Module 6: Dataframes and Spark SQL

Case Study I Solution

Case Study: Instacart
Domain: E-commerce

Instacart is a grocery ordering and delivery app that aims to make it easy to fill your refrigerator and pantry with your personal favorites and staples when you need them. Instacart's data science team plays a big part in providing this delightful shopping experience. Currently, they use transactional data to develop models that predict which products a user will buy again, try for the first time, or add to their cart next during a session.

The dataset is a relational set of files describing customers' orders over time. The dataset is anonymized and contains a sample of over 3 million grocery orders from more than 200,000 Instacart users.

Tasks:

As a Big Data consultant, you are helping the data science team to explore the dataset using Spark:

1. Load data into Spark DataFrame
2. Merge all the data frames based on the common key and create a single DataFrame
3. Check missing data
4. List the most ordered products (top 10)
5. Do people usually reorder the same previous ordered products?
6. List most reordered products
7. Most important department and aisle (by number of products)
8. Get the Top 10 departments
9. List top 10 products ordered in the morning (6 AM to 11 AM)
10. Create a spark-submit application for the same and print the findings in the log

Dataset: You can download the required dataset from here

Solution:

- You should enter inside the PySpark Shell. ***Type pyspark2*** to enter the python spark shell for running further codes.

Step 1: Load data into Spark DataFrames

```
aisles =
spark.read.csv("hdfs://nameservice1/user/edureka_294428/aisles.csv").toDF("a
isle_id","aisle")
departments =
spark.read.csv("hdfs://nameservice1/user/edureka_294428/departments.csv").t
oDF("department_id","department")
products =
spark.read.csv("hdfs://nameservice1/user/edureka_294428/products.csv").toDF
("product_id","product_name","aisle_id","department_id")
orders =
spark.read.csv("hdfs://nameservice1/user/edureka_294428/orders.csv").toDF("
order_id","user_id","evala_set","order_number","order_dow","order_hour_of_day",
"days_since_prior_order")
orders_train =
spark.read.csv("hdfs://nameservice1/user/edureka_294428/order_products__tr
ain.csv").toDF("order_id","product_id","add_to_cart_order","reordered")
```

Step 2: Merge all the data frames based on the common key and create a single DataFrame

```
df = products.join(aisles, aisles.aisle_id ==
products.aisle_id,"left").drop(aisles.aisle_id).join(departments,
departments.department_id ==
products.department_id,"left").drop(departments.department_id).join(orders_tr
ain, orders_train.product_id ==
products.product_id,"right").drop(orders_train.product_id).join(orders,
orders.order_id == orders_train.order_id, "left").drop(orders.order_id)
```

Step 3: Check missing data
Generate filtered condition dynamically

```
filtered = df.filter(" is not null and ".join(df.columns) + " is not null")
filtered.count()
```

Step 4: List the most ordered products (top 10)

```
from pyspark.sql.functions import col
filtered.groupBy("product_name").count().sort(col("count").desc()).show(10,Fals
e)
```

Step 5: Do people usually reorder the same previous ordered products

```
from pyspark.sql.functions import *
filtered.groupBy("product_name").agg(F.avg("reordered"),F.count("reordered"))
.sort(col("count(reordered)").desc()).show(10,False)
```

Step 6: List most reordered products

```
filtered.groupBy("product_name").agg(avg("reordered"),count("reordered")).s
how(10,False)
```


Step 7: Most important department and aisle (by number of products)

```
df_prods = products.join(aisles, aisles.aisle_id ==
products.aisle_id,"left").drop(aisles.aisle_id).join(departments,
departments.department_id ==
products.department_id,"left").drop(departments.department_id)
df_prods.groupBy("aisle").count().sort(col("count").desc()).show(10,False)
df_prods.groupBy("department").count().sort(col("count").desc()).show(10,False
)
```

Step 8: Get the Top 10 departments

filtered.groupBy("department").count().sort(col("count").desc()).show(10,False)

Step 9: List top 10 products ordered in the morning (6 AM to 11 AM)

filtered.filter("order_hour_of_day >= 6 and order_hour_of_day<=11").groupBy("product_name").count().sort(col("count").desc()).show(10,False)

Step 10: Create a spark-submit application for the same and print the findings in log