**Order Data analysis (Joins)**

1. **Problem Statement: What is the daily product revenue for CLOSED or COMPLETE orders?**

**Solution:**

Spark.sql(“SELECT o.order\_date, p.product\_name, sum(ot.order\_item\_subtotal) as revenue from order o join order1 ot on o.order\_id = ot.order\_item\_order\_id join product1 p on p.product\_id = ot.order\_item\_product\_id where o.order\_status in (‘CLOSED’, ‘COMPLETE’) group b y o.order\_date, p.product\_name”).show

1. **Load the required data in to DF like categories, customer, departments, order\_items, orders and products.**

**Solution:**

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("OrderAnalysis").getOrCreate()

categories\_df = spark.read.csv("dbfs:/FileStore/tables/categories.csv", header=True, inferSchema=True)

categories\_df.show()

customers\_df = spark.read.csv("dbfs:/FileStore/tables /customers.csv", header=True, inferSchema=True)

customers\_df.show()

departments\_df = spark.read.csv("dbfs:/FileStore/tables /departments.csv", header=True, inferSchema=True)

departments\_df.show()

order\_items\_df = spark.read.csv("dbfs:/FileStore/tables /order\_items.csv", header=True, inferSchema=True)

order\_items\_df.show()

orders\_df = spark.read.csv("dbfs:/FileStore/tables /orders.csv", header=True, inferSchema=True)

orders\_df.show()

products\_df = spark.read.csv("dbfs:/FileStore/tables /products.csv", header=True, inferSchema=True)

products\_df.show()

1. **Get the count for each order status.**

**Solution:**

from pyspark.sql import SparkSession

from pyspark.sql.functions import \*

order\_status\_count = orders\_df.groupBy("order\_status").count()

order\_status\_count.show()

1. **Filter only COMPLETE or CLOSED orders.**

**Solution:**

from pyspark.sql import SparkSession

from pyspark.sql.functions import \*

filtered\_orders\_df = orders\_df.filter((col("order\_status") == "COMPLETE") | (col("order\_status") == "CLOSED"))

filtered\_orders\_df.show()

1. **Join the products , order\_items and orders tables and calculate daily product revenue.**

**Solution:**

from pyspark.sql import SparkSession

from pyspark.sql.functions import \*

joined\_df = (

order\_items\_df

.join(products\_df, order\_items\_df.product\_id == products\_df.product\_id)

.join(filtered\_orders\_df, order\_items\_df.order\_id == filtered\_orders\_df.order\_id)

.select(

filtered\_orders\_df.order\_date,

products\_df.product\_name,

order\_items\_df.order\_item\_subtotal.alias("product\_revenue")

)

)

daily\_product\_revenue = joined\_df.groupBy("order\_date", "product\_name").agg(sum("product\_revenue").alias("daily\_product\_revenue"))

1. **Write the data in to the table Daily product revenue in Hive.**

**Solution:**

from pyspark.sql import SparkSession

from pyspark.sql.functions import \*

from pyspark.sql import HiveContext

hive\_context = HiveContext(spark)

hive\_context.sql("CREATE TABLE IF NOT EXISTS DailyProductRevenue (order\_date STRING, product\_name STRING, daily\_product\_revenue DOUBLE)")

daily\_product\_revenue.write.mode("overwrite").saveAsTable("DailyProductRevenue")