電子商務數據庫

一、專題目的

建立一個電子商務網站的數據庫，用於管理產品、客戶和訂單。此系統旨在提供方便的商品管理和訂單處理功能，並支持多種數據查詢和報告功能。

主要功能包括：

- 產品管理：記錄產品的基本信息、價格和庫存等。
- 客戶管理：記錄客戶的基本信息、聯絡方式和購買記錄等。
- 訂單管理：管理客戶的訂單、訂單詳情和付款信息。
- 產品評價：紀錄可互對產品的評價，幫助改進產品質量。
- 客戶反饋：收集和處理客戶的意見和建議，提升客戶滿意度和服務品質。

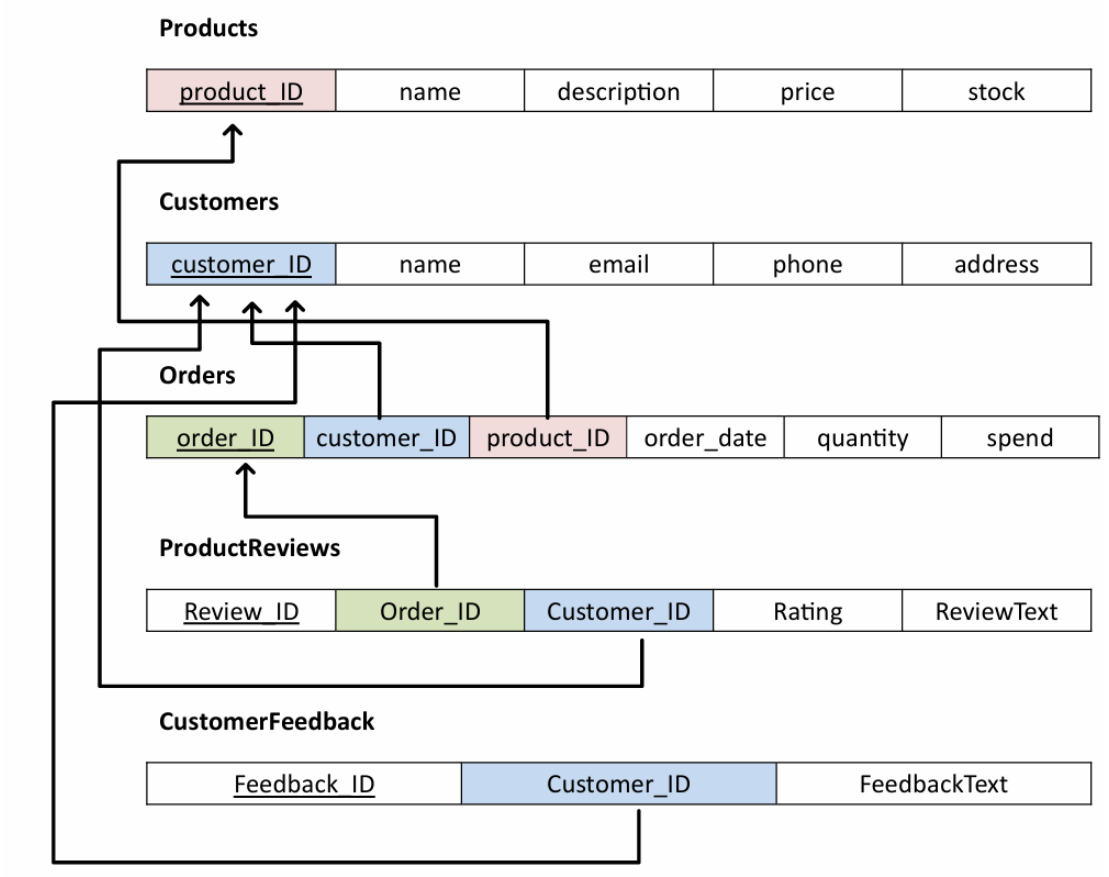
二、系統架構

設計數據庫的實體-關係模型和關聯模型，定義各個表格及其關係。

- **實體和關係**：

- **產品（Products）**：記錄產品的基本信息。
- **客戶（Customers）**：記錄客戶的基本信息。
- **訂單（Orders）**：管理客戶的訂單信息。
- **產品評價（ProductReviews）**：紀錄客戶對產品評價。
- **客戶反饋（CustomerFeedback）**：紀錄客戶對本次服務的評價。

➢ **Relational Model**：



➢ **關係**

- 一位客戶可以有多個訂單（1 對多）。
- 一個訂單只會有一位客戶（1 對 1）。
- 一個訂單只會有一個產品（1 對 1）。
- 一個產品評價只會有一位客戶 ID 和一個產品 ID。（1 對 1）。
- 一位客戶只會給一份客戶反饋（1 對 1）。

➢ **正規化**

以減少數據冗餘和增加數據一致性

1. 每個表格都應該有唯一的主鍵。

➔從上面畫的 Relational Model 可以看出 Products 表格只有一個主鍵 product_ID；Customers 表格只有一個主鍵 customer_ID；Orders 表格只有一個主鍵 order_ID；ProductReviews 表格只有一個主鍵 Review_ID；CustomerFeedback 表格只有一個主鍵 Feedback_ID 符合第一個正規化規則。

2. 每個非主鍵屬性應該完全依賴於主鍵。

➜在我們的五個表格中，每個非主鍵屬性都完全依賴於其主鍵，無法獨立存在。因此，我們可以確定這些表格符合第二個正規化規則。

3. 不應該有任何非主鍵屬性依賴於另一個非主鍵屬性。

➜在 Products 中，name、desciption、price 和 stock 都是依賴於 product_ID；在 Customers 中，name、email、phone 和 address 都是依賴於 customer_ID；在 Orders 中，customer_ID 和 product_ID 屬於外鍵，而 order_date、quantity 和 spend 都是依賴於 order_ID；在 ProductReviews 中，Order_ID 和 Customer_ID 屬於外鍵，而 Rating 和 ReviewText 都是依賴於 Review_ID；在 CustomerFeedback 中，Customer_ID 屬於外鍵，而 FeedbackText 是依賴於 Feedback_ID，屬性之間皆互相獨立；因此符合正規化第三規則。

三、Python 創建 Tables

四、執行指令 Select

建立完 table 後，我們在 MySQL 上新增 20 筆訂單資料，但只有 10 件產品。10 位客戶一人購買兩次，所以有 20 筆訂單資料、產品評價和客戶反饋。

- **執行固定 select 指令**

```
1 •  SELECT * FROM Products;
2 •  SELECT * FROM Customers;
3 •  SELECT * FROM Orders;
4 •  SELECT * FROM ProductReviews;
5 •  SELECT * FROM CustomerFeedback;
```

印出所有 tables：

product_ID	name	description	price	stock
1	棒球帽	藍F	1200.00	50
2	棒球帽	紅F	1200.00	60
3	平沿帽	黑F	1300.00	30
4	平沿帽	灰F	1300.00	20
5	老帽	藍F	1100.00	70
6	老帽	綠F	1100.00	80
7	運動帽	黃F	1400.00	90
8	運動帽	紫F	1400.00	100
9	紳士帽	黑F	1600.00	10
10	紳士帽	白F	1600.00	15
NULL	NULL	NULL	NULL	NULL

customer_ID	name	email	phone	address
1	Mike Lin	mike@gmail.com	0912345679	101 Pine St
2	Sarah Wang	sarah@gmail.com	0987654322	102 Maple St
3	James Lee	james@gmail.com	0912223334	103 Cedar St
4	Eva Liu	eva@gmail.com	0912345680	104 Birch St
5	Kevin Wu	kevin@gmail.com	0987654323	105 Walnut St
6	Nancy Chang	nancy@gmail.com	0912223335	106 Willow St
7	Tony Huang	tony@gmail.com	0912345681	107 Poplar St
8	Betty Ho	betty@gmail.com	0987654324	108 Cypress St
9	Gary Chiu	gary@gmail.com	0912223336	109 Spruce St
10	Alice Chen	alice@gmail.com	0912345682	110 Fir St
NULL	NULL	NULL	NULL	NULL

(以下皆有 20 筆資料，但是因為一次最多只能截圖 15 筆資料，所以只顯示出這樣)

order_ID	customer_ID	product_ID	order_date	quantity	spend
1	1	4	2024-05-20 12:03:22	1	1300.00
2	2	5	2024-05-21 14:09:20	2	2200.00
3	4	7	2024-05-22 10:30:00	3	4200.00
4	3	6	2024-05-22 10:30:00	1	1100.00
5	5	8	2024-05-24 12:45:00	2	2800.00
6	6	9	2024-05-25 13:15:00	1	1600.00
7	7	10	2024-05-26 14:00:00	1	1600.00
8	8	1	2024-05-27 15:30:00	2	2400.00
9	9	2	2024-05-28 16:00:00	3	3600.00
10	10	3	2024-05-29 17:45:00	2	2600.00
11	1	3	2024-11-20 01:03:02	1	1300.00
12	2	4	2024-02-21 13:09:56	2	2600.00
13	3	8	2024-03-01 17:38:23	1	1400.00
14	4	10	2024-01-23 18:27:40	3	4800.00
15	5	4	2024-02-24 21:45:00	2	2600.00

Review_ID	Order_ID	Customer_ID	Rating	ReviewText
1	1	1	5	非常滿意!
2	2	2	4	好商品!
3	3	3	5	質量很好!
4	4	4	3	不錯的選擇
5	5	5	4	挺好的!
6	6	6	5	很喜歡!
7	7	7	4	物超所值!
8	8	8	3	可以接受
9	9	9	4	物有所值!
10	10	10	5	非常棒!
11	11	1	1	差評!
12	12	2	1	不再回購!
13	13	3	5	質量很好!
14	14	4	3	不錯的選擇
15	15	5	2	還可以!

Feedback_ID	Customer_ID	FeedbackText
1	1	價格合理
2	2	物流效率高
3	3	服務周到
4	4	客服態度好!
5	5	物流很快!
6	6	產品包裝不錯
7	7	商品描述詳細
8	8	服務態度很好
9	9	出貨迅速
10	10	商品品質好
11	1	不要!
12	2	物流很快!
13	3	產品包裝不錯
14	4	商品描述詳細
15	5	服務態度很好

➢ **執行不特定多個 select 指令**

```
1 ●  SELECT * FROM Products
2    order by price,stock;
3 ●  SELECT * FROM Customers;
4 ●  SELECT *
5    FROM Orders
6    order by order_date,customer_ID;
7 ●  SELECT * FROM ProductReviews;
8 ●  SELECT * FROM CustomerFeedback;
```

(1) 我們依照產品的價格和庫存量排序，由低到高。

Ex:

product_ID = 3、4 的價格是一樣的，再用庫存量來排序，因此 product_ID=4 會排序在 product_ID=3 前面。

product_ID	name	description	price	stock
1	棒球帽	藍F	1200.00	50
2	棒球帽	紅F	1200.00	60
3	平沿帽	黑F	1300.00	30
4	平沿帽	灰F	1300.00	20
5	老帽	藍F	1100.00	70
6	老帽	綠F	1100.00	80
7	運動帽	黃F	1400.00	90
8	運動帽	紫F	1400.00	100
9	紳士帽	黑F	1600.00	10
10	紳士帽	白F	1600.00	15
NULL	NULL	NULL	NULL	NULL

product_ID	name	description	price	stock
5	老帽	藍F	1100.00	70
6	老帽	綠F	1100.00	80
1	棒球帽	藍F	1200.00	50
2	棒球帽	紅F	1200.00	60
4	平沿帽	灰F	1300.00	20
3	平沿帽	黑F	1300.00	30
7	運動帽	黃F	1400.00	90
8	運動帽	紫F	1400.00	100
9	紳士帽	黑F	1600.00	10
10	紳士帽	白F	1600.00	15
NULL	NULL	NULL	NULL	NULL

(2) 依照購買的日期和時間由過去到現在排序，但如果是同時間結帳的話再接著用客戶編號由小到大來排序。

order_ID	customer_ID	product_ID	order_date	quantity	spend
1	1	4	2024-05-20 12:03:22	1	1300.00
2	2	5	2024-05-21 14:09:20	2	2200.00
3	4	7	2024-05-22 10:30:00	3	4200.00
4	3	6	2024-05-22 10:30:00	1	1100.00
5	5	8	2024-05-24 12:45:00	2	2800.00
6	6	9	2024-05-25 13:15:00	1	1600.00
7	7	10	2024-05-26 14:00:00	1	1600.00
8	8	1	2024-05-27 15:30:00	2	2400.00
9	9	2	2024-05-28 16:00:00	3	3600.00
10	10	3	2024-05-29 17:45:00	2	2600.00
11	1	3	2024-11-20 01:03:02	1	1300.00
12	2	4	2024-02-21 13:09:56	2	2600.00
13	3	8	2024-03-01 17:38:23	1	1400.00
14	4	10	2024-01-23 18:27:40	3	4800.00
15	5	4	2024-02-24 21:45:00	2	2600.00

order_ID	customer_ID	product_ID	order_date	quantity	spend
14	4	10	2024-01-23 18:27:40	3	4800.00
12	2	4	2024-02-21 13:09:56	2	2600.00
15	5	4	2024-02-24 21:45:00	2	2600.00
13	3	8	2024-03-01 17:38:23	1	1400.00
20	10	2	2024-05-01 10:45:00	2	2400.00
1	1	4	2024-05-20 12:03:22	1	1300.00
2	2	5	2024-05-21 14:09:20	2	2200.00
4	3	6	2024-05-22 10:30:00	1	1100.00
3	4	7	2024-05-22 10:30:00	3	4200.00
5	5	8	2024-05-24 12:45:00	2	2800.00
6	6	9	2024-05-25 13:15:00	1	1600.00
7	7	10	2024-05-26 14:00:00	1	1600.00
8	8	1	2024-05-27 15:30:00	2	2400.00
9	9	2	2024-05-28 16:00:00	3	3600.00
10	10	3	2024-05-29 17:45:00	2	2600.00

➢ **執行不特定多個 select 之外任意指令**

我們額外舉例這三個指令 Insert、Update、total sum

(1) 如果該店家想新增產品，可以只 Insert 一個新的 Product 內容。範例是新增一項新品貝雷帽、黑色 Free size、價格 800、庫存 30 個。

(2) 需要修改產品庫存的話，可利用 update 更新庫存量。這裡是修改產品 ID 為 1 的產品庫存，此產品庫存減 10 並更新。

Ex:

product_ID	name	description	price	stock
1	棒球帽	藍F	1200.00	50
2	棒球帽	紅F	1200.00	60
3	平沿帽	黑F	1300.00	30
4	平沿帽	灰F	1300.00	20
5	老帽	藍F	1100.00	70
6	老帽	綠F	1100.00	80
7	運動帽	黃F	1400.00	90
8	運動帽	紫F	1400.00	100
9	紳士帽	黑F	1600.00	10
10	紳士帽	白F	1600.00	15
NULL	NULL	NULL	NULL	NULL

product_ID	name	description	price	stock
1	棒球帽	藍F	1200.00	40
2	棒球帽	紅F	1200.00	60
3	平沿帽	黑F	1300.00	30
4	平沿帽	灰F	1300.00	20
5	老帽	藍F	1100.00	70
6	老帽	綠F	1100.00	80
7	運動帽	黃F	1400.00	90
8	運動帽	紫F	1400.00	100
9	紳士帽	黑F	1600.00	10
10	紳士帽	白F	1600.00	15
11	貝雷帽	黑F	800.00	30
NULL	NULL	NULL	NULL	NULL

(3) 查詢庫存總數量(依照順序由上往下執行，先執行到產品 ID=1 庫存量-10，再進行總和得到 545)

total_stock
545

四、程式碼

three.py - C:/Users/annie/OneDrive/Desktop/three.py (3.12.3)

File Edit Format Run Options Window Help

```
import mysql.connector as mycont
from mysql.connector import errorcode

# Set connection parameters
config = {
    'user': '11011142',
    'password': '11011142',
    'host': '140.135.65.53',
    'database': 'DB11011142',
    'buffered': True
}

try:
    # Create a connection
    conn = mycont.connect(**config, use_pure=True)
    if conn.is_connected():
        print('Connected to MySQL database')
        cursor = conn.cursor()
    # 循环接受额外指令
        while True:
            additional_query = input("Enter additional SQL query (or 'exit' to quit): ")
            if additional_query.lower() == 'exit':
                break
            cursor.execute(additional_query)
            if additional_query.strip().lower().startswith("select"):
                rows = cursor.fetchall()
                print("Result of additional query:")
                for row in rows:
                    print(row)
            else:
                conn.commit()  # For non-SELECT queries, commit the transaction
                print(f"Query executed successfully: {additional_query}")

except mycont.Error as err:
    print(f"Error: {err}")
finally:
    if 'conn' in locals() and conn.is_connected():
        conn.close()
        print("Connection closed")
```

four.py - C:/Users/annie/OneDrive/Desktop/four.py (3.12.3)

File Edit Format Run Options Window Help

```
import mysql.connector as mycont
from mysql.connector import errorcode

# Set connection parameters
config = {
    'user': '11011142',
    'password': '11011142',
    'host': '140.135.65.53',
    'database': 'DB11011142',
    'buffered': True
}

try:
    # Create a connection
    conn = mycont.connect(**config, use_pure=True)
    if conn.is_connected():
        print('Connected to MySQL database')
        cursor = conn.cursor()

         # Fixed SELECT statements to retrieve all data from tables
        select_queries = [
            "SELECT * FROM Products;",
            "SELECT * FROM Customers;",
            "SELECT * FROM Orders;",
            "SELECT * FROM ProductReviews;",
            "SELECT * FROM CustomerFeedback;"
        ]

        for select_query in select_queries:
            cursor.execute(select_query)
            rows = cursor.fetchall()
            print(f"Result of query '{select_query}':")
            for row in rows:
                print(row)



except mycont.Error as err:
    print(f"Error: {err}")
finally:
    if 'conn' in locals() and conn.is_connected():
        conn.close()
        print("Connection closed")
```