Contents

[Relational databases 1](#_Toc68940083)

[Relational Database Types 1](#_Toc68940084)

[RDS (Relational Database Service) 2](#_Toc68940085)

[Automated Backups 2](#_Toc68940086)

[DB Snapshots 2](#_Toc68940087)

[Read Replicas 3](#_Toc68940088)

[Non Relational Databases 4](#_Toc68940089)

[Dynamo DB 4](#_Toc68940090)

[Data Warehousing 5](#_Toc68940091)

[Redshift 5](#_Toc68940092)

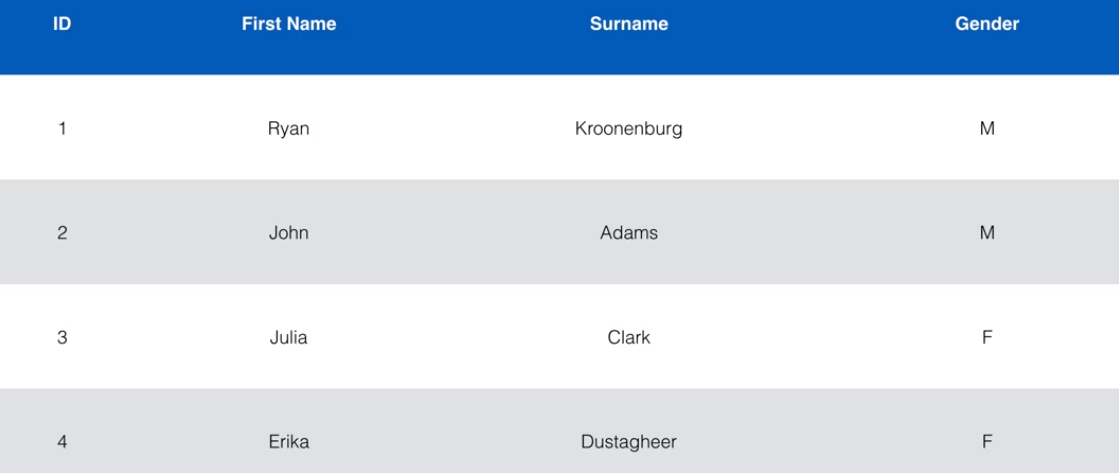
[Elasticache 6](#_Toc68940093)

[**RDS Lab** 8](#_Toc68940094)

[References: 10](#_Toc68940095)

# Relational databases

* Relational Databases are what most of us are all used to. They have been around since the 70’s.Think of a Traditional spreadsheet
  + Database
  + Tables
  + Row
  + Fields (Columns)



## Relational Database Types

* Microsoft SQL Server
* Oracle
* MySQL Server
* PostgreSQL
* Aurora
* MariaDB

# RDS (Relational Database Service)

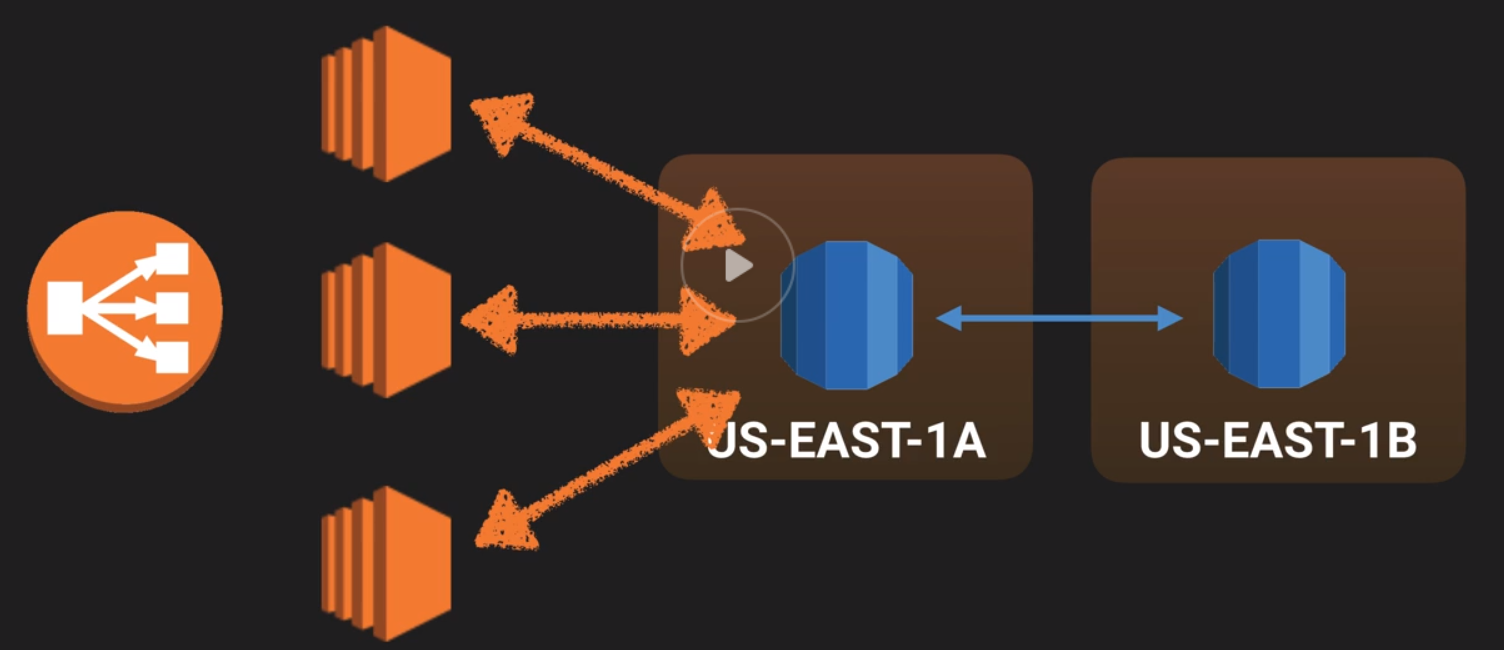
## Automated Backups

* There are two different types of Backups for AWS. Automated Backups and Database Snapshots.
* Automated Backups allow you to recover your database to any point in time within a “retention period". The retention period can be between one and 35 days.
* Automated Backups will take a full daily snapshot and will also store transaction logs throughout the day.
* When you do a recovery, AWS will first choose the most recent daily backup, and then apply transaction logs relevant to that day. This allows you to do a point in time recovery down to a second, within the retention period.
* Automated Backups are enabled by default. The backup data is stored in S3 and you get free storage space equal to the size of your database.so if you have an RDS instance of 10 GB, you will get 10 Gb worth of Storage
* Backups are taken within a defined window. During the backup window, Storage I/O may be suspended while your data is being backed up and you may experience elevated latency

## DB Snapshots

* DB Snapshots are done manually (i.e they are user initiated.) They are stored even after you delete the original RDS instance, unlike automated backups.
* Whenever you restore either an Automatic Backup or a manual snapshot, the restored version of the database will be a new RDS instance with a new DNS endpoint.
* Encryption at Rest is supported for all 6 RDBMS in AWS. Encryption is done using the AWS KMS.
* Once your RDS instance is encrypted, the data stored at rest in the underlying storage is encrypted, as are its automated backups, read replicas, and snapshots.
* At the present time, encrypting an existing DB instance is not supported. To use Amazon RDS encryption for an existing database, you must first create a snapshot, make a copy of that snapshot and encrypt the copy.

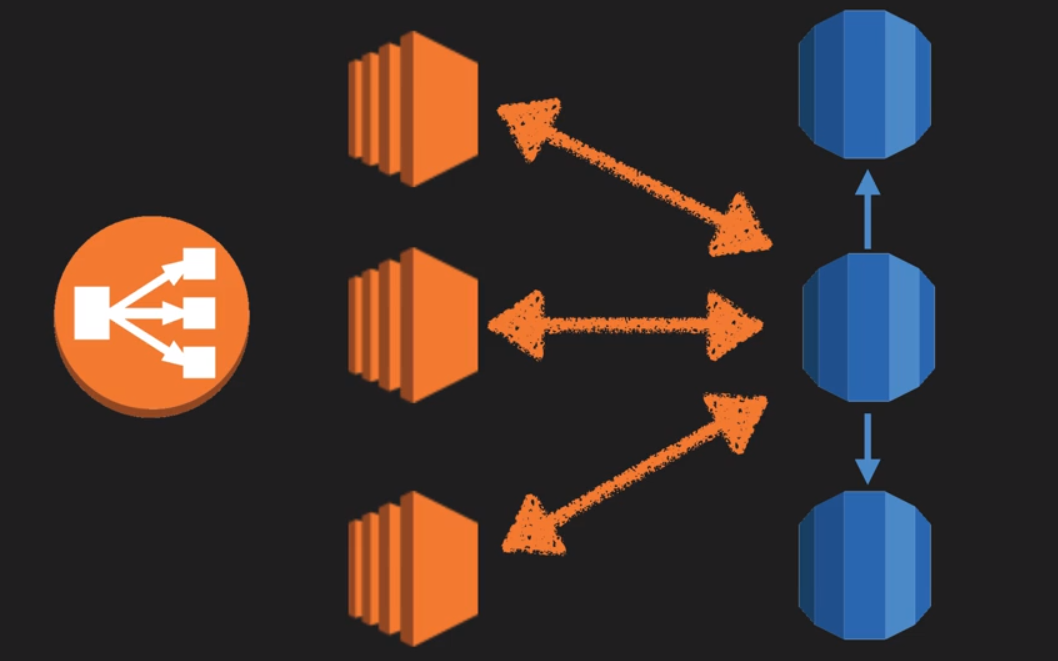
**Multi-AZ**



* Multi-AZ allows you to have an exact copy of your production database in another Availability Zone.
* AWS handles the replication for you, so when your production database is written to, this write will automatically be synchronized to the stand by database.
* In the event of planned database maintenance, DB instance failure or an Availability Zone failure, Amazon RDS will automatically failover to the standby so that database operations can resume quickly without administrative intervention.
* Multi-AZ is for Disaster Recovery only. It is not primarily used for improving performance. For performance improvement, you need Read Replicas.

## Read Replicas

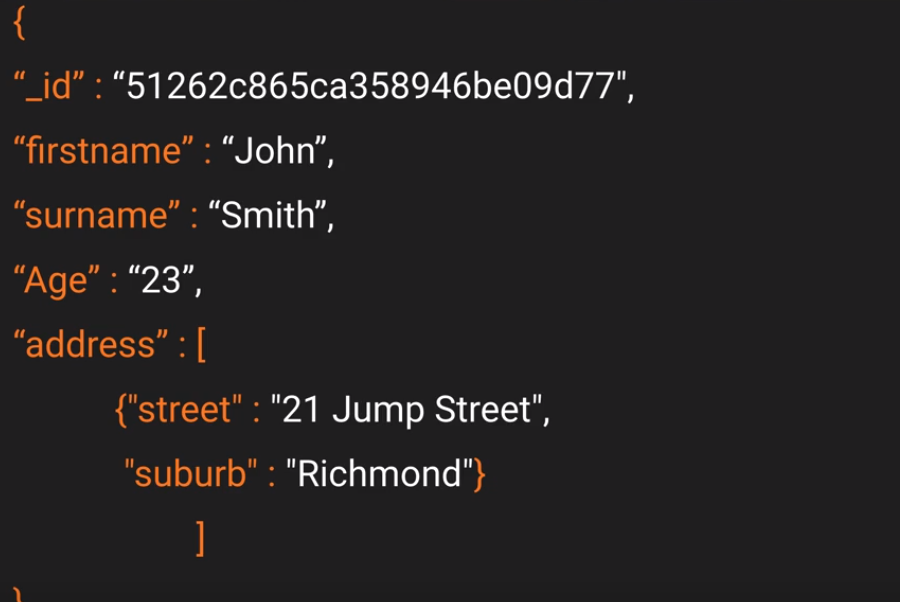
* Read replicas allow you to have a read-only copy of your production database.
* This is achieved by using Asynchronous replication from the primary RDS instance to the read replica.
* You use read replicas primarily for very read-heavy database workloads.



# Non Relational Databases

* Database
  + Collection = Table
  + Document = Row
  + Key Value Pairs = Fields

JSON/NoSQL



## Dynamo DB

* Amazon DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single digit millisecond latency at any scale.
* It is a fully managed database and supports both document and key-value data models.
* Its flexible data model and reliable performance make it a great fit for mobile, web, gaming, ad-tech, IOT and many other applications.
* Stored on SSD Storage
* Spread across 3 geographically distinct data centers
* Eventual consistent reads (default) – consistency across all copies of data is usually reached within a second. Repeating a Read after a short time should return the updated data. (Best Read performance)
* Strongly consistent reads – returns a result that reflects all writes that received a successful response prior to the read.

Components of Dynamo DB

* Tables: Data is stored in tables. Eg: Movie
* Items: Table is collection of Items. Item is group of attributes Eg: {‘id’: ‘1’,’Name’: ‘Avengers’}
* Attribute: fundamental data element Eg: id and Name in above example are attributes
* Primary Key: Two kinds of primary Keys
  + Partition Key: Equivalent to Primary Key
  + Paritition Key & Sort Key: Equivalent to Composite Primary Key
* Just like aurora in mysql/postgres where amazon has rewritten database engines for performance, in the nosql space, amazon has rewritten mongodb engine for better performance that is Document DB
* Provisioned Throughput Capacity
  + Write Throughput $0.0065 per hour for every 10 units.
  + Read Throughput $0.0065 per hour for every 50 units.
  + Storage costs of $0.25Gb per month
* DynamoDB offers “push button” scaling, meaning that you can scale your database on the fly, without any down time.
* Expensive for writes but it is extremely cheaper for Reads

# Data Warehousing

* Used for Business Intelligence. Tools like Cognos, Jaspersoft,SQL Server, Reporting Services, Oracle Hyperion, SAP NetWeaver.
* Used to pull in very large and complex datasets. Usually used by Management to do queries on data (such as current performance vs targets etc )

## Redshift

* Amazon redshift is a peta byte-scale Datawarehouse service in the cloud.
* Redshift is a fast, fully managed data warehouse that makes it simple and cost-effective to analyze all your data using standard SQL and your existing Business Intelligence (BI) tools.
* It allows you to run complex analytic queries against petabytes of structured data, using sophisticated query optimization, columnar storage on high-performance local disks, and massively parallel query execution.
* Designed for online analytical processing (OLAP)
* Do not use for online transaction processing (OLTP)
* Not ideal for small databases < 100 GB
* Not Designed for BLOB Data
* Columnar Data Storage – Instead of storing data as a series of rows, Amazon redshift organizes the data by column.
* Unlike row-based systems, which are ideal for transaction processing, column-based systems are ideal for data warehousing and analytics, where queries often involve aggregates performed over large datasets.
* Since only the columns involved in the queries are processed and columnar data is stored sequentially on the storage media, column-based systems require far fewer I/Os, greatly improving query performance.
* Advanced Compression – It uses best possible compression techniques when loading data into Tables
* Massively Parallel Processing – Redshift automatically distributes data and query load across all nodes. Redshift makes it easy to add nodes to your Data warehouse and enables you to maintain fast query performance as your data warehouse grows
* Customers can start with single node after that slowly scale to multi-node cluster
* If you plan to keep the cluster over a year, reserve compute nodes for 1-3 years
* Regularly create snapshots of the cluster(s)
* You can start with small for just $0.25 per hour with no commitments or upfront costs and scale to a petabyte or more for $1,000 per terabyte per year, less than a tenth of most other data warehousing solutions.
* Configuration
  + Single node (160Gb)
  + Multi-Node – Leader Node (Manages client connections and receives queries)
  + Compute Node – (store data and perform queries and computations). Up to 128 compute nodes.
* Can be used only through AWS Console

# Elasticache

* ElastiCache is a Web service that makes it easy to deploy, operate and scale an in-memory cache in the cloud.
* The service improves the performance of web applications by allowing you to retrieve information from fast, managed, in-memory caches, instead of relying entirely on slower disk-based databases.
* Elasticache supports two open-source in-memory caching engines
  + Memcached
  + Redis
* Can be used to significantly improve latency and throughput for many read-heavy application workloads (such as social networking, gaming,media sharing and Q&A portals) or compute-intensive workloads (such as a recommendation engine).
* Caching improves application performance by storing critical pieces of data in memory for low-latency access.
* Cached information may include the results of I/O-intensive database queries or the results of computationally-intensive calculations.
* **Memcached**
  + A widely adopted memory object caching system.
  + Elasticache is protocol compliant with Memcached, so popular tools that you use today with existing Memcached environments will work seamlessly with the service.
* **Redis**
  + A popular open-source in-memory key-value store that supports data structures such as sorted sets and lists.
  + Elasticache supports Master/Slave replication and Multi-AZ which can be used to achieve cross AZ redundancy.
  + Elasticache is a good choice if your database is particularly read heavy and not prone to frequent changing.
  + Redshift is a Good choice if the reason your database is feeling stress is because management keep running OLAP transactions on it etc

OLAP

OLAP Transaction Example:

Net Profit for EMEA and Pacific for the Digital Radio Product.

Pulls in large numbers of Records

Sum of Radios sold in EMEA

Sum of Radios sold in Pacific

Unit cost of Radio in each Region

Sales price of each Radio

Sales Price – Unit Cost

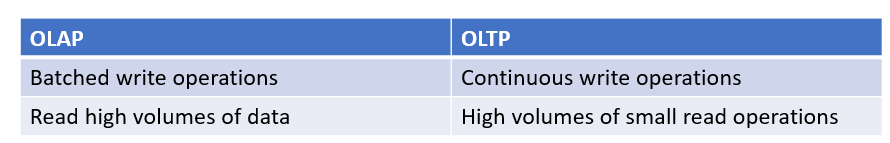
Data Warehousing databases use different type of architecture both from a database perspective and Infrastructure layer.

OLTP vs OLAP

* Online Transaction Processing (OLTP) differs from OLAP Online Analytics Processing in terms of the types of queries you will run.
* OLTP Example:

Order number 2120421

Pull up a row of Data such as Name, Date, Address to deliver to, Delivery status etc.



* RDS – OLTP - Amazon RDS (Relational Database Services) is a Fully Managed Database Services to host
  + Microsoft SQL Server
  + MySQL
  + PostegreSQL
  + Oracle
  + Aurora
  + MariaDB
* DynamoDB – No SQL
* RedShift – OLAP
* Elasticache – In Memory Caching

# **RDS Lab**

* Login to AWS Console
* Search for RDS and click on RDS
* Click on Create Database from Amazon RDS Dashboard
* Select MySQL (Engine Options)
* Templates – Free Tier
* Provide Password & Confirm Password
* Select Public Access Yes
* Accept all other default values
* Click on Create Database
* Launch EC2 Instance
  + Select Redhat Enterprise Linux 8
  + Add below Bash Script

#!/bin/bash

yum install httpd php php-mysqlnd -y

yum update -y

chkconfig httpd on

service httpd start

echo "<?php phpinfo();?>" > /var/www/html/index.php

* Add Inbound Rule to RDS Security Group to allow access from EC2 isntance



* Open the Browser with public IP of RDS EC2 Instance to confirm if apache is installed in EC2 instance
* Open the Browser with publicip/connect.php

**index.php**

**<?php  
  
// Show all information, defaults to INFO\_ALL  
phpinfo();  
  
?>**

**connect.php**

<?php

$username = "admin";

$password = "admin123";

$hostname = "database-1.cotcpwin2ao3.us-east-1.rds.amazonaws.com";

// Create connection

$conn = new mysqli($hostname, $username, $password);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

echo "Connected successfully";

?>

**If it is Redhat or CentOS OS we need to enable**

sudo setsebool -P httpd\_can\_network\_connect\_db=1

# References:

* RDS vs Aurora

<https://www.stratoscale.com/blog/dbaas/aurora-vs-rds/>

* Dynamo Db is used from the code directly

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GettingStarted.html>

* Setting up DynamoDB Local

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/DynamoDBLocal.html>