YAZHINI S - PYSPARK CODING ASSESSMENT

1. **Implement Processing JSON and CSV data with PySpark**

**CSV FILE CODE**:

from pyspark.sql import SparkSession

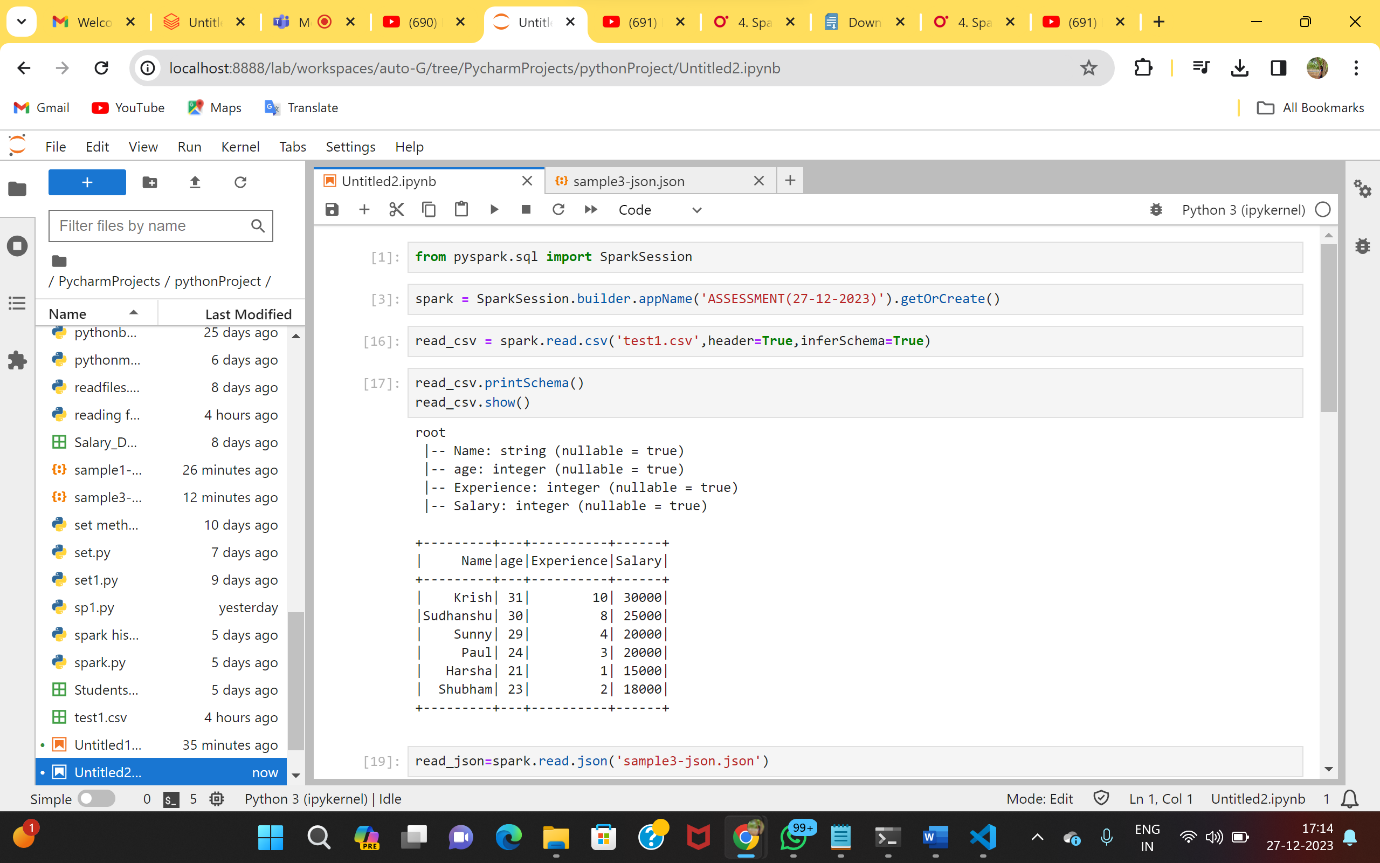
spark = SparkSession.builder.appName('ASSESSMENT(27-12-2023)').getOrCreate()

read\_csv = spark.read.csv('test1.csv',header=True,inferSchema=True)

read\_csv.printSchema()

read\_csv.show()

**Reading CSV file:**



**JSON FILE CODE:**

from pyspark.sql import SparkSession

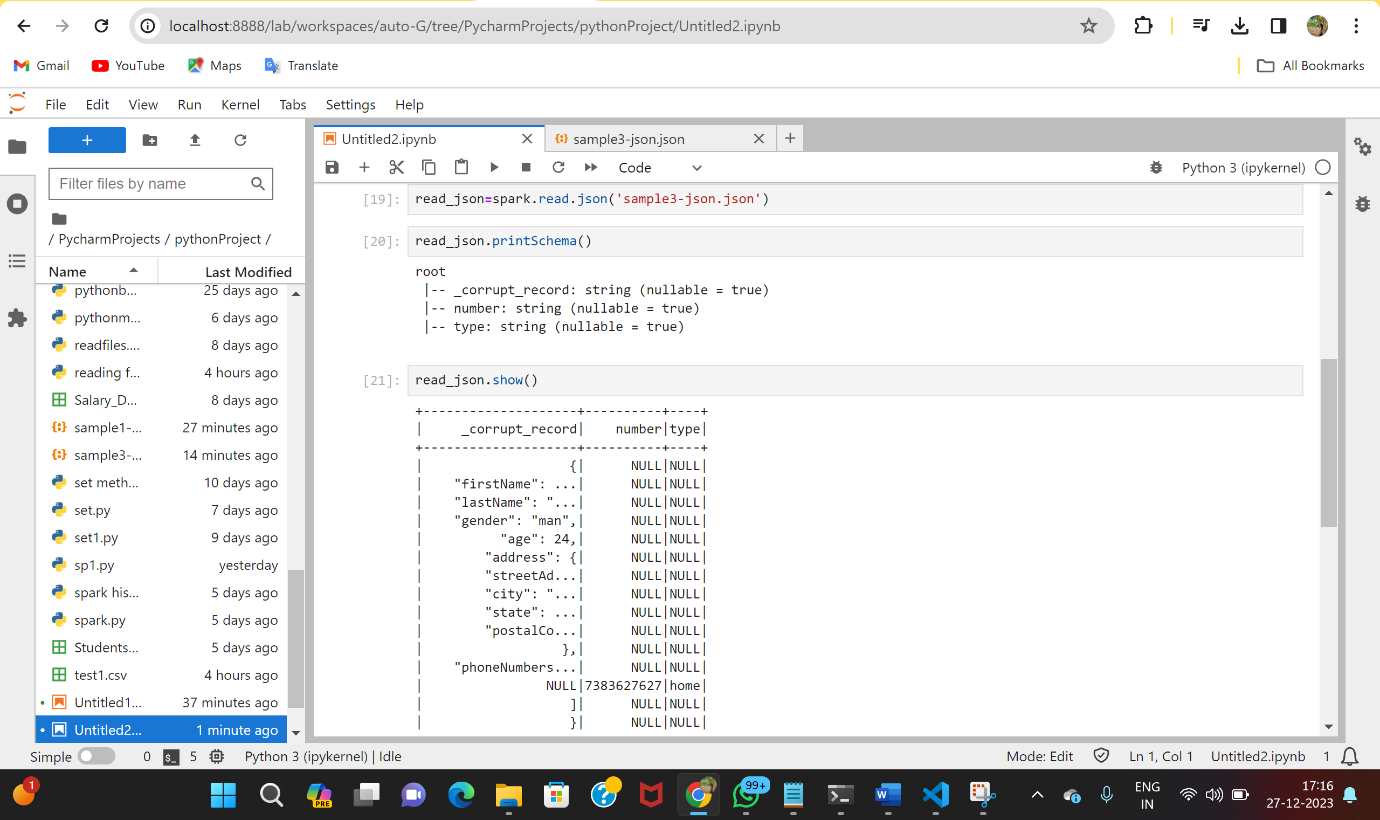
spark = SparkSession.builder.appName('ASSESSMENT(27-12-2023)').getOrCreate()

read\_json=spark.read.json('sample3-json.json')

read\_json.printSchema()

read\_json.show()

**Reading JSON file:**



2)**Explaining ETL**

**ETL :**

🡪ETL stands for Extract, Transform and Load.

🡪 ETL is the powerhouse that combines the simplicity of Python with the scalability and performance of Spark.

**ETL WORKFLOW:**

Extract:

🡪Retrieve data from various sources like databases, files, or APIs.  
Transform:

🡪Clean, aggregate, and manipulate data to fit your analysis needs.  
Load:

🡪Store the transformed data into a database or data warehouse for analysis.

**FEATURES OF ETL:**

Performance:

🡪 PySpark leverages in-memory computing, making ETL processes faster than ever.  
Ease of Use:

🡪 Python developers can seamlessly transition to PySpark due to its Pythonic syntax.  
Scalability:

🡪Handle massive datasets with ease, thanks to Spark’s distributed processing.  
Rich Ecosystem:

🡪 PySpark integrates with popular tools and libraries, making it versatile for various data tasks.

**3 & 4)Creating & Tranforming Using Spark SQL**

🡪Initialize Spark Session:

Start by initializing a Spark session, which is the entry point for interacting with Spark functionality**.**

🡪Extract Data:

Load data from different sources like CSV, JSON, or databases into Spark DataFrames.

🡪Transform Data:

Apply transformations on the DataFrames, such as selecting columns, filtering, renaming, or aggregating.

🡪Create Database and Tables:

Define a database and create tables within it to organize your data.

🡪Perform SQL Operations:

Utilize Spark SQL to run SQL queries on the created tables, enabling complex data manipulations.

🡪Load Data into Target:

Store the transformed data into a target location or database.

Code:

from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("etl\_process").getOrCreate()

read\_csv = spark.read.csv('annual.csv',header=True,inferSchema=True)

read\_csv.show()

read\_csv.head(4)

transformed\_csv=read\_csv.select("Year","Variable\_name","Value","Industry\_aggregation\_NZSIOC")\

.filter((read\_csv["Year"]==2021) & (read\_csv["Variable\_name"].isin(["Total income","Total Expenditure"])))

transformed\_csv.show()

spark.sql(“USE mydatabase”

read\_csv.createOrReplaceTempView("annual\_survay")

spark.sql("SELECT \* FROM annual\_survay").show()

Output: