MONGODB

1) Connection :

https://docs.mongodb.com/manual/reference/connection-string/

- Connection string with user and password

```
//      MongoClientURI uri = new
MongoClientURI("mongodb://waction:w@ct10n@localhost:27017/?authSource=local");
//      MongoClient mongoClient = new MongoClient(new MongoClientURI("mongodb://localhost:27017"));

        MongoCredential credential = MongoCredential.createCredential("waction", "admin", "w@ct10n".toCharArray());
        MongoClient mongoClient = new MongoClient(new ServerAddress("localhost", 27017),
Arrays.asList(credential));

        System.out.println("...database names..."+mongoClient.getDatabaseNames());
        System.out.println("...connection point..."+mongoClient.getConnectPoint());
        System.out.println("...address..."+mongoClient.getAddress());
        System.out.println("...all addresses..."+mongoClient.getAllAddress());
        System.out.println("...drop database...");
//        mongoClient.dropDatabase("");
        System.out.println("...list databases..."+mongoClient.listDatabases());
        System.out.println("...set ssl properties...");
//        mongoClient.getMongoClientOptions().builder()
        System.out.println("...used database names..."+mongoClient.getUsedDatabases());


        DB database = mongoClient.getDB("local");
        System.out.println("...collection names..."+database.getCollectionNames());

        DBCollection collection = database.getCollection("startup_log");
        System.out.println("...collection count..."+collection.getCount());
        System.out.println("...collection full name..."+collection.getFullName());
        System.out.println("...collection name..."+collection.getName());
        System.out.println("...collection drop...");
//        collection.drop();
        System.out.println("...drop indexes...");
//        collection.dropIndexes()
        System.out.println("...get indexes...");
        List<DBObject> indexList = collection.getIndexInfo();
        for (DBObject dbObject : indexList) {
            System.out.println(dbObject);
        }
```

Multiple server address for shard or mongos instance

```
MongoCredential credential1 = MongoCredential.createCredential("", "", new char[] {});
MongoCredential credential2 = MongoCredential.createCredential("", "", new char[] {});
MongoCredential credential3 = MongoCredential.createCredential("", "", new char[] {});

ServerAddress server1 = new ServerAddress("192.168.1.2",50001);
ServerAddress server2 = new ServerAddress("192.168.1.2",50002);
ServerAddress server3 = new ServerAddress("192.168.1.2",50003);

MongoClientOptions options = MongoClientOptions.builder().sslEnabled(true).build();

MongoClient mongoClient = new ~~MongoClient~~(Arrays.asList(server1,server2,server3),
                                Arrays.asList(credential1, credential2, credential3),
                                options);
```

2) SSL :

1. MongoClientOptions.builder().sslEnabled(true).build();

   https://stackoverflow.com/questions/42300169/connecting-to-mongodb-with-ssl-from-java-app

3) Case sensitivity :

Collation : https://docs.mongodb.com/manual/reference/collation/

4) Src and target Privilege Checks :

https://docs.mongodb.com/manual/tutorial/manage-users-and-roles/
https://docs.mongodb.com/manual/reference/privilege-actions/

5) Database selection :

mongoClient.listDatabases()

6) Assessment :

Learn more about **sharding**

- **Collection level sharding**
- **Replication Set**

Source Assessment :

- List of databases.
- List of Collections.
- Count and size(Maximum 16MB limit) of documents in a collection
- Column data type.. If possible

- List of indexes (https://docs.mongodb.com/manual/tutorial/manage-indexes/)
- Read only Views(https://docs.mongodb.com/manual/core/views/)
- Change streams for CDC (https://docs.mongodb.com/manual/changeStreams/)

Compatibility :

- Query generation ??

7) Customise screen

8) Database creation

9) DataLoad

- IL
- IL+CDC
- ONLY_CDC

Striim Docs :

- https://www.striim.com/docs/en/mongodb-writer.html

Documents Shared by Ganesh for Reference:

- https://git.striim.com/projects/PRODUCT/repos/product/pull-requests/11363/overview
- https://webaction.atlassian.net/wiki/spaces/WNKB/pages/452886542/Striim+Integration+MongoDBReader+to+CosmosDBWriter

- MongoDBReader
- MongoCosmosDBWriter
- Cosmos-UI design

Sharding :

- How to create the user : https://medium.com/johnjjung/how-to-create-a-user-with-collection-specific-access-in-mongodb-5023c16ccff

- **Setting up sharded cluster in localhost/single machine**

- **Setup config server**

- mkdir configdb
- mongod --configsvr --dbpath configdb --port 27010 --replSet r1
- mongo --port 27010
- rs.initiate();

- **Setup and start the shard servers**
- mkdir datadb1
- mongod --shardsvr --port 50001 --dbpath datadb1 --replSet repl1

- mkdir datadb2
- mongod --shardsvr --port 50002 --dbpath datadb2 --replSet repl2

- **Connect to each shard server and initiate the replSet**
- mongo --port 50001
- rs.initiate();

- mongo --port 50002
- rs.initiate();

- **Setup mongos**
- mongos --configdb r1/localhost:27010 --port 27011

- **Connect to mongos and add shards**
- mongo --port 27011
- sh.addShard("repl1/localhost:50001");
- sh.addShard("repl2/localhost:50002");
- sh.enableSharding('test');
- sh.shardCollection("test.testcollection", { _id : 1 }, true)

- **Insert data**
- db.testcollection.insert({ _id : 1, data : "hellohh" })
- **Split collection**
- sh.splitAt("test.testcollection", { _id : 10})

- **Connect to individual shard servers to see the actual data.**
- mongo --port 50001
- mongo —port 50002
- db.testcollection.find();

- **Check shard status**

- sh.status();

- **References**
- https://docs.mongodb.com/manual/tutorial/deploy-shard-cluster/
- https://sanaulla.info/2015/02/02/setting-up-sharded-mongodb-cluster-in-localhost/

- My References :

Dataset for MongoDB Compass :
https://github.com/ozlerhakan/mongodb-json-files

MongoDB to Azure Cosmos migration Service : useful link for our UI
https://docs.microsoft.com/en-us/azure/dms/tutorial-mongodb-cosmos-db?toc=/azure/cosmos-db/toc.json?toc=/azure/cosmos-db/toc.json

MongoDB Limitations :
https://docs.mongodb.com/manual/reference/limits/

MongoDB Dump :
https://docs.microsoft.com/en-us/azure/cosmos-db/mongodb/tutorial-mongotools-cosmos-db#mongoexportmongoimport

Throughput:
https://willvelida.medium.com/understanding-and-optimizing-throughput-in-azure-cosmos-db-7dd71593c8d3

MongoDB sharding :

References :

1. https://medium.com/the-glitcher/mongodb-sharding-9c5357a95ec1 - Vertical n horizontal scaling - practical example with docker - would be useful for implementation
2. https://medium.com/@tudip/mongodb-sharding-replication-and-clusters-d95a6595bd2c
3. How to install without homebrew -> https://medium.com/nerd-for-tech/how-to-install-mongodb-on-macos-big-sur-d6df6ac69618

   https://www.mongodb.com/download-center/community/releases/archive

4. Tested using -> https://github.com/justmeandopensource/learn-mongodb
5. PR for mongodb-data-sync usage -> https://git.striim.com/projects/PRODUCT/repos/product/commits/21d32b569d7cddd30a610d5f94b7f48d8a98dd4a#Adapters/Sources/MongoDBReader/src/main/java/com/striim/mongodbreader/connection/LocalConnection.java
6. **Java mongodb-driver-sync usage:** http://mongodb.github.io/mongo-java-driver/4.0/driver/tutorials/commands/

Sharding Notes:

1) You can't create a shard collection unless the sharding is enabled for the database.
2) If a collection is inserted with documents before sharding it, all the documents will be stored only in the primary shard server of the shard cluster.
3) If you want to shard a collection with pre-existing documents, we should create an index for the shard key initially before sharding. Even after sharding the collection, all the documents that are present already as well as newly inserted documents will be stored only in the primary shard.
4) mongodb://192.168.1.2:50007,192.168.1.2:50008,192.168.1.2:50009/shard?replicaSet=shard3rs
   If you provide the replicaSet name, it will try to fetch datas from the primary instance. If the PRIMARY instance gets shut down any of its replicas will act as primary.

5) It is possible to get datas only from any one or with the secondary hostnames alone.. But if it goes down, it's not possible to retrieve the data. Good to provide all the replica hostnames and replicaSet names to fetch the datas.
6) If a database is created using mongos and sharding is enabled, it is available on all the shard instances
7) If a collection is created using mongos, it is available on the primary shard instance. On hashed sharding the empty collection, it will be available on all the shard instances.
8) When we do range sharding , it will be available only on the primary shard. Only by splitting up the collection, the documents get stored in any of the other shards based on the shard key. Duplicate documents are available if we use range sharding(It rearranges after sometime...)

- Throughput / bandwidth in networking
- Sharding - horizontal scaling
- MongoDB Replication - replica set
- MongoDB Clusters

MongoDB Cluster contains three separate components

- Config Server (**mongod**) - store metadata and configuration settings for the cluster. used to store the metadata in an organised manner so that it can be retrieved reliably and consistently.
- Query Routers (**mongos**) - interface between client applications and the sharded cluster.
- Shard servers (**mongod**) - Each shard can be deployed as a replica set to provide redundancy and high availability. Together, the cluster's shards hold the entire data set for the cluster.
- Replicas and replica set - A single primary member is used as the base for applying changes to secondary members.

Auth Types : Configuration

- Kerberos GSSAPI -> https://medium.com/hackernoon/mongodb-kerberos-a3dfdf322d1c

JAVA Driver :

- **http://mongodb.github.io/mongo-java-driver/2.13/getting-started/quick-tour/#getting-started-with-java-driver**
- **To test SSL : https://github.com/mongodb/mongo-java-examples**
- **To create Index -**
  - **https://www.bmc.com/blogs/mongodb-indexes/**
  - **https://mongodb.github.io/mongo-java-driver/4.1/driver-reactive/tutorials/indexes/**
  - **https://docs.mongodb.com/drivers/java/sync/v4.3/fundamentals/indexes/**
  - **https://www.quickprogrammingtips.com/mongodb/how-to-get-all-indexes-in-a-mongodb-database-as-a-script.html**
  - **https://www.programcreek.com/java-api-examples/?class=com.mongodb.client.MongoCollection&method=createIndex - example 10**
  - **https://github.com/mongodb/mongo-java-driver/blob/3.6.x/driver-core/src/main/com/mongodb/client/model/Indexes.java**

Cosmos mongodb api indexing -
https://docs.microsoft.com/en-us/azure/cosmos-db/mongodb/mongodb-indexing

Cosmos sql api -
https://docs.microsoft.com/en-us/azure/cosmos-db/index-overview

Points to Discuss:

UI references:

- Cosmos sql api support - https://overflow.io/s/TUABC096?node=04517133

Source :

- Source Tile for MongoDB
- Connection params :
    - Connection URL
        - IL - mongos ipaddress

        Eg: localhost:27017,localhost:27018,localhost:27019
        - IL_CDC / CDC

        Provide a tooltip or info button to give the instance detail of only one shard.
        Eg: localhost:27017
    - UserName
    - Password - if it contains `: / ? # [ ] @` Do percent encoding.
    - authDB
    - authType - SingleSelect
      Allowed values : NoAuth, Default, SCRAMSHA1, MONGODBCR, GSSAPI, PLAIN, MONGODBX509
    - **`authMechanismProperties`**
    - readPreference - primary, primaryPreferred, secondary, secondaryPreferred, nearest

SSL -

- useSSL
- **`tlsCAFile`**
- **`tlsCertificateKeyFile`**
- **`tlsCertificateKeyFilePassword`**

Target :

- Target Tile -

  Disable all other targets except
    - Cosmos sequel api
    - Mongo Cosmos api
- Connection Params for Mongo Cosmos api:

Database selection page :

Source Assessment : think about scoring

- List of selected databases
- collections(Size and count of documents)
- Indexes in collection
- Views

Compatibility :

- Read this for source side checks : https://docs.mongodb.com/manual/reference/limits/
- Read about cosmos mongo api support for target side compatibility

Map Schemas Page :

Customization Page :

Schema migration Page :

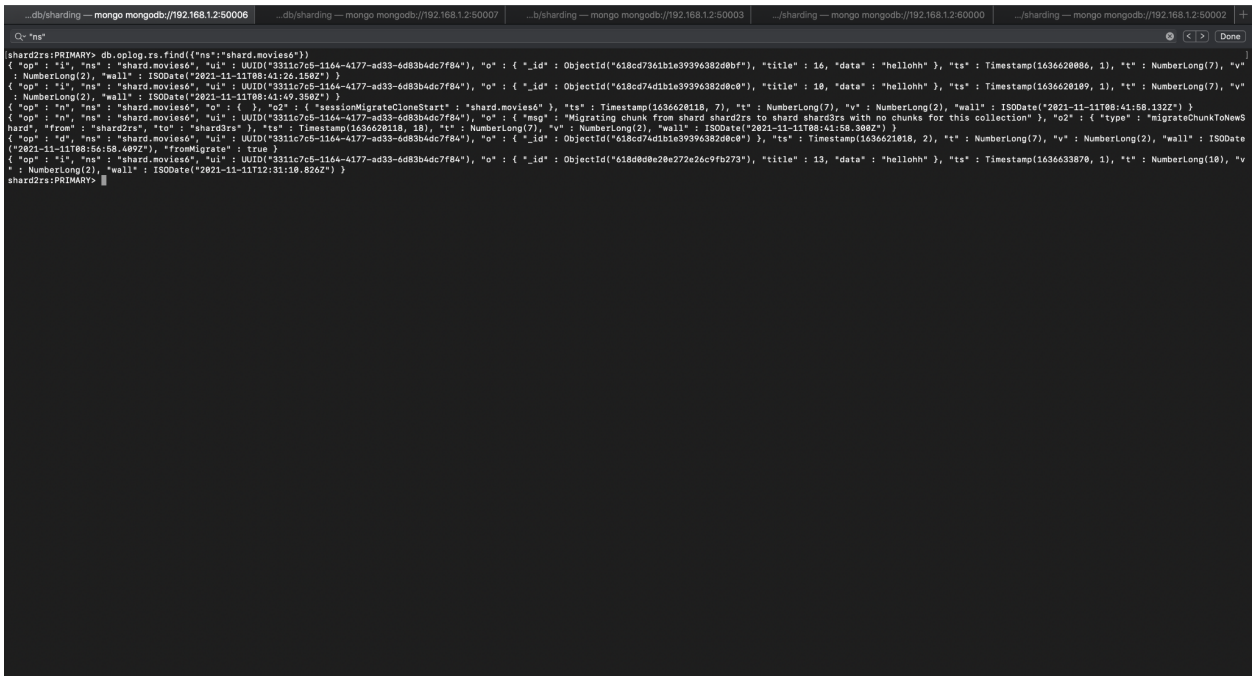
Dataload config customisation Page :

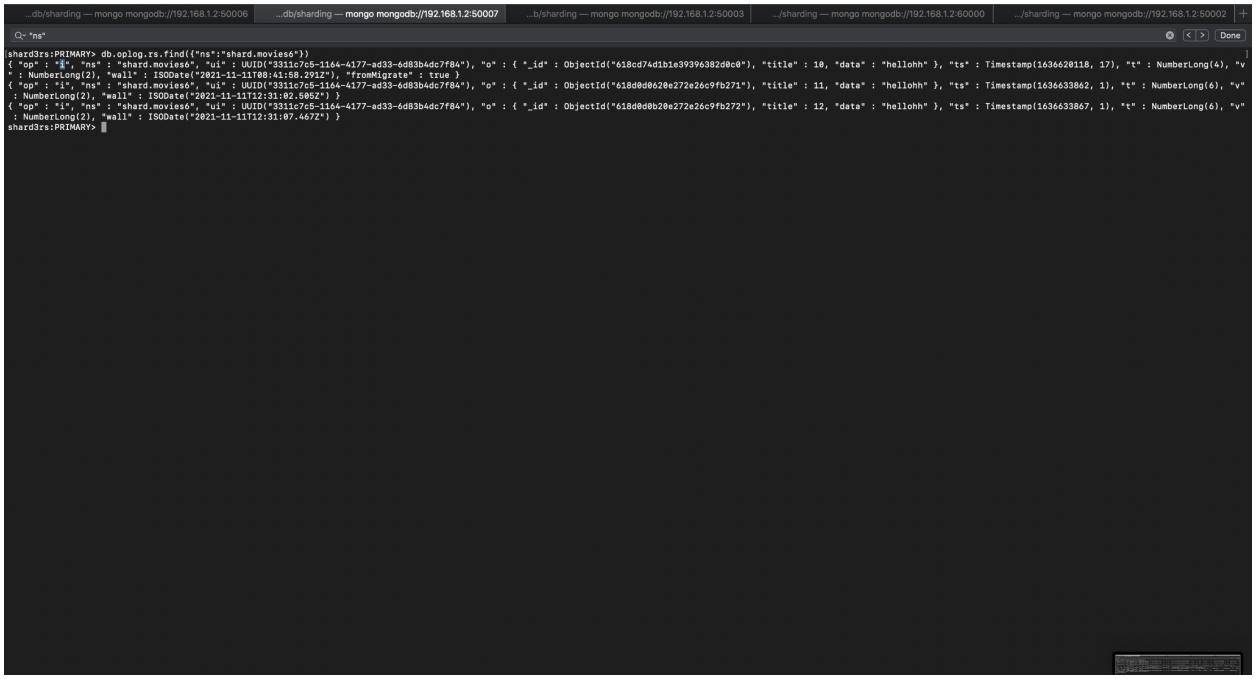
Dataload Page :

Questions :

1) Supported Sources and Targets?

OPLOG :


...db/sharding — mongo mongodb://192.168.1.2:50006
...db/sharding — mongo mongodb://192.168.1.2:50007
...b/sharding — mongo mongodb://192.168.1.2:50003
.../sharding — mongo mongodb://192.168.1.2:60000
.../sharding — mongo mongodb://192.168.1.2:50002
"ns"
Done
shard2rs:PRIMARY> db.oplog.rs.find({"ns":"shard.movies6"})
{ "op" : "i", "ns" : "shard.movies6", "ui" : UUID("3311c7c5-1164-4177-ad33-6d83b4dc7f84"), "o" : { "_id" : ObjectId("618cd7361b1e39396382d0bf"), "title" : 16, "data" : "hellohh" }, "ts" : Timestamp(1636620086, 1), "t" : NumberLong(7), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T08:41:26.150Z") }
{ "op" : "i", "ns" : "shard.movies6", "ui" : UUID("3311c7c5-1164-4177-ad33-6d83b4dc7f84"), "o" : { "_id" : ObjectId("618cd74d1b1e39396382d0c0"), "title" : 10, "data" : "hellohh" }, "ts" : Timestamp(1636620109, 1), "t" : NumberLong(7), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T08:41:49.350Z") }
{ "op" : "n", "ns" : "shard.movies6", "o" : { }, "o2" : { "sessionMigrateCloneStart" : "shard.movies6" }, "ts" : Timestamp(1636620118, 7), "t" : NumberLong(7), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T08:41:58.132Z") }
{ "op" : "n", "ns" : "shard.movies6", "ui" : UUID("3311c7c5-1164-4177-ad33-6d83b4dc7f84"), "o" : { "msg" : "Migrating chunk from shard shard2rs to shard shard3rs with no chunks for this collection" }, "o2" : { "type" : "migrateChunkToNewShard", "from" : "shard2rs", "to" : "shard3rs" }, "ts" : Timestamp(1636620118, 18), "t" : NumberLong(7), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T08:41:58.300Z") }
{ "op" : "d", "ns" : "shard.movies6", "ui" : UUID("3311c7c5-1164-4177-ad33-6d83b4dc7f84"), "o" : { "_id" : ObjectId("618cd74d1b1e39396382d0c0") }, "ts" : Timestamp(1636621018, 2), "t" : NumberLong(7), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T08:56:58.409Z"), "fromMigrate" : true }
{ "op" : "i", "ns" : "shard.movies6", "ui" : UUID("3311c7c5-1164-4177-ad33-6d83b4dc7f84"), "o" : { "_id" : ObjectId("618d0d0e20e272e26c9fb273"), "title" : 13, "data" : "hellohh" }, "ts" : Timestamp(1636633870, 1), "t" : NumberLong(10), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T12:31:10.826Z") }
shard2rs:PRIMARY>


...db/sharding — mongo mongodb://192.168.1.2:50006
...db/sharding — mongo mongodb://192.168.1.2:50007
...b/sharding — mongo mongodb://192.168.1.2:50003
.../sharding — mongo mongodb://192.168.1.2:60000
.../sharding — mongo mongodb://192.168.1.2:50002
"ns"
Done
shard3rs:PRIMARY> db.oplog.rs.find({"ns":"shard.movies6"})
{ "op" : "n", "ns" : "shard.movies6", "ui" : UUID("3311c7c5-1164-4177-ad33-6d83b4dc7f84"), "o" : { "_id" : ObjectId("618cd74d1b1e39396382d0c0"), "title" : 10, "data" : "hellohh" }, "ts" : Timestamp(1636620118, 17), "t" : NumberLong(4), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T08:41:58.291Z"), "fromMigrate" : true }
{ "op" : "i", "ns" : "shard.movies6", "ui" : UUID("3311c7c5-1164-4177-ad33-6d83b4dc7f84"), "o" : { "_id" : ObjectId("618d0d0620e272e26c9fb271"), "title" : 11, "data" : "hellohh" }, "ts" : Timestamp(1636633862, 1), "t" : NumberLong(6), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T12:31:02.985Z") }
{ "op" : "i", "ns" : "shard.movies6", "ui" : UUID("3311c7c5-1164-4177-ad33-6d83b4dc7f84"), "o" : { "_id" : ObjectId("618d0d0b20e272e26c9fb272"), "title" : 12, "data" : "hellohh" }, "ts" : Timestamp(1636633867, 1), "t" : NumberLong(6), "v" : NumberLong(2), "wall" : ISODate("2021-11-11T12:31:07.467Z") }
shard3rs:PRIMARY>