Name:-Preksha Patel

Sapid:-60004210126

Branch:- Computer Engineering

Div:-C2; Batch:-1

BDI-EXPERIMENT-08

BDI

Name = Preksha A. Patel
Sapid = 60004210126
Branch = Computer Engineering
Div = C2, Batch = 1

Experiment no. 8

Aim = To implement Spark SQL using PySpark

Theory =
Spark SQL is a component of Apache spark designed for querying structured and semi-structured data using SQL like syntax. It extends spark capabilities & includes SQL queries making data querying more accessible to users familiar to SQL. Spark SQL seamlessly integrates with other spark components enabling efficient data processing and analysis.
queries: SQL support Spark SQL is used to execute SQL queries against structured data.
- Data Frame API: Spark SQL introduces DataFrames, which provides user friendly & optimized analysis.
- Unified data Access = It supports multiple data sources such as request, JSON, Head relation

Advantages =
1) Familarity = Spark SQL leverages SQL familarity enabling users to easily transition existing SQL skills to spark for data processing task.
2) Performance = By initialising spark in memory processing & optimisation techniques, spark SQL requires high performance for analytical & data processing workloads.

Disadvantages :-

1) Limited SQL support while Spark SQL supports subset SQL syntax, it may lack some advanced SQL features.
2) Resource intensive :- in memory processing & optimization techniques can be resource intensive.

Conclusion :-

Thus we have successfully implemented and understood Apache Spark

Code & Output:

```
[ ]  from pyspark.sql import SparkSession

     spark = SparkSession.builder.appName("Online IRIS dataset").getOrCreate()

     url = "/content/sample_data/Iris.csv"

     df = spark.read.csv(url, header = False, inferSchema =  True)

     columns = ["id", "sepal_length", "sepal_width", "petal_length", "petal_width", "class"]
     df = df.toDF(*columns)

     df.createOrReplaceTempView("iris_data")

     result1 = spark.sql("Select * from iris_data  WHERE class = 'Iris-setosa'")
     result2 = spark.sql("Select * from iris_data  WHERE sepal_length > 7.0")
     result3 = spark.sql("Select class, COUNT(*) from iris_data group by class ")

     result1.show()
     result2.show()
     result3.show()

     spark.stop()
```

```
+---+------------+-----------+------------+-----------+-----------+
| id|sepal_length|sepal_width|petal_length|petal_width|      class|
+---+------------+-----------+------------+-----------+-----------+
|  1|         5.1|        3.5|         1.4|        0.2|Iris-setosa|
|  2|         4.9|        3.0|         1.4|        0.2|Iris-setosa|
|  3|         4.7|        3.2|         1.3|        0.2|Iris-setosa|
|  4|         4.6|        3.1|         1.5|        0.2|Iris-setosa|
|  5|         5.0|        3.6|         1.4|        0.2|Iris-setosa|
|  6|         5.4|        3.9|         1.7|        0.4|Iris-setosa|
|  7|         4.6|        3.4|         1.4|        0.3|Iris-setosa|
|  8|         5.0|        3.4|         1.5|        0.2|Iris-setosa|
|  9|         4.4|        2.9|         1.4|        0.2|Iris-setosa|
| 10|         4.9|        3.1|         1.5|        0.1|Iris-setosa|
| 11|         5.4|        3.7|         1.5|        0.2|Iris-setosa|
| 12|         4.8|        3.4|         1.6|        0.2|Iris-setosa|
| 13|         4.8|        3.0|         1.4|        0.1|Iris-setosa|
| 14|         4.3|        3.0|         1.1|        0.1|Iris-setosa|
| 15|         5.8|        4.0|         1.2|        0.2|Iris-setosa|
| 16|         5.7|        4.4|         1.5|        0.4|Iris-setosa|
| 17|         5.4|        3.9|         1.3|        0.4|Iris-setosa|
| 18|         5.1|        3.5|         1.4|        0.3|Iris-setosa|
| 19|         5.7|        3.8|         1.7|        0.3|Iris-setosa|
| 20|         5.1|        3.8|         1.5|        0.3|Iris-setosa|
+---+------------+-----------+------------+-----------+-----------+
only showing top 20 rows
```

IRIS DATASET:

id	sepal_length	sepal_width	petal_length	petal_width	class
103	7.1	3.0	5.9	2.1	Iris-virginica
106	7.6	3.0	6.6	2.1	Iris-virginica
108	7.3	2.9	6.3	1.8	Iris-virginica
110	7.2	3.6	6.1	2.5	Iris-virginica
118	7.7	3.8	6.7	2.2	Iris-virginica
119	7.7	2.6	6.9	2.3	Iris-virginica
123	7.7	2.8	6.7	2.0	Iris-virginica
126	7.2	3.2	6.0	1.8	Iris-virginica
130	7.2	3.0	5.8	1.6	Iris-virginica
131	7.4	2.8	6.1	1.9	Iris-virginica
132	7.9	3.8	6.4	2.0	Iris-virginica
136	7.7	3.0	6.1	2.3	Iris-virginica

class	count(1)
Species	1
Iris-virginica	50
Iris-setosa	50
Iris-versicolor	50

Titanic Dataset - Perform SQL Queries to find:

a. What is the number of passengers who survived the Titanic Disaster?

b. How many female passengers were on board the Titanic?

c. What is the average age of passengers in each passenger class?

```python
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Online IRIS dataset").getOrCreate()

url = "/content/sample_data/Titanic-Dataset.csv"

df = spark.read.csv(url, header = False, inferSchema =  True)

columns = ["PassengerId","Survived","Pclass","Name","Sex","Age","SibSp","Parch","Ticket","Fare","Cabin","Embarked"]
df = df.toDF(*columns)

df.createOrReplaceTempView("titanic")

result1 = spark.sql("Select * from titanic  WHERE survived = 1")
result2 = spark.sql("Select  count(*) from titanic where sex = 'female'")
result3 = spark.sql("Select avg(Age) from titanic group by Pclass ")

result1.show()
result2.show()
result3.show()

spark.stop()
```

```
+-----------+--------+------+--------------------+------+----+-----+-----+----------------+--------+-----+--------+
|PassengerId|Survived|Pclass|                Name|   Sex| Age|SibSp|Parch|          Ticket|    Fare|Cabin|Embarked|
+-----------+--------+------+--------------------+------+----+-----+-----+----------------+--------+-----+--------+
|          2|       1|     1|Cumings, Mrs. Joh...|female|  38|    1|    0|        PC 17599| 71.2833|  C85|       C|
|          3|       1|     3|Heikkinen, Miss. ...|female|  26|    0|    0|STON/O2. 3101282|   7.925| NULL|       S|
|          4|       1|     1|Futrelle, Mrs. Ja...|female|  35|    1|    0|          113803|    53.1| C123|       S|
|          9|       1|     3|Johnson, Mrs. Osc...|female|  27|    0|    2|          347742| 11.1333| NULL|       S|
|         10|       1|     2|Nasser, Mrs. Nich...|female|  14|    1|    0|          237736| 30.0708| NULL|       C|
|         11|       1|     3|Sandstrom, Miss. ...|female|   4|    1|    1|         PP 9549|    16.7|   G6|       S|
|         12|       1|     1|Bonnell, Miss. El...|female|  58|    0|    0|          113783|   26.55| C103|       S|
|         16|       1|     2|Hewlett, Mrs. (Ma...|female|  55|    0|    0|          248706|      16| NULL|       S|
|         18|       1|     2|Williams, Mr. Cha...|  male|NULL|    0|    0|          244373|      13| NULL|       S|
|         20|       1|     3|Masselmani, Mrs. ...|female|NULL|    0|    0|            2649|   7.225| NULL|       C|
|         22|       1|     2|Beesley, Mr. Lawr...|  male|  34|    0|    0|          248698|      13|  D56|       S|
|         23|       1|     3|"McGowan, Miss. A...|female|  15|    0|    0|          330923|  8.0292| NULL|       Q|
|         24|       1|     1|Sloper, Mr. Willi...|  male|  28|    0|    0|          113788|    35.5|   A6|       S|
|         26|       1|     3|Asplund, Mrs. Car...|female|  38|    1|    5|          347077| 31.3875| NULL|       S|
|         29|       1|     3|"O'Dwyer, Miss. E...|female|NULL|    0|    0|          330959|  7.8792| NULL|       Q|
|         32|       1|     1|Spencer, Mrs. Wil...|female|NULL|    1|    0|        PC 17569|146.5208|  B78|       C|
|         33|       1|     3|Glynn, Miss. Mary...|female|NULL|    0|    0|          335677|    7.75| NULL|       Q|
|         37|       1|     3|    Mamee, Mr. Hanna|  male|NULL|    0|    0|            2677|  7.2292| NULL|       C|
|         40|       1|     3|Nicola-Yarred, Mi...|female|  14|    1|    0|            2651| 11.2417| NULL|       C|
|         44|       1|     2|Laroche, Miss. Si...|female|   3|    1|    2|   SC/Paris 2123| 41.5792| NULL|       C|
+-----------+--------+------+--------------------+------+----+-----+-----+----------------+--------+-----+--------+
only showing top 20 rows

+--------+
|count(1)|
+--------+
|     314|
+--------+

+------------------+
|          avg(Age)|
+------------------+
|              NULL|
| 25.14061971830986|
|38.233440860215055|
| 29.87763005780347|
+------------------+
```

Wine Quality Dataset Example

a. How many wines are considered high quality (quality score of 7 or higher)

b. What is the average alcohol content of the wines in the dataset

```
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Online IRIS dataset").getOrCreate()

url = "/content/sample_data/winequality-red.csv"

df = spark.read.csv(url, header = False, inferSchema =  True)

columns = ["fixed acidity","volatile acidity","citric acid","residual sugar","chlorides","free sulfur dioxide","total sulfur dioxide","density","pH","sulphates","alcohol","quality"]
df = df.toDF(*columns)

df.createOrReplaceTempView("wine")

result1 = spark.sql("Select count(*) from wine  WHERE quality >= 7.0")
result3 = spark.sql("Select avg(alcohol) from wine ")

result1.show()
result3.show()

spark.stop()
```

```
+--------+
|count(1)|
+--------+
|     217|
+--------+

+------------------+
|      avg(alcohol)|
+------------------+
|10.422983114446502|
+------------------+
```

California Housing Dataset Example:

a. How many houses have a median value above $5000,000 in California

b. What is the average age of the houses in the dataset

```python
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Online IRIS dataset").getOrCreate()

url = "/content/sample_data/california_housing_train.csv"

df = spark.read.csv(url, header = False, inferSchema =  True)

columns = ["longitude", "latitude", "housing_median_age", "total_rooms", "total_bedrooms", "population", "households", "median_income", "median_house_value"]
df = df.toDF(*columns)

df.createOrReplaceTempView("california")

result1 = spark.sql("Select count(*) from california WHERE median_house_value > 50000")
result3 = spark.sql("Select avg(housing_median_age) from california ")

result1.show()
result3.show()

spark.stop()
```

```
+--------+
|count(1)|
+--------+
|   16820|
+--------+

+----------------------+
|avg(housing_median_age)|
+----------------------+
|      28.58935294117647|
+----------------------+
```