**Mongo Prompt –**

>> mongosh –help

>>mongoexport –help

>>show db

>>pwd – Print working directory

>>cd

>> Dir

>>Use database\_name – to open the database.

>>Cls – To clear the screen.

**To export a file from prompt in JSON format-**

>> uri stands for Uniform resource identifier – mongodb+srv://user:password@<cluster>.mongodb.net/database

>> mongoexport --uri=’’ ” --collection= --out=

Ex-

>> mongoexport --uri=”mongodb+srv://admin1:admin1@mongocv1.pjvp0jb.mongodb.net/import\_exportDB” --collection=import\_data --out=import\_data.json

**To import the JSON –**

>>mongoimport –uri=”” –collection= --file=

Ex-

>> mongoimport --uri="mongodb+srv://admin1:admin1@mongocv1.pjvp0jb.mongodb.net/import\_exportDB" --collection=import\_data --file=import\_data.json

**To Import and export data in Binary JSON format –**

>> Binary JSON is not human readable formant.

>> It is optimized for speed and performance.

>> With the help of Binary JSON format, we can import or export the file and databases.

**To export Binary JSON Database–**

>>mongodump –uri=””

Ex-

>> mongodump --uri="mongodb+srv://admin1:admin1@mongocv1.pjvp0jb.mongodb.net/import\_exportDB"

**To import the Binary JSON Database-**

>>mongorestore –uri”” folder\_name/database\_name

Ex-

>> mongorestore --uri="mongodb+srv://admin1:admin1@mongocv1.pjvp0jb.mongodb.net/import\_exportDB" dump/import\_exportDB

**Creating Database using Shell –**

>> use database\_name(Db name which you want to create)

**To Create a collection-**

>>db.createCollection(“New\_test\_1”)

**To Delete/Drop a collection –**

>>db.collection\_name.drop()

Ex-

>>db.new\_test\_2.drop()

**To Drop a Database-**

>>To drop a database we need to make sure that we are in the database which we are trying to delete.

>>db.dropDatabase()

Find and findOne Method–

**FindOne**

>>db.collection\_name.findOne(optional\_query, Projection)

Ex-

>>db.grades.findOne()

**Find-**

>>db.grades.find(optional\_query, Projections)

**Ex-**

>>db.grades.findOne()

>> The query will ask to type “it” to show more documents from the row.

Ex-

>>db.grades.find()

Ex with conditions(Key,Value) Pair-

>>db.grades.find({“class\_id”:100})

>>db.grades.find({“student\_id”:01})

Ex with multiple conditions(Key, Values)-

>>db.grades.find({“student\_id”:01, “class\_id”:270})

**Comparison OPERATORS –**

>>$eq – Equals to

>>$gt – greater than

>>$gte - greater than or equal to

>>$in- Any value in the array

>>$lt- less than

>>$lte – less than or equal to

>>$ne – Not equals to

>>$lt = less than which is equivalent to Less than sign (“<”)

>>{field:{operator:value}}

Ex-

>>{“salary”:{$lt:50000}})

**Logical Operators-**

**>>$and**

Ex:

>> db.trips.find({$and:[{"tripduration":{$gt:400}},{"birth year":{$gt:1988}}]})

**>>$or**

Ex:

>> db.trips.find({$or:[{"start station name":{$type:10}},{"end station name":{$type:10}}]})

**>>$nor**

**>>$not**

**$expr Operator:**

>>db.routes.find({$expr:{$eq:[“src\_airport”,”dst\_airport”]}})

**Element Operator**

**>>$exists**

Ex:

>>

**>>$type – It represents data types.**

* **Double : 1**
* **String : 2**
* **Null : 10**

Ex:

>>

**Cursor Method –**

* **Count**

Ex:

>>db.trips.find().count()

>>db.trips.find({“tripduration”:{$gt:300}}).count()

>>db.trips.find({“tripduration”:{$gt:300,$lt:400}}).count()

>> db.companies.find({"number\_of\_employees":{$gt:100000}}).count()

* **Sort : (for Asce:1, For Desc:-1)**

**ASC**

>>db.trips.find({“tripduration”:{$gt:300,$lt:400}}).sort({“tripduration”:1})

**DESC**

>> db.trips.find({“tripduration”:{$gt:300,$lt:400}}).sort({“tripduration”:1})

**Sorting with multiple conditions-**

>>db.trips.find().sort(“start station id”:1,”tripduration”:-1)

* **Limit**

>>db.trips.find().limit(1)

>>db.trips.find().limit(5)

* Skip
* **Size**

>>db.trips.find().limit(5).size()

**Projection Method** – By specifying the fields you want to include in the query results, you can retrieve only the necessary data.

>>To include the specified field, we use 1

>> To exclude the field, we use 0

Ex:

>> db.companies.find({},{"number\_of\_employees":1,"name":1,"\_id":0}).sort({"number\_of\_employees":-1}).limit(10)

>>db.comapnies.find({},{“name”:1,”founded\_year”:1})

>>db.comapnies.find({},{“name”:1,”founded\_year”:1,”\_id”:0})

**Querying Embedded Fields**: To query embedded fields we use dot(.) notation.

Ex:

>> db.inspections.find({"address.zip":10304})

**Querying Array :**

>>$all

>>$size

>> $elematch

**Inserting documents:**

>> By creating a collection, we can insert documents in the collection.

Insert function –

>> db.collection.insert(<document or array of document>, {write concern : <Document>, ordered: <Boolean>})

>>db.testcollection.insert({“Name”:”Tom Hank”, “Profession”:”Actor”})

>>

**To Delete the Documents:**

>>deleteOne

>>deleteMany