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 modelling 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. *vendors* table: This stores information about the vendors. It has vendor_id (String), vendor_email (String), and vendor_name (String), vendor_password fields in each record.
    b. *customers* table: This stores information about the customers. It has customer_id (String), customer_email(String), password (String) and address (String) fields in each record.
    c. *orders* table: This stores the actual orders data. It contains order_id (Int), total_value (Float), customer_id(String, Foriegn key from customers table), vendor_id(String, foreign key from vendors table) and order_quantity (Int), reward_point (Int) fields in each record.
    d. *Items* table: This stores the product information. It contains product_id(String), product_name(String), product_description(String), vendor_id(String, Foriegn key from vendors table), product_price(Float), emi_available(String) fields in each record.

2. **src/setup.py:** This directory has the customer related information available in the csv file.
    a. Accesses the customer information and details available in customers.csv and populates the *customers* table. To keep the scope minimum we are having information of only 10 customers in the file.
    b. Accesses the vendor information and details available in vendors.csv and populates the data in the *vendors* table. To keep the scope minimum assume that there are only 5 vendors available.
    c. Access the items/products information items.csv and populate those in the *items* table. Here again to keep the scope minimum we will have 10 types of products.

d. Access the details available in the orders data and save it in the orders.csv file and then it will be populated in the *orders* table. To keep the scope minimum, currently we have only 45 orders available in the csv file.

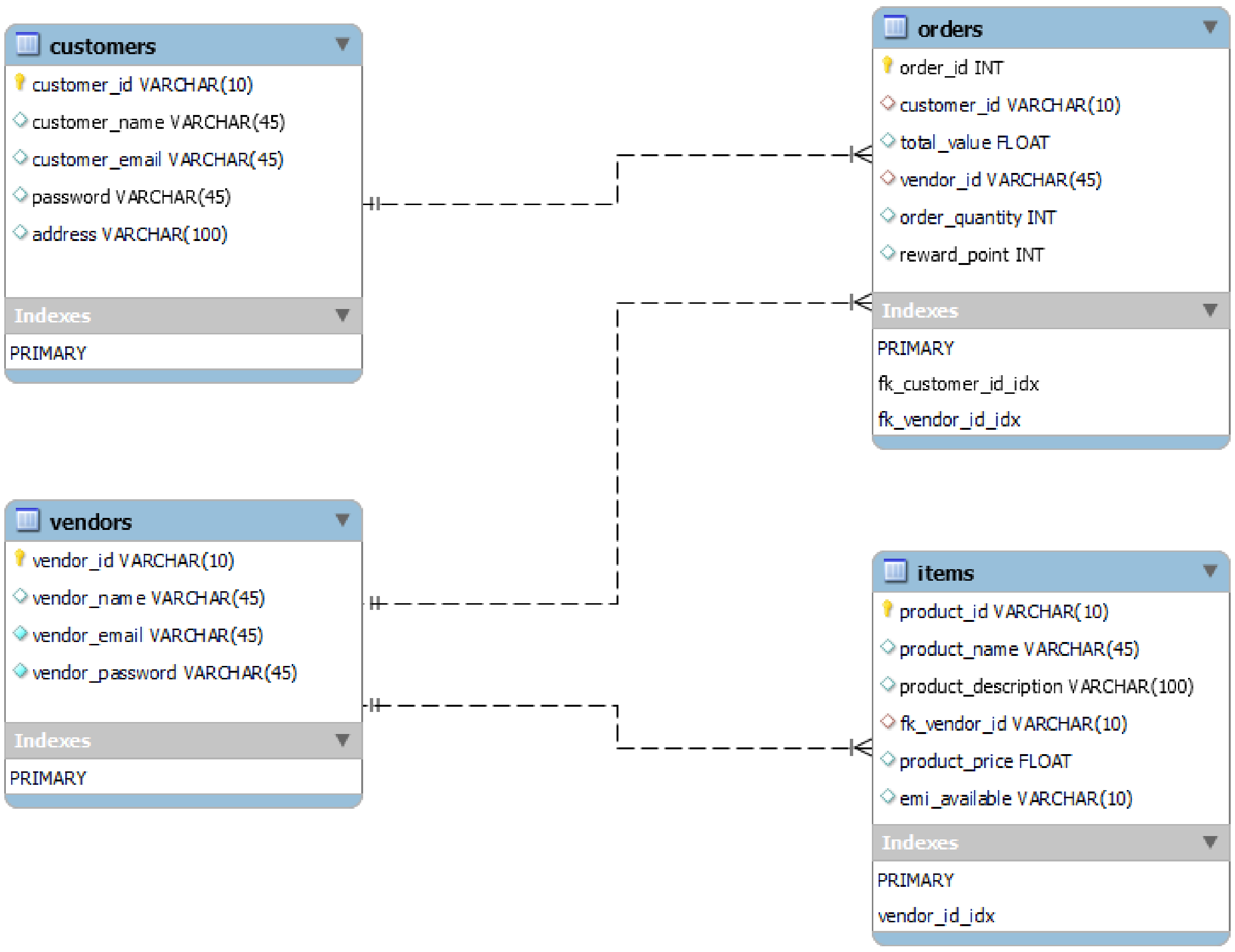
3. **src/database.py**: This file hosts the Database class which provides base access to the MySQL database. It has the connection information stored as static variables. It provides functionality for connecting, fetching, and inserting a document.

4. **src/main.py**: Here, you need to add the functions required for the various problems. Initially, it contains examples of how to connect and use the various models to store and retrieve data.

Problem Statement

In this project you are supposed to perform creation, insertion and deletion of the data using a python program. You are supposed to create a database class and write relevant methods in it to perform the basic CRUD operation on the data.

Please find the ER diagram below that will help you in designing the database tables.



1. (**Easy**) Creating the schema and required tables using python program or using MySQL workbench UI
    a. Create/access a schema named **ecommerce_record.** You can use a python program or MySQL UI to create the schema name. **Make sure that before creating the schema there are no duplicate schemas available in the database.**
    b. Create the required tables mentioned in the above ER diagram, you can use python program or MySQL UI to create the tables. The tables should have the same column name and primary key and foreign keys.

2. **(Medium**) Functionality to perform insert and read operation on the table created in the above task using python.
    a. Write an insert method to perform insert operation on the customer, vendor, items and orders tables with the data available in the relevant csv file. To perform this you will be making changes in the setup.py, because that is where we have accessed all the files.
    b. Once the data is available in all the tables. Write logic to perform at least 5 insert operations in the orders table. These insert operations should be for different customers.
    c. Write a read method that should fetch all the records that are available in the orders table and print it on the console.

3. **(Hard)** Performing analysis on the available data and creating a report
    a. Use the data fetched in the previous task to find the maximum and minimum value order from the orders table.
    b. Find and print all the order details with *total_value* more than the average order value ordered from all customers.
    c. Create a new table named *customer_leaderboard(customer_id, order_id, total_value, customer_name, customer_email)* and insert the highest ordered purchase for each of the registered customers. If there are no orders you can ignore that 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 (**25%**) : **25 Points**
    - Problem statement - point 2.b (**10%**) : **10 Points**
    - Problem statement - point 2.c (**20%**) : **20 Points**
    - Problem statement - point 3.a (**5%**) : **5 Points**
    - Problem statement - point 3.b (**5%**) : **5 Points**
    - Problem statement - point 3.c (**5%**) : **5 Points**

Program Instructions

1. Download the zipped folder named **M01-Project03-Ecommerce-Data-Storage.zip**, unzip it on your local machine, and save it. Go into the directory named **M01-Project03-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 be needing MySQL installed in your system to verify the changes that you are making. Please install **MySQL** and **MySQL Workbench**.

4. First you need to write functionality in setup.py, to fill the data in the created database. Make sure that while running setup.py you are not getting any error. You can check the MySQL workbench to verify if the data is available in the database.

5. You can now examine and run **main.py**. This will currently run various simple calls. As you solve 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 could 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 **M01-Project03-Ecommerce-Data-Storage**, and upload the new zip file as **M01-Project03-Ecommerce-Data-Storage.zip**. It is now ready for evaluation.