Python Assignment

Monday, May 27, 2024
6:33 PM

Question 1:
You are provided with two files containing sales data from two different regions. Your task is to create a Python script that:

1. Extracts data from these above files.
2. Transforms the data according to the specified business rules.
3. Loads the transformed data into a database of your choice (SQLite preferred).
4. Writes SQL queries to validate the data in database.

Order Region A:

- order_region_a.csv (sharepoint.com)
- Password: order_region_a

Order Region B:

- order_region_b.csv (sharepoint.com)
- Password: order_region_b

Schema:

`OrderId`: Order's id

`OrderItemId`: Item ids of the purchase orders.

`QuantityOrdered`: Number of items Ordered for an order.

`ItemPrice`: Price of each item in INR.

`PromotionDiscount`: Discount of an Order.

Business Rules:

1. Combine the data from both regions into a single table.
2. Add a column `total_sales` which is calculated as `QuantityOrdered * ItemPrice`.
3. Add a column region to identify the region of the sales record (A or B).
4. Ensure that there are no duplicate entries based on `OrderId`.
5. Add a new column `net_sale`, calculated as `total_sales - PromotionDiscount`.
6. Exclude orders where the total sales amount is negative or zero after applying discounts.
7. Load the transformed data into a the database of your choice.

Tasks:

1. Extract Data:
    a. Write a Python function to read data from the provided files.
2. Transform Data:
    a. Implement Python transformations based on the business rules mentioned above.
    b. Ensure data is cleaned, and all business rules are applied (including removal of duplicates and filtering based on net_sales).
3. Load Data:
    a. Create a table sales_data.
    b. Write a function to load the transformed data into the database.
4. Write SQL Queries and Python functions to validate data :
    a. Count the total number of records.

b. Find the total sales amount by region.
c. Find the average sales amount per transaction.
d. Ensure there are no duplicate `OrderId` values.

Submission:

- Commit the code and SQL queries to github public repository and share the link.
- A README file explaining,
  - How to run the program including DB Setup if there are any.
  - Any assumptions or decisions made during the implementation.

Question 2:

You will create an API that interacts with an external API (JokeAPI), processes the data, and stores it in a database.

Requirements:

1. Framework: You can use Django REST Framework, Flask, or FastAPI to build your API.
2. External API: Use the JokeAPI (https://sv443.net/jokeapi/v2/) to fetch jokes.
3. Database: Store the fetched jokes in a database of your choice.

Task Details:

You need to build an API endpoint that performs the following operations:

1. Calls the JokeAPI to fetch a minimum of 100 jokes.
2. Extracts and processes the following columns from the fetched jokes:
   a. category
   b. type
   c. joke (for "single" type) or setup and delivery (for "twopart" type)
   d. flags.nsfw
   e. flags.political
   f. flags.sexist
   g. safe
   h. lang
3. Stores the processed data in a database table.

Submission:

- Source Code: Include all relevant source code files in your github public repository.
- README File: Provide instructions on how to set up and run your project.