

# Azure Databricks Case Study

Submitted By- Aniroop Gupta  
DE BATCH-1

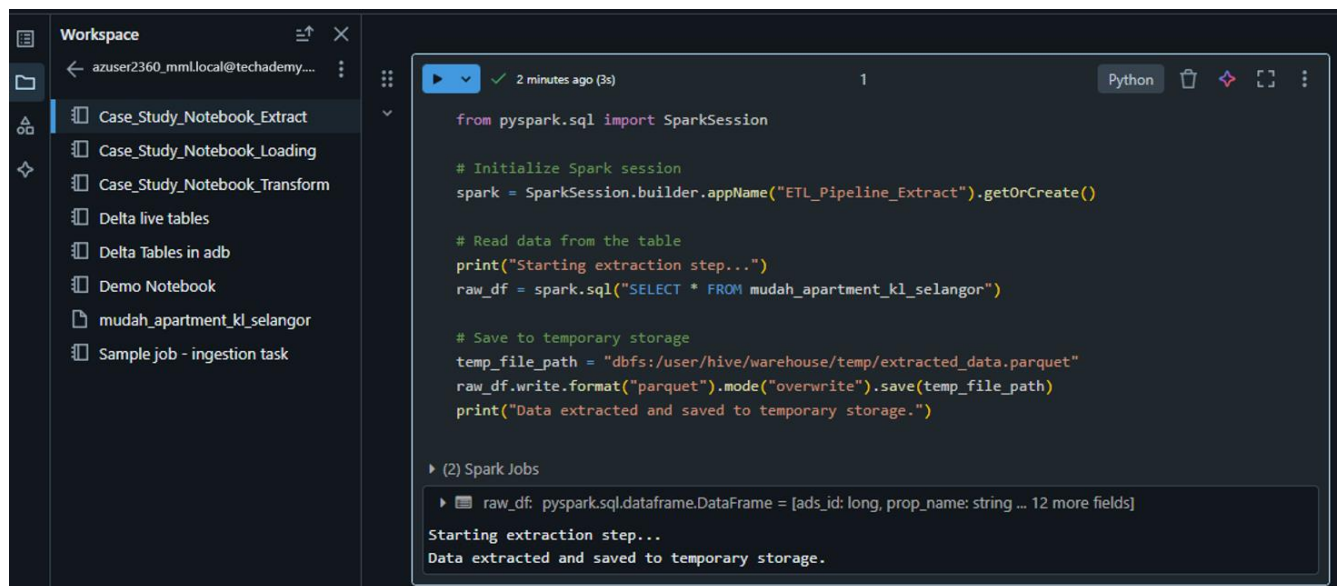
**Task given:** Create an ETL pipeline of ingestion & transform and load queries on any data set and initiate the pipeline from workflow using the notebook.

## Steps:

- Create a notebook with ETL queries.
- Run the notebook from workflow pipeline in azure databricks workspace.

## 1. Notebook for Ingestion (Extract):

This notebook will load the raw data into the Databricks environment, either from a file or a table.



The screenshot displays the Azure Databricks workspace interface. On the left, a sidebar shows the 'Workspace' tree with a list of notebooks: 'Case\_Study\_Notebook\_Extract', 'Case\_Study\_Notebook\_Loading', 'Case\_Study\_Notebook\_Transform', 'Delta live tables', 'Delta Tables in adb', 'Demo Notebook', 'mudah\_apartment\_kl\_selangor', and 'Sample job - ingestion task'. The main area shows the 'Case\_Study\_Notebook\_Extract' notebook, which has been executed successfully 2 minutes ago. The code in the notebook is as follows:

```
from pyspark.sql import SparkSession

# Initialize Spark session
spark = SparkSession.builder.appName("ETL_Pipeline_Extract").getOrCreate()

# Read data from the table
print("Starting extraction step...")
raw_df = spark.sql("SELECT * FROM mudah_apartment_kl_selangor")

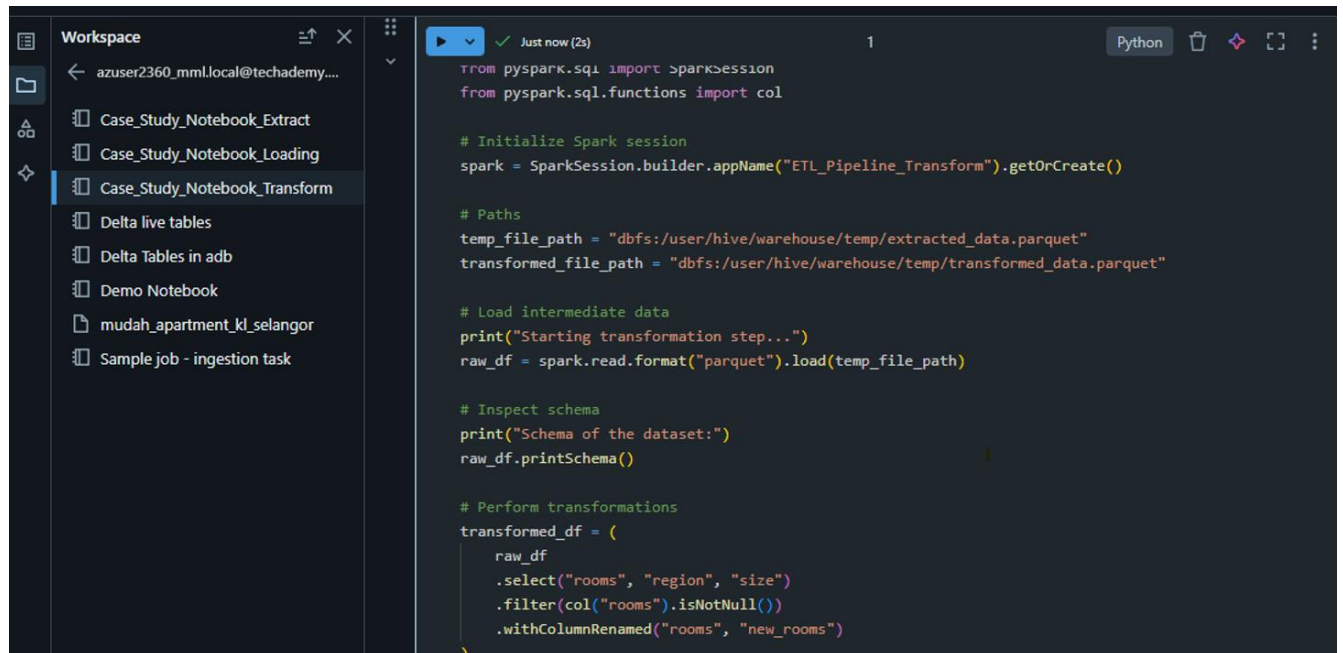
# Save to temporary storage
temp_file_path = "dbfs:/user/hive/warehouse/temp/extracted_data.parquet"
raw_df.write.format("parquet").mode("overwrite").save(temp_file_path)
print("Data extracted and saved to temporary storage.")
```

Below the code, the execution results are shown under the heading '(2) Spark Jobs'. It indicates that the raw data has been successfully loaded into a DataFrame and saved to the specified temporary storage path.

```
raw_df: pyspark.sql.dataframe.DataFrame = [ads_id: long, prop_name: string ... 12 more fields]
Starting extraction step...
Data extracted and saved to temporary storage.
```

## 2. Notebook for Transformation

This notebook will read the extracted data, apply transformations, and prepare it for loading.



```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col

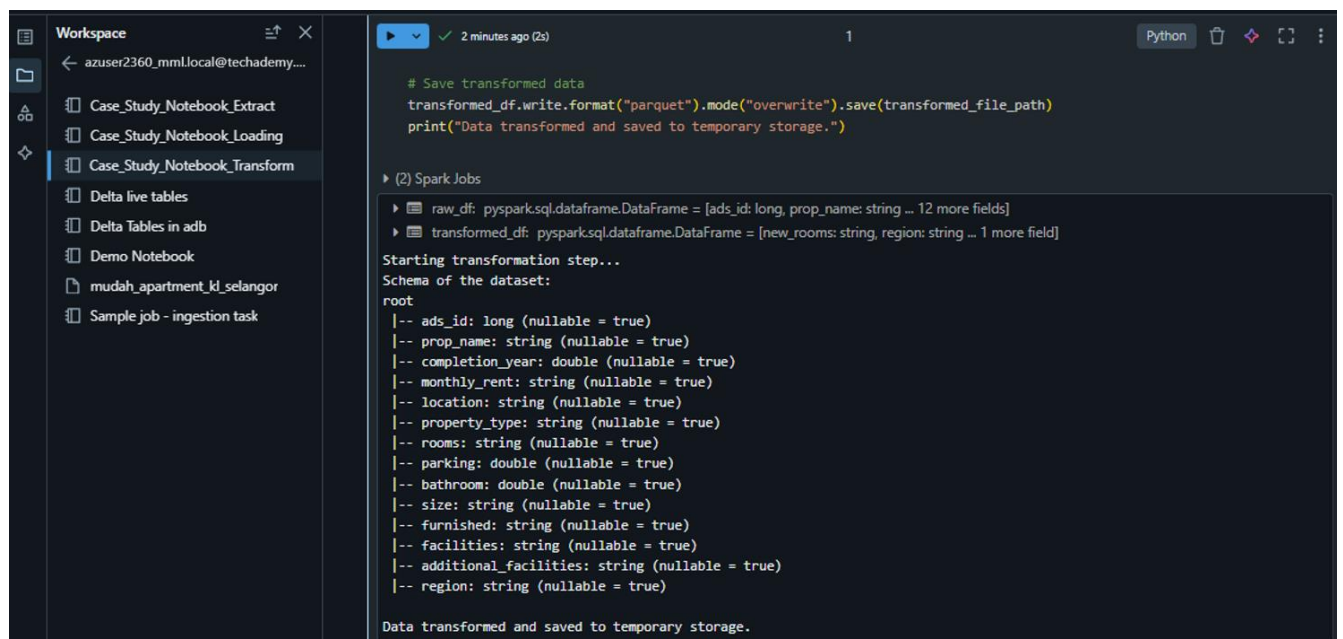
# Initialize Spark session
spark = SparkSession.builder.appName("ETL_Pipeline_Transform").getOrCreate()

# Paths
temp_file_path = "dbfs:/user/hive/warehouse/temp/extracted_data.parquet"
transformed_file_path = "dbfs:/user/hive/warehouse/temp/transformed_data.parquet"

# Load intermediate data
print("Starting transformation step...")
raw_df = spark.read.format("parquet").load(temp_file_path)

# Inspect schema
print("Schema of the dataset:")
raw_df.printSchema()

# Perform transformations
transformed_df = (
    raw_df
    .select("rooms", "region", "size")
    .filter(col("rooms").isNotNull())
    .withColumnRenamed("rooms", "new_rooms")
)
```



```
# Save transformed data
transformed_df.write.format("parquet").mode("overwrite").save(transformed_file_path)
print("Data transformed and saved to temporary storage.")
```

▶ (2) Spark Jobs

- ▶ raw\_df: pyspark.sql.dataframe.DataFrame = [ads\_id: long, prop\_name: string ... 12 more fields]
- ▶ transformed\_df: pyspark.sql.dataframe.DataFrame = [new\_rooms: string, region: string ... 1 more field]

Starting transformation step...

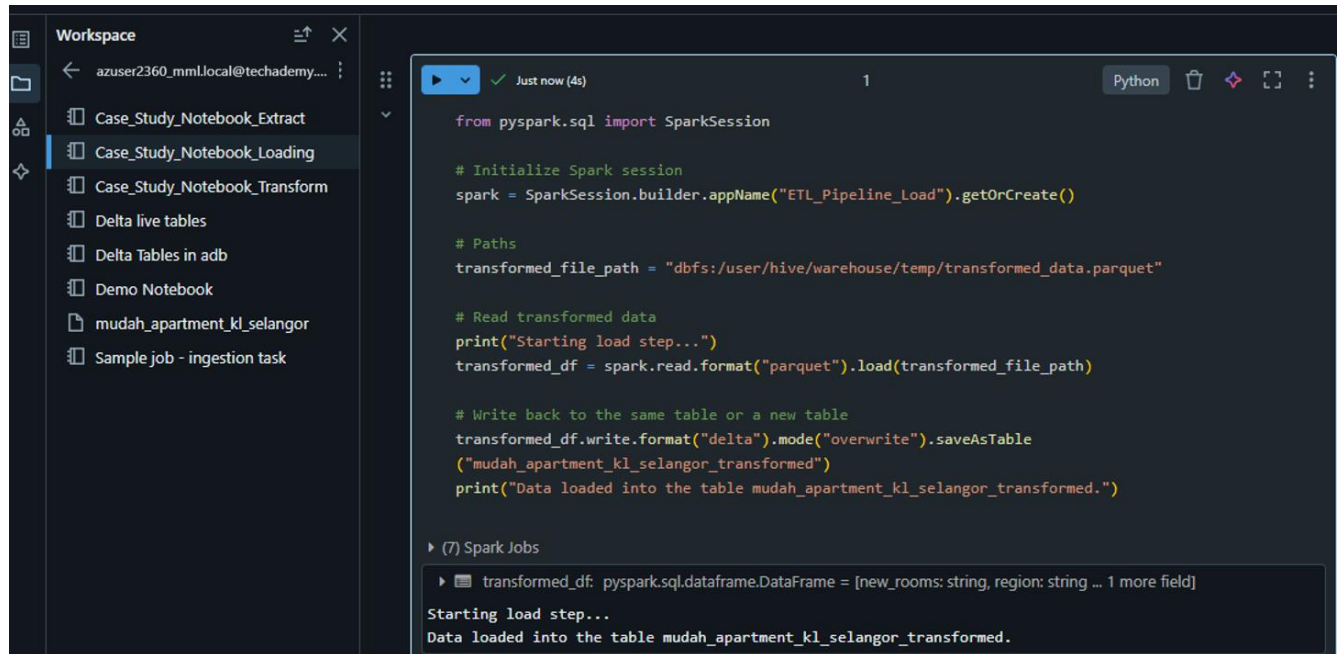
Schema of the dataset:

```
root
 |-- ads_id: long (nullable = true)
 |-- prop_name: string (nullable = true)
 |-- completion_year: double (nullable = true)
 |-- monthly_rent: string (nullable = true)
 |-- location: string (nullable = true)
 |-- property_type: string (nullable = true)
 |-- rooms: string (nullable = true)
 |-- parking: double (nullable = true)
 |-- bathroom: double (nullable = true)
 |-- size: string (nullable = true)
 |-- furnished: string (nullable = true)
 |-- facilities: string (nullable = true)
 |-- additional_facilities: string (nullable = true)
 |-- region: string (nullable = true)
```

Data transformed and saved to temporary storage.

### 3. Notebook for Loading

This notebook will load the transformed data into the final destination, such as a Delta table or another storage format.



The screenshot shows a Databricks workspace with a notebook titled "Case\_Study\_Notebook\_Loading". The notebook code is as follows:

```
from pyspark.sql import SparkSession

# Initialize Spark session
spark = SparkSession.builder.appName("ETL_Pipeline_Load").getOrCreate()

# Paths
transformed_file_path = "dbfs:/user/hive/warehouse/temp/transformed_data.parquet"

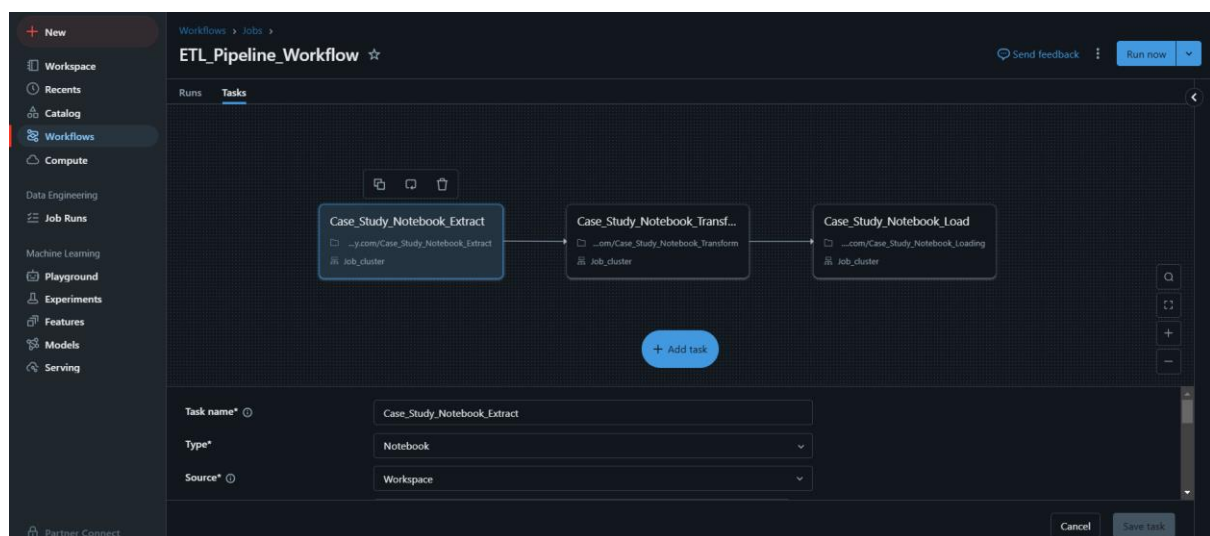
# Read transformed data
print("Starting load step...")
transformed_df = spark.read.format("parquet").load(transformed_file_path)

# Write back to the same table or a new table
transformed_df.write.format("delta").mode("overwrite").saveAsTable(
    "mudah_apartment_kl_selangor_transformed")
print("Data loaded into the table mudah_apartment_kl_selangor_transformed.")
```

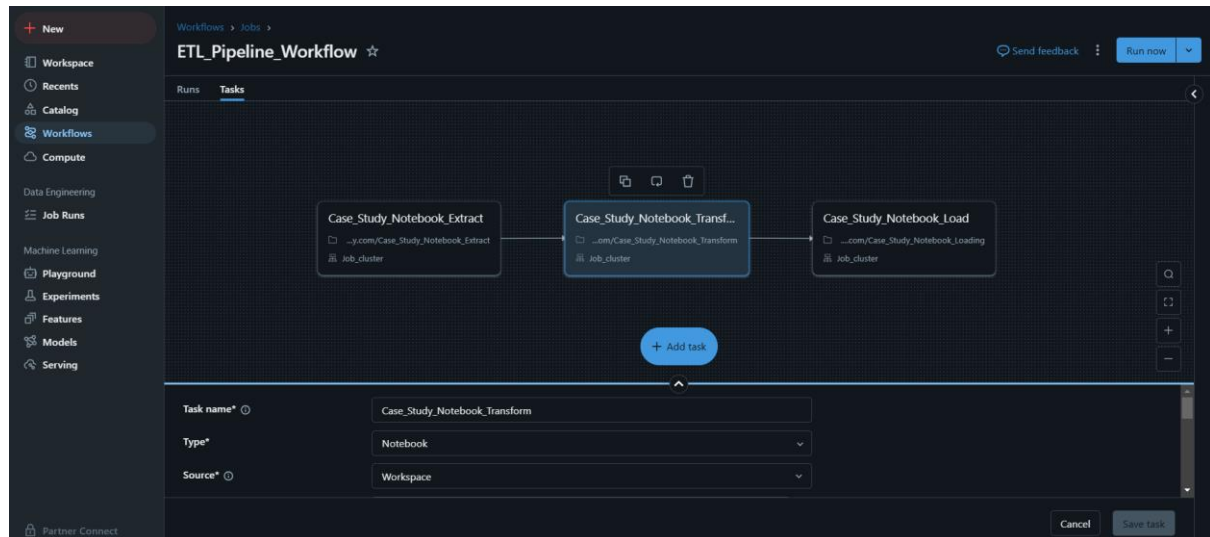
The output of the notebook shows the Spark session is initialized, the data is read from the specified path, and it is then written to a new Delta table named "mudah\_apartment\_kl\_selangor\_transformed".

### 4. Creating Databricks Workflow:

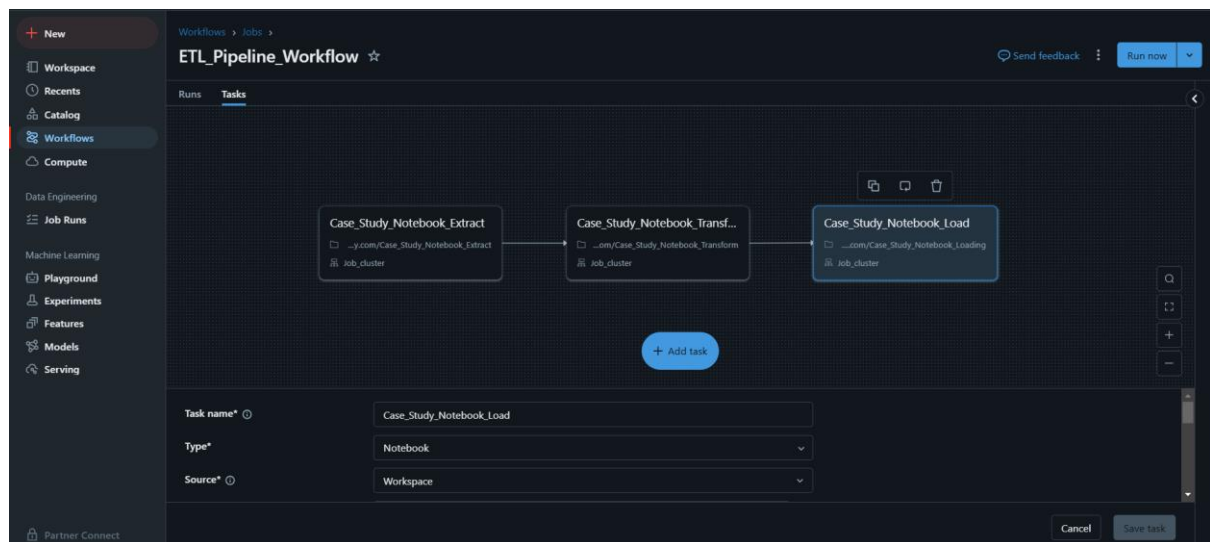
- For Extract process-



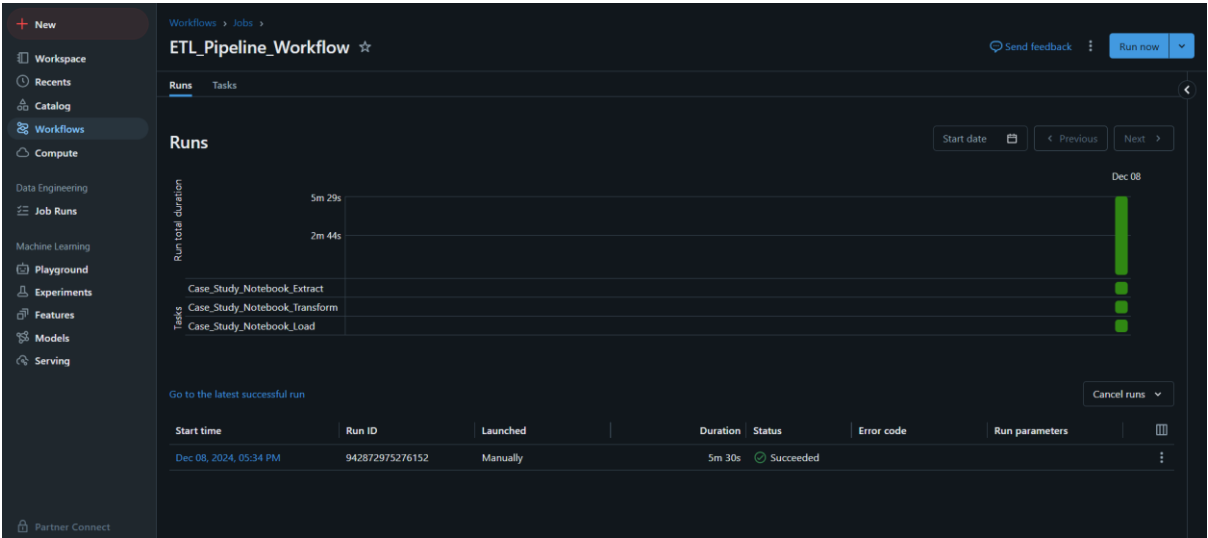
- For Transformation process-



- For Loading process-



# 5. Running the Workflows:



- Hence, The ETL Pipeline consisting processes of Extraction, Transformation, Loading has successfully created and running.