**Exercise - 5**

**MongoDB Restaurant Analysis**

import json

from pymongo import MongoClient

if \_\_name\_\_ == "\_\_main\_\_":

client = MongoClient("mongodb://localhost:27017")

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

db=client['restaurant']

collection=db['rescollection']

with open("C:/Users/amelgirx/restaurants-dataset.json","r",encoding="utf-8")as file:

record=file.read()

record=record.replace("\n","")

record=record.replace("}{","},{")

record="["+record+"]"

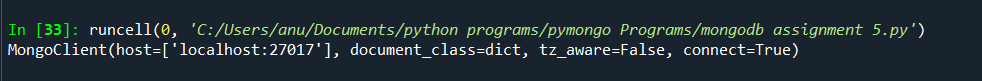
file\_data=json.loads(record)

if isinstance(file\_data, list):

collection.insert\_many(file\_data)

else:

collection.insert\_one(file\_data)

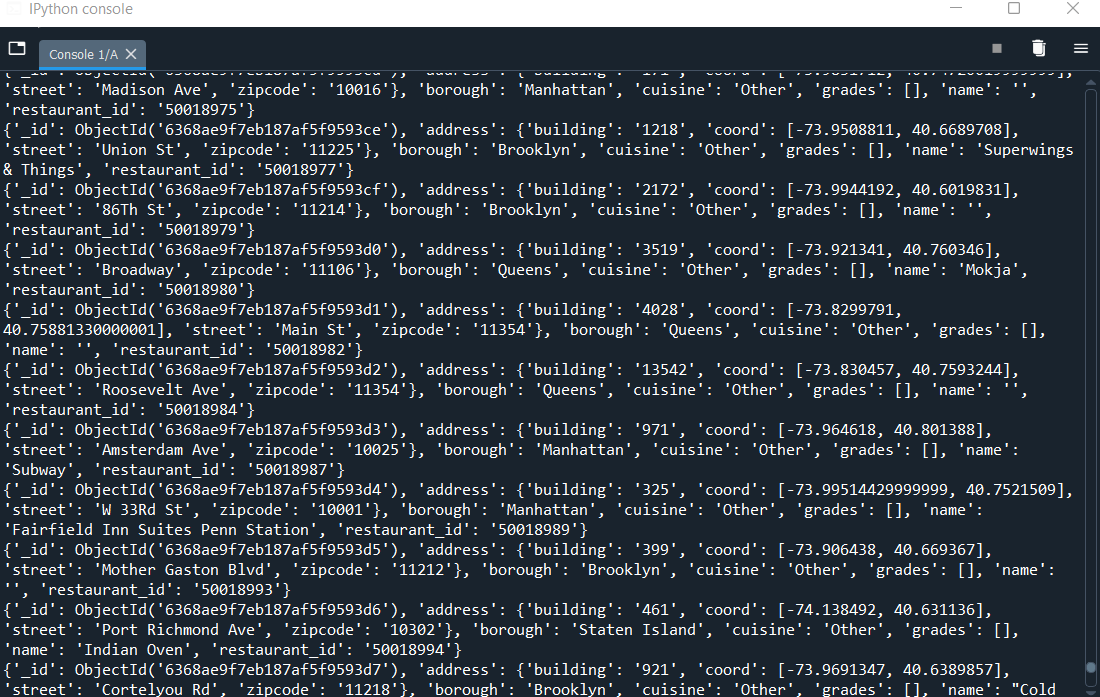


**# 2.Display all the documents in the collection restaurants**.

all\_documents=collection.find({})

for i in all\_documents:

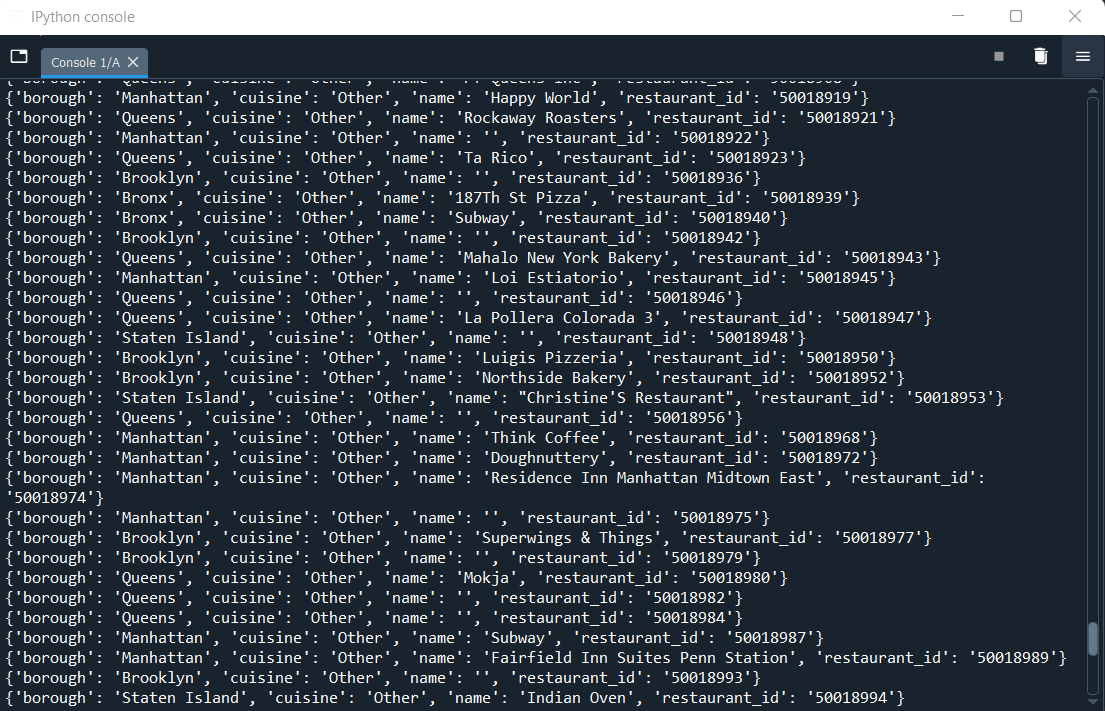
print(i)



**# 3.Display the fields restaurant\_id, name, borough, and zip code, but exclude the field \_id for all the documents in the collection restaurant.** all\_documents=collection.find({},{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'zipcode':1})

for i in all\_documents:

print(i)

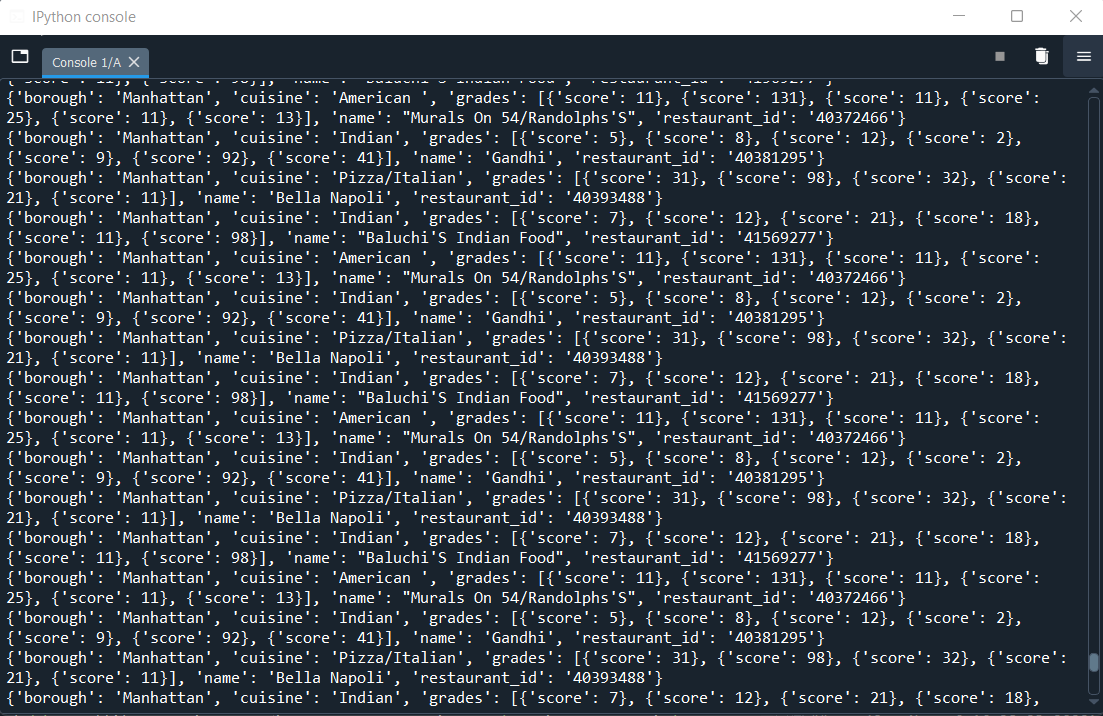


**# 4.Find the restaurants who achieved a score more than 90.**

all\_documents=collection.find({"grades.score":{'$gt':90}},{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)



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

all\_documents=collection.find({'$and':[{"grades.score":{'$gt':80}},{"grades.score":{'$lt':100}}]},{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)

Text

Description automatically generated

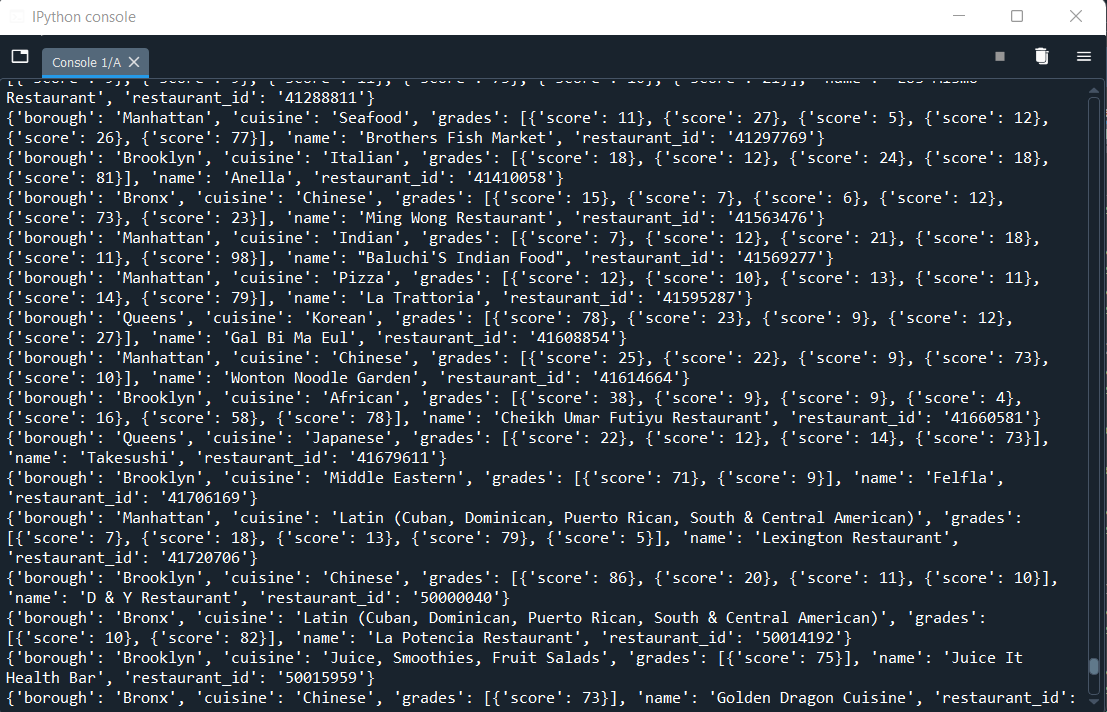
**# 6. Write Query to show the restaurants that do not prepare any cuisine of american & their grade score > 70.**

all\_documents=collection.find({'$and':[{"grades.score":{'$gt':70}},{"cuisine":{'$ne':'American '}}]},

{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)

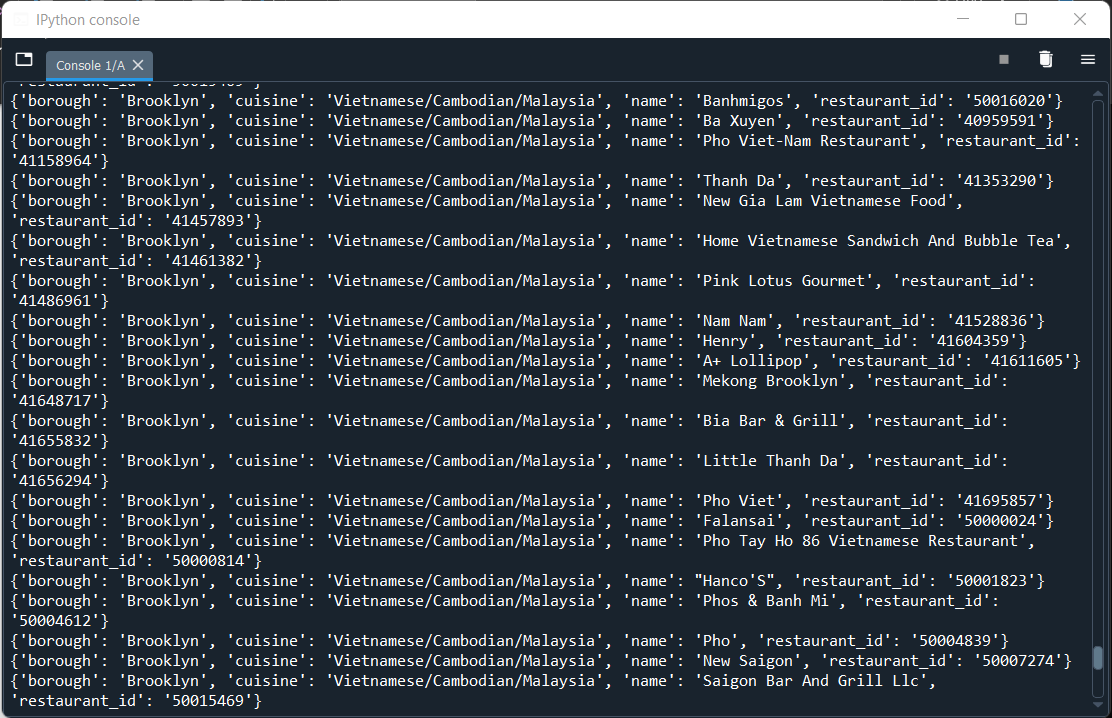


**# 7.Write a MongoDB query to arrange the name of the cuisine in an ascending order and for that same borough arranged in descending order.**

all\_documents=collection.find({},{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1}).sort([('cuisine',1),('borough',-1)])

for i in all\_documents:

print(i)

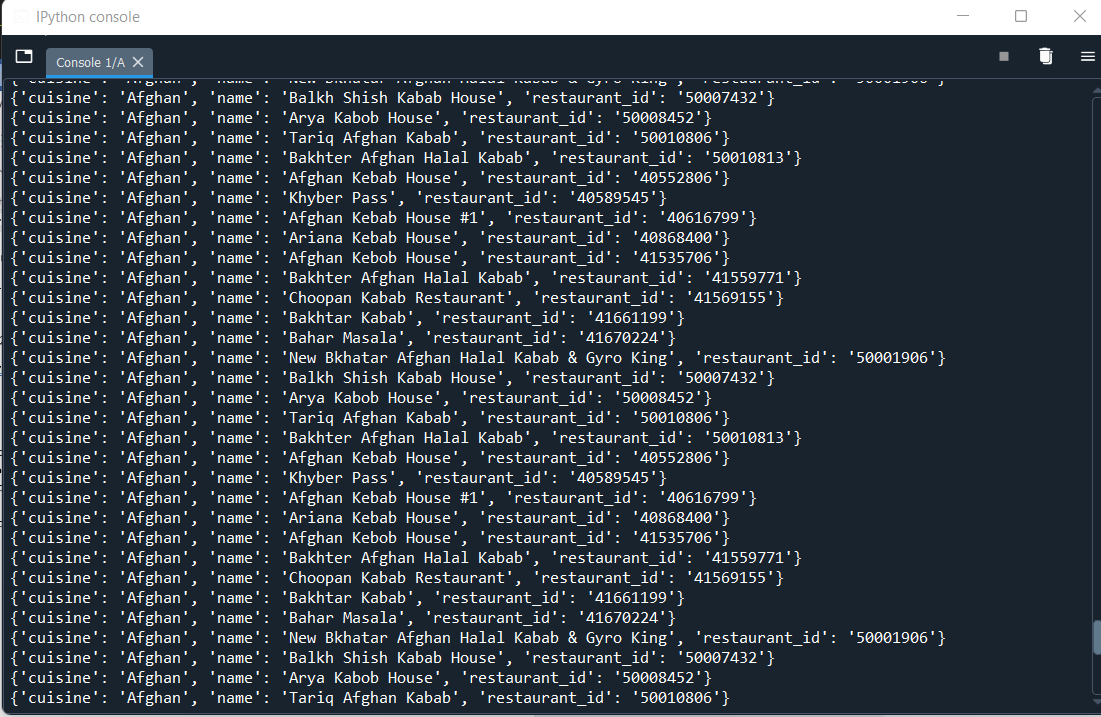


**# 8.Write a MongoDB query to arrange the name of the cuisine in descending order..**

all\_documents=collection.find({},{'\_id':0,'name':1,'restaurant\_id':1,'cuisine':1}).sort('cuisine',-1)

for i in all\_documents:

print(i)



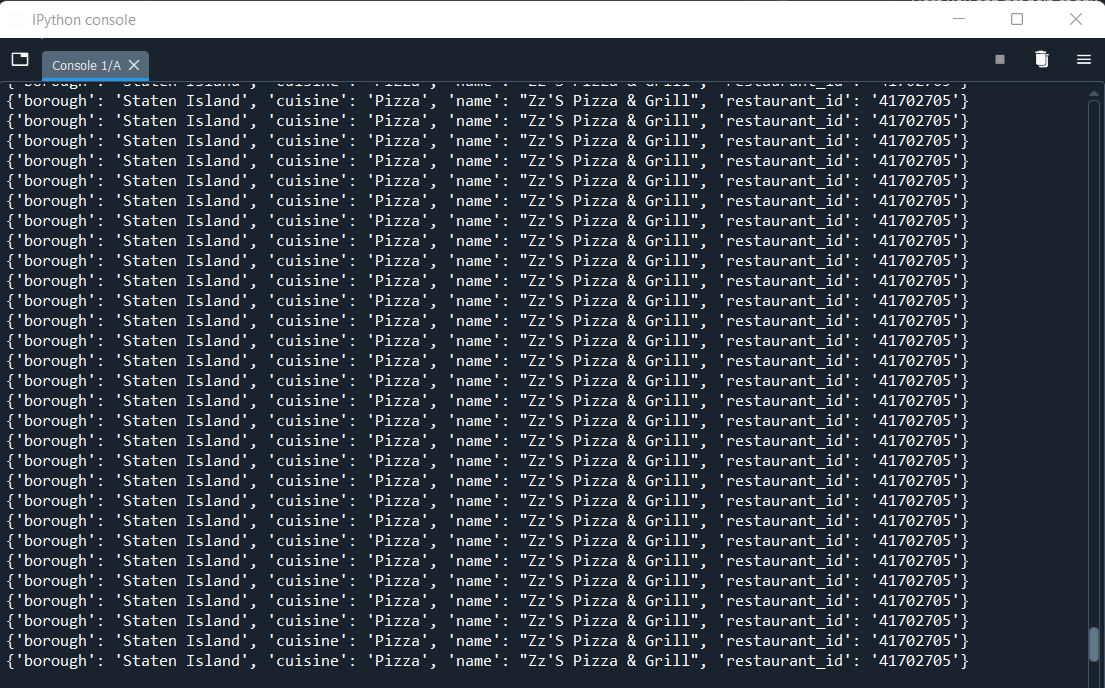
**# 9.Show the restaurant Id, name, borough and cuisines for those restaurants which prepared dish except 'American' and 'Chinese' or restaurant's name begins with letter 'Bil'.**

all\_documents=collection.find({'$or':[{"cuisine":{'$nin':['American ','Chinese']}},{"name":{'$regex':'^Bil'}}]},

{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1}).sort('name',1)

for i in all\_documents:

print(i)

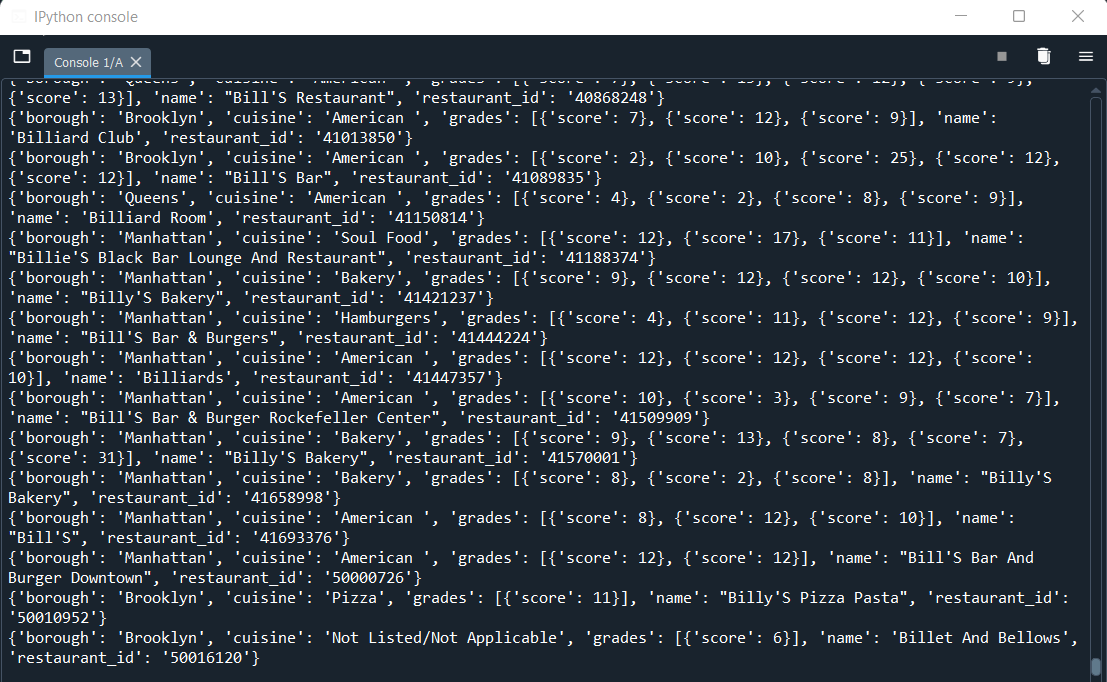


**# 10. Show the restaurant Id, name, borough and cuisines and score for restaurant's name begins with letter 'Bil'.**

all\_documents=collection.find({"name":{'$regex':'^Bil'}},{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)



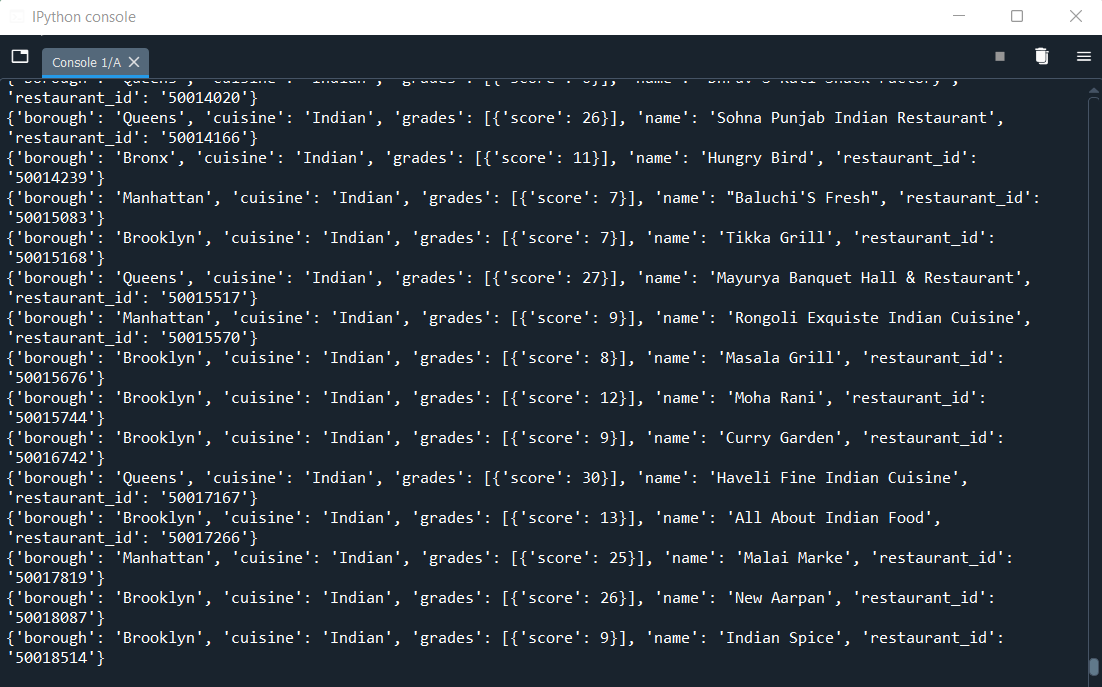
**# 11.Show the restaurant Id, name, borough and cuisines and score for restaurant serving “Indian” as cuisines.**

all\_documents=collection.find({"cuisine":'Indian'},

{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)



**# 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.**

all\_documents=collection.find({"name":{'$regex':'bi$'}},

{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)

Text

Description automatically generated

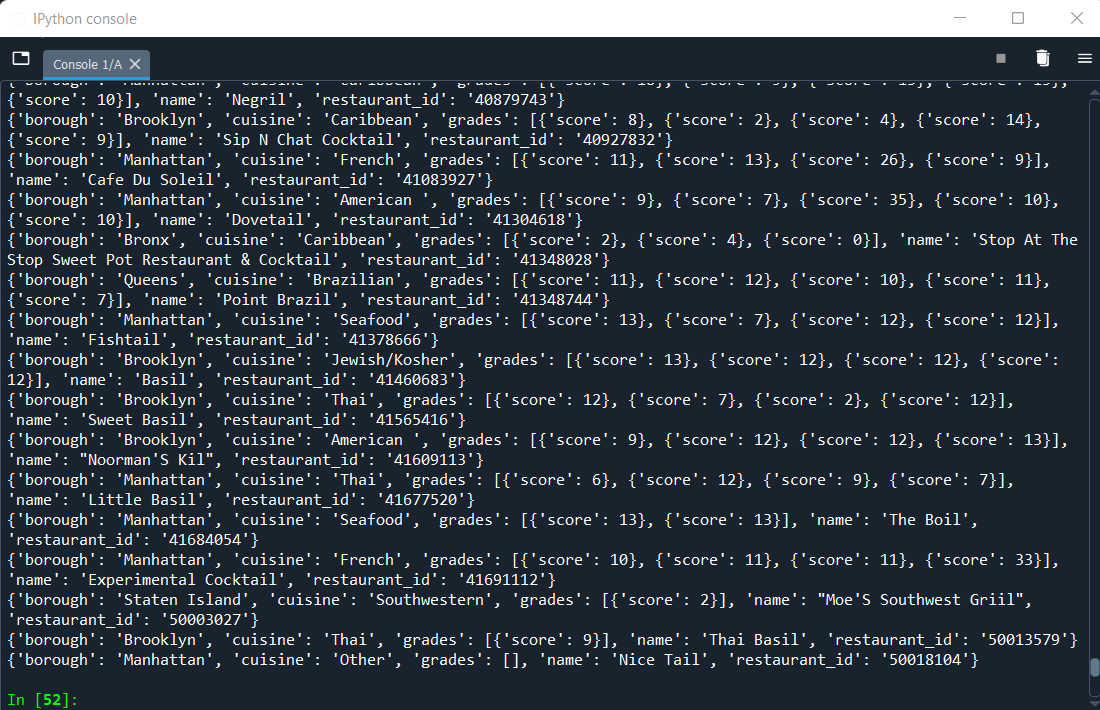
**# 13.Write a MongoDB query to find the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' as last three letters for its name.**

all\_documents=collection.find({"name":{'$regex':'il$'}},

{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)



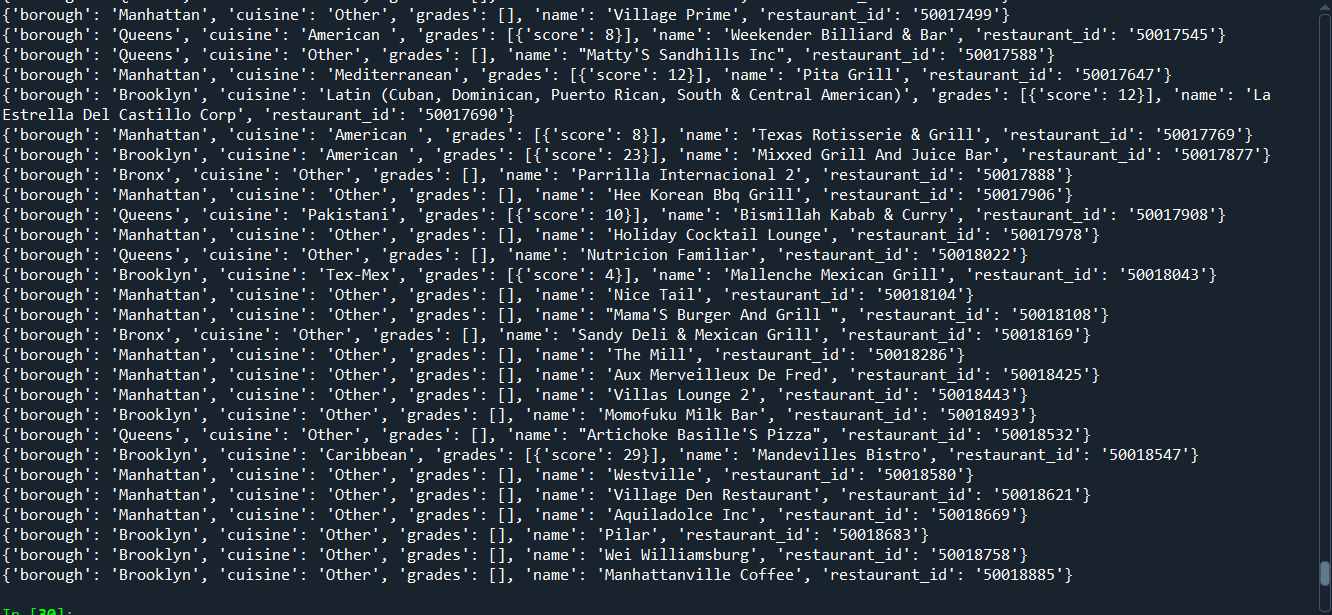
**# 14.Write a query to show all the restaurant Id, name, borough, cuisines, and score for those restaurants which contain 'il' anywhere in its name.**

all\_documents=collection.find({"name":{'$regex':'.\*il.\*'}},

{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)

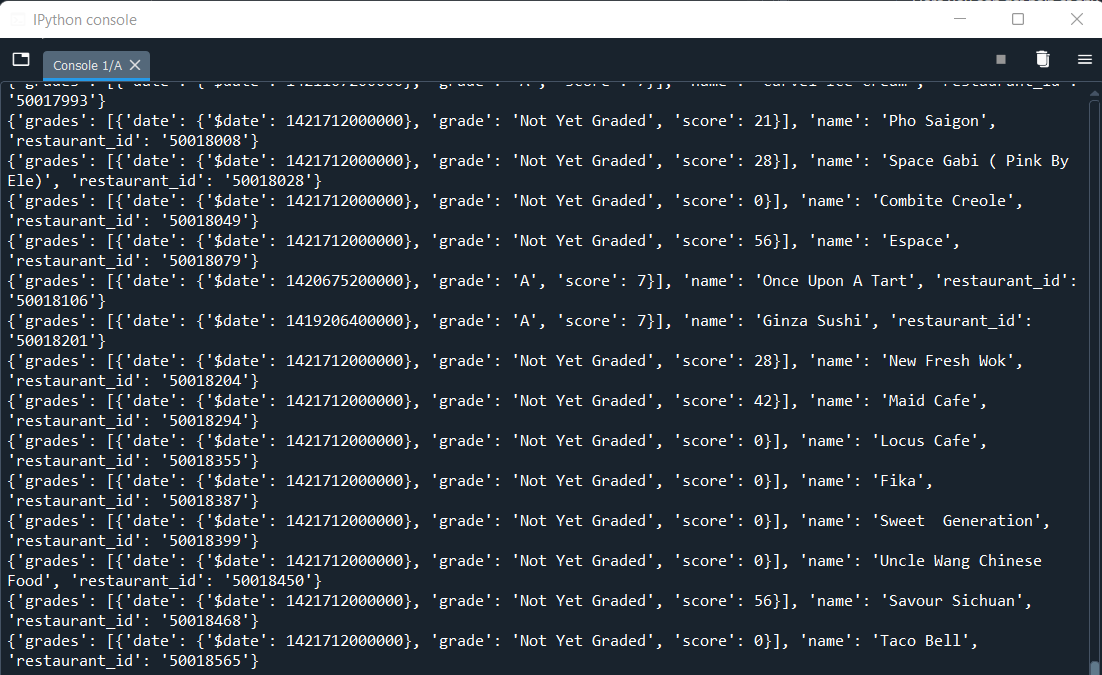


**# 15.Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.**

all\_documents=collection.find({"grades.score" :{'$mod' : [7,0]}},{"\_id":0,"restaurant\_id" : 1,"name":1,"grades":1})

for i in all\_documents:

print(i)



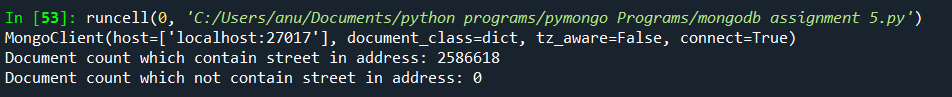
**# 16.Show document/record counts that has street and not street in addresses.**

street\_in\_address\_count=collection.count\_documents({'address.street' :{'$exists' : 1}})

print("Document count which contain street in address:",street\_in\_address\_count)

street\_not\_in\_address\_count=collection.count\_documents({'address.street' :{'$exists' : 0}})

print("Document count which not contain street in address:",street\_not\_in\_address\_count)



**# 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**

all\_documents=collection.find({'$and':[{"cuisine":{'$ne':'American '}},{"grades.score":{'$gt':70}},{"address.coord" : {'$lt' : -65.754168}}]},

{'\_id':0,'name':1,'restaurant\_id':1,'borough':1,'cuisine':1,'grades.score':1})

for i in all\_documents:

print(i)

