Task : 14/09/2020

**Task 1 : Perform following Queries:**

// 1. Delete the location where locaction="Mumbai" for any one employee

db.Profile.deleteOne**({**Location**:**"Mumbai"**})**

// 2. Delete last inserted record from your collection

db.Profile.find**()**.sort**({**\_id**:**-1**})**.limit**(**1**)**

// 3. Delete all records where location is mumbai

db.Profile.deleteMany**({**Location**:**"Mumbai"**})**

// 4. Delete all records with location Delhi and name Amit

db.Profile.deleteMany**({**Location**:**"Delhi"**,**Name**:**"Amit"**})**

// 5. Delete all records from collection

db.Profile.deleteMany**()**

**Task 2 : Perform following Queries:**

// 1. Update the fees where RegNo="R001", put fees =40000

db.Registration.updateMany**({**RegNo**:**1**},{**$set**:{**Fees**:**40000**}})**

// 2. Increase fees by 1000 for all course

db.Registration.updateMany**({},** **{**$inc**:** **{**Fees**:**1000 **}})**

// 3. increase fees by 10% for JAVA course

db.Registration.updateMany**({**Course**:**"JAVA"**},{**Fees**:{**$add**:{**$mul**:** **{**Fees**:**1.1**}}}})**

**Task 3: Perform following Queries:**

db.person.insertMany

**(** **[**

**{** \_id **:** "1001"**,** name**:** "Franklin Roosevelt"**,** quote**:** "More than just an end to war, we want an end to the beginnings of all wars." **},**

**{** \_id **:** "1002"**,** name**:** "peter Dale Scott"**,** quote**:**"I guess that when you invade a nation of warlords, you end up having to deal with warlords." **},**

**{** \_id **:** "1003"**,** name**:** "Robert E. Lee"**,** quote**:** "What a cruel thing war is... to fill our hearts with hatred instead of love for our neighbors."**},**

**{** \_id **:** "1004"**,** name**:** "William Tecumseh Sherman"**,** quote **:** "War is cruelty. There is no use trying to reform it. The crueler it is, the sooner it will be over." **}**

**]** **);**

db.person.find**()**.pretty**();**

db.person.createIndex**({**"quote"**:**"text"**})**

db.person.find**(** **{** $text**:** **{** $search**:** "war" **}** **}** **)**.pretty**();**

db.person.find**({**$text**:{**$search**:**"\"cruel\""**}})**

**Task 4: Perform following Queries:**

// Observe retrieval time, with and without indexing

db.Profile.getIndexes**()**

db.Profile.createIndex**({**UID**:**1**})**

db.Profile.createIndex**({**Name**:**1**,**Location**:**1**})**

db.Profile.dropIndex**({**UID**:**1**})**

db.createCollection**(**'ABCD'**)**

for**(**i=0**;**i<300000**;**++i**)** db.ABCD.insert**({**"SID"**:**1**,**"Name"**:**"ANURAG"**});**

db.ABCD.dropIndex**({**S\_id**:**1**})**

**Task 5: Perform following Queries:**

db.createCollection**(**'userprofile'**)**

db.userprofile.insert**({**contact**:**"1234567809"**,**dob**:**"01-01-1991"**,**gender**:**"M"**,**name**:**"ABC"**,**user\_name**:**"abcuser"**})**

db.userprofile.ensureIndex**({**contact**:**1**,**name**:**1**})**

db.userprofile.find**({**name**:**"ABC"**})**.explain**()**

**Output:**

**{**

"queryPlanner" **:** **{**

"plannerVersion" **:** 1.0**,**

"namespace" **:** "anuragdb.userprofile"**,**

"indexFilterSet" **:** false**,**

"parsedQuery" **:** **{**

"name" **:** **{**

"$eq" **:** "ABC"

**}**

**},**

"queryHash" **:** "01AEE5EC"**,**

"planCacheKey" **:** "4C5AEA2C"**,**

"winningPlan" **:** **{**

"stage" **:** "COLLSCAN"**,**

"filter" **:** **{**

"name" **:** **{**

"$eq" **:** "ABC"

**}**

**},**

"direction" **:** "forward"

**},**

"rejectedPlans" **:** **[**

**]**

**},**

"serverInfo" **:** **{**

"host" **:** "DESKTOP-DFM8E3D"**,**

"port" **:** 27017.0**,**

"version" **:** "4.4.1"**,**

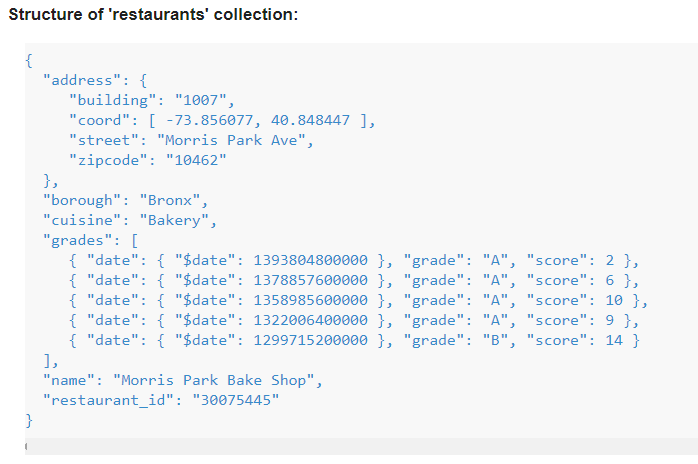
"gitVersion" **:** "ad91a93a5a31e175f5cbf8c69561e788bbc55ce1"

**},**

"ok" **:** 1.0

**}**

**LAB Task : Perform following Queries:**



// 1. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name

db.restaurants.find**({**name**:**/Reg/**},{**\_id**:**0**,**restaurant\_id**:**1**,**name**:**1**,**borough**:**1**,**cuisine**:**1**})**

// 2. Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish.

db.restaurants.find**({**$and**:[{**borough**:**"Bronx"**},{**cuisine**:{**$in**:[**"American"**,**"Chinese"**]}}]})**

// 3. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronxor Brooklyn.

db.restaurants.find**({**borough**:{**$in**:[**"Staten Island"**,**"Queens"**,**"Bronxor Brooklyn"**]}},{**\_id**:**0**,**restaurant\_id**:**1**,**name**:**1**,**borough**:**1**,**cuisine**:**1**})**

// 4. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.

db.restaurants.find**({**borough**:{**$nin**:[**"Staten Island"**,**"Queens"**,**"Bronxor Brooklyn"**]}},{**\_id**:**0**,**restaurant\_id**:**1**,**name**:**1**,**borough**:**1**,**cuisine**:**1**})**

// 5. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score which is not more than 10

db.restaurants.find**({**"grades.score"**:{**$lte**:**10**}},{**\_id**:**0**,**restaurant\_id**:**1**,**name**:**1**,**borough**:**1**,**cuisine**:**1**})**

// 6. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

db.restaurants.find**({**$or**:[{**cuisine**:{**$nin**:[**"American"**,**"Chinese"**]}},{**name**:**/^Wil/**}]},{**\_id**:**0**,**restaurant\_id**:**1**,**name**:**1**,**borough**:**1**,**cuisine**:**1**})**

// 7. Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates..

var d1 = new Date**(**"2014-03-03T00:00:00Z"**)**

db.restaurants.find**({**$and**:[{**"grades.grade"**:**"A"**},{**"grades.score"**:**11**},{**"grades.date"**:**d1**}]},{**\_id**:**0**,**restaurant\_id**:**1**,**name**:**1**,**grades**:**1**})**

// 8. Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z

var d2 = new Date**(**"2014-08-11T00:00:00Z"**)**

db.restaurants.find**({**"grades.date"**:**d2**,**"grades.grade"**:**"A"**,**"grades.score"**:**9**},{**\_id**:**0**,**restaurant\_id**:**1**,**name**:**1**,**grades**:**1**})**

// 9. Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and upto 52

db.restaurants.find**({**"address.coord.1"**:{**$lte**:**52**,**$gt**:**42**}},{**"\_id"**:**0**,**"restaurant\_id"**:**1**,**"name"**:**1**,**"address"**:**1**,**"address.coord"**:**1**})**

// 10. Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns

db.restaurants.find**()**.sort**({**name**:**1**})**