**MySql Commands**

1. docker run --detach --env MYSQL\_ROOT\_PASSWORD=dummypassword --env MYSQL\_USER=social-media-user --env MYSQL\_PASSWORD=dummypassword --env MYSQL\_DATABASE=social-media-database --name mysql --publish 3306:3306 mysql:8-oracle

**to connect to mysql**

\connect social-media-user@localhost:3306

to use schema

\use social-media-database

toggle to sql mode

\sql

**MongoDB Commands**

**To create database:**

use <dbname>

Ex: use sampleDB

**To create collection**

db.<collectionname>.insertOne({“key”:”value”})

Ex: db.Employee.insertOne({"name":"Parvathi"});

**To check collections in DB**

db.Employee.find();

**To change the port of mongo server in cmd prompt**

sudo mongod --port 27018

to run the mongo server on this port mongo --port 27018

**CRED commands**

**insert**

db.FlightInfo.insert(

{flightName:"G157",

"airlines":"AirIndia",

"arrival":"HYD",

"departure":"MAC"

})

**insertMany:**

db.FlightInfo.insertMany([

{flightName:"G157",

"airlines":"AirIndia",

"arrival":"HYD",

"departure":"MAC"

},

{"flightName":"G196",

"airlines":"Indigo",

"arrival":"HYD",

"departure":"CHK"

}

])

**deleteOne**: -- delete only first entry

db.FlightInfo.deleteOne({flightName:’G157’})

**deleteMany**: -- delete all matched records

db.FlightInfo.deleteMany({arrival:’HYD’})

**updateOne:** -- updates first matched document

db.FlightInfo.updateOne({flightName:”G157”},{$set:{marker:”updated”}})

**updateMany:** -- updates all documents matced with the crieteria

db.FlightInfo.updateMany({arrival:’HYD’},{$set:{marker:’updated2’}})

**update:** replaces the whole record with matched criteria

db.FlightInfo.update({arrival:’HYD’},{marker:’updated2’})

**replaceOne:**

db.FlightInfo.replaceOne({\_id:ObjectId("63b5e9cfaadea29883a11b9a")

},{flightName:"M157",

"airlines":"AirAsia",

"arrival":"MUM",

"departure":"GOA"

})

**findOne**: fetches first matched record

db.FlightInfo.findOne({marker:"updated2"});

**find**: fetches all matched records

db.FlightInfo.find({marker:"updated2"});

**Projection**:

suppose you want to fecth only specific attributes from a document we can filter out those.

Let’s add passenger’s data.

db.passengers.insertMany([{name:”parvathi”,age:33},{name:”surya”,age:35},{name:”Eshaan”,age:6}]);

If i want to fetch only passenger’s name

db.passengers.find({},{name:1})

It will fetch name along with Object Id.If i don’t want to print ID

db.passengers.find({},{name:1,\_id:0})

Here first curly braces is empty means we want to fetch all the documents. 1 means that attribute needs to be present in result list 0 means we are skipping that attribute in the resultset.This filter happens on mongoDB server

**Embedded Documents:** Nested documents or arrays

if we want to update nested documents for flights

db.FlightInfo.updateMany({},{$set: {status: {description:”delayed”, lastupdated : “Today”}}})

status is an embedded/ inner document which going to be updated to all documents.

If we want to add one more inner object inside status

db.FlightInfo.updateMany({},{$set: {status: {description:”on-time”, lastupdated : “one hour ago”, updatedBy:{admin:”parvathi”}}}})

It will add one more nested document.

**Arrays:**

we can add array elements also to the documents.

db.passengers.updateOne({name:”parvathi”},{$set:{hobbies:[“gardening”,”trave’”,”crochet”]}})

if we want to fetch perticular person hobbies

db.passengers.findOne({name:”parvathi”}).hobbies

if we want fetch persons based on hobbies

db.passengers.find({hobbies:”gardening”})

If want to fetch inner document like updatedBy in the FlightInfo

db.FlightInfo.find({“status.description”:”on-time”})

db.FlightInfo.find({“status.updatedBy.admin”:”parvathi”})

inner objects should be fetched using dot.