Ecommerce Data Storage

Introduction

Let's assume that you have to keep the record of the customers and vendors along with the different products available for purchase and the orders placed by different customers.

The focus of the project will be to learn data modeling and implement various functionality based on that. For database storage you will be accessing MySQL, information related to MySQL and several commands are discussed in weekly notebooks. You can refer to it to gain some hints on functionality.

Housekeeping points

- This is a minimal example and may not follow some standard practices.
- We focus on the main flow, and not much error handling.
- To avoid handling command-line arguments, config file paths are hard-coded in the source files - not a general practice.

Program Organization

The simple program is structured in various layers.

1. **Data model**:
    a. ***users* table**: This stores information about the users. It has the following fields
        user_id (String),
        user_email (String),
        user_name (String),
        user_password(String),
        user_address(String)
        is_vendor(int)
    A user is segregated either as a customer or a vendor based on the column '**is_vendor**' . When **is_vendor** is 1, the row represents the details of a vendor and when it is 0, then it represents the details of a customer.
    User_address field will be null in case of vendors.

b. ***orders* table**: This stores the actual orders data. It contains the following fields
order_id (Int),
total_value (Float),
customer_id(String, Foriegn key from users table),
vendor_id(String, Foreign key from users table)
order_quantity (Int),
reward_point (Int)

c. ***products* table**: This stores the product information. It contains the following fields
product_id(String),
product_name(String),
product_description(String),
vendor_id(String, Foriegn key from users table),
product_price(Float),
emi_available(String)

2. **config:** This folder contains the information, in the form of csv files, which is used to populate the tables required in this application.
3. **src/setup.py:** This program accesses all the csv files, reads the data and populates the required tables with each csv file. Following are more details on these files...
    a. Accesses the details available in users.csv and populates the *users* table. To keep the scope minimum we have 15 users in this file.

    b. Accesses the details available in products.csv and populates the *products* table. Here again to keep the scope minimum we have 10 types of products.

    c. Accesses the details available in orders.csv file and populates the *orders* table. To keep the scope minimum, we have 45 orders in the file.

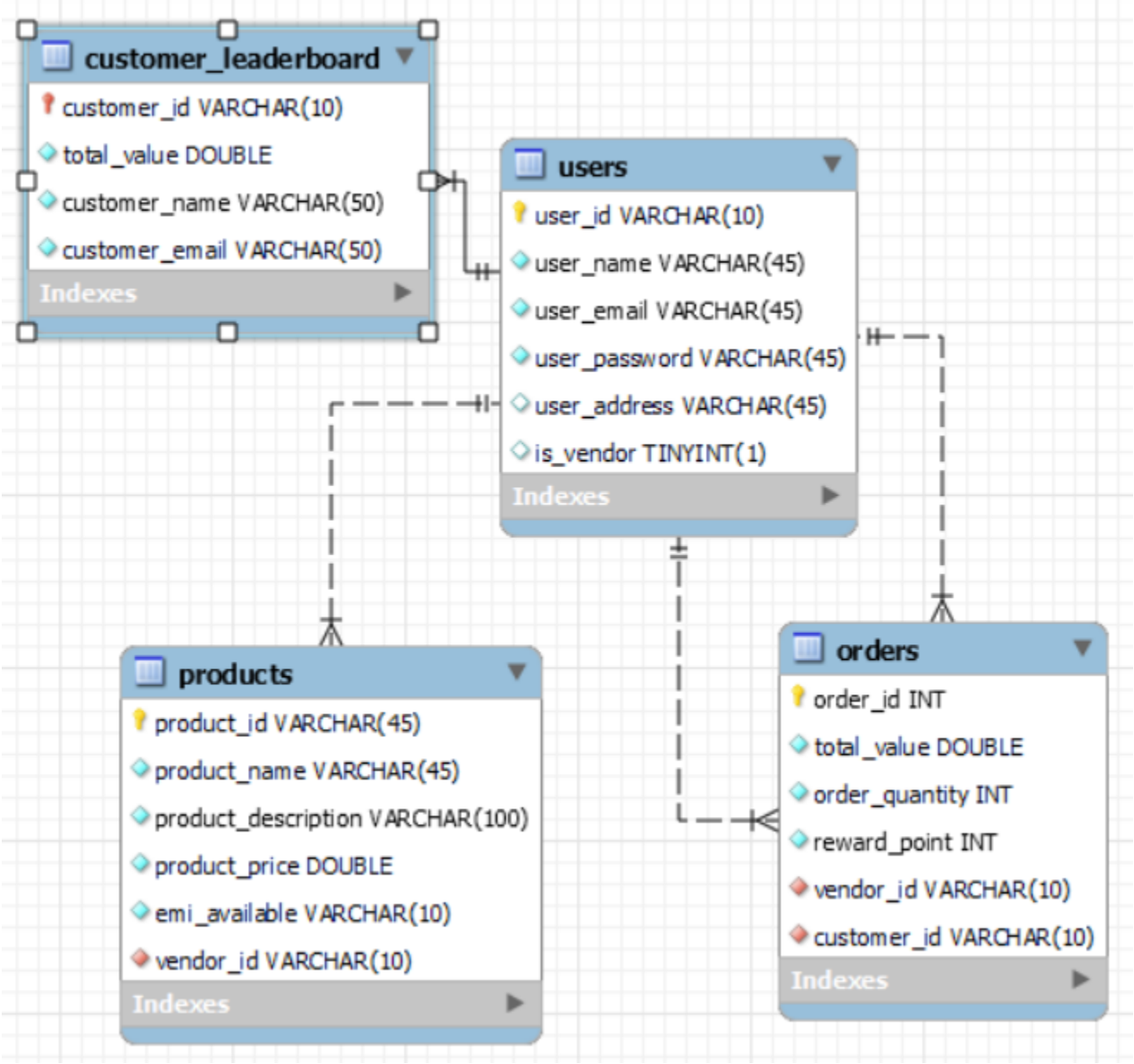
4. **src/database.py**: This file hosts a number of functions which provide functionality of connecting to a database and select (fetching) and insert operation on the tables.

5. **src/main.py**: In this file you are required to implement the logic (adding functions) to solve different problems defined in the problem statement.
This file contains examples of connecting to a database and using different models to store and retrieve data.

Problem Statement

In this project you are expected to write a Python program to implement the functionality of connecting to a database and perform the basic CRUD operations.

Following is an ER diagram of the database "**ecommerce_record**" we have designed



1. (**Easy**) - Creating schema and tables. Use either Python program or MySQL Workbench UI for this activity.
    a. Create a database named **ecommerce_record.**
    b. Create the required tables, mentioned in the ER diagram.

Do ensure that the tables and their respective columns have the same name as mentioned.
Also make a note on creating the required primary and foreign keys

2. (**Medium**) - Implement functionality to implement insert and select operations
    a. In the setup.py file, you have to write the correct insert method to perform insert operation on the ***users, products*** and ***orders*** tables with the data available in the relevant csv file. The placeholders where you have to write the code have been marked.
    b. After the above step, write 5 insert operations on the ***orders*** table. Each insert operation should be for a different customer.
    c. Write a read operation that will fetch all the records of the orders table and print the data on the console.

3. **(Hard)** Performing analysis data and creating a report
    a. From the **orders** table write a query to find the maximum and minimum order value. Print the values on the console.
    b. Fetch and print the orders with value (***total_value***) greater than the average value (***total_value***) of all the orders in the ***orders*** table
    c. Create a new table in the name of customer_leaderboard. It should have the following fields
        customer_id (string)
        total_value (float),
        customer_name (string)
        customer_email (string)
    Insert one row for each registered customer who has made a purchase. And fill the (***total_value***) of this table, with the highest purchase value (***total_value***) of the **orders** table for a customer. In case a customer has not made any purchase, then do not add a record of the customer.

Evaluation Rubric

Total Project Points: **100**

- Basic compilation without errors (**10%**) : **10 Points**
- Correctness:
  Correctness of implementation
    - Problem statement - point 1.a (**10%**) : **10 Points**
    - Problem statement - point 1.b (**10%**) : **10 Points**
    - Problem statement - point 2.a (**20%**) : **20 Points**
    - Problem statement - point 2.b (**10%**) : **10 Points**
    - Problem statement - point 2.c (**10%**) : **10 Points**
    - Problem statement - point 3.a (**10%**) : **10 Points**
    - Problem statement - point 3.b (**10%**) : **10 Points**
    - Problem statement - point 3.c (**10%**) : **10 Points**

Program Instructions

1. Download the zipped folder named **C03-Project01-02-Ecommerce Data Storage.zip**, unzip it on your local machine, and save it. Go into the directory named **C03-Project01-02-Ecommerce Data Storage.**

2. Make sure you have Python 3.6 or higher installed. At your command prompt, run:

   ```
   $ python --version
   Python 3.7.3
   ```
   If not installed, install the latest available version of Python 3.

3. You will needMySQL for this project. Do install **MySQL** and **MySQL Workbench** on your system. You will need this to verify your changes you are making in this project

4. First task will be to write the missing functionality in the file **setup.py**. This will add the required data in the tables of the database.
   Do ensure, while executing the file **setup.py**, that you do not get any error.
   Verify your changes, that data has been inserted in the tables, using MySQL Workbench

5. After the above step, examine and run **main.py**. This runs various simple calls. As you write solutions to the problems, you'll be frequently modifying and running this file. You can comment or modify the initial code as needed.

   ```
   $ python3 main.py (On many Linux platforms)
                        OR
   $ python main.py  (On Windows platforms)
   ```

   In any case, one of these two commands should work.

6. Alternatively, you can install a popular Python **IDE**, such as **PyCharm** or **Visual Studio Code**, and select a command to build the project from there.

7. Once the program is ready to submit, zip the parent folder **C03-Project01-02-Ecommerce Data Storage**, and upload the new zip file as **C03-Project01-02-Ecommerce Data Storage.zip**.
   It is now ready for evaluation.