity for each application is no longer required.
    - Aurora Serverless manages individual database capacity for you.

Amazon Aurora Serverless – Snapshots

- You can create an Aurora Serverless cluster when restoring from snapshot of an Aurora Provisioned DB cluster

- You can take a snapshot of the Aurora Serverless DB Cluster

- The cluster volume for an Aurora Serverless cluster is always encrypted.
  - You can choose the encryption key, but not turn off encryption.

- To copy or share a snapshot of an Aurora Serverless cluster, you encrypt the snapshot using your own KMS key.

AMAZON ELASTICACHE

Amazon Elasticache

Introduction

Review Topic : AWS Services

ElastiCache

- Is an AWS fully managed web service
- It is an in-memory key value data store engine in the cloud
  - It improves the performance of web applications by allowing for the retrieval of information from a fast, managed, in-memory system (instead of reading from the DB itself)
  - Improves response times for user transactions and queries
  - Can enhance response time for read-intensive And/Or compute-intensive workloads
    - Examples: social networking, gaming, media sharing, Q&A portals
- It offloads the read workload from the main DB instances (less I/O load on the DB)
- It does this by storing the results of frequently accessed pieces of data (or computationally intensive calculations) in-memory
- Integrates with Cloudwatch
- Deployed using EC2 instances

Review Topic : AWS Services

ElastiCache

- Elasticache EC2 nodes deployed can not be accessed from the internet, nor can they be accessed by EC2 instances in other VPCs
- Can be on-demand or Reserved Instances too (NOT Spot instances)
- Access to Elasticache nodes is controlled by VPC security groups and Subnet groups
- You need to configure VPC Subnet groups for Elasticache (VPC that hosts EC2 instances and the Elasticache cluster)
  - Changing the subnet group of an existing Elasticache cluster is not currently supported
- If an Elasticache node fails it is automatically replaced by AWS Elasticache (fully managed service)
- Elasticache nodes are launched in clusters, and can span more than one subnet of the same subnet group which was associated with the cluster when creating it
- Your application connects to your cluster using endpoints.
  - An endpoint is a node or cluster's unique address (use these endpoints rather than the IP addresses in your application
- A cluster can have one or more nodes included within

ElastiCache - Memcached

- Is not persistent
  - Can not be used as a data store
  - If the node fails, the cached data (in the node) is lost
- Ideal front-end for data stores (RDS, DynamoDB…etc)
- Use cases:
  - Cache contents of a DB
  - Cache data from dynamically generated webpages
  - Transient session data
  - High frequency counters for admission control in high volume web Apps
- Does not support Multi-AZ failover, replication, NOR does it support Snapshots for backup/resore
  - Node failure means data loss
- You can, however, place your Memcached nodes in different AZs to minimize the impact of an AZ failure and to contain the data loss in such an incident
  - You can horizontally partition your data across those nodes

Review Topic : AWS Services

ElastiCache - Redis

- Is persistent, using the snapshotting feature.
  - At any time, you can restore your data by creating a new Redis cluster and populating it with data from a backup.
- Use cases: Web, Mobile Apps, Healthcare Apps, Financial Apps, Gaming, Ad-Tech, and IoT
- Supports Redis master/slave replication
- Supports snapshots (automatic and manual) to S3
  - The backup can be used to restore a cluster or to seed a new cluster
  - The backup includes cluster metadata and all data in the cluster
- You can copy your snapshots to other AWS regions (indirectly though)
  - You do this by:
    - First exporting the snapshot from Elasticache to an S3 bucket in the same region
    - Then you copy the exported copy of the snapshot to the destination region
    - This can be handy if you want to seed a new cluster in the other region, rather than waiting to populate a new cluster from the other region's database

Review Topic : AWS Services

ElastiCache – Redis Multi-AZ support

- Multi-AZ is done by creating read replica(s) in another AZ in the same region
- **Clustering mode disabled :** Your Redis cluster can have only one shard
    - One shard can have one read/write primary node and 0-to-5 read only replicas
    - You can distribute the replicas over multiple AZs in the same region
    - Replication from the primary node to the read replica is asynchronous
    - Applications can read from any node in the cluster, but can write to the primary node only
- **Clustering mode enabled:** Your Redis cluster can have up to 15 shards,
    - With the data partitioned across the shards
    - Each shard has one primary node and –5 read only replicas
- Snapshots can slow down your nodes, better take snapshots from the read replicas