**Documentation**

**Prerequisites:**

- Jupyter Notebook

- Python libraries: pandas, sqlite3

**Step 1: Data Extraction**

Data Extraction

- Read data from the CSV file into a DataFrame

```python

import pandas as pd

# Define the CSV file path

csv\_file = 'E:\\BIG DATA BY ineuron\\PuntPartenerTestAssignment\\ASSIGNMENT\\Updated\_sales\_data.csv'

# Extract: Read data from CSV into a DataFrame

df = pd.read\_csv(csv\_file)

# Verify data extraction

df.head(5)

```

**Step 2: Data Transformation**

Data Transformation

- Error: Some values are null and in string format.

- Solution: Convert to numeric values

```python

df['Quantity Ordered'] = pd.to\_numeric(df['Quantity Ordered'], errors='coerce')

df['Price Each'] = pd.to\_numeric(df['Price Each'], errors='coerce')

```

- Calculate total sales, extract year and month, and aggregate data

- Transform: Calculate total sales for each order

```python

df['total\_sales'] = df['Quantity Ordered'] \* df['Price Each']

```

- Error Found: Unknown string format in 'Order Date'

- Solution: Specify the expected date format

```python

df['Order Date'] = pd.to\_datetime(df['Order Date'], format='%m/%d/%y %H:%M', errors='coerce')

```

- Transform: Extract year and month from the order\_date

```python

df['Order Year'] = df['Order Date'].dt.year

df['Order Month'] = df['Order Date'].dt.month

```

- Aggregate: Calculate total sales for each product in each month

```python

product\_sales = df.groupby(['Product', 'Order Year', 'Order Month'])['total\_sales'].sum().reset\_index()

```

**Step 3: Data Loading**

Data Loading

- Create a SQLite database and define tables to store the transformed data

```python

import sqlite3

# Create a SQLite database and a connection

conn = sqlite3.connect('sales\_data.db')

# Define tables and load data into them

df.to\_sql('orders', conn, index=False, if\_exists='replace')

product\_sales.to\_sql('product\_sales', conn, index=False, if\_exists='replace')

```

**Step 4: SQL Queries**

SQL Queries

- What was the total sales amount for each product in the last quarter (last 3 months)?

```sql

SELECT product, SUM(total\_sales) as total\_sales

FROM product\_sales

WHERE order\_year = (SELECT MAX(Order) FROM product\_sales)

AND order\_month >= (SELECT MAX(order\_month) FROM product\_sales) - 2

GROUP BY product;

```

- List the top 5 products by total sales amount for the entire dataset.

```sql

SELECT product, SUM(total\_sales) as total\_sales

FROM product\_sales

GROUP BY product

ORDER BY total\_sales DESC

LIMIT 5;

```

- Calculate the monthly average sales for each product over the entire dataset.

```sql

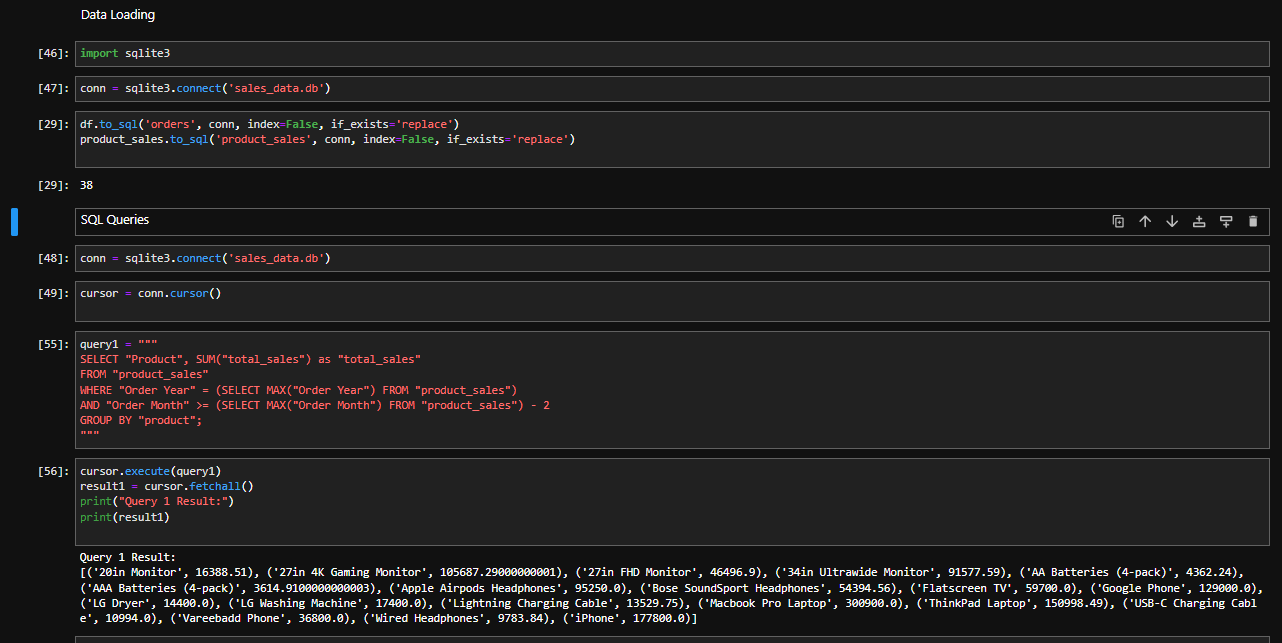
SELECT product, AVG(total\_sales) as avg\_sales

FROM product\_sales

GROUP BY product;

```

Run Query images:





**Step 5: Data Validation and Error Handling (Bonus)**

Data Validation

- Check for missing values in essential columns

- Handle missing values appropriately (e.g., log and/or skip rows)

```python

missing\_values = df.isnull().sum()

if any(missing\_values):

# Handle missing values

```

- Check for anomalies in data (negative quantities or prices)

- Handle anomalies appropriately (e.g., correct values, log issues)

```python

anomalies = df[(df['Quantity Ordered'] < 0) | (df['Price Each'] < 0)]

if not anomalies.empty:

# Handle anomalies

```

Error Handling

- Handle exceptions during the ETL process

- Handle other exceptions and log appropriately

```python

try:

# Code for ETL

except FileNotFoundError as e:

# Log the error and exit gracefully

print(f"Error: {e}")

except Exception as e:

# Handle other exceptions and log appropriately

print(f"Error: {e}")

```