This submission template is a convenient document for you to provide the screenshots and explanations for Assignment 5.0. This submission template is intended to be used in conjunction with the Assignment 5.0 Instructions document. The instructions document illustrates how to correctly execute each SQL construct, explains important theoretical and practical details, and contains the complete set of instructions on how to complete this lab.

**Name**:

**Date:**

**Section Two**

The screenshots needs to show your user name and the date loaded.

*15. Screenshot of the loaded file.*

Graphical user interface, text, application, email

Description automatically generated

*16. Screenshot of the loaded data frame.*

Graphical user interface, text, application, email

Description automatically generated

*17. Provide the query command and the resulting data set*

Wifi\_df.printSchema()

Graphical user interface, text, application, email

Description automatically generated

*18. Briefly describe the structure of the data frame.*

The wifi\_df dataframe has three columns: “geometry”, “properties” and “type”

1. The “geometry” column is a struct that has two sub-fields which are “coordinates” and “type” of a geographical location. “coordinates”, an array and “type”, a string.
2. The “properties” columns is also a struct, it stores various attributes such as device\_address, device\_connectedto, device\_lat,device\_long, device\_serial, device\_tags, etl\_updateddtimestamp, insiade\_outside,is\_current, landmark, neighborhood\_id, neighborhood\_name,org1, org2
3. The “type” column is a string that specifies the type of the data frame.

*21. Provide the query command and the resulting data set*

Wifi\_df.select('geometry.coordinates','properties.ObjectId', 'properties.device\_address','properties.device\_connectedto','properties.device\_lat', 'properties.device\_long','properties.device\_serial', 'properties.etl\_updatedtimestamp', 'properties.neighborhood\_id', 'properties.neighborhood\_name').show()

Table

Description automatically generated with low confidence

*22. Provide the query command and the resulting data set*

Wifi\_df.select('geometry.coordinates','properties.ObjectId', 'properties.device\_address','properties.device\_connectedto','properties.device\_lat', 'properties.device\_long','properties.device\_serial', 'properties.etl\_updatedtimestamp', 'properties.neighborhood\_id', 'properties.neighborhood\_name').write.mode("overwrite").saveAsTable("Wifi\_tbl")

Graphical user interface, text, application

Description automatically generated

%sql

select \* from Wifi\_tbl

Graphical user interface, text, application

Description automatically generated

*23. Provide the query command and the resulting data set*

%sql

select \* from Wifi\_tbl limit 10

**Graphical user interface, text, application, table, email

Description automatically generated**

*24. Provide the query command and the resulting data set including chart*

**%sql**

**SELECT**

**neighborhood\_name as neighborhood,**

**count(\*) AS num\_devices,**

**avg(device\_lat) OVER (PARTITION BY neighborhood\_id) AS avg\_lat,**

**avg(device\_long) OVER (PARTITION BY neighborhood\_id) AS avg\_long,**

**date(etl\_updatedtimestamp) as date**

**FROM Wifi\_tbl**

**GROUP BY neighborhood\_id,device\_lat,device\_long, neighborhood\_name, date(etl\_updatedtimestamp)**

Table

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

*25. Very briefly explain what you have discovered based on your data set from the query above.*

The query above provides a summary of the total number of devices in each neighborhood and the average location of the devices in that neighborhood at different points in time. These results will help to identify neighborhoods with higher or lower densities of devices. The query above also extracts the date from the etl\_updatedtimestamp columns. Which will help to analysis the data by date.

*26. Provide the query command and the resulting data set including chart*

from pyspark.sql.functions import col, count, date\_format

spark.sql("""select device\_address, count(ObjectId) as number\_devices,

neighborhood\_name as neighborhood,

date(etl\_updatedtimestamp) as date

from Wifi\_tbl

group by device\_address,neighborhood\_name,date(etl\_updatedtimestamp)

order by count(ObjectId) desc""").display()

Graphical user interface, application

Description automatically generated

A picture containing graphical user interface

Description automatically generated

*27: Very briefly explain what you have discovered based on your data set from the query above.*

The result of query above helps us to know which device\_address has the most number of devices. And it also provides additional information like the neighborhood and date the devices was last updated.

*Extra Credit (2 points): Note how one of the columns in the original data frame is an array of coordinates. Look to use the explode function to extract those coordinates into a separate flattened data frame.*

from pyspark.sql.functions import explode

Wifi\_df.select(explode("geometry.coordinates").alias("coordinates"))

Graphical user interface, text, application

Description automatically generated

Use the **Ask your Facilitator Discussion Board** if you have any questions regarding the how to approach this assignment.

Save your assignment as ***lastnameFirstname\_lassignment5.doc*** and submit it in the *Assignments* section of the course.

For help uploading files please refer to the *Technical Support* page in the syllabus.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Criterion | A | B | C | D | F | Letter Grade |
| Correctness and Completeness of Results (70%) | All steps' results are entirely complete and correct | About ¾ of the steps' results are correct and complete | About half of the steps' results are correct and complete | About ¼ of the steps' results are correct and complete | Virtually none of the step's results are correct and complete |  |
| Constitution of SQL/Python and Explanations (30%) | Excellent use and integration of appropriate SQL/Python constructs and supporting explanations | Good use and integration of appropriate SQL/Python constructs and supporting explanations | Mediocre use and integration of appropriate SQL/Python constructs and supporting explanations | Substandard use and integration of appropriate SQL/Python constructs and supporting explanations | Virtually all SQL/Python constructs and supporting explanations are unsuitable or improperly integrated |  |
|  |  |  |  |  | Assignment Grade: |  |