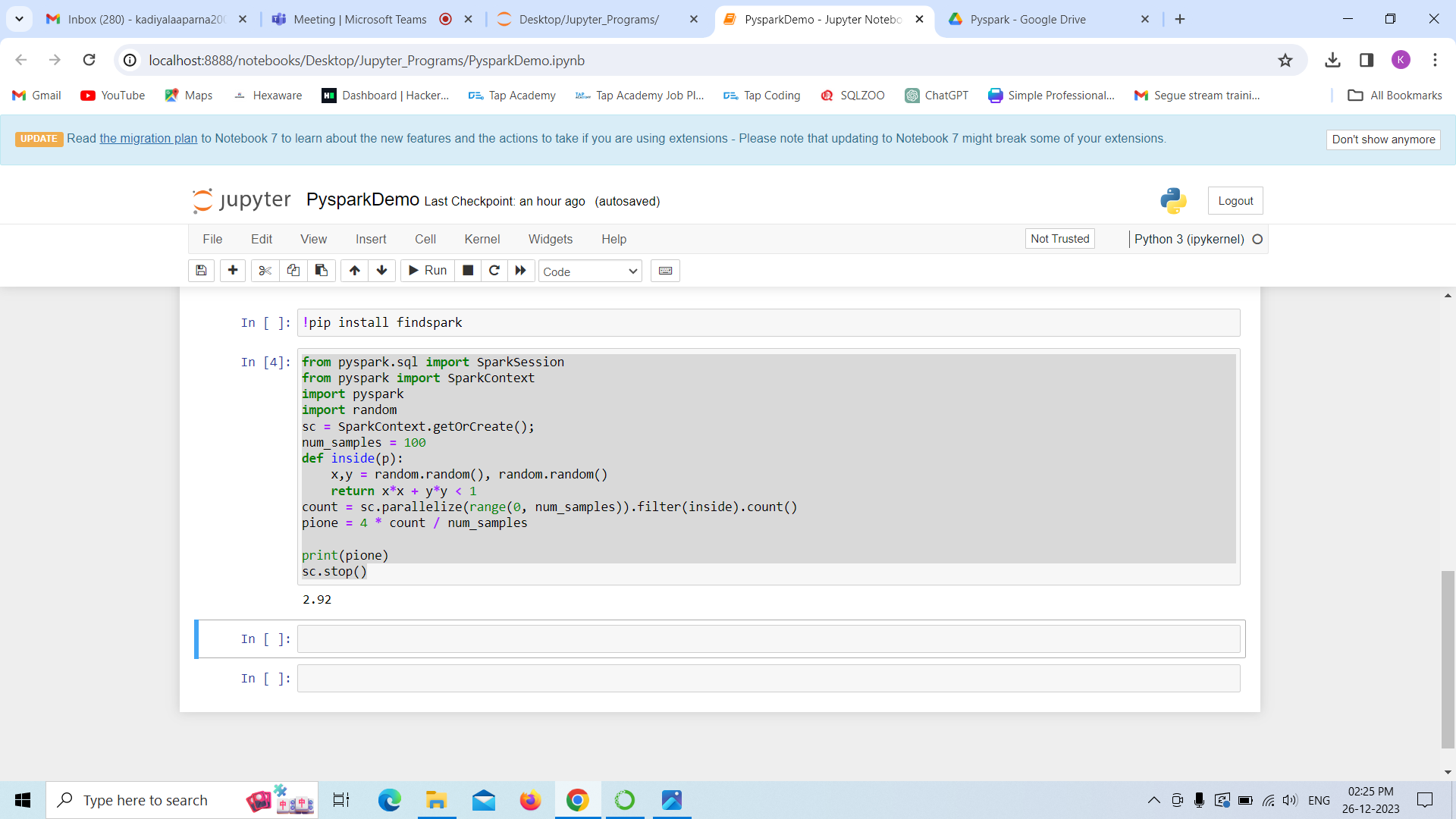
**Assignment(26-12-2023)**

**Installation of Anaconda.**

* Today we have seen how to download Anaconda and Setup the things.
* Download Anaconda from the Anaconda website by choosing the version that matches to our operating system and download the installer.
* The Anaconda installer will launch a setup. Click "Next" or "Continue" through the wizard.
* we will be asked to read and agree to the license terms. Accept the terms.
* Click "Install" to begin the installation process.
* The installer will copy files and set up Anaconda. This might take a few minutes.
* Once the installation is complete, you'll see a message indicating success. You can now close the installer.
* Open a new terminal or command prompt and type **conda --version** to verify that Anaconda has been installed successfully.

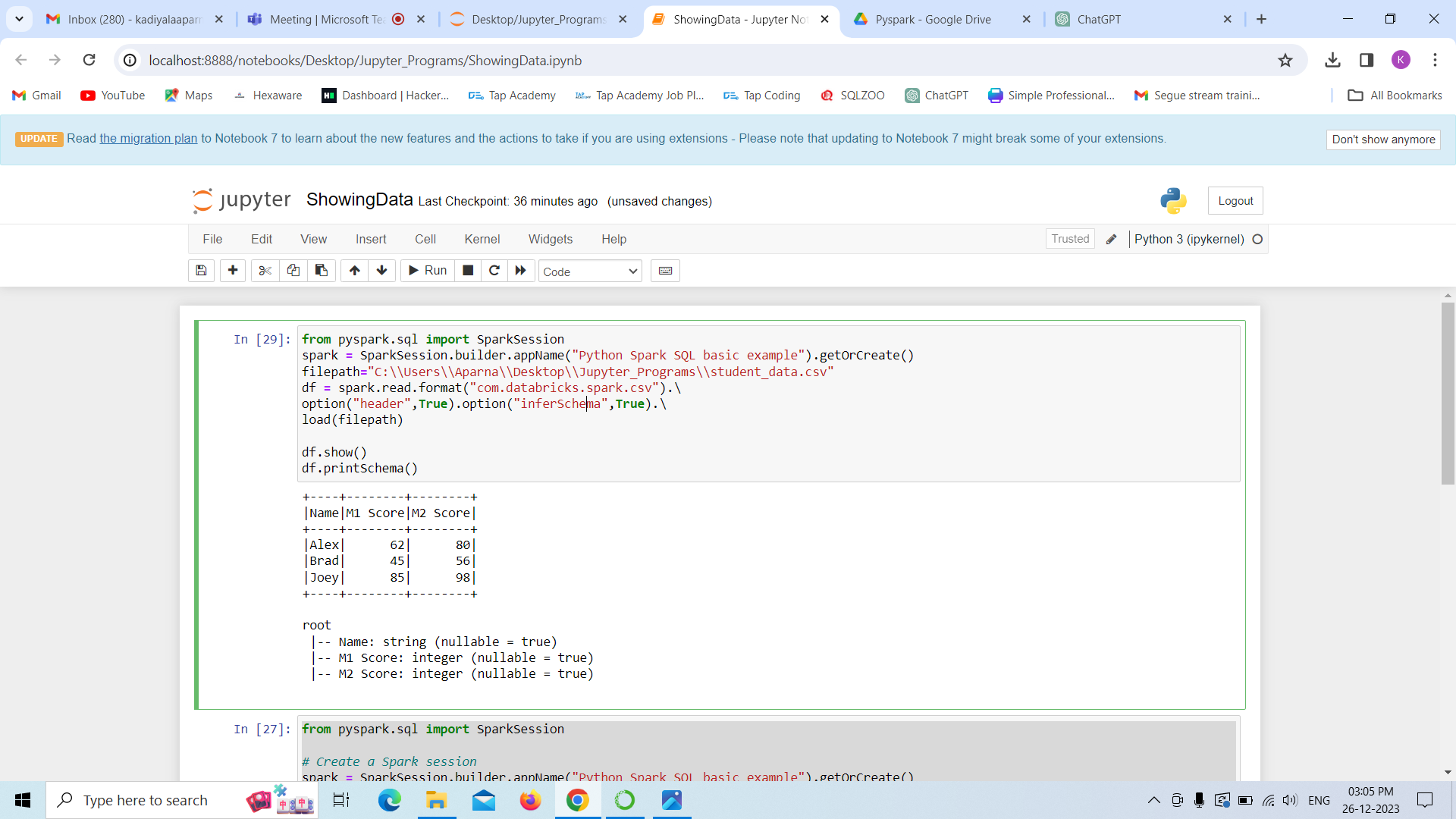
Running **sample Programs** in Jupiter with PySpark.

This program display random value with random() function in with SparkSession.



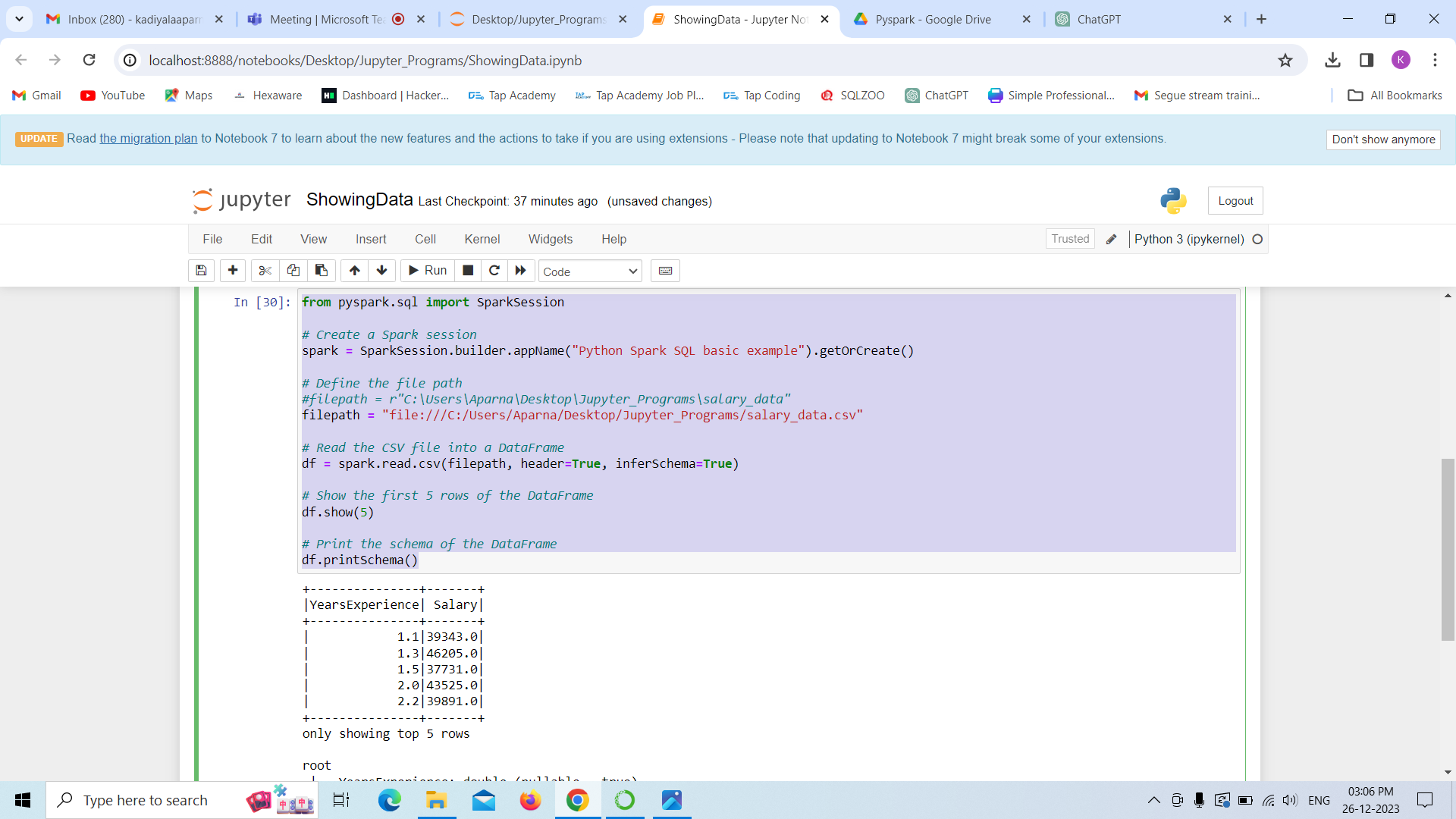
**Sample Program 2**

This program reads the data from CSV file and display it.



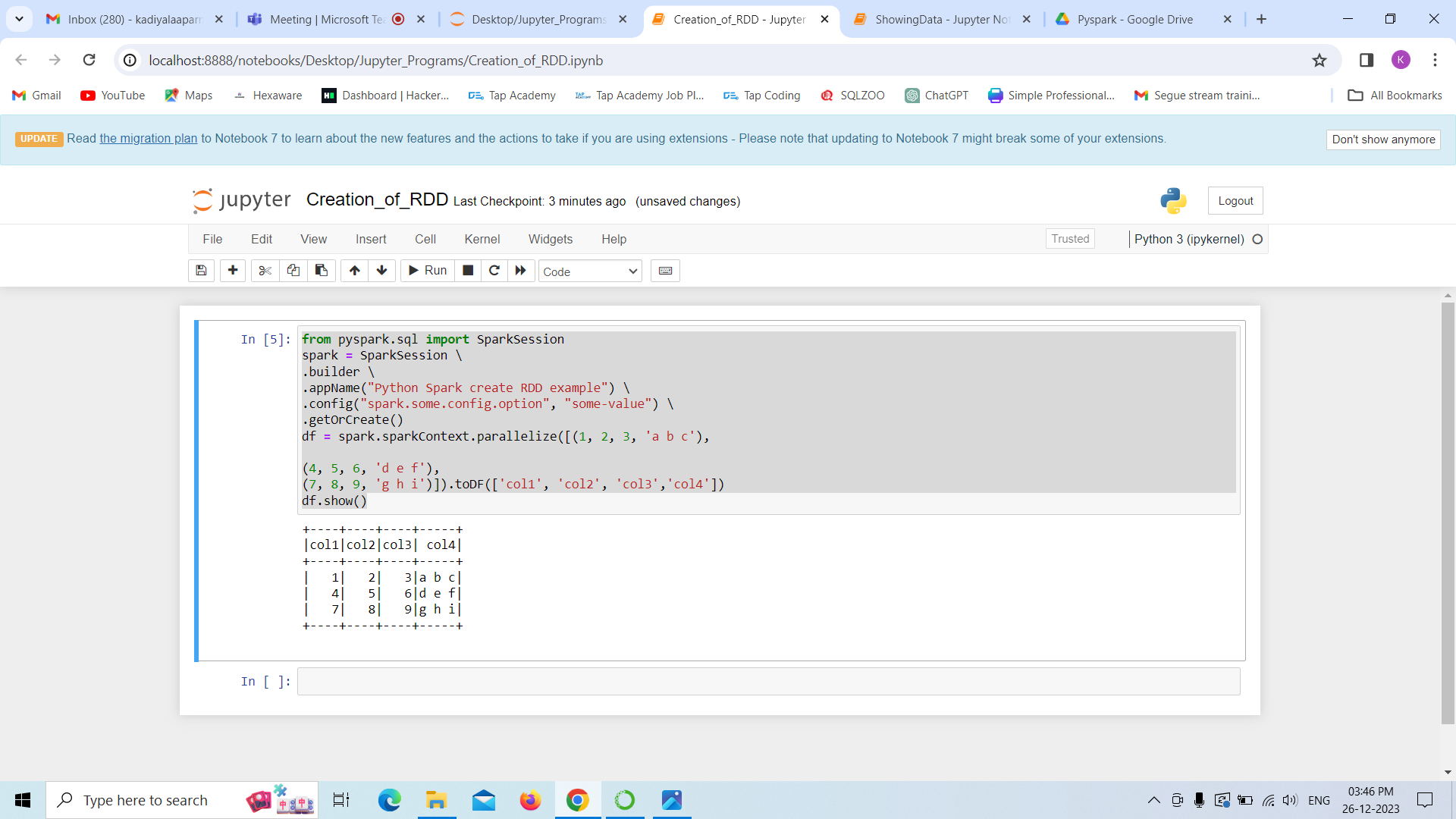
**Sample Program 3**

This program reads the data from CSV file and display it.



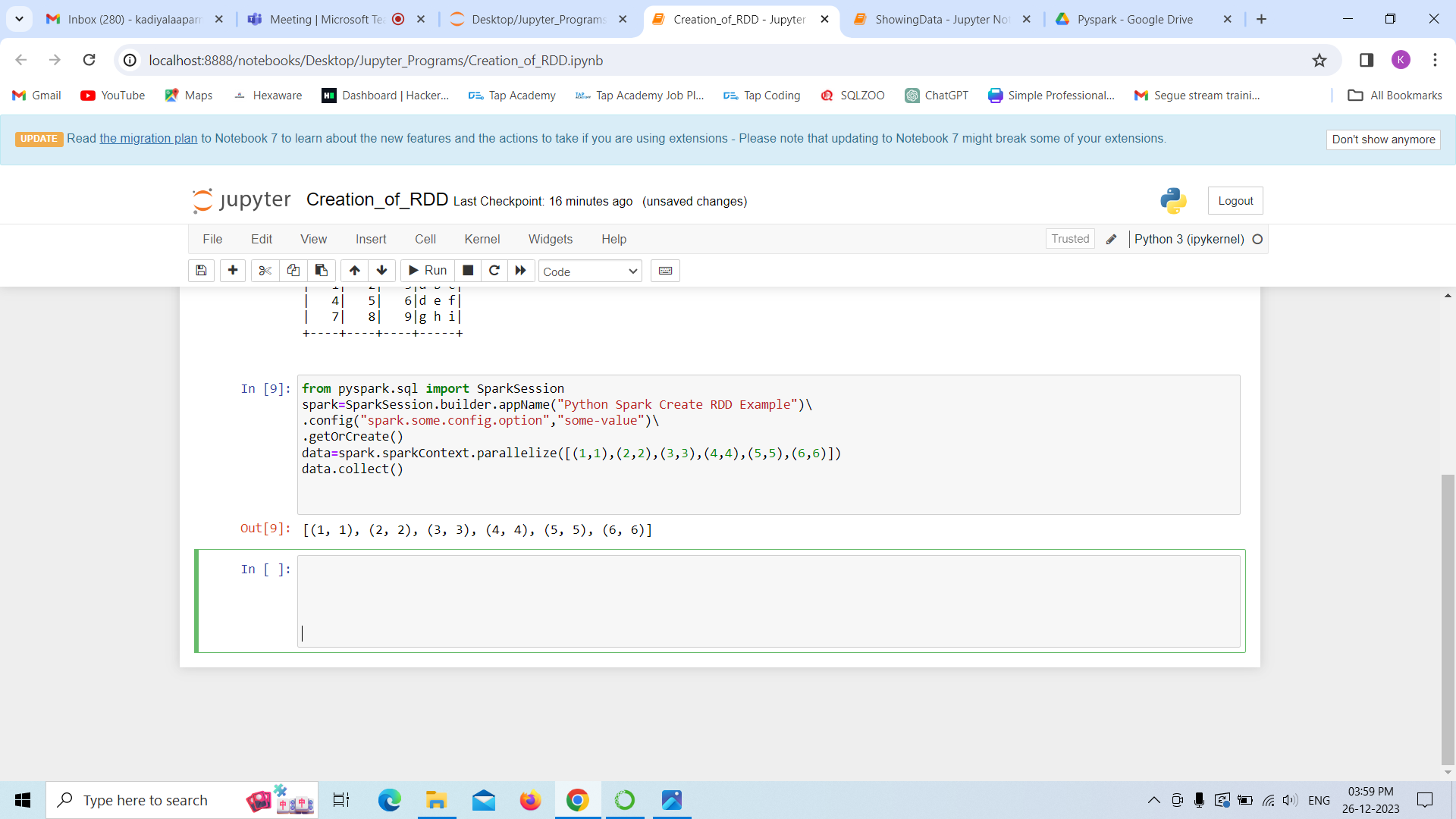
**Sample Program 4**

This program creating spark RDD with parallelize function along with that we have to add data and display it with help of show function



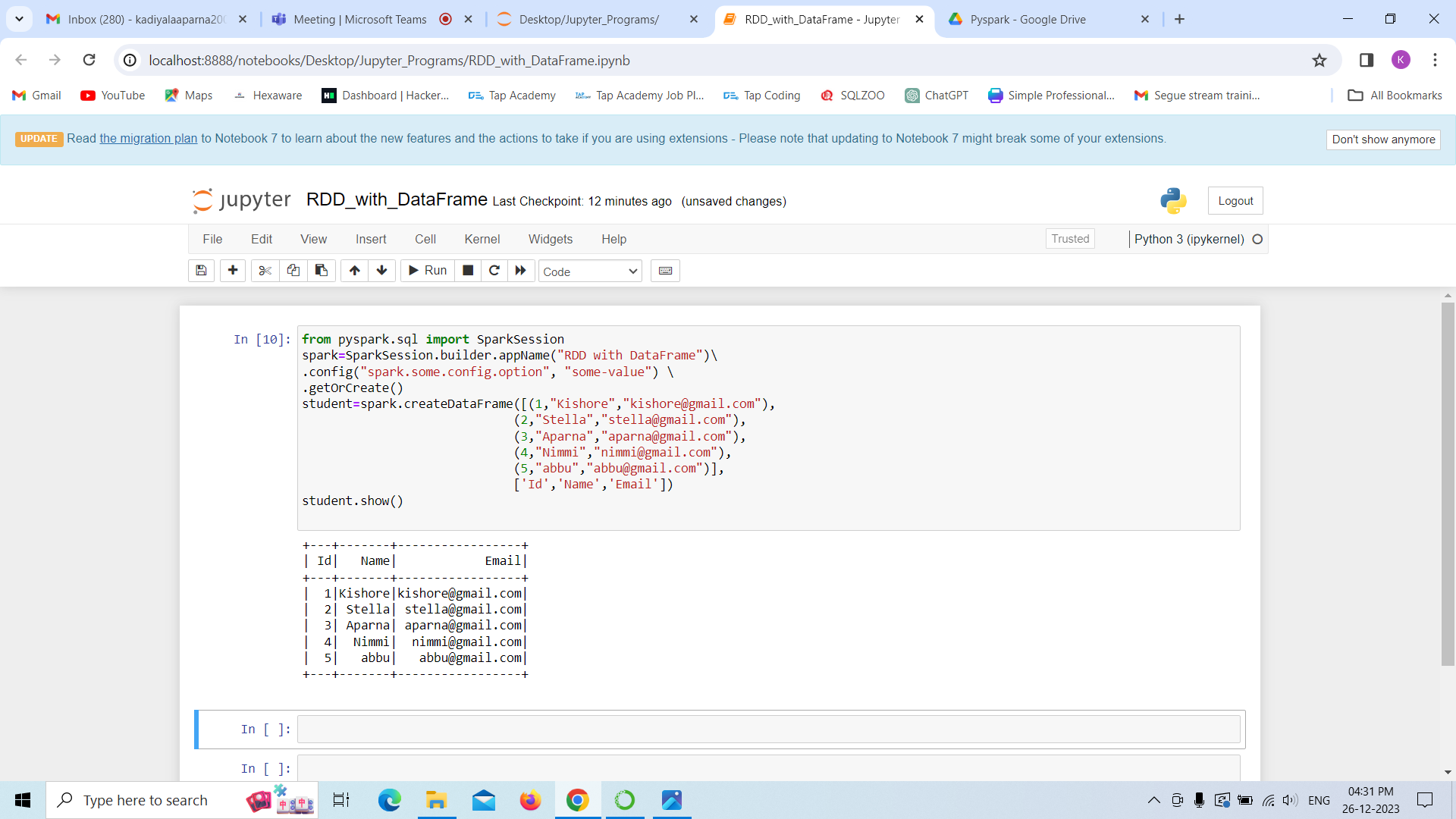
**Sample Program 5**

This program creating spark RDD with parallelize function along with that we have to add data and display it with help of show function



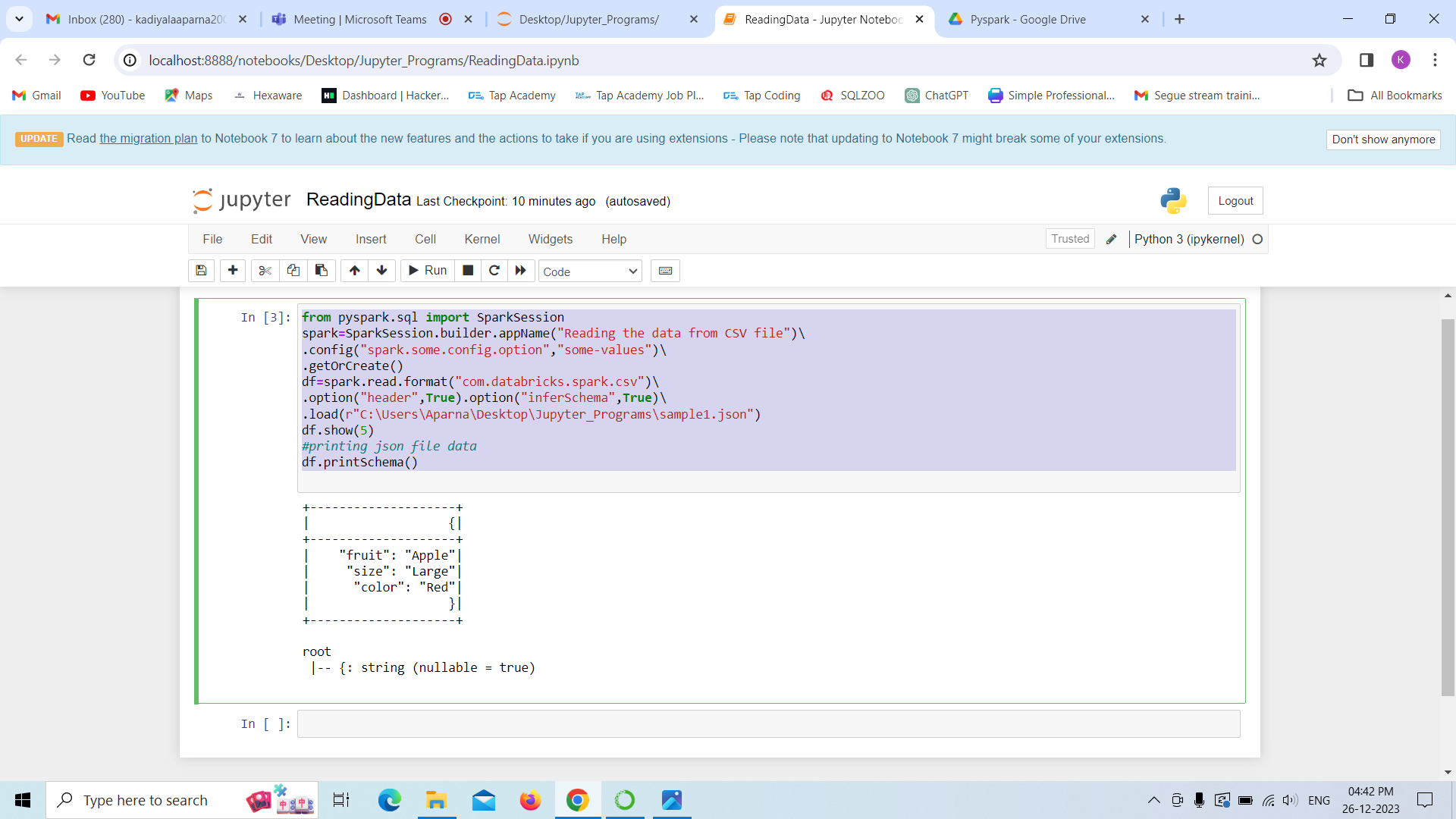
**Sample Program 6**

This program creating spark RDD with createDataFrame function along with that we have to add data and display it with help of show function



**Sample Program 7**

This program creating spark RDD with parallelize function along with that we have to read data from JSON file and display it with help of show function



**Introduction to ETL**

ETL stands for Extract, Transform , Load.

It's a process used in data integration and warehousing to move data from source systems to a target system, with necessary transformations applied during the process.

Let’s see exact meaning of these ETL terms.

**Extract:** Here ,we have to extract our required data from multiple sources like flat files, storage systems etc.

**Transform**:Here, we have to transform our data which involves manipulation like applying aggregating functions, using joins or like cleaning data all these things will involve over here.

**Load:** All these transformed data would be loading into the our required databases or datawarehouses.

Various ETL tools and frameworks, both open source and commercial, are available to streamline and automate the ETL process. Examples include Apache NiFi, Talend, and Microsoft SSIS (SQL Server Integration Services).

Sample Program

from pyspark.sql import SparkSession

from pyspark.sql.functions import col, concat, lit, floor, rand

# Initialize a Spark session

spark = SparkSession.builder.appName(“ComplexETL”).getOrCreate()

# Define the external source and target paths

source\_path = ‘your\_path’ # Update with your actual source file path

target\_path = “your\_output\_path” # Update with your desired target file path

# Extract: Read data from an external CSV file

df = spark.read.csv(source\_path, header=True,schema = ‘cust\_id int, first\_name string,last\_name string,cust\_order int,cust\_status string’)

# Transformation 1: Concatenate First and Last Names

df = df.withColumn(“full\_name”, concat(col(“first\_name”), lit(“ “), col(“last\_name”)))

# Transformation 2: Calculate Net Salary (subtract 10% as taxes)

df = df.withColumn(“net\_salary”, floor(lit(10000) + rand() \* lit(50)) )

#adding age column

df = df.withColumn(“age”, floor(lit(20) + rand() \* lit(31)))

# Transformation 3: Filter by Age (age >= 30)

df = df.filter(col(“age”) >= 30)

# Transformation 4: Group by Age and Calculate Average Salary

avg\_salary\_by\_age = df.groupBy(“age”).agg({“net\_salary”: “avg”}).withColumnRenamed(“avg(salary)”, “avg\_salary”)

# Transformation 5: Sort by Age

df = df.orderBy(“age”)

# Save the transformed data to an external CSV file

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