MongoDB Task 5 (All Steps)

Overall notes:

rsConfigDB
- — rep1 (port: 33301)

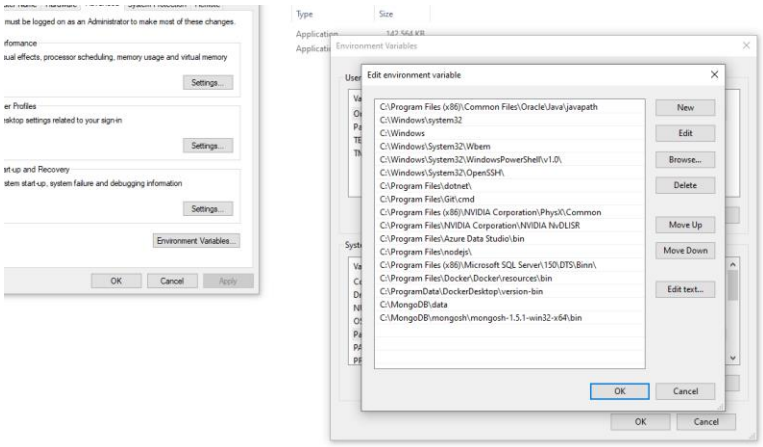
rsShard1
- — rep1 (port: 33311)
- — rep2 (port: 33312)

rsShard2
- — rep1 (port: 33321)
- — rep2 (port: 33322)

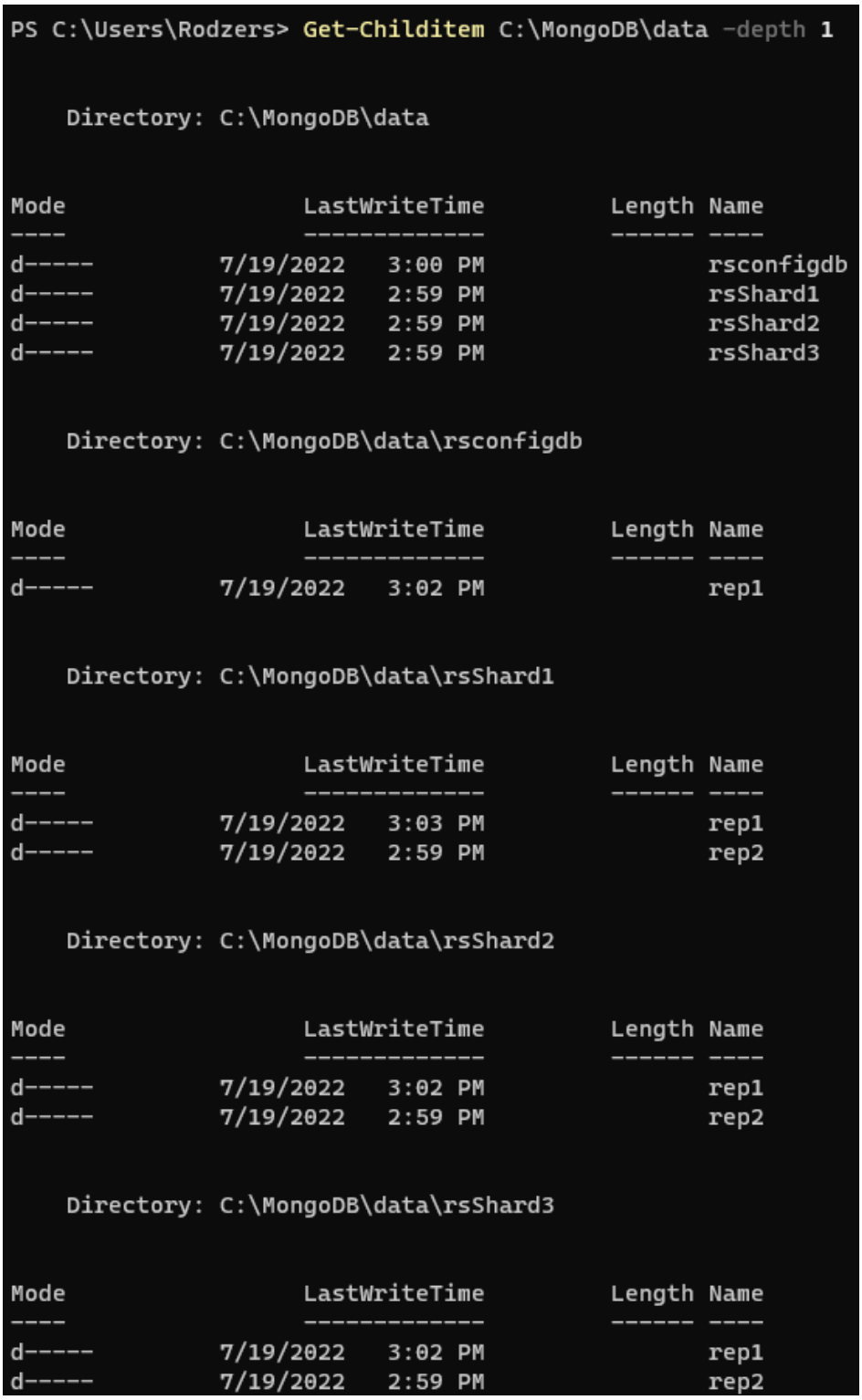
rsShard3
- — rep1 (port: 33331)
- — rep2 (port: 33332)

SQL Script is at the bottom.

1. Added “mongosh” and “mongoDB Server” to the environmental path variables.



2. Created a directory for the database that contains a “rsconfigdb” directory, and three directories for the shards, each containing two folders for replicaSets.



```
PS C:\Users\Rodzers> Get-Childitem C:\MongoDB\data -depth 1


    Directory: C:\MongoDB\data


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         7/19/2022   3:00 PM                rsconfigdb
d-----         7/19/2022   2:59 PM                rsShard1
d-----         7/19/2022   2:59 PM                rsShard2
d-----         7/19/2022   2:59 PM                rsShard3


    Directory: C:\MongoDB\data\rsconfigdb


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         7/19/2022   3:02 PM                rep1


    Directory: C:\MongoDB\data\rsShard1


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         7/19/2022   3:03 PM                rep1
d-----         7/19/2022   2:59 PM                rep2


    Directory: C:\MongoDB\data\rsShard2


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         7/19/2022   3:02 PM                rep1
d-----         7/19/2022   2:59 PM                rep2


    Directory: C:\MongoDB\data\rsShard3


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         7/19/2022   3:02 PM                rep1
d-----         7/19/2022   2:59 PM                rep2
```

3. Created “rsconfigdb” config server and initiated it.

```
PS C:\Users\Rodzers> mongod --configsvr --replSet "configrs" -
dbpath "C:/MongoDB/data/rsconfigdb/rep1" --port 33301
```

```
PS C:\Users\Rodzers> mongosh -host localhost --port 33301
```

```
test> rs.initiate({_id: "configrs", configsvr: true, members:
[{_id: 0, host: "localhost:33301"}]})
```

4. Created two replicas of the “rsShard1” shard server and initiated them.

4.1. rep1.

```
PS C:\Users\Rodzers> mongod --shardsvr --replSet "rsShard1" --
dbpath "C:/MongoDB/data/rsShard1/rep1" --port 33311
```

```
PS C:\Users\Rodzers> mongosh -host localhost --port 33311
```

```
test> rs.initiate({_id: "rsShard1", members: [{_id: 0, host:
"localhost:33311"}]})
```

4.2. rep2.

```
PS C:\Users\Rodzers> mongod --shardsvr --replSet "rsShard1" --
dbpath "C:/MongoDB/data/rsShard1/rep2" --port 33312
```

```
PS C:\Users\Rodzers> mongosh -host localhost --port 33312
```

```
test> rs.initiate({_id: "rsShard1", members: [{_id: 0, host:
"localhost:33312"}]})
```

5. Created two replicas of the “rsShard2” shard server.

5.1. rep1.

```
PS C:\Users\Rodzers> mongod --shardsvr --replSet "rsShard2" --
dbpath "C:/MongoDB/data/rsShard2/rep1" --port 33321
```

```
PS C:\Users\Rodzers> mongosh -host localhost --port 33321
```

```
test> rs.initiate({_id: "rsShard2", members: [{_id: 0, host:
"localhost:33321"}]})
```

5.2. rep2.

```
PS C:\Users\Rodzers> mongod --shardsvr --replSet "rsShard2" --
dbpath "C:/MongoDB/data/rsShard2/rep2" --port 33322
```

```
PS C:\Users\Rodzers> mongosh -host localhost --port 33322
```

```
test> rs.initiate({_id: "rsShard2", members: [{_id: 0, host:
"localhost:33322"}]})
```

6. Created two replicas of the “rsShard3” shard server.

6.1. rep1.

```
PS C:\Users\Rodzers> mongod --shardsvr --replSet "rsShard3" --
dbpath "C:/MongoDB/data/rsShard3/rep1" --port 33331
```

```
PS C:\Users\Rodzers> mongosh -host localhost --port 33331
```

```
test> rs.initiate({_id: "rsShard3", members: [{_id: 0, host:
"localhost:33331"}]})
```

6.2. rep2.

```
PS C:\Users\Rodzers> mongod --shardsvr --replSet "rsShard3" --
dbpath "C:/MongoDB/data/rsShard3/rep2" --port 33332
```

```
PS C:\Users\Rodzers> mongosh -host localhost --port 33332
```

```
test> rs.initiate({_id: "rsShard3", members: [{_id: 0, host:
"localhost:33332"}]})
```

7. Created the router and added the newly created shards to it.

```
PS C:\Users\Rodzers> mongos --configdb rs1/localhost:33301 --port
22222
```

```
PS C:\Users\Rodzers> mongosh --host localhost --port 22222
```

```
[direct: mongos] test> sh.addShard("rsShard1/localhost:33311")
```

```
[direct: mongos] test> sh.addShard("rsShard1/localhost:33312")
```

```
[direct: mongos] test> sh.addShard("rsShard2/localhost:33321")
```

```
[direct: mongos] test> sh.addShard("rsShard2/localhost:33322")
```

```
[direct: mongos] test> sh.addShard("rsShard3/localhost:33331")
```

```
[direct: mongos] test> sh.addShard("rsShard3/localhost:33332")
```

8. Enabled sharding on “flights’ database.

```
[direct: mongos] test> sh.enableSharding("flights")
```

9. Added settings on the config database.

```
[direct: mongos] test> use config
```

Chunksize set to 1MB.

```
[direct: mongos] config> db.settings.insertOne({_id: "chunksize",
value: 1})
```

10. Added the shard key on the **airport name** from which the flight departs.

```
[direct: mongos] config> use flights
switched to db flights

[direct: mongos] flights> sh.enableSharding("flights")

[direct: mongos] flights> db.flights.ensureIndex({dep_airport_name:
1})

[direct: mongos] flights>
sh.shardCollection("flights.flight_collection",
{"dep_airport_name": 1})
```

Checking what I've done

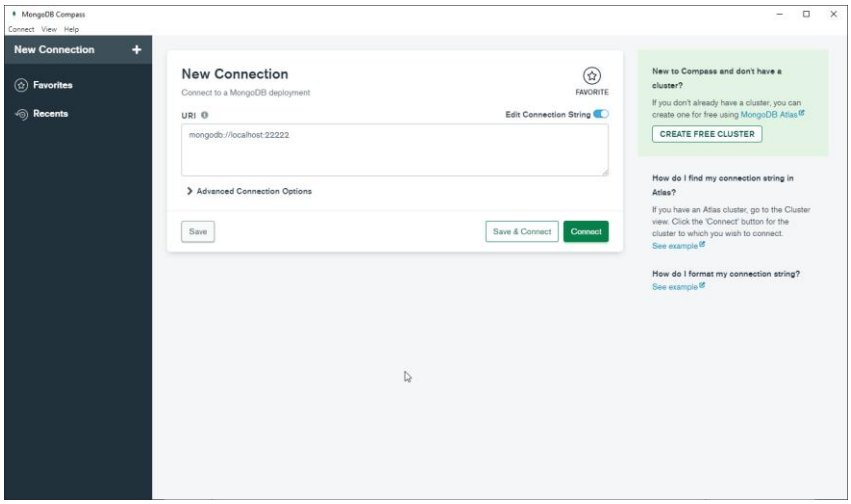
```
[direct: mongos] flights> sh.status()


...
shardingVersion
{
  _id: 1,
  minCompatibleVersion: 5,
  currentVersion: 6,
  clusterId: ObjectId("62d6ad64be5e3af0d9e12a8b")
}
---
shards
[
  {
    _id: 'rsShard1',
    host: 'rsShard1/localhost:33311',
    state: 1,
    topologyTime: Timestamp({ t: 1658238875, i: 4 })
  },
  {
    _id: 'rsShard2',
    host: 'rsShard2/localhost:33321',
    state: 1,
    topologyTime: Timestamp({ t: 1658238958, i: 4 })
  },
  {
    _id: 'rsShard3',
    host: 'rsShard3/localhost:33331',
    state: 1,
    topologyTime: Timestamp({ t: 1658238983, i: 5 })
  }
]
---
active mongoses
[ { '5.0.9': 1 } ]
---
autosplit
{ 'Currently enabled': 'yes' }
---
balancer
{
  'Currently enabled': 'yes',
```

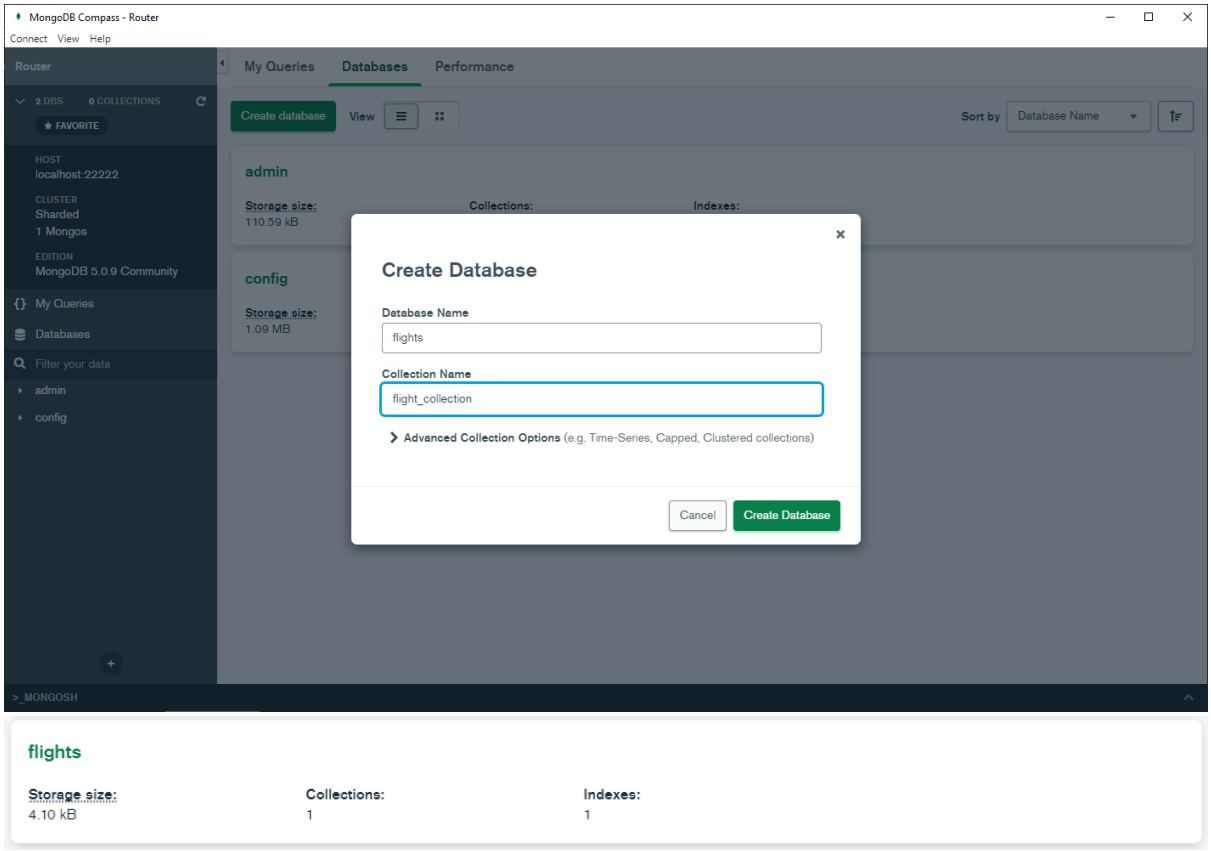
```
  'Currently running': 'no',
  'Failed balancer rounds in last 5 attempts': 0,
  'Migration Results for the last 24 hours': { '682': 'Success' }
}
---
databases
[
  {
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {
      'config.system.sessions': {
        shardKey: { _id: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [
          { shard: 'rsShard1', nChunks: 342 },
          { shard: 'rsShard2', nChunks: 341 },
          { shard: 'rsShard3', nChunks: 341 }
        ],
        chunks: [
          'too many chunks to print, use verbose if you want to force print'
        ],
        tags: []
      }
    }
  },
  {
    database: {
      _id: 'flightdb',
      primary: 'rsShard3',
      partitioned: false,
      version: {
        uuid: UUID("76e9e7a4-c05e-481b-86f3-8451df1661e7"),
        timestamp: Timestamp({ t: 1658248731, i: 16 }),
        lastMod: 1
      }
    },
    collections: {}
  },
  {
    database: {
      _id: 'flights',
      primary: 'rsShard3',
      partitioned: true,
      version: {
        uuid: UUID("a5d1694a-5700-4f6b-b7bb-1a8077f00e12"),
        timestamp: Timestamp({ t: 1658248611, i: 15 }),
        lastMod: 1
      }
    },
    collections: {
      'flights.flight_collection': {
        shardKey: { dep_airport_name: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [ { shard: 'rsShard3', nChunks: 1 } ],
        chunks: [
          { min: { dep_airport_name: MinKey() }, max: { dep_airport_name: MaxKey() }, 'on shard': 'rsShard3', 'last modified': Timestamp({ t: 1, i:
0 }) }
        ],
        tags: []
      }
    }
  }
]

...
```

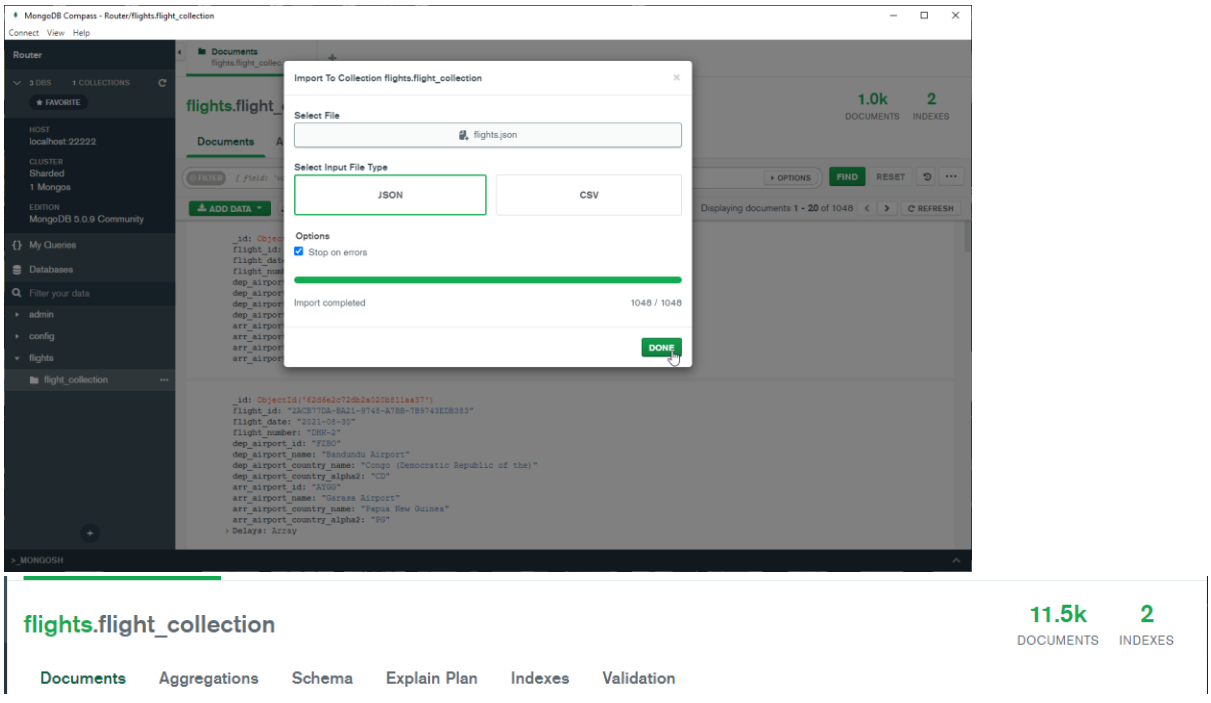
11. Established a connection to the router through MongoDB Compass.



12. Created the flights database with the previously set values. (flights.flight_collection)



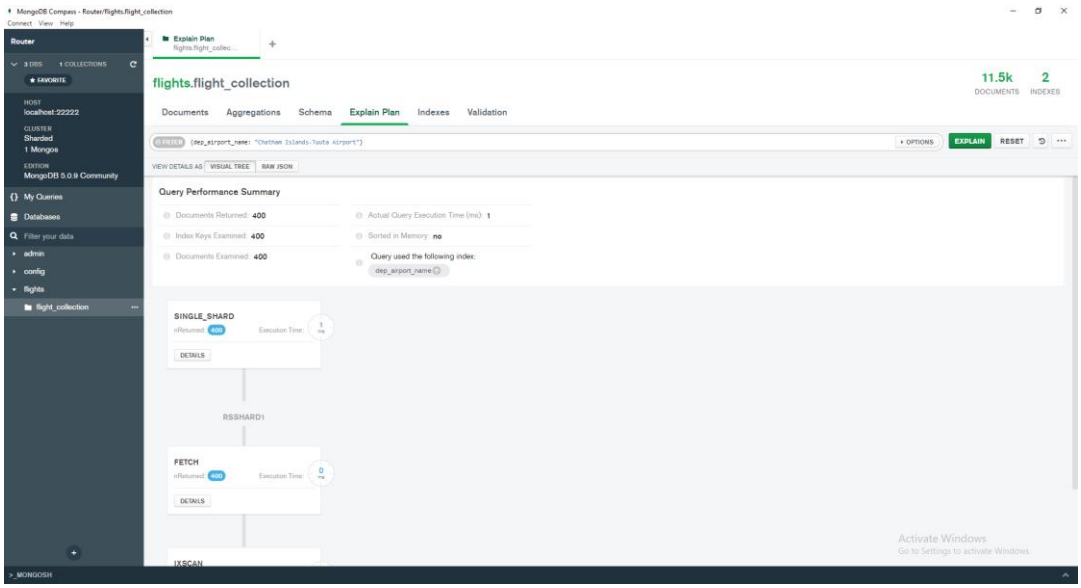
13. Imported my dataset. (Again and again..)

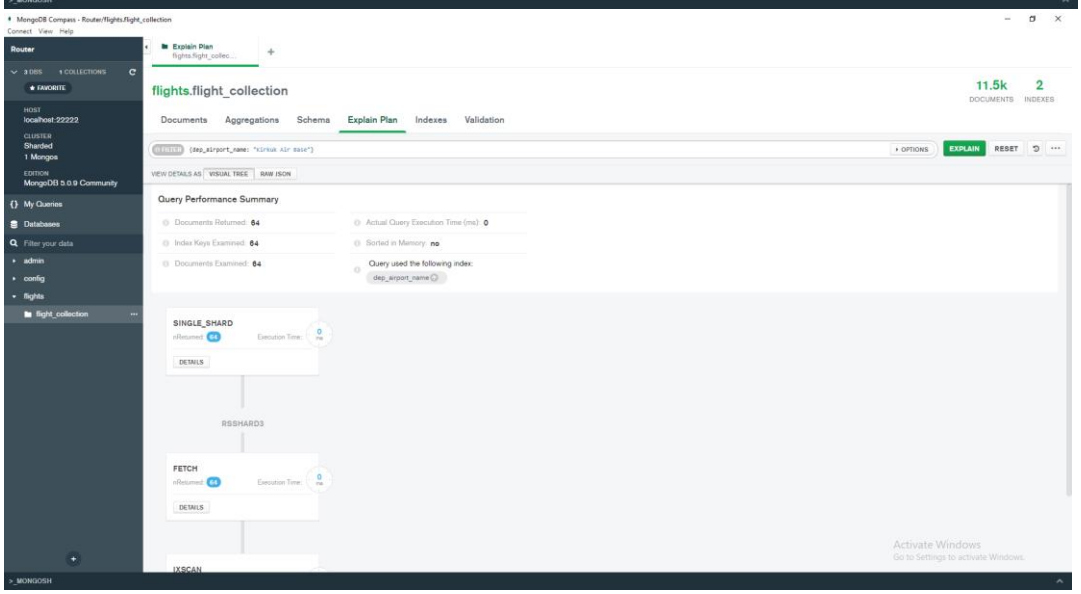


14. Checking the results of my actions.

Ended up having 2 chunks per shard.

Did some queries on “Explain Plan” and everything seems to work correctly.
(First Value = RSSHARD1, Second Value = RSSHARD3)





15. Create a query on the database to select all the flights to a specific airport with at least 1 delay with a specific delay type.

In my opinion there's multiple ways to solve this specific problem since the condition is "at least 1".

```
{ arr_airport_name: "Garasa Airport", "Delays.delay_type_id": 91 }
```

```
{ arr_airport_name: "Garasa Airport" , "Delays.delay_type_id": { $exists: true, $in: [91] } }
```

```
{ arr_airport_name: "Garasa Airport" , "Delays.delay_type_id": { $exists: true, $ne: null, $in: [91] } }
```

SQL Script

```
SELECT

f.id AS flight_id,
f.date AS flight_date,
f.name AS flight_number,
dep_a.id AS dep_airport_id,
dep_a.name AS dep_airport_name,

(SELECT c.name
FROM [CountryInfo].[dbo].[Countries] c
WHERE dep_a.iso_country = c.alpha2Code) AS
dep_airport_country_name,

dep_a.iso_country AS dep_airport_country_alpha2,
arr_a.id AS arr_airport_id,
arr_a.name AS arr_airport_name,

(SELECT c.name
FROM [CountryInfo].[dbo].[Countries] c
WHERE arr_a.iso_country = c.alpha2Code) AS
arr_airport_country_name,

arr_a.iso_country AS arr_airport_country_alpha2,

(SELECT d.delay_type_id, d.minutes
FROM [flightmanagement].[dbo].[delay] d
WHERE d.flight_id = f.id
FOR JSON PATH
) 'Delays'

FROM [flightmanagement].[dbo].[flight] f
INNER JOIN [flightmanagement].[dbo].[airport] dep_a
ON f.departure_id = dep_a.id
INNER JOIN [flightmanagement].[dbo].[airport] arr_a
ON f.arrival_id = arr_a.id
FOR JSON PATH --ROOT('Flights') --,INCLUDE_NULL_VALUES
```