Name: Khanh Nguyen

Date: 11/20/2022

**E-Commerce Analytics.**

Project 2

DESCRIPTION

You work for an e-commerce company as a big data consultant. Your job entails analyzing sales data. The company operates at a number of locations around the world. They want you to analyze the data from their sales transactions on a daily and weekly basis and want you to derive significant insights to understand their sales in various cities and states. You've also been asked to include a variety of other details (that are provided below) about the product evaluation.

**Description:**

 Use Spark features for data analysis to derive valuable insights

**Domain:**E-commerce

**Analysis to be done:**Exploratory analysis to determine actionable insights

**Dataset File:** olist\_public\_dataset.csv

**Dataset Description:**

1. Id
2. order\_status
3. order\_products\_value
4. order\_freight\_value
5. order\_items\_qty
6. order\_purchase\_timestamp
7. order\_aproved\_at
8. order\_delivered\_customer\_date
9. customer\_city
10. customer\_state
11. customer\_zip\_code\_prefix
12. product\_name\_lenght
13. product\_description\_lenght
14. product\_photos\_qty
15. review\_score

**Insights on Historical Data**

**1. Daily insights**

**a. Sales**

* Total sales
* Total sales in each city
* Total sales in each state

1. **Orders**

* Total number of orders
* City-wise order distribution
* State-wise order distribution
* Average review score per order
* Average freight charges per order
* Average time taken to approve the orders (order approved –

order purchased)

* Average order delivery time

**2.       Weekly insights**

**a. Sales**

* Total sales
* Total sales in each city
* Total sales in each state

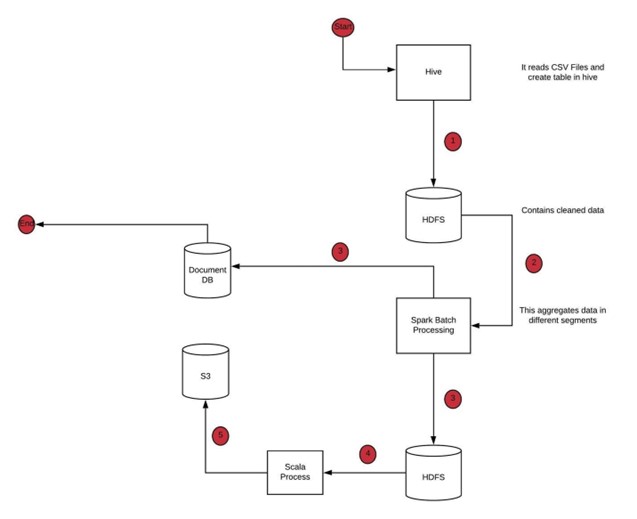
**b. Orders**

* Total number of orders
* City-wise order distribution
* State-wise order distribution
* Average review score per order
* Average freight charges per order
* Average time taken to approve the orders (order approved – order purchased)
* Average order delivery time

**c . Total freight charges**

**d. Distribution of freight charges in each city**

**Approach**



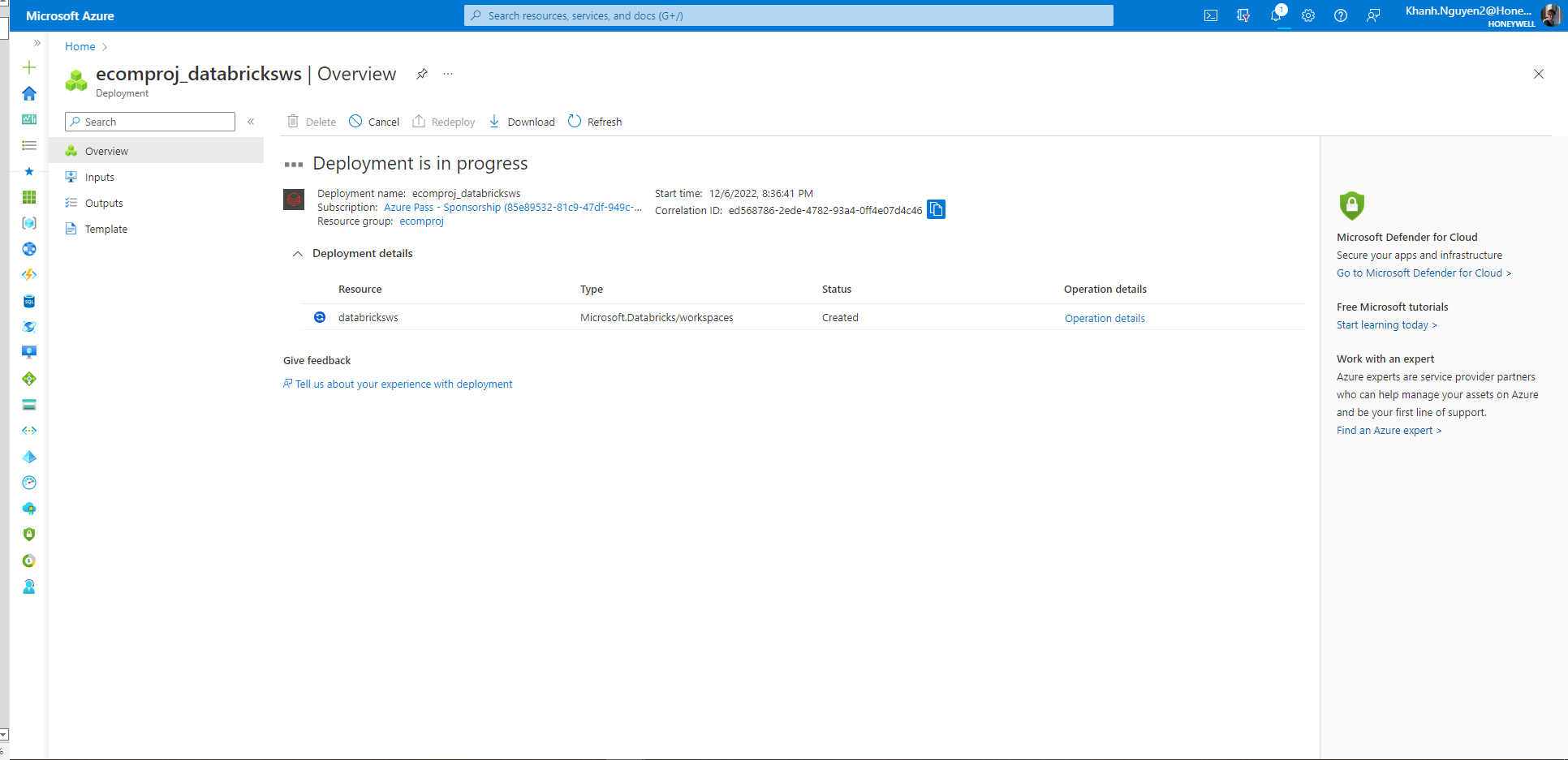
**Tasks to perform:**

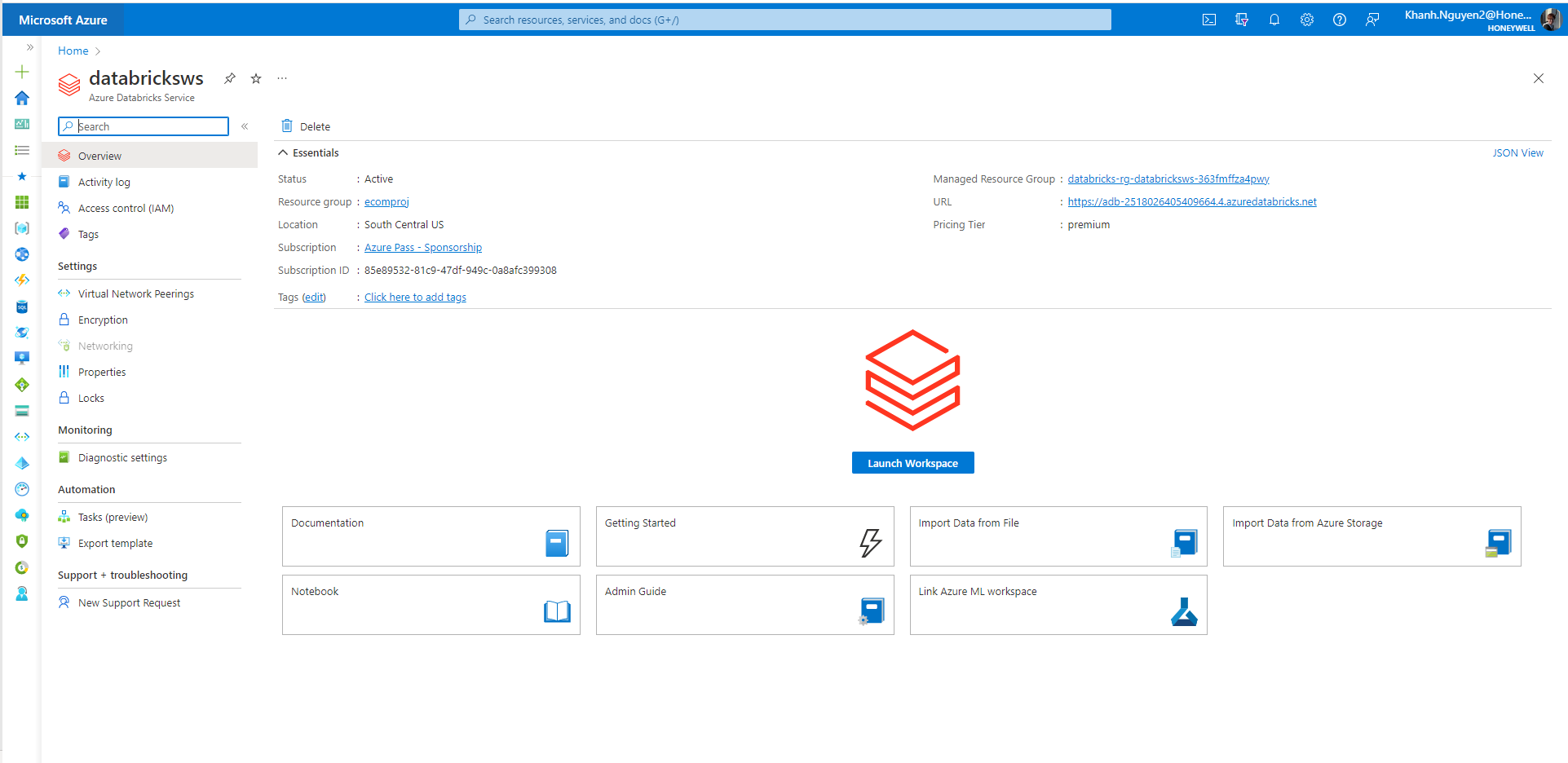
**Week 1: Overview and basic configurations**

**Step 1:**Choose a suitable cloud provider and set up a Spark shell environment – **Microsoft Azure**

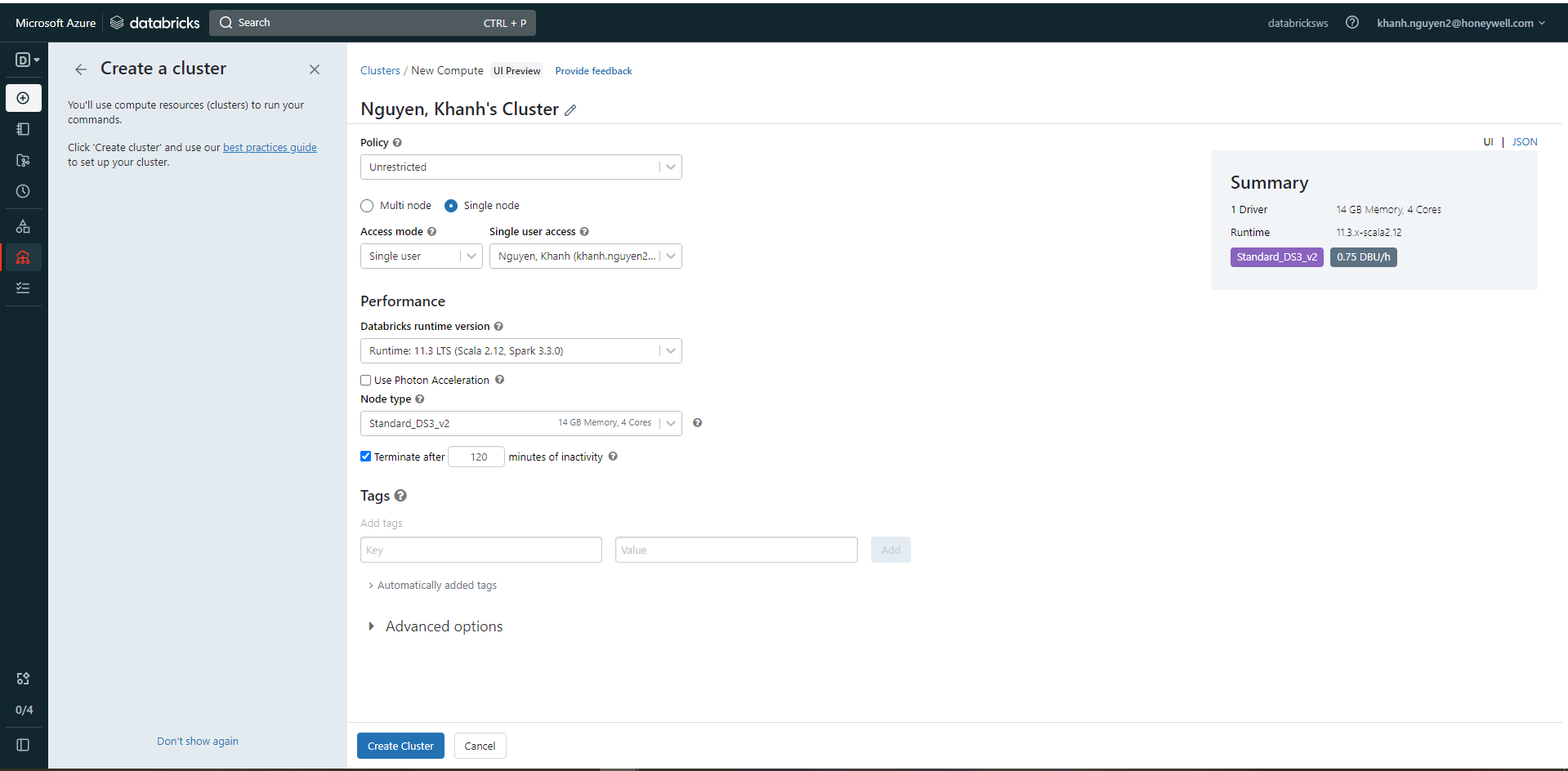
**Step 2:** Configure the necessary dependencies

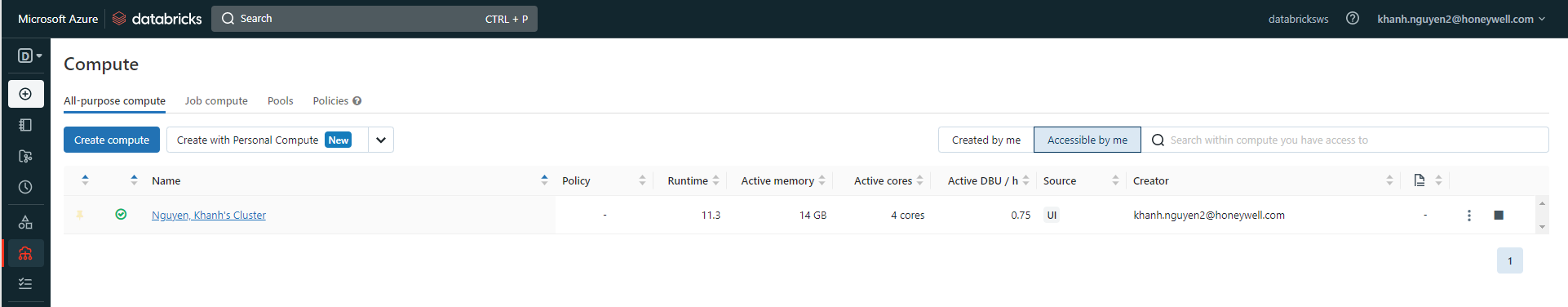
* + Create a new Azure Databricks workspace and a resource group





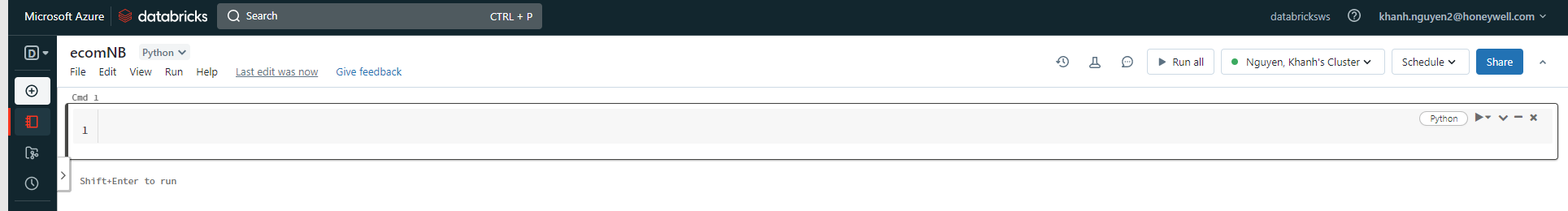
* + Launch Databricks Workspace then Create the Cluster to run the jobs

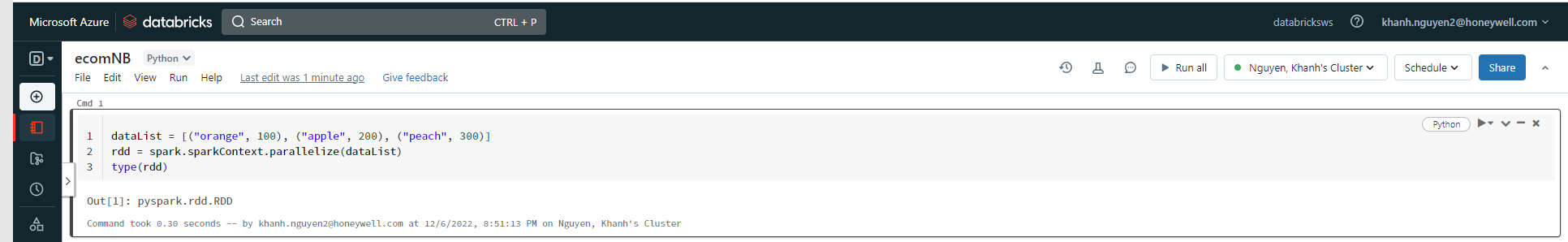




**Step 3:** Execute basic Spark commands to make sure Spark is ready

* + Create a Notebook and run the basic Spark commands





**Step 4:** Use README.md for details, instructions, and commands

* + Read the ‘README.md’ file and understood.

**Week 2: Data ingestion**

**Step 1:**Upload the entire data into Hive from CSV using cloud provider cluster

              setup (such as, EMR)

1. Log in to PuTTY with the username “hadoop”
2. Enter the command given below:

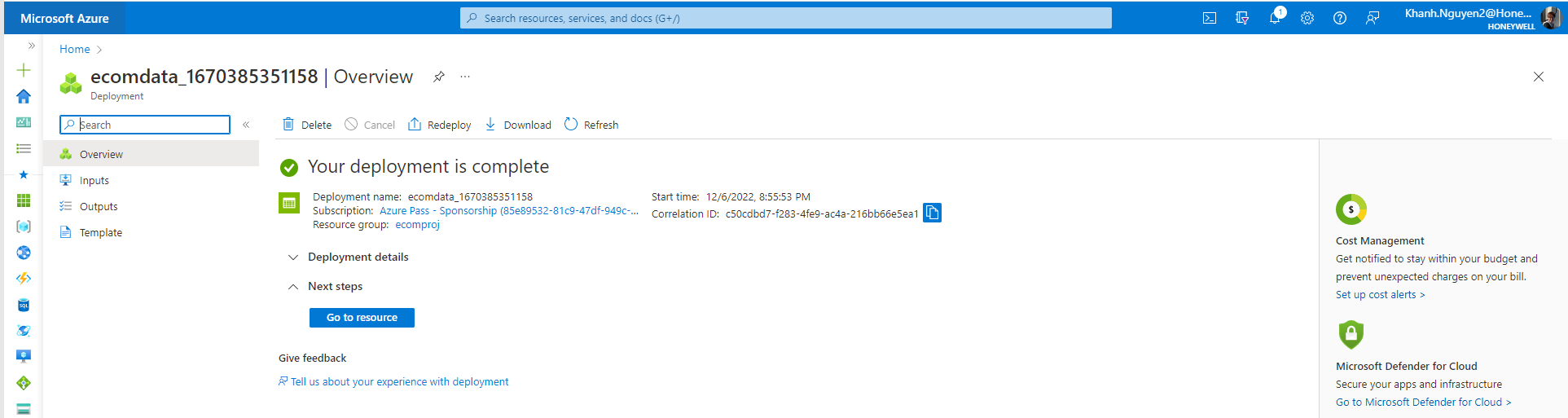
Command: hive

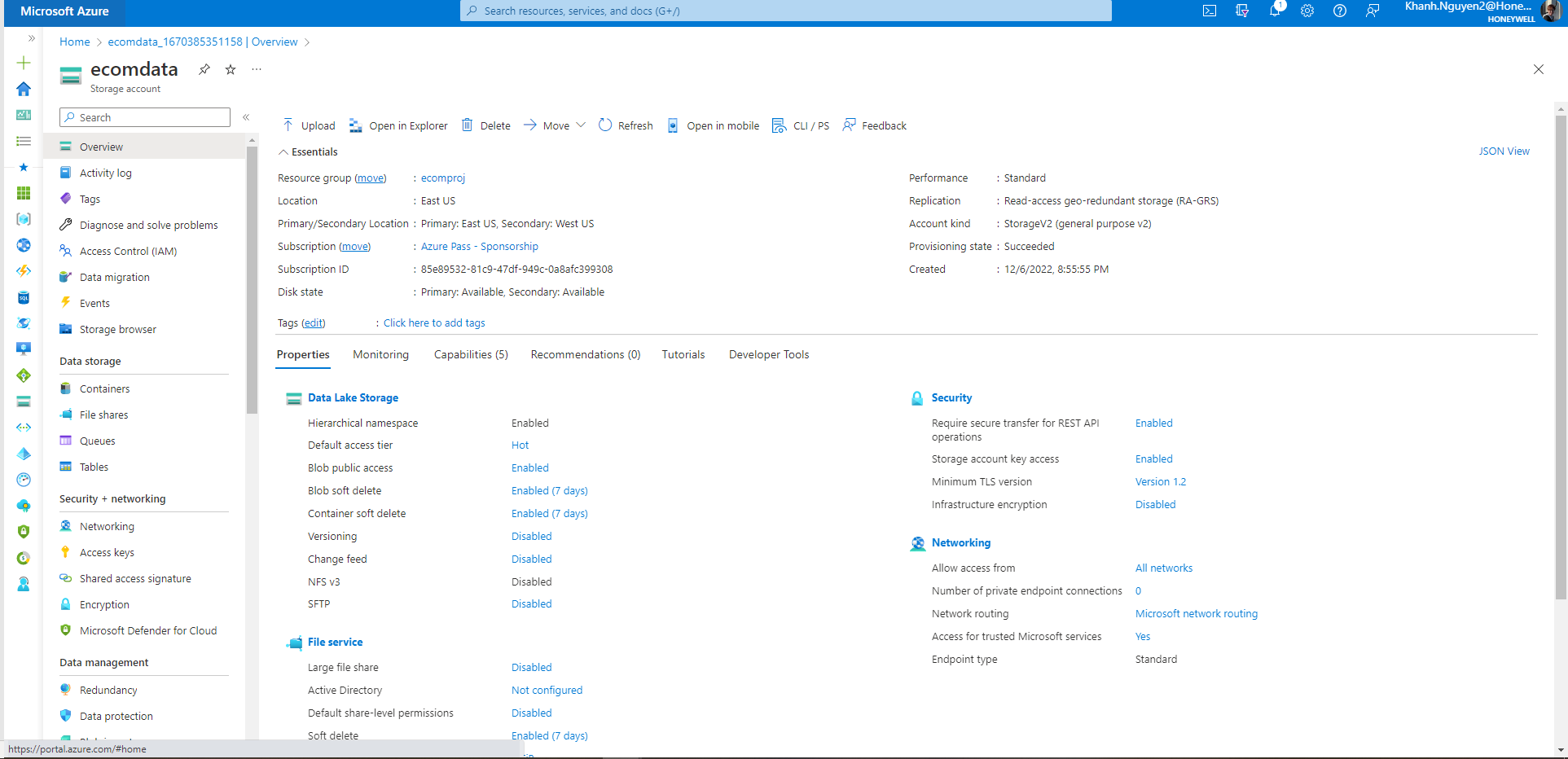
1. Create a database
2. Create a table with all the relevant details

* As I’m using Databricks, the setup takes care of all of the above configurations.

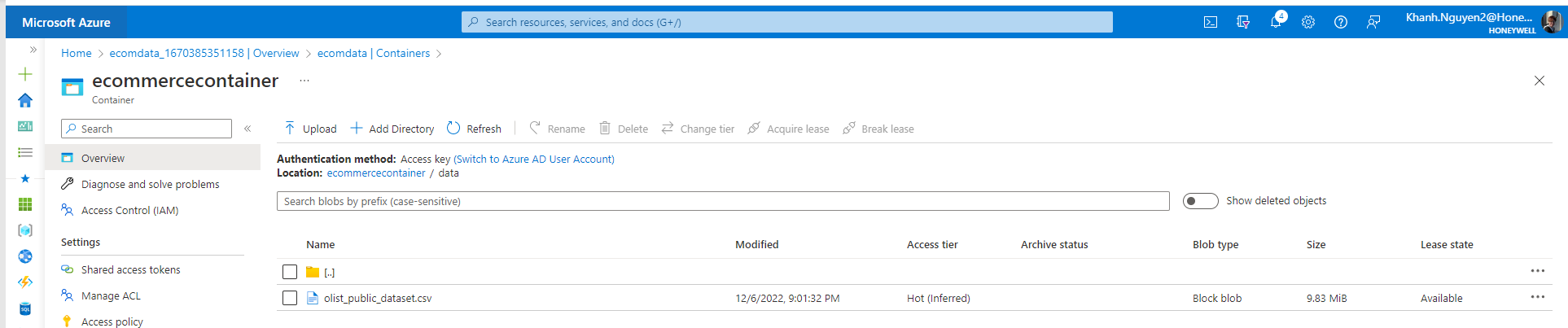
**Step 2:** Create a bucket (for example: S3 and Azure Blob) and upload the csv file

* Create an Azure Gen2 Storage Account named “ecomdata”





* Create a new container called “ecommercecontainer” and upload the input file into a folder called “data”



**Step 3:** Load the data from the bucket into the Hive table

containerName = "ecommercecontainer"

storageAccountName = " ecomdata"

accountkey = " pH8OmS7sQDnGT+IaaTTKahXM8LWvMmPFNUnyV8ljrnK83bebEpx+gCNBrdHCOMHBMD9brg6vcnDc+AStB4oVOA==” #Copied from Access keys

config = "fs.azure.sas." + containerName+ "." + storageAccountName + ".blob.core.windows.net"

spark.conf.set("fs.azure.account.key.{storage}.dfs.core.windows.net".format(storage=storageAccountName), accountkey)

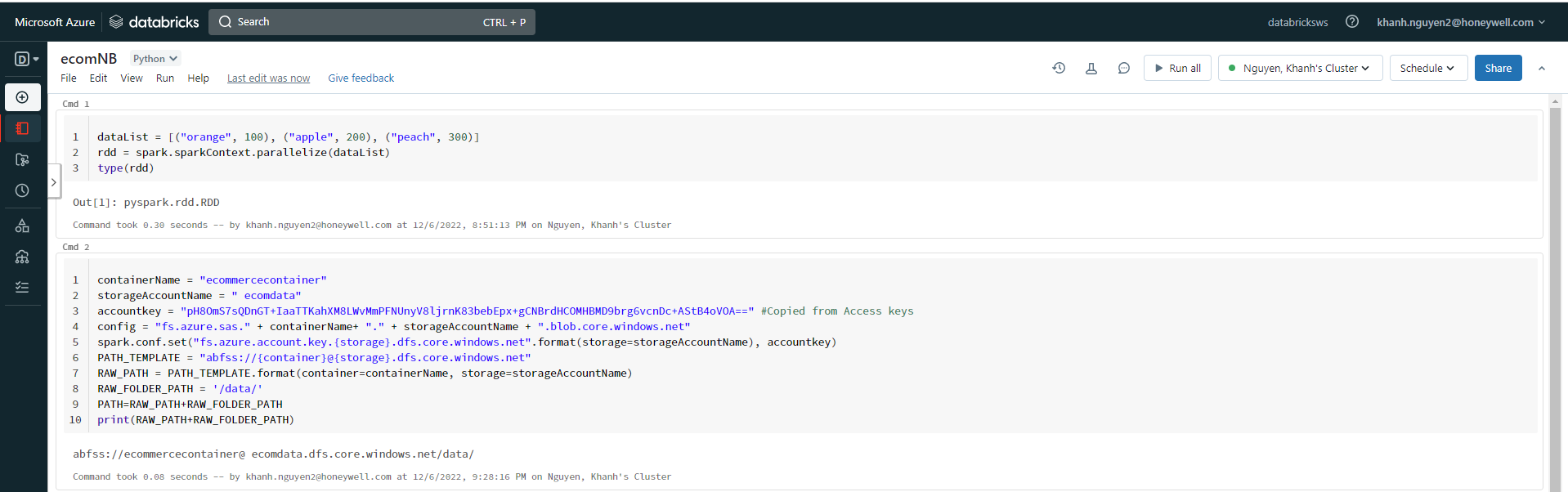
PATH\_TEMPLATE = "abfss://{container}@{storage}.dfs.core.windows.net"

RAW\_PATH = PATH\_TEMPLATE.format(container=containerName, storage=storageAccountName)

RAW\_FOLDER\_PATH = '/data/'

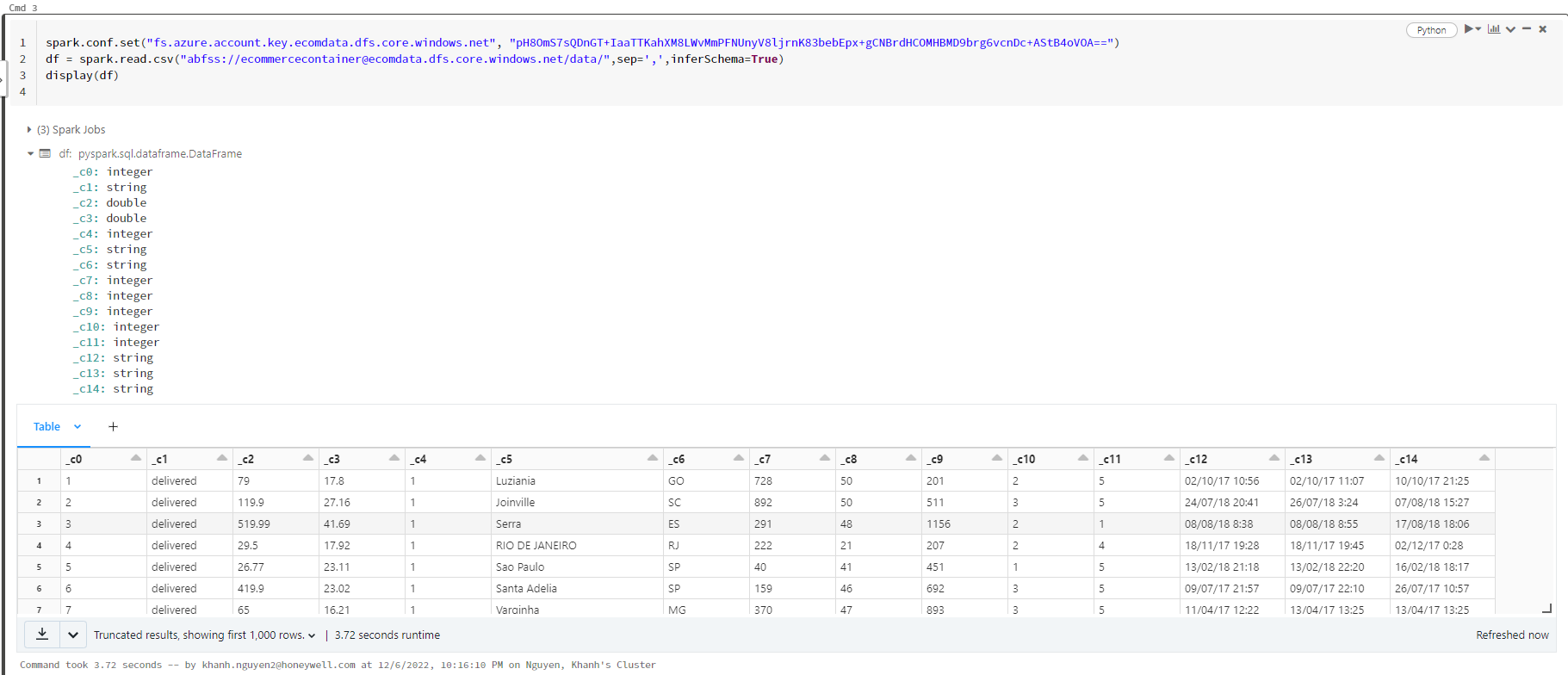
PATH=RAW\_PATH+RAW\_FOLDER\_PATH

print(RAW\_PATH+RAW\_FOLDER\_PATH)



spark.conf.set("fs.azure.account.key.ecomdata.dfs.core.windows.net", "pH8OmS7sQDnGT+IaaTTKahXM8LWvMmPFNUnyV8ljrnK83bebEpx+gCNBrdHCOMHBMD9brg6vcnDc+AStB4oVOA==")

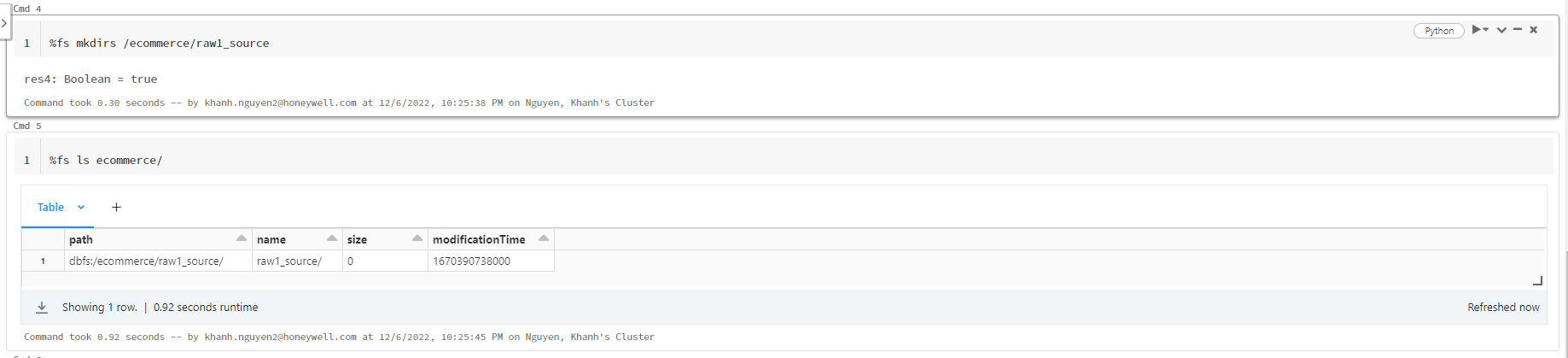
df=spark.read.csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/data/",sep=',',inferSchema=True)

display(df)

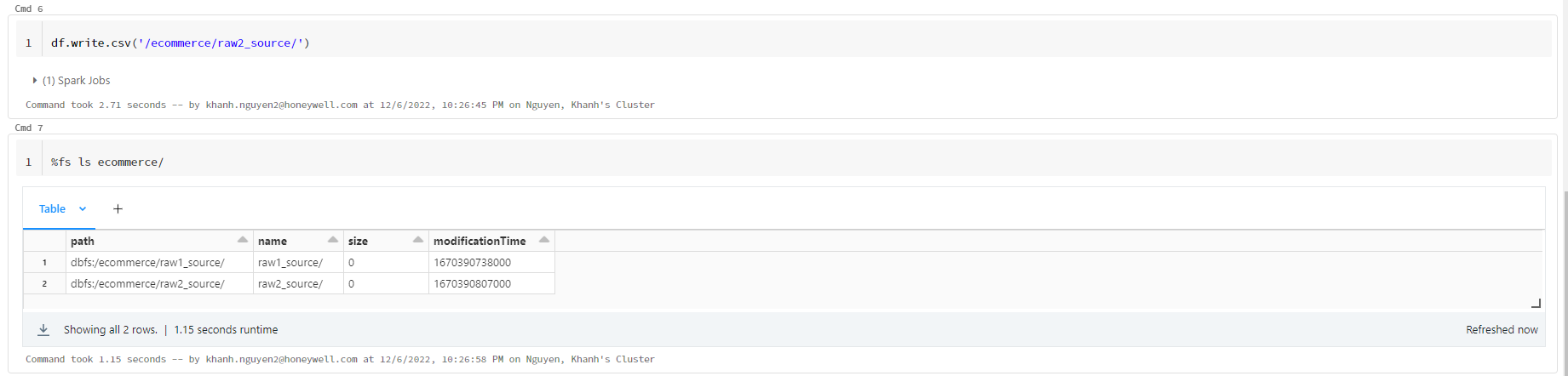
**Step 4:**Create a new directory in HDFS and copy the data from Hive into HDFS

%fs mkdirs /ecommerce/raw1\_source

%fs ls ecommerce/

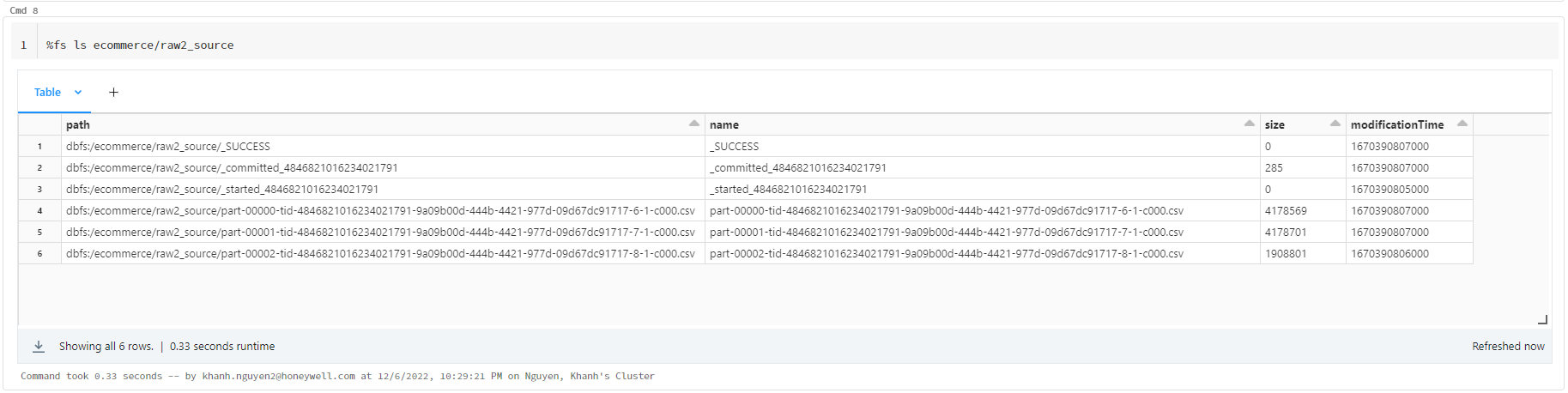


df.write.csv('/ecommerce/raw2\_source/')



**Step 5:**Check if the data has been successfully loaded in the HDFS path

 %fs ls ecommerce/raw2\_source



**Week 3: Data streaming**

**Step 1:**Connect to Spark shell with all the dependencies (Hive, Hadoop,and HDFS).

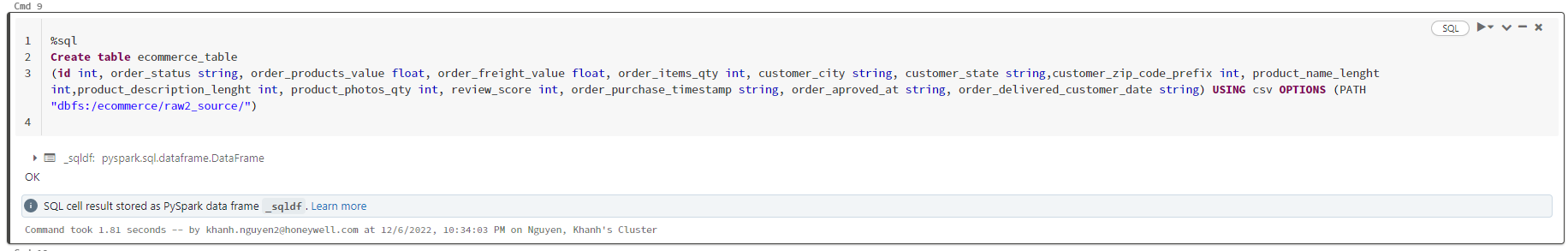
1. Create Schema of the CSV files
2. Create a Spark session

* Add Object Storage Service details as per the Cloud provider
* Add all variables to your environment as they contain sensitive data

%sql

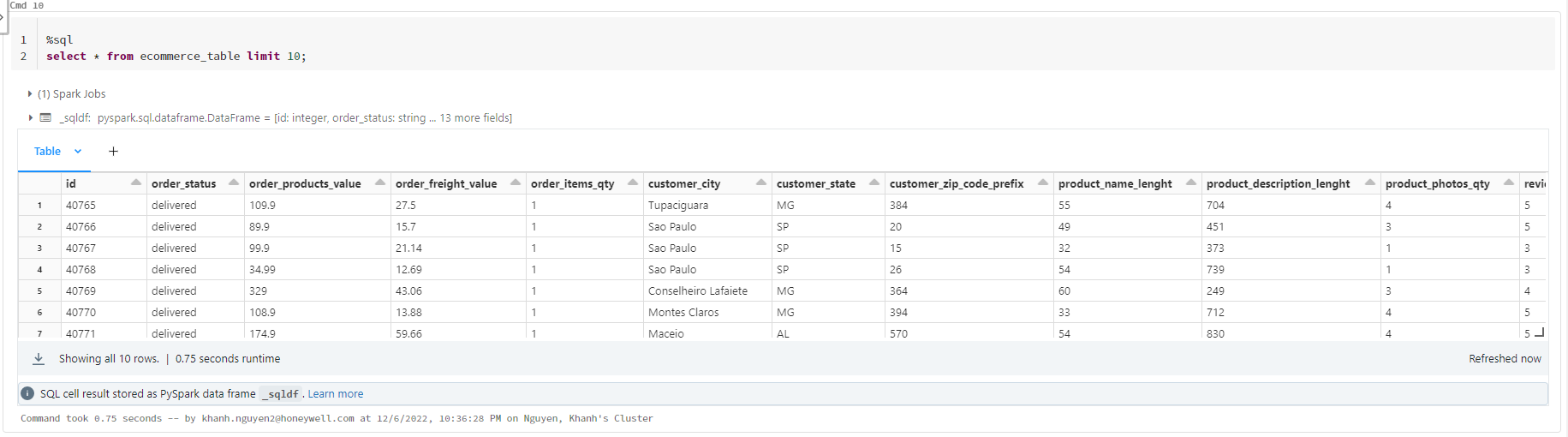
Create table ecommerce\_table

(id int, order\_status string, order\_products\_value float, order\_freight\_value float, order\_items\_qty int, customer\_city string, customer\_state string,customer\_zip\_code\_prefix int, product\_name\_lenght int,product\_description\_lenght int, product\_photos\_qty int, review\_score int, order\_purchase\_timestamp string, order\_aproved\_at string, order\_delivered\_customer\_date string) USING csv OPTIONS (PATH "dbfs:/ecommerce/raw2\_source/")



%sql

select \* from ecommerce\_table limit 10;

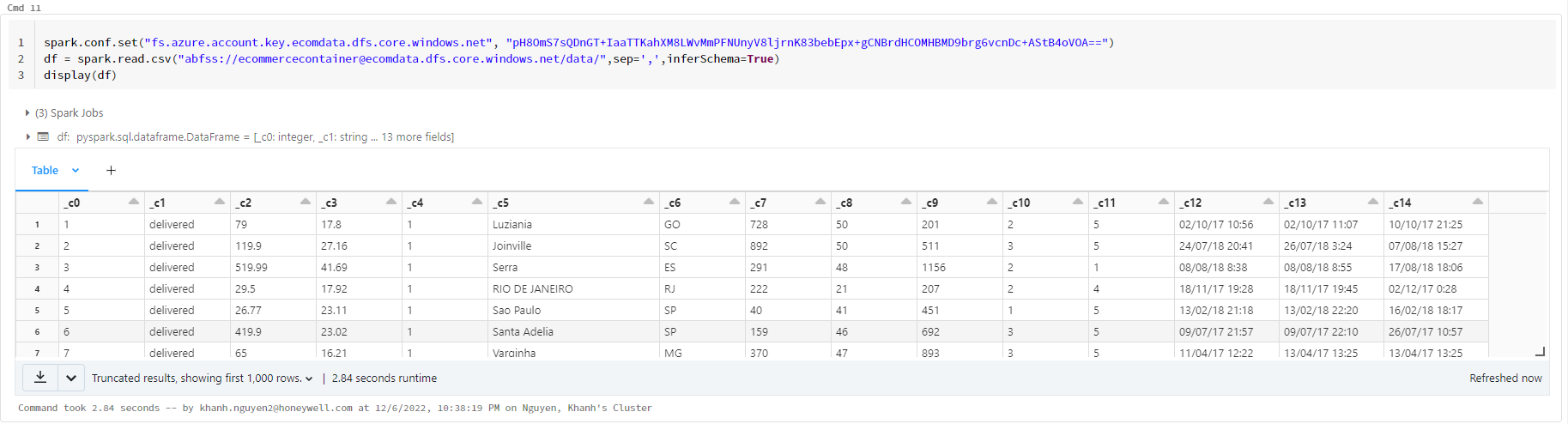


**Step 2:** Read the CSV file and convert the file to a data frame

spark.conf.set("fs.azure.account.key.ecomdata.dfs.core.windows.net", "pH8OmS7sQDnGT+IaaTTKahXM8LWvMmPFNUnyV8ljrnK83bebEpx+gCNBrdHCOMHBMD9brg6vcnDc+AStB4oVOA==")

df=spark.read.csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/data/",sep=',',inferSchema=True)

display(df)



**Step 3:** Convert “order\_purchase\_timestamp” to week and day using UDF

* Converted the timestamp to week and day using Pyspark.

from pyspark.sql import functions as F

df = spark.table("ecommerce\_table")

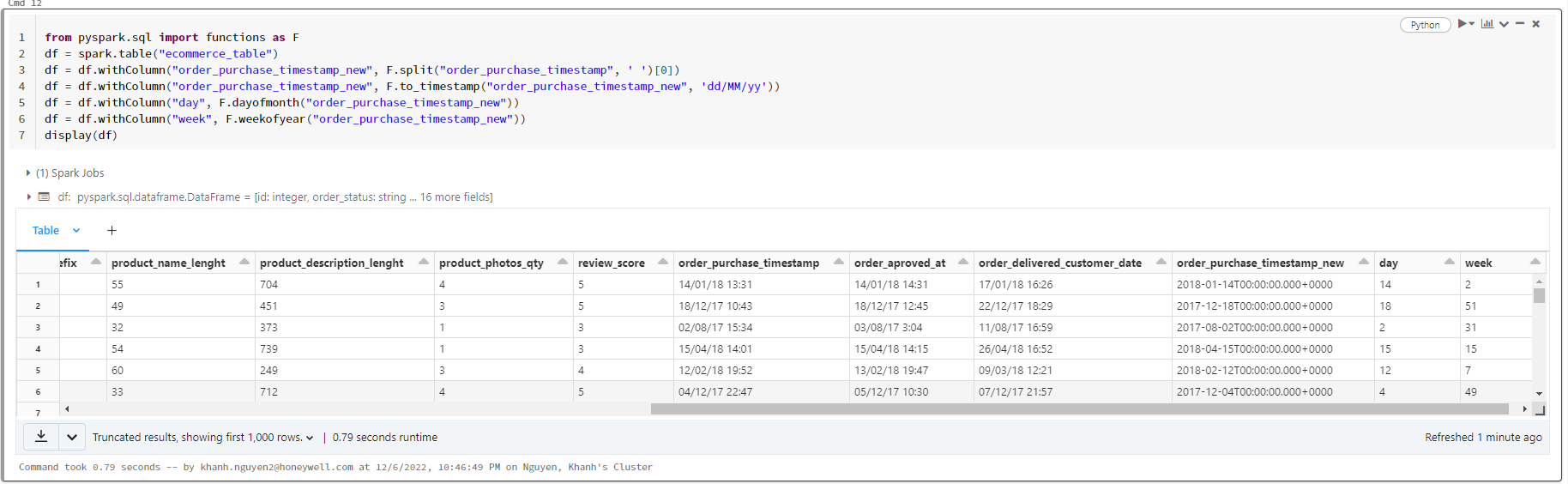
df = df.withColumn("order\_purchase\_timestamp\_new", F.split("order\_purchase\_timestamp", ' ')[0])

df = df.withColumn("order\_purchase\_timestamp\_new", F.to\_timestamp("order\_purchase\_timestamp\_new", 'dd/MM/yy'))

df = df.withColumn("day", F.dayofmonth("order\_purchase\_timestamp\_new"))

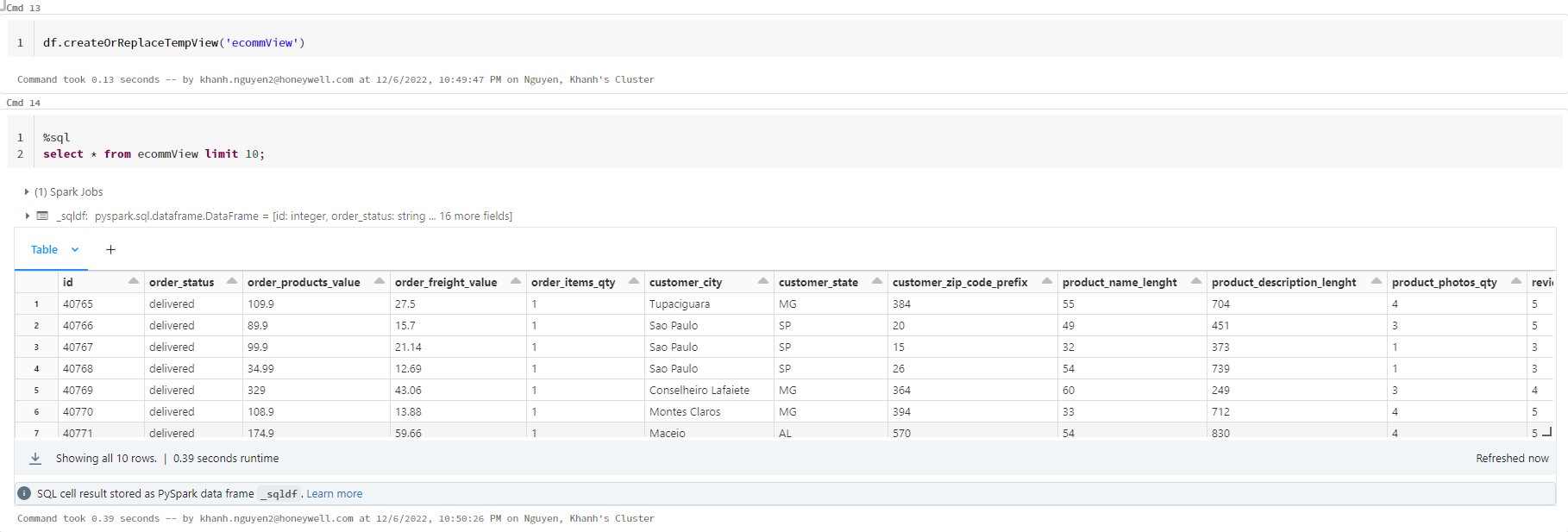
df = df.withColumn("week", F.weekofyear("order\_purchase\_timestamp\_new"))

display(df)



**Step 4:**  Calculate the following data:

* Create a temp view



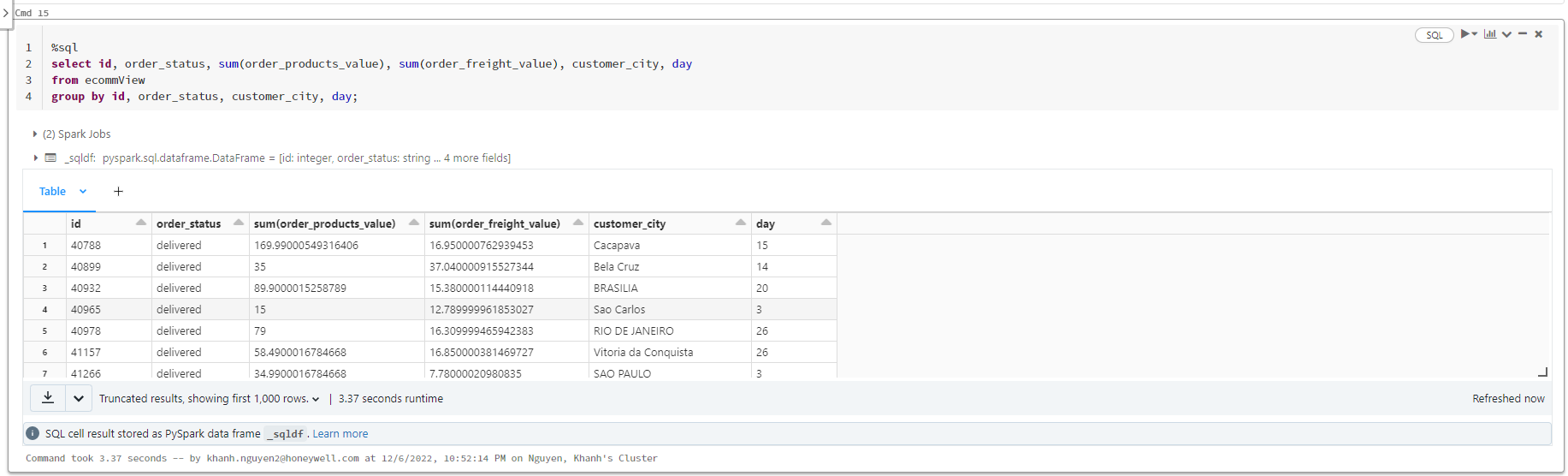
1. Total sales and order distribution per day and week for each city

%sql

select id, order\_status, sum(order\_products\_value), sum(order\_freight\_value), customer\_city, day

from ecommView

group by id, order\_status, customer\_city, day;

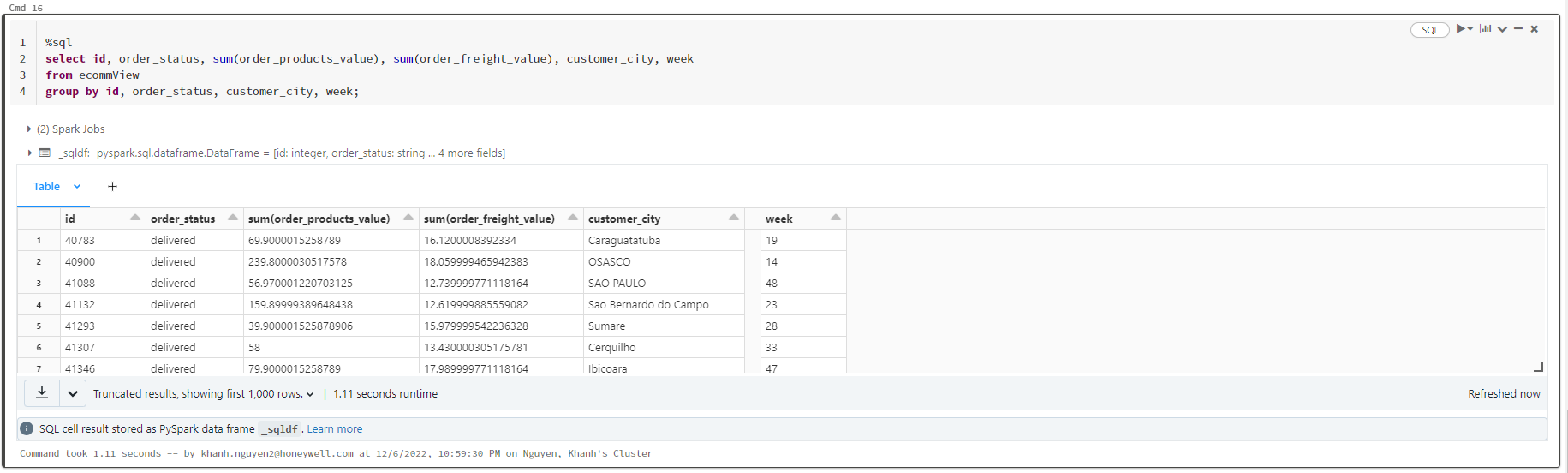


%sql

select id, order\_status, sum(order\_products\_value), sum(order\_freight\_value), customer\_city, week

from ecommView

group by id, order\_status, customer\_city, week;



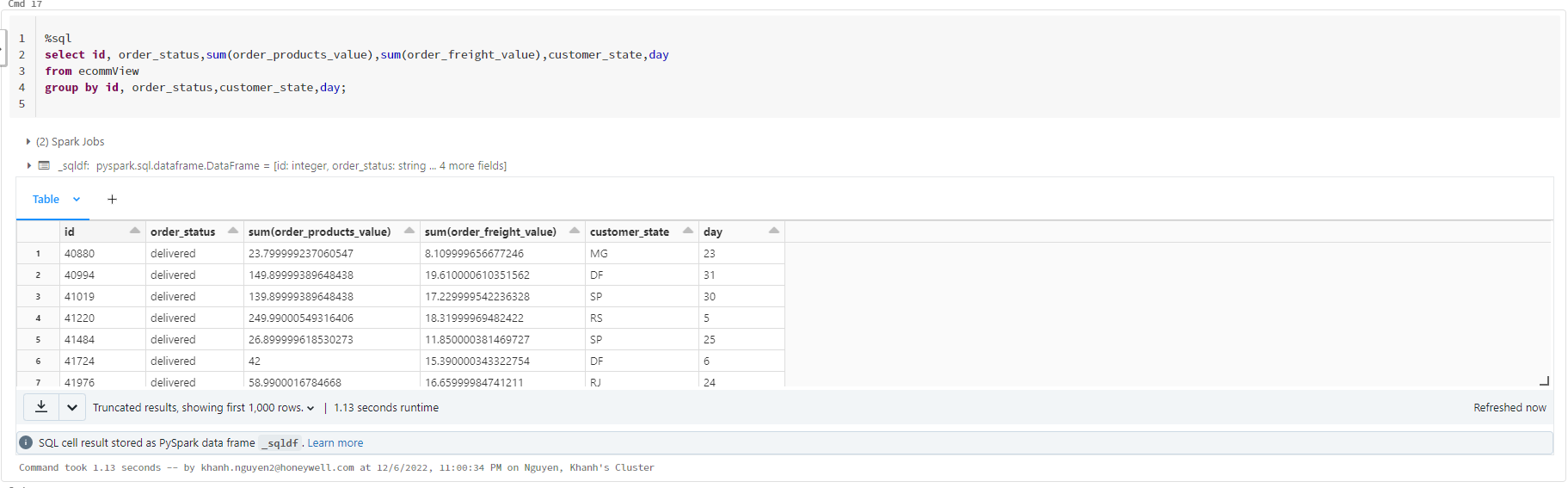
1. Total sales and order distribution per day and week for each state

%sql

select id, order\_status,sum(order\_products\_value),sum(order\_freight\_value),customer\_state,day

from ecommView

group by id, order\_status,customer\_state,day;

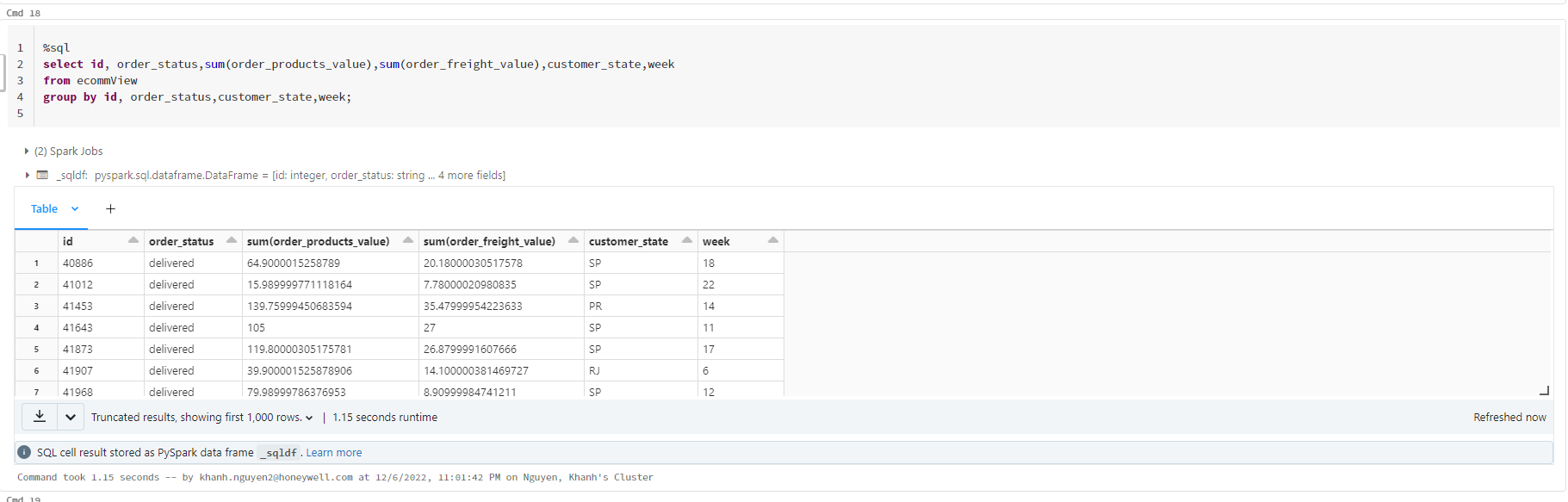


%sql

select id, order\_status,sum(order\_products\_value),sum(order\_freight\_value),customer\_state,week

from ecommView

group by id, order\_status,customer\_state,week;



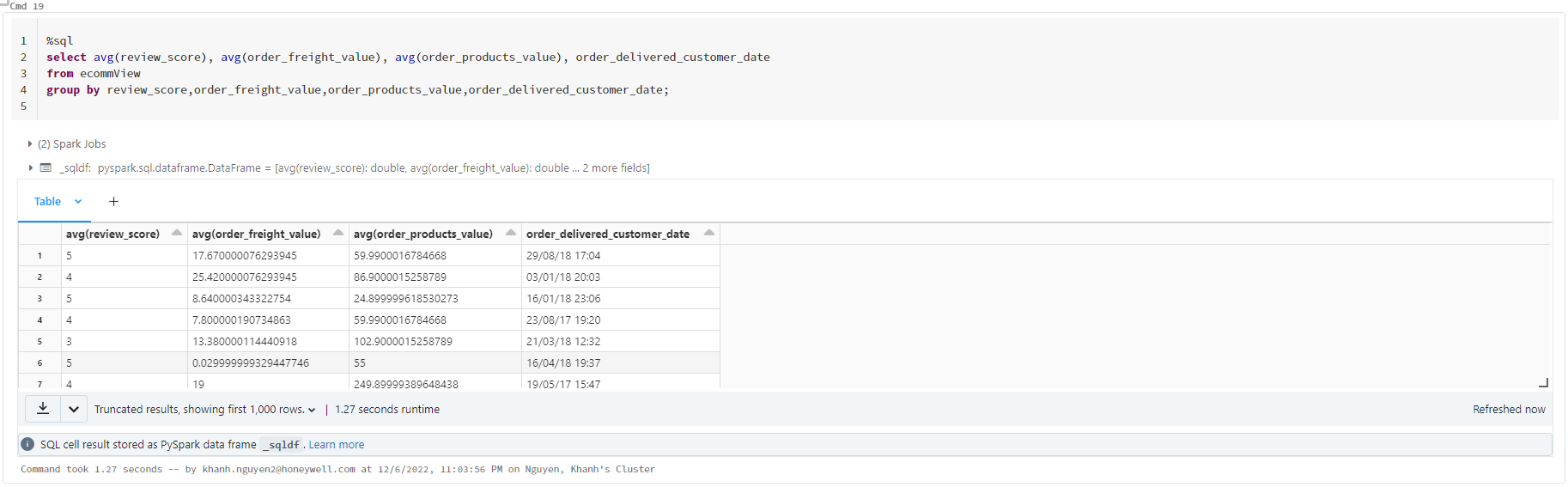
1. Average review score, average freight value, average order approval, and delivery time

%sql

select avg(review\_score), avg(order\_freight\_value), avg(order\_products\_value), order\_delivered\_customer\_date

from ecommView

group by review\_score,order\_freight\_value,order\_products\_value,order\_delivered\_customer\_date;



1. The freight charges per city and total freight charges

%sql

select order\_freight\_value,customer\_city from ecommView group by order\_freight\_value,customer\_city;



%sql

select sum(order\_freight\_value) from ecommView;



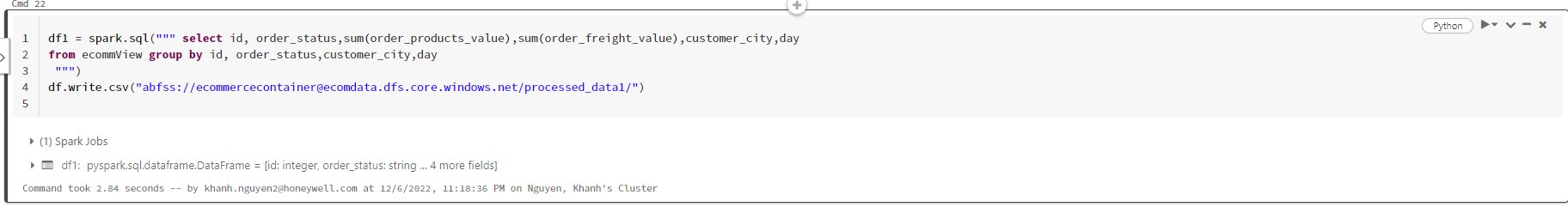
**Week 4: Data analysis and visualization**

**Step 1:**Write the results into HDFS

df1 = spark.sql(""" select id, order\_status,sum(order\_products\_value),sum(order\_freight\_value),customer\_city,day

from ecommView group by id, order\_status,customer\_city,day

""")

df.write.csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/processed\_data1/")

df1 = spark.sql(""" select id, order\_status,sum(order\_products\_value),sum(order\_freight\_value),customer\_city,week

from ecommview group by id, order\_status,customer\_city,week

""")

df.write.mode("overwrite").csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/processed\_data2/")

df1 = spark.sql(""" select id, order\_status,sum(order\_products\_value),sum(order\_freight\_value),customer\_state,day

from ecommview group by id, order\_status,customer\_state,day

""")

df.write.mode("overwrite").csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/processed\_data3/")

df1 = spark.sql(""" select id, order\_status,sum(order\_products\_value),sum(order\_freight\_value),customer\_state,week

from ecommview group by id, order\_status,customer\_state,week

""")

df.write.mode("overwrite").csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/processed\_data4/")

df1 = spark.sql(""" select avg(review\_score), avg(order\_freight\_value), avg(order\_products\_value), order\_delivered\_customer\_date from ecommview group by review\_score,order\_freight\_value,order\_products\_value,order\_delivered\_customer\_date;

""")

df.write.mode("overwrite").csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/processed\_data5/")

df1 = spark.sql(""" select order\_freight\_value,customer\_city from ecommview group by order\_freight\_value,customer\_city;

""")

df.write.mode("overwrite").csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/processed\_data6/")

df1 = spark.sql(""" select sum(order\_freight\_value) from ecommview;

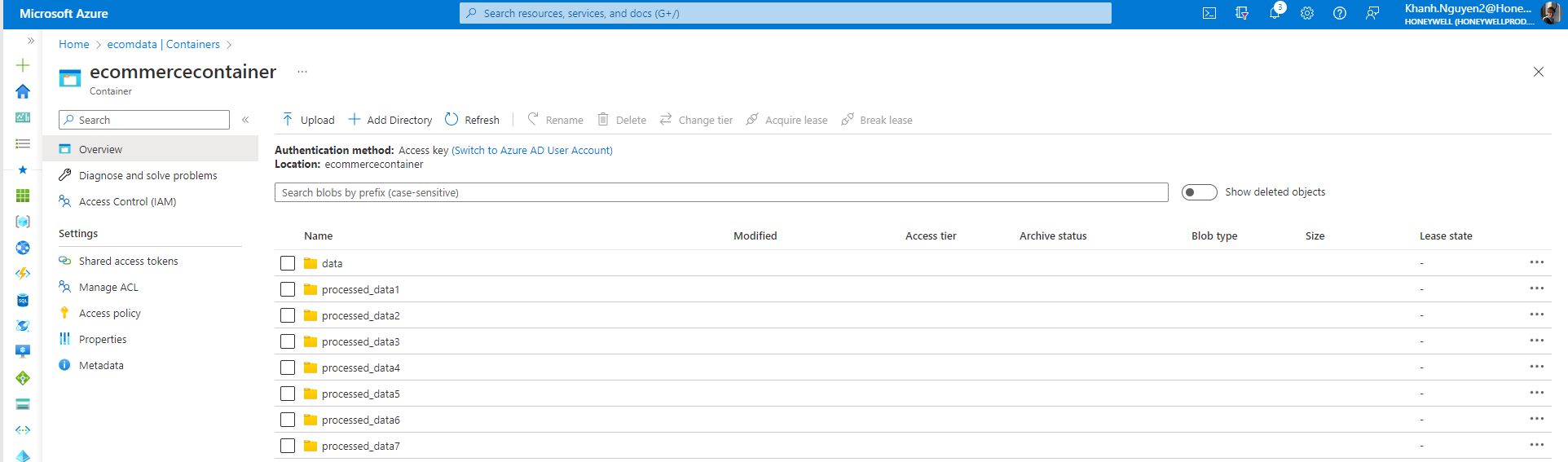
""")

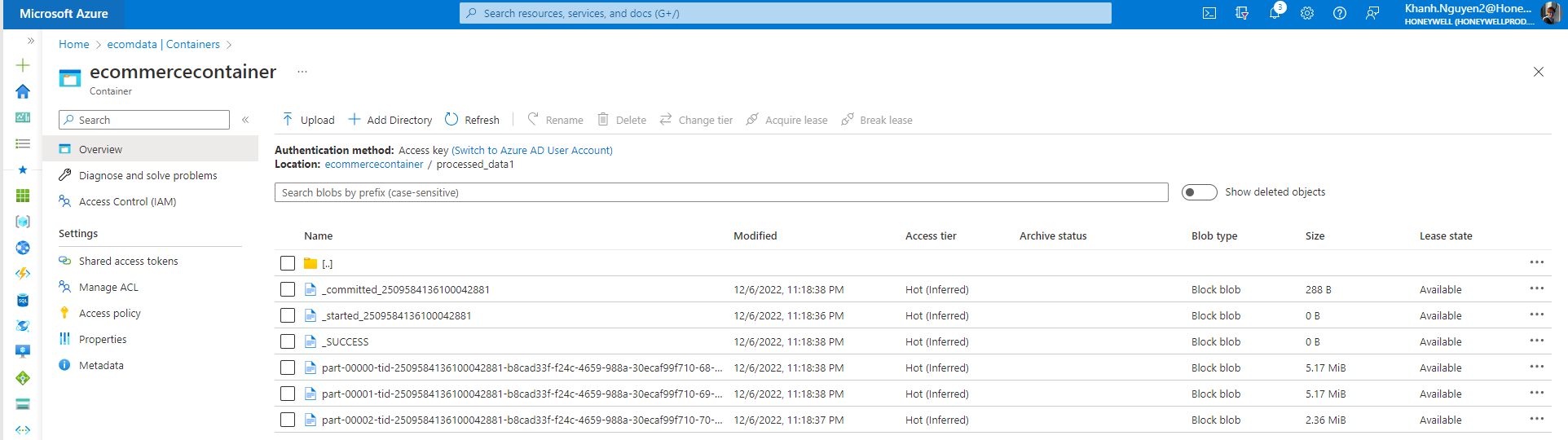
df.write.mode("overwrite").csv("abfss://ecommercecontainer@ecomdata.dfs.core.windows.net/processed\_data7/")

Example of a chart



**Step 2:** Save the final dataset into object storage service per the cloud platform

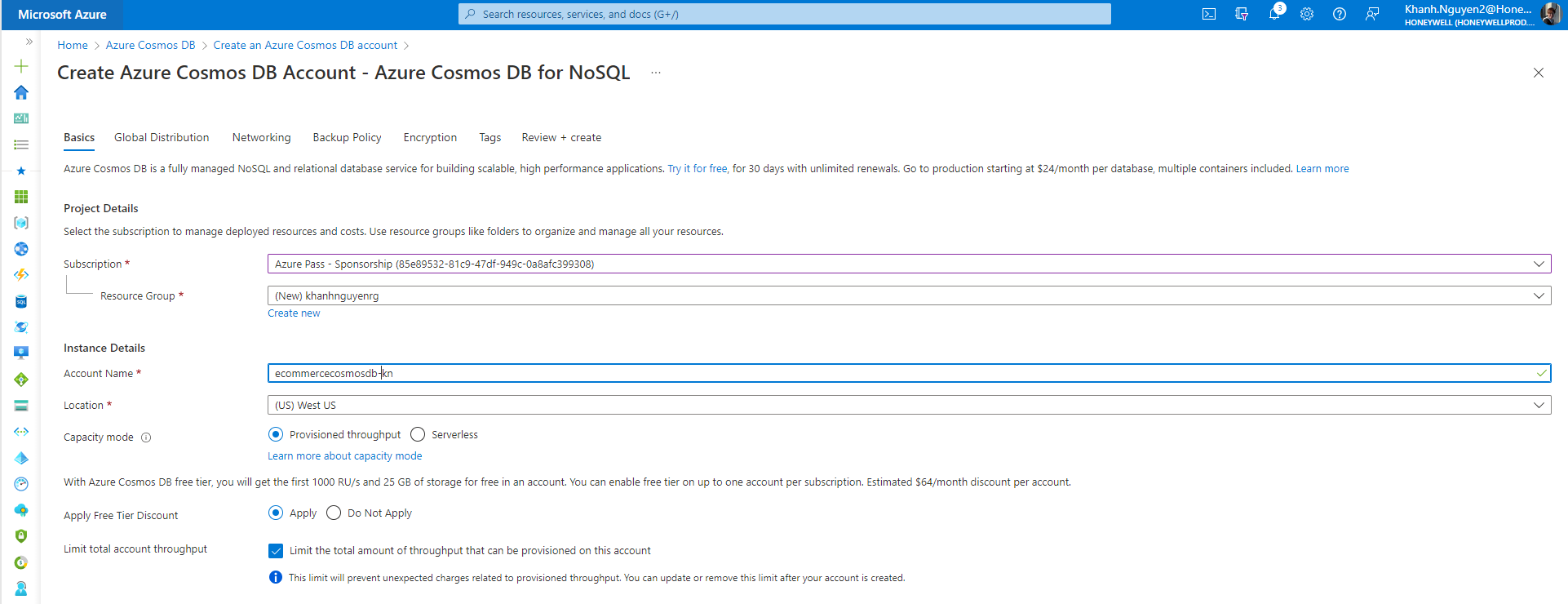


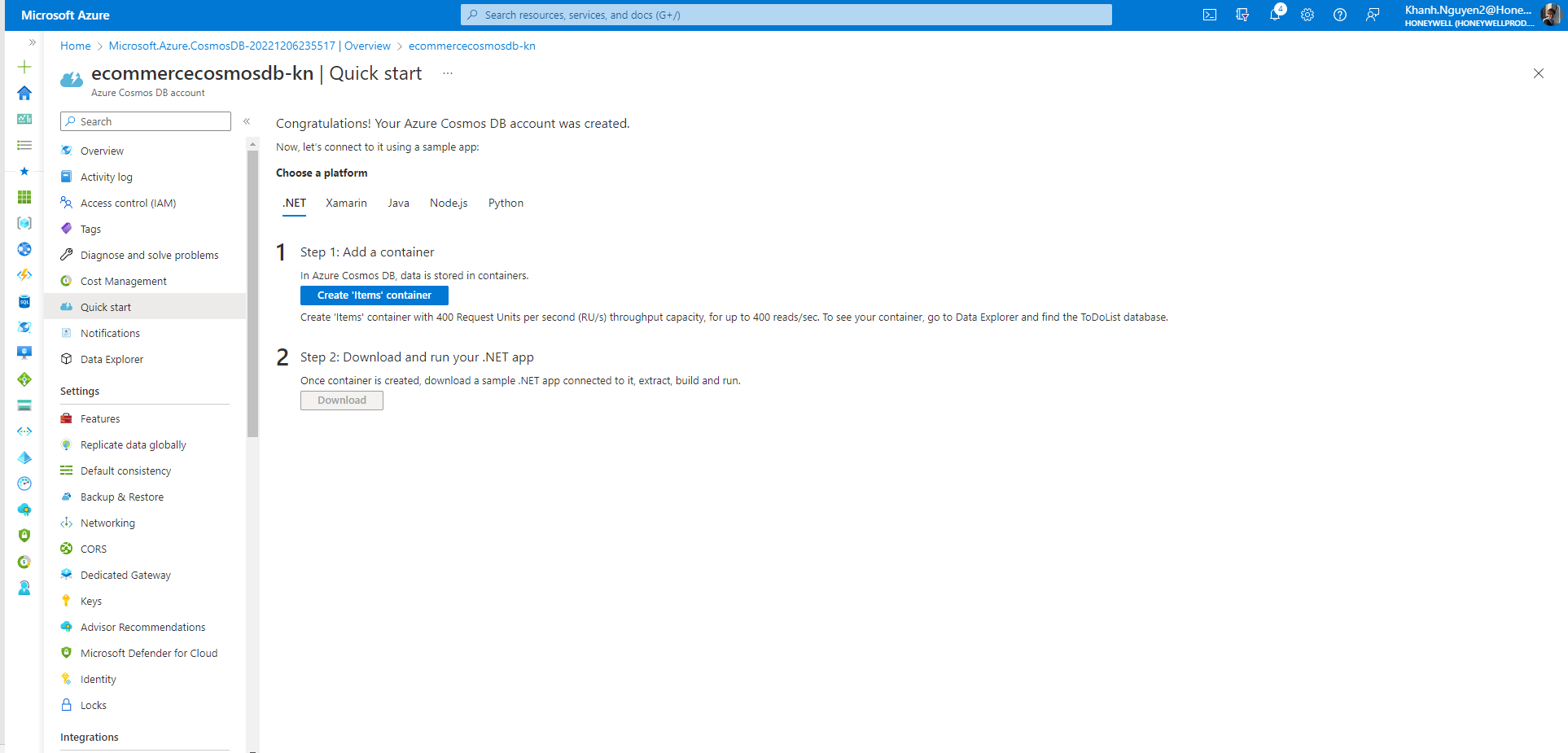


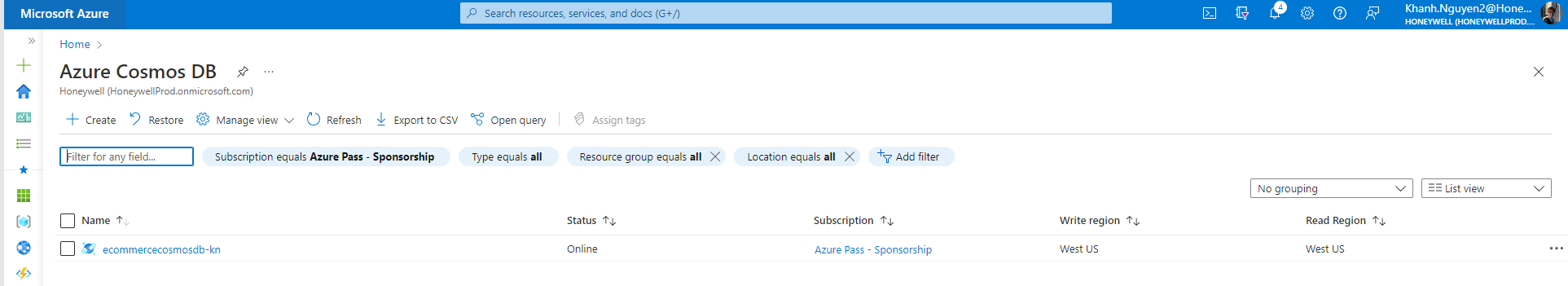
**Step 3:** Create a DB cluster that is also a NoSQL using the relevant service on the

             cloud platform

* Azure CosmosDB is created to store data insights







**Step 4:**Save insights in the NoSQL DB mentioned in the previous step

* Used the the Azure Cosmos DB Data Migration tool, which imported data Storage Account into Azure Cosmos containers and tables. Therefore, all of the steps in the project are covered and completed.