AWS Fundamentals: RDS + Aurora + ElastiCache

Uday Manchanda

August 8, 2021

1 RDS

- Relational Database Service
- Managed DB service for DBs that use SQL
- Postgres, MySQL, MariaDB, Oracle, Microsoft SQL
- Using RDS vs deploying DB on EC2
  - Managed service: automated provisioning, OS patching
  - Continous Backups
  - Monitoring dashboards
  - Read replicas
  - Multi AZ setup for DR
  - Scaling capability
  - storage backed by EBS
  - Cannot SSH into your instances
- RDS Backups
  - Backups are automatically enabled in RDS
  - Automated Backups: daily full backup, transaction logs, 7 days retention

1.1 Storage Auto Scaling

- Helps increase storage on RDS DB instance dynamically
- If RDS detects you are running out of free DB storage, it scales automatically
- Useful for apps with unpredictable workloads
- Supports all RDS DB engines

1.2 Read replicas vs Multi AZ

- Read Replicas for read stability
  - Up to 5 read replicas
  - Within AZ, Cross AZ, or Cross Region
  - 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
- RR Use case
  - Prod DB taking on normal load
  - You want to run a reporting application to run some analytics
  - Create RR to run the new workload there
  - Prod DB is unaffected
  - RR are used for SELECT type kind of statements (no INSERT/UPDATE etc)
- RR Network Cost
  - 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
  - ASYNC replication: For example, if your application reads from the RR before they had the chance to to replicate the data then you may get all the data. Called *eventually consistent asynchronous replication*
- RDS Multi AZ (Disaster Recovery)
  - SYNC replication
  - EX: RDS DB in AZ A and RDS DB standby in AZ B
  - When your application writes to the master, that change needs to be replicated to the standby to be accepted
  - One DNS name
  - If there is a problem with master, automatic failover to standby
  - that standby will get promoted to master
  - Not used for scaling
  - Useful for disaster recovery
- RDS from Single AZ to Multi AZ
  - Zero downtime operation
  - click on modify for DB, enable multi AZ

  - What happens internally
    * A snapshot is taken
    * New DB is restored from the snapshot in a new AZ
    * Synchronization is established between the two databases

1.3 Encryption and Security

- At rest encryption: data not in movement
  - Encrypt master and RR with AWS KMS (key management services)
  - Must be defined at launch time
  - If the master is not encrypted, neither can the RRs
- In flight encryption
  - SSL certificates to encrypt data to RDS in flight
  - Provide SSL options with trust certificate when connecting to DB
- Encryption Operations
  - Snapshots of encrypted RDS DBs are encrypted and vice-versa
  - Can copy a snapshot into an encrypted one
  - To encrypt an un-encrypted DB
    * Create snapshot of the un-encrypted DB
    * Copy the snapshot and enable encryption for the snapshot
    * Restore the DB from the encrypted snapshot
    * Migrate applications to the new DB and delete the old one
  - RDS DBs are deployed within private subnet
  - Security works by leveraging security groups
  - Access mgt handled with IAM policies, username/password
  - RDS IAM authentication
    * works with mysql and postgres
    * Don't need a password, just an authentication token obtained through IAM and RDS API calls
    * Token has lifetime of 15 mins
    * Easier to centrally manage users/roles with IAM

2 Aurora

- Proprietary technology from AWS
- Postgres and MySQL are supported
- "Cloud optimized" - better performance than RDS
- Storage grows in 10 gb increments
- Failover is instantaneous. High Availability

2.1 High Availability and Read Scaling

- 6 copies of data across 3 AZ (4 needed for writes, 3 for reads)
- Self healing, striped storage
- One Aurora instance takes writes
- Automated failover very quick
- One master and up to 15 RRs
- Support for Cross Region Replication

2.2 DB Cluster

- Master writes to shared storage volume (which expands)
- Writer endpoint always points to master, useful if autofailing. It's a DNS name
- Even if master fails, the clients can talk to the writer endpoint and is automatically redirected to the right instance
- Can set up autoscaling on your RRs
- Reader endpoint - helps with connection load balancing, connects to all your RRs
- Backtrack: restore data at any point without using backups
- Usually pick either: one writer with multiple readers or serverless (expand/contract based on demand)
- Writers and readers are separate
- You should either select the writer endpoint to write or a reader endpoint to read

2.3 Security

- Encryption at rest with KMS
- In-flight with SSL
- Pretty much the same as RDS

2.4 Advanced Concepts

- Replica autoscaling
  - If there is increased load on the reader endpoints
  - Replica autoscaling will allow more reader endpoints to be created
- Custom Endpoints
  - Define a subset of aurora instances as a custom endpoint
  - EX: have two larger RRs to run analytic queries on
  - Point those RRs to the custom endpoint, not the reader endpoint
  - Set up several custom endpoints for different types of workloads
- Serverless
  - Automated DB instantiation and auto scaling based on actual usage
  - Useful for unpredictable workloads
  - No capacity planning required
- Multi-Master
  - If you want immediate failover for write node (HA)
  - Every node does R/W vs promoting a RR as the new master
- Global Aurora
  - Aurora Global DB: 1 primary region, up to 5 secondary regions, up to 16 RR per secondary region
- Can add ML based predictions to your applications via SQL (SageMaker/Comprehend)

3 ElastiCache

- The way to get managed Redis or memcached
- Caches are in-memory DBs with really high performance, low latency
- Reduces load off DBs
- Makes application stateless
- Involves heavy application code changes

3.1 Solution Architecture

- DB Cache
  - Application queries ElastiCache, which sees if the query has already been made
  - if its already been made - cache hit, gets the answer straight from elasticache
  - if not - cache miss. We need to fetch the data from the DB and then write to elasticache
  - relieves load in RDS
- user session store
  - makes app stateless
  - app writes session data to cache
  - If the user is re-directed, retrieve session from elasticache

3.2 Redis vs Memcached

- Redis
  - Multi AZ with autofailover
  - Read replicas to scale reads and have high availability
  - Data durability
  - Sorta like RDS
- Memcached
  - Multi node for partitioning of data (sharding)
  - no HA, non persistent, no backup
  - Multi-threaded architecture

3.3 Cache Security

- Does not support IAM authentication
- Redis Auth (SSL in flight)

3.4 Patterns

- Lazy loading: all the read data is cached, data can become stale in cache
- Write through: adds or update data in the cache when written to a DB (no stale data)
- Session store: store temp session data in a cache

3.5 Use Cases

- Gaming leaderboard
  - Redis sorted sets guarantee uniqueness and element ordering
  - When a new element is added, its ranked in real time, then added in correct order