**Advanced Databases**

**MongoDB CA**

**Student Number:** C19366191

**Student Name:** Lina Mir

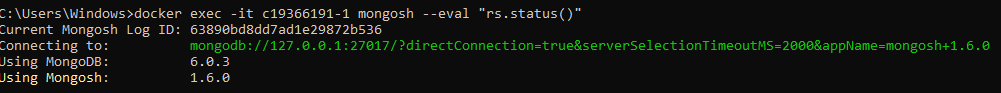
**Programme Code:** TU856

1. **a. Setting up the cluster and replication**

This cluster has three nodes connected to it.

The three nodes are c19366191-1, c19366191-2, c19366191-3

Rs.status() was used on the cluster



Here it shows that it is connecting to MongoDB and which version it is currently using. It also shows the version of Mongosh and the log ID.

Text

Description automatically generated

This shows the different node members in the cluster. This node is called c19366191-1. It is a primary node. This node c19366191-1 is pointing to port number 27017.

Text

Description automatically generated

In this node with an ID of 1, the name of it is c19366191-2 and this is a secondary node. This secondary node is pointing to the primary node, which is 27017.

Below is the third node with an ID of 2 and is a secondary node called c19366191-3. This third node is also pointing to the primary node, 27107.

Text

Description automatically generated

**b**. **Create a replica set**

A replica set was created

docker exec -it c19366191-1 mongosh --eval  
"rs.initiate({\_id: \"C19366191RepSet\", members: [{\_id: 0,  
host: \"c19366191-1\"},{\_id: 1, host: \"c19366191-2\"},{\_id: 2,  
host: \"c-19366191-3\"}]})"

this creates a replica set called C19366191RepSet

**c. Create a database**

A database called c19366191 was created:

A black background with white text

Description automatically generated with low confidence

1. **Porting the data to Mongo**

The python file called C19366191.py converts the data from Cassandra to mongodb.

This is a snip of some of the data that was inserted into the c19366191 database.

It finds the factresult tables data and presents it nicely using pretty().

Text

Description automatically generated

1. **Working with the Golf collection in MongoDB:**
   1. **Basic query on golf data involving a text field.**

This is the query that was run and the output. It shows document information on tournaments with Irish open.

Text

Description automatically generated

This is some of the explain stats output for this query

Text

Description automatically generated

This query had to perform a collection scan and read all the documents because no optimisation has been applied.

This is showing that the mongodb was moving in a forward direction.

There are no rejected plans because there is no optimization applied.

The number of returned documents is 2.

Text

Description automatically generated

Here it just shows that 7 documents were examined and factresults were found. Irish Open was filtered in the name and the database this was done on was c19366191.

* 1. **Adding a secondary index to golf data on a text field.**

The index that was created was below



This index was created on the player’s name.

This is the query that was done after the index was made

Text

Description automatically generated

The output from explain stats on this query is below

Text

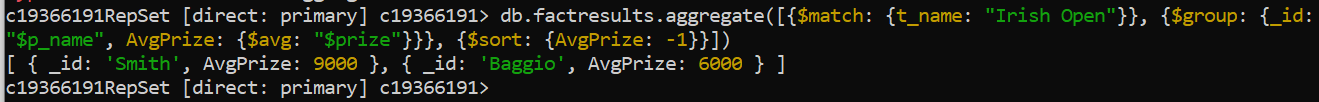
Description automatically generated



This shows that an index scan was done, and the index is called p\_name\_1. The direction is forward, and the rejected plans are nothing again. The docs that were examined are just 1 because the index was on player names, the search was done on player names.

1. **Working with aggregation in MongoDB:**
   1. **Create an aggregation pipeline**

The aggregation that was created is below:



This gets the average prize for the Irish open tournaments.

Text

Description automatically generated

This is the output for the explain. The flow of this aggregate was collection scan, a few projections, and a filter. It returned 2 documents, and the direction is forward.

* 1. **Add relevant indexes and reorder your stages.**

These indexes were created to try and improve the performance.

Text

Description automatically generated

The below screenshot is the output from the aggregate again.

* 1. **Optimize the stage execution**

Text

Description automatically generated

This performed a collection scan, and the stages are detailed here. It performed a few projections and a filter, and it returned 2 rows. The direction on this one was forward. The number of docs that were examined were 7.

1. **Replication working**

The primary node was stopped. This was the c19366191-1 node.

Text

Description automatically generated

From this it can see that once node 1 stopped, the second node, c19366191-2 became a primary node. The first node turned to a secondary node.