Case Study : MongoDB Restaurant Analysis

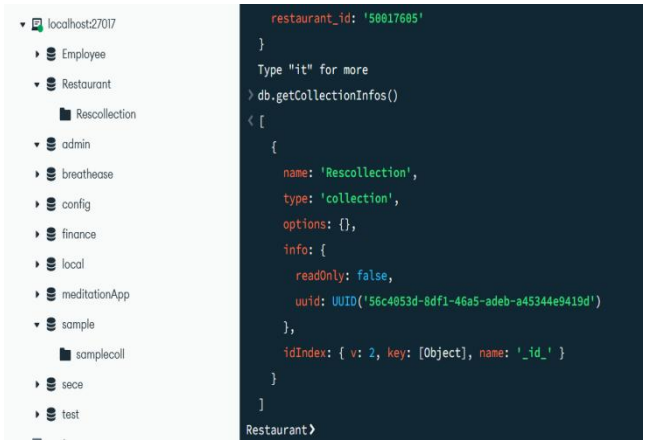
- Sruthy Suji (23AD140)

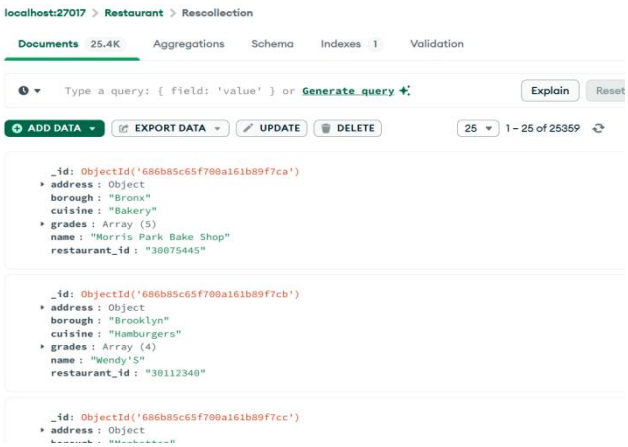
1. Create database – restaurant, create collection – rescollection. Insert the documents into collections.

Collection created under localhost:27017, db- Restaurant, collection name - Rescollection

Query : db.getCollectionInfos()

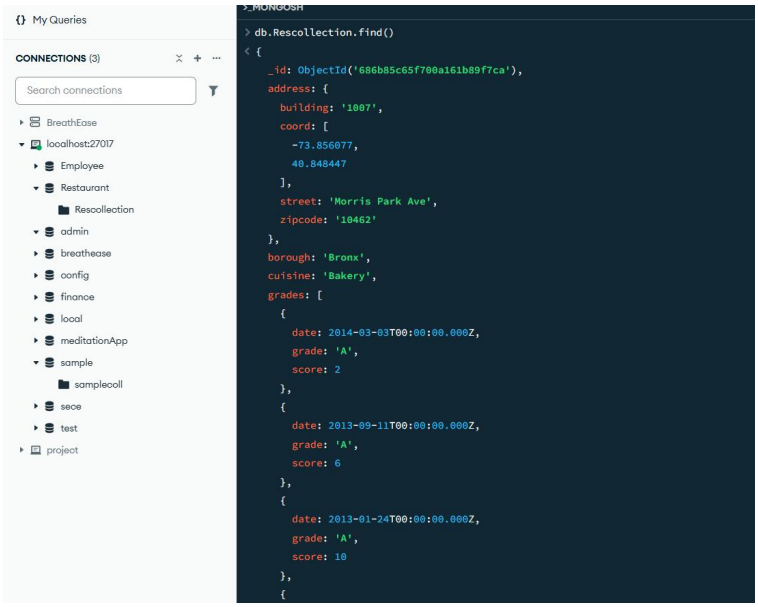
Data has been imported to mongoDB.





2. Display all the documents in the collection restaurants.

Query : db.Rescollection.find()



3. Display the fields restaurant_id, name, borough, and zip code, but exclude the field _id for all the documents in the collection restaurant.

Query :
db.Rescollection.find({},{_id:0,restaurant_id:1,name:1,borough:1,"address.zipcode":1})

```
>_MONGOSH
> db.Rescollection.find({},{_id:0,restaurant_id:1,name:1,borough:1,"address.zipcode":1})
< {
    address: {
      zipcode: '10462'
    },
    borough: 'Bronx',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
  {
    address: {
      zipcode: '11225'
    },
    borough: 'Brooklyn',
    name: "Wendy'S",
    restaurant_id: '30112340'
  }
  {
    address: {
      zipcode: '10019'
    },
    borough: 'Manhattan',
    name: 'Dj Reynolds Pub And Restaurant',
```

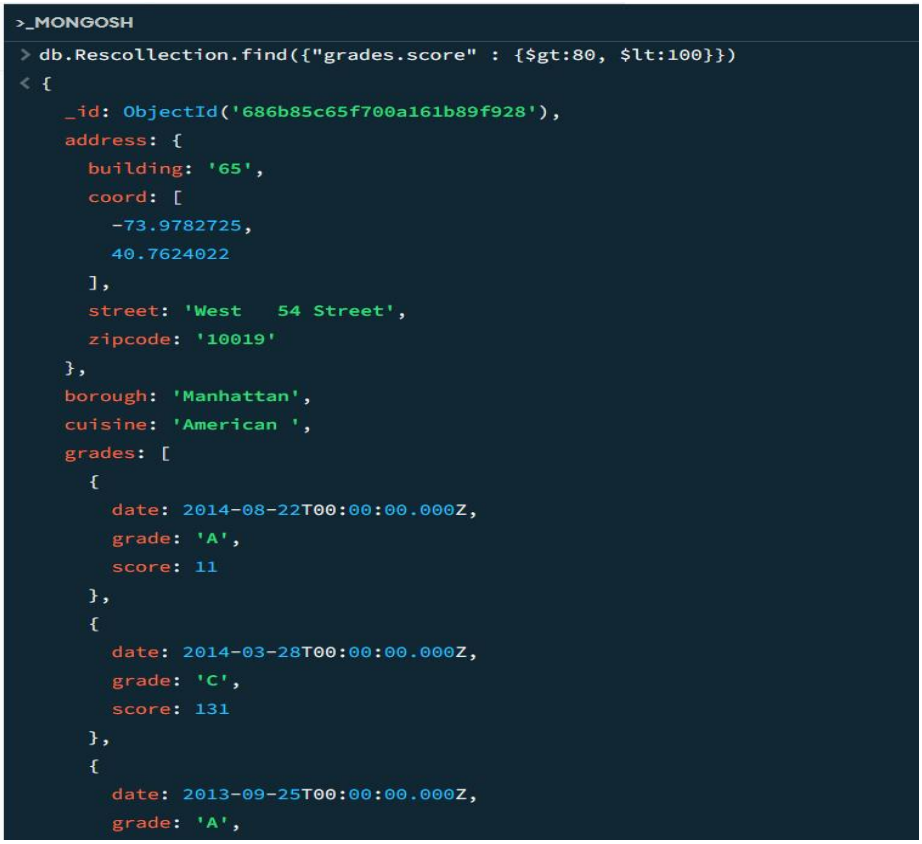
4. Find the restaurants who achieved a score more than 90.

Query : db.Rescollection.find({"grades.score" : {$gt:90}})

```
>_MONGOSH
> db.Rescollection.find({"grades.score" : {$gt:90}})
< {
    _id: ObjectId('686b85c65f700a161b89f928'),
    address: {
      building: '65',
      coord: [
        -73.9782725,
        40.7624022
      ],
      street: 'West   54 Street',
      zipcode: '10019'
    },
    borough: 'Manhattan',
    cuisine: 'American ',
    grades: [
      {
        date: 2014-08-22T00:00:00.000Z,
        grade: 'A',
        score: 11
      },
      {
        date: 2014-03-28T00:00:00.000Z,
        grade: 'C',
        score: 131
      },
      {
        date: 2013-09-25T00:00:00.000Z,
        grade: 'A',
```

5. Show the restaurants that achieved a score, more than 80 but less than 100.

Query : db.Rescollection.find({"grades.score" : {$gt:80, $lt:100}})



6. Write Query to show the restaurants that do not prepare any cuisine of american & their grade score > 70.

Query : db.Rescollection.find($and{cuisine: { $not: /^ *American *$/i }} , "grades.score" : {$gt:70}})

(Or)

db.Rescollection.find({$and:[{"grades.score":{$gt:70}},{cuisine:{$ne:"American "}}]}).limit(3)

7. Write a MongoDB query to arrange the name of the cuisine in an ascending order and for that same borough arranged in descending order.

Query : db.Rescollection.find().sort({cuisine:1,borough:-1})

```
>_MONGOSH
> db.Rescollection.find().sort({cuisine:1,borough:-1})
< {
    _id: ObjectId('686b85c75f700a161b8a2823'),
    address: {
      building: '259-11',
      coord: [
        -73.708831,
        40.73748399999999
      ],
      street: 'Hillside Avenue',
      zipcode: '11004'
    },
    borough: 'Queens',
    cuisine: 'Afghan',
    grades: [
      {
        date: 2014-09-15T00:00:00.000Z,
        grade: 'A',
        score: 13
      },
      {
        date: 2013-09-18T00:00:00.000Z,
        grade: 'A',
        score: 7
      },
      {
        date: 2013-04-18T00:00:00.000Z,
```

8. Write a MongoDB query to arrange the name of the cuisine in descending order and cuisines not equal to "Italian"

Query : db.Rescollection.find({cuisine:{$ne:"Italian "}}).sort({cuisine:-1}).limit(3)

```
>_MONGOSH
> db.Rescollection.find({cuisine:{$ne:"Italian "}}).sort({cuisine:-1}).limit(3)
< {
    _id: ObjectId('686b85c65f700a161b8a020b'),
    address: {
      building: '157',
      coord: [
        -73.996577,
        40.719515
      ],
      street: 'Mott Street',
      zipcode: '10013'
    },
    borough: 'Manhattan',
    cuisine: 'Vietnamese/Cambodian/Malaysia',
    grades: [
      {
        date: 2014-09-29T00:00:00.000Z,
        grade: 'B',
        score: 26
      },
      {
        date: 2014-05-14T00:00:00.000Z,
        grade: 'C',
```

9. Show the restaurant Id, name, borough and cuisines for those restaurants which prepared dish except 'American ' and 'Chinese' or restaurant's name begins with letter 'Bil'.

Query : db.Rescollection.find({ $or: [{ cuisine: { $nin: ["American ", "Chinese"] } }, { name: { $ne: "Bil" } }] }, { _id: 0, restaurant_id: 1, name: 1, borough: 1, cuisine: 1 })

```
>_MONGOSH
> db.Rescollection.find( { $or: [ { cuisine: { $nin: ["American ", "Chinese"] } }, { name: { $ne: "Bil" } } ] },
    { _id: 0, restaurant_id: 1, name: 1, borough: 1, cuisine: 1 })
< {
    borough: 'Bronx',
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  }
  {
    borough: 'Brooklyn',
    cuisine: 'Hamburgers',
    name: "Wendy'S",
    restaurant_id: '30112340'
  }
  {
    borough: 'Manhattan',
    cuisine: 'Irish',
    name: 'Dj Reynolds Pub And Restaurant',
    restaurant_id: '30191841'
  }
  {
    borough: 'Brooklyn',
    cuisine: 'American ',
    name: 'Riviera Caterer',
    restaurant_id: '40356018'
  }
  {
```

10. Show the restaurant Id, name, borough and cuisines and max score for restaurant serving “Indian” as cuisines.

Query : db.Rescollection.aggregate([{ $match: { cuisine: "Indian" } },{$project: {_id: 0,restaurant_id: 1, name: 1,borough: 1, cuisine: 1, max_score: { $max: "$grades.score" }} }])

```
>_MONGOSH
> db.Rescollection.aggregate([{ $match: { cuisine: "Indian" } },{$project: {_id: 0,restaurant_id: 1,
      name: 1,borough: 1, cuisine: 1, max_score: { $max: "$grades.score" }} }])
< {
    borough: 'Manhattan',
    cuisine: 'Indian',
    name: 'Mughlai Restaurant',
    restaurant_id: '40370243',
    max_score: 11
  }
  {
    borough: 'Manhattan',
    cuisine: 'Indian',
    name: 'Agra Restaurant',
    restaurant_id: '40375376',
    max_score: 28
  }
  {
    borough: 'Queens',
    cuisine: 'Indian',
    name: 'Annam Braham Restaurant',
    restaurant_id: '40380520',
    max_score: 13
  }
  {
    borough: 'Manhattan',
    cuisine: 'Indian',
    name: 'Gandhi',
```

11. Write a MongoDB query to find the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'bi' as last three letters for its name.

Query :db.Rescollection.find({ name: { $regex: /bi$/i } },{_id: 0,restaurant_id: 1, name: 1,borough: 1, cuisine: 1, "grades.score": 1})

```
>_MONGOSH
> db.Rescollection.aggregate([{ $match: { cuisine: "Indian" } },{$project: {_id: 0,restaurant_id: 1,
       name: 1,borough: 1, cuisine: 1, max_score: { $max: "$grades.score" }} }])
< {
    borough: 'Manhattan',
    cuisine: 'Indian',
    name: 'Mughlai Restaurant',
    restaurant_id: '40370243',
    max_score: 11
  }
  {
    borough: 'Manhattan',
    cuisine: 'Indian',
    name: 'Agra Restaurant',
    restaurant_id: '40375376',
    max_score: 28
  }
  {
    borough: 'Queens',
    cuisine: 'Indian',
    name: 'Annam Braham Restaurant',
    restaurant_id: '40380520',
    max_score: 13
  }
  {
    borough: 'Manhattan',
    cuisine: 'Indian',
    name: 'Gandhi',
```

12. Write a MongoDB query to find the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' as last three letters for its name.

Query :db.Rescollection.find({ name: { $regex: /il$/i } },{_id: 0,restaurant_id: 1, name: 1,borough: 1, cuisine: 1, "grades.score": 1})

```
>_MONGOSH
> db.Rescollection.find({ name: { $regex: /il$/i } },{_id: 0,restaurant_id: 1
     name: 1,borough: 1, cuisine: 1, "grades.score": 1})
< {
    borough: 'Manhattan',
    cuisine: 'Brazilian',
    grades: [
      {
        score: 9
      },
      {
        score: 12
      },
      {
        score: 4
      },
      {
        score: 18
      },
      {
        score: 24
      }
    ],
    name: 'Via Brazil',
    restaurant_id: '40682609'
  }
  {
```

13. show frequency count of restaurants by cuisines names.
Query : db.rescollection.aggregate([{$group: {_id:"$cuisine", freqeuncy:{$sum:1}}}, {$sort:{frequency:-1}}])

```
>_MONGOSH
> db.Rescollection.aggregate([{$group: {_id:"$cuisine", freqeuncy:{$sum:1}}}, {$sort:{frequency:-1}}])
< {
    _id: 'Creole',
    freqeuncy: 24
  }
  {
    _id: 'Chinese',
    freqeuncy: 2418
  }
  {
    _id: 'Filipino',
    freqeuncy: 26
  }
  {
    _id: 'Spanish',
    freqeuncy: 637
  }
  {
    _id: 'Eastern European',
    freqeuncy: 65
  }
```

14. Write a query to show all the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' anywhere in its name.
Query :db.Rescollection.find({ name: { $regex: /il/i } },{_id: 0,restaurant_id: 1, name: 1,borough: 1, cuisine: 1, "grades.score": 1})

```
>_MONGOSH
> db.Rescollection.find({ name: { $regex: /il/i } },{_id: 0,restaurant_id: 1,
      name: 1,borough: 1, cuisine: 1, "grades.score": 1})
< {
    borough: 'Brooklyn',
    cuisine: 'Delicatessen',
    grades: [
      {
        score: 10
      },
      {
        score: 10
      },
      {
        score: 8
      },
      {
        score: 10
      },
      {
        score: 13
      },
      {
        score: 9
      }
    ],
    name: "Wilken'S Fine Food",
    restaurant_id: '40356483'
```

15. Show document/record counts that has street and not street in addresses.
Query :
db.rescollection.countDocuments({"address.street":{$exists:true}})
db.rescollection.countDocuments({"address.street":{$exists:false}})

```
> db.Rescollection.countDocuments({"address.street":{$exists:true}})
< 25359
Restaurant>
```

16. 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

Query :db.Rescollection.find({$and : [{cuisine:{$ne:"American "}},{"grades.score":{$gt:70}},{"address.coord.1":{$lt:65.754168}}]})

17. 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.

Query :db.Rescollection.find({ "grades.score": { "$mod": [7, 0] } },
{ "restaurant_id": 1, "name": 1, "grades": 1, "_id": 0 })

```
>_MONGOSH
> db.Rescollection.find( { "grades.score": { "$mod": [7, 0] } },
    { "restaurant_id": 1, "name": 1, "grades": 1, "_id": 0 })
< {
    grades: [
      {
        date: 2014-03-03T00:00:00.000Z,
        grade: 'A',
        score: 2
      },
      {
        date: 2013-09-11T00:00:00.000Z,
        grade: 'A',
        score: 6
      },
      {
        date: 2013-01-24T00:00:00.000Z,
        grade: 'A',
        score: 10
      },
      {
        date: 2011-11-23T00:00:00.000Z,
        grade: 'A',
```

DATA ANALYSIS USING PYTHON AND MONGODB
dataset - restaurant data

1. Create database – restaurant, create collection – rescollection. Insert the documents into collections.
2. Display all the documents in the collection restaurants.

```
[2]: pip install pymongo

Requirement already satisfied: pymongo in c:\anaconda\lib\site-packages (4.13.2)
Requirement already satisfied: dnspython<3.0.0,>=1.16.0 in c:\anaconda\lib\site-packages (from pymongo) (2.7.0)
Note: you may need to restart the kernel to use updated packages.

[4]: from pymongo import MongoClient
     #create a mongoclient object and connect to mongoDB
     client = MongoClient("mongodb://localhost:27017/")

[12]: db_name = "Restaurant"
      collection_name = "Rescollection"
      if db_name in client.list_database_names():
          print("database already exist")

      database already exist

[19]: import pandas as pd
      df= pd.read_json(r"C:\Users\Sruthy suji\Downloads\restaurants-dataset.json",lines=True)
      records = df.to_dict(orient="records")

[30]: db= client[db_name]
      collection=db[collection_name]
      if records:
          result = collection.insert_many(records)
      else :
          print("no records found")

[44]: restaurant= collection.find().limit(15)
      list(restaurant)

[44]: [{'_id': ObjectId('686b85c65f700a161b89f7ca'),
        'address': {'building': '1007',
         'coord': [-73.856077, 40.848447],
         'street': 'Morris Park Ave',
         'zipcode': '10462'},
        'borough': 'Bronx',
        'cuisine': 'Bakery',
        'grades': [{'date': datetime.datetime(2014, 3, 3, 0, 0),
          'grade': 'A',
          'score': 2},
         {'date': datetime.datetime(2013, 9, 11, 0, 0), 'grade': 'A', 'score': 6},
         {'date': datetime.datetime(2013, 1, 24, 0, 0), 'grade': 'A', 'score': 10},
         {'date': datetime.datetime(2011, 11, 23, 0, 0), 'grade': 'A', 'score': 9},
         {'date': datetime.datetime(2011, 3, 10, 0, 0), 'grade': 'B', 'score': 14}],
        'name': 'Morris Park Bake Shop',
        'restaurant_id': '30075445'},
```

3. Display the fields restaurant_id, name, borough, and zip code, but exclude the field _id for all the documents in the collection restaurant.

```
results = collection.find({},{"_id":0,"restaurant_id":1,"name":1,"borough":1,"address.zipcode":1}).limit(10)
for doc in results:
    print(doc)
```

```
{'address': {'zipcode': '10462'}, 'borough': 'Bronx', 'name': 'Morris Park Bake Shop', 'restaurant_id': '30075445'}
{'address': {'zipcode': '11225'}, 'borough': 'Brooklyn', 'name': "Wendy'S", 'restaurant_id': '30112340'}
{'address': {'zipcode': '10019'}, 'borough': 'Manhattan', 'name': 'Dj Reynolds Pub And Restaurant', 'restaurant_id': '30191841'}
{'address': {'zipcode': '11224'}, 'borough': 'Brooklyn', 'name': 'Riviera Caterer', 'restaurant_id': '40356018'}
{'address': {'zipcode': '11374'}, 'borough': 'Queens', 'name': 'Tov Kosher Kitchen', 'restaurant_id': '40356068'}
{'address': {'zipcode': '11369'}, 'borough': 'Queens', 'name': 'Brunos On The Boulevard', 'restaurant_id': '40356151'}
{'address': {'zipcode': '10314'}, 'borough': 'Staten Island', 'name': 'Kosher Island', 'restaurant_id': '40356442'}
{'address': {'zipcode': '11234'}, 'borough': 'Brooklyn', 'name': "Wilken'S Fine Food", 'restaurant_id': '40356483'}
{'address': {'zipcode': '11219'}, 'borough': 'Brooklyn', 'name': 'Regina Caterers', 'restaurant_id': '40356649'}
{'address': {'zipcode': '11226'}, 'borough': 'Brooklyn', 'name': 'Taste The Tropics Ice Cream', 'restaurant_id': '40356731'}
```

4. Find the restaurants who achieved a score more than 90.

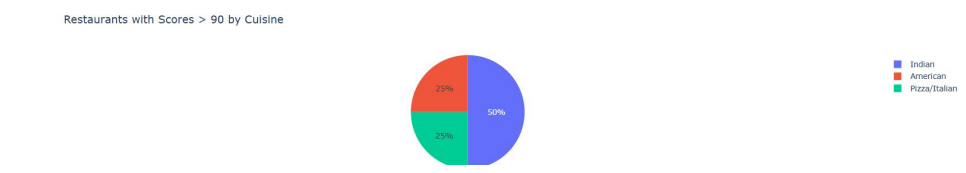
```
mongo_query = [
    {"$match": {"grades.score": {"$gt": 90}}},
    {"$group": {
        "_id": "$cuisine",
        "count": {"$sum": 1}
    }}
]

# 3. Run Query
results = list(collection.aggregate(mongo_query))
print(results)

# 4. Convert to DataFrame
df = pd.DataFrame(results)
df.columns = ['Cuisine', 'Count']

# 5. Plot using Plotly
fig = px.pie(df, values='Count', names='Cuisine', title='Restaurants with Scores > 90 by Cuisine')
fig.show()
```

```
[{'_id': 'Indian', 'count': 6}, {'_id': 'American ', 'count': 3}, {'_id': 'Pizza/Italian', 'count': 3}]
```



5. Show the restaurants that achieved a score, more than 80 but less than 100.

```
mongo_query = [
    {"$unwind": "$grades"},
    {"$match": {
        "grades.score": {"$gt": 80, "$lt": 100}
    }},
    {"$project": {
        "_id": 0,
        "name": 1,
        "cuisine": 1,
        "borough": 1,
        "score": "$grades.score",
        "grade": "$grades.grade"
    }}
]

results = list(collection.aggregate(mongo_query))
df = pd.DataFrame(results)

print(df.head())
```

```
     borough        cuisine                          name  score grade
0  Manhattan         Indian                       Gandhi     92     C
1  Manhattan  Pizza/Italian                Bella Napoli     98     C
2  Manhattan       American   West 79Th Street Boat Basin Cafe     89     C
3     Queens           Thai                  Spicy Shallot     84     C
4  Manhattan       American                Bistro Caterers     84     C
```

6. Write Query to show the restaurants that do not prepare any cuisine of american & their grade score > 70.

```
mongo_query=[
    {"$unwind": "$grades"},  # Unwind grades array to filter on individual scores
    {"$match": {
        "cuisine": {"$ne": "American"},
        "grades.score": {"$gt": 70}
    }},
    {"$project": {
        "_id": 0,
        "name": 1,
        "cuisine": 1,
        "borough": 1,
        "score": "$grades.score",
        "grade": "$grades.grade"
    }}
]
results = list(collection.aggregate(mongo_query))
df = pd.DataFrame(results)

print(df.head())
```

```
      borough                                        cuisine  \
0  Manhattan                                       American
1  Manhattan                                         Indian
2  Manhattan                                       American
3  Manhattan                                  Pizza/Italian
4      Bronx  Latin (Cuban, Dominican, Puerto Rican, South &...

                          name  score grade
0    Murals On 54/Randolphs'S    131     C
1                      Gandhi     92     C
2  Live Bait Bar & Restaurant     71     C
3               Bella Napoli      98     C
4   El Molino Rojo Restaurant     76     C
```

7. Write a MongoDB query to arrange the name of the cuisine in an ascending order and for that same borough arranged in descending order.

```
mongo_query=[
     {"$sort": {"cuisine":1,"borough":-1}},
    {"$project": {
        "_id": 0,
        "name": 1,
        "cuisine": 1,
        "borough": 1,
        "score": "$grades.score",
        "grade": "$grades.grade"
    }}
]
results = list(collection.aggregate(mongo_query))
df = pd.DataFrame(results)

print(df.head())
```

```
  borough cuisine                      name                    score  \
0  Queens  Afghan  Choopan Kabab Restaurant  [12, 15, 13, 5, 17, 13]
1  Queens  Afghan        Afghan Kebob House      [13, 7, 21, 2, 11]
2  Queens  Afghan          Arya Kabob House                    [11]
3  Queens  Afghan             Bakhtar Kabab                [4, 5, 5]
4  Queens  Afghan  Choopan Kabab Restaurant  [12, 15, 13, 5, 17, 13]

                grade
0  [A, B, A, P, B, A]
1     [A, A, B, A, A]
2                 [A]
3           [A, A, A]
4  [A, B, A, P, B, A]
```

8. Write a MongoDB query to arrange the name of the cuisine in descending order and cuisines not equal to "Italian"

```
[55]: mongo_query=[
      {
      "$match":{"cuisine":{"$ne":"Italian"}}
      },
          {
          "$sort":{"cuisine":1}
          },
          {
          "$project":{
              "_id": 0,
              "name": 1,
              "cuisine": 1,
              "borough": 1,
              "score": "$grades.score",
              "grade": "$grades.grade"}
          }
      ]
      results = list(collection.aggregate(mongo_query))
      df = pd.DataFrame(results)

      print(df.head())
```

```
     borough cuisine                      name                   score  \
0  Manhattan  Afghan      Afghan Kebab House #1       [12, 19, 13, 2, 9]
1     Queens  Afghan  Choopan Kabab Restaurant  [12, 15, 13, 5, 17, 13]
2     Queens  Afghan          Tariq Afghan Kabab                  [12]
3   Brooklyn  Afghan                Bahar Masala       [38, 12, 6, 10, 6]
4  Manhattan  Afghan         Ariana Kebab House            [3, 2, 2, 13]

                grade
0    [A, B, A, C, A]
1  [A, B, A, P, B, A]
2                 [A]
3    [C, A, P, A, A]
```

9. Show the restaurant Id, name, borough and cuisines for those restaurants which prepared dish except 'American ' and 'Chinese' or restaurant's name begins with letter 'Bil'.

```
mongo_query=[
{
"$match":{"$or":[{"cuisine":{"$nin":["American","Chinese"]}},{"name": {"$regex": "^Bil", "$options": "i"}}]}
},{
"$project":{
    "_id":0,
    "restaurant_id":1,
    "name":1,
    "borough":1,
    "cuisine":1
}
}
]
results = list(collection.aggregate(mongo_query))
df=pd.DataFrame(results)
print(df.head())
```

```
     borough        cuisine                          name restaurant_id
0      Bronx         Bakery          Morris Park Bake Shop      30075445
1   Brooklyn     Hamburgers                       Wendy'S      30112340
2  Manhattan          Irish  Dj Reynolds Pub And Restaurant      30191841
3   Brooklyn       American              Riviera Caterer      40356018
4     Queens  Jewish/Kosher           Tov Kosher Kitchen      40356068
```

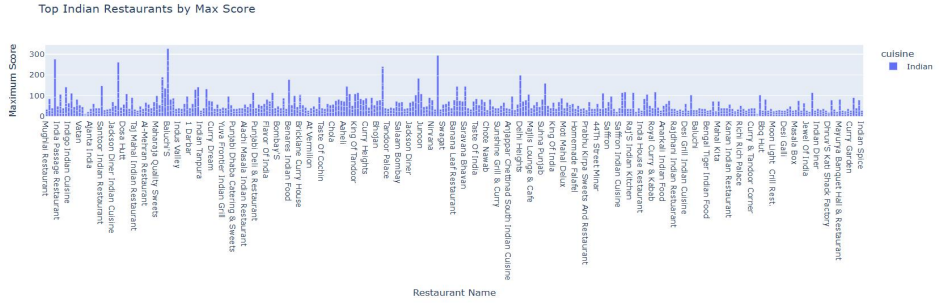
10. Show the restaurant Id, name, borough and cuisines and max score for restaurant serving “Indian” as cuisines.

```
#Show the restaurant Id, name, borough and cuisines and max score for restaurant serving “Indian” as cuisines.
mongo_query=[
{
"$match":{"cuisine":"Indian"}
},{
"$project":{
    "_id":0,
    "restaurant_id":1,
    "name":1,
    "borough":1,
    "cuisine":1,
    "max_score":{"$max":"$grades.score"}
}
}
]
results= list(collection.aggregate(mongo_query))
df=pd.DataFrame(results)
print(df.head())

fig = px.bar(df,x='name',y='max_score',  color='cuisine',title='Top Indian Restaurants by Max Score', labels={'name': 'Restaurant Name', 'max_score': 'Maximum Score'},
             height=500)

fig.show()
```

```
     borough cuisine                    name restaurant_id  max_score
0  Manhattan  Indian        Mughlai Restaurant       40370243         11
1  Manhattan  Indian           Agra Restaurant       40375376         28
2     Queens  Indian  Annam Braham Restaurant       40380520         13
3  Manhattan  Indian                 Gandhi       40381295         92
4   Brooklyn  Indian  India Passage Restaurant       40384479         16
```

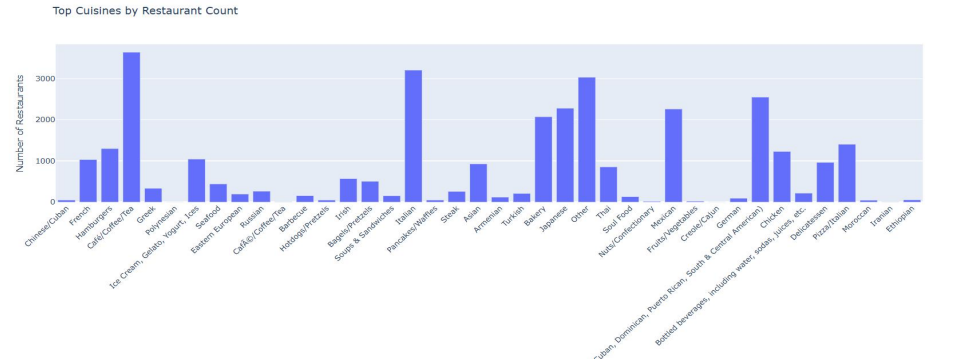


11. Show frequency count of restaurants by cuisines names.

```
mongo_query=[
{
"$group":{"_id":"$cuisine","freq":{"$sum":1}}
},
{
"$project":{
    "_id":0,
    "cuisine": "$_id",
    "frequency": "$freq"}},
 {
      "$sort": { "freq": -1 }
    },
    {
      "$limit":40
    }
]
results= list(collection.aggregate(mongo_query))
df=pd.DataFrame(results)
print(results)

fig = px.bar(df, x='cuisine', y='frequency', title='Top Cuisines by Restaurant Count', labels={'cuisine': 'Cuisine', 'frequency': 'Number of Restaurants'},
             height=700)
fig.update_layout(xaxis_tickangle=-45)
fig.show()
```

[{'cuisine': 'Chinese/Cuban', 'frequency': 48}, {'cuisine': 'French', 'frequency': 1032}, {'cuisine': 'Hamburgers', 'frequency': 1299}, {'cuisine': 'Café/Coffee/Tea', 'frequency': 3642}, {'cuisine': 'Greek', 'frequency': 333}, {'cuisine': 'Polynesian', 'frequency': 3}, {'cuisine': 'Ice Cream, Gelato, Yogurt, Ices', 'frequency': 1044}, {'cuisine': 'Seafood', 'frequency': 441}, {'cuisine': 'Eastern European', 'frequency': 195}, {'cuisine': 'Russian', 'frequency': 264}, {'cuisine': 'Café/Coffee/Tea', 'frequency': 6}, {'cuisine': 'Barbecue', 'frequency': 156}, {'cuisine': 'Hotdogs/Pretzels', 'frequency': 48}, {'cuisine': 'Irish', 'frequency': 570}, {'cuisine': 'Bagels/Pretzels', 'frequency': 504}, {'cuisine': 'Soups & Sandwiches', 'frequency': 153}, {'cuisine': 'Italian', 'frequency': 3207}, {'cuisine': 'Pancakes/Waffles', 'frequency': 48}, {'cuisine': 'Steak', 'frequency': 258}, {'cuisine': 'Asian', 'frequency': 927}, {'cuisine': 'Armenian', 'frequency': 120}, {'cuisine': 'Turkish', 'frequency': 210}, {'cuisine': 'Bakery', 'frequency': 2073}, {'cuisine': 'Japanese', 'frequency': 2280}, {'cuisine': 'Other', 'frequency': 3033}, {'cuisine': 'Thai', 'frequency': 855}, {'cuisine': 'Soul Food', 'frequency': 132}, {'cuisine': 'Nuts/Confectionary', 'frequency': 18}, {'cuisine': 'Mexican', 'frequency': 2262}, {'cuisine': 'Fruits/Vegetables', 'frequency': 21}, {'cuisine': 'Creole/Cajun', 'frequency': 3}, {'cuisine': 'German', 'frequency': 93}, {'cuisine': 'Latin (Cuban, Dominican, Puerto Rican, South & Central American)', 'frequency': 2550}, {'cuisine': 'Pizza/Italian', 'frequency': 1404}, {'cuisine': 'Moroccan', 'frequency': 54}, {'cuisine': 'Ethiopian', 'frequency': 54}]



12. Write a MongoDB query to find the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'bi' as last three letters for its name.

```
# Write a MongoDB query to find the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'bi' as last three letters for its name.
mongo_query=[
{
"$match":{"name":{"$regex": "Bi$", "$options": "i"}
}},
    {
        "$project":{
        "_id":0,
        "restaurant_id":1,
        "name":1,
        "borough":1,
        "cuisine":1,
        "score": "$grades.score",
        }
    }
]
results= list(collection.aggregate(mongo_query))
df=pd.DataFrame(results)
print(df.head())
```

```
     borough   cuisine           name restaurant_id             score
0     Queens   Chinese      Mr Wasabi      40997900  [10, 12, 10, 12]
1     Queens  Japanese       Gowasabi      41431748         [7, 13, 9]
2   Brooklyn  Japanese         Wasabi      50002381   [13, 18, 2, 31]
3   Brooklyn    Korean  Little Dokebi      50003236          [11, 11]
4  Manhattan  Japanese         Hanabi      50010310             [9]
```

13. Write a MongoDB query to find the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' as last three letters for its name.

```
#Write a MongoDB query to find the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' as last three letters for its name.
mongo_query=[
{
    "$match":{"name":{"$regex":"il$","$options":"i"}}
},
    {
        "$project":{
        "_id":0,
        "restaurant_id":1,
        "name":1,
        "borough":1,
        "cuisine":1,
        "score": "$grades.score",
        }
    }
 ]
results= list(collection.aggregate(mongo_query))
df=pd.DataFrame(results)
print(df.head())
```

```
     borough    cuisine                      name restaurant_id  \
0  Manhattan  Brazilian                Via Brazil      40682609
1   Brooklyn  Caribbean  D & P Restaurant & Cocktail      40796794
2  Manhattan  Caribbean                   Negril      40879743
3   Brooklyn  Caribbean          Sip N Chat Cocktail      40927832
4  Manhattan     French              Cafe Du Soleil      41083927

                score
0   [9, 12, 4, 18, 24]
1  [10, 12, 9, 35, 13]
2  [18, 5, 13, 13, 10]
3     [8, 2, 4, 14, 9]
4      [11, 13, 26, 9]
```

14. Write a query to show all the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' anywhere in its name.

```
#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.
mongo_query=[
{
        "$match": { "grades.score": { "$mod": [7, 0] }  # score % 7 == 0
        }
    },
    { "$project": {
            "_id": 0,
            "restaurant_id": 1,
            "name": 1,
            "score": "$grades.score"
        }
    }
]
results= list(collection.aggregate(mongo_query))
df=pd.DataFrame(results)
print(df.head())
```

```
                        name restaurant_id                 score
0       Morris Park Bake Shop      30075445    [2, 6, 10, 9, 14]
1             Riviera Caterer      40356018       [5, 7, 12, 12]
2     Brunos On The Boulevard      40356151      [38, 10, 7, 13]
3             May May Kitchen      40358429  [21, 7, 56, 27, 27]
4  1 East 66Th Street Kitchen      40359480         [3, 4, 6, 0]
```

15. 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.

```
# Write a query to show all the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' anywhere in its name.
mongo_query=[
{
"$match":{"name":{"$regex":"il","$options":"i"}}
},
    {
        "$project":{
        "_id":0,
        "restaurant_id":1,
        "name":1,
        "borough":1,
        "cuisine":1,
        "score": "$grades.score",
        }
    }
]
results= list(collection.aggregate(mongo_query))
df=pd.DataFrame(results)
print(df.head())
```

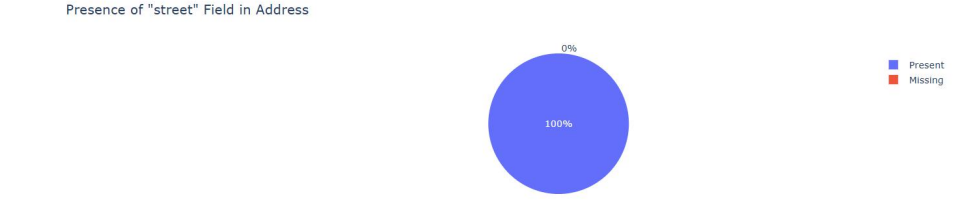
```
      borough       cuisine                  name restaurant_id  \
0   Brooklyn  Delicatessen    Wilken'S Fine Food      40356483
1       Bronx      American             Wild Asia      40357217
2   Manhattan      American   Angelika Film Center      40362274
3      Queens      American       Snack Time Grill      40363590
4   Brooklyn       Chinese      Golden Pavillion      40363920

                      score
0  [10, 10, 8, 10, 13, 9]
1             [11, 4, 3]
2          [9, 4, 13, 5]
3    [7, 11, 7, 10, 0, 4]
4        [13, 10, 4, 13]
```

16. Show document/record counts that has street and not street in addresses.

```
has_street = collection.count_documents({"address.street": {"$exists": True}})
missing_street = collection.count_documents({"address.street": {"$exists": False}})
df = pd.DataFrame({
    'status': ['Present', 'Missing'],
    'count': [has_street, missing_street]
})
print(df)
fig = px.pie(df,
             names='status',
             values='count',
             title='Presence of "street" Field in Address')
fig.show()
```

```
    status  count
0  Present  76077
1  Missing      0
```



17. 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

```
mongo_query = [
    {
        "$match": {"cuisine": { "$ne": "American" }, "grades.score": { "$gt": 70 },"address.coord.0": { "$lt": 65.754168 }
        }},
    {"$project": {
            "_id": 0,
            "restaurant_id": 1,
            "name": 1,
            "borough": 1,
            "cuisine": 1,
            "score": "$grades.score",
            "longitude": "$address.coord.0"
        }
    }
]
results= list(collection.aggregate(mongo_query))
df=pd.DataFrame(results)
print(df.head())
fig = px.bar(df.head(20),  x='name', y='score', color='cuisine', title='Top Restaurants (Non-American, Score > 70, Longitude < 65.75)',
             labels={'name': 'Restaurant Name', 'score': 'Score'},
             height=500)
fig.update_layout(xaxis_tickangle=-45)
fig.show()
```

```
     borough                                        cuisine  \
0  Manhattan                                       American
1  Manhattan                                         Indian
2  Manhattan                                       American
3  Manhattan                                  Pizza/Italian
4      Bronx  Latin (Cuban, Dominican, Puerto Rican, South &...

                        name restaurant_id                        score  \
0   Murals On 54/Randolphs'S      40372466       [11, 131, 11, 25, 11, 13]
1                    Gandhi       40381295         [5, 8, 12, 2, 9, 92, 41]
2  Live Bait Bar & Restaurant      40387237              [9, 12, 58, 13, 71]
3              Bella Napoli        40393488            [31, 98, 32, 21, 11]
4   El Molino Rojo Restaurant      40393688  [10, 6, 25, 12, 12, 14, 26, 76]

  longitude
0        []
1        []
2        []
3        []
4        []
```