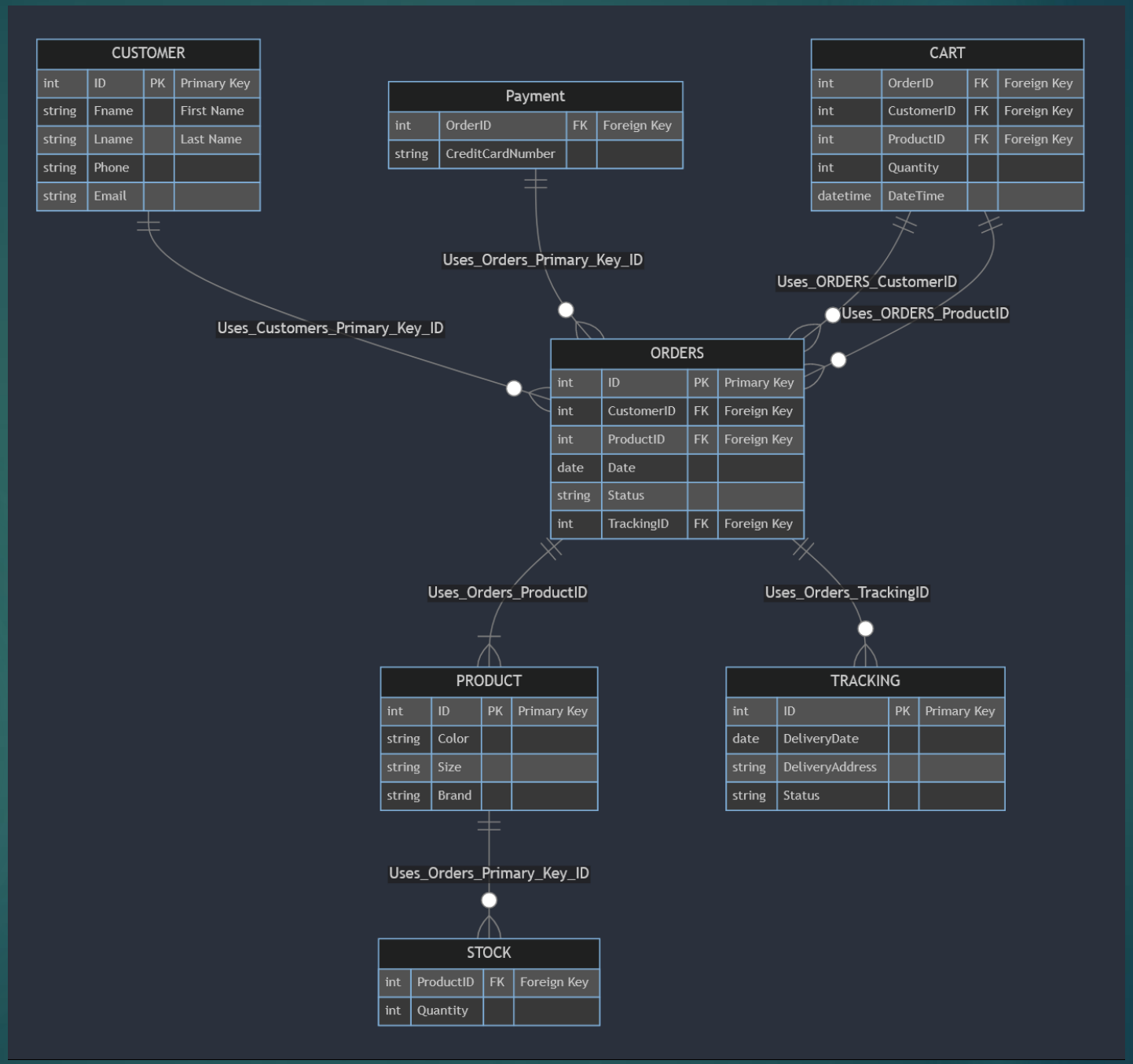
GROUP 6

BEN MISINGO
CALVIN SCHMEICHEL
MEHTAB KHALSA
MOHAMED T YUSUF
AARON

Problem statement

- As online shopping advances, the need for reliability, quality and quickness remains constant; customers constantly need to get quality products worth their money. We realized some aspects of clothing stores could improve, particularly t-shirts. T-shirt fans like us want to be able to buy different brand T-shirts from one shop instead of bouncing from shop to shop. Hence why we are creating a system; to make it easy for customers to find the T-shirts they want quickly, buy them easily and wear them happily.

Schema


CUSTOMER
int ID PK Primary Key
string Fname First Name
string Lname Last Name
string Phone
string Email
Payment
int OrderID FK Foreign Key
string CreditCardNumber
CART
int OrderID FK Foreign Key
int CustomerID FK Foreign Key
int ProductID FK Foreign Key
int Quantity
datetime DateTime
Uses_Orders_Primary_Key_ID
Uses_ORDERS_CustomerID
Uses_ORDERS_ProductID
Uses_Customers_Primary_Key_ID
ORDERS
int ID PK Primary Key
int CustomerID FK Foreign Key
int ProductID FK Foreign Key
date Date
string Status
int TrackingID FK Foreign Key
Uses_Orders_ProductID
Uses_Orders_TrackingID
PRODUCT
int ID PK Primary Key
string Color
string Size
string Brand
TRACKING
int ID PK Primary Key
date DeliveryDate
string DeliveryAddress
string Status
Uses_Orders_Primary_Key_ID
STOCK
int ProductID FK Foreign Key
int Quantity

Functionalities

The code consists of the following key components:

handle_purchase() function: This function is called when the user clicks the "Purchase" button. It retrieves the user's email, T-shirt color, size, and quantity from the respective input fields. It then generates a mock receipt string using f-strings and prints it to the console. Finally, it clears the email input field for the next entry.

Main window creation: The main window is created using the tk.Tk() method. The title of the window is set to "T-Shirt Purchase" using the title() method.

User input fields: The code creates several input fields for the user to enter their personal information, including email, first name, last name, and phone number. Each input field is created using the tk.Entry() method, and a label is associated with each input field using the tk.Label() method.

T-shirt color selection: The user can choose the desired T-shirt color by selecting one of the available options (Red, Blue, Green, Yellow). Radio buttons are created for each color option using the tk.Radiobutton() method. The selected color is stored in the color_var variable, which is of type tk.StringVar().

T-shirt size selection: Similar to the color selection, the user can choose the T-shirt size by selecting one of the available options (Large, Medium, Small). Radio buttons are created for each size option using the tk.Radiobutton() method. The selected size is stored in the size_var variable, which is of type tk.StringVar().

Functionalities(continued)

Quantity selection: The user can choose the quantity of T-shirts they want to purchase by selecting one of the available options (1, 5, 10). Radio buttons are created for each quantity option using the tk.Radiobutton() method. The selected quantity is stored in the quantity_var variable, which is of type tk.StringVar().

Purchase button: The "Purchase" button is created using the tk.Button() method. When clicked, it calls the handle_purchase() function.

GUI event loop: The code enters the GUI event loop using the mainloop() method of the main window. This ensures that the GUI remains responsive and can handle user interactions.

System functions

Code Documentation

- **Importing the required library:**
- The `mysql.connector` library is imported to establish a connection to the MySQL database.
- Establishing a connection to the database: The mysql.connector.connect() method is used to establish a connection to the MySQL database. The host, port, username, password, and database name are provided as parameters.
- **Creating a cursor object:** A cursor object is created using the db.cursor() method to execute SQL queries.
- **Defining the printsql() function:** The printsql() function is defined to iterate through the query results and print them.
- **Dropping the existing database (if exists):** The mycursor.execute() method is used to execute the SQL query DROP DATABASE IF EXISTS myStore;. This query drops the existing database named "myStore" if it exists.
- **Creating a new database:** The mycursor.execute() method is used to execute the SQL query CREATE DATABASE myStore;. This query creates a new database named "myStore".
- **Using the newly created database:** The mycursor.execute() method is used to execute the SQL query USE myStore;. This query sets the newly created "myStore" database as the current database.

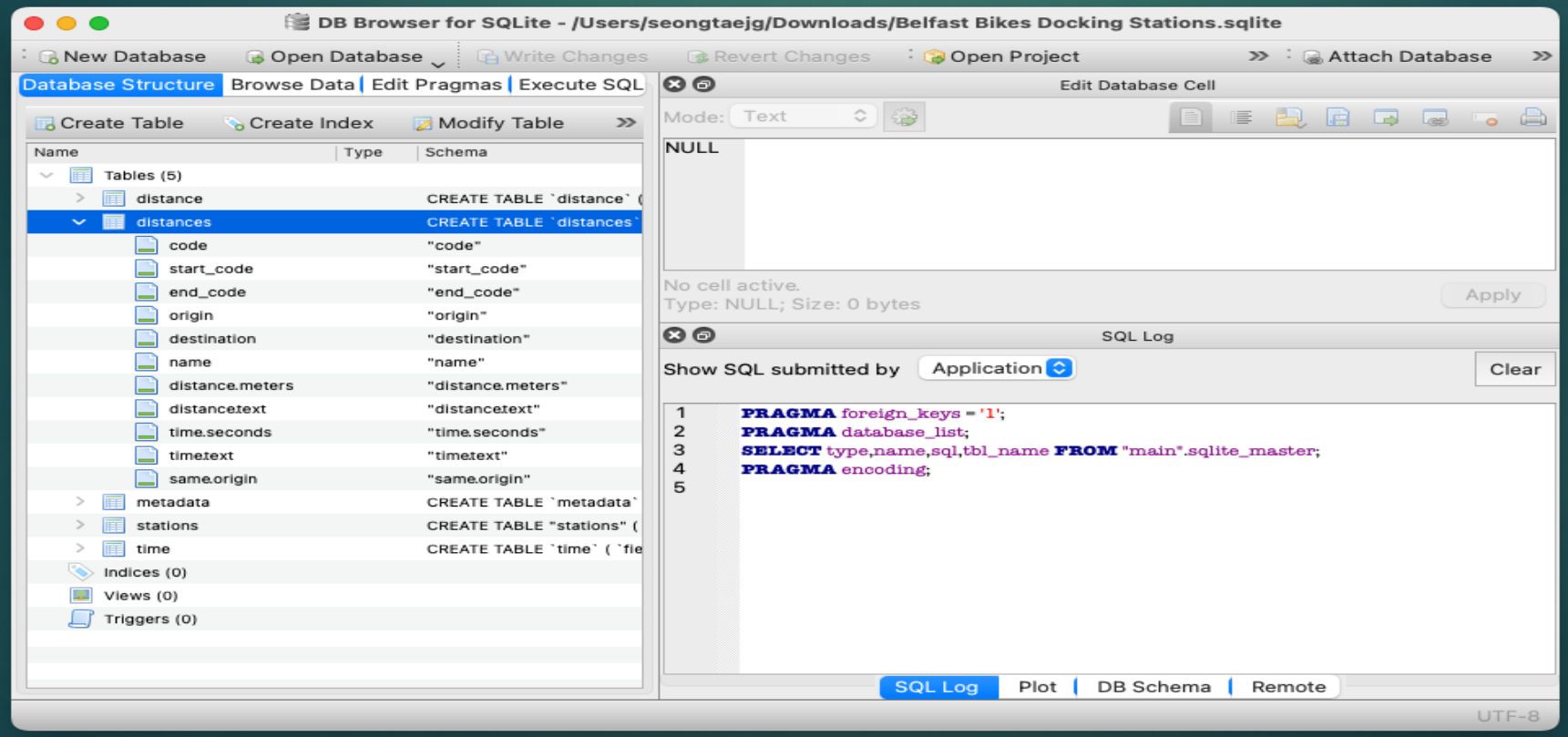
DBMS Tables

- **Creating tables for the store management system:**
  - *The mycursor.execute() method is used to execute a series of SQL queries to create the following tables:*
    - **Customers table:** Stores information about customers, including their ID, first name, last name, phone number, and email address.
    - **Products table:** Stores details about products, such as the ID, color, size, and brand.
    - **Tracking table:** Tracks the delivery status of orders, including the delivery date, delivery address, status, and a unique tracking ID.
    - **Orders table:** Manages the orders placed by customers, including the order ID, customer ID, product ID, order date, status, and tracking ID.
    - **Stock table:** Keeps track of the stock quantity for each product.
    - **Payment table:** Handles payment information for orders, storing the order ID and credit card number.
    - **Cart table:** Tracks the items added to the cart by customers, including the customer ID, product ID, and quantity.

Tables continued

- **Inserting sample data into the Products table:** The mycursor.execute() method is used to execute an SQL query to insert sample data into the Products table. The INSERT INTO statement is used to specify the columns (Color, Size, Brand) and the values for each row.
- **Inserting sample data into the Customers table:** The mycursor.execute() method is used to execute an SQL query to insert sample data into the Customers table. The INSERT INTO statement is used to specify the columns (Fname, Lname, Phone, Email) and the values for each row.
- **Retrieving data from the Customers table:** The mycursor.execute() method is used to execute an SQL query to select all data from the Customers table. The query is then passed to the printsql() function to print the query results.
- **Retrieving data from the Products table:** The mycursor.execute() method is used to execute an SQL query to select all data from the Products table. The query is then passed to the printsql() function to print the query results.

DB Browser for SQLite



What it is?

- *DB Browser for SQLite* (DB4S) is a high quality, visual, open-source tool to create, design, and edit database files compatible with SQLite.

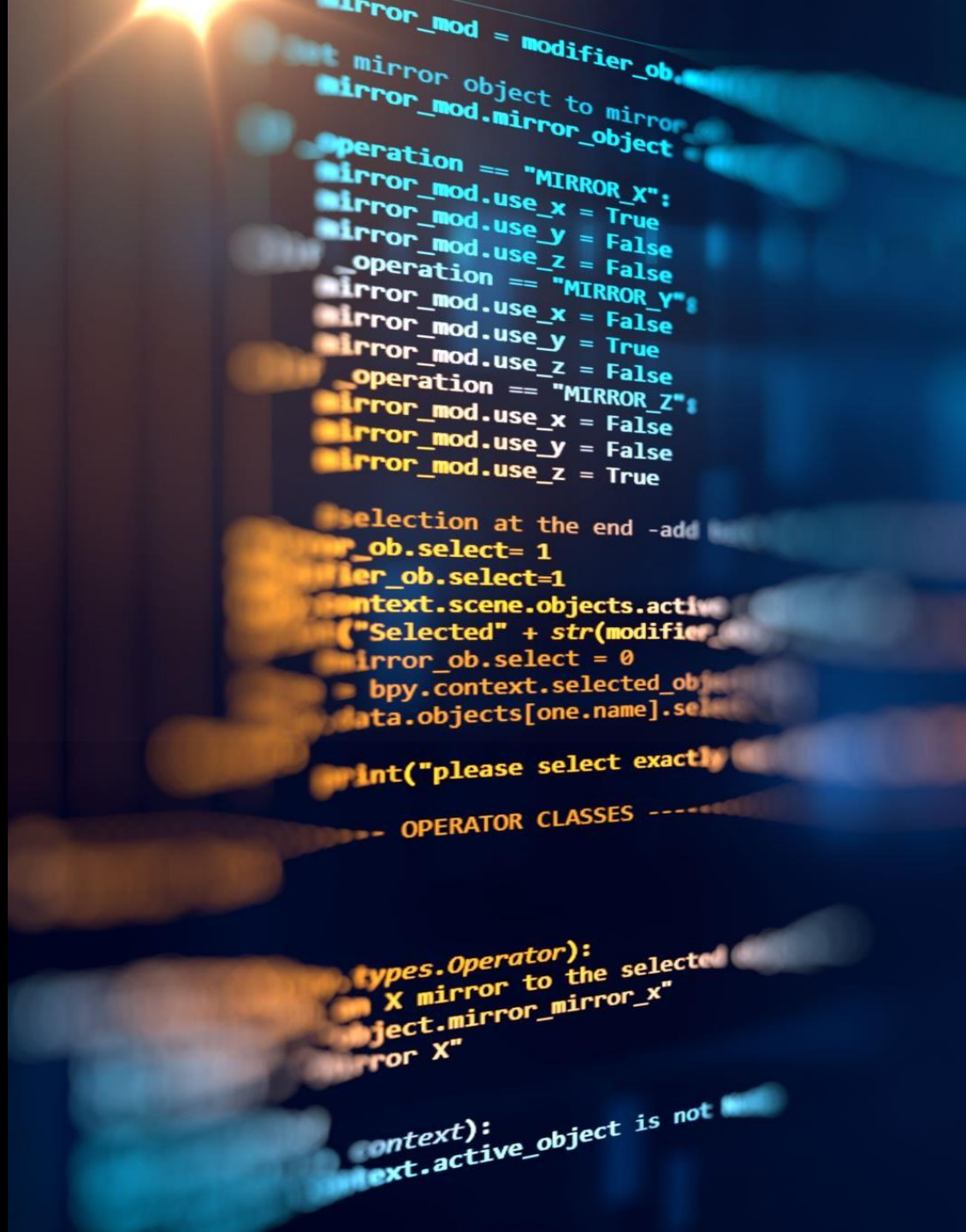
- DB4S is for users and developers who want to create, search, and edit databases. DB4S uses a familiar spreadsheet-like interface, so complicated SQL commands do not have to be learned.

Controls and wizards are available for users to:

- Create and compact database files
- Create, define, modify and delete tables
- Create, define, and delete indexes
- Browse, edit, add, and delete records
- Search records
- Import and export records as text
- Import and export tables from/to CSV files
- Import and export databases from/to SQL dump files
- Issue SQL queries and inspect the results
- Examine a log of all SQL commands issued by the application
- Plot simple graphs based on table or query data

Python Script Overview

- Python Script Overview
- The provided code sets up the necessary database structure for a store management system. It creates the required tables and inserts sample data for testing purposes. The code can serve as a foundation for building a complete store management application.
- Please note that additional code and user interface implementation may be required to utilize the database effectively in a real-world scenario.
- This code is written in Python and utilizes the tkinter library to create a graphical user interface (GUI) for handling T-shirt purchases. The code allows users to enter their personal information, select the desired T-shirt color, size, and quantity, and then generate a mock receipt.



GUI(Receipt)

```
Database Generated
Generate Customer Entry
(1, 'John', 'Doe', '123-456-7890', 'john.doee@example.com')
(2, 'John', 'Doe', '123-456-7890', 'john.doe@example.com')
(3, 'Calvin', 'Schmeichel', '3203080121', 'CalvinSchmeichel@SCSU.com\n')
Generate Tracking Entry
(1, datetime.date(2023, 12, 7), '720 4th Ave S, St Cloud, MN 56301\n', 'In Transit', 'SLMI98YFA3')
Generate Order Entry
4
(1, 3, 4, datetime.datetime(2023, 12, 7, 0, 0), 'Processing', 1)
Generate Cart Entry
(3, 4, 5, '2023-11-30')
Generate Payment Entry
(1, 374245455400126)
=========================================================================
Making Receipt
=========================================================================
Thank you for shopping with us! Here is your Receipt!
Calvin Schmeichel bought 5 Large Blue T-Shirt(s) today (2023-11-30)
This purchase was made with 374245455400126 and plans to arrive at 720 4th Ave S, St Cloud, MN 56301
 on 2023-12-07
```

Video Demo