**MongoDB**

**Assignment: 1**

**Query/Find Documents: -**

1. **get all documents**

A screenshot of a computer

Description automatically generated with medium confidence

1. **get all documents with writer set to "Quentin Tarantino"**

**Graphical user interface, text, application, email

Description automatically generated**

1. **get all documents where actors include "Brad Pitt"**

**Graphical user interface, text, application, email

Description automatically generated**

1. **get all documents with franchise set to "The Hobbit"**

**Graphical user interface, text, application, email

Description automatically generated**

1. **get all movies released in the 90s**

**Graphical user interface, text, application, email

Description automatically generated**

1. **get all movies released before the year 2000 or after 2010**

**Graphical user interface, text, application, email

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

**UPDATE DOCUMENTS:**

**1. add a synopsis to "The Hobbit: An Unexpected Journey" : "A reluctant hobbit, Bilbo Baggins, sets out to the Lonely Mountain with a spirited group of dwarves to reclaim their mountain home - and the gold within it - from the dragon Smaug."**

**2. add a synopsis to "The Hobbit: The Desolation of Smaug": "The dwarves, along with Bilbo Baggins and Gandalf the Grey, continue their quest to reclaim Erebor, their homeland, from Smaug. Bilbo Baggins is in possession of a mysterious and magical ring."**

**3. add an actor named "Samuel L. Jackson" to the movie "Pulp Fiction"**

**A screenshot of a computer

Description automatically generated with medium confidence**

**TEXT SEARCH:**

1. **Find all movies that have a synopsis that contains the word "Bilbo"**
2. **find all movies that have a synopsis that contains the word "Gandalf"**

**Graphical user interface

Description automatically generated with medium confidence**

1. **find all movies that have a synopsis that contains the word "Bilbo" and not the word "Gandalf"**

**A screenshot of a computer

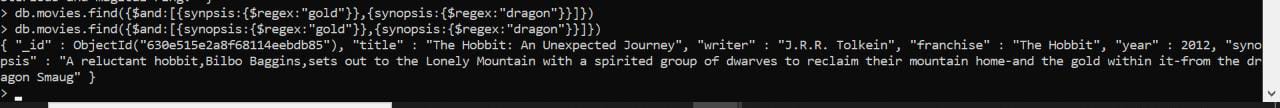
Description automatically generated with medium confidence**

1. **find all movies that have a synopsis that contains the word "dwarves" or "hobbit"**

**A screenshot of a computer

Description automatically generated with medium confidence**

1. **find all movies that have a synopsis that contains the word "gold" and "dragon"**

****

**DELETE DOCUMENTS:**

1. **Delete the movie "Pee Wee Herman's Big Adventure"**
2. **Delete the movie "Avatar"**

**Text

Description automatically generated**

**RELATIONSHIPS:**

**users Collection:**

**Graphical user interface, text, application, email

Description automatically generated**

**posts Collection:**

**Graphical user interface, text, application, email

Description automatically generated**

**Graphical user interface, text, application, email

Description automatically generated**

**comments Collection:**

**Graphical user interface, application

Description automatically generated**

**Querying related collections:**

1. **Find all users**

**Text

Description automatically generated**

1. **Find all posts**

**Text

Description automatically generated**

1. **Find all posts that was authored by "GoodGuyGreg"**
2. **Find all posts that was authored by "ScumbagSteve"**

**A screenshot of a computer

Description automatically generated with medium confidence**

1. **Find all comments**

**Text

Description automatically generated**

1. **Find all comments that was authored by "GoodGuyGreg"**
2. **Find all comments that was authored by "ScumbagSteve"**

**Graphical user interface, text

Description automatically generated**

1. **Find all comments belonging to the post "Reports a bug in your code"**

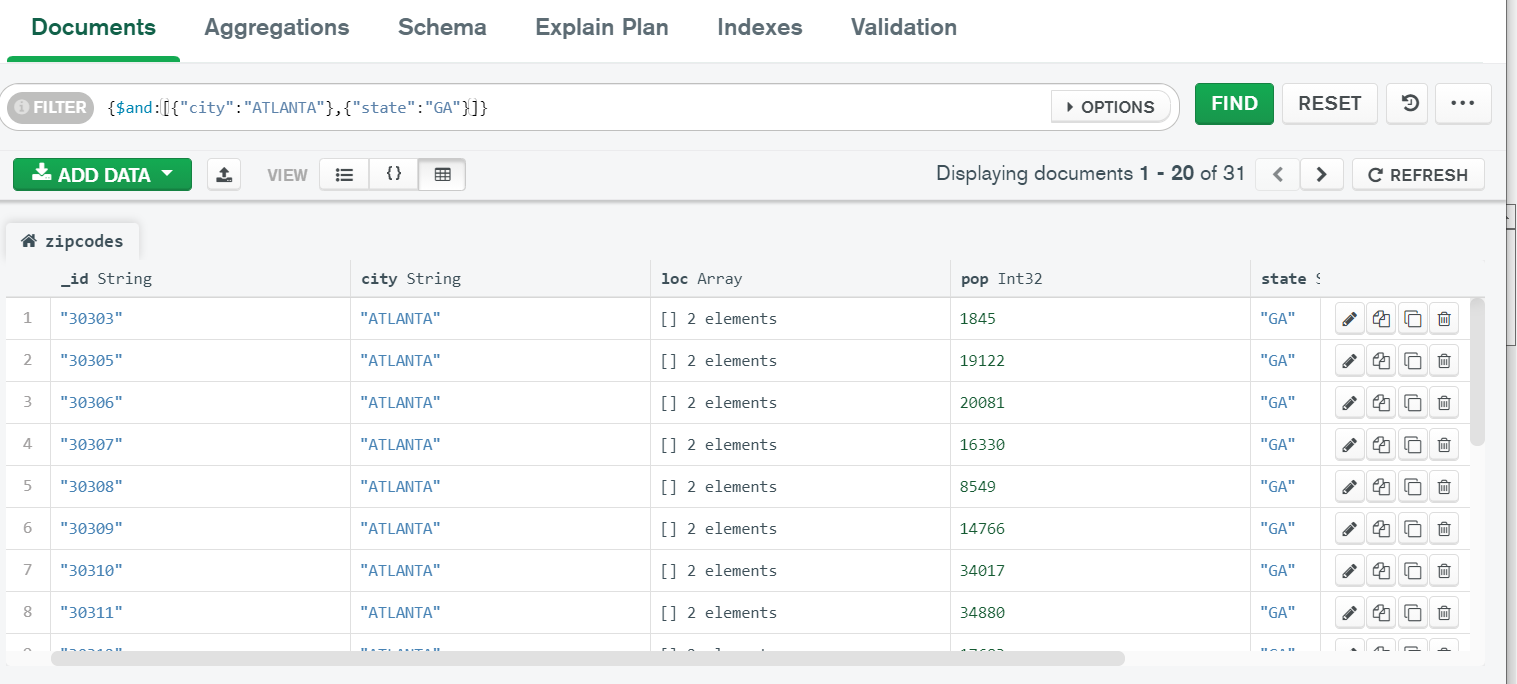


**Assignment -2**

**Atlanta population**

1. **use db.zipcodes.find() to filter results to only the results where city is**

**ATLANTA and state is GA**

****

1. **use db.zipcodes.aggregate with $match to do the same as above**

**A picture containing text

Description automatically generated**

**3. use $group to count the number of zip codes in Atlanta.**

**Text

Description automatically generated**

**4 use $group to find the total population in Atlanta.**

**Text

Description automatically generated**

**Populations By State**

1. **use aggregate to calculate the total population for each state**

**Text

Description automatically generated**

1. **sort the results by population, highest first**

**Text

Description automatically generated**

**3. limit the results to just the first 3 results. What are the top 3 states in population?**

**Text

Description automatically generated**

**Populations by City**

1. **use aggregate to calculate the total population for each city (you have to use city/state combination). You can use a combination for the \_id of the $group: {city: '$city', state: '$state'}**

**Text

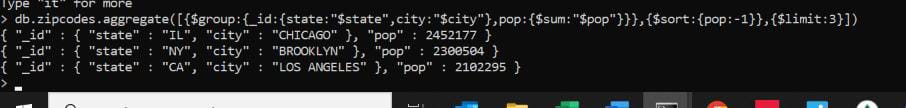
Description automatically generated**

1. **sort the results by population, highest first**

**Text

Description automatically generated**

1. **limit the results to just the first 3 results. What are the top 3 cities in population?**

****

1. **What are the top 3 cities in population in Texas?**

**A screenshot of a computer

Description automatically generated**

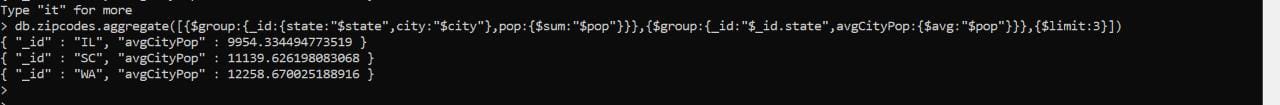
**Bonus**

1. **Write a query to get the average city population for each state.**

**A computer screen capture

Description automatically generated with medium confidence**

1. **What are the top 3 states in terms of average city population?**

****

**Assignment – 3**

1. **Write a MongoDB query to display all the documents in the collection restaurants.**

**Graphical user interface, text

Description automatically generated**

1. **Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine for all the documents in the collection restaurant.**

**A picture containing text

Description automatically generated**

1. **Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine, but exclude the field \_id for all the documents in the collection restaurant.**

**A picture containing text, outdoor, screenshot

Description automatically generated**

1. **Write a MongoDB query to display the fields restaurant\_id, name, borough and zip code, but exclude the field \_id for all the documents in the collection restaurant.**

db.restaurants.find({},{"restaurant\_id":1,"name":1,"borough":1,"address.zipcode" :1,"\_id":0});

1. **Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx**

db.restaurants.find({"borough": "Bronx"}).limit(5);

1. **Write a MongoDB query to display all the restaurant which is in the borough Bronx.**

db.restaurants.find({"borough": "Bronx"});

1. **Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.**

db.restaurants.find({"borough": "Bronx"}).skip(5).limit(5);

1. **Write a MongoDB query to find the restaurants who achieved a score more than 90**

db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 90}}}});

1. **Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100.**

db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 80 , $lt :100}}}});

1. **Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100. less than -95.754168.**

db.restaurants.find({"address.coord" : {$lt : -95.754168}});

1. **Write a MongoDB query to find the restaurants that do not prepare any cuisine of American and their grade score more than 70 and latitude less than -65.754168.**

db.restaurants.find(

{$and:

[

{"cuisine" : {$ne :"American "}},

{"grades.score" : {$gt : 70}},

{"address.coord" : {$lt : -65.754168}}

]

}

);

1. **Write a MongoDB query to find the restaurants which do not prepare any cuisine of American and achieved a score more than 70 and located in the longitude less than -65.754168.**

db.restaurants.find(

{

"cuisine" : {$ne : "American "},

"grades.score" :{$gt: 70},

"address.coord" : {$lt : -65.754168}

}

);

1. **Write a MongoDB query to find the restaurants which do not prepare any cuisine of American and achieved a grade point 'A' not belongs to the in descending order.**

db.restaurants.find( {

"cuisine" : {$ne : "American "},

"grades.grade" :"A",

"borough": {$ne : "Brooklyn"}

}

).sort({"cuisine":-1});

1. **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Will as first three letters for its name.**

db.restaurants.find(

{name: /^Wil/},

{

"restaurant\_id" : 1,

"name":1,"borough":1,

"cuisine" :1

}

);

1. **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.**

db.restaurants.find(

{name: /ces$/},

{

"restaurant\_id" : 1,

"name":1,"borough":1,

"cuisine" :1

}

);

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(

{

"borough": "Bronx" ,

$or : [

{ "cuisine" : "American " },

{ "cuisine" : "Chinese" }

]

}

);

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

db.restaurants.find(

{

"borough": "Bronx" ,

$or : [

{ "cuisine" : "American " },

{ "cuisine" : "Chinese" }

]

}

);

1. **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","Bronx","Brooklyn"]}},

{

"restaurant\_id" : 1,

"name":1,"borough":1,

"cuisine" :1

}

);

1. **Write a MongoDB query to find the restaurant ld, 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","Bronx","Brooklyn"]}},

{

"restaurant\_id" : 1,

"name":1,"borough":1,

"cuisine" :1

}

);

1. **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" :

{ $not:

{$gt : 10}

}

},

{

"restaurant\_id" : 1,

"name":1,"borough":1,

"cuisine" :1

}

);

1. **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: [

{name: /^Wil/},

{"$and": [

{"cuisine" : {$ne :"American "}},

{"cuisine" : {$ne :"Chinees"}}

]}

]}

,{"restaurant\_id" : 1,"name":1,"borough":1,"cuisine" :1}

);

1. **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.**

db.restaurants.find(

{

"grades.date": ISODate("2014-08-11T00:00:00Z"),

"grades.grade":"A" ,

"grades.score" : 11

},

{"restaurant\_id" : 1,"name":1,"grades":1}

);

1. **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 ISO Date "2014-08-11T00:00:00Z"**

db.restaurants.find(

{ "grades.1.date": ISODate("2014-08-11T00:00:00Z"),

"grades.1.grade":"A" ,

"grades.1.score" : 9

},

{"restaurant\_id" : 1,"name":1,"grades":1}

);

1. **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": {$gt : 42, $lte : 52}

},

{"restaurant\_id" : 1,"name":1,"address":1,"coord":1}

);

1. **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});

1. **Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns**

db.restaurants.find().sort(

{"name":-1} );

1. **Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.**

db.restaurants.find().sort(

{"cuisine":1,"borough" : -1,}

);

1. **Write a MongoDB query to know whether all the addresses contains the street or not.**

db.restaurants.find(

{"address.street" :

{ $exists : true }

}

);

1. **Write a MongoDB query which will select all documents in the restaurants collection where the coord field value is Double**

db.restaurants.find(

{"address.coord" :

{$type : 1}

}

);

1. **Write a MongoDB query which will select the restaurant Id name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.**

db.restaurants.find(

{"grades.score" :

{$mod : [7,0]}

},

{"restaurant\_id" : 1,"name":1,"grades":1}

);

1. **Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.**

db.restaurants.find(

{ name :

{ $regex : "mon.\*", $options: "i" }

},

{

"name":1,

"borough":1,

"address.coord":1,

"cuisine" :1

}

);

1. **Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain 'Mad' as first three letters of its name.**

db.restaurants.find(

{ name :

{ $regex : /^Mad/i, }

},

{

"name":1,

"borough":1,

"address.coord":1,

"cuisine" :1

}

);