## **SQL CODE:**

CREATE DATABASE ecom

USE ecom

CREATE TABLE customer(

    cid INT PRIMARY KEY,

    cname VARCHAR(50),

    address VARCHAR(150),

    mob VARCHAR(15)

)

CREATE TABLE product(

    pid INT PRIMARY KEY,

    pname varchar(50),

    descri VARCHAR(200),

    price FLOAT

)

CREATE TABLE orders(

    oid INT PRIMARY KEY,

    date VARCHAR(25),

    cid INT,

    pid INT,

    qty INT,

    price FLOAT,

    total\_amount FLOAT,

    FOREIGN KEY(cid) REFERENCES customer(cid),

    FOREIGN KEY(pid) REFERENCES product(pid)

)

SELECT \* FROM customer

SELECT \* FROM product

SELECT \* FROM orders

## **Customer class (customer.py):**

class customer:

    def \_\_init\_\_(self, cid, cname, address, mob):

        self.cid = cid

        self.cname = cname

        self.address = address

        self.mob = mob

    def get\_customer(self):

        return (self.cid, self.cname, self.address, self.mob)

## **Products class(product.py):**

class product:

    def \_\_init\_\_(self, pid, pname, desc, price):

        self.pid = pid

        self.pname = pname

        self.desc = desc

        self.price = price

    def get\_product(self):

        return (self.pid, self.pname, self.desc, self.price)

## **Orders class(orders.py):**

import datetime

class order:

    def \_\_init\_\_(self, oid, cid, pid, qty, price):

        self.oid = oid

        now = datetime.datetime.now()

        self.date = now.strftime("%d-%m-%Y")

        self.cid = cid

        self.pid = pid

        self.qty = qty

        self.price = price

        self.total\_amt = self.qty \* self.price

    def make\_order(self):

        return (self.oid, self.date, self.cid, self.pid, self.qty, self.price, self.total\_amt)

## **Main class file(main\_module.py):**

from entity.customer import customer

from entity.product import product

from entity.order import order

import pyodbc

def one\_way\_command(conn, query, values):

    try:

        cur = conn.cursor()

        cur.execute(query, values)

        cur.commit()

        cur.close()

        print("Query Excuted")

    except pyodbc.Error as e:

        print(e)

def two\_way\_command(conn, query):

    try:

        cur = conn.cursor()

        cur.execute(query)

        result = cur.fetchall()

        return result

    except pyodbc.Error as e:

        print(e)

class Menu:

    def \_\_init\_\_(self):

        try:

            self.conn = pyodbc.connect(

                "DRIVER={ODBC Driver 17 for SQL Server};"

                "SERVER=THARUN\SQLEXPRESS;"

                "DATABASE=ecom;"

                "Trusted\_Connection=yes;"

            )

        except pyodbc.Error as e:

            print(e)

    def Add\_customer(self, cid, cname, address, mob):

        new\_customer = customer(cid, cname, address, mob)

        values = new\_customer.get\_customer()

        query = "INSERT INTO customer VALUES (?, ?, ?, ?)"

        one\_way\_command(self.conn, query, values)

    def Add\_product(self, pid, pname, desc, price):

        new\_product = product(pid, pname, desc, price)

        values = new\_product.get\_product()

        query = "INSERT INTO product VALUES (?, ?, ?, ?)"

        one\_way\_command(self.conn, query, values)

    def Order\_product(self, oid, cid, pid, qty):

        product\_price = two\_way\_command(self.conn, f"SELECT price FROM product WHERE pid = {pid}")

        price = product\_price[0][0]

        new\_order = order(oid, cid, pid, qty, price)

        values = new\_order.make\_order()

        query = "INSERT INTO orders VALUES (?, ?, ?, ?, ?, ?, ?)"

        one\_way\_command(self.conn, query, values)

    def display\_all\_order(self):

        query = "SELECT \* FROM orders"

        result = two\_way\_command(self.conn, query)

        print("------------------------------------------")

        for i in result:

            print(f"{i.oid}  {i.date}  {i.cid} {i.pid}  {i.qty}  {i.price}  {i.total\_amount}")

        print("------------------------------------------")

    def display\_order\_id(self):

        query = "SELECT TOP 1 oid FROM orders ORDER BY oid DESC"

        result = two\_way\_command(self.conn, query)

        print(f"Last order ID - {result[0][0]}")

    def display\_customer\_records(self):

        query = """SELECT c.cid, c.cname, sum(total\_amount) AS sum\_total FROM orders o

                INNER JOIN customer c

                ON c.cid = o.cid

                GROUP BY c.cid, c.cname"""

        result = two\_way\_command(self.conn, query)

        print("------------------------------")

        for i in result:

            print(f"{i.cid}  {i.cname}  {i.sum\_total}")

        print("------------------------------")

m1 = Menu()

status = "Online"

operations = ["1. Add customer", "2. Add product", "3. Add order", "4. Display all order", "5. Display order ID", "6. Display customer details", "7. Exit"]

for i in operations:

    print(i)

while status == "Online":

    response = int(input("Enter operations to perform: "))

    if response == 1:

        m1.Add\_customer(cid=int(input("Enter CID: ")), cname=input("Enter Name: "), address=input("Enter Location: "), mob=input("Enter number: "))

    elif response == 2:

        m1.Add\_product(pid=int(input("Enter PID: ")), pname=input("Enter Product name: "), desc=input("Enter Description: "), price=int(input("Enter price: ")))

    elif response == 3:

        m1.Order\_product(oid=int(input("Enter OrderID: ")), cid=int(input("Enter CustomerID: ")), pid=int(input("Enter ProductID: ")), qty=int(input("Enter Quantity: ")))

    elif response == 4:

        m1.display\_all\_order()

    elif response == 5:

        m1.display\_order\_id()

    elif response == 6:

        m1.display\_customer\_records()

    elif response == 7:

        status = "Offline"

        print("Logged Out!!")

    else:

        print("Invalid response!!")

        break