Mongo Db Commands

mongod –version: for checking version

mongosh: connect to mongo db shell

show dbs: display all the databases

show collections: display all the tables

use company: creates a database named company

db.createCollection(“emp”): creates a table/collection and makes the database active

db.Employee.insertOne({ID: 1, Name:”Pratyush”}): creates a table Employee and inserts data in it

db.Employee.find(): retrieves all the records

db.Employee.insertMany([{}]): insert multiple records

db.Employee.updateOne({name: “Prayushi”},{$set:{department:”Finance”}}): update one record

db.Employee.updateMany({},{$set:{Hobby:"Procrastinating"}}): add column with common value for all collections

db.Employee.updateOne({Name:'Aaditya'},{$set:{idcards:{hasPanCard:true, hasAadharCard:true}}}): for nested document

db.Employee.find({'idcards.hasAadharCard':true}): will display only those entry which have hasAadharCard as true

db.Employee.find().count(): gives the total number of records

db.Employee.find({Salary:{$gt:60000}}): displays records having salary greater than 60000.  
\* For the above query we can also use $lt, $gte, $lte, $eq, $ne \*

db.Employee.updateMany({},{$inc:{id:1}}): increments id by 1 everytime we create a new record

db.Employee.find({Name: {$in:['Aaditya', 'Santosh']}}): similar to in operator in sql

db.Employee.find({Name: {$nin:['Aaditya', 'Santosh']}}): similar to not in operator in sql

db.Employee.find().skip(2).limit(2): similar to offset and limit in sql

db.Employee.find().sort({Salary: 1}): similar to order by ASC

db.Employee.find().sort({Salary: -1}): similar to order by DESC

db.Employee.deleteOne({Name:'Krushna'}): delete 1 record

db.Employee.aggregate([{$group:{\_id:'$Hobby', count:{$sum:1}}}]): similar to group by clause in sql

db.Students.find({}, { StudName: 1, Grade: 1, \_id: 0 }): will give only StudName and Grade column with their records

db.Employee.updateMany({},{$unset:{Hobby:""}}): remove data from a specific column

db.Employee.updateOne({Name: "Pratyush"}, {$push:{Hobby: "Watching anime"}}): add data in a form of array

db.Employee.updateMany({Name:{$in:["Prayushi", "Prakash", "Suchismita"]}}, {$push:{Hobby:{$each:["Working", "Travelling"]}}}): updates many fields and adds data in a form of array

db.Employee.find({$expr:{$gt:[{$size:"$Hobby"}, 1]}}): find records where there is more than 1 hobby

db.Employee.find({$expr:{$lt:[{$size:"$Hobby"}, 2]}}): find records where there is less than 2 hobby  
\* We use $Hobby since we consider it as an array here \*

db.Employee.updateOne({Name: "Pratyush"}, {$set:{DOJ: new Date("2004-01-03")}}): add date manually

db.Employee.updateMany({}, {$set:{DOJ: new Date()}}): sets date automatically using system date

db.Employee.find({\_id:ObjectId("674951f11b02be3ffb0d8190")}, {Name:1, Age:1, \_id:1}): retrieve a specific entry with specified columns using \_id

db.Employee.find({}, {Name:1, Age:1}): will retrieve all records with specified columns

db.Employee.find({Name:{$in:["Pratyush", "Prayushi"]}}, {Name:1, Age:1, \_id:1}): display specific entries with specific columns

db.Employee.find({Name:{$regex:/^S/}}): find employees whose name starts with ‘S’

db.Employee.find({Name:{$regex:/h$/}}): find employees whose name starts with ‘h’

db.Employee.find({Name:{$regex:/l/, $options:'i'}}): searches for Name which has ‘l’ in it and makes it case insensitive

db.Employee.updateMany({Name:{$in:["Pratyush", "Nihal"]}}, {$set:{Salary:null}}): set salary column to null

db.Employee.updateMany({}, {$unset:{Salary:""}}): remove the salary column from all records

const department = db.Department.findOne({DeptName: "Finance"}): create a variable for finding records having department as Finance  
db.Staff.find({DepartmentId: department.DeptId}): now we can find the entries from other collection using the above created variable

db.Staff.find({DepartmentId: department.DeptId}).count(): we can get the count of specific field now with the help of the above created variable