Relational Database Service – RDS

- Relational Database Service – RDS is a web service that makes it easier to set up, operate, and scale a relational database in the cloud.
- provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks such as hardware provisioning, database setup, patching, and backups.
- features & benefits
  - CPU, memory, storage, and IOPs can be scaled independently.
  - manages backups, software patching, automatic failure detection, and recovery.
  - automated backups can be performed as needed, or manual backups can be triggered as well. Backups can be used to restore a database, and the restore process works reliably and efficiently.
  - provides Multi-AZ high availability with a primary instance and a synchronous standby secondary instance that can failover seamlessly when a problem occurs.
  - provides elasticity & scalability by enabling Read Replicas to increase read scaling.
  - supports MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server, and the new, MySQL-compatible Aurora DB engine
  - supports IAM users and permissions to control who has access to the RDS database service
  - databases can be further protected by putting them in a VPC, using SSL for data in transit and encryption for data in rest
  - However, **as it is a managed service, shell (root ssh) access to DB instances is not provided**, and this restricts access to certain system procedures and tables that require advanced privileges.

RDS Components

DB Instance

- is a basic building block of RDS
- is an isolated database environment in the cloud
- each DB instance runs a DB engine. AWS currently supports MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server & Aurora DB engines
- can be accessed from AWS command-line tools, RDS APIs, or the AWS Management RDS Console.
- computation and memory capacity of a DB instance is determined by its DB instance class, which can be selected as per the needs
- supports three storage types: Magnetic, General Purpose (SSD), and Provisioned IOPS (SSD), which differ in performance and price

- each DB instance has a DB instance identifier, which is a customer-supplied name and must be unique for that customer in an AWS region. It uniquely identifies the DB instance when interacting with the RDS API and AWS CLI commands.
- each DB instance can host multiple user-created databases or a single Oracle database with multiple schemas.

DB Parameter Groups

- A DB parameter group contains engine configuration values that can be applied to one or more DB instances of the same instance type
- help define configuration values specific to the selected DB Engine *for e.g. max_connections, force_ssl , autocommit*

DB Option Groups

- Some DB engines offer tools or optional features that simplify managing the databases and making the best use of data.
- RDS makes such tools available through option groups *for e.g. Oracle Application Express (APEX), SQL Server Transparent Data Encryption, and MySQL Memcached support.*

RDS Multi-AZ Instance Deployment

- RDS automatically creates a primary DB Instance and synchronously replicates the data to a standby instance in a different AZ.
- RDS performs an automatic failover to the standby, so that database operations can be resumed as soon as the failover is complete.
- RDS Multi-AZ deployment maintains the same endpoint for the DB Instance after a failover, so the application can resume database operation without the need for manual administrative intervention.
- In a Multi-AZ deployment,
  - RDS automatically provisions and maintains a **synchronous standby replica in a different Availability Zone**.
  - Copies of data are stored in different AZs for greater levels of data durability.
  - Primary DB instance is **synchronously replicated** across Availability Zones to a standby replica to provide
    - data redundancy,
    - eliminate I/O freezes during snapshots and backups
    - and minimize latency spikes during system backups.

    - DB instances may have increased write and commit latency compared to a Single AZ deployment, due to the synchronous data replication
    - Transaction success is returned only if the commit is successful both on the primary and the standby DB
    - There might be a change in latency if the deployment fails over to the standby replica, although AWS is engineered with low-latency network connectivity between Availability Zones.
- When using the BYOL licensing model, a license for both the primary instance and the standby replica is required
- For production workloads, it is recommended to use Multi-AZ deployment with Provisioned IOPS and DB instance classes (m1.large and larger), optimized for Provisioned IOPS for fast, consistent performance.
- When Single-AZ deployment is modified to a Multi-AZ deployment (for engines other than SQL Server or Amazon Aurora)
    - RDS takes a snapshot of the primary DB instance from the deployment and restores the snapshot into another Availability Zone.
    - RDS then sets up synchronous replication between the primary DB instance and the new instance.
    - This avoids downtime during conversion from Single AZ to Multi-AZ.
- An existing Single AZ instance can be converted into a Multi-AZ instance by modifying the DB instance without any downtime.

RDS Read Replicas

- RDS Read Replicas provide enhanced performance and durability for RDS.
- RDS Read Replicas allow elastic scaling beyond the capacity constraints of a single DB instance for read-heavy database workloads.
- RDS Read replicas enable increased scalability and database availability in the case of an AZ failure.
- Read Replicas can help reduce the load on the source DB instance by routing read queries from applications to the Read Replica.
- Read replicas can also be promoted when needed to become standalone DB instances.
- RDS read replicas can be Multi-AZ i.e. set up with their own standby instances in a different AZ.
- One or more replicas of a given source DB Instance can serve high-volume application read traffic from multiple copies of the data, thereby increasing aggregate read throughput.
- RDS uses DB engines' built-in replication functionality to create a special type of DB instance called a Read Replica from a source DB instance. It uses the

engines' native **asynchronous replication** to update the read replica whenever there is a change to the source DB instance.
- Read Replicas are eventually consistent due to asynchronous replication.
- RDS sets up a secure communications channel using public-key encryption between the source DB instance and the read replica, even when replicating across regions.
- Read replica operates as a DB instance that allows only read-only connections. Applications can connect to a read replica just as they would to any DB instance.
- Read replicas are available in RDS for MySQL, MariaDB, PostgreSQL, Oracle, and SQL Server as well as Aurora.
- RDS replicates all databases in the source DB instance.

RDS Read Replicas Use Cases

- Scaling beyond the compute or I/O capacity of a single DB instance for read-heavy database workloads, directing excess read traffic to Read Replica(s)
- Serving read traffic while the source DB instance is unavailable *for e.g. If the source DB instance cannot take I/O requests due to backups I/O suspension or scheduled maintenance, the read traffic can be directed to the Read Replica(s).* **However, the data might be stale.**
- Business reporting or data warehousing scenarios where business reporting queries can be executed against a Read Replica, rather than the primary, production DB instance.
- Implementing disaster recovery by promoting the read replica to a standalone instance as a disaster recovery solution, if the primary DB instance fails.

RDS Security

- DB instance can be hosted in a VPC for the greatest possible network access control.
- IAM policies can be used to assign permissions that determine who is allowed to manage RDS resources.
- Security groups allow control of what IP addresses or EC2 instances can connect to the databases on a DB instance.
- RDS supports encryption in transit using SSL connections
- RDS supports encryption at rest to secure instances and snapshots at rest.

RDS Backups, Snapshot

- Automated backups

    - are enabled by default for a new DB instance.
    - enables recovery of the database to any point in time, with database change logs, during the backup retention period, up to the last five minutes of database usage.
- DB snapshots are manual, user-initiated backups that enable backup of the DB instance to a known state, and restore to that specific state at any time.

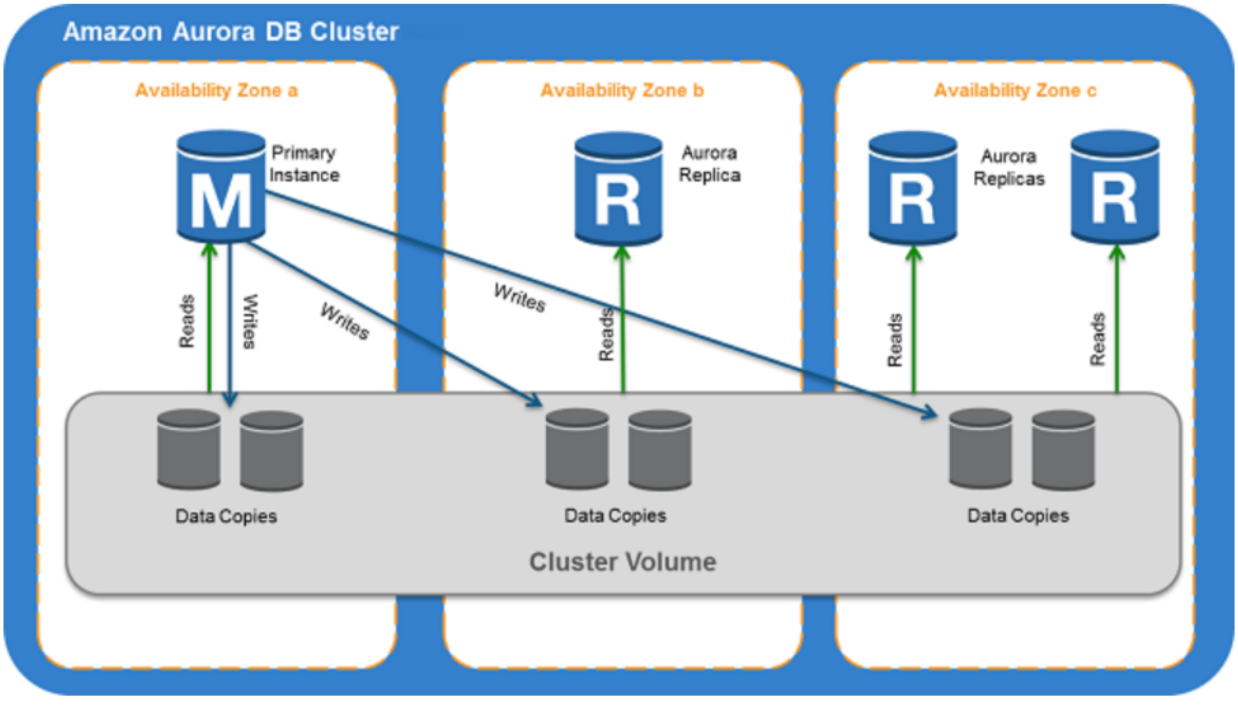
RDS Pricing

- Instance class
    - Pricing is based on the class (e.g., micro) of the DB instance consumed.
- Running time
    - Usage is billed in one-second increments, with a minimum of 10 mins.
- Storage
    - Storage capacity provisioned for the DB instance is billed per GB per month
- I/O requests per month
    - Total number of storage I/O requests made in a billing cycle
- Backup storage
    - Automated backups & any active database snapshots consume storage
- Data transfer
    - Internet data transfer out of the DB instance.
- Reserved Instances
    - In addition to regular RDS pricing, reserved DB instances can be purchased

AWS RDS Aurora

- AWS RDS Aurora is a relational database engine that combines the speed and reliability of high-end commercial databases with the simplicity and cost-effectiveness of open-source databases.
- is a fully managed, MySQL- and PostgreSQL-compatible, relational database engine i.e. applications developed with MySQL can switch to Aurora with little or no changes.
- delivers up to 5x the performance of MySQL and up to 3x the performance of PostgreSQL without requiring any changes to most MySQL applications
- is fully managed as RDS manages the databases, handling time-consuming tasks such as provisioning, patching, backup, recovery, failure detection, and repair.
- can scale storage automatically, based on the database usage, from 10GB to 128TiB in 10GB increments with no impact on database performance

Aurora DB Clusters



- Aurora DB cluster consists of one or more DB instances and a cluster volume that manages the data for those DB instances.
- A cluster volume is a virtual database storage volume that spans multiple AZs, with each AZ having a copy of the DB cluster data
- Two types of DB instances make up an Aurora DB cluster:
  - **Primary DB instance**
    - Supports read and write operations, and performs all data modifications to the cluster volume.
    - Each 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.
    - Each DB cluster can have up to 15 Aurora Replicas in addition to the primary DB instance.
    - Provides high availability by locating Replicas in separate AZs
    - Aurora automatically fails over to a Replica in case the primary DB instance becomes unavailable.
    - Failover priority for Replicas can be specified.
    - Replicas can also offload read workloads from the primary DB instance

Aurora Connection Endpoints

- **Cluster endpoint (or writer endpoint)** for a DB cluster connects to the current primary DB instance for that DB cluster.
- Cluster endpoint is the only one that can perform write operations such as DDL statements as well as read operations
- Each DB cluster has one cluster endpoint and one primary DB instance
- Cluster endpoint provides failover support for read/write connections to the DB cluster. If a DB cluster's current primary DB instance fails, Aurora automatically fails over to a new primary DB instance.
- During a failover, the DB cluster continues to serve connection requests to the cluster endpoint from the new primary DB instance, with minimal interruption of service.
- Reader endpoint for a DB cluster provides load-balancing support for read-only connections to the DB cluster.
- Use the reader endpoint for read operations, such as queries.
- Reader endpoint reduces the overhead on the primary instance by processing the statements on the read-only Replicas.
- Each DB cluster has one reader endpoint.
- If the cluster contains one or more Replicas, the reader endpoint load balances each connection request among the Replicas.

High Availability and Replication

- Aurora is designed to offer greater than 99.99% availability
- provides data durability and reliability
  - by **replicating the database volume six ways across three Availability Zones in a single region**
  - **backing up the data continuously to S3.**
- **transparently recovers from physical storage failures; instance failover typically takes less than 30 seconds.**
- automatically fails over to a new primary DB instance, if the primary DB instance fails, by either promoting an existing Replica to a new primary DB instance or creating a new primary DB instance
- automatically divides the database volume into 10GB segments spread across many disks. Each 10GB chunk of the database volume is replicated six ways, across three Availability Zones
- **provides self-healing storage. Data blocks and disks are continuously scanned for errors and repaired automatically.**

- Replicas share the same underlying volume as the primary instance. Updates made by the primary are visible to all Replicas.
- **Any Replica can be promoted to become primary without any data loss and therefore can be used for enhancing fault tolerance in the event of a primary DB Instance failure.**
- To increase database availability, 1 to 15 replicas can be created in any of 3 AZs, and RDS will automatically include them in failover primary selection in the event of a database outage

Aurora Failovers

- Aurora automatically fails over, if the primary instance in a DB cluster fails, in the following order:
  - If Aurora Read Replicas are available, promote an existing Read Replica to the new primary instance.
  - If no Read Replicas are available, then create a new primary instance.

If there are multiple Aurora Read Replicas, the criteria for promotion is based on the **priority** that is defined for the Read Replicas.

- Priority numbers can vary from 0 to 15 and can be modified at any time.

Backup and Restore

- Automated backups are always enabled on Aurora DB Instances.
- Backups do not impact database performance.
- Aurora also allows the creation of manual snapshots.
- Aurora automatically maintains 6 copies of the data across 3 AZs and will automatically attempt to recover the database in a healthy AZ with no data loss.
- If in any case, the data is unavailable within Aurora storage,
  - DB Snapshot can be restored or
  - the point-in-time restore operation can be performed to a new instance. The latest restorable time for a point-in-time restore operation can be up to 5 minutes in the past.

Aurora Scaling

- Aurora storage scaling is built-in and will automatically grow, up to 64 TB (soft limit), in 10GB increments with no impact on database performance.
- There is no need to provision storage in advance

- Compute Scaling
  - Instance scaling
    - Vertical scaling of the master instance. Memory and CPU resources are modified by changing the DB Instance class.
    - **scaling the read replica and promoting it to master using forced failover which provides a minimal downtime**
  - Read scaling
    - provides horizontal scaling with up to 15 read replicas
- Auto Scaling
  - Scaling policies to add read replicas with min and max replica count based on scaling CloudWatch CPU or connections metrics condition

Aurora Backtrack

- Backtracking “rewinds” the DB cluster to the specified time
- Backtracking performs in-place restore and does not create a new instance. There is minimal downtime associated with it.

Aurora Serverless

- Amazon Aurora Serverless is an on-demand, autoscaling configuration for the MySQL-compatible and PostgreSQL-compatible editions of Aurora.
- An Aurora Serverless DB cluster automatically starts up, shuts down, and scales capacity up or down based on the application’s needs.
- enables running database in the cloud without managing any database instances.
- provides a relatively simple, cost-effective option for infrequent, intermittent, or unpredictable workloads.
- use cases include
  - Infrequently-Used Applications
  - New Applications – where the needs and instance size are yet to be determined.
  - Variable and Unpredictable Workloads – scale as per the needs
  - Development and Test Databases
  - Multi-tenant Applications