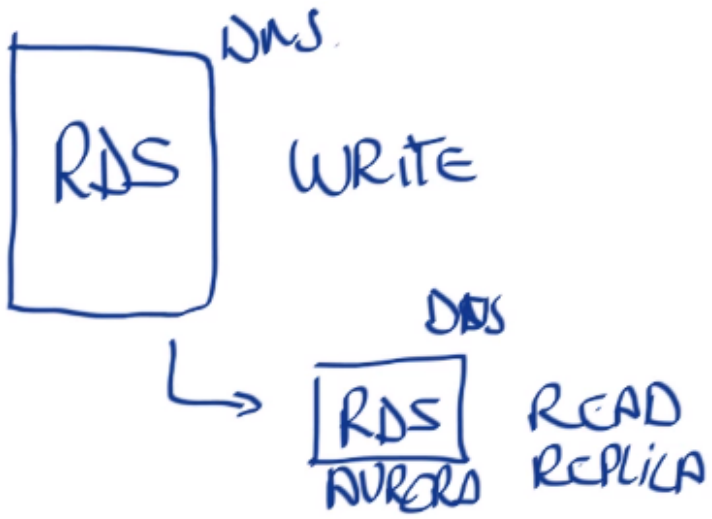
10 - HANDS ON: CRIANDO AURORA DATABASE COM REPLICAS

NÃO É FREETIER



- Vamos montar a instancia principal (main - write) e depois faremos suas replicas (read);

Edition

- (●) Amazon Aurora with MySQL compatibility
- ( ) Amazon Aurora with PostgreSQL compatibility

Capacity type Info

- (●) Provisioned
  You provision and manage the server instance sizes.
- ( ) Serverless
  You specify the minimum and maximum amount of resources needed, and Aurora scales the capacity based on database load. This is a good option for intermittent or unpredictable workloads.

▼ **Replication features** Info

Single-master replication is currently selected

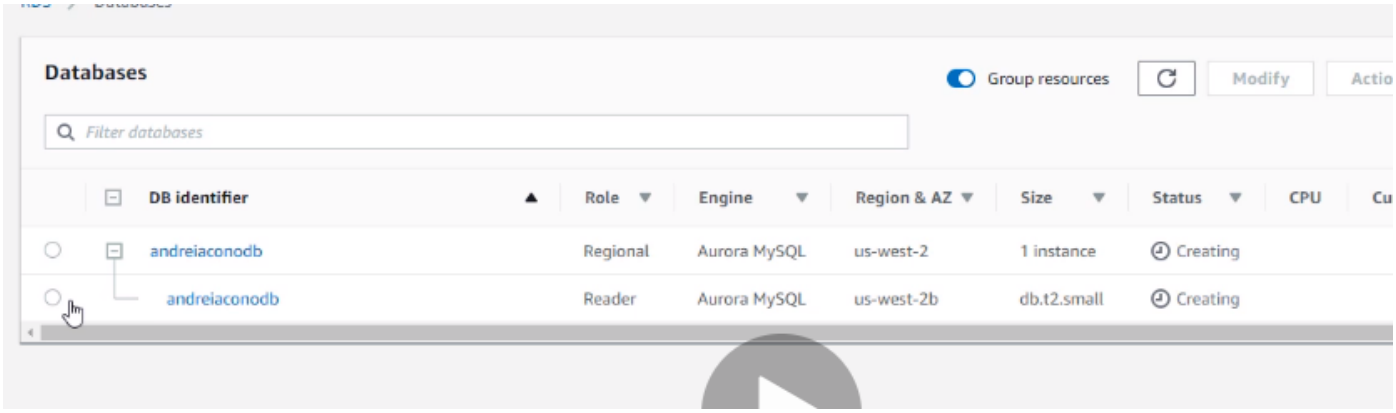
- (●) Single-master
  Supports multiple reader instances connected to the same storage volume as a single writer instance. This is a good general-purpose option for most workloads.
- ( ) Multi-master
  Supports multiple writer instances connected to the same storage volume. This is a good option for when continuous writer availability is required.

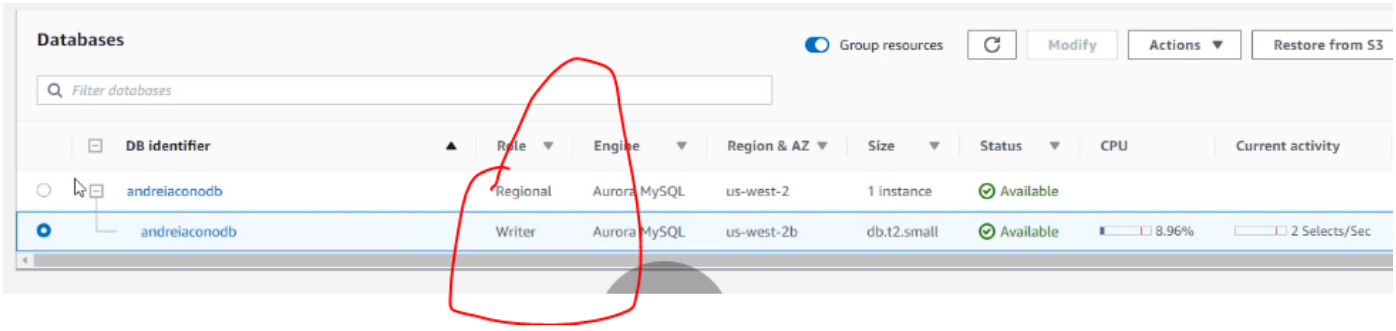
- Precisamos verificar a compatibilidade de versões sempre.

capacity type > serverless , não se preocupa com o tamanho do banco de dados.

FAILOVER

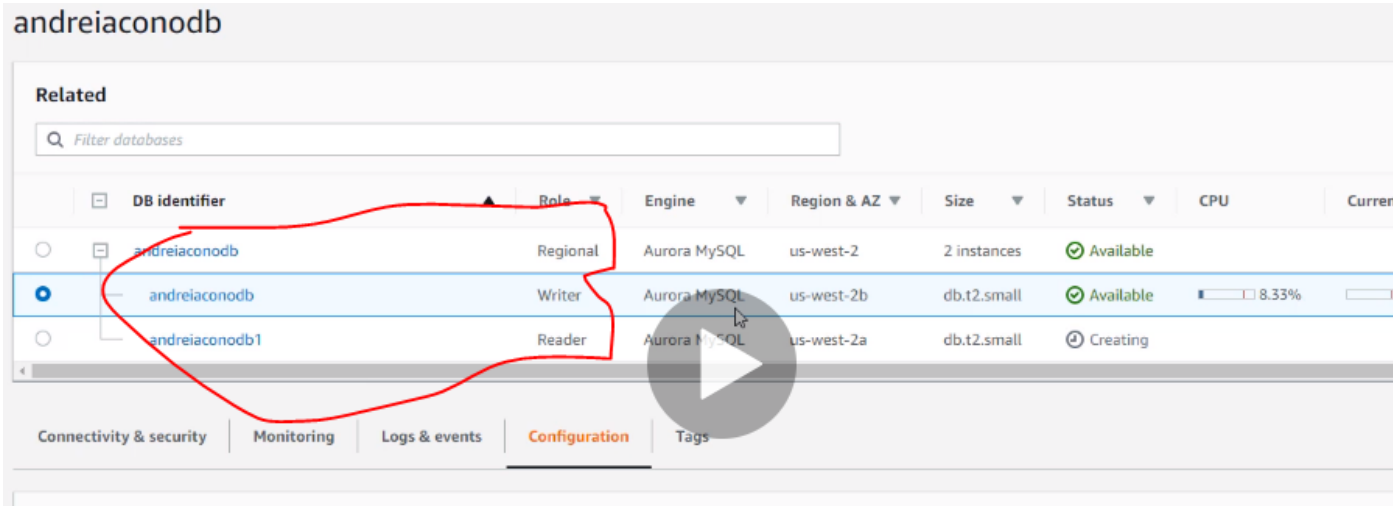
- Temos 15 tier de fail over
    - Se o rds é o principal significa que ele possui o tier 0.
    - Na primeira copia teremos o tier 1.
- Prioridade para assumir a database principal caso a AZ falhe.





- O RDS criou uma instancia e dentro dessa instancia temos uma database, como é a primeira, entra com a opção de writer.

- A criação da replica precisa ser feita no grupo e nao na database em si.


andreiaconodb
Related
Filter databases
DB identifier
Role
Engine
Region & AZ
Size
Status
CPU
andreiaconodb
Regional
Aurora MySQL
us-west-2
2 instances
Available
andreiaconodb
Writer
Aurora MySQL
us-west-2b
db.t2.small
Available
8.33%
andreiaconodb1
Reader
us-west-2a
db.t2.small
Creating
Connectivity & security
Monitoring
Logs & events
Configuration
Tags