databricks ASP 2.1L - Spark SQL Lab

Spark SQL Lab

Tasks

1. Create a DataFrame from the `events` table
2. Display the DataFrame and inspect its schema
3. Apply transformations to filter and sort `macOS` events
4. Count results and take the first 5 rows
5. Create the same DataFrame using a SQL query

Methods

- SparkSession (https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.sql.SparkSession.html?highlight=sparksession): `sql`, `table`
- DataFrame (https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.sql.DataFrame.html) transformations: `select`, `where`, `orderBy`

- DataFrame (https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.sql.DataFrame.html) actions: `select`, `count`, `take`
- Other DataFrame (https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.sql.DataFrame.html) methods: **`printSchema`**, **`schema`**, **`createOrReplaceTempView`**

```
%run ../../Includes/Classroom-Setup-SQL
```

```
Deleted the working directory dbfs:/user/odl_user_534131@databrickslabs.com/dbacademy/aspwd/asp_2_1l_spark_sql_lab


Your working directory is
dbfs:/user/odl_user_534131@databrickslabs.com/dbacademy/aspwd

The source for this dataset is
wasbs://courseware@dbacademy.blob.core.windows.net/apache-spark-programming-with-databricks/v02/

Skipping install of existing dataset to
dbfs:/user/odl_user_534131@databrickslabs.com/dbacademy/aspwd/datasets

Out[5]: DataFrame[key: string, value: string]
```

1. Create a DataFrame from the `events` table

- Use SparkSession to create a DataFrame from the **`events`** table

```
# TODO
events_df = spark.table("events")
```

2. Display DataFrame and inspect schema

- Use methods above to inspect DataFrame contents and schema

```
events_df.printSchema()
```

```
root
 |-- device: string (nullable = true)
 |-- ecommerce: struct (nullable = true)
 |    |-- purchase_revenue_in_usd: double (nullable = true)
 |    |-- total_item_quantity: long (nullable = true)
 |    |-- unique_items: long (nullable = true)
 |-- event_name: string (nullable = true)
 |-- event_previous_timestamp: long (nullable = true)
 |-- event_timestamp: long (nullable = true)
 |-- geo: struct (nullable = true)
 |    |-- city: string (nullable = true)
 |    |-- state: string (nullable = true)
 |-- items: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- coupon: string (nullable = true)
 |    |    |-- item_id: string (nullable = true)
 |    |    |-- item_name: string (nullable = true)
 |    |    |-- item_revenue_in_usd: double (nullable = true)
 |    |    |-- price_in_usd: double (nullable = true)
 |    |    |-- quantity: long (nullable = true)
 |-- traffic_source: string (nullable = true)
```

3. Apply transformations to filter and sort `macOS` events

- Filter for rows where `device` is `macOS`
- Sort rows by `event_timestamp`

```
# TODO
mac_df = (events_df.
          filter(events_df.device=="macOS")
          .sort(events_df.event_timestamp)
         )
```

4. Count results and take first 5 rows

- Use DataFrame actions to count and take rows

```
# TODO
num_rows = mac_df.count()
rows = mac_df.take(5)
```

4.1: CHECK YOUR WORK

```
from pyspark.sql import Row

assert(num_rows == 1938215)
assert(len(rows) == 5)
assert(type(rows[0]) == Row)
```

5. Create the same DataFrame using SQL query

- Use SparkSession to run a SQL query on the `events` table
- Use SQL commands to write the same filter and sort query used earlier

```
# TODO
mac_sql_df = spark.sql("select * from events where device='macOS' order by event_timestamp")

display(mac_sql_df)
```

	device	ecommerce	event_name	event_previ
1	macOS	▶ {"purchase_revenue_in_usd": null, "total_item_quantity": null, "unique_items": null}	mattresses	null
2	macOS	▶ {"purchase_revenue_in_usd": null, "total_item_quantity": null, "unique_items": null}	mattresses	1592322041
3	macOS	▶ {"purchase_revenue_in_usd": null, "total_item_quantity": null, "unique_items": null}	reviews	1592538997
4	macOS	▶ {"purchase_revenue_in_usd": null, "total_item_quantity": null, "unique_items": null}	guest	1592538390
5	macOS	▶ {"purchase_revenue_in_usd": null, "total_item_quantity": null, "unique_items": null}	main	null
6	macOS	▶ {"purchase_revenue_in_usd": null, "total_item_quantity": null, "unique_items": null}	original	1592539226
7	macOS	▶ {"purchase_revenue_in_usd": null, "total_item_quantity": null, "unique_items": null}	main	null

Truncated results, showing first 1000 rows.

5.1: CHECK YOUR WORK

- You should only see `macOS` values in the `device` column
- The fifth row should be an event with timestamp `1592539226602157`

```
verify_rows = mac_sql_df.take(5)
assert (mac_sql_df.select("device").distinct().count() == 1 and len(verify_rows) == 5 and verify_rows[0]['device'] ==
"macOS"), "Incorrect filter condition"
assert (verify_rows[4]['event_timestamp'] == 1592539226602157), "Incorrect sorting"
del verify_rows
```

Classroom Cleanup

```
classroom_cleanup()
```

```
Dropped the database dbacademy_odl_user_534131_databrickslabs_com_aspwd_asp_2_1l_spark_sql_lab
```