Matheus Vinicius Ferreira Figueiredo Teixeira

301236904

# Exercise 1

## Once your configuration is setup correctly take a screenshot showing all the processes and name it “screenshot #1XX” where XX are the last two digits of your student id. (Hint: ps aux | mongod) add the screen shot to your analysis report.

A screenshot of a computer

Description automatically generated

## Retrieve all the inserted data and take a screenshot name it “screenshot #2XX”, where XX are the last two digits of your student id. (hint: use the find method) add the screen shot to your analysis report.

A screenshot of a computer

Description automatically generated

## Check the Sharding status that illustrates the distribution of data; and take a screenshot name it “screenshot #3XX”, where XX are the last two digits of your student id. add the screen shot to your analysis report.

A screenshot of a computer

Description automatically generated

## Please insert all the linux/Mongodb instructions you used, into your analysis report.

I have created shell scripts instead of typing each command which would be a mistake prone approach.

### Script 1

#!/bin/bash

echo "killing mongod and mongos"

sudo killall mongod

sudo killall mongos

echo "removing data files"

sudo rm -rf /data\_matheus/config

sudo rm -rf /data\_matheus/shard\*

################## Shard0

echo "starting servers for shard 0"

sudo mkdir -p /data\_matheus/shard0/rs0 /data\_matheus/shard0/rs1 /data\_matheus/shard0/rs2

sudo mongod --replSet s0 --logpath "s0-r0.log" --dbpath /data\_matheus/shard0/rs0 --port 37004 --fork --shardsvr

sudo mongod --replSet s0 --logpath "s0-r1.log" --dbpath /data\_matheus/shard0/rs1 --port 37005 --fork --shardsvr

sudo mongod --replSet s0 --logpath "s0-r2.log" --dbpath /data\_matheus/shard0/rs2 --port 37006 --fork --shardsvr

echo "Configuring s0 replica set"

sudo mongo --port 37004 << 'EOF'

config = { \_id: "s0", members:[

{ \_id : 0, host : "localhost:37004" },

{ \_id : 1, host : "localhost:37005" },

{ \_id : 2, host : "localhost:37006" }]};

rs.initiate(config)

EOF

################## Shard1

echo "starting servers for shard 1"

sudo mkdir -p /data\_matheus/shard1/rs0 /data\_matheus/shard1/rs1 /data\_matheus/shard1/rs2

sudo mongod --replSet s1 --logpath "s1-r0.log" --dbpath /data\_matheus/shard1/rs0 --port 47004 --fork --shardsvr

sudo mongod --replSet s1 --logpath "s1-r1.log" --dbpath /data\_matheus/shard1/rs1 --port 47005 --fork --shardsvr

sudo mongod --replSet s1 --logpath "s1-r2.log" --dbpath /data\_matheus/shard1/rs2 --port 47006 --fork --shardsvr

echo "Configuring s1 replica set"

sudo mongo --port 47006 << 'EOF'

config = { \_id: "s1", members:[

{ \_id : 0, host : "localhost:47004" },

{ \_id : 1, host : "localhost:47005" },

{ \_id : 2, host : "localhost:47006" }]};

rs.initiate(config)

EOF

################## Config servers

echo "Starting config servers"

sudo mkdir -p /data\_matheus/config/config-a /data\_matheus/config/config-b

sudo mongod --configsvr --dbpath /data\_matheus/config/config-a --replSet conf --port 57015 --fork --logpath "cfg-a.log"

sudo mongod --configsvr --dbpath /data\_matheus/config/config-b --replSet conf --port 57016 --fork --logpath "cfg-b.log"

sudo mongo --port 57015 << 'EOF'

rs.initiate(

{

\_id: "conf",

configsvr: true,

members: [

{ \_id : 0, host : "localhost:57015" },

{ \_id : 1, host : "localhost:57016" }

]

}

)

EOF

################## Mongos (query router)

sudo mongos --configdb "conf/localhost:57015,localhost:57016" --fork --logpath “log.mongos0” --port 57200

################## End of config ########################

############## See processes to take screenshot

ps aux | grep mongo

### Script 2

#!/bin/bash

sudo mongo --port 57200 <<'EOF'

print("Adding shard s0...");

db.adminCommand( { addshard : "s0/"+"localhost:37004" } );

print("Adding shard s1...");

db.adminCommand( { addshard : "s1/"+"localhost:47006" } );

print("Enabling sharding for College04 (Creating DB)");

db.adminCommand({enableSharding: "College04"});

print("Sharding collection College04.students...");

db.adminCommand({shardCollection: "College04.students", key: {student\_id:1}});

print("Switching to database College04...");

use College04;

print("Adding students");

db.students.insert({student\_id:301236904,name:"matheus",age:35})

db.students.insert({student\_id:301236905,name:"gabriel",age:20})

db.students.insert({student\_id:301236906,name:"bernice",age:21})

db.students.insert({student\_id:301236907,name:"apurva",age:22})

db.students.insert({student\_id:301236908,name:"alireza",age:23})

db.students.insert({student\_id:301236909,name:"leor",age:24})

db.students.insert({student\_id:301236910,name:"arunima",age:25})

db.students.insert({student\_id:301236911,name:"jinu",age:26})

db.students.insert({student\_id:301236912,name:"jaykumar",age:27})

db.students.insert({student\_id:301236913,name:"artem",age:28})

print("Printing students");

db.students.find()

EOF

### This part I typed directly in terminal because it was just a small piece of code

sudo mongo --port 57200 <<'EOF'

print("Checking Sharding Status ...");

db.printShardingStatus()

EOF

# Exercise 2

## In this exercise you will research and investigate a NOSQL stores. If your first name starts with a letter between A – M you are assigned to investigate Redis datastore.

### Key features

Redis is an open-source, in-memory data structure store known for its speed and versatility (Redis, n.d.-a). It supports various data structures, including strings, hashes, lists, sets, sorted sets, bitmaps, hyperloglogs, geospatial indexes, and streams (Redis, n.d.-a). Key features include replication, Lua scripting, transactions, and on-disk persistence (Redis, n.d.-a). High availability is provided through Redis Sentinel, and automatic partitioning is enabled with Redis Cluster, making it suitable for scalable, resilient applications (Redis, n.d.-a).

### Type(s) of the datastore

Redis functions primarily as an in-memory key–value database but also serves as a cache and message broker with optional durability, supporting data types like strings, lists, maps, sets, sorted sets, hyperloglogs, bitmaps, streams, and spatial indexes (Wikipedia, n.d.; Redis, n.d.-a). This versatility allows its use in diverse applications, including caching, real-time analytics, and messaging systems.

### Data distribution and Support for fault tolerance

For data distribution, Redis employs sharding and partitioning to distribute data across multiple nodes (Redis, n.d.-a). Redis Cluster provides automatic data partitioning, enhancing scalability and performance by splitting the dataset among multiple nodes (Redis, n.d.-b). Each node can host multiple Redis instances, called "shards," which manage portions of the dataset (Redis, n.d.-c). This approach allows Redis to efficiently handle large datasets and high-throughput workloads.

Redis ensures fault tolerance through replication and automatic failover (Redis, n.d.-a). Redis Sentinel provides high availability by monitoring instances and handling failover if a primary node fails (Redis, n.d.-a). Additionally, Redis supports asynchronous replication with non-blocking synchronization and auto-reconnection to maintain data consistency (Redis, n.d.-a). In Redis Cluster, fault tolerance is achieved by partitioning data across nodes and providing replicas for each master node; if a master fails, a replica takes over (Redis, n.d.-b).

### Some companies that use the store.

Several companies utilize Redis for its speed and scalability. BioCatch uses Redis for real-time fraud detection, managing over 5 billion transactions per month (Raina, 2022). Scopely relies on Redis for leaderboards and workload management to enhance gaming experiences (Raina, 2022). Gap Inc. employs Redis for real-time shipping information, improving customer satisfaction (Raina, 2022). Inovonics uses Redis for fast data ingestion and real-time analytics in IoT devices (Raina, 2022).

**References**

Raina, A. S. (2022, July 12). Redis Use Case Examples for Developers. Retrieved from <https://redis.io/blog/5-industry-use-cases-for-redis-developers/>

Redis. (n.d.-a). Introduction to Redis. Retrieved from <https://redis.io/about/>

Redis. (n.d.-b). Redis Cluster Specification. Retrieved from <https://redis.io/docs/latest/operate/oss_and_stack/reference/cluster-spec/>

Redis. (n.d.-c). Terminology in Redis Enterprise Software. Retrieved from <https://redis.io/docs/latest/operate/rs/references/terminology/>

Wikipedia. (n.d.). Redis. In Wikipedia. Retrieved from <https://en.wikipedia.org/wiki/Redis>