

Assignment – Day 14

-Sarthak Niranjana Kulkarni (Maverick)

- sarthakkul2311@gmail.com

- (+91) 93256 02791

21/11/2024 (Thursday)

Spark-SQL Practice: -

1. Loading a CSV file into a Spark DataFrame with specified options and creating a temporary SQL view.

```
# File location and type
file_location = "/FileStore/tables/simple_zipcodes-1.csv"
file_type = "csv"
# CSV options
infer_schema = "false"
first_row_is_header = "false"
delimiter = ","
# The applied options are for CSV files. For other file types, these will
be ignored.
df = spark.read.format(file_type) \
    .option("inferSchema", infer_schema) \
    .option("header", first_row_is_header) \
    .option("sep", delimiter) \
    .load(file_location)
display(df)
df.createOrReplaceTempView("tempdata")
```

Table					
	RecordNum...	Country	City	Zipcode	State
1	1	US	PARC PARQUE	704	PR
2	2	US	PASEO COSTA DEL SUR	704	PR
3	10	US	BDA SAN LUIS	709	PR
4	49347	US	HOLT	32564	FL
5	49348	US	HOMOSASSA	34487	FL
6	61391	US	CINGULAR WIRELESS	76166	TX
7	61392	US	FORT WORTH	76177	TX
8	61393	US	FT WORTH	76177	TX
9	54356	US	SPRUCE PINE	35585	AL
10	76511	US	ASH HILL	27007	NC
11	4	US	URB EUGENE RICE	704	PR
12	39627	US	MESA	85209	AZ
13	39628	US	MESA	85210	AZ
14	49345	US	HILLIARD	32046	FL
15					

21 rows | 1.38 seconds runtime

2. Querying a temporary SQL view and selecting specific columns from a DataFrame in Spark.

```
spark.sql("select * from tempdata").show()  
df.select("_c0", "_c1").show(5)
```

► (2) Spark Jobs

_c0	_c1	_c2	_c3	_c4
RecordNumber	Country	City	Zipcode	State
1	US	PARC PARQUE	704	PR
2	US	PASEO COSTA DEL SUR	704	PR
10	US	BDA SAN LUIS	709	PR
49347	US	HOLT	32564	FL
49348	US	HOMOSASSA	34487	FL
61391	US	CINGULAR WIRELESS	76166	TX
61392	US	FORT WORTH	76177	TX
61393	US	FT WORTH	76177	TX
54356	US	SPRUCE PINE	35585	AL
76511	US	ASH HILL	27007	NC
4	US	URB EUGENE RICE	704	PR
39827	US	MESA	85209	AZ
39828	US	MESA	85210	AZ
49345	US	HILLIARD	32046	FL
49346	US	HOLDER	34445	FL
3	US	SECT LANAUSSSE	704	PR
54354	US	SPRING GARDEN	36275	AL

76512	US	ASHEBORO	27203	NC
-------	----	----------	-------	----

only showing top 20 rows

_c0	_c1
RecordNumber	Country
1	US
2	US
10	US
49347	US

only showing top 5 rows

3. Filtering rows from a temporary SQL view in Spark based on a column value condition.

```
spark.sql("""SELECT * From tempdata WHERE _c4='AZ'""").show(5)
```

```
+-----+-----+-----+-----+
| _c0|_c1| _c2| _c3|_c4|
+-----+-----+-----+-----+
|39827| US|MESA|85209| AZ|
|39828| US|MESA|85210| AZ|
+-----+-----+-----+-----+
```

4. Loading a CSV file with headers into a Spark DataFrame and creating a temporary SQL view named "customer".

```
# File location and type
file_location = "/FileStore/tables/simple_zipcodes-1.csv"
file_type = "csv"

# CSV options
infer_schema = "false"
first_row_is_header = "true"
delimiter = ","

# The applied options are for CSV files. For other file types, these
will be ignored.
df = spark.read.format(file_type) \
    .option("inferSchema", infer_schema) \
    .option("header", first_row_is_header) \
    .option("sep", delimiter) \
    .load(file_location)

display(df)
df.createOrReplaceTempView("customer")
```

df: pyspark.sql.dataframe.DataFrame = [RecordNumber: string, Country: string ... 3 more fields]

Table

	RecordNumber	Country	City	Zipcode	State
1	1	US	PARC PARQUE	704	PR
2	2	US	PASEO COSTA DEL SUR	704	PR
3	10	US	BDA SAN LUIS	709	PR
4	49347	US	HOLT	32564	FL
5	49348	US	HOMOSASSA	34487	FL
6	61391	US	CINGULAR WIRELESS	76166	TX
7	61392	US	FORT WORTH	76177	TX
8	61393	US	FT WORTH	76177	TX
9	54356	US	SPRUCE PINE	35585	AL
10	76511	US	ASH HILL	27007	NC
11	4	US	URB EUGENE RICE	704	PR
12	39827	US	MESA	85209	AZ
13	39828	US	MESA	85210	AZ
14	49345	US	HILLIARD	32046	FL
15	49346	US	HOLDER	34445	FL

20 rows | 0.89 seconds runtime

5. Displaying all records from a SQL view and selecting specific columns from a DataFrame in Spark.

```
spark.sql("select * from customer").show()
df.select("RecordNumber", "Country").show(5)
```

RecordNumber	Country	City	Zipcode	State
1	US	PARC PARQUE	704	PR
2	US	PASEO COSTA DEL SUR	704	PR
10	US	BDA SAN LUIS	709	PR
49347	US	HOLT	32564	FL
49348	US	HOMOSASSA	34487	FL
61391	US	CINGULAR WIRELESS	76166	TX
61392	US	FORT WORTH	76177	TX
61393	US	FT WORTH	76177	TX
54356	US	SPRUCE PINE	35585	AL
76511	US	ASH HILL	27007	NC
4	US	URB EUGENE RICE	704	PR
39827	US	MESA	85209	AZ
39828	US	MESA	85210	AZ
49345	US	HILLIARD	32046	FL
49346	US	HOLDER	34445	FL
3	US	SECT LANAUSSSE	704	PR
54354	US	SPRING GARDEN	36275	AL
54355	US	SPRINGVILLE	35146	AL

```

+-----+-----+
|RecordNumber|Country|
+-----+-----+
|          1|    US|
|          2|    US|
|         10|    US|
|       49347|    US|
|       49348|    US|
+-----+-----+
only showing top 5 rows

```

6. Filtering rows from the "customer" SQL view in Spark where the state is 'PR'.

```
spark.sql("""SELECT * From customer WHERE state='PR'""").show(5)
```

```

+-----+-----+-----+-----+-----+
|RecordNumber|Country|          City|Zipcode|State|
+-----+-----+-----+-----+-----+
|          1|    US|    PARC PARQUE|    704|    PR|
|          2|    US|PASEO COSTA DEL SUR|    704|    PR|
|         10|    US|    BDA SAN LUIS|    709|    PR|
|          4|    US|    URB EUGENE RICE|    704|    PR|
|          3|    US|    SECT LANAUSSSE|    704|    PR|
+-----+-----+-----+-----+-----+

```

7. Filtering and ordering rows from the "customer" SQL view in Spark by specific states.

```
spark.sql("""select * FROM customer WHERE state in ('PR','AZ','FL')order by state """).show(10)
```

```

+-----+-----+-----+-----+-----+
|RecordNumber|Country|          City|Zipcode|State|
+-----+-----+-----+-----+-----+
|       39827|    US|          MESA|  85209|    AZ|
|       39828|    US|          MESA|  85210|    AZ|
|       49347|    US|          HOLT|  32564|    FL|
|       49348|    US|    HOMOSASSA|  34487|    FL|
|       49345|    US|    HILLIARD|  32046|    FL|
|       49346|    US|    HOLDER|  34445|    FL|
|          1|    US|    PARC PARQUE|    704|    PR|
|          2|    US|PASEO COSTA DEL SUR|    704|    PR|
|         10|    US|    BDA SAN LUIS|    709|    PR|
|          4|    US|    URB EUGENE RICE|    704|    PR|
+-----+-----+-----+-----+-----+
only showing top 10 rows

```

8. Grouping and counting the number of records for each state from the "customer" SQL view in Spark.

```
spark.sql("""SELECT state,count(*) as count FROM customer GROUP BY state""").show()
```

```
+-----+-----+
|state|count|
+-----+-----+
|  AZ |    2|
|  NC |    3|
|  AL |    3|
|  TX |    3|
|  FL |    4|
|  PR |    5|
+-----+-----+
```

Spark-SQL Summary: -

The provided code demonstrates the process of loading and processing CSV files using Apache Spark. Initially, a CSV file is read into a Spark DataFrame with no schema inference and no header, then displayed and queried through a temporary SQL view named tempdata. Queries include displaying all data, selecting specific columns (_c0, _c1), and filtering rows where column _c4 equals 'AZ'.

Later, the CSV file is reloaded with headers enabled into a DataFrame and assigned to a new SQL view named customer. Queries on this view include retrieving all data, selecting specific columns (RecordNumber, Country), filtering rows where the state column equals 'PR', and filtering states in a specific list (PR, AZ, FL) while ordering the results by state. Additionally, a query is used to group the data by the state column and count the number of records for each state. The code showcases Spark's SQL and DataFrame APIs for data analysis and transformations. The code also demonstrates how to read the same CSV file multiple times with different configurations, such as enabling header parsing in the second instance. Through these queries, Spark's ability to handle large datasets efficiently with SQL-like syntax for filtering, grouping, and ordering operations is highlighted, making it suitable for big data processing tasks.