**Order Data analysis (Joins)**

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

orderitemsDF=spark.read.option("header",True).option("inferschema",True).csv("order\_items.csv")

orderitemsDF.createOrReplaceTempView("orderitems")

orderitemsDF.printSchema()

>>> orderitemsDF.printSchema()

root

|-- order\_item\_id: integer (nullable = true)

|-- order\_item\_order\_id: integer (nullable = true)

|-- order\_item\_product\_id: integer (nullable = true)

|-- order\_item\_quantity: integer (nullable = true)

|-- order\_item\_subtotal: double (nullable = true)

|-- order\_item\_product\_price: double (nullable = true)

ordersDF=spark.read.option("header",True).option("inferschema",True).csv("orders.csv")

ordersDF.createOrReplaceTempView("orders")

ordersDF.printSchema()

root

|-- order\_id: integer (nullable = true)

|-- order\_date: string (nullable = true)

|-- order\_customer\_id: integer (nullable = true)

|-- order\_status: string (nullable = true)

productsDF=spark.read.option("header",True).option("inferschema",True).csv("products.csv")

productsDF.createOrReplaceTempView("products")

productsDF.printSchema()

>>> productsDF.printSchema()

root

|-- product\_id: integer (nullable = true)

|-- product\_category\_id: integer (nullable = true)

|-- product\_name: string (nullable = true)

|-- product\_description: string (nullable = true)

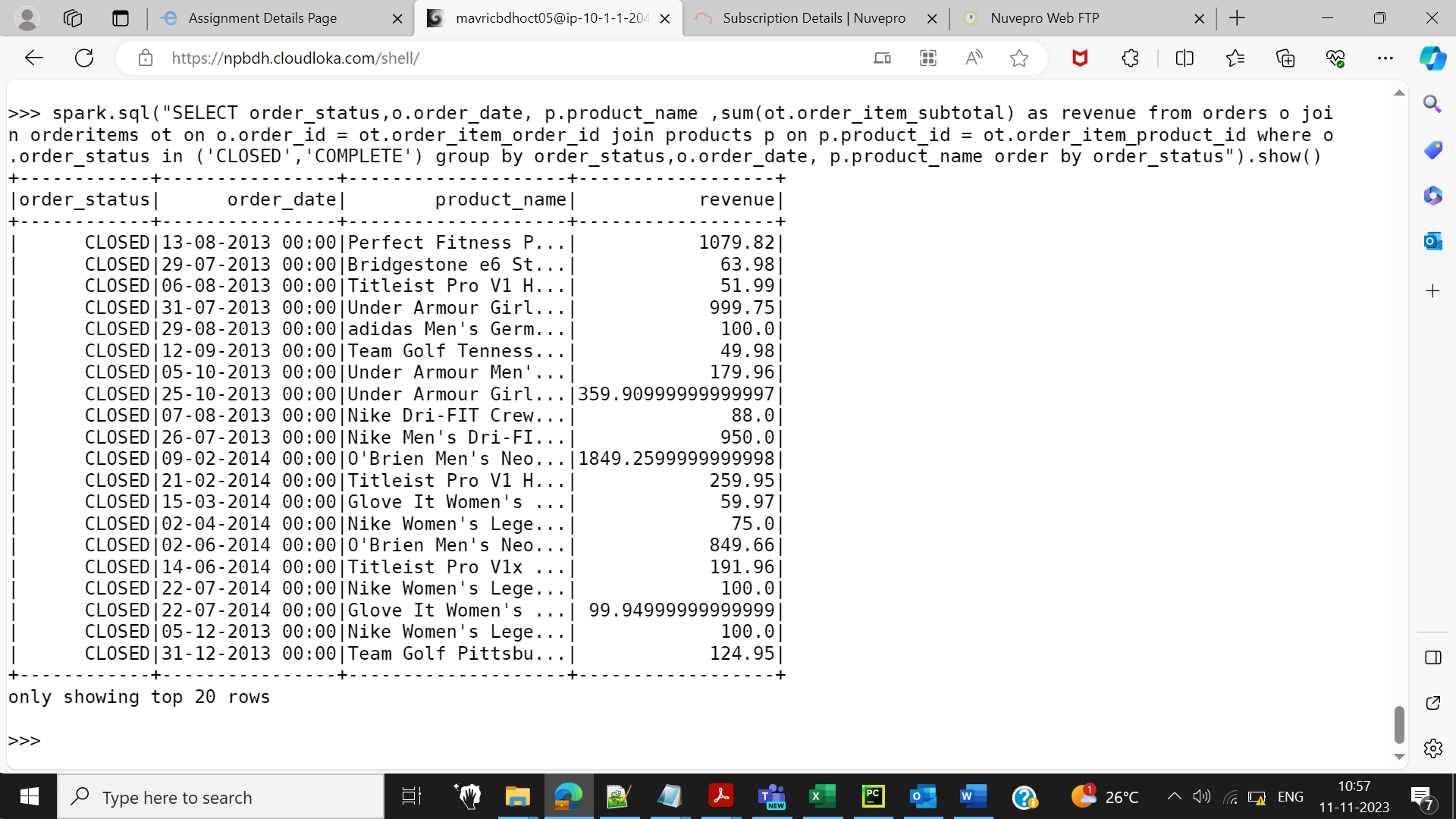
|-- product\_price: double (nullable = true)

|-- product\_image: string (nullable = true)

1.Problem Statement: What is the daily product revenue for CLOSED or

COMPLETE orders?

spark.sql("SELECT order\_status,o.order\_date, p.product\_name ,sum(ot.order\_item\_subtotal) as revenue from orders o join orderitems ot on o.order\_id = ot.order\_item\_order\_id join products p on p.product\_id = ot.order\_item\_product\_id where o.order\_status in ('CLOSED','COMPLETE') group by order\_status,o.order\_date, p.product\_name order by order\_status").show()



3.Get the count for each order status

spark.sql("SELECT order\_status, count(order\_id) from orders group by order\_status").show()

+---------------+---------------+

| order\_status|count(order\_id)|

+---------------+---------------+

|PENDING\_PAYMENT| 15030|

| COMPLETE| 22899|

| ON\_HOLD| 3798|

| PAYMENT\_REVIEW| 729|

| PROCESSING| 8275|

| CLOSED| 7556|

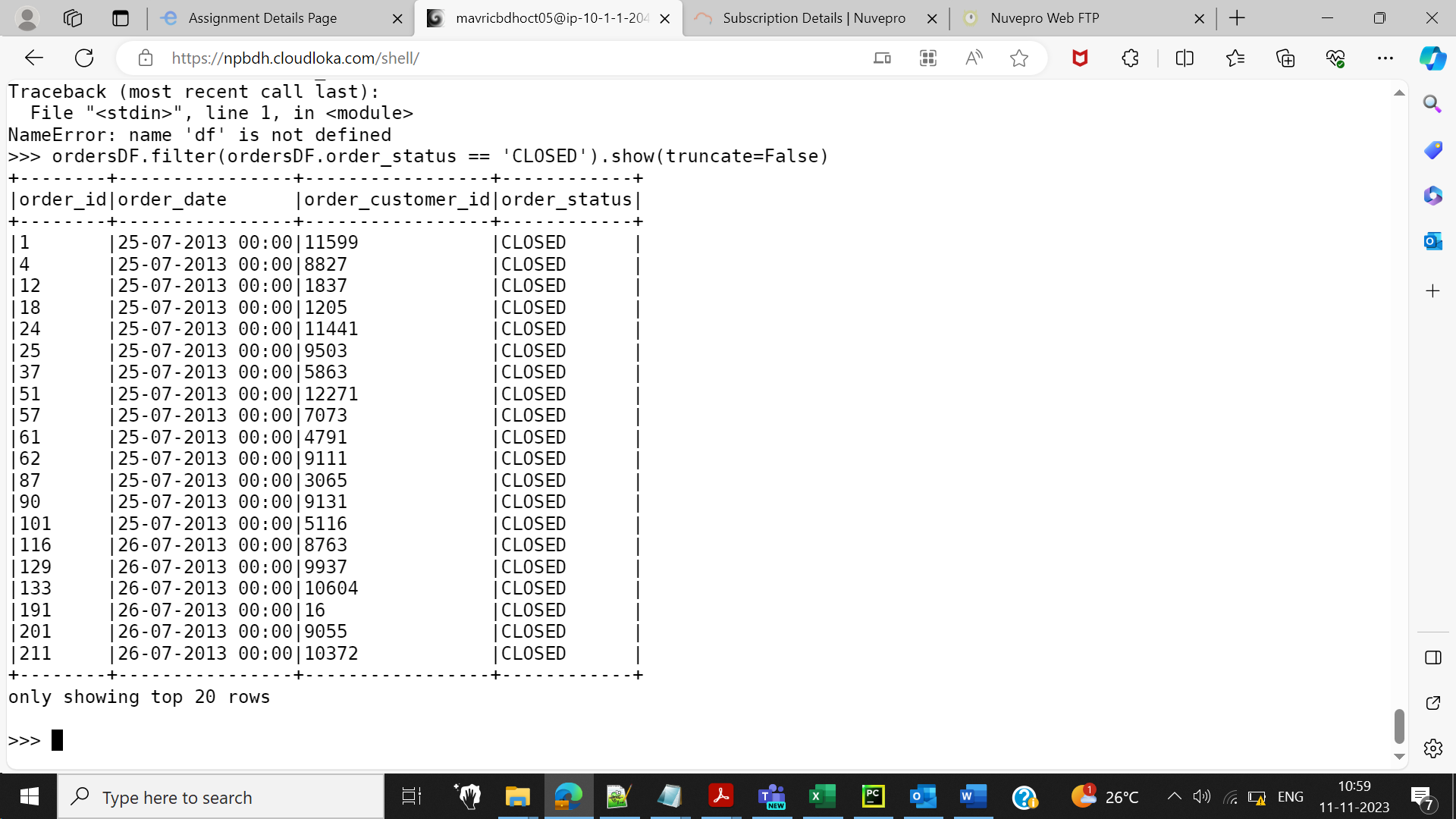
|SUSPECTED\_FRAUD| 1558|

| PENDING| 7610|

| CANCELED| 1428|

4.Filter only COMPLETE or CLOSED orders

ordersDF.filter(ordersDF.order\_status == 'CLOSED').show(truncate=False)



ordersDF.filter(ordersDF.order\_status == 'COMPLETE').show(truncate=False)

A screenshot of a computer

Description automatically generated

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

spark.sql("SELECT o.order\_date, p.product\_name ,sum(ot.order\_item\_subtotal) as revenue from orders o join orderitems ot on o.order\_id = ot.order\_item\_order\_id join products p on p.product\_id = ot.order\_item\_product\_id group by o.order\_date, p.product\_name").show()

A screenshot of a computer

Description automatically generated

6.Write the data in to the table Daily product revenue in Hive

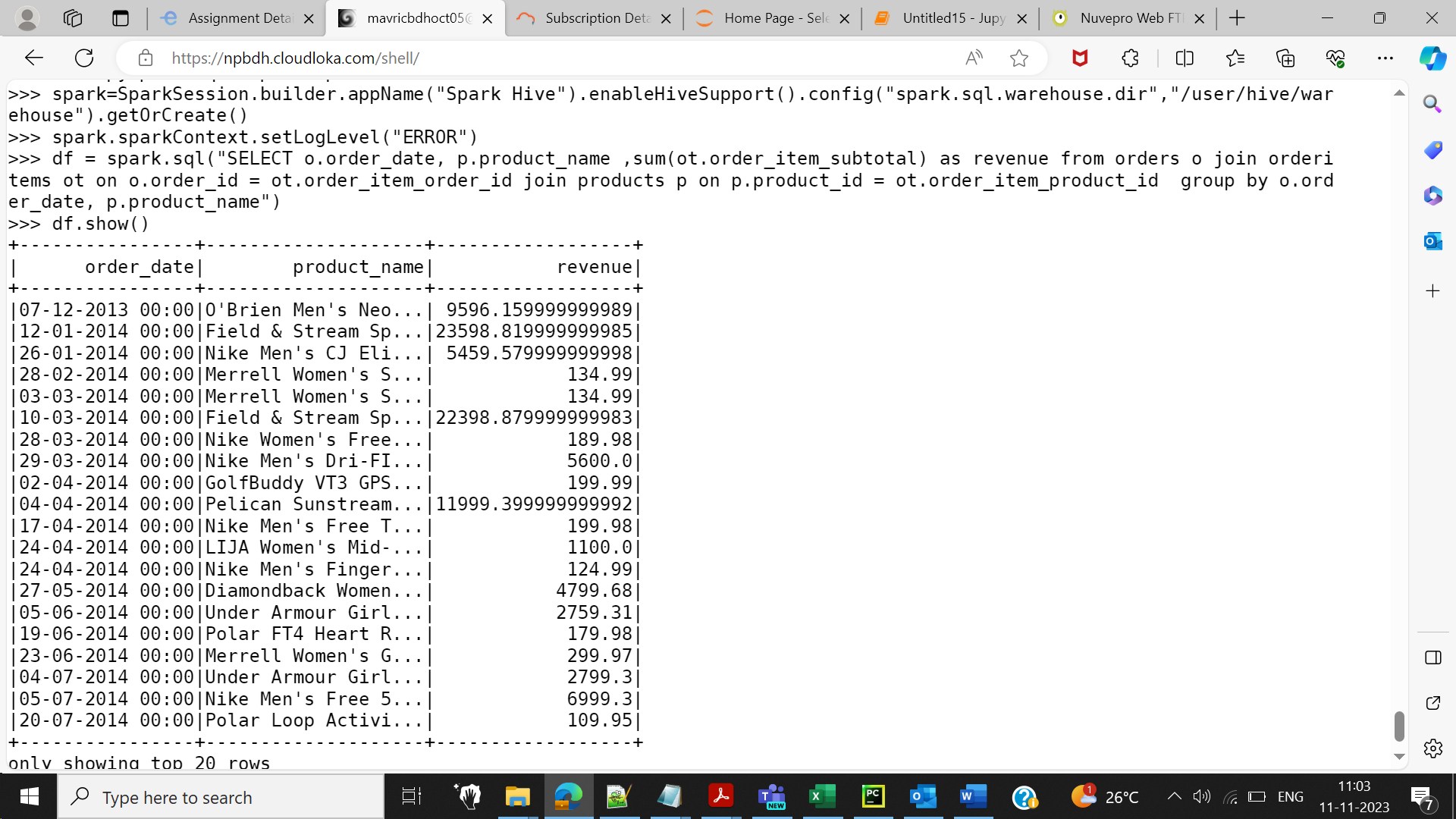
from pyspark.sql import SparkSession

spark=SparkSession.builder.appName("Spark Hive").enableHiveSupport().config("spark.sql.warehouse.dir","/user/hive/warehouse").getOrCreate()

spark.sparkContext.setLogLevel("ERROR")

df = spark.sql("SELECT o.order\_date, p.product\_name ,sum(ot.order\_item\_subtotal) as revenue from orders o join orderitems ot on o.order\_id = ot.order\_item\_order\_id join products p on p.product\_id = ot.order\_item\_product\_id group by o.order\_date, p.product\_name")

df.show()



df.write.partitionBy("product\_name").mode("overwrite").saveAsTable("aastha\_lab.daily\_revenue")

HIVE SHELL:

use aastha\_lab;

show tables;

select \* from daily\_revenue limit 10;

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated