Bytewise Fellowship

Task 7: Extract, Transform and Load (ETL) Task on Sales Data using Python

Name	Muhammad Ahsan Saleem
Date	16th July, 2024
Submitted to:	Muhammad Bilal

Introduction:

Task 7 of the Bytewise Fellowship program, focused on the ETL (Extract, Transform, Load) process using Python. The task involves reading sales data from a CSV file, performing necessary transformations to clean the data, and loading the cleaned data into a PostgreSQL database. The specific data cleaning steps include ensuring order_id is an integer, removing rows with product_id of 0, capping amount at 1500 Rs, removing rows with null status, and eliminating duplicates.

Task 7 - ETL Process on Sales Data using Python

Task Statement: You'll be given a csv dataset file and you'll be extracting data from it through pandas or pyspark. Then there will be the transformation phase and after performing that you'll try to load the final data to a database.

There are some problems with the data, and you need to remove them during the transformation phase to make data useful for everyone:

1. An order id should always exist as an integer.
2. A product id cannot be 0.
3. We never had a product priced more than 1500 Rs. so, any item with amount greater. than 1500 Rs is an anomaly, and it should be treated as 1500 Rs.
4. A status of an item can never be null or None, if it is then it's an anomaly and item rows to be considered as fake orders and should not be kept in final data.
5. There must be no duplication in final data.

Directions to solve the problem:

- You'll be learning how to CREATE a table in database with pre-defined schema.
- Will be figuring out how in the best way you can remove anomalies from the data.
- How to read the data with pandas or pyspark.

Steps and Code:

In this section, I will outline the steps taken to complete the ETL process on the sales data. This includes data extraction, data transformation to handle anomalies, and loading the cleaned data into a PostgreSQL database. Each step is accompanied by the corresponding Python code

Step 1: Setup

- Install the necessary libraries if they are not already installed:

Code:

```
pip install pandas sqlalchemy psycopg2
```

Step 2: Extract Data

- Read the data from the CSV file

Code:

```python
import pandas as pd

# Read the CSV file
file_path = 'dataset.csv'
sales_data = pd.read_csv(file_path)

# Display the first few rows of the dataframe
print(sales_data.head())
```

Step 3: Transform Data

- Perform the required data transformations:

1. Ensure **order_id** is an integer:

Code:

```python
# Ensure order_id is an integer
sales_data['order_id'] = pd.to_numeric(sales_data['order_id'],
errors='coerce').dropna().astype(int)
```

2. Remove rows where **product_id** is 0:

Code:

```
# Remove rows where product_id is 0
sales_data = sales_data[sales_data['product_id'] != 0]
```

3. Ensure **order_id** is an integer:

Code:

```
# Cap prices at 1500 Rs
sales_data['amount'] = sales_data['amount'].apply(lambda x: 1500.0 if x > 1500 else
x)
```

4. Cap **amount** at 1500 Rs:

Code:

```
# Convert 'amount' column to float
sales_data['amount'] = sales_data['amount'].astype(float)

# Cap prices at 1500 Rs
sales_data['amount'] = sales_data['amount'].apply(lambda x: 1500.0 if x > 1500 else
x)
```

5. Remove rows where **status** is null:

Code:

```
# Remove rows where status is null
sales_data = sales_data[sales_data['status'].notnull()]
```

6. Remove duplicates:

Code:

```
# Remove duplicates
sales_data = sales_data.drop_duplicates()
```

Step 4: Load Data

- Load the cleaned data into a PostgreSQL database using **sqlalchemy** and **psycopg2**:

Code:

```
from sqlalchemy import create_engine

# Create a connection to the PostgreSQL database
engine =
create_engine('postgresql+psycopg2://postgres:password@localhost:5432/SalesManagemen
t')

# Define the table schema
table_name = 'sales_data_cleaned'
sales_data.to_sql(table_name, engine, schema='sales', if_exists='replace',
index=False)

print(f"Data successfully loaded into the table '{table_name}' in the PostgreSQL
database.")
```

Conclusion:

This task demonstrated the ETL process using Python, focusing on data extraction, transformation to remove anomalies, and loading into a PostgreSQL database. The skills acquired through this task are essential for advanced data engineering and analytics..