COSC 4301 – Database Theory and Practice
Lab 02

Utilizing the Ecommerce database from Lab 01:

Customers

customer_id, first_name, middle_initial, last_name, email, phone, address

Products

product_id, name, description, price, category_id, image_url, stock_quantity

Orders

order_id, customer_id, order_date, total_amount, status

Items

item_id, order_id , product_id, quantity, unit_price

Categories

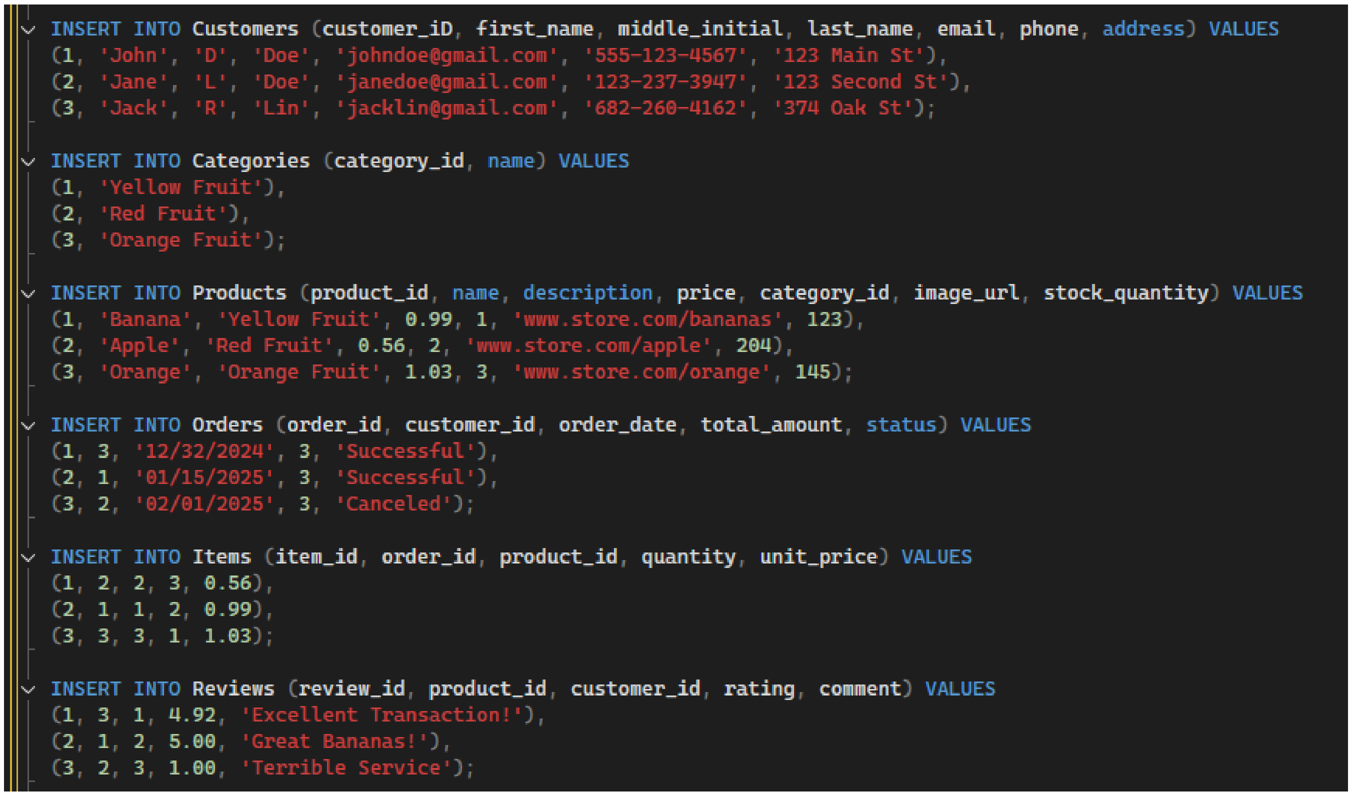
category_id, name

Reviews

review_id, product_id , customer_id, rating, comment

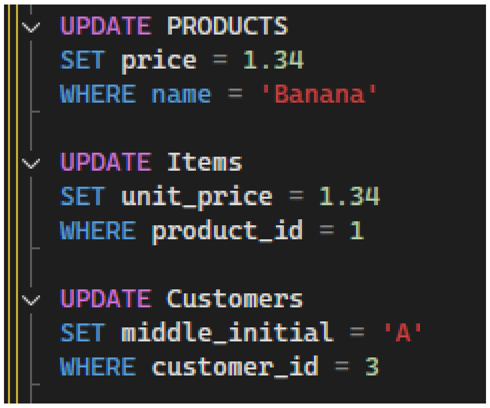
perform the following tasks:

1. Insert data into all the relations in the database (at least three tuples for each relation).



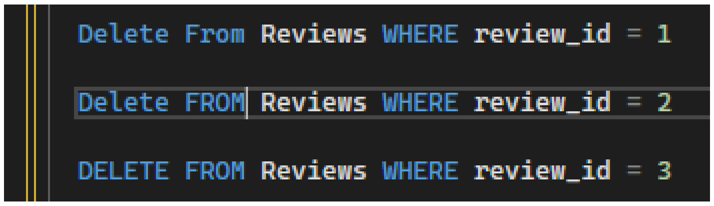
```
INSERT INTO Customers (customer_iD, first_name, middle_initial, last_name, email, phone, address) VALUES
(1, 'John', 'D', 'Doe', 'johndoe@gmail.com', '555-123-4567', '123 Main St'),
(2, 'Jane', 'L', 'Doe', 'janedoe@gmail.com', '123-237-3947', '123 Second St'),
(3, 'Jack', 'R', 'Lin', 'jacklin@gmail.com', '682-260-4162', '374 Oak St');

INSERT INTO Categories (category_id, name) VALUES
(1, 'Yellow Fruit'),
(2, 'Red Fruit'),
(3, 'Orange Fruit');

INSERT INTO Products (product_id, name, description, price, category_id, image_url, stock_quantity) VALUES
(1, 'Banana', 'Yellow Fruit', 0.99, 1, 'www.store.com/bananas', 123),
(2, 'Apple', 'Red Fruit', 0.56, 2, 'www.store.com/apple', 204),
(3, 'Orange', 'Orange Fruit', 1.03, 3, 'www.store.com/orange', 145);

INSERT INTO Orders (order_id, customer_id, order_date, total_amount, status) VALUES
(1, 3, '12/32/2024', 3, 'Successful'),
(2, 1, '01/15/2025', 3, 'Successful'),
(3, 2, '02/01/2025', 3, 'Canceled');

INSERT INTO Items (item_id, order_id, product_id, quantity, unit_price) VALUES
(1, 2, 2, 3, 0.56),
(2, 1, 1, 2, 0.99),
(3, 3, 3, 1, 1.03);

INSERT INTO Reviews (review_id, product_id, customer_id, rating, comment) VALUES
(1, 3, 1, 4.92, 'Excellent Transaction!'),
(2, 1, 2, 5.00, 'Great Bananas!'),
(3, 2, 3, 1.00, 'Terrible Service');
```

2. Update the three tuples from three different relations.



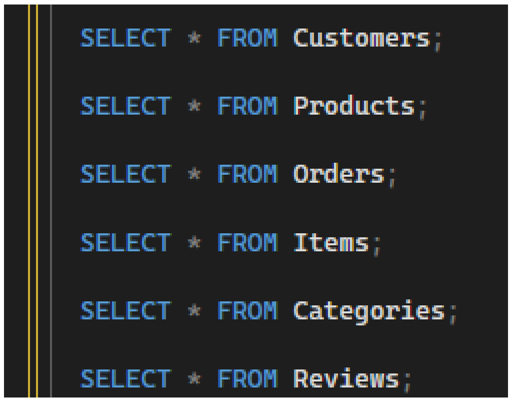
```
UPDATE PRODUCTS
SET price = 1.34
WHERE name = 'Banana'

UPDATE Items
SET unit_price = 1.34
WHERE product_id = 1

UPDATE Customers
SET middle_initial = 'A'
WHERE customer_id = 3
```

3. Perform three deletion operations.



```
Delete From Reviews WHERE review_id = 1

Delete FROM Reviews WHERE review_id = 2

DELETE FROM Reviews WHERE review_id = 3
```

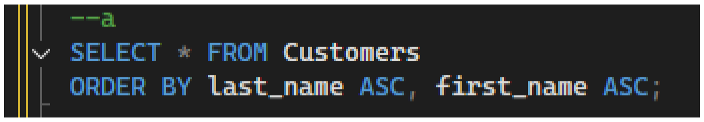
4. Retrieve all data for each relation.



```
SELECT * FROM Customers;

SELECT * FROM Products;

SELECT * FROM Orders;

SELECT * FROM Items;

SELECT * FROM Categories;

SELECT * FROM Reviews;
```

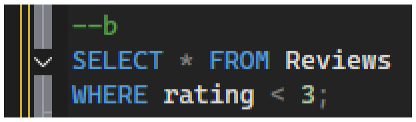
5. Define the SQL statement for the following queries:

   a. Retrieve all customers alphabetized.



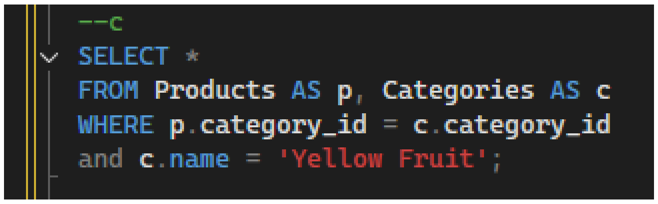
```
--a
SELECT * FROM Customers
ORDER BY last_name ASC, first_name ASC;
```

b. Retrieve all reviews with a rating less than three stars.



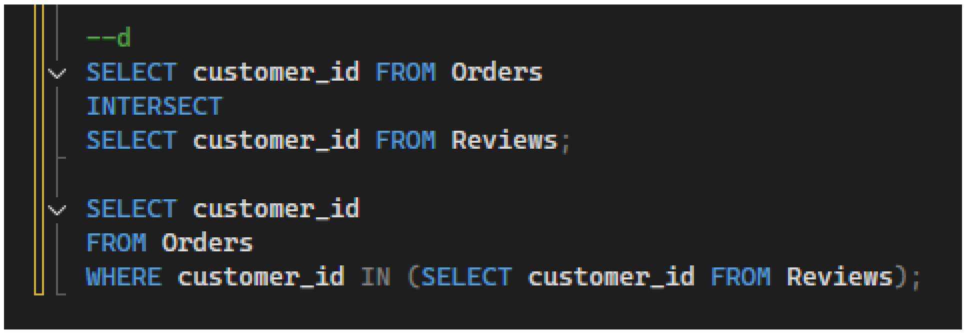
```sql
--b
SELECT * FROM Reviews
WHERE rating < 3;
```

c. Retrieve all products based on a specific category name.



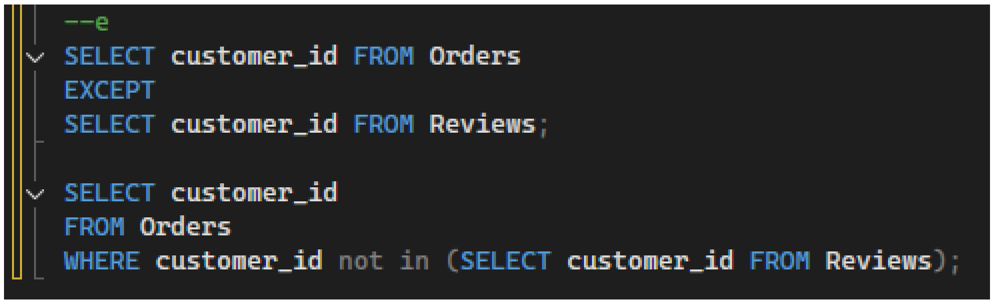
```sql
--c
SELECT *
FROM Products AS p, Categories AS c
WHERE p.category_id = c.category_id
and c.name = 'Yellow Fruit';
```

d. Develop two queries that produce the same relation using **intersect** and **in**.



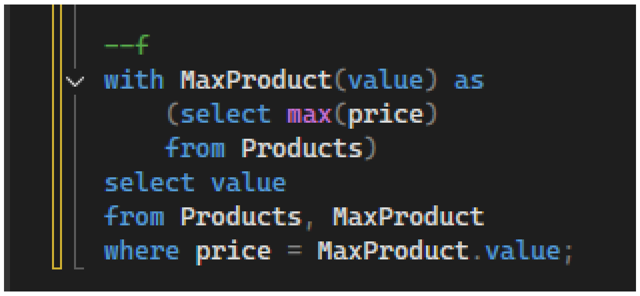
```sql
--d
SELECT customer_id FROM Orders
INTERSECT
SELECT customer_id FROM Reviews;

SELECT customer_id
FROM Orders
WHERE customer_id IN (SELECT customer_id FROM Reviews);
```

e. Develop two queries that produce the same relation using **except** and **not in**.



```sql
--e
SELECT customer_id FROM Orders
EXCEPT
SELECT customer_id FROM Reviews;

SELECT customer_id
FROM Orders
WHERE customer_id not in (SELECT customer_id FROM Reviews);
```

f. Develop a query that utilizes **with** and an aggregate function.



```sql
--f
with MaxProduct(value) as
    (select max(price)
    from Products)
select value
from Products, MaxProduct
where price = MaxProduct.value;
```

FULL CODE:

```sql
use ECommerce

--Customers Table
CREATE TABLE Customers (
	customer_id INT,
	first_name VARCHAR(15),
	middle_initial CHAR(1),
	last_name VARCHAR(15),
	email VARCHAR(30),
	phone VARCHAR(15),
	address VARCHAR(30)
	PRIMARY KEY (customer_id)
);

--Products Table
CREATE TABLE Products (
	product_id INT,
	name VARCHAR(30),
	description VARCHAR(250),
	price DECIMAL(10,2),
	category_id INT,
	image_url VARCHAR(100),
	stock_quantity INT,
	PRIMARY KEY (product_id)
);

--Orders Table
CREATE Table Orders (
	order_id INT,
	customer_id INT,
	order_date VARCHAR(10),
	total_amount INT,
	status VARCHAR(15),
	PRIMARY KEY (order_id)
);

--Items Table
CREATE TABLE Items (
	item_id INT,
	order_id INT,
	product_id INT,
	quantity INT,
	unit_price DECIMAL(10,2)
	PRIMARY KEY (item_id)
);

--Categories Table
CREATE TABLE Categories (
	category_id INT,
	name VARCHAR(30),
	PRIMARY KEY (category_id)
);

--Reviews Table
CREATE TABLE Reviews (
	review_id INT,
	product_id INT,
	customer_id INT,
	rating DECIMAL(10,2),
        comment VARCHAR(100)
        PRIMARY KEY (review_id)
);

--Add the foreign keys
ALTER TABLE Products
ADD FOREIGN KEY (category_id) REFERENCES Categories(category_id)

ALTER TABLE Orders
ADD FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)

ALTER TABLE Items
ADD FOREIGN KEY (order_id) REFERENCES Orders(order_id)

ALTER TABLE Items
ADD FOREIGN KEY (product_id) REFERENCES Products(product_id)

ALTER TABLE Reviews
ADD FOREIGN KEY (product_id) REFERENCES Products(product_id)

ALTER TABLE Reviews
ADD FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)

--Lab 2

--1
INSERT INTO Customers (customer_iD, first_name, middle_initial, last_name, email, phone,
address) VALUES
(1, 'John', 'D', 'Doe', 'johndoe@gmail.com', '555-123-4567', '123 Main St'),
(2, 'Jane', 'L', 'Doe', 'janedoe@gmail.com', '123-237-3947', '123 Second St'),
(3, 'Jack', 'R', 'Lin', 'jacklin@gmail.com', '682-260-4162', '374 Oak St');

INSERT INTO Categories (category_id, name) VALUES
(1, 'Yellow Fruit'),
(2, 'Red Fruit'),
(3, 'Orange Fruit');

INSERT INTO Products (product_id, name, description, price, category_id, image_url,
stock_quantity) VALUES
(1, 'Banana', 'Yellow Fruit', 0.99, 1, 'www.store.com/bananas', 123),
(2, 'Apple', 'Red Fruit', 0.56, 2, 'www.store.com/apple', 204),
(3, 'Orange', 'Orange Fruit', 1.03, 3, 'www.store.com/orange', 145);

INSERT INTO Orders (order_id, customer_id, order_date, total_amount, status) VALUES
(1, 3, '12/32/2024', 3, 'Successful'),
(2, 1, '01/15/2025', 3, 'Successful'),
(3, 2, '02/01/2025', 3, 'Canceled');

--2
INSERT INTO Items (item_id, order_id, product_id, quantity, unit_price) VALUES
(1, 2, 2, 3, 0.56),
(2, 1, 1, 2, 0.99),
(3, 3, 3, 1, 1.03);

INSERT INTO Reviews (review_id, product_id, customer_id, rating, comment) VALUES
(1, 3, 1, 4.92, 'Excellent Transaction!'),
(2, 1, 2, 5.00, 'Great Bananas!'),
(3, 2, 3, 1.00, 'Terrible Service');

--3
UPDATE PRODUCTS
```

```
SET price = 1.34
WHERE name = 'Banana'

UPDATE Items
SET unit_price = 1.34
WHERE product_id = 1

UPDATE Customers
SET middle_initial = 'A'
WHERE customer_id = 3

--4
Delete From Reviews WHERE review_id = 1

Delete FROM Reviews WHERE review_id = 2

DELETE FROM Reviews WHERE review_id = 3

SELECT * FROM Customers;

SELECT * FROM Products;

SELECT * FROM Orders;

SELECT * FROM Items;

SELECT * FROM Categories;

SELECT * FROM Reviews;

--5
--a
SELECT * FROM Customers
ORDER BY last_name ASC, first_name ASC;

--b
SELECT * FROM Reviews
WHERE rating < 3;

--c
SELECT *
FROM Products AS p, Categories AS c
WHERE p.category_id = c.category_id
and c.name = 'Yellow Fruit';

--d
SELECT customer_id FROM Orders
INTERSECT
SELECT customer_id FROM Reviews;

SELECT customer_id
FROM Orders
WHERE customer_id IN (SELECT customer_id FROM Reviews);

--I added this for the next 2 queries to show data
UPDATE Reviews
SET customer_id = 1
WHERE customer_id = 3

--e
SELECT customer_id FROM Orders
```

```
EXCEPT
SELECT customer_id FROM Reviews;

SELECT customer_id
FROM Orders
WHERE customer_id not in (SELECT customer_id FROM Reviews);

--f
with MaxProduct(value) as
       (select max(price)
       from Products)
select value
from Products, MaxProduct
where price = MaxProduct.value;
```