**Order Data analysis (Joins)**

Hdfs dfs -put orderitems.csv

Hdfs dfs -put orders.csv

Hdfs dfs -put products.csv

Hdfs dfs -ls

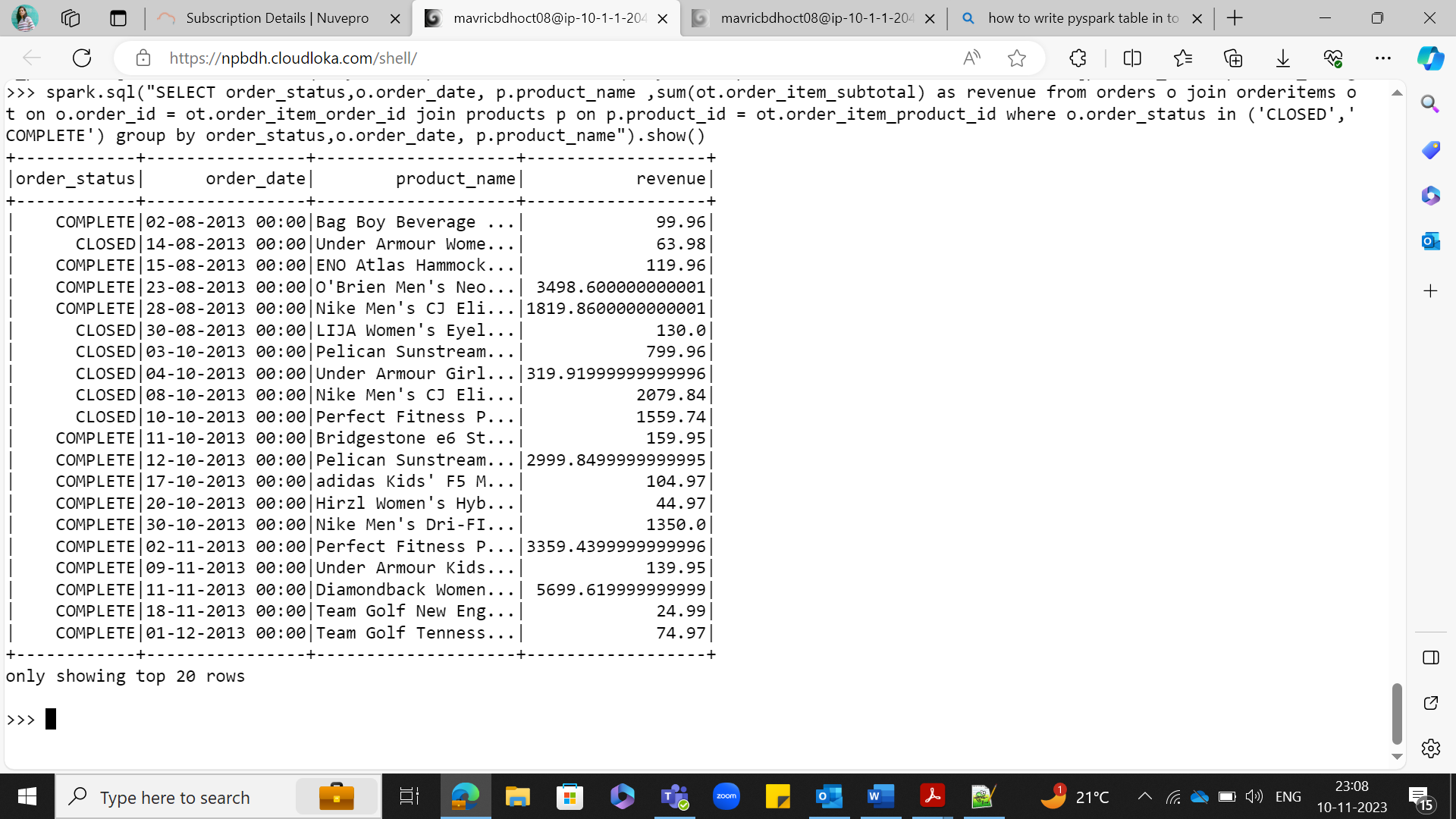
A screenshot of a computer

Description automatically generated

**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").show()



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

**DF 1 – orderitems:**

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

**orderitemsDF.createOrReplaceTempView("orderitems")**

**orderitemsDF.printSchema()**

A screenshot of a computer

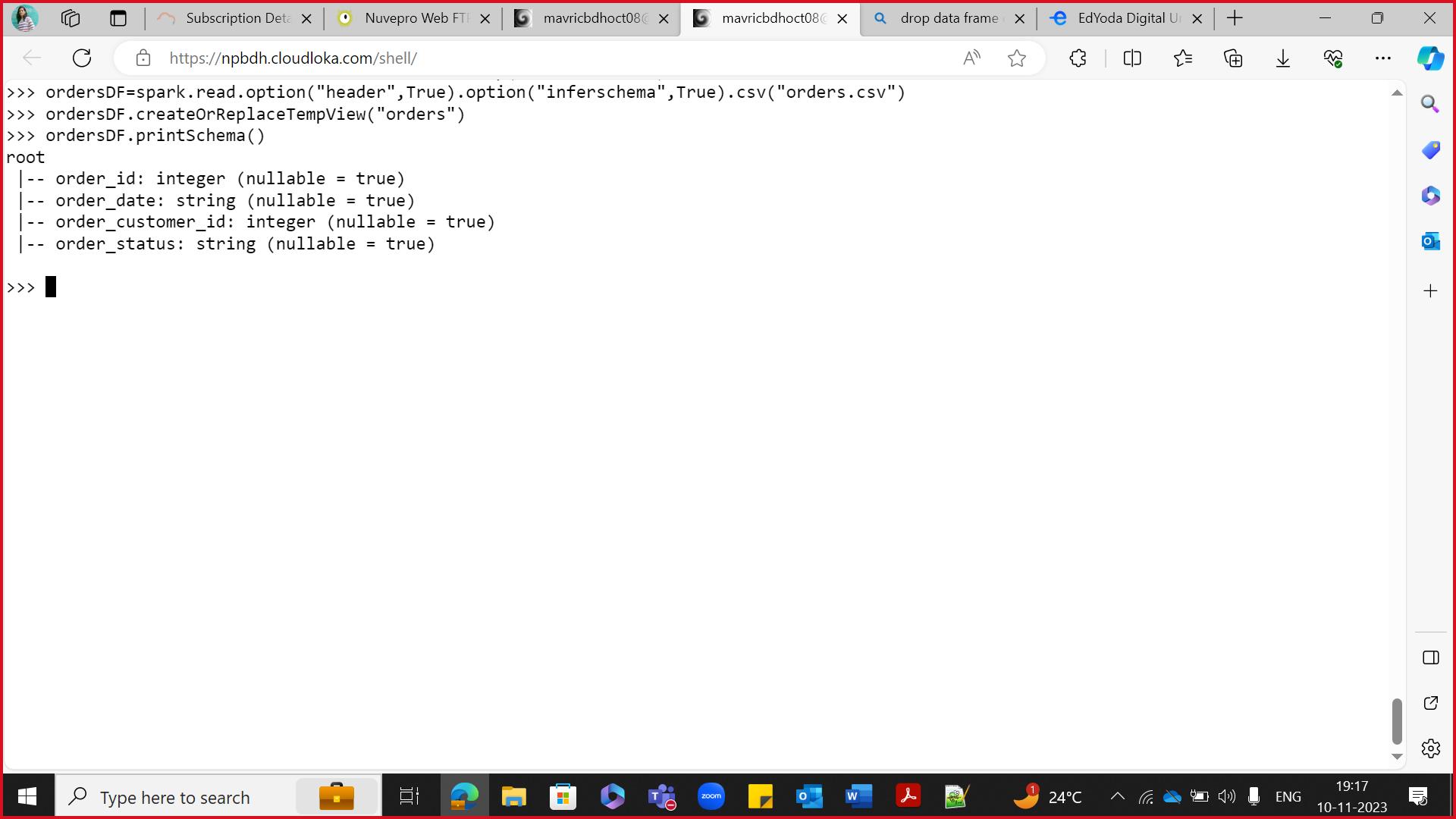
Description automatically generated

**DF 2: orders**

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

**ordersDF.createOrReplaceTempView("orders")**

**ordersDF.printSchema()**



**DF 3: products**

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

**productsDF.createOrReplaceTempView("products")**

**productsDF.printSchema()**

A screenshot of a computer

Description automatically generated

**3.Get the count for each order status**  
spark.sql("SELECT order\_status, count(order\_id) from Orders group by order\_status").show()

A screenshot of a computer

Description automatically generated

spark.sql("SELECT count(\*) from (SELECT o.order\_date, p.product\_name ,sum(ot.order\_item\_subtotal) as revenue from orders o join ord

eritems 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 o.order\_date, p.product\_name)a").show()

A screenshot of a computer

Description automatically generated

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

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

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

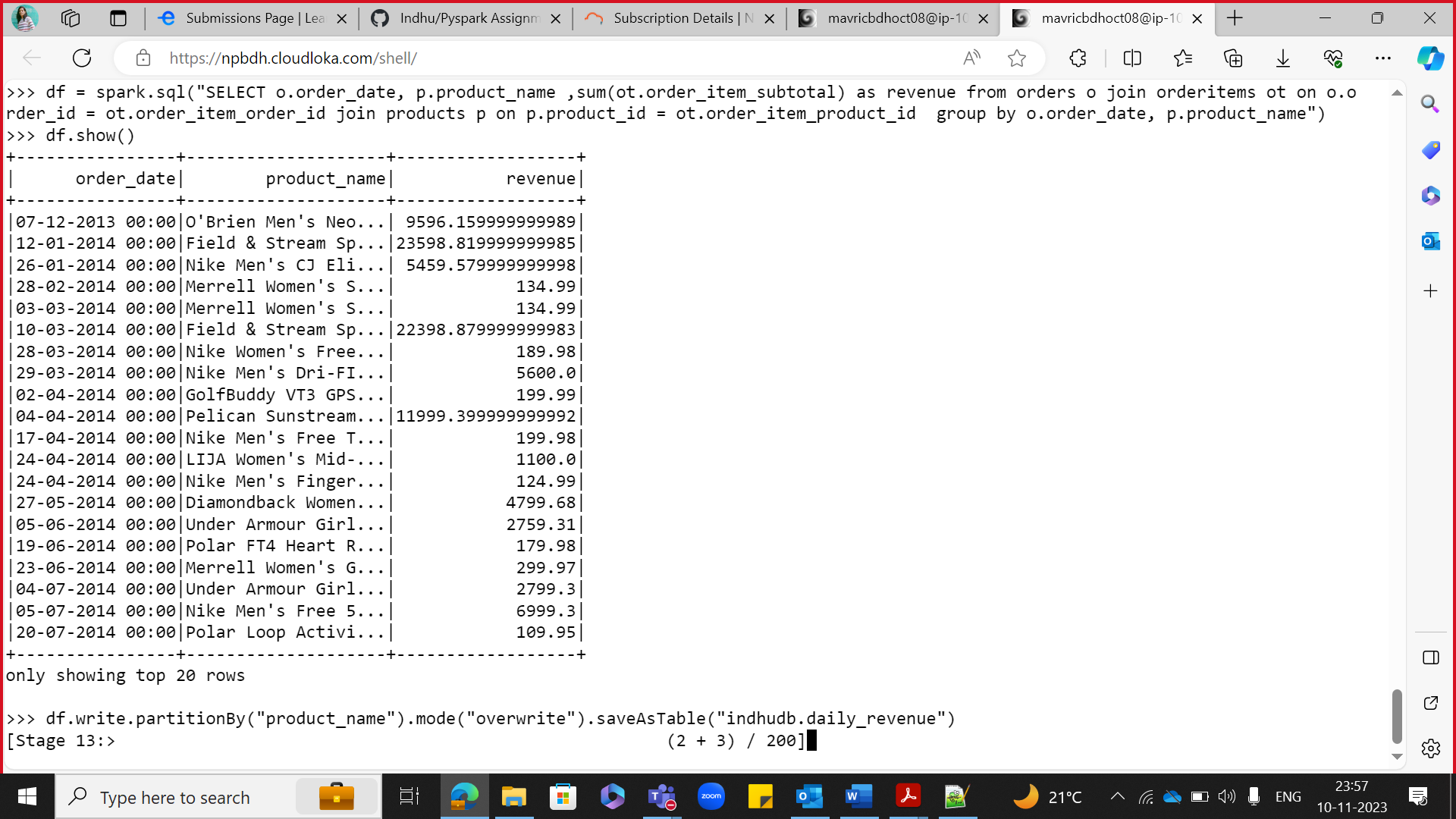


A screenshot of a computer

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()



**Count**

spark.sql("SELECT count(\*) from (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)a").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

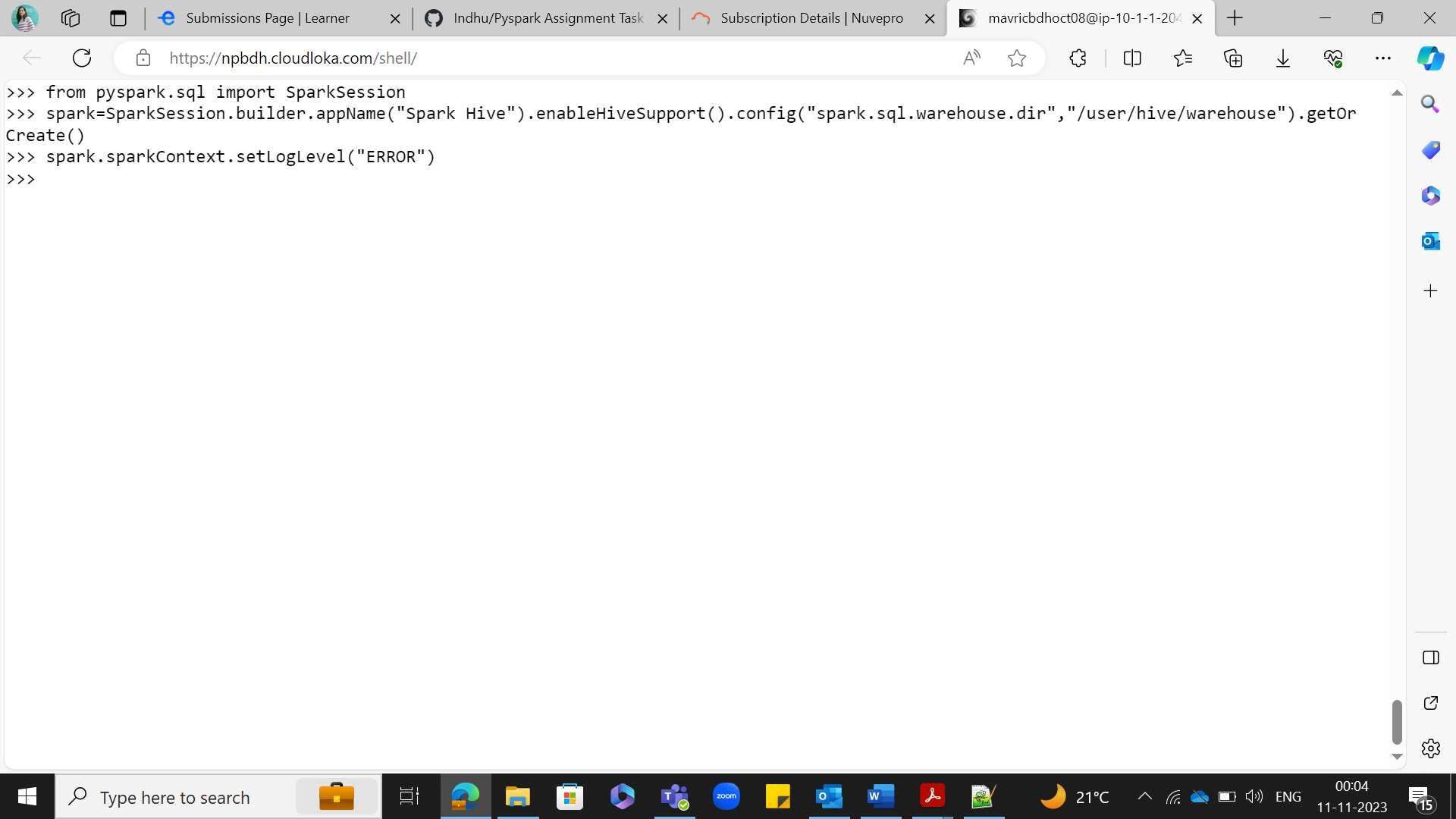
spark=SparkSession.builder.appName("Spark Hive").enableHiveSupport().config("spark.sql.warehouse.dir","/user/hive/warehouse").getOrCreate()

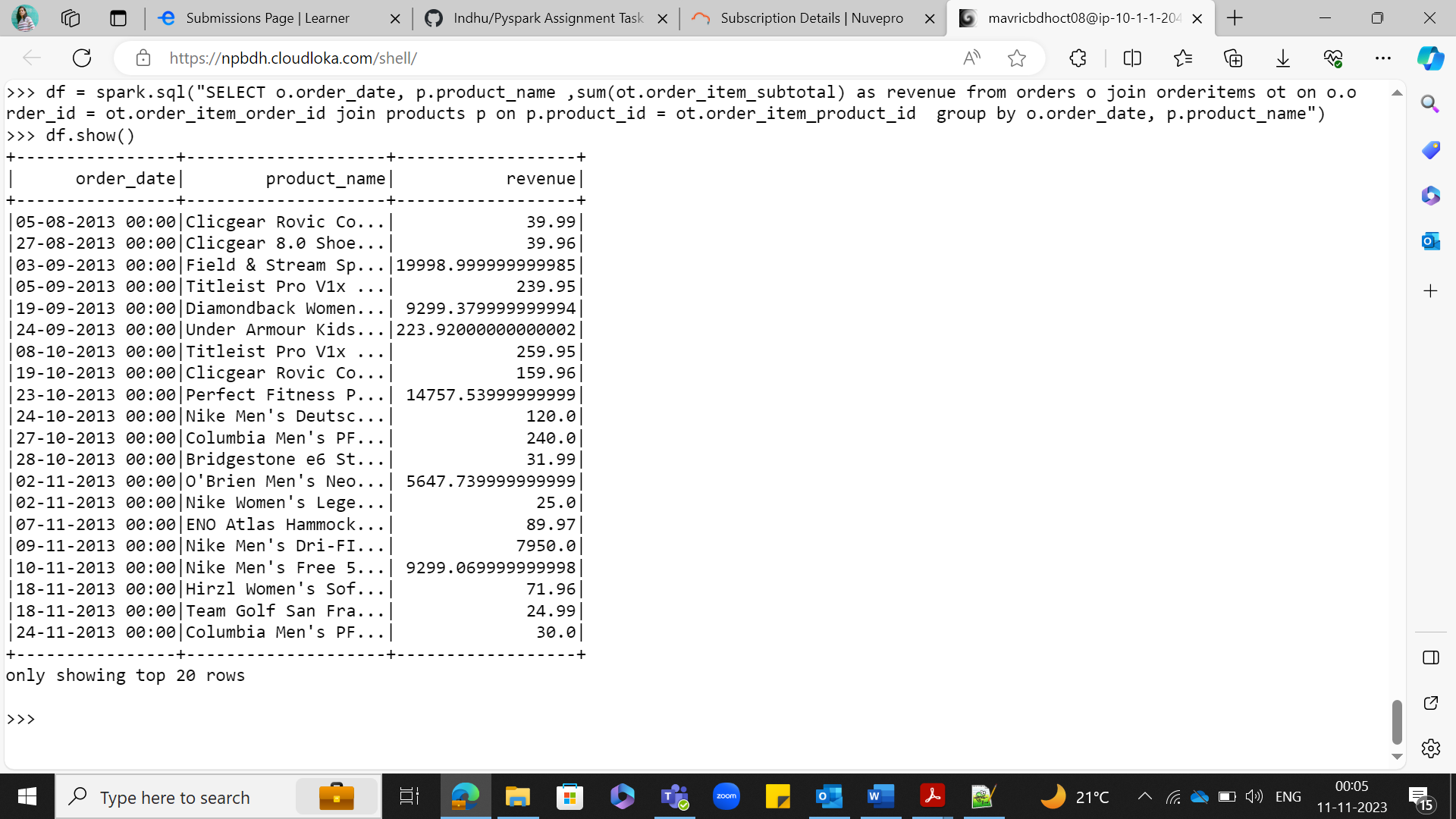
spark.sparkContext.setLogLevel("ERROR")

df = spark.sql("SELECT o.order\_date, p.product\_name ,sum(ot.order\_item\_subtotal) as revenue from orders o join ordersitems 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("indhudb.daily\_revenue")





A screenshot of a computer

Description automatically generated

**IN HIVE SHELL:**

use indhudb;

show tables;

A screenshot of a computer

Description automatically generated

select \* from daily\_revenue limit 10;

A screenshot of a computer

Description automatically generated

select count(\*) from daily\_revenue;

**Output = 14344**

A screenshot of a computer

Description automatically generated