Lab 1: Using Copilot for SQL Query Generation & Optimization

Objective: Generate and optimize SQL queries to analyze sales data.

Step 1: Load Data into a Database

- Import sales.csv into a SQL database.
    - Example (SQL Server):

```
CREATE TABLE Sales (
  SalesOrderNumber VARCHAR(10),
  SalesOrderLineNumber INT,
  OrderDate DATE,
  CustomerName VARCHAR(50),
  EmailAddress VARCHAR(50),
  Item VARCHAR(50),
  Quantity INT,
  UnitPrice DECIMAL(10, 2),
  TaxAmount DECIMAL(10, 2)
);
-- Use BULK INSERT or SSMS Import Wizard to load sales.csv
```

- Verify data: SELECT TOP 5 * FROM Sales;

Step 2: Generate a Simple Query with Copilot

- Prompt Copilot: "Write a SQL query to find total sales by customer for July 2019."
- Copilot Output (example):

```
SELECT
  CustomerName,
  SUM(Quantity * UnitPrice) AS TotalSales
FROM Sales
WHERE OrderDate BETWEEN '2019-07-01' AND '2019-07-31'
GROUP BY CustomerName
ORDER BY TotalSales DESC;
```

- Run the query and review results (e.g., Christy Zhu: $3399.99, Julio Ruiz: $3374.99).

Step 3: Optimize the Query

- Prompt Copilot: "Optimize this query for performance on a large dataset."
- Copilot Suggestion:

```
-- Add index recommendation
CREATE NONCLUSTERED INDEX IX_Sales_OrderDate
ON Sales (OrderDate) INCLUDE (CustomerName, Quantity, UnitPrice);


SELECT
  CustomerName,
  SUM(Quantity * CAST(UnitPrice AS DECIMAL(10, 2))) AS TotalSales
FROM Sales
WHERE OrderDate >= '2019-07-01' AND OrderDate <= '2019-07-31'
GROUP BY CustomerName
WITH (INDEX(IX_Sales_OrderDate))
ORDER BY TotalSales DESC;
```

- **Why Optimized?**:
    - Index on OrderDate speeds up filtering.
    - Explicit date range avoids potential type mismatches.
    - INCLUDE clause covers columns in the query, reducing lookups.

Step 4: Validate Results

- Run the optimized query and compare execution time with the original (use SET STATISTICS TIME ON; in SQL Server).

Lab 2: Using Copilot to Create Fabric Pipelines

Objective: Build a pipeline in Microsoft Fabric to ingest, transform, and store sales data.

Step 1: Set Up Fabric Environment

- Access Microsoft Fabric (via Power BI or Azure portal).
- Create a workspace and a Lakehouse (e.g., SalesLakehouse).

Step 2: Ingest Data with Copilot

- In Fabric, go to the Data Factory section and create a new pipeline.
- Prompt Copilot: "Create a pipeline to load sales.csv from a local file into a Lakehouse."
- Copilot Output (pseudocode for Fabric activities):
    - **Copy Data Activity**:
        - Source: File System (upload sales.csv).
        - Sink: Lakehouse table (SalesRaw).
        - Mapping: Auto-map columns.
- Run the pipeline and verify data in SalesRaw (e.g., SELECT * FROM SalesRaw LIMIT 5;).

Step 3: Transform Data with Copilot

- Prompt Copilot: "Add a transformation to calculate TotalSales and filter for July 2019."
- Copilot Output (Dataflow or Notebook script):

```
# PySpark in Fabric Notebook
from pyspark.sql import functions as F

# Load raw data
df = spark.read.table("SalesRaw")

# Add TotalSales column and filter
df_transformed = df.withColumn("TotalSales", F.col("Quantity") * F.col("UnitPrice")) \
    .filter((F.col("OrderDate") >= "2019-07-01") & (F.col("OrderDate") <= "2019-07-31"))

# Write to new table
```

df_transformed.write.mode("overwrite").format("delta").saveAsTable("SalesTransformed")

- Run the transformation and check SalesTransformed.

Step 4: Automate Pipeline

- Add a schedule trigger (e.g., daily) to the pipeline via Copilot's UI suggestions.
- Test the end-to-end flow: ingestion → transformation → storage.

Lab 3: Optimizing Power BI Reports with Copilot

Objective: Build and optimize a Power BI report using sales data.

Step 1: Connect to Data

- Open Power BI Desktop.
- Connect to the SalesTransformed table in Fabric Lakehouse (or import sales.csv directly).
- Load data and ensure OrderDate is recognized as a date type.

Step 2: Create a Basic Report

- Prompt Copilot (in Power BI): "Create a report showing total sales by item for July 2019."
- Copilot Output:
    - Visual: Bar chart.
    - X-Axis: Item.
    - Values: Sum of TotalSales (create measure if needed: TotalSales = SUM(Sales[Quantity] * Sales[UnitPrice])).
    - Filter: OrderDate = July 2019.
- Result: Top items like "Road-150 Red" (~~$64,408.86) and "Mountain-100 Silver"~~ ($23,799.93) appear.

Step 3: Optimize the Report

- Prompt Copilot: "Optimize this report for faster loading."
- Copilot Suggestions:
    - **Measure Optimization**:

dax

CollapseWrapCopy

TotalSalesOptimized =

CALCULATE(

  SUMX(Sales, Sales[Quantity] * Sales[UnitPrice]),

  FILTER(Sales, Sales[OrderDate] >= DATE(2019, 7, 1) && Sales[OrderDate] <= DATE(2019, 7, 31))

)

    - **Data Model**: Remove unused columns (e.g., EmailAddress if not needed).
    - **Visuals**: Use aggregated data (e.g., pre-aggregate in Fabric) instead of raw rows.
- Apply changes and test refresh time.

Step 4: Add Insights

- Use Power BI's Copilot to generate insights: "What are the key trends in this data?"
- Example Output: "Sales of Road-150 Red dominate, with consistent daily orders from July 1-31, 2019."
- Add a slicer for OrderDate and a KPI card for total sales.