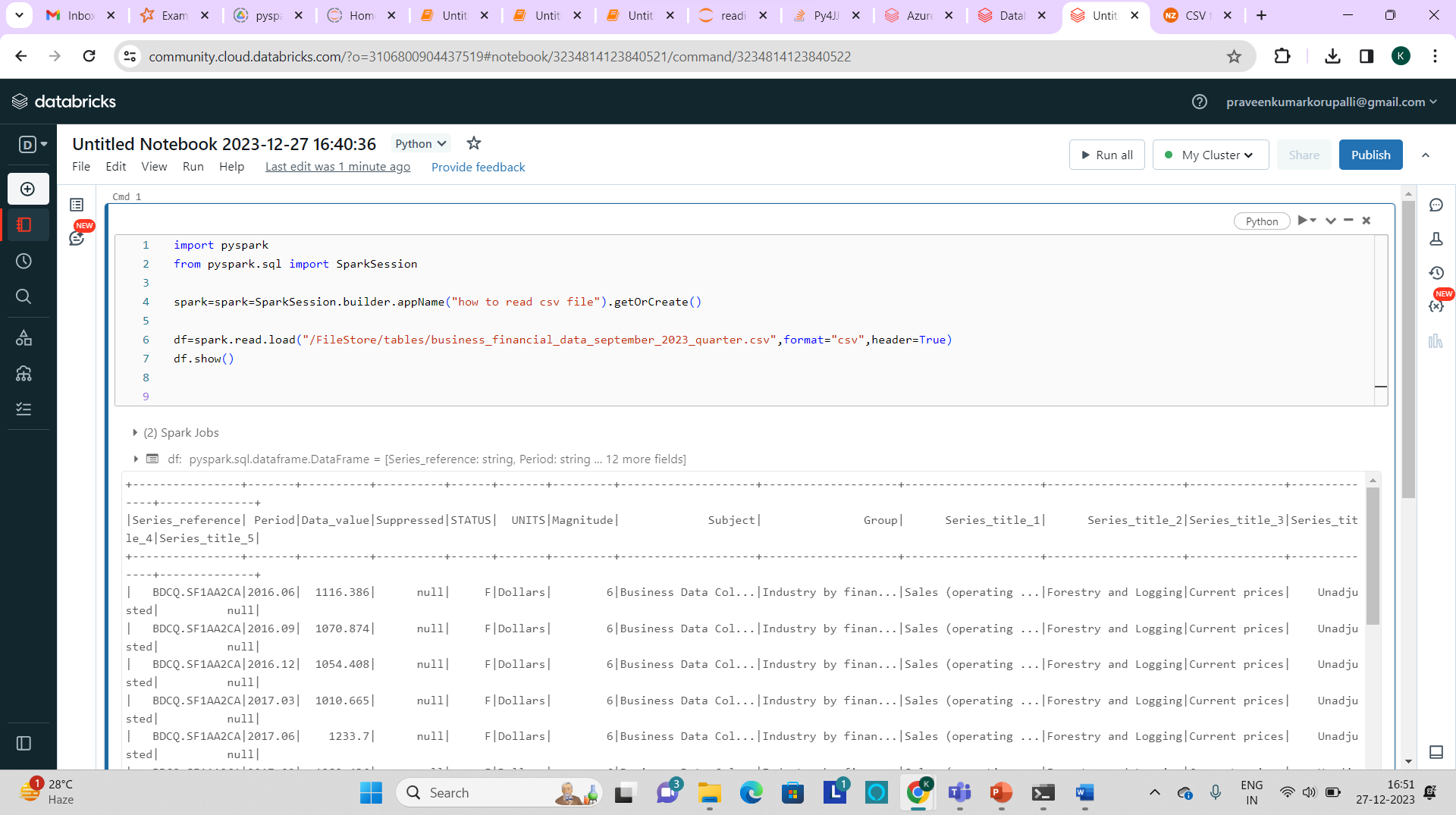
Coding Assessment

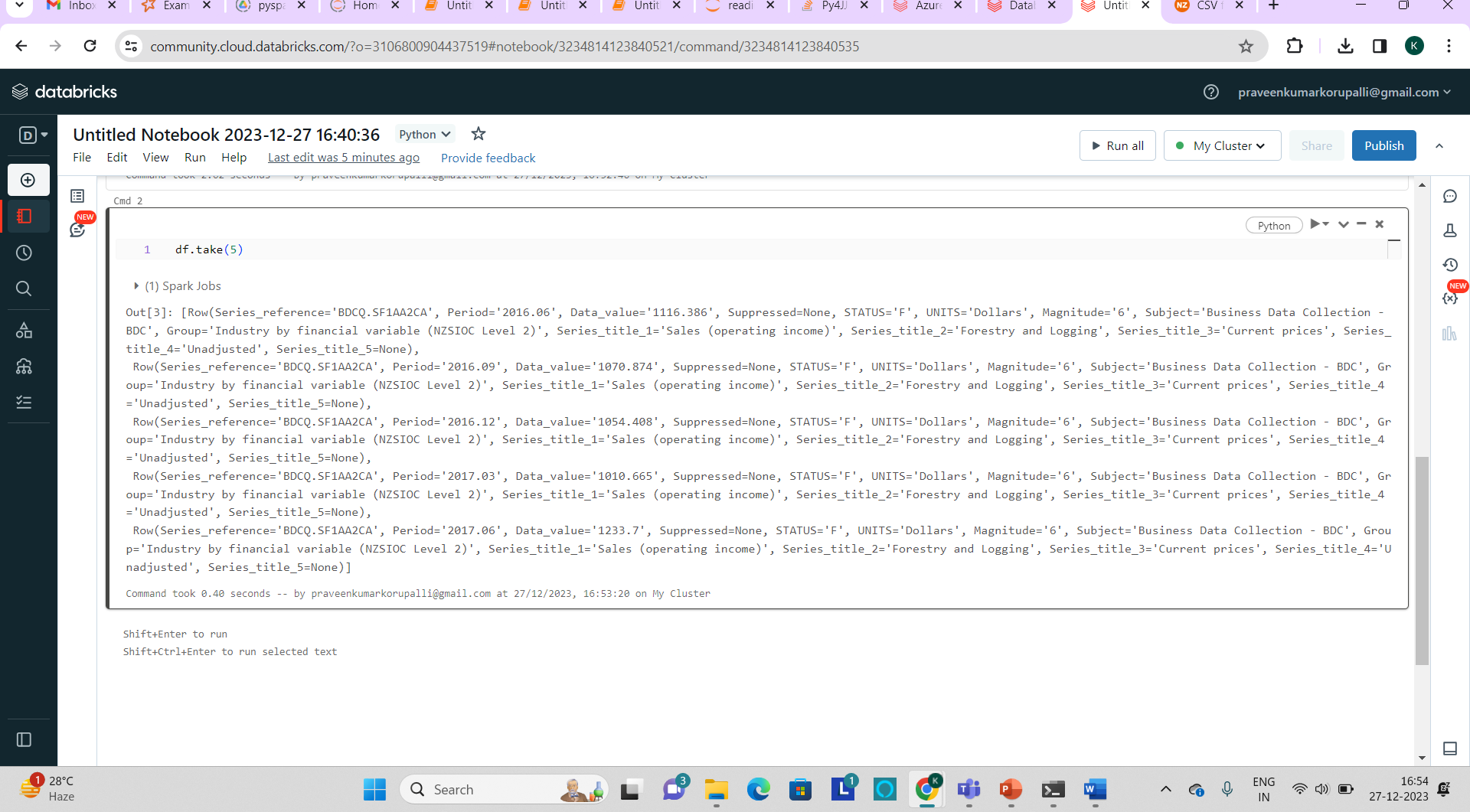
Q1. Implement Processing Json and csv data with pyspark.

To read a csv file we have to give a command like spark.read.load(“source\_path”,format=”csv”,sep=”,”,header=True)

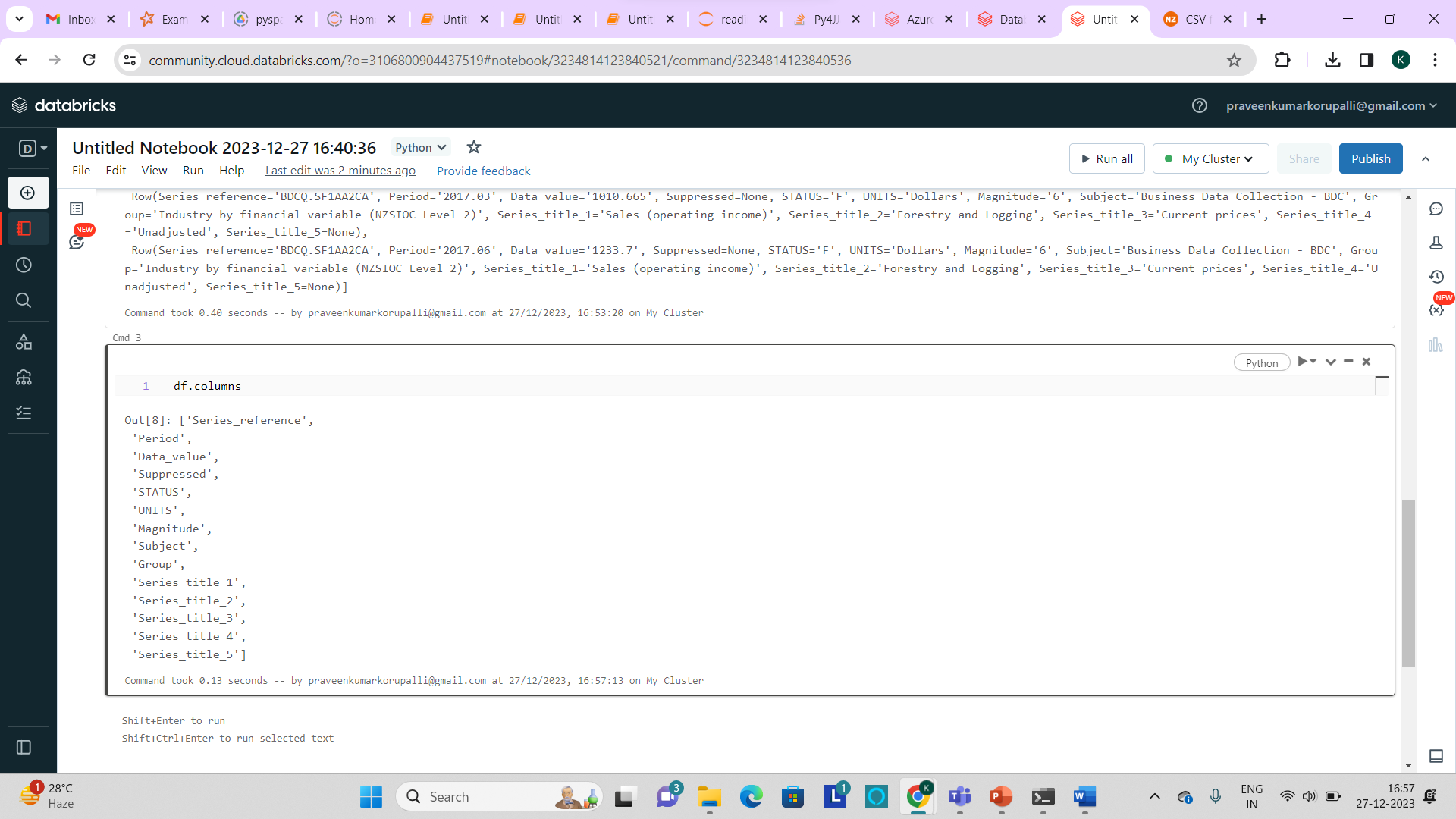


To display some rows we use “df.show()” command.

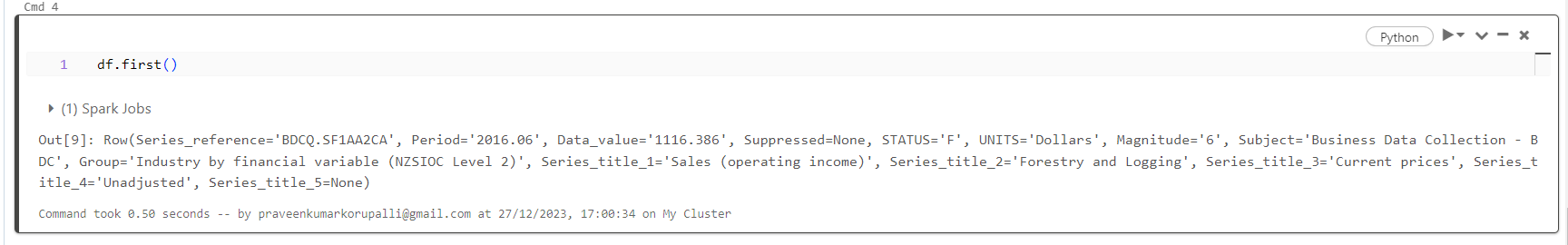
To take required data we use “df.take()” command.



df.columns is used to display the columns in the csv file.



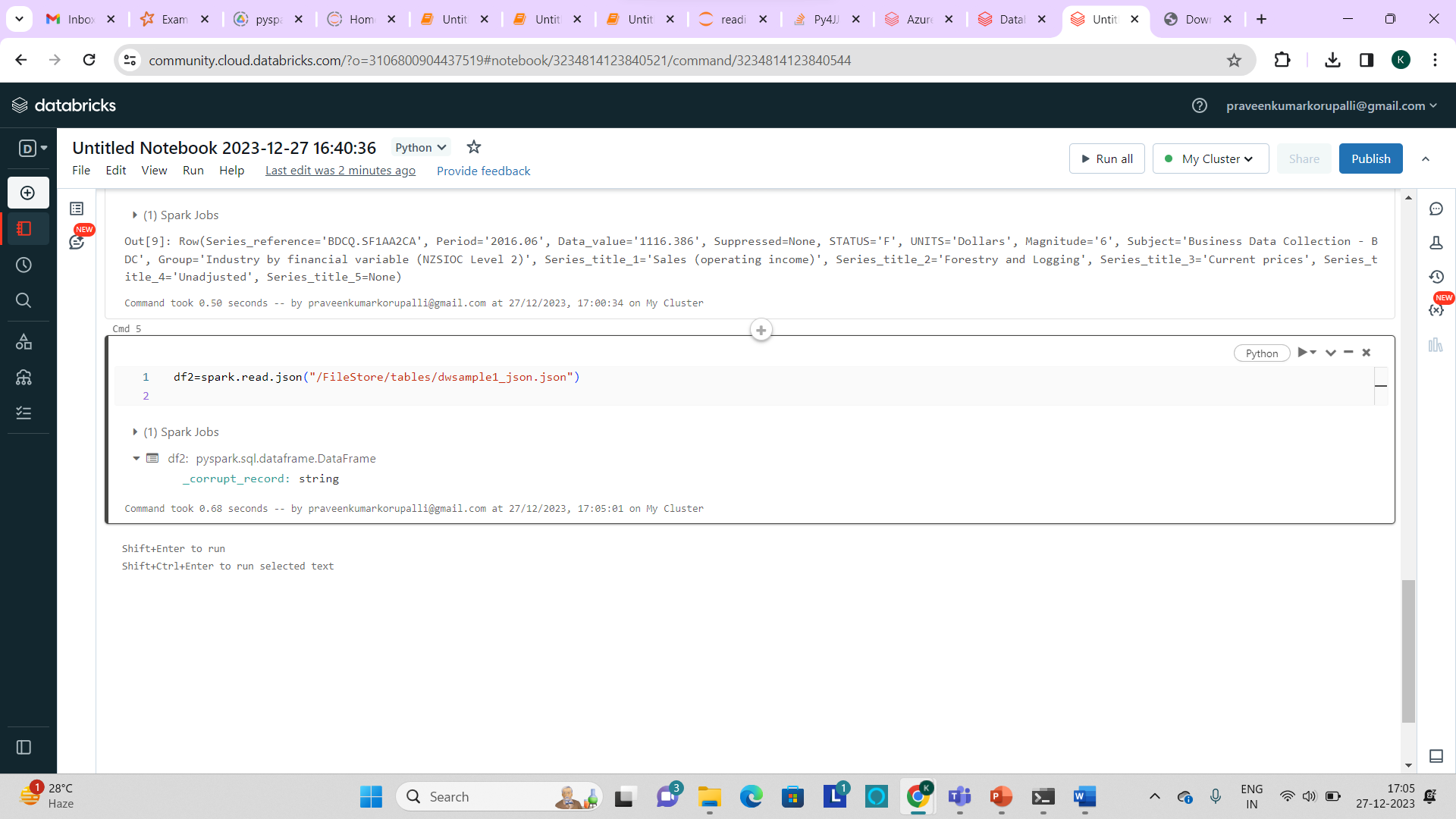
df.first() is used to take the first row from the csv file.



**Reading a Json file:**

To read a json file the syntax is:

df2=spark.read.json(“file\_path”)



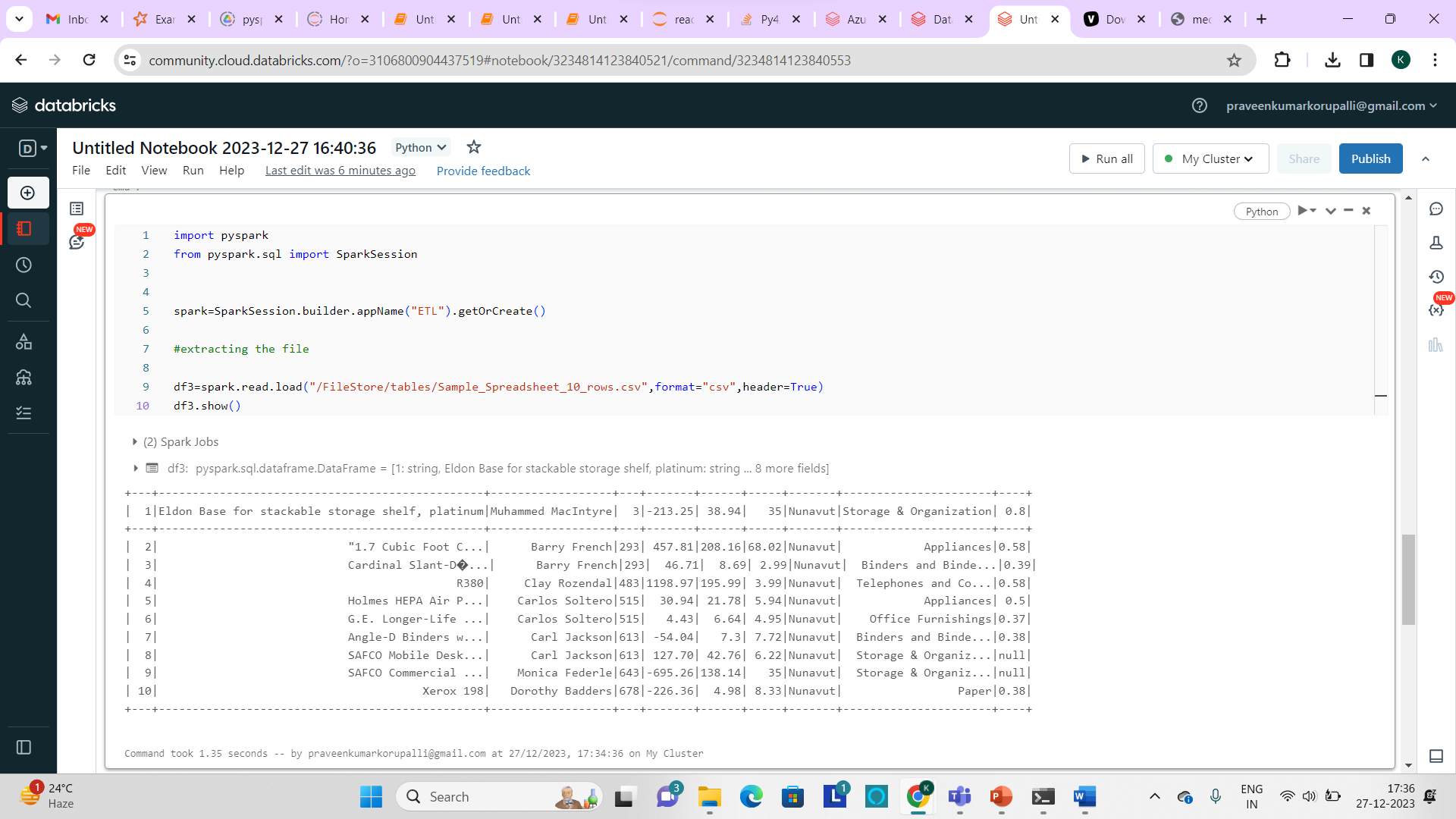
**Q2.Explain ETL (Extract, Transform, Load) with PySpark**

Extract: The files are extracted from different data sources.

Transform: The extracted files are transformed like adding the columns and renaming the columns and some sql operations are perfomed on the data file.

Load: The file is stored in the storage devices and in the local disks.

**1.Extracting the File:**

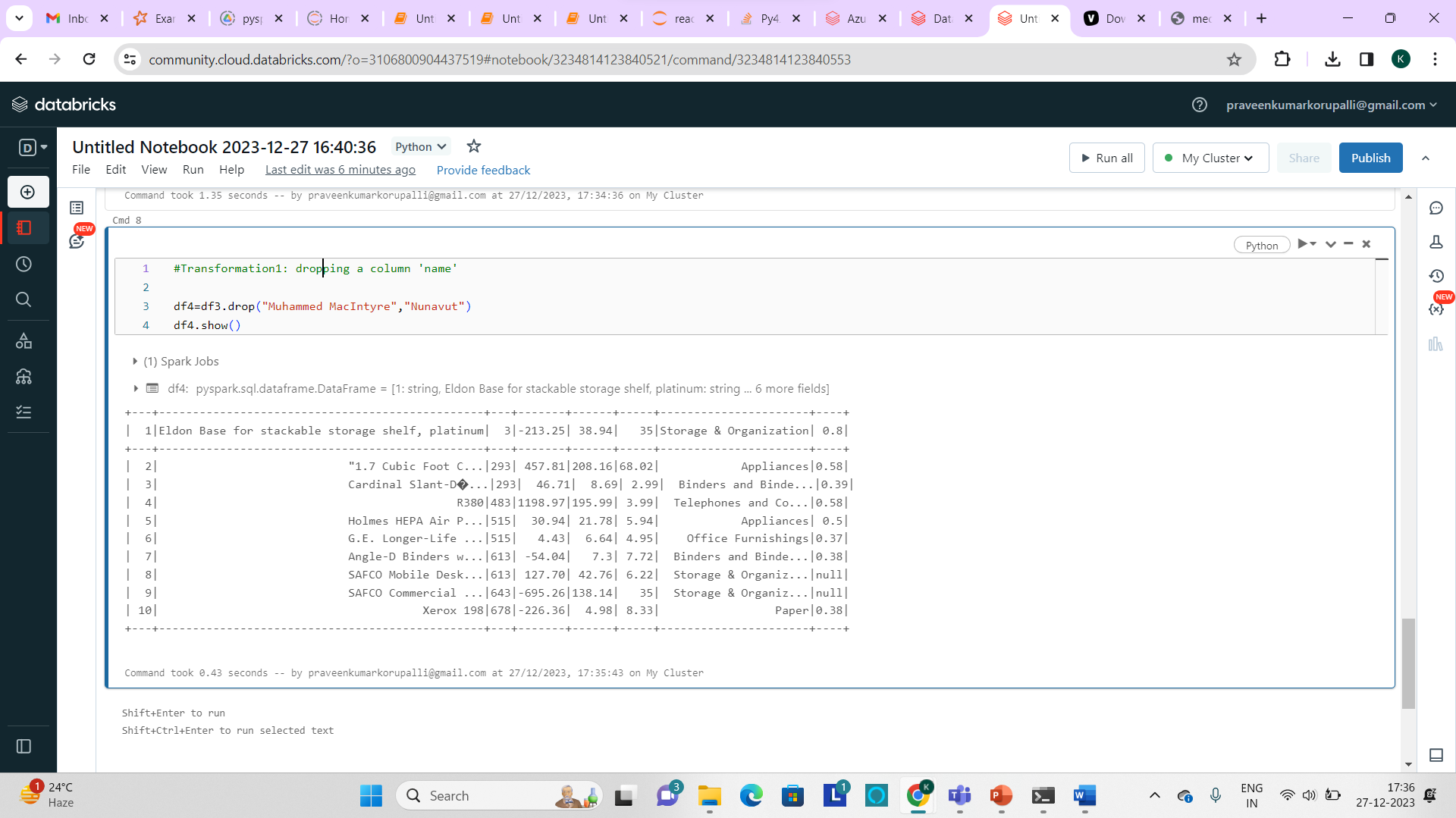


**2.Transformation:**

**Dropping a column:**

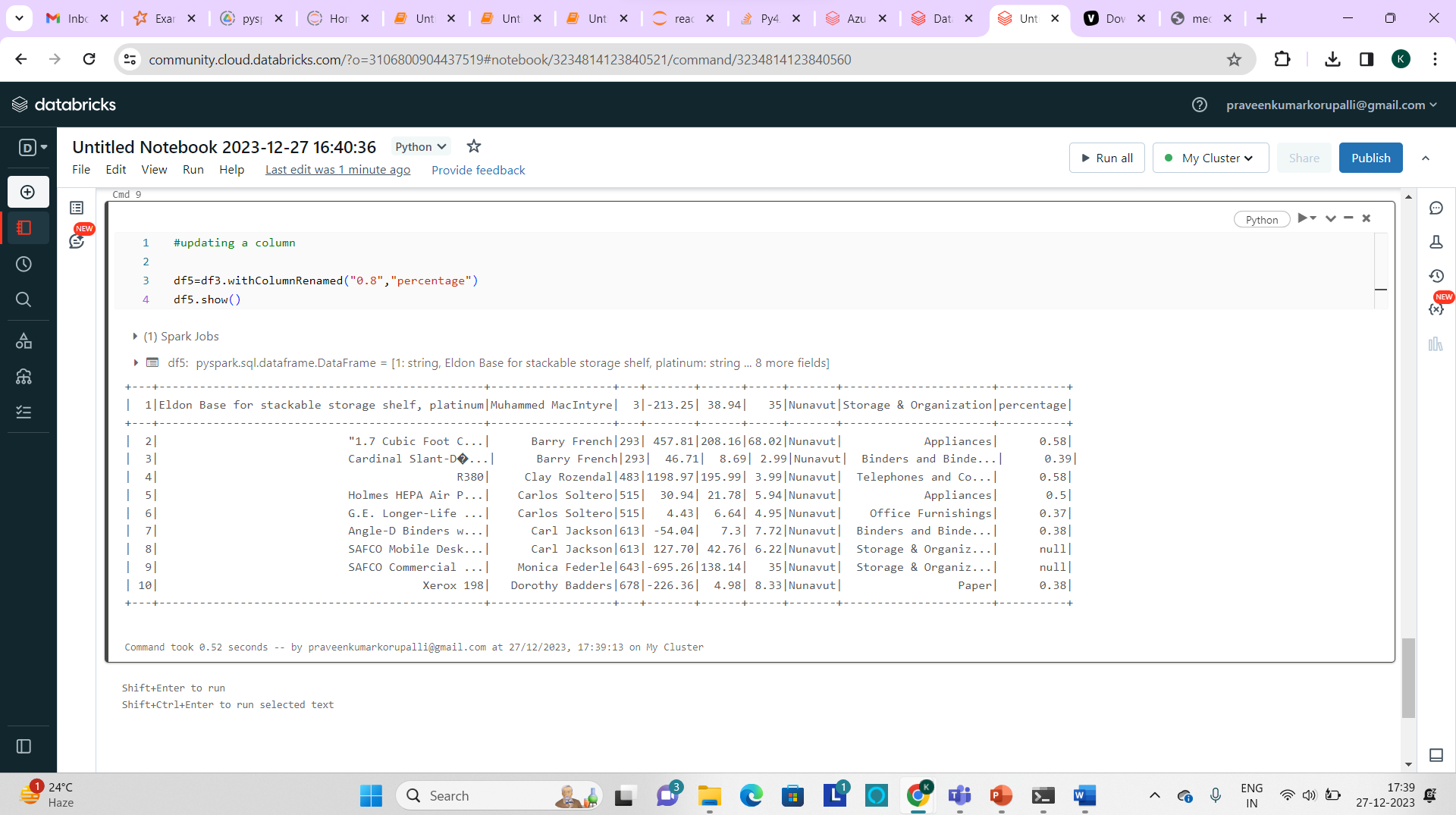
df4=df3.drop(“column\_names”)

df4.show()

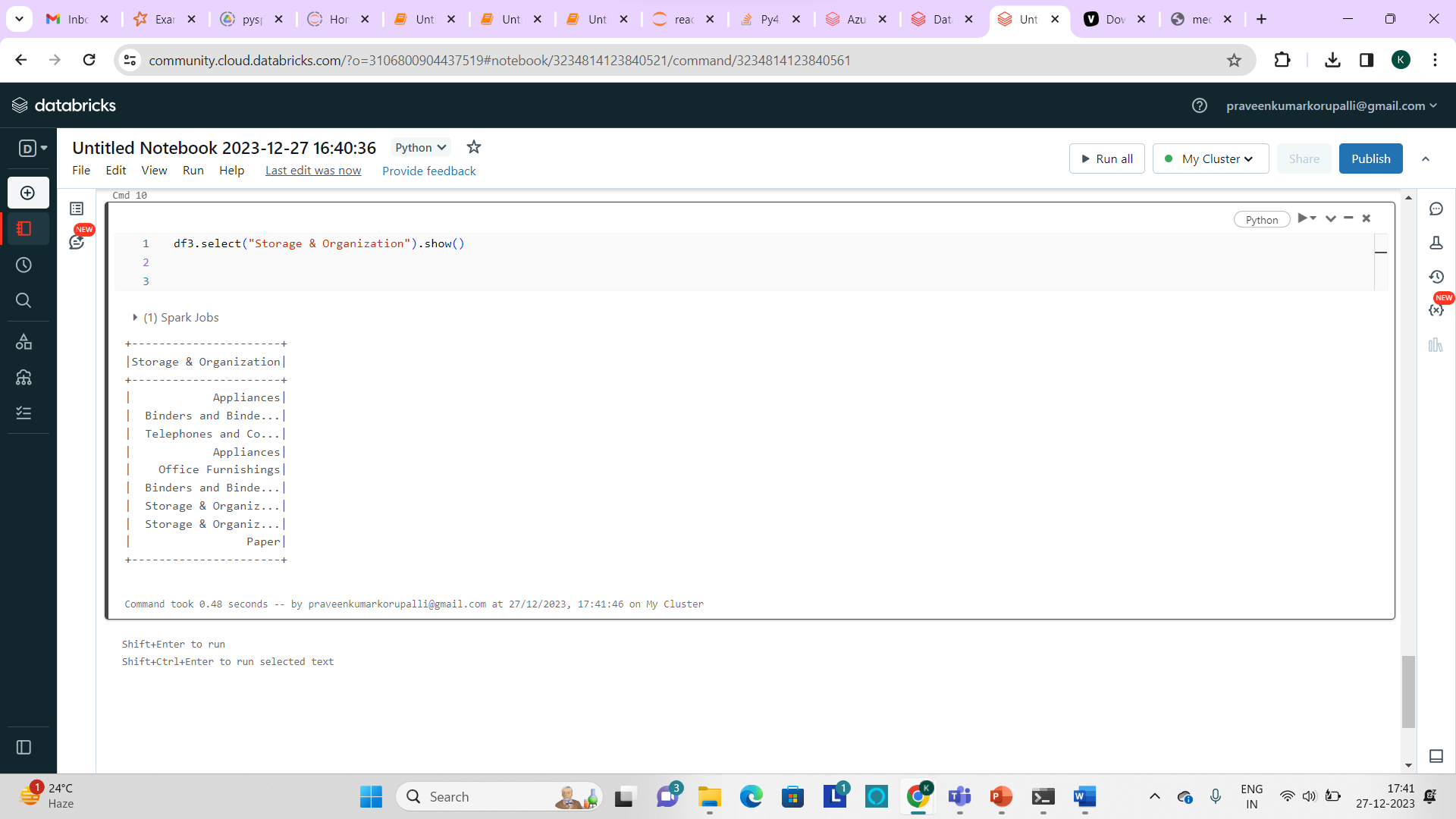


**Updating a column:**

Here the column “0.8” is renamed to “percentage”



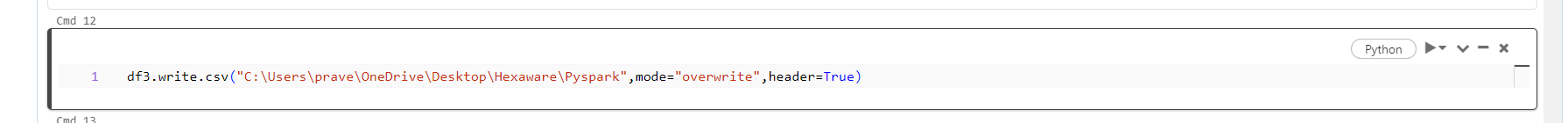
**Displaying selected columns:**



**Load:**

We can load the file into the specified directory by using the below syntax:

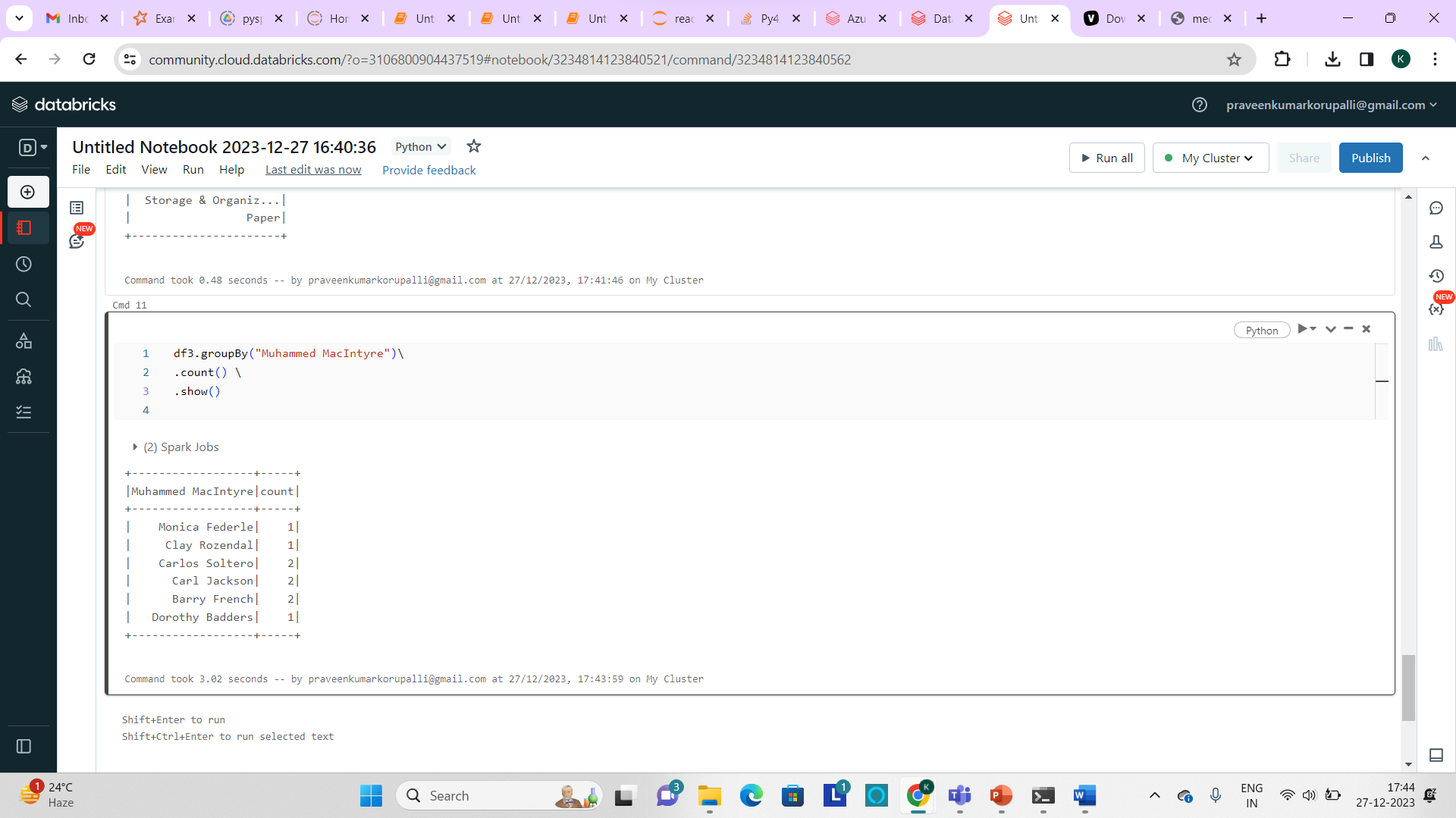
Df.write.csv(“target\_path”,mode=”overwrite”,header=”True”)



This will create a csv file in the specified path which you have been provided.

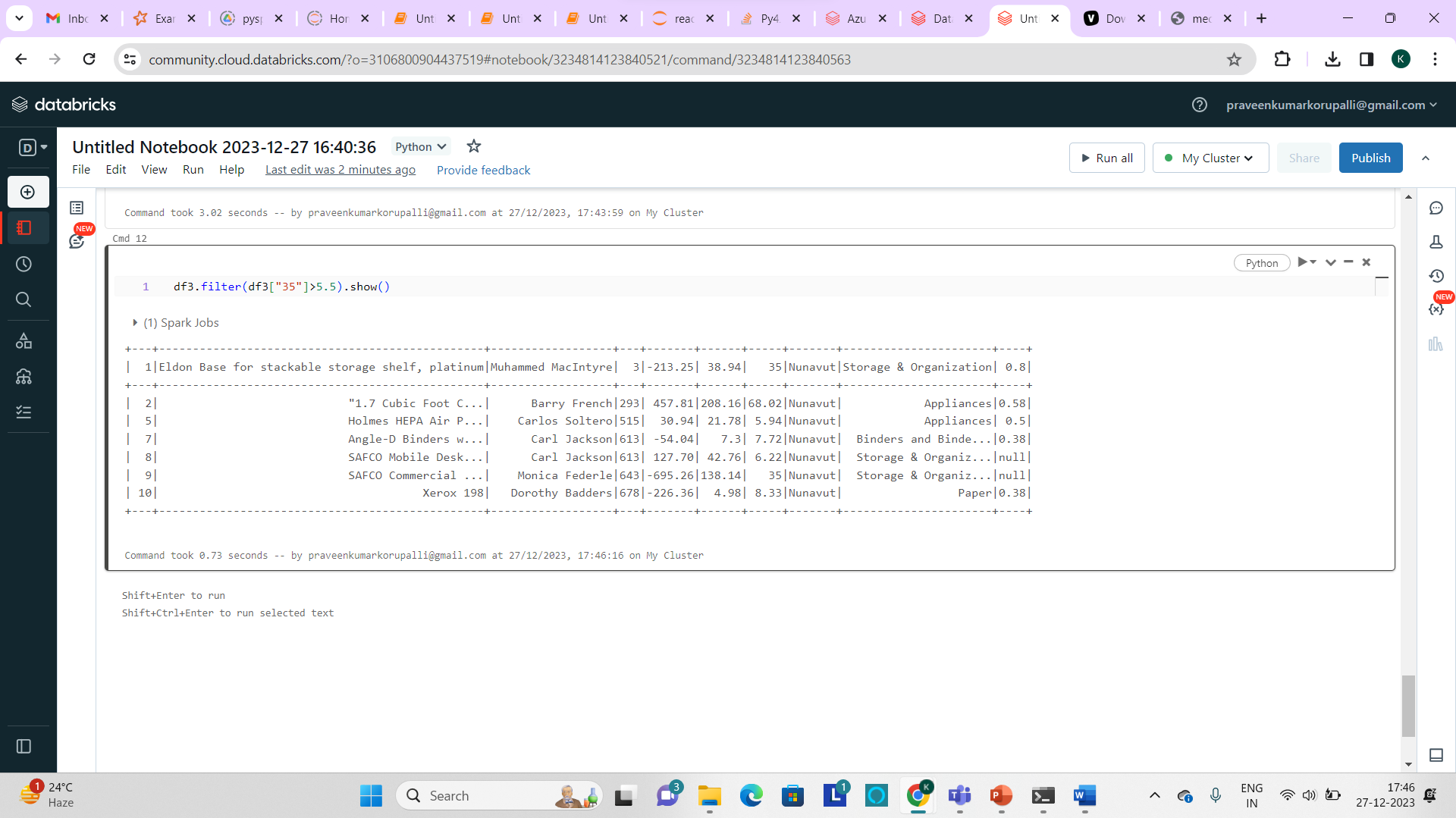
**Q3.Using Spark SQL - Transformations such as Filter, Join, Simple Aggregations, GroupBy.**

GroupBy:



**Filter:**

Used to filter and retrieve the required data by giving some specified condition.



**Aggregate Functions:** we can use the aggregate functions like minimum, maximum and count.

