Azure Databricks Coding Challenge

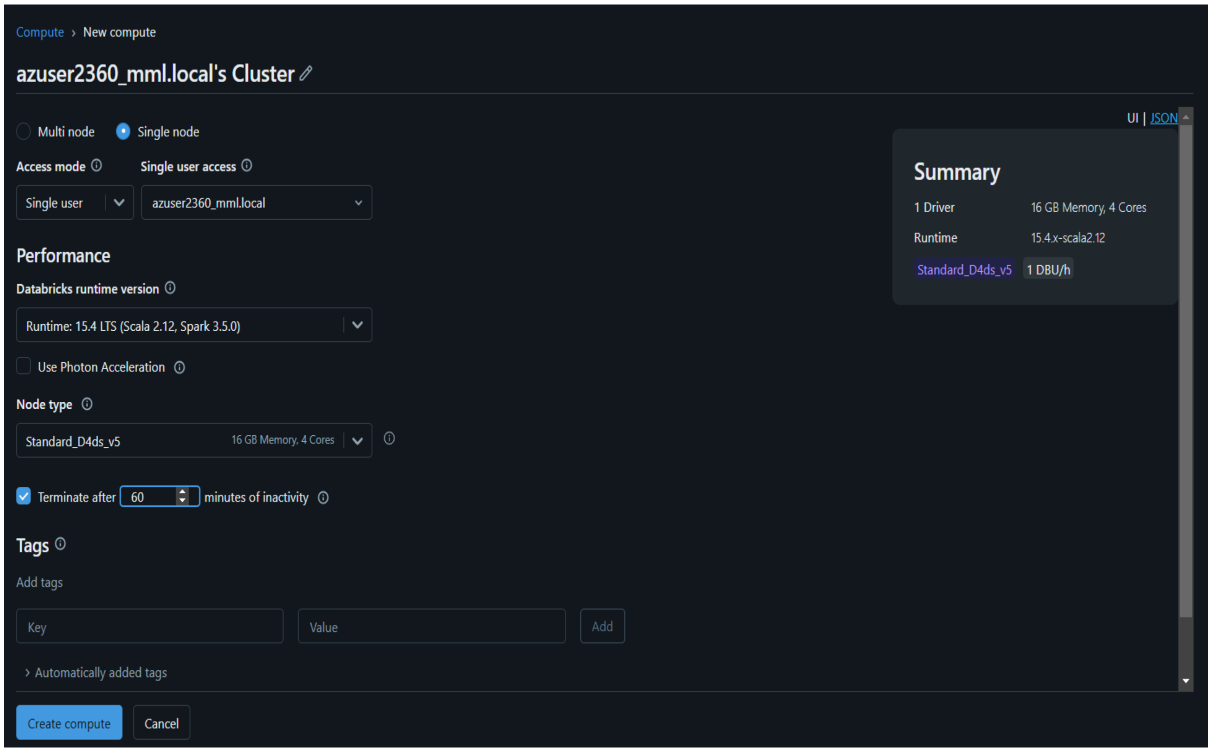
Submitted By: Aniroop Gupta
DE Batch 1

1. Creation of the Cluster:

Created a Databricks cluster with appropriate configurations to execute the notebook efficiently. Attached the notebook to the cluster for seamless code execution.

To create a cluster:

• Log in to Azure Databricks.
• Go to the Compute section and click on Create Cluster.
  Cluster Name: azuser2360_mml.local's Cluster
  Databricks Runtime Version: 15.4 LTS(Scala 2.12, Spark 3.5.0)
  Node Type: Single Node, Standard_D4ads_v5
• Click Create Cluster and wait until the status is "Running."

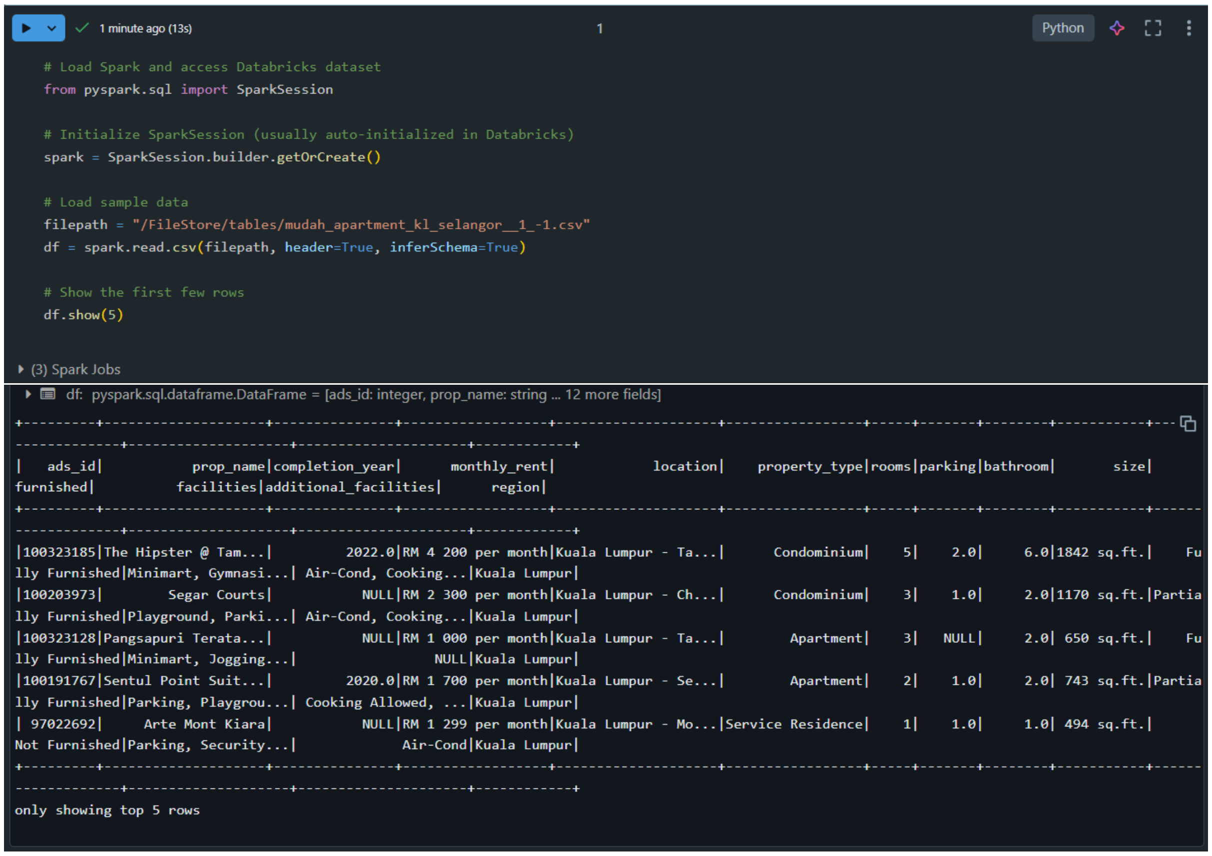


Access mode: Single user
Single user access: azuser2360_mml.local
Databricks runtime version: Runtime: 15.4 LTS (Scala 2.12, Spark 3.5.0)
Use Photon Acceleration
Node type: Standard_D4ds_v5 16 GB Memory, 4 Cores
Terminate after 60 minutes of inactivity

Summary
1 Driver 16 GB Memory, 4 Cores
Runtime 15.4.x-scala2.12
Standard_D4ds_v5 1 DBU/h

2. Create a DataFrame from a Databricks Dataset:

Loaded the dataset mudah-apartment-kl-selangor (1).csv into a Spark DataFrame using the spark.read.csv function. Displayed the first 5 rows of the DataFrame to ensure successful loading.

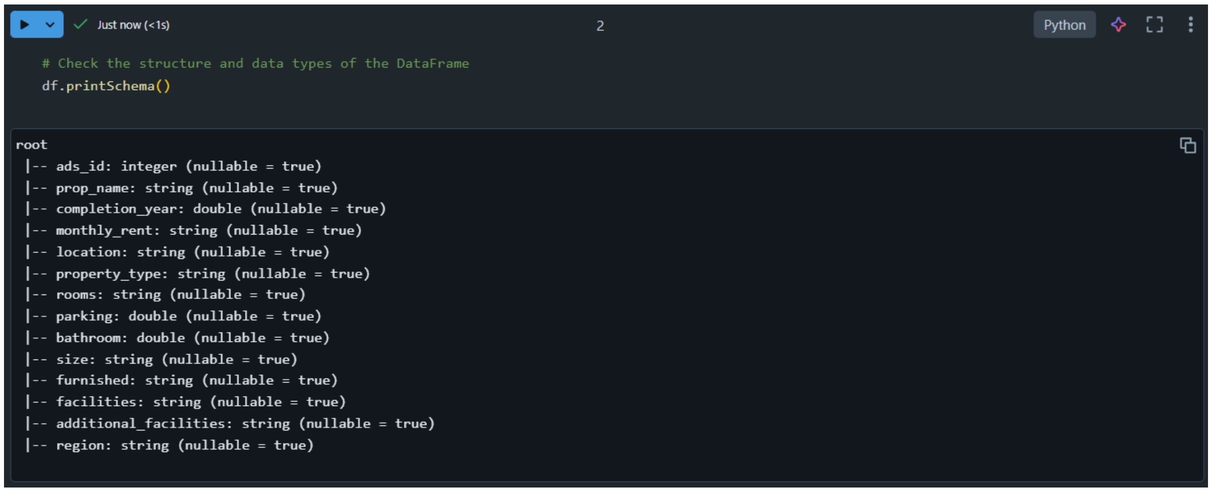


```python
# Load Spark and access Databricks dataset
from pyspark.sql import SparkSession

# Initialize SparkSession (usually auto-initialized in Databricks)
spark = SparkSession.builder.getOrCreate()

# Load sample data
filepath = "/FileStore/tables/mudah_apartment_kl_selangor__1_-1.csv"
df = spark.read.csv(filepath, header=True, inferSchema=True)

# Show the first few rows
df.show(5)
```

```
df: pyspark.sql.dataframe.DataFrame = [ads_id: integer, prop_name: string ... 12 more fields]
+---------+--------------------+---------------+------------------+--------------------+----------------+-----+-------+--------+-----------+---
------------+--------------------+---------------------+------------+
|   ads_id|           prop_name|completion_year|      monthly_rent|            location|   property_type|rooms|parking|bathroom|       size|
furnished|          facilities|additional_facilities|      region|
+---------+--------------------+---------------+------------------+--------------------+----------------+-----+-------+--------+-----------+------
------------+--------------------+---------------------+------------+
|100323185|The Hipster @ Tam...|         2022.0|RM 4 200 per month|Kuala Lumpur - Ta...|     Condominium|    5|    2.0|     6.0|1842 sq.ft.|     Fu
lly Furnished|Minimart, Gymnasi...| Air-Cond, Cooking...|Kuala Lumpur|
|100203973|         Segar Courts|           NULL|RM 2 300 per month|Kuala Lumpur - Ch...|     Condominium|    3|    1.0|     2.0|1170 sq.ft.|Partia
lly Furnished|Playground, Parki...| Air-Cond, Cooking...|Kuala Lumpur|
|100323128|Pangsapuri Terata...|           NULL|RM 1 000 per month|Kuala Lumpur - Ta...|       Apartment|    3|   NULL|     2.0| 650 sq.ft.|     Fu
lly Furnished|Minimart, Jogging...|                 NULL|Kuala Lumpur|
|100191767|Sentul Point Suit...|         2020.0|RM 1 700 per month|Kuala Lumpur - Se...|       Apartment|    2|    1.0|     2.0| 743 sq.ft.|Partia
lly Furnished|Parking, Playgrou...| Cooking Allowed, ...|Kuala Lumpur|
| 97022692|      Arte Mont Kiara|           NULL|RM 1 299 per month|Kuala Lumpur - Mo...|Service Residence|    1|    1.0|     1.0| 494 sq.ft.|
Not Furnished|Parking, Security...|             Air-Cond|Kuala Lumpur|
+---------+--------------------+---------------+------------------+--------------------+----------------+-----+-------+--------+-----------+------
------------+--------------------+---------------------+------------+
only showing top 5 rows
```

3. View Schema of the Data:

Displayed the schema of the dataset using the printSchema() function to understand the column names and data types.

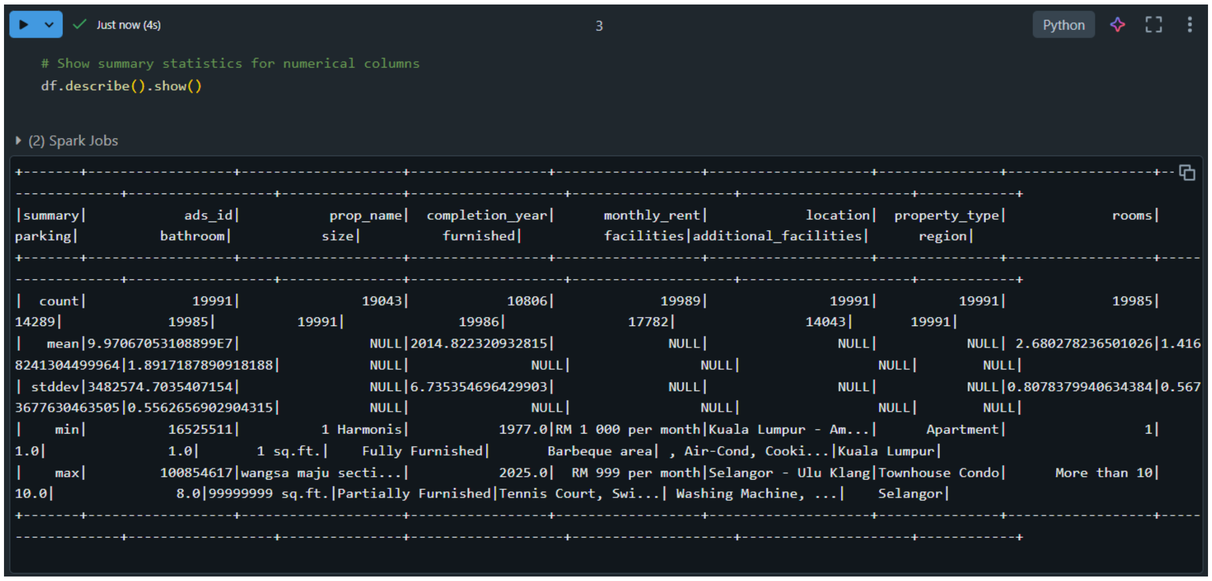


```python
# Check the structure and data types of the DataFrame
df.printSchema()
```

```
root
 |-- ads_id: integer (nullable = true)
 |-- prop_name: string (nullable = true)
 |-- completion_year: double (nullable = true)
 |-- monthly_rent: string (nullable = true)
 |-- location: string (nullable = true)
 |-- property_type: string (nullable = true)
 |-- rooms: string (nullable = true)
 |-- parking: double (nullable = true)
 |-- bathroom: double (nullable = true)
 |-- size: string (nullable = true)
 |-- furnished: string (nullable = true)
 |-- facilities: string (nullable = true)
 |-- additional_facilities: string (nullable = true)
 |-- region: string (nullable = true)
```

4. Display Summary Statistics:

Used the describe() function to generate summary statistics for numeric columns, providing insights into the data distribution.



Just now (4s) 3 Python

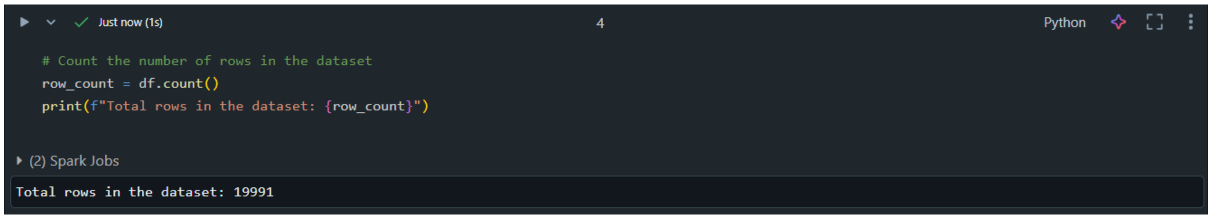
```
# Show summary statistics for numerical columns
df.describe().show()
```

▸ (2) Spark Jobs

```
+-------+-----------------+-------------------+-----------------+------------------+-------------------+--------------+------------------+--
------------+-----------------+-------------+------------------+-------------------+--------------------+-----------+
|summary|           ads_id|          prop_name|  completion_year|      monthly_rent|           location| property_type|             rooms|
parking|         bathroom|         size|         furnished|        facilities|additional_facilities|      region|
+-------+-----------------+-------------------+-----------------+------------------+-------------------+--------------+------------------+-----
------------+-----------------+-------------+------------------+-------------------+--------------------+-----------+
|  count|            19991|              19043|            10806|             19989|              19991|         19991|             19985|
14289|            19985|        19991|             19986|             17782|               14043|      19991|
|   mean|9.97067053108899E7|               NULL|2014.822320932815|              NULL|               NULL|          NULL| 2.680278236501026|1.416
8241304499964|1.8917187890918188|          NULL|              NULL|               NULL|                NULL|       NULL|
| stddev|3482574.7035407154|               NULL|6.735354696429903|              NULL|               NULL|          NULL|0.8078379940634384|0.567
3677630463505|0.5562656902904315|          NULL|              NULL|               NULL|                NULL|       NULL|
|    min|         16525511|         1 Harmonis|           1977.0|RM 1 000 per month|Kuala Lumpur - Am...|     Apartment|                 1|
1.0|              1.0|      1 sq.ft.|   Fully Furnished|      Barbeque area| , Air-Cond, Cooki...|Kuala Lumpur|
|    max|        100854617|wangsa maju secti...|           2025.0|  RM 999 per month|Selangor - Ulu Klang|Townhouse Condo|      More than 10|
10.0|              8.0|99999999 sq.ft.|Partially Furnished|Tennis Court, Swi...| Washing Machine, ...|   Selangor|
+-------+-----------------+-------------------+-----------------+------------------+-------------------+--------------+------------------+-----
------------+-----------------+-------------+------------------+-------------------+--------------------+-----------+
```

5. Count Rows in Dataset:

Counted the total number of rows in the dataset using the count() function to understand the dataset's size.



Just now (1s) 4 Python

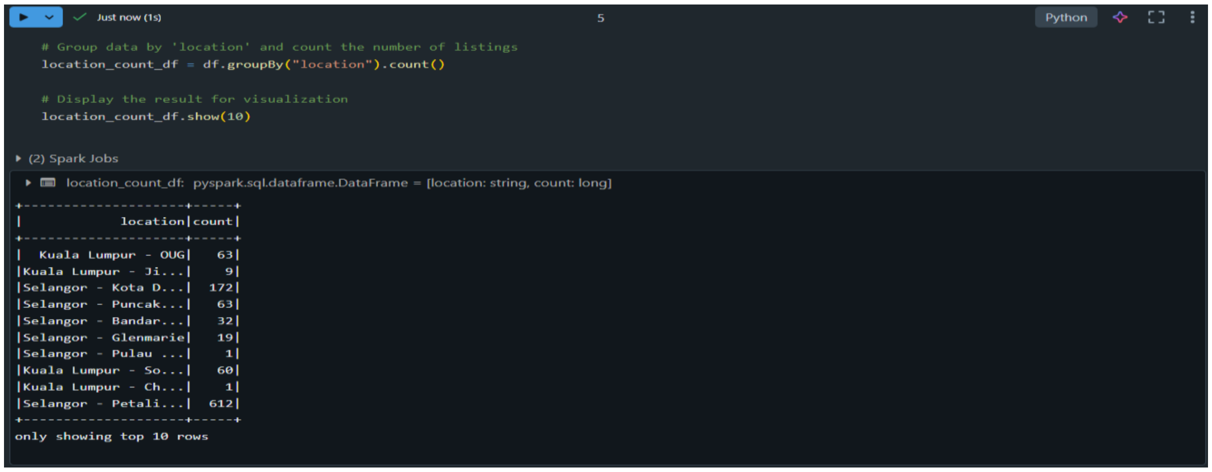
```
# Count the number of rows in the dataset
row_count = df.count()
print(f"Total rows in the dataset: {row_count}")
```

▸ (2) Spark Jobs

```
Total rows in the dataset: 19991
```

6. Group and Aggregate Data Sort Data:

Grouped the dataset by the location column and calculated the average monthly_rent to analyze rent trends.



Just now (1s) 5 Python

```
# Group data by 'location' and count the number of listings
location_count_df = df.groupBy("location").count()

# Display the result for visualization
location_count_df.show(10)
```

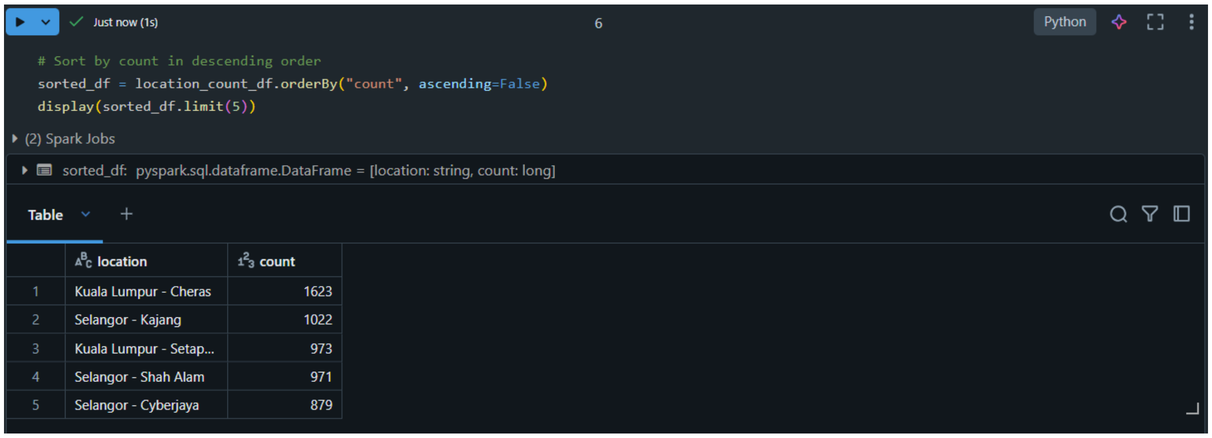
▸ (2) Spark Jobs

▸ location_count_df: pyspark.sql.dataframe.DataFrame = [location: string, count: long]

```
+--------------------+-----+
|            location|count|
+--------------------+-----+
|  Kuala Lumpur - OUG|   63|
|Kuala Lumpur - Ji...|    9|
|Selangor - Kota D...|  172|
|Selangor - Puncak...|   63|
|Selangor - Bandar...|   32|
|Selangor - Glenmarie|   19|
|Selangor - Pulau ...|    1|
|Kuala Lumpur - So...|   60|
|Kuala Lumpur - Ch...|    1|
|Selangor - Petali...|  612|
+--------------------+-----+
only showing top 10 rows
```

7. Sort Data:

Sorted the aggregated data in descending order of location count using the orderBy() function to identify the count of locations.



```python
# Sort by count in descending order
sorted_df = location_count_df.orderBy("count", ascending=False)
display(sorted_df.limit(5))
```

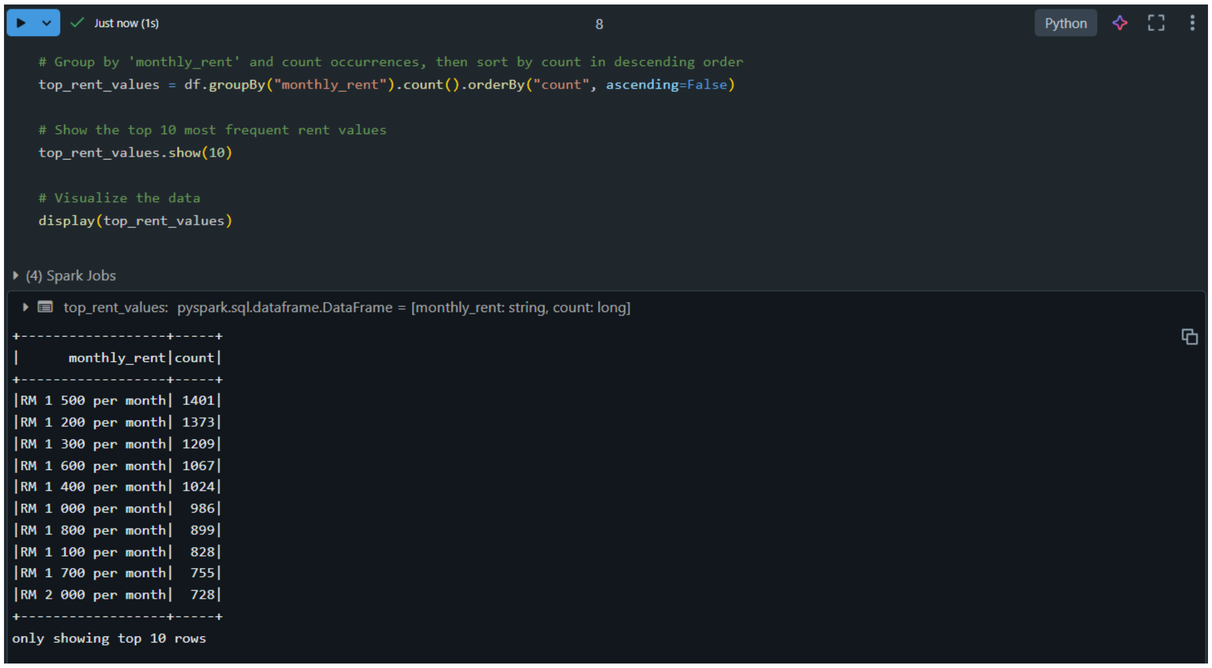
(2) Spark Jobs

sorted_df: pyspark.sql.dataframe.DataFrame = [location: string, count: long]

	location	count
1	Kuala Lumpur - Cheras	1623
2	Selangor - Kajang	1022
3	Kuala Lumpur - Setap...	973
4	Selangor - Shah Alam	971
5	Selangor - Cyberjaya	879

8. Top 10 Most Frequent Monthly Rent Values:

Grouped by monthly_rent and counted occurrences, then displayed the top 10 most frequent rent values in descending order.



```python
# Group by 'monthly_rent' and count occurrences, then sort by count in descending order
top_rent_values = df.groupBy("monthly_rent").count().orderBy("count", ascending=False)

# Show the top 10 most frequent rent values
top_rent_values.show(10)

# Visualize the data
display(top_rent_values)
```

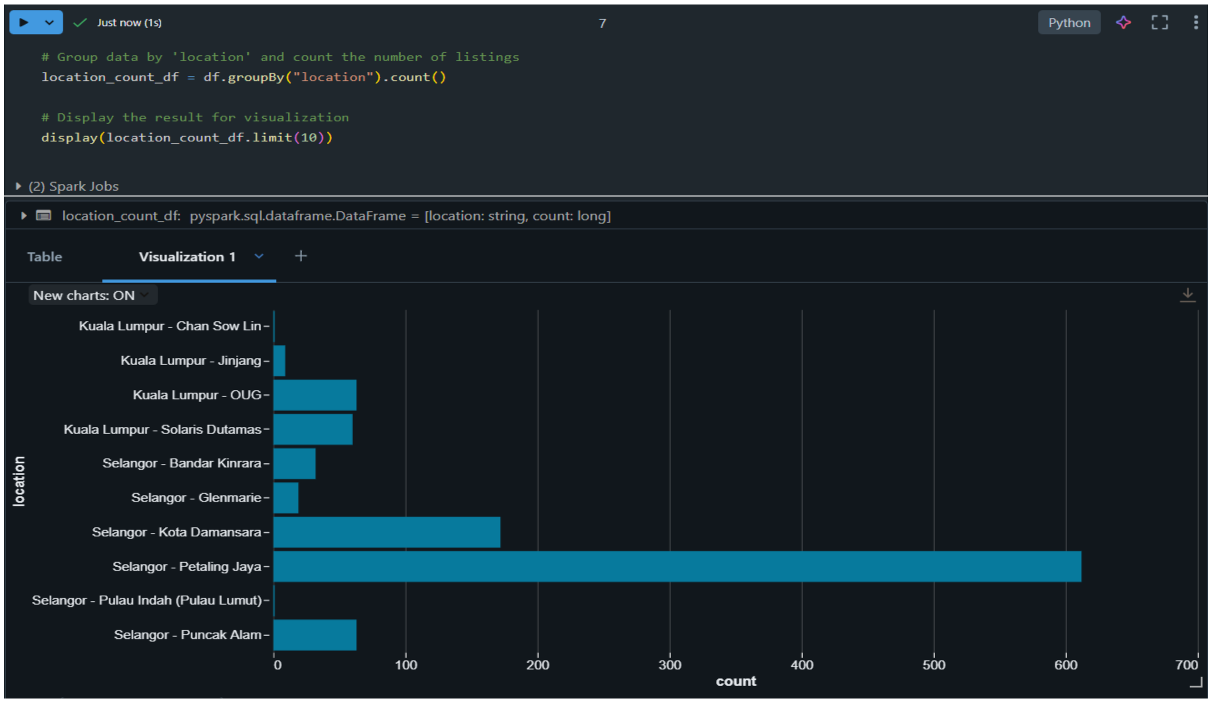
(4) Spark Jobs

top_rent_values: pyspark.sql.dataframe.DataFrame = [monthly_rent: string, count: long]

```
+------------------+-----+
|      monthly_rent|count|
+------------------+-----+
|RM 1 500 per month| 1401|
|RM 1 200 per month| 1373|
|RM 1 300 per month| 1209|
|RM 1 600 per month| 1067|
|RM 1 400 per month| 1024|
|RM 1 000 per month|  986|
|RM 1 800 per month|  899|
|RM 1 100 per month|  828|
|RM 1 700 per month|  755|
|RM 2 000 per month|  728|
+------------------+-----+
only showing top 10 rows
```

9. Create a Visualizations:

Visualized the count of listings by location using a bar chart to analyze the distribution of data across different areas.



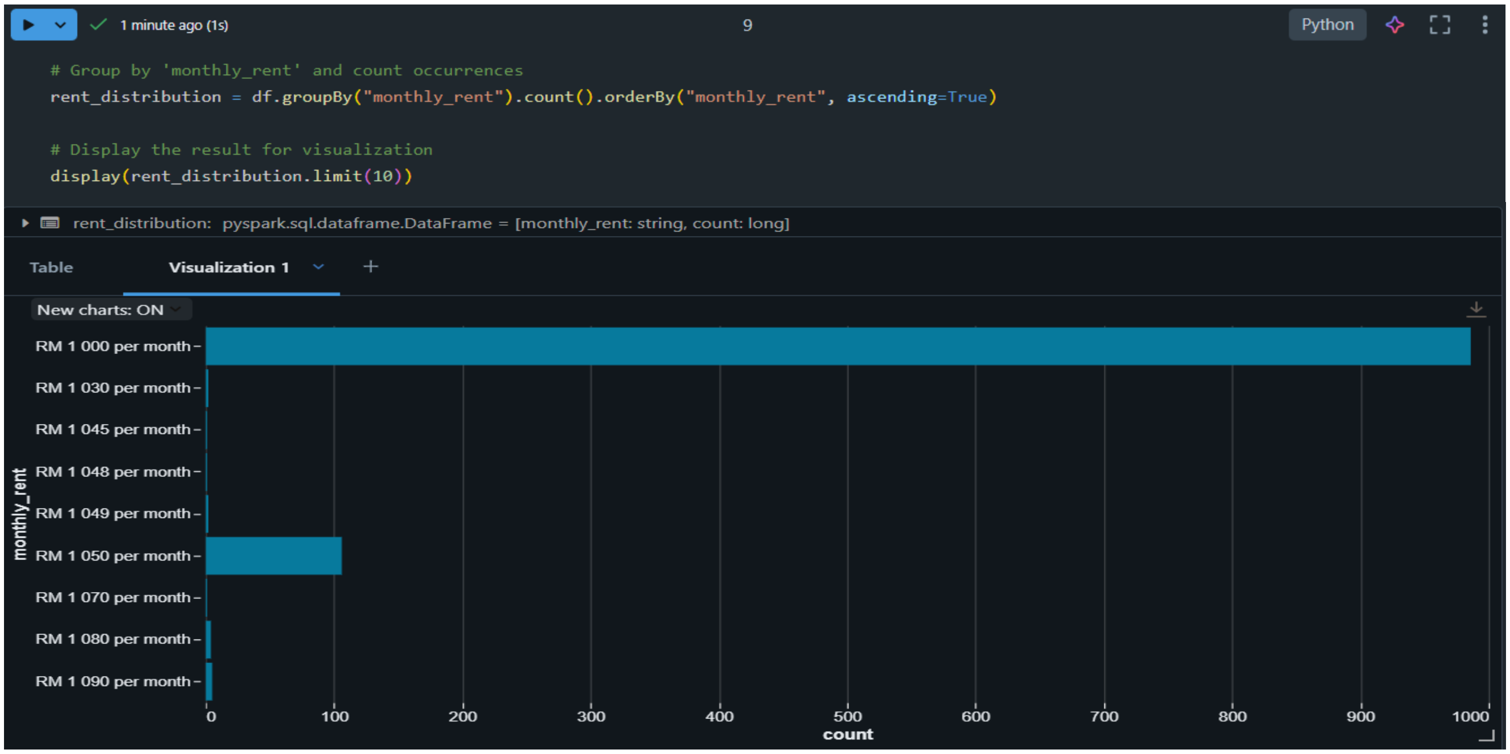
```python
# Group data by 'location' and count the number of listings
location_count_df = df.groupBy("location").count()

# Display the result for visualization
display(location_count_df.limit(10))
```

(2) Spark Jobs

location_count_df: pyspark.sql.dataframe.DataFrame = [location: string, count: long]

10. Another Visualization:

Created a bar chart to display the distribution of monthly rent values, showing the frequency of each rent amount.



```python
# Group by 'monthly_rent' and count occurrences
rent_distribution = df.groupBy("monthly_rent").count().orderBy("monthly_rent", ascending=True)

# Display the result for visualization
display(rent_distribution.limit(10))
```

rent_distribution: pyspark.sql.dataframe.DataFrame = [monthly_rent: string, count: long]