

# AMAZON WEB SERVICES

By – Mr. Ram Rawat





## CHAPTER-5

# AWS Architecture and Database Service

# AWS Well-Architected Framework Design

- The AWS Well-Architected Framework is a guide developed by Amazon Web Services (AWS) to help cloud architects build secure, high-performing, resilient, and efficient application infrastructure.
- It's built on six pillars.
- The framework's design principles are the core philosophies that underpin these pillars.

# AWS Well-Architected Framework Design

## 1. Operational Excellence:

Focus on monitoring and improving processes.

Example: Automating backups of a database instead of doing it manually.

Principle1: Automate your processes

Principle2: Make frequent, small changes

## 2. Security:

Protect data and systems.

Example: Encrypting customer data in a banking app

Principle1: Implement a strong identity foundation

Principle2: Automate security best practices

# AWS Well-Architected Framework Design

## 3. Reliability:

System should work correctly and recover quickly from failures.

Example: If one server goes down, another one should handle traffic.

Principle1: Automatically recover from failure

Principle2: Test recovery procedures

## 4. Performance Efficiency:

Use resources efficiently.

Example: Using the right size of EC2 instance (not too large, not too small).

Principle1: Use the right resource for the right job

Principle2: Democratize advanced technology



# AWS Well-Architected Framework Design

## 5. Cost Optimization:

Avoid unnecessary costs.

Example: Shutting down unused servers at night.

Principle1: Adopt a consumption-based model

Principle2: Analyze and attribute costs

## 6. Sustainability:

Reduce environmental impact.

Example: Using energy-efficient AWS data centers.

Principle1: Use managed services

Principle2: Anticipate and adopt new, more efficient hardware and services

# Relational Database Service(RDS)

- A **relational database** stores data in **tables** (rows and columns).
- Each table has relationships with others using **primary keys and foreign keys**.
- **Querying** is done using **SQL (Structured Query Language)**.

## Example:

- Table Students (Roll\_No, Name, Course)
- Table Results (Roll\_No, Marks)

Here, Roll\_No connects both tables.

# Amazon RDS

- Amazon RDS = Relational Database Service by AWS.
- It is a fully managed service that allows you to **run relational databases in the cloud** without managing hardware and most admin tasks.

Think of it like:

- You focus on your application and queries.
- AWS takes care of infrastructure, backups, updates, scaling, and high availability.



# Challenges Without AWS RDS

Normally, when you run a relational database:

- You must install and configure the database software (MySQL, PostgreSQL, etc.).
- You must manage **backups, security, patching, scaling**.
- You must ensure **high availability** if servers crash.

This requires **time, money, and expertise**.

# Key Features of Amazon RDS

- **Automated Backups** → AWS automatically backs up your database daily.
- **Scalability** → You can increase CPU, memory, or storage easily.
- **Read Replicas** → Improves performance by allowing multiple read-only copies.
- **Security** → Data encryption, VPC isolation.
- **Monitoring**

# Supported Databases in RDS

RDS supports 4 popular relational database engines:

- MySQL
- PostgreSQL
- Microsoft SQL Server
- Amazon Aurora (AWS's own optimized relational database)



# MySQL on RDS

- MySQL = a very popular open-source database.
- It stores data in tables (rows & columns), just like an Excel sheet.
- Normally, you need to download, install, update, and manage MySQL on your computer or server. But with Amazon RDS (Relational Database Service), AWS does all that work for you.
- You just say: “I want a MySQL database”, and AWS creates it in the cloud.
- **MySQL on RDS = MySQL database, but fully managed by AWS (no need to worry about installation, updates, or hardware).**



# PostgreSQL on RDS

- An advanced open-source relational database.
- It is more powerful than MySQL because it supports complex queries, advanced data types (like JSON), and large applications.
- Normally, you would install and manage PostgreSQL on your own server.
- But with Amazon RDS, AWS runs PostgreSQL for you in the cloud.
- Benefits:
  - Fully Managed – AWS handles installation, updates, backups, and scaling.
  - Supports Complex Data – Can store and query advanced data like JSON, GIS (maps/locations), and large datasets.
  - High Availability – AWS automatically switches to a backup database if one fails.
  - Secure & Scalable – Data is encrypted, and AWS makes it easy to grow when users increase



# Microsoft SQL Server on RDS

- Microsoft SQL Server = a commercial relational database developed by Microsoft.
- It is widely used in enterprises for business applications, banking systems, ERP, and corporate data storage.
- Normally, companies install it on Windows servers, which requires a lot of maintenance (patching, licensing, updates).
- With Amazon RDS, AWS runs SQL Server in the cloud and manages everything for you.





# Microsoft SQL Server on RDS - Benefits

- Fully Managed – AWS takes care of software installation, patching, and backups.
- Windows Integration – Works well with Microsoft products like .NET apps, Excel, and Active Directory.
- Scalable – Can handle both small applications and large enterprise workloads.
- Licensing Options – You can use “License Included” (pay through AWS) or “Bring Your Own License” if you already own it.
- Secure & Reliable – AWS provides encryption, multi-AZ failover, and automated monitoring.



# Amazon Aurora (AWS's own optimized relational)

- Aurora is AWS's own relational database, built to be faster and more reliable than MySQL or PostgreSQL.
- It is fully compatible with both MySQL and PostgreSQL, meaning apps built for them can run on Aurora without changes.
- Aurora is cloud-native, meaning it was designed specially for AWS to give high speed, automatic scaling, and strong security.
- AWS claims Aurora is up to 5 times faster than MySQL and 3 times faster than PostgreSQL.



## Amazon Aurora (AWS's own optimized relational database) - Example

- Suppose you are building an **online shopping app** like Amazon Flipkart.
- Thousands of users search products, add to cart, and make purchases at the same time.
- If you use **MySQL on RDS**, it might slow down when traffic is very high.
- But if you use **Amazon Aurora**, it handles **millions of requests quickly**, without downtime.

# NOSQL

- NoSQL databases are different from traditional SQL (relational) databases.
- SQL databases store data in tables with rows and columns (like Excel sheets).
- NoSQL databases store data in a more flexible way – not fixed rows/columns.
- “NoSQL” actually means Not Only SQL → these databases can handle many different data types (structured, semi-structured, unstructured).
- Eg: Suppose you are building a chat application like WhatsApp. Every user has different types of data: name, phone number, profile picture, status, chats, media files. In SQL (relational database), you would need many tables and rigid structure.



# Amazon DynamoDB – NOSQL Server

- DynamoDB is a NoSQL database service provided by AWS.
- Unlike MySQL, PostgreSQL, or Aurora (which are relational databases with tables, rows, and columns), DynamoDB stores data in a key-value or document format.
- It is fully managed, serverless, and super fast, designed for applications that need millisecond response time.
- **Amazon DynamoDB = A fast, serverless NoSQL database on AWS, best for apps needing high speed, scalability, and real-time performance.**



# Key Features

- 1.NoSQL Database – Stores data as items (records) inside tables, but without fixed rows/columns like relational DBs.
- 2.Scalable – Can handle millions of requests per second automatically.
- 3.Serverless – You don't manage servers; AWS does everything behind the scenes.
- 4.Highly Available – Data is automatically stored across multiple AWS regions.
- 5.Flexible Data Model – Can store structured (usernames, emails) and unstructured (JSON documents, metadata) data.



# Amazon Redshift

- Amazon Redshift is AWS's data warehouse service.
- Unlike MySQL, PostgreSQL, or DynamoDB (which are used for day-to-day transactions like storing user data, payments, etc.), Redshift is used for analytics.
- It allows you to analyze huge amounts of data (terabytes to petabytes) very quickly.
- Think of it as a big storage + super-fast calculator for data.



# EXAMPLE

- Imagine a **shopping website** like Flipkart.
- The main database (MySQL/PostgreSQL) handles daily work: user logins, orders, payments.
- But the company also wants to know:
  - “Which products sold the most in 2024?”
  - “Which city buys more electronics?”
  - “What time do users shop the most?”
- These **business analysis questions** need lots of data and complex queries.
- Here, they use **Amazon Redshift** as a **data warehouse** to analyze sales data and generate reports.

Launch an  
Amazon RDS DB instance

# DIGITAL LEARNING CONTENT



**Parul<sup>®</sup> University**



[www.paruluniversity.ac.in](http://www.paruluniversity.ac.in)