Database Types

- **RDBMS** (= SQL / OLTP): RDS, Aurora – great for joins
- **NoSQL database**: DynamoDB (~JSON), ElastiCache (key / value pairs),
- **Neptune (graphs)** – no joins, no SQL
- **Object Store**: S3 (for big objects) / Glacier (for backups / archives)
- ➢ • **Data Warehouse** (= SQL Analytics / BI): Redshift (OLAP), Athena
- ➢ • **Search**: ElasticSearch (JSON) – free text, unstructured searches
- ➢ • **Graphs**: Neptune – displays relationships between data

What is RDS?

- RDS stands for Relational Database Service
- It's a managed DB service for DB use SQL as a query language.
- It allows you to create databases in the cloud that are managed by AWS
- Postgres, MySQL, MariaDB, Oracle, Microsoft SQL Server
- Aurora (AWS Proprietary database)

Advantage over using RDS versus deploying DB on EC2

- Managed service:
- OS patching level
- Continuous backups and restore to specific timestamp (Point in TimeRestore)!
- Monitoring dashboards
- Read replicas for improved read performance
- Multi AZ setup for DR (Disaster Recovery)
- Maintenance windows for upgrades
- Scaling capability (vertical and horizontal)
- **BUT you can't SSH into your instances**

Read Replicas for read scalability

- Up to 5 Read Replicas
- Within AZ, Cross AZ or Cross Region
- Replication is ASYNC, so reads are eventually consistent
- Replicas can be promoted to their own DB
- Applications must update the connection string to leverage read replicas

RDS Multi AZ (Disaster Recovery)

- SYNC replication
- One DNS name – automatic app failover to standby
- Increase availability
- Failover in case of loss of AZ, loss of network, instance or storage failure
- No manual intervention in apps
- Not used for scaling

RDS Backups

- Backups are automatically enabled in RDS
- **Automated backups:**
  - Daily full snapshot of the database
  - Capture transaction logs in real time
  - ability to restore to any point in time
  - 7 days retention (can be increased to 35 days)
- **DB Snapshots:**
  - Manually triggered by the user
  - Retention of backup for as long as you want

RDS Encryption

- Encryption at rest capability with AWS KMS - AES-256 encryption
- SSL certificates to encrypt data to RDS in flight
- **To enforce SSL:**
- PostgreSQL: rds.force_ssl=1 in the AWS RDS Console (Paratemer Groups)
- MySQL: Within the DB:
- GRANT USAGE ON *.* TO 'mysqluser'@'%' REQUIRE SSL;
- **To connect using SSL:**

  Provide the SSL Trust certificate(can be download from AWS)

  Provide SSL options when connecting to database

RDS Security

- RDS databases are usually deployed within a private subnet, not in a public one
- RDS Security works by leveraging security groups (the same concept as for EC2 instances) – it controls who can communicate with RDS
- IAM policies help control who can manage AWS RDS
- Traditional Username and Password can be used to login to the
- database
- IAM users can now be used too (for MySQL / Aurora)

RDS Takeaways :

- **Your responsibility:**
- Check the ports / IP / security group inbound rules in DB's SG
- In-database user creation and permissions
- Creating a database with or without public access
- Ensure parameter groups or DB is configured to only allow SSL connections
- **AWS responsibility:**
- No SSH access ,No manual DB patching, No manual OS patching, No way to audit the underlying instance

RDS Takeaways :

- Read replicas are used for SELECT only kind of statements (no INSERT, UPDATE, DELETE)
- **IAM Authentication (versus traditional username / password):**
- Works for MySQL, PostgreSQL
- Lifespan of an authentication token is 15 minutes
- Easy to use EC2 Instance Roles to connect to the RDS database