**Databases on AWS**

**Sections**

1. Database 101
2. Create RDS Instance – Demo
3. RDS Backups. Multi-AZ and Read Replicas
4. RDS Demo
5. DynamoDB
6. Advanced DynamoDB
7. Redshift
8. Aurora
9. Elasticache
10. Database Migration Service (DMS)
11. Caching Strategies on AWS
12. EMR (Elastic Map-Reduce) Overview - BIG Data

**Database 101**

1. Relational Databases on AWS (**6 Engines**)
   1. SQL Server
   2. Oracle
   3. MySQL Server
   4. PostgresSQL
   5. Aurora (Amazon specific MySQL/Postgres compatible) => Aurora has Serverless version
   6. MariaDB
2. RDS Key Features
   1. Multi-AZ
      1. Disaster Recovery (**Failover is Automatic**)
      2. **1 DNS entry**
      3. **2 Availability Zones**
      4. **Synchronous Replication**
   2. Read Replica
      1. For performance
      2. Each has its own DNS entry
      3. Can be **Manually** Promoted to **stand-alone**
      4. **Asynchronous Replication**
      5. **Up to 5 Copies**
      6. Need Automatic Backup Turned on
      7. Read Replica can have Read Replica
3. NoSQL
   1. DynamoDB – Amazon NoSQL => **Serverless**
   2. DocumentDB – MongoDB compatible => **NOT-Serverless**
4. Data Warehousing (OLAP) – **Redshift**
5. ElastiCache => **NOT-Serverless**
   1. Memcached
   2. Redis

**Create RDS Instance – Demo**

1. RDS runs on Virtual Machines
2. You can not login these Virtual Machines
3. Patching of the RDS OS and DB is Amazon responsibility
4. RDS **NOT Serverless** (**Aurora** has serverless version)

**RDS Backups. Multi-AZ and Read Replicas**

1. Backups
   1. Automated Backups (**Checkbox is checked** by default to enable)
      1. Backed up (Storage Volume snapshot for the entire DB instance) daily in the 30 minutes backup window
      2. Retention period – 1 to 35 days
      3. Point in time recovery – May have 5 minutes loss. Use AWS CLI to find the Latest Restorable Time
      4. Backups are stored in S3. You get free storage equal to the instance DB storage or 20GB
      5. During backup, DB may experience elevated latency
   2. Database Snapshots
      1. Done manually
      2. Stores after deletion of RDS
2. Restore – Either restore from an Automatic Backup of Snapshot, the DB will be a new instance
3. Encryption
   1. Available for all RDS types (SQL server, MySQL, Aurora, etc.)
   2. Encryption is done using the AWS KMS
   3. Once RDS instance is encrypted, the data store, automated backups, read replicas, and snapshots are all encrypted
4. Multi-Az (Disaster Recovery)
   1. An exact copy of DB in another AZ
   2. Replication is Synchronous
   3. Automatic failover in case of disasters
   4. Reboot with failover to simulate failover
   5. Multi-AZ is available for SQL Server, Oracle, MySQL, PostgresSQL, MariaDB. **Aurora** is fault tolerant by itself
5. Read Replica (Performance)
   1. Read-only copy of the DB
   2. Replication Asynchronously
   3. Available for all RDS types (SQL server can do in-region read replica)
   4. Must have automatic backup turn on to deploy read replicas
   5. You can have up to 5 read replicas
   6. You can have read replicas of read replicas
   7. Its own DNS
   8. Read replicas can be Multi-AZ
   9. Read replicas can be promoted to stand-alone (Breaks the replication)
   10. Read replicas can be in a different region

**RDS Demo**

**DynamoDB**

1. Amazon’s **NoSQL** Database, Support both **Key-value** & **document data models**
2. **Properties**
   1. **Fast** - Stored on SSD
   2. **Scalable**- Auto scaled is turned on by default
   3. **Fully managed**
   4. **Durability** - Spread across 3 geographically distinct data centers
   5. **Secure –** All data (including backups) in DynamoDB is **Encrypted** at restusing KMS
3. **Consistency**
   1. **Eventual** **Consistent** Reads (**Default**)
      1. 1 second Rule => consistent is usually reached within 1 second
      2. Best reading performance
   2. **Strongly** **Consistent** Reads
      1. Immediately consistent

**Advanced DynamoDB**

1. DynamoDB Accelerator (**DAX**) - **Cache**
   1. Fully managed, highly available (DAX has Replicas for failover in another AZ), in-memory cache
   2. 10X performance improvement
   3. Milliseconds => Microseconds
   4. No need to manage caching logic
   5. Compatible with DynamoDB API calls
2. **On-Demand Capacity** (Pricing)
   1. Default is previsioned capacity
   2. On-demand / Pay-per-request pricing
   3. Balance cost and performance
   4. No minimum capacity
   5. Pay more per request (So if capacity is known, better provision it)
3. Backups – No performance impact on On-demand & Point-in-time
   1. **On-Demand Backup** & Restore
      1. Full backups at any time
      2. Zero impact on table performance or availability
      3. Consistent within seconds and retained until deleted
      4. Operates **within same region** as the source table
   2. **Point-in-time Recovery** (PITR) - **Not** enabled by default
      1. Protects against accidental writes or deletes
      2. Restore to any point in the last 35 days
      3. **Incremental** backups
      4. Latest restorable time **5 minutes**
4. **Transactions**
   1. DynamoDB support transactions
   2. Two underlying reads or writes – prepare / commit
   3. Up to **25 items** or **4 MB** or data
5. **Streams** – Transaction logs equivalent in Relational terminology
   1. Stored for 24 hours
   2. Inserts, updates, and deletes
   3. Combine with Lambda for functionality like stored procedurses
6. **Global Tables** - Managed Multi-master, **Multi-region Replication**
   1. Globally distributed applications
   2. Based on DynamoDB streams – Need to enable streams
   3. Multi-region redundancy for DR (Disaster Recovery) or HA (High Availability)
   4. No application rewrites
   5. Replication latency under 1 second
7. Database Migration Service (**DMS**)
   1. Source database remains operational
   2. **DynamoDB is not a source, but can be used as target** (At this time)
8. Security - **Encryption**
   1. **All data in DynamoDB is encrypted at rest using KMS**
   2. Site-to-site VPN
   3. Direct Connect (DX)
   4. IAM policy and roles
   5. Fine-grained access
   6. Monitor by CloudWatch and CloudTrail
   7. VPC Endpopints

**Redshift**

1. What is Redshift
   1. Amazon Data Warehouse
   2. Scalable – small or large
   3. Large scale - Petabyte
   4. Low cost – 1/10 of most other data warehouse solutions
2. Redshift configurations
   1. Single Node (160G)
   2. Multi-node
      1. Leader Node (Manages client connections and receives queries)
      2. Compute Node (Store data and perform queries and computations). Up to 128 Compute Nodes
3. Advanced Compression
   1. Columnar data store, much better compression compared with relational databases
   2. No need for indexes or materialized views
   3. When loading data into an empty table, compression is automatic for the most appropriate compression scheme
4. Massive parallel computing
   1. Automatically distribute data and query load across all nodes
   2. Redshift makes it easy to add nodes
5. Backups
   1. **Enabled by default with 1 day retention period**
   2. Max retention period is 35 days
   3. At least 3 copies of the data
      1. At original
      2. On compute nodes
      3. A backup in Amazon S3
   4. It can asynchronously replicate the snapshots to S3 in another region for disaster recovery
6. Prices
   1. Compute Nodes Hours – SUM of the total number of hours on all compute nodes
   2. Lead Node is not charged
   3. Backup is charged
   4. Data Transfer is charged (within the VPC)
7. Security
   1. **Always encrypted in transit SSL**
   2. **Always encrypted at rest AES-256**
   3. Redshift takes care of Key Management by default
      1. You have the option to manage the keys through HSM (Hardware Security Module) and AWS KMS (Key Management Service)
8. Availability
   1. Currently only 1 AZ
   2. You can restore snapshot to other AZs in case of outage
   3. It is possible to make cross-region snapshot

**Aurora**

1. What is Aurora
   1. Amazon priority **High-end** Relational Data
   2. Compatible to MySQL & PostgresSQL
   3. Cost effective
2. Simple – Easily migrate from MySQL RDS instance
3. Performance (Fast)
   1. 5 times faster than MySQL
   2. 3 times faster than PostgreSQL
4. Scalable
   1. 10GB – 64TB => Storage AUTO-scaling
   2. Compute resource up to 32vCPSs and 244GB of memory
5. Durability
   1. 2 copies of data contained in each AZ with minimum 3 **AZs**. 6 copies of data
6. Availability
   1. Aurora transparently handles loss of up to 2 copies of data without affecting writing and up to 3 copies of data without affecting reading
   2. Aurora is self-healing. Continuously scanned for error and repaired automatically
7. Security
   1. **Data encryption at rest is an available option on data**, backups, snapshots, and replicas
8. Backups
   1. **Automatic backups are always enabled**. Backups does not have impact on performance
   2. You can take snapshots. It does not have impact on performance
   3. You can share snapshots with other **AWS accounts**
9. Three types of Aurora replicas
   1. Aurora replicas (Currently 15) – In region
   2. MySQL Read Replicas (Currently 5)
   3. PostgresSQL (Currently 1)
   4. Automatic Failover – **Only Available with Aurora Replicas**
10. Aurora has serverless version
    1. A serverless DB cluster automatically starts up / shuts down and scale capacity up and down
    2. Relatively simple, cost effective for infrequent, unpredictable workloads

**Elasticache**

1. What is Elasticache
   1. AWS **In-memory** cache (web service) for **performance**
2. Support 2 open source caching engine
   1. Memcached
      1. Simple / Horizontal scale / Multi-thread
   2. Redis
      1. Advanced data types / ranking sorting / pub & sub capability / Persistence / **Multi-AZ** / **Backup & restore**

**Database Migration Service (DMS)**

1. What is DMS
   1. A service to migrate databases into / from / between AWS environments and On-premise or combinations or other Cloud Providers (Azure for example)
   2. A DMS is a server (Instance) in AWS running the AWS replication software
   3. The DMS software runs the scheduled database migration tasks
2. Migration Types
   1. Homogenous Migration (Same database engine source & target)
   2. Heterogenous Migration (Different database engine source & target)
3. AWS Schema Conversion Tool (SCT)
   1. It is needed by Heterogenous Migration
   2. SCT is needed for creating ALL or part of the target schemas

**Caching Strategies on AWS**

1. What services have caching capabilities
   1. Cloudfront
   2. API Gateway
   3. Elasticache – Memcached and Redis
   4. DynamoDB Accelerator (DAX)

**EMR (Elastic Map-Reduce) Overview - BIG Data**

1. What is Amazon EMR (Elastic Map-Reduce), Amazon uses open source tools
   1. Apache Spark
   2. Apache Hive
   3. Apache HBase
   4. Apache Flink
   5. Apache Hudi
   6. Presto
2. Scale / Speed / Cost
   1. Run Petabyte-scale analysis
   2. 3 times faster than standard Apache Spark
   3. Less than 50% cost of the traditional On-premise solutions
3. EMR Clusters (Collection of EC2 instances)
   1. Master Node - Track state
   2. Core Node - Run tasks and store to HDFS
   3. Task Node (Optional) – Run tasks but does not store to HDFS
   4. Each node can communicate to each other
4. EMR Logs
   1. Logs stored at the Master node
   2. Periodically archive the logs files stored on the master node to Amazon S3, so it won’t get lost for cluster termination, shutdown, or error
   3. EMR archive logs to S3 at **5 minute** interval
   4. You can only set to archive logs at **the first set-up the cluster**

**Port Numbers**

1. SQL Server 1433
2. MySQL 3306
3. Oracle 1521
4. PostgresSQL 5432
5. MariaDB 3306
6. Aurora – Either MySQL or PostgresSQL

**Summary**