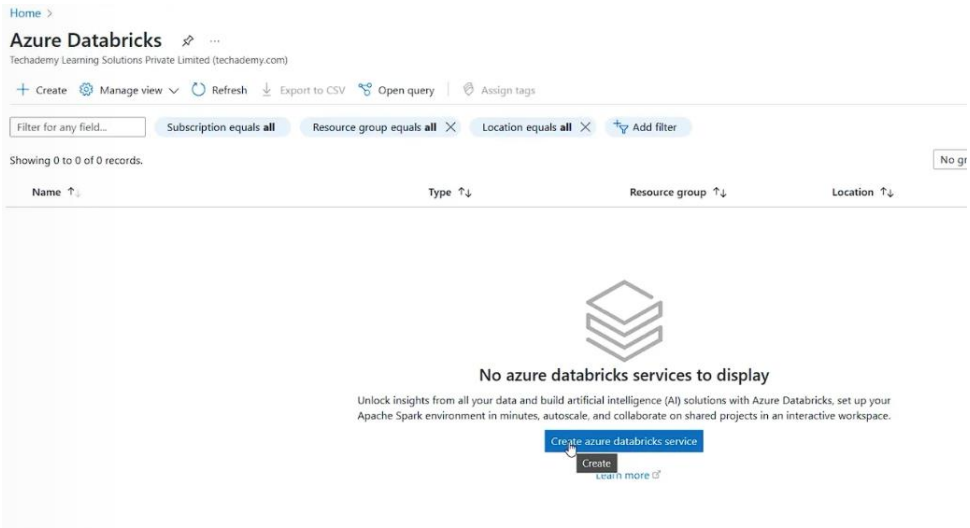
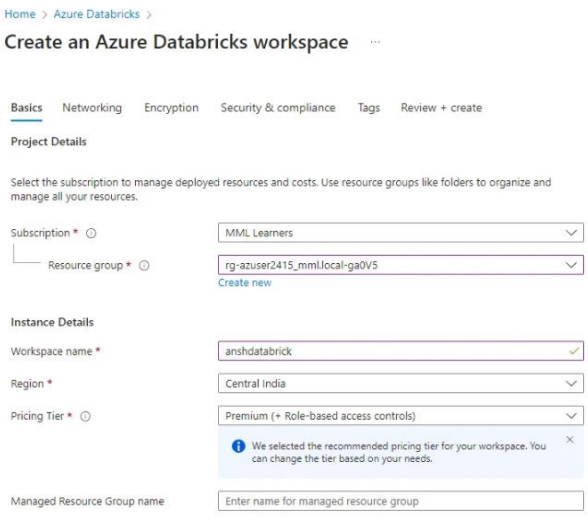
Ansh Ranjan
Azure Data
EXERCISE 5: DataBricks for Data Engineering

TASK 1: Deploy a DataBricks workspace

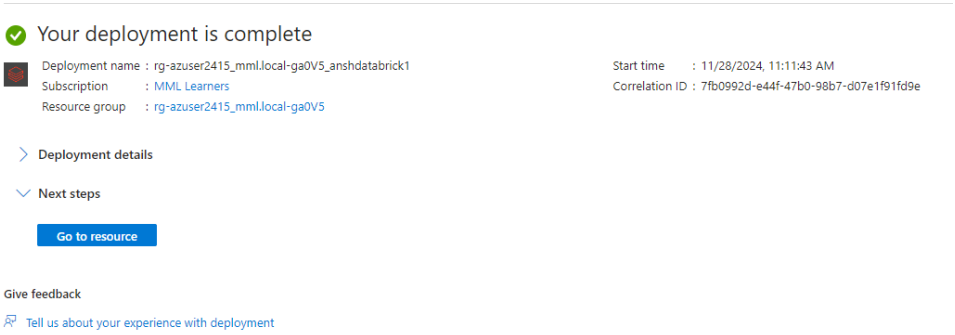
1. **Sign in to the Azure Portal**:
    - Go to Azure Portal and log in with your credentials.
2. **Create a New Resource**:
    - Click **Create a resource** on the left-hand menu.
    - Search for **Azure Databricks** in the search bar.
    - Select **Azure Databricks** from the results and click **Create**.



3. **Configure the Workspace**:
    - **Resource Group**: Choose an existing resource group or create a new one.
    - **Workspace Name**: Provide a unique name for your Databricks workspace.
    - **Region**: Select the Azure region closest to your users or data source for better performance.
    - **Pricing Tier**: Select a pricing tier based on your requirements (Standard, Premium, or Trial).



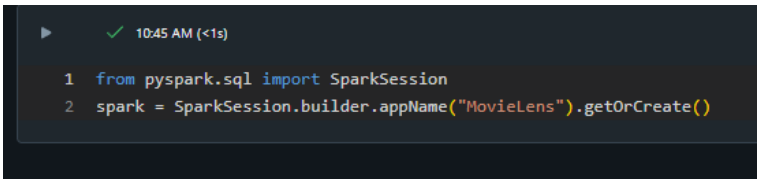
4. **Networking (Optional)**:
    - Configure network settings if required, such as deploying the workspace in a Virtual Network (VNet).
5. **Review + Create**:
    - Click **Review + Create** to validate your configuration.
    - Once validation is successful, click **Create** to deploy the workspace.
6. **Wait for Deployment**:
    - The deployment process might take a few minutes. Monitor the progress in the **Notifications** section of the portal.



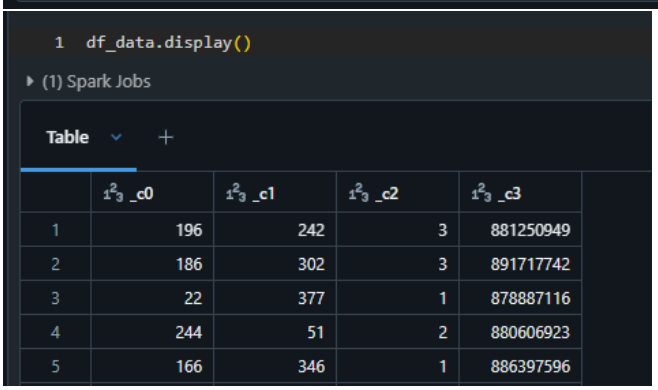
7. **Access the Workspace**:
    - Once deployment is complete, go to the **Resource** to access the Databricks workspace.
    - Click the **Launch Workspace** button to open the Azure Databricks environment.

TASK 2: Process a big sample dataset

1. Go to your DataBricks Workspace > Create > New Notebook and initialize your spark session. You will need a cluster to run queries



2. Extraction



	_c0	_c1	_c2	_c3
1	196	242	3	881250949
2	186	302	3	891717742
3	22	377	1	878887116
4	244	51	2	880606923
5	166	346	1	886397596

5. Since no column names are provided, we need to add them manually

```
column_mapping = {
    "_c0": "UserId",
    "_c1": "movieId",
    "_c2": "rating",
    '_c3' : 'timeStamp'
}

for col_old, col_new in column_mapping.items():
    df_data=df_data.withColumnRenamed(col_old, col_new)
```

6. Getting Schema of the dataframe

```
df_data.printSchema()
```

```
root
 |-- UserId: integer (nullable = true)
 |-- movieId: integer (nullable = true)
 |-- rating: integer (nullable = true)
 |-- timeStamp: integer (nullable = true)
```

7. Repeating same steps for other dataframe and creating Views

```
12:07 PM (<1s)

df_data.createOrReplaceTempView('Data')
df_user.createOrReplaceTempView('User')
```

8. Deriving Analytics from the data
   Gender wise user breakdown

```
spark.sql("select gender, count(*) from User group by gender").show()
```

(2) Spark Jobs

```
+------+--------+
|gender|count(1)|
+------+--------+
|     F|     273|
|     M|     670|
+------+--------+
```

Give the top 5 movies which are reviewed maximum number of times

```
spark.sql("select movieID, count(*) as Num_times from Data group by movieID order by Num_times desc limit 5").show()
```

(2) Spark Jobs

```
+-------+---------+
|movieID|Num_times|
+-------+---------+
|     50|      583|
|    258|      509|
|    100|      508|
|    181|      507|
|    294|      485|
+-------+---------+
```

List the top 10 movies which received highest number of 5 star ratings

```
spark.sql("select movieID, count(*) as Num_5star_reviews from Data where rating = 5 group by movieID order by Num_5star_reviews desc limit 5").show()
```

(2) Spark Jobs

```
+-------+-----------------+
|movieID|Num_5star_reviews|
+-------+-----------------+
|     50|              325|
|    100|              227|
|    127|              214|
|    174|              202|
|     56|              188|
+-------+-----------------+
```

TASK 3: DataBricks Key Features and use cases

Key Features of Databricks

- Unified platform for **data engineering, analytics, and ML**
- Built on **Apache Spark** for fast, distributed processing
- Supports **Delta Lake** with ACID transactions and schema enforcement
- Collaborative **notebooks** with multi-language support (Python, SQL, etc.)
- Built-in **visualizations** and **MLflow** for ML lifecycle
- **Auto-scaling clusters**, CI/CD, and cloud integration (Azure, AWS, GCP)

Use Cases

- **ETL & Data Pipelines**: Ingest, transform, and clean large datasets
- **Data Lakehouse**: Unified storage and analytics using Delta Lake
- **Machine Learning**: Build, train, and deploy ML models
- **Real-Time Analytics**: Process streaming data
- **Business Intelligence**: Connect with Power BI/Tableau for reporting