Case Study-4

Azure Databricks Case Study

Name: **Harish E.R**

Q. create a ETL pipeline of ingestion & transform and load queries on any data set and initiate the pipeline from workflow using notebook.

1. **Notebook for Ingestion (Extract)**
   This notebook will load the raw data into the Databricks environment, either from a file or a table.

Yesterday (1s) 1 Python

```python
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.")
```

```
raw_df: pyspark.sql.dataframe.DataFrame
    ads_id: long
    prop_name: string
    completion_year: double
    monthly_rent: string
    location: string
    property_type: string
    rooms: string
    parking: double
    bathroom: double
    size: string
    furnished: string
    facilities: string
    additional_facilities: string
    region: string

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.

Yesterday (1s) 1 Python

```python
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")
)
```

```python
# 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.

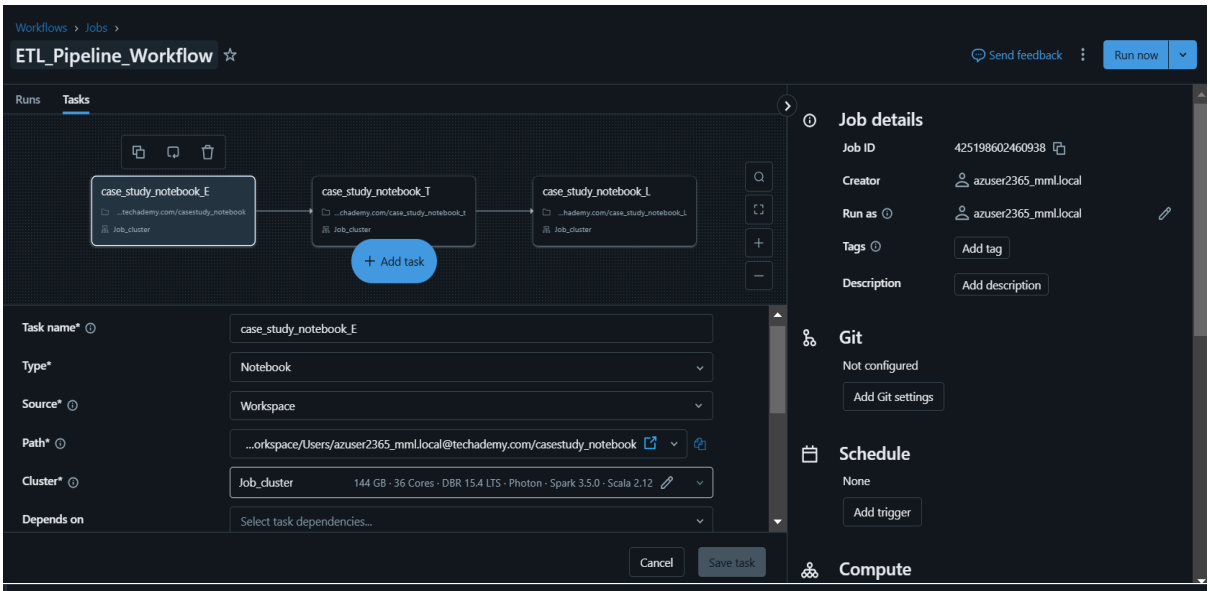
Yesterday (6s) 1 Python

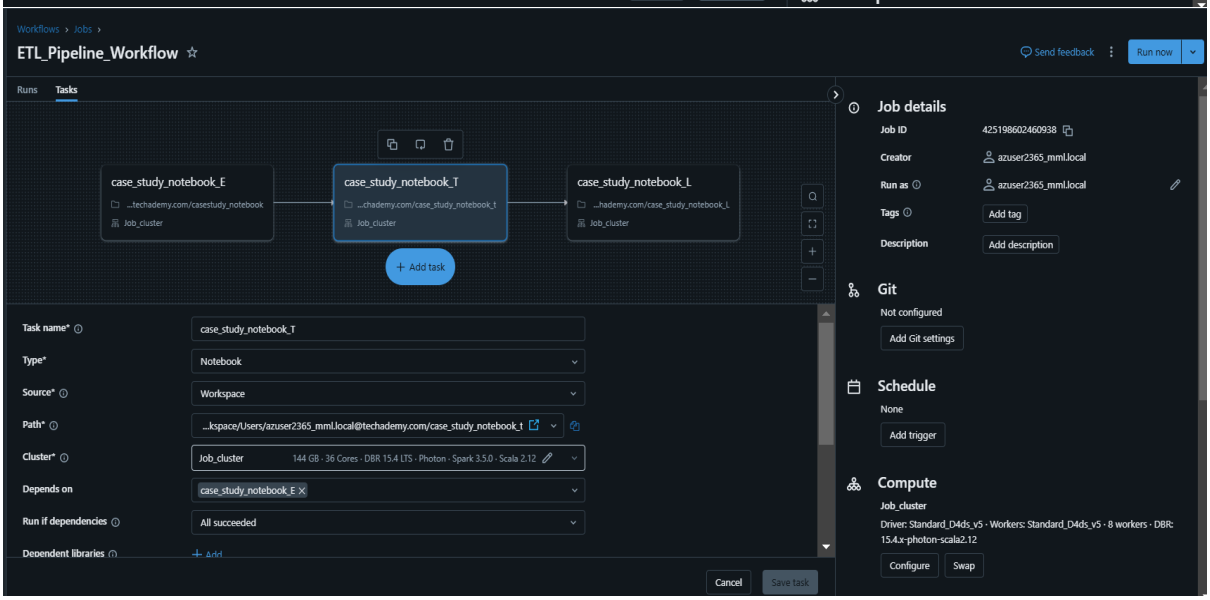
```python
from pyspark.sql import SparkSession

# 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.")
```

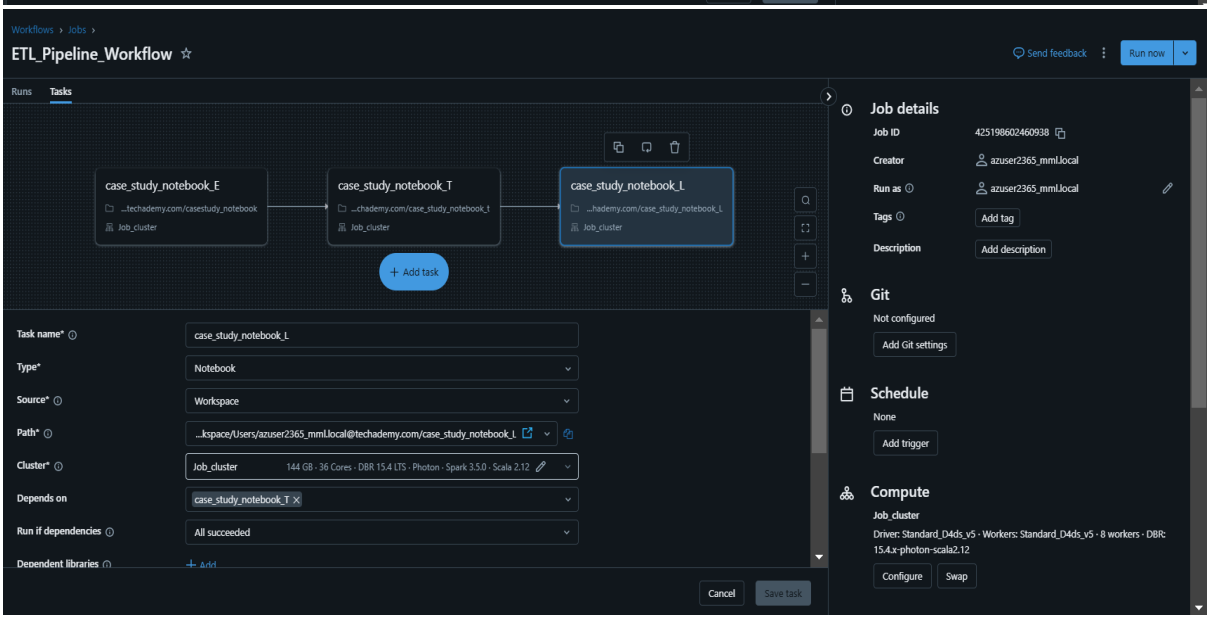
(7) Spark Jobs

```
transformed_df: pyspark.sql.dataframe.DataFrame
        new_rooms: string
        region: string
        size: string

Starting load step...
Data loaded into the table mudah_apartment_kl_selangor_transformed.
```

4. Create a Databricks Workflow







5. Run the Workflow