RDS

What is Database?

A database is an organized collection of structured data that is stored electronically in a computer system. It allows for efficient storage, retrieval, and manipulation of information. Databases are crucial for applications and systems to manage and handle data effectively.

There are various types of databases available, each designed to cater to specific data storage and retrieval requirements. Some of the most common types include:

1. **Relational Databases (SQL databases):** These databases store data in tables with rows and columns and use Structured Query Language (SQL) for querying and managing data. Examples include:
    - MySQL
    - PostgreSQL
    - Oracle Database
    - Microsoft SQL Server
    - SQLite
2. **NoSQL Databases:** NoSQL (Not Only SQL) databases are designed to handle unstructured, semi-structured, or rapidly changing data. They provide flexibility and scalability and are categorized into different types:
    - **Document-based Databases:** Store data in JSON-like documents. Examples include MongoDB and Couchbase.

```
{
  name: "al",
  age: 18,
  status: "D",
  groups: [ "politics", "news" ]
}
```

- **Key-Value Stores:** Store data in key-value pairs. Examples include Redis and Amazon DynamoDB.


```
{
    "EmployeeId": "701998",
    "Email": "abc@xyz.com",
    "DOB": "12-12-1995",
    "ActiveEmployee": true,
    "Address": [
        {
            "AddressType": "Recidence",
            "AddrLine1": "52-A",
            "AddrLine2": "Anand Colony",
            "AddrLine3": "Ring Road",
            "Landmark": "Near Golkund Square",
            "City": "Nashik",
            "State": "Maharashtra"
        },
        {
            "AddressType": "Permanent",
            "AddrLine1": "Plot No. 43",
            "AddrLine2": "Bluewing City",
            "AddrLine3": "MG Road",
            "Landmark": "Near Raddison Hotel",
            "City": "MHOW",
            "State": "Madhya Pradesh"
        }
    ]
}
```


- **Column Family Stores:** Store data in columns rather than rows. Examples include Cassandra and HBase.
- **Graph Databases:** Designed to handle highly interconnected data. Examples include Neo4j and Amazon Neptune.

3. **Object-Oriented Databases:** These databases store objects rather than data in tables. They are used in object-oriented programming environments. Examples include db4o and ObjectDB.
4. **In-memory Databases:** These databases primarily rely on main memory (RAM) for data storage and retrieval, providing extremely fast data access. Examples include Redis and Memcached.
5. **Cloud Databases:** Databases offered as a service in the cloud, managed by cloud service providers. Examples include Amazon RDS, Microsoft Azure SQL Database, and Google Cloud Spanner.
6. **Time-Series Databases:** Specifically designed for handling time-series data (data points indexed in time order). Examples include InfluxDB and Prometheus.
7. **NewSQL Databases:** Attempt to combine the benefits of traditional SQL databases with NoSQL scalability and performance. Examples include Google Spanner and CockroachDB.

The choice of database type depends on factors such as data structure, volume, scalability needs, performance requirements, and the specific use case or application for which the database is intended. Organizations often use a combination of databases, known as polyglot persistence, to cater to different data storage and retrieval needs within their systems.

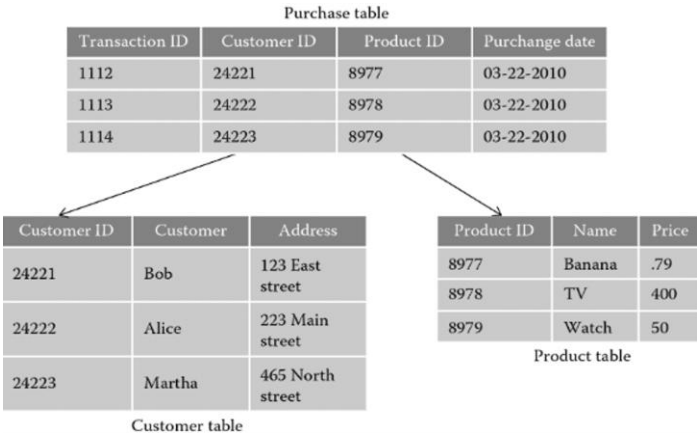
Amazon Relational Database Service (RDS):

Amazon Relational Database Service (RDS) is a cloud-based service provided by Amazon Web Services (AWS) that simplifies the process of setting up, operating, and scaling a relational database in the cloud. RDS supports various popular database engines, including MySQL, PostgreSQL, MariaDB, Oracle, SQL Server, and Amazon Aurora.

Key features of Amazon RDS include:

1. **Ease of Setup:** RDS automates many administrative tasks, such as database setup, patching, backups, and maintenance, reducing the operational overhead for users.
2. **Scalability:** RDS offers scalable storage and compute resources. Users can easily scale their database instance's compute power or storage capacity based on their application requirements.
3. **High Availability:** RDS provides built-in features for high availability, including automated backups, automated failover, and multi-AZ deployments for increased redundancy and fault tolerance.
4. **Security:** RDS offers various security features, including network isolation using Amazon VPC, encryption at rest using AWS KMS, SSL/TLS connections for data in transit, and database access control through IAM roles and database-specific user accounts.
5. **Monitoring and Metrics:** RDS integrates with AWS CloudWatch, allowing users to monitor database performance metrics, set up alarms, and gain insights into database health and performance.
6. **Backup and Restore:** RDS supports automated backups and enables point-in-time recovery, allowing users to restore databases to specific points in time within a retention period.
7. **Compatibility:** RDS supports multiple database engines, making it easy for users to migrate their existing on-premises databases or applications to the AWS cloud.

Amazon RDS is suitable for various use cases, including web applications, mobile apps, e-commerce platforms, analytics, and enterprise applications requiring a reliable and scalable relational database infrastructure. Its managed nature allows developers and businesses to focus more on their applications and less on database administration tasks.



Purchase table

Transaction ID	Customer ID	Product ID	Purchange date
1112	24221	8977	03-22-2010
1113	24222	8978	03-22-2010
1114	24223	8979	03-22-2010

Customer ID	Customer	Address
24221	Bob	123 East street
24222	Alice	223 Main street
24223	Martha	465 North street

Customer table

Product ID	Name	Price
8977	Banana	.79
8978	TV	400
8979	Watch	50

Product table

NoSQL (Not Only SQL):

NoSQL (Not Only SQL) is a term used to refer to a broad category of databases that offer an alternative to traditional relational databases (SQL databases). Unlike relational databases, which use a structured schema and SQL language for querying, NoSQL databases are designed to handle vast amounts of unstructured or semi-structured data and offer greater flexibility and scalability.

Key characteristics of NoSQL databases include:

1. **Schema Flexibility:** NoSQL databases are schema-less or schema-flexible, allowing users to store different types of data without requiring a predefined schema. This makes it easier to handle unstructured, semi-structured, or rapidly evolving data.
2. **Scalability:** NoSQL databases are designed to scale horizontally, allowing them to handle large amounts of data across distributed servers or clusters. They are well-suited for handling Big Data and high-traffic applications.
3. **High Performance:** NoSQL databases often prioritize performance and can handle high volumes of data and high read/write throughput. They are optimized for specific use cases, such as real-time analytics, IoT data, and content management systems.
4. **Types of NoSQL Databases:**
    - **Document-based Databases:** These store data in JSON or BSON-like documents. Examples include MongoDB, Couchbase, and Amazon DocumentDB.
    - **Key-Value Stores:** Data is stored in key-value pairs. Examples include Redis, DynamoDB, and Riak.
    - **Column Family Stores:** Data is stored in columns rather than rows. Examples include Cassandra, HBase, and ScyllaDB.
    - **Graph Databases:** These databases are designed to handle highly interconnected data. Examples include Neo4j and Amazon Neptune.
5. **Use Cases:** NoSQL databases are commonly used in scenarios requiring flexibility, scalability, and real-time data processing, such as web applications, mobile apps, gaming, IoT, social media, and data analytics.
6. **ACID vs. BASE:** While traditional SQL databases follow ACID (Atomicity, Consistency, Isolation, Durability) principles, NoSQL databases often follow BASE (Basically Available, Soft state, Eventually consistent) principles, prioritizing availability and scalability over strong consistency.

It's important to note that the choice between SQL and NoSQL databases often depends on specific use cases, data requirements, scalability needs, and the nature of the application being developed. Organizations may use a combination of both types of databases (polyglot persistence) to meet different data storage and retrieval needs within their systems.

Amazon RDS Overview

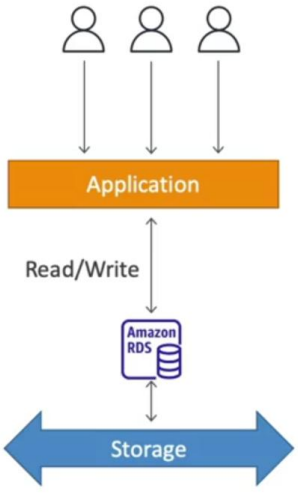
- RDS stands for Relational Database Service
- It's a managed DB service for DB use SQL as a query language.
- It allows you to create databases in the cloud that are managed by AWS
  - Postgres
  - MySQL
  - MariaDB
  - Oracle
  - Microsoft SQL Server
  - Aurora (AWS Proprietary database)

Advantage over using RDS vs Deploying DB on Ec2

- RDS is a managed service:
  - Automated provisioning, OS patching
  - Continuous backups and restore to specific timestamp (Point in Time Restore)!
  - Monitoring dashboards
  - Read replicas for improved read performance
  - Multi AZ setup for DR (Disaster Recovery)
  - Maintenance windows for upgrades
  - Scaling capability (vertical and horizontal)
  - Storage backed by EBS (gp2 or io1)
- BUT you can't SSH into your instances

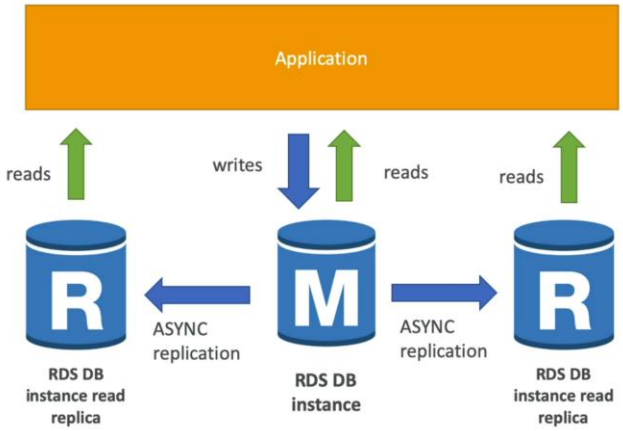
RDS – Storage About Scaling

- Helps you increase storage on your RDS DB instance dynamically
- When RDS detects you are running out of free database storage, it scales automatically
- Avoid manually scaling your database storage
- You have to set **Maximum Storage Threshold** (maximum limit for DB storage)
- Automatically modify storage if:
  - Free storage is less than 10% of allocated storage
  - Low-storage lasts at least 5 minutes
  - 6 hours have passed since last modification
- Useful for applications with **unpredictable workloads**
- Supports all RDS database engines (MariaDB, MySQL, PostgreSQL, SQL Server, Oracle)



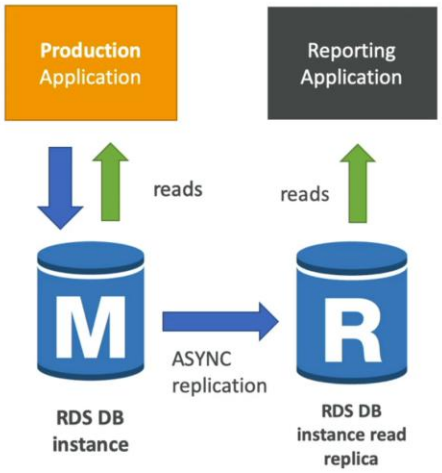
RDS Read Replicas for read scalability:

- Up to 15 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



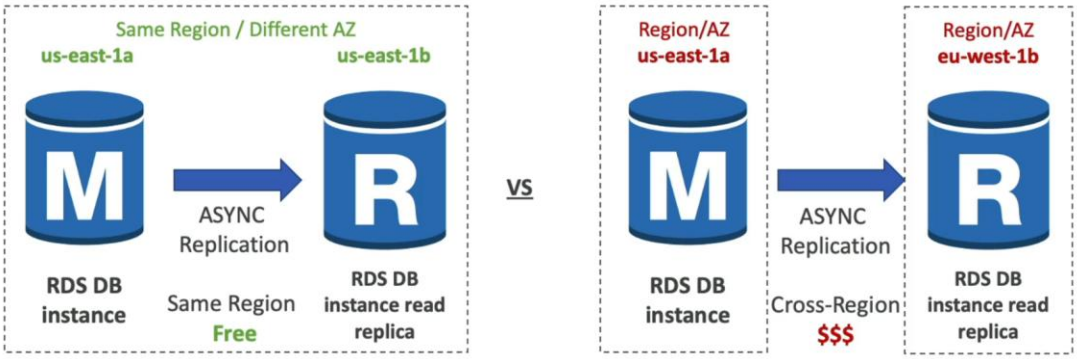
RDS Read Replicas - Use Cases

- You have a production database that is taking on normal load
- You want to run a reporting application to run some analytics
- You create a Read Replica to run the new workload there
- The production application is unaffected
- Read replicas are used for SELECT (=read) only kind of statements (not INSERT, UPDATE, DELETE)



RDS Read Replicas – Network Cost

- In AWS there's a network cost when data goes from one AZ to another
- For RDS Read Replicas within the same region, you don't pay that fee

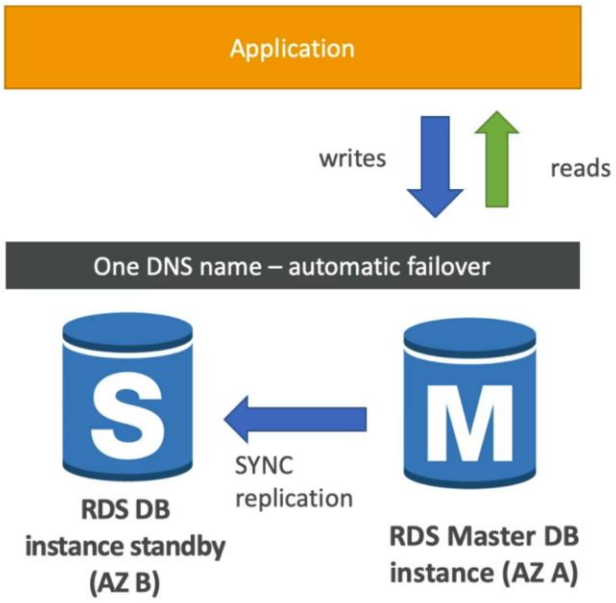


RDS Multi AZ (Disaster Recovery)

- 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

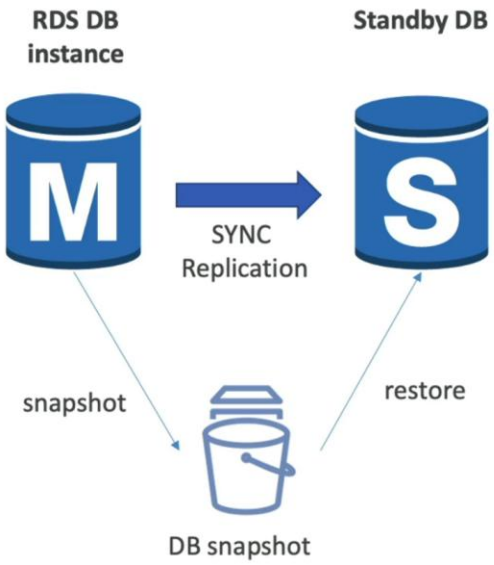
- Note: The Read Replicas be setup as Multi AZ for Disaster Recovery (DR)

Yes



RDS – From Single-AZ to Multi-AZ

- Zero downtime operation (no need to stop the DB)
- Just click on "modify" for the database
- The following happens internally:
  - A snapshot is taken
  - A new DB is restored from the snapshot in a new AZ
  - Synchronization is established between the two databases



Lab:

Crate a database

down the client for connecting the database

https://sqlectron.github.io/