Project by Faraz

Table of Content

1. Introduction:

Venturing into the Pakistani cuisine market offers promising opportunities for restaurant entrepreneurs who harness data-driven insights to make informed decisions. By analyzing the availability of Pakistani cuisine in various boroughs, assessing competition, evaluating food quality, targeting niche markets, choosing the best location, and developing innovative concepts, entrepreneurs can optimize their restaurant venture. Leveraging data analysis in these areas enables them to align with customer preferences and demands, ultimately increasing the likelihood of success and profitability in their restaurant business.

2. Business Question(s):

- What is the count of restaurants for Pakistani cuisine and borough?
- What is the average score for each cuisine?
- What is the average score for each borough?

3. Business Objective(s):

Venturing into the Pakistani cuisine market as a restaurant entrepreneur can be a lucrative opportunity if you leverage data-driven insights to make informed decisions. Here are some ways to use data analysis to optimize your restaurant venture:

Identify underserved cuisines: Analyze the availability of Pakistani cuisine in various boroughs to determine if there is an unmet demand for this type of food. Look for areas with limited or no Pakistani restaurants, suggesting a potential market gap.

Assess competition: Determine the number of Pakistani restaurants in the target area and identify their strengths and weaknesses. This analysis will help you gauge the competition and better position your restaurant to differentiate itself.

Evaluate food quality: Research customer reviews and food quality scores of existing Pakistani restaurants in the area. This information will help you understand customer preferences, which can guide your menu and pricing decisions.

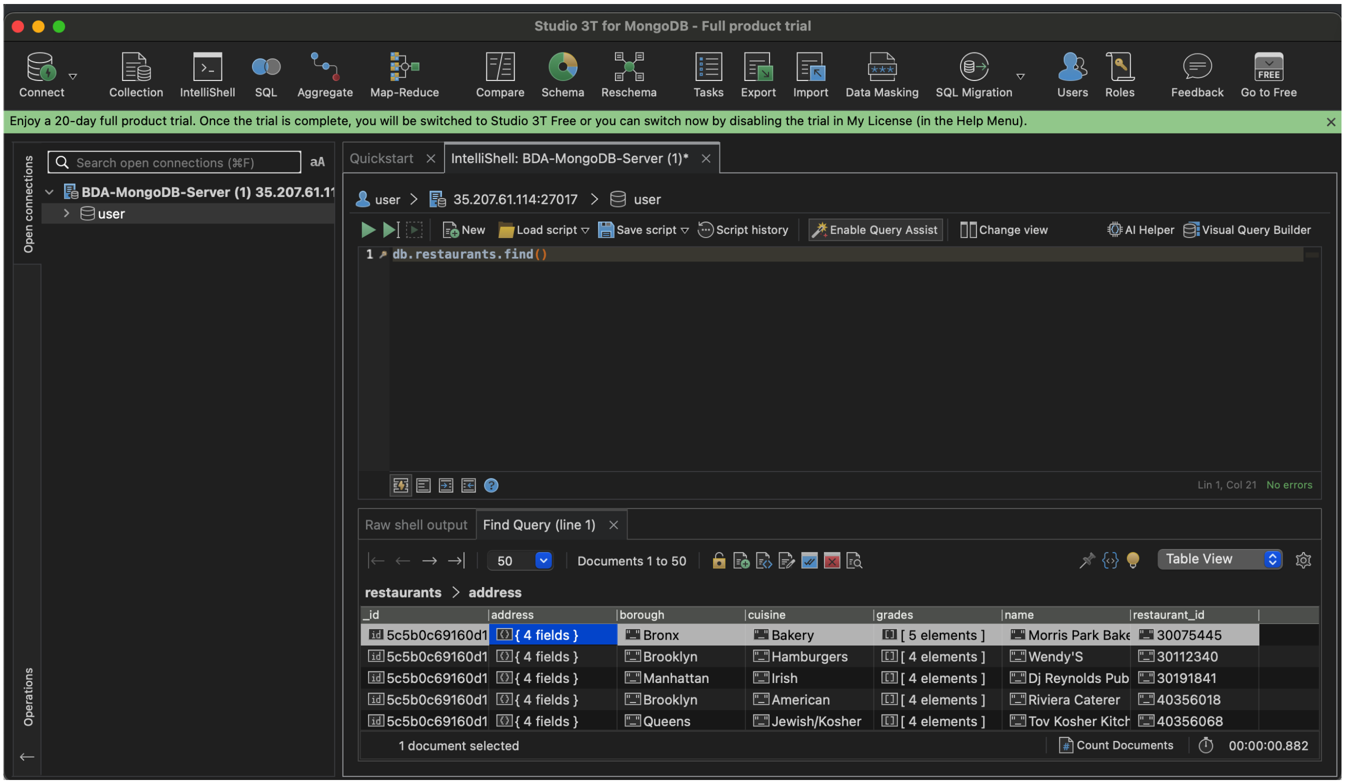
Target niche markets: Identify specific sub-cuisines or dishes within Pakistani cuisine that are underrepresented or in high demand, such as regional specialties or fusion dishes. This strategy can help you cater to a specific market segment and attract a loyal customer base.

Choose the best location: Analyze the demographics, foot traffic, and accessibility of potential locations to find the best fit for a Pakistani restaurant. Consider factors like proximity to public transportation, parking availability, and nearby attractions.

Develop innovative concepts: Use data on customer preferences and industry trends to create unique and innovative restaurant concepts that cater to the target market's tastes. This could include offering a modern twist on traditional dishes, incorporating sustainable practices, or providing a memorable dining experience through ambiance and service.

By leveraging data analysis in these areas, entrepreneurs can increase their likelihood of success when venturing into the Pakistani cuisine market. This data-driven approach will enable them to create establishments that align with customer preferences and demands, ultimately contributing to a successful and profitable restaurant business.

4. Data Exploration:



Enjoy a 20-day full product trial. Once the trial is complete, you will be switched to Studio 3T Free or you can switch now by disabling the trial in My License (in the Help Menu).

BDA-MongoDB-Server (1) 35.207.61.11
user

user > 35.207.61.114:27017 > user

```
1 db.restaurants.find()
```

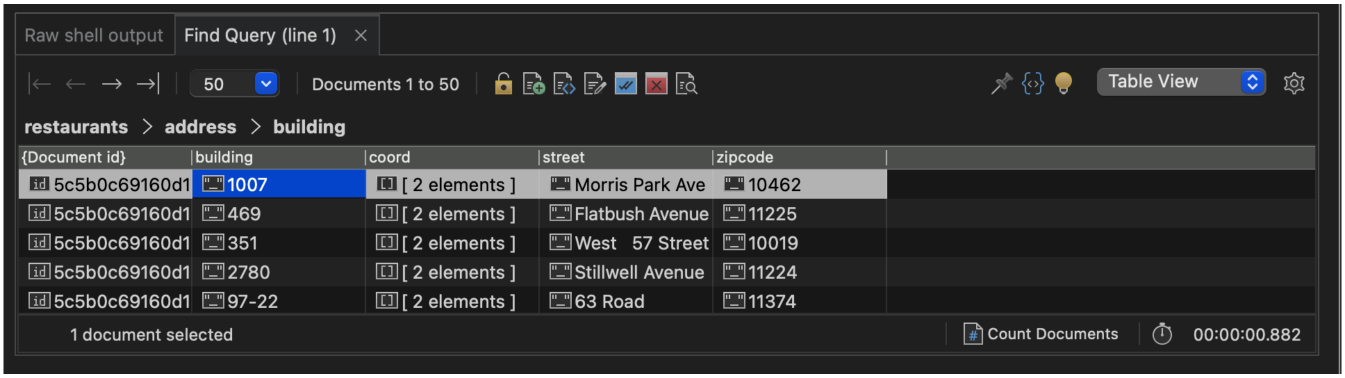
50 Documents 1 to 50

restaurants > address

_id	address	borough	cuisine	grades	name	restaurant_id
5c5b0c69160d1	{ 4 fields }	Bronx	Bakery	[5 elements]	Morris Park Bake	30075445
5c5b0c69160d1	{ 4 fields }	Brooklyn	Hamburgers	[4 elements]	Wendy'S	30112340
5c5b0c69160d1	{ 4 fields }	Manhattan	Irish	[4 elements]	Dj Reynolds Pub	30191841
5c5b0c69160d1	{ 4 fields }	Brooklyn	American	[4 elements]	Riviera Caterer	40356018
5c5b0c69160d1	{ 4 fields }	Queens	Jewish/Kosher	[4 elements]	Tov Kosher Kitch	40356068

1 document selected   Count Documents   00:00:00.882

This code is a MongoDB query that retrieves all documents from the "restaurants" collection. In the context of a database with restaurant information, this query would return all the records stored in the "restaurants" collection without any filtering or sorting. It's equivalent to saying "find all the restaurant documents in the database."



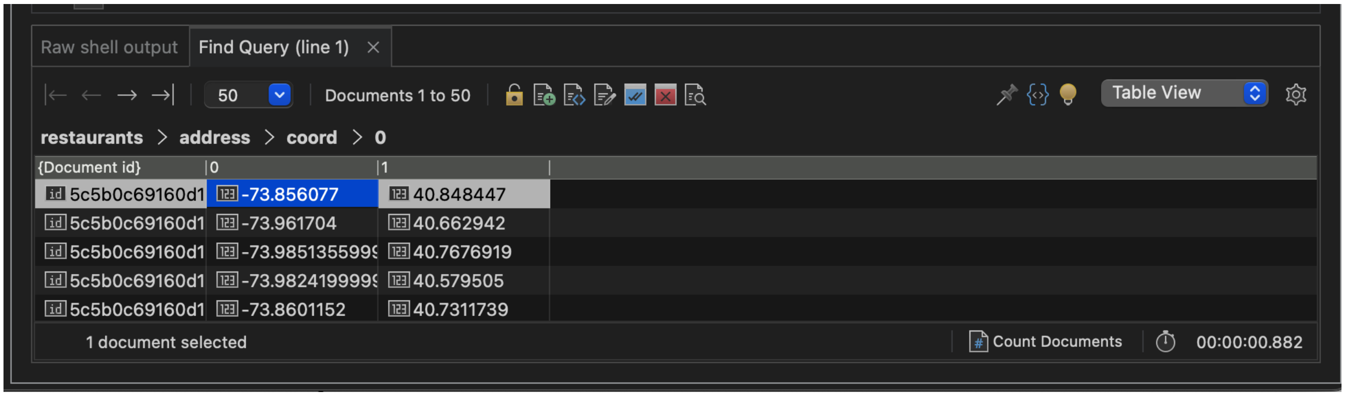
50 Documents 1 to 50

restaurants > address > building

{Document id}	building	coord	street	zipcode
5c5b0c69160d1	1007	[2 elements]	Morris Park Ave	10462
5c5b0c69160d1	469	[2 elements]	Flatbush Avenue	11225
5c5b0c69160d1	351	[2 elements]	West 57 Street	10019
5c5b0c69160d1	2780	[2 elements]	Stillwell Avenue	11224
5c5b0c69160d1	97-22	[2 elements]	63 Road	11374

1 document selected   Count Documents   00:00:00.882

We can then look into the fields to explore the data, the address has building, coord, street, zipcode.

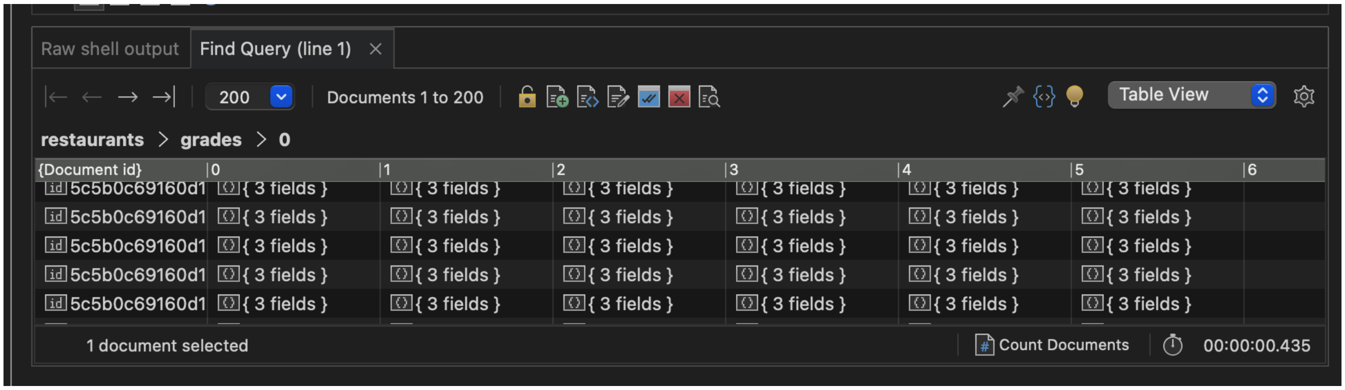


50 Documents 1 to 50

restaurants > address > coord > 0

{Document id}	0	1
5c5b0c69160d1	-73.856077	40.848447
5c5b0c69160d1	-73.961704	40.662942
5c5b0c69160d1	-73.9851355999	40.7676919
5c5b0c69160d1	-73.9824199999	40.579505
5c5b0c69160d1	-73.8601152	40.7311739

1 document selected   Count Documents   00:00:00.882



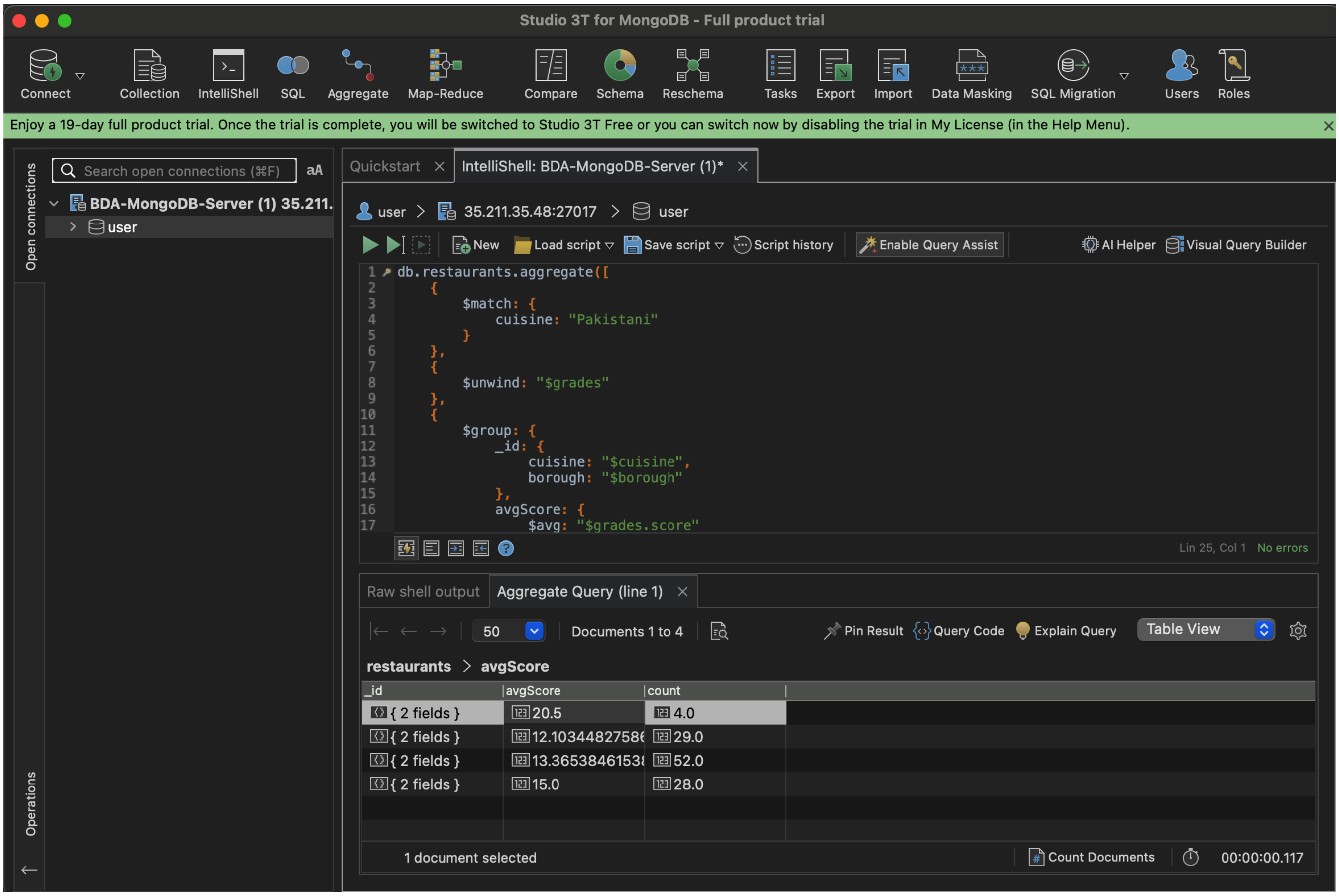
restaurants > grades > 0

{Document id}	0	1	2	3	4	5	6
5c5b0c69160d1	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	
5c5b0c69160d1	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	
5c5b0c69160d1	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	
5c5b0c69160d1	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	
5c5b0c69160d1	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	{ 3 fields }	

1 document selected

Count Documents 00:00:00.435

5. Database Queries & Results:



Enjoy a 19-day full product trial. Once the trial is complete, you will be switched to Studio 3T Free or you can switch now by disabling the trial in My License (in the Help Menu).

BDA-MongoDB-Server (1) 35.211.
user

user > 35.211.35.48:27017 > user

```
db.restaurants.aggregate([
    {
        $match: {
            cuisine: "Pakistani"
        }
    },
    {
        $unwind: "$grades"
    },
    {
        $group: {
            _id: {
                cuisine: "$cuisine",
                borough: "$borough"
            },
            avgScore: {
                $avg: "$grades.score"
```

Lin 25, Col 1 No errors

restaurants > avgScore

_id	avgScore	count
{ 2 fields }	20.5	4.0
{ 2 fields }	12.1034482758	29.0
{ 2 fields }	13.3653846153	52.0
{ 2 fields }	15.0	28.0

1 document selected

00:00:00.117

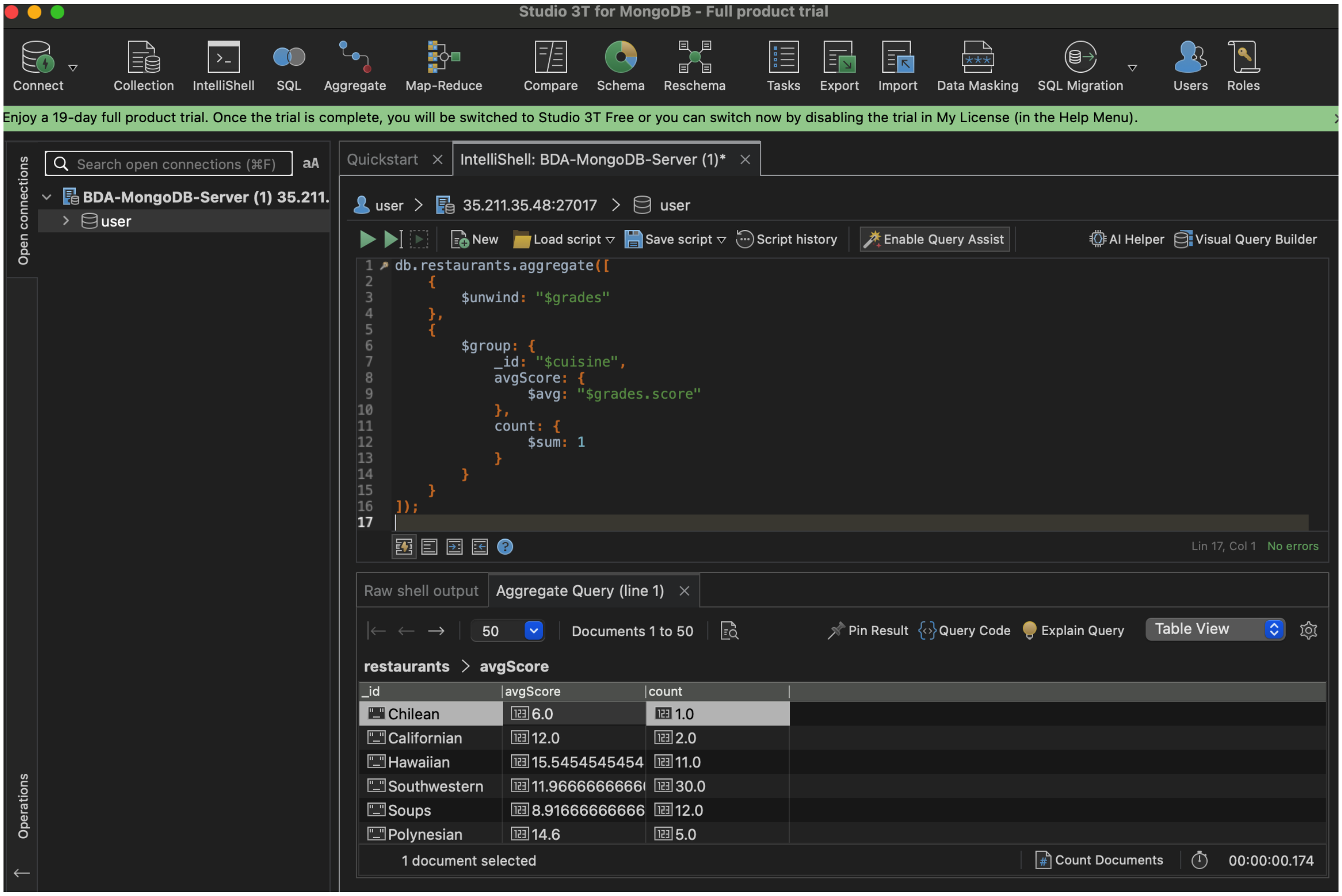
This code is an aggregation pipeline operating on the "restaurants" collection in a MongoDB database. The pipeline has three stages:

$match stage: Filters the documents in the collection to only include those with a "cuisine" field equal to "Pakistani".

$unwind stage: Unwinds the "grades" array in the documents, creating a new document for each element in the "grades" array. This effectively flattens the array, so each document in the output of this stage will have a single grade entry instead of an array of grades.

$group stage: Groups the documents by their "cuisine" and "borough" fields. The resulting documents will have a composite _id field containing both "cuisine" and "borough". For each group, the pipeline calculates the average of the "grades.score" field (stored in the avgScore field) and the total count of documents in the group (stored in the count field).

The output of the pipeline will be a list of documents, each representing a unique combination of "Pakistani" cuisine and a borough. Each document will contain the average score and the number of restaurants with "Pakistani" cuisine in that borough.



```
db.restaurants.aggregate([
    {
        $unwind: "$grades"
    },
    {
        $group: {
            _id: "$cuisine",
            avgScore: {
                $avg: "$grades.score"
            },
            count: {
                $sum: 1
            }
        }
    }
]);
```

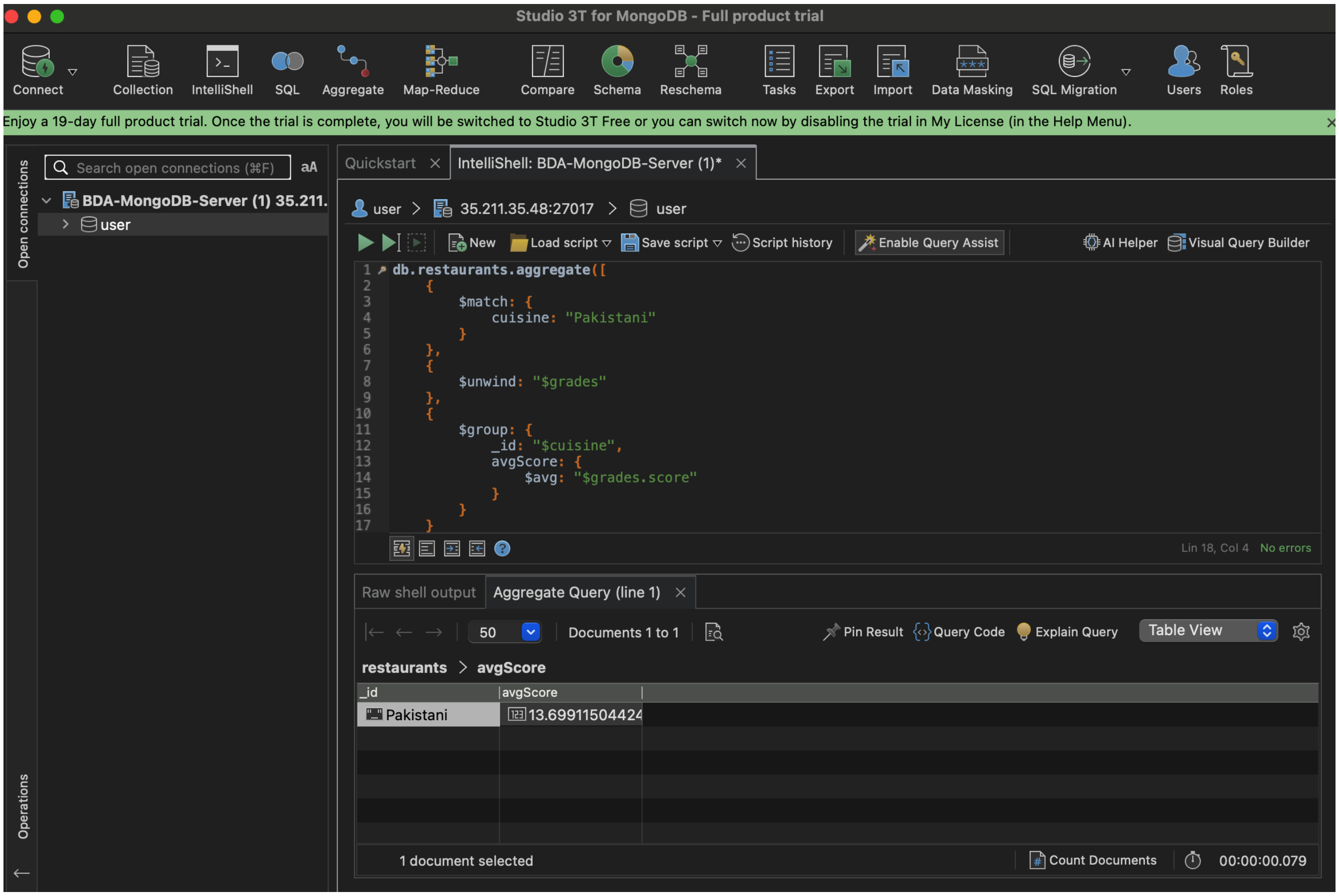
_id	avgScore	count
Chilean	6.0	1.0
Californian	12.0	2.0
Hawaiian	15.5454545454	11.0
Southwestern	11.96666666666	30.0
Soups	8.91666666666	12.0
Polynesian	14.6	5.0

This aggregation pipeline operates on the "restaurants" collection in the MongoDB database. The pipeline consists of two stages:

$unwind stage: This stage unwinds the "grades" array, which means it creates a new document for each element in the "grades" array, effectively flattening the array. As a result, each document in the output of this stage will have a single grade entry instead of an array of grades.

$group stage: This stage groups the documents based on their "cuisine" field. For each unique cuisine, it calculates the average of the "grades.score" field (stored in the avgScore field) and the total count of documents in that group (stored in the count field).

The output of the pipeline will be a list of documents, each representing a unique cuisine type with its corresponding average score and the number of restaurants with that cuisine.



Enjoy a 19-day full product trial. Once the trial is complete, you will be switched to Studio 3T Free or you can switch now by disabling the trial in My License (in the Help Menu).

```
db.restaurants.aggregate([
    {
        $match: {
            cuisine: "Pakistani"
        }
    },
    {
        $unwind: "$grades"
    },
    {
        $group: {
            _id: "$cuisine",
            avgScore: {
                $avg: "$grades.score"
            }
        }
    }
```

Lin 18, Col 4 No errors

restaurants > avgScore

_id	avgScore
Pakistani	13.69911504424

1 document selected 00:00:00.079

This code is an aggregation pipeline operating on the "restaurants" collection in a MongoDB database. The pipeline consists of three stages:

$match stage: Filters the documents in the collection to only include those with a "cuisine" field equal to "Pakistani".

$unwind stage: Unwinds the "grades" array in the documents, creating a new document for each element in the "grades" array. This effectively flattens the array, so each document in the output of this stage will have a single grade entry instead of an array of grades.

$group stage: Groups the documents by their "cuisine" field. The resulting documents will have an _id field containing the "cuisine" value. For each group, the pipeline calculates the average of the "grades.score" field (stored in the avgScore field).

The output of the pipeline will be a list of documents, each representing a unique cuisine type (in this case, only "Pakistani" cuisine). Each document will contain the average score of the restaurants with that cuisine. Note that this code does not include the "borough" field in the grouping, and it does not calculate the count of restaurants.

6. Analysis:

In [28]:

```
!pip install pymongo==4.0
```

Loading [MathJax]/extensions/Safe.js

```
Collecting pymongo==4.0
  Downloading pymongo-4.0-cp39-cp39-macosx_10_9_x86_64.whl (351 kB)
     |████████████████████████████████| 351 kB 544 kB/s eta 0:00:01
Installing collected packages: pymongo
  Attempting uninstall: pymongo
    Found existing installation: pymongo 3.12.0
    Uninstalling pymongo-3.12.0:
      Successfully uninstalled pymongo-3.12.0
Successfully installed pymongo-4.0
```

In [34]:

```python
from pymongo import MongoClient
from pandas import json_normalize
import json
from pprint import pprint as pp
import pandas as pd
from datetime import datetime
```

In [2]:

```python
client = MongoClient('mongodb://user:DqZvY5Ch7@35.207.26.170:27017/user')
db=client.user
print (db)
```

```
Database(MongoClient(host=['35.207.26.170:27017'], document_class=dict, tz_aware=False,
connect=True), 'user')
```

In [65]:

```python
with client:
    db = client.user
    qresult = db.restaurants.aggregate([
        {
            "$match": {
                "cuisine": "Pakistani"
            }
        },
        {
            "$group": {
                "_id": {
                    "cuisine": "$cuisine",
                    "borough": "$borough"
                },
                "count": {
                    "$sum": 1
                }
            }
        }
    ])
    print(type(qresult))
```

```
<class 'pymongo.command_cursor.CommandCursor'>
```

In [66]:

```python
# store the cursor results,ie qresult into a python list, ie res

res=list(qresult)

# store the result list into a dataframe for analysis, etc
# the json_normalize funtion flattens any embedded documents

# note df is not flattended
df = pd.DataFrame.from_records(res)

# note df2 is flattened
df2 = pd.DataFrame.from_records(json_normalize(res))
```

In [67]:

```python
df2
```

Loading [MathJax]/extensions/Safe.js

Out[67]:

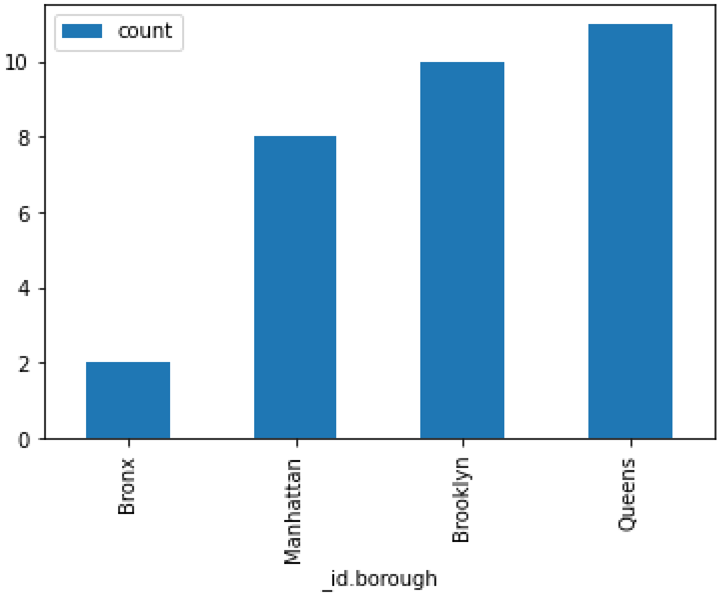
	count	_id.cuisine	_id.borough
0	2	Pakistani	Bronx
1	8	Pakistani	Manhattan
2	10	Pakistani	Brooklyn
3	11	Pakistani	Queens

In [61]:

```
df2.plot(kind='bar', x='_id.borough', y='count')
```

Out[61]:

```
<AxesSubplot:xlabel='_id.borough'>
```



In [76]:

```
with client:
    db = client.user
    qresult = db.restaurants.aggregate([
        {
            "$unwind": "$grades"
        },
        {
            "$group": {
                "_id": "$cuisine",
                "avgScore": { "$avg": "$grades.score" }
            }
        }
    ])

print(type(qresult))

res=list(qresult)
df2 = pd.DataFrame.from_records(json_normalize(res))
df2
```

```
<class 'pymongo.command_cursor.CommandCursor'>
```

Out[76]:

	_id	avgScore
0	Chilean	6.000000
1	Californian	12.000000
2	Hawaiian	15.545455
3	Southwestern	11.966667
4	Soups	8.916667
...	...	...
80	Hotdogs/Pretzels	5.588235
81	Chinese/Cuban	13.830508
82	Bottled beverages, including water, sodas, jui...	10.336245
83	Ethiopian	10.112903
84	Russian	12.968051

85 rows × 2 columns

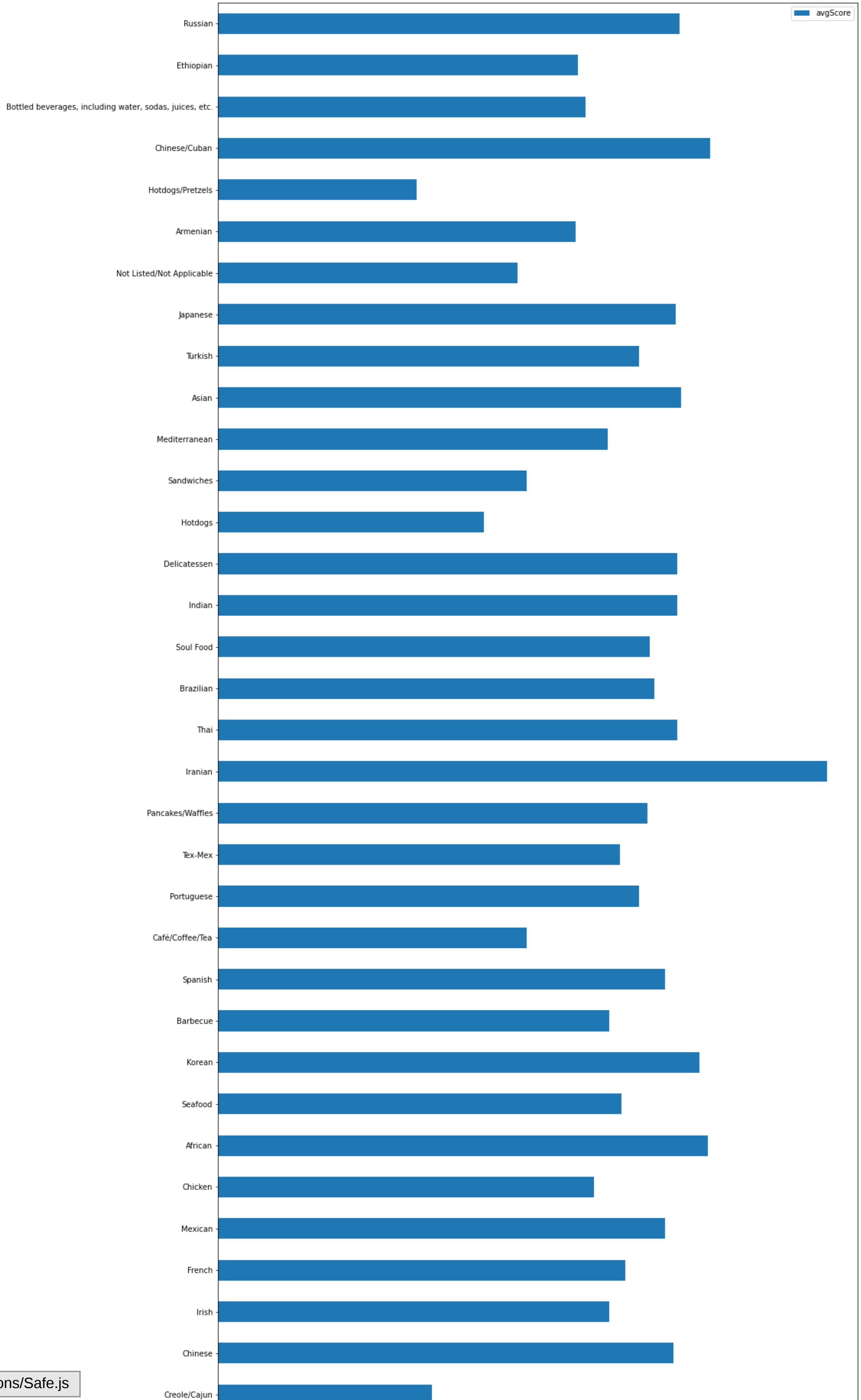
In [78]:

```
df2.plot(kind='barh', y="avgScore", x='_id', figsize=(15,85))
```

Out[78]:

```
<AxesSubplot:ylabel='_id'>
```

Loading [MathJax]/extensions/Safe.js



Loading [MathJax]/extensions/Safe.js

In [79]:

```python
with client:
    db = client.user
    qresult = db.restaurants.aggregate([
        {
            "$match": {
                "cuisine": "Pakistani"
            }
        },
        {
            "$unwind": "$grades"
        },
        {
            "$group": {
                "_id": "$cuisine",
                "avgScore": { "$avg": "$grades.score" }
            }
        }
    ])

    print(type(qresult))

    res = list(qresult)
    df2 = pd.DataFrame.from_records(json_normalize(res))
    print(df2)
```

```
<class 'pymongo.command_cursor.CommandCursor'>
         _id   avgScore
0  Pakistani  13.699115
```

In [81]:

```
with client:
    db = client.user
    qresult = db.restaurants.aggregate([
        {
            "$unwind": "$grades"
        },
        {
            "$group": {
                "_id": "$borough",
                "avgScore": { "$avg": "$grades.score" }
            }
        }
    ])
    print(type(qresult))

    res = list(qresult)
    df2 = pd.DataFrame.from_records(json_normalize(res))
    print(df2)
```

```
<class 'pymongo.command_cursor.CommandCursor'>
             _id   avgScore
0        Missing   9.632911
1  Staten Island  11.370958
2          Bronx  11.036186
3       Brooklyn  11.447976
4      Manhattan  11.418151
5         Queens  11.634865
```

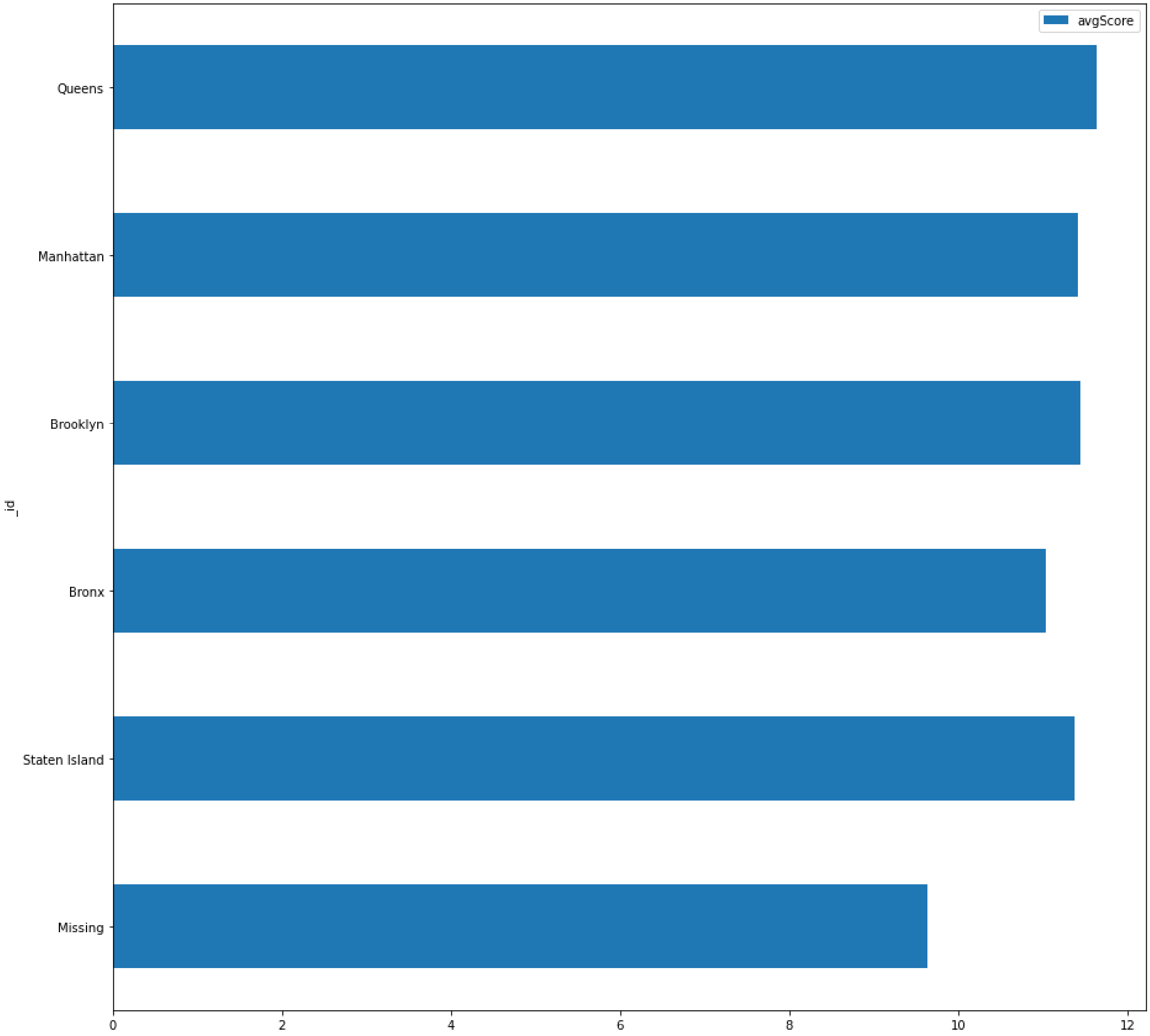
In [82]:

```
df2.plot(kind='barh', y="avgScore", x='_id', figsize=(15,15))
```

Out[82]:

```
<AxesSubplot:ylabel='_id'>
```

Loading [MathJax]/extensions/Safe.js



7. Decisions:

Based on the data provided, it can be concluded that there is a significant market potential for Pakistani cuisine in the restaurant industry. The data highlights the distribution of Pakistani restaurants across different boroughs and the average food quality scores.

Distribution of Pakistani Restaurants:

Bronx: 2 Manhattan: 8 Brooklyn: 10 Queens: 11 Average Food Quality Scores by Borough:

Missing: 9.632911 Staten Island: 11.370958 Bronx: 11.036186 Brooklyn: 11.447976 Manhattan: 11.418151 Queens: 11.634865 Average Food Quality Score for Pakistani Cuisine: 13.699115

Considering the above results, it is evident that there is a demand for Pakistani cuisine, particularly in Brooklyn and Queens, which have the highest number of Pakistani restaurants. Additionally, the average food quality score for Pakistani cuisine is higher than the scores for individual boroughs, indicating that Pakistani cuisine is well-regarded by customers.

To harness the promising opportunities in the Pakistani cuisine market, restaurant entrepreneurs should:

Focus on areas with a higher demand for Pakistani cuisine, such as Brooklyn and Queens. Strive for high-quality food offerings, as the higher average food quality score for Pakistani cuisine indicates the potential for customer satisfaction. Utilize data-driven insights to identify and target niche markets, ensuring that their restaurant offerings cater to specific customer preferences. Leverage data analysis for selecting the best location, taking into account factors like population density, accessibility, and local competition. Develop innovative concepts, incorporating unique twists to traditional Pakistani dishes and offering a memorable dining experience. By leveraging data-driven insights, entrepreneurs can make informed decisions that will improve their chances of success and profitability in the Pakistani cuisine restaurant business.

8. Conclusions:

In conclusion, the Pakistani cuisine market presents a promising opportunity for restaurant entrepreneurs who utilize data-driven insights to make informed decisions. The data indicates a strong demand for Pakistani cuisine, particularly in Brooklyn and Queens, and a high average food quality score. By focusing on areas with high demand, maintaining high-quality food offerings, targeting niche markets, selecting the best location based on data analysis, and developing innovative concepts, entrepreneurs can optimize their restaurant venture. This approach will align with customer preferences and demands, increasing the likelihood of success and profitability in the Pakistani cuisine restaurant business.