

電商管理資料庫

書面報告

組別：第 8 組

組長：A11323015 王佳祐

組員：B11123224 余雨芹

A11223006 李昀榮

A11323010 鄭羽彤

A11323036 李明熹

四、正規化

4.1 正規化過程

1. Product、Suppliers、Categories：

- 未正規化：

A	B	C	D	E	F	G	H	I	J
Product_ID	Product_Name	Product_Description	Price	Supplier_ID	Supplier_Name	Supplier_Phone	Supplier_Email	Category_ID	Category_Name
P_0001	Garnier Fructis Shampoo	Revitalizing shampoo for all hair types	\$18.56	S_0005	L'Oréal	140-896-7254	gblaske4@infoseek.co.jp	CA_0005	Hair Care
P_0002	Apple iPhone 15 Pro Max	Latest flagship smartphone	\$1,199.00	S_0022	Apple Inc.	285-948-5214	jroughl@blinklist.com	CA_0001	Electronics
P_0003	Nike Air Force 1	Classic sneakers	\$99.99	S_0025	Nike, Inc.	737-441-1964	wfrierio@google.fr	CA_0003	Shoes
P_0004	Levi's 501 Jeans	Iconic denim jeans	\$79.99	S_0031	HP Inc.	941-191-3498	asmittenu@walmart.com	CA_0002	Clothing
P_0005	Ray-Ban Wayfarer Sunglasses	Classic sunglasses	\$159.99	S_0040	Warner Bros. Discovery	132-798-3947	jsavile13@bloglovin.com	CA_0004	Accessories
P_0006	KitchenAid Stand Mixer	Versatile kitchen appliance	\$399.99	S_0007	Instant Pot Company	302-652-4937	tatton6@free.fr	CA_0007	Kitchenware

說明：

- 每個供應商和類別的信息都重複儲存在 Product 資料表中。
- 想要新增一個新的產品但該產品的供應商信息還沒有被記錄，就無法將該產品的信息插入到 Product 資料表中。
- 刪除一個供應商的記錄可能會導致其他相關信息的丟失，例如：刪除了一個供應商的記錄，可能會同時刪除該供應商的所有產品信息。

- 第 1 正規化（1NF）：

表格已經滿足 1NF，因為...

- 每個欄位的資料都是不可再分的
- 每行都是唯一的

- 第 2 正規化（2NF）：

將未正規化的資料表拆分成三個資料表：

1. Products：Product_ID、Product_Name、Product_Description、Price。

Product_ID	Product_Name	Product_Description	Price	Supplier_ID	Category_ID
P_0001	Garnier Fructis Shampoo	Revitalizing shampoo for all hair types	\$18.56	S_0005	CA_0005
P_0002	Apple iPhone 15 Pro Max	Latest flagship smartphone	\$1,199.00	S_0022	CA_0001
P_0003	Nike Air Force 1	Classic sneakers	\$99.99	S_0025	CA_0003
P_0004	Levi's 501 Jeans	Iconic denim jeans	\$79.99	S_0031	CA_0002
P_0005	Ray-Ban Wayfarer Sunglasses	Classic sunglasses	\$159.99	S_0040	CA_0004
P_0006	KitchenAid Stand Mixer	Versatile kitchen appliance	\$399.99	S_0007	CA_0007

2. Suppliers：Supplier_ID、Supplier_Name、Supplier_Phone、Supplier_Email。

Supplier_ID	Supplier_Name	Supplier_Phone	Supplier_Email
S_0005	L'Oréal	140-896-7254	gblaske4@infoseek.co.jp
S_0022	Apple Inc.	285-948-5214	jroughl@blinklist.com
S_0025	Nike	737-441-1964	wfrierio@google.fr
S_0031	Levi Strauss & Co.	941-191-3498	asmittenu@walmart.com
S_0040	Luxottica	132-798-3947	jsavile13@bloglovin.com
S_0007	Whirlpool Corporation	302-652-4937	tatton6@free.fr

3. Categories : Category_ID、Category_Name。

Category_ID	Category_Name
CA_0005	Hair Care
CA_0001	Electronics
CA_0003	Shoes
CA_0002	Clothing
CA_0004	Accessories
CA_0007	Kitchenware

● 第 3 正規化 (3NF) :

目前的拆分已經符合 3NF :

- 在產品表 中，所有非主鍵屬性 (Product_Name, Product_Description, Price, Supplier_ID, Category_ID) 都直接依賴於主鍵 Product_ID。
- 在供應商表 和 類別表 中，也沒有非主鍵屬性之間的依賴關係。

2. Inventory、Warehouse :

● 未正規化 :

Inventory_ID	Product_ID	Product_Name	Product_Description	Price	Supplier_ID	Category_ID	Quantity	Warehouse_ID	Location
I_0001	P_0001	Garnier Fructis Shampoo	Revitalizing shampoo for all hair types	\$18.56	S_0005	CA_0005	105	W_0003、W_0002	219 Willow Creek Drive、218 Willow Creek Drive
I_0002	P_0002	Apple iPhone 15 Pro Max	Latest flagship smartphone	\$1,199.00	S_0022	CA_0001	88	W_0005	221 Willow Creek Drive
I_0003	P_0003	Nike Air Force 1	Classic sneakers	\$99.99	S_0025	CA_0003	14	W_0004	220 Willow Creek Drive
I_0004	P_0004	Levi's 501 Jeans	Iconic denim jeans	\$79.99	S_0031	CA_0002	109	W_0005、W_0001	221 Willow Creek Drive、217 Willow Creek Drive
I_0005	P_0005	Ray-Ban Wayfarer Sunglasses	Classic sunglasses	\$159.99	S_0040	CA_0004	22	W_0003	219 Willow Creek Drive

說明 :

- 如果需要修改某個產品的價格 (如 : P_0002 的價格或是 W_0003 的地址變動)，需要在所有出現該產品的地方手動更新，容易導致不一致。
- 若新增一個新產品到庫存，但該產品尚未有數量或倉庫 (例如暫未安排庫+存)，需要在每個欄位填入 NULL，導致不必要的數據問題。

● 第 1 正規化 (1NF) :

Inventory_ID	Product_ID	Product_Name	Product_Description	Price	Supplier_ID	Category_ID	Quantity	Warehouse_ID	Location
I_0001	P_0001	Garnier Fructis Shampoo	Revitalizing shampoo for all types	\$18.56	S_0005	CA_0005	105	W_0003	219 Willow Creek Drive
I_0001	P_0001	Garnier Fructis Shampoo	Revitalizing shampoo for all types	\$18.56	S_0005	CA_0005	105	W_0002	218 Willow Creek Drive
I_0002	P_0002	Apple iPhone 15 Pro Max	Latest flagship smartphone	\$1,199.00	S_0022	CA_0001	88	W_0005	221 Willow Creek Drive
I_0003	P_0003	Nike Air Force 1	Classic sneakers	\$99.99	S_0025	CA_0003	14	W_0004	220 Willow Creek Drive
I_0004	P_0004	Levi's 501 Jeans	Iconic denim jeans	\$79.99	S_0031	CA_0002	109	W_0005	221 Willow Creek Drive
I_0004	P_0004	Levi's 501 Jeans	Iconic denim jeans	\$79.99	S_0031	CA_0002	109	W_0001	217 Willow Creek Drive
I_0005	P_0005	Ray-Ban Wayfarer Sunglasses	Classic sunglasses	\$159.99	S_0040	CA_0004	22	W_0003	219 Willow Creek Drive

說明 :

- 欄位 Warehouse_ID 和 Location 包含多個值，違反原子性。
- 拆分每個倉庫和地點，使每一行只包含單一的倉庫及其對應的地點。

● 第 2 正規化 (2NF) :

將 Inventory 資料表拆分成三個資料表 :

1. Inventory : Inventory_ID 、Product_ID 、Quantity 、Warehouse_ID 。

Inventory_ID	Product_ID	Quantity	Warehouse_ID
I_0001	P_0001	105	W_0003
I_0001	P_0001	105	W_0002
I_0002	P_0002	88	W_0005
I_0003	P_0003	14	W_0003
I_0004	P_0004	109	W_0005
I_0004	P_0004	109	W_0001
I_0005	P_0005	22	W_0003

2. Products : Product_ID 、Product_Name 、Product_Description 、Price 。

Product_ID	Product_Name	Product_Description	Price	Supplier_ID	Category_ID
P_0001	Garnier Fructis Shampoo	Revitalizing shampoo for all hair types	\$18.56	S_0005	CA_0005
P_0002	Apple iPhone 15 Pro Max	Latest flagship smartphone	\$1,199.00	S_0022	CA_0001
P_0003	Nike Air Force 1	Classic sneakers	\$99.99	S_0025	CA_0003
P_0004	Levi's 501 Jeans	Iconic denim jeans	\$79.99	S_0031	CA_0002
P_0005	Ray-Ban Wayfarer Sunglasses	Classic sunglasses	\$159.99	S_0040	CA_0004
P_0006	KitchenAid Stand Mixer	Versatile kitchen appliance	\$399.99	S_0007	CA_0007

3. Warehouse : Warehouse_ID 和 Location 。

Warehouse_ID	Location
W_0001	217 Willow Creek Drive
W_0002	218 Willow Creek Drive
W_0003	219 Willow Creek Drive
W_0004	220 Willow Creek Drive
W_0005	221 Willow Creek Drive

● 第 3 正規化 (3NF) :

目前的拆分已經符合 3NF :

- 在產品表中，Supplier_ID 和 Category_ID 已經是外鍵，符合 3NF 。
- 在倉庫表中，Location 直接依賴於 Warehouse_ID，符合 3NF 。
- 在庫存表中，所有屬性都直接依賴於 Inventory_ID，符合 3NF 。

3. Customer :

● 未正規化 :

Customer_ID	Customer_Name	Phone	Email	Location
CU_0001	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	884 Scott Circle, Gongchang Zhen, China
CU_0002	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashford Street, Nueva Manoa, Bolivia
CU_0003	Bamby Craker	715-166-5128	bcraker2@comsenz.com	691 Green Ridge Way, Barlinek, Poland

說明 :

- Location 欄位將地址、城市和國家混合存儲，不易分割或篩選。
- 修改某個地理位置的部分信息（例如：城市名稱）會變得困難。

● 第 1 正規化 (1NF) :

原始資料已符合 1NF :

- 每一個欄位都是不可再分的（如 Location 欄位已經具體到地址、城市和國家）。
- 每一行都是唯一的。
- 主鍵為 Customer_ID。
- 第 2 正規化（2NF）：
將 Location 此欄拆分成 Country、City、Address 三個欄位：

Customer_ID	Customer_Name	Customer_Phone	Customer_Email	Country	City	Address
CU_0001	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	China	Gongchang Zhe	884 Scott Circle
CU_0002	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	Bolivia	Nueva Manoa	744 Bashford Street
CU_0003	Bamby Craker	715-166-5128	bcraker2@comsenz.com	Poland	Barlinek	691 Green Ridge Way

- 第 3 正規化（3NF）：
表格已經滿足 3NF，因為...
所有非主鍵欄位，依賴的都是主鍵。沒有傳遞依賴。

4. Order、OrderDetails：

- 未正規化：

Order_ID	Customer_ID	Order_Date	Total_Amount	Order_State	Inventory_ID	Product_Name	Quantity	Unit_Price	Customer_Name	Customer_Phone	Customer_Email	Location
O_0001	CU_0359	2023-12-12 13:17:02	\$3.99	completed	I_0019	Garnier Shampoo	1	\$3.99	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	884 Scott Circle, China
O_0002	CU_0317	2023-12-12 19:20:50	\$2,024.97	returned	I_0013, I_0209	Generic Item 1, Premium Item	3, 5	\$9.99, \$399.00	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashford Street, Bolivia
O_0003	CU_0390	2023-12-12 20:44:30	\$403.90	completed	I_0090	Nike Air Force 1	4	\$39.99	Bamby Craker	715-166-5128	bcraker2@comsenz.com	691 Green Ridge Way, Poland

- 第 1 正規化（1NF）：

Order_ID	Customer_ID	Order_Date	Total_Amount	Order_State	Inventory_ID	Product_Name	Quantity	Unit_Price	Customer_Name	Customer_Phone	Customer_Email	Location
O_0001	CU_0359	2023-12-12 13:17:02	\$3.99	completed	I_0019	Garnier Shampoo	1	\$3.99	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	884 Scott Circle, China
O_0002	CU_0317	2023-12-12 19:20:50	\$2,024.97	returned	I_0013	Generic Item 1	3	\$9.99	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashford Street, Bolivia
O_0002	CU_0317	2023-12-12 19:20:50	\$2,024.97	returned	I_0209	Premium Item	5	\$399.00	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashford Street, Bolivia
O_0003	CU_0390	2023-12-12 20:44:30	\$403.90	completed	I_0090	Nike Air Force 1	4	\$39.99	Bamby Craker	715-166-5128	bcraker2@comsenz.com	691 Green Ridge Way, Poland

說明：

- 欄位 Inventory_ID、Product_Name、Quantity 和 Unit_Price 在 O_0002 訂單中包含多個值，違反了 1NF 的「原子性」原則。
- 將含有多個商品的訂單拆分多行，保證每一行只儲存單一的商品資訊
- 第 2 正規化（2NF）：

將 Orders 資料表拆分成兩個資料表：

1.Orders：Order_ID、Customer_ID、Order_Date、Total_Amount、Order_State

Order_ID	Customer_ID	Order_Date	Total_Amount	Order_State
O_0001	CU_0359	2023-12-12 13:17:02	\$3.99	completed
O_0002	CU_0317	2023-12-12 19:20:50	\$2,024.97	returned
O_0003	CU_0390	2023-12-12 20:44:30	\$403.90	completed

2. **OrderDetails** : Order_ID 、 Inventory_ID 、 Quantity 、 Unit_Price 。

Order_ID	Inventory_ID	Quantity	Unit_Price
O_0001	I_0019	1	\$3.99
O_0002	I_0013	3	\$9.99
O_0002	I_0209	5	\$399.00
O_0003	I_0090	4	\$39.99

因先前已建立 Customer 表，所以引用即可：

Customer_ID	Customer_Name	Customer_Phone	Customer_Email
CU_0001	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org
CU_0002	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com
CU_0003	Bamby Craker	715-166-5128	bcraker2@comsenz.com
CU_0004	Leah Saunder	484-415-6859	lsaunder3@g.co

● 第 3 正規化 (3NF) :

Product_ID	Product_Name	Product_Description	Price
P_0001	Garnier Fructis Shampoo	Revitalizing shampoo for all hair types	\$18.56
P_0002	Apple iPhone 15 Pro Max	Latest flagship smartphone	\$1,199.00
P_0003	Nike Air Force 1	Classic sneakers	\$99.99
P_0004	Levi's 501 Jeans	Iconic denim jeans	\$79.99

說明：

- Customer_Name、Customer_Phone 和 Customer_Email 直接依賴於 Customer_ID。
- 在 OrderDetails 表中，Unit_Price 依賴於 Inventory_ID，所以產品資訊引用先前建立的 Products 表。

5. Returns :

● 未正規化：

Return_ID	Order_ID	Return_Date	Customer_ID	Customer_Name	Customer_Phone	Customer_Email	Location	Product_Name	Quantity	Unit_Price	Total_Amount	Order_State
R_0001	O_0002	2023-12-17 19:20:50	CU_0317	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashford Street, Bolivia	Generic Item 1, Premium Iter 3, 5	3	\$9.99, \$399.00	\$29.97, \$1,995	returned
R_0002	O_0004	2023-12-18 0:24:43	CU_0340	John Smith	801-222-3333	jsmith@email.com	123 Main St, USA	Levi's Jeans	2	\$79.99	\$159.98	returned
R_0003	O_0028	2023-12-20 1:37:31	CU_0405	Jane Doe	810-555-8888	janedoe@email.com	456 Oak Lane, UK	iPhone 15	1	\$1,199.00	\$1,199.00	completed

說明：如果新增退貨記錄，但沒有商品資訊（如 Product_Name 和 Quantity），則無法插入數據，因為所有欄位都是強制必須填寫的。

● 第 1 正規化 (1NF) :

Return_ID	Order_ID	Return_Date	Customer_ID	Customer_Name	Customer_Phone	Customer_Email	Location	Product_Name	Quantity	Unit_Price	Total_Amount	Order_State
R_0001	O_0002	2023-12-17 19:20:50	CU_0317	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashf	Generic Item 1	3	\$9.99	\$29.97	returned
R_0001	O_0002	2023-12-17 19:20:50	CU_0317	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashf	Premium Item	5	\$399.00	\$1,995	returned
R_0002	O_0004	2023-12-18 0:24:43	CU_0340	John Smith	801-222-3333	jsmith@email.com	123 Main	Levi's Jeans	2	\$79.99	\$159.98	returned
R_0003	O_0028	2023-12-20 1:37:31	CU_0405	Jane Doe	810-555-8888	janedoe@email.com	456 Oak L	iPhone 15	1	\$1,199.00	\$1,199.00	completed

說明：

- 欄位 Order_Product_Name、Quantity 和 Unit_Price 在 R_0001 訂單中包含多個值，違反了 1NF 的原子性原則。
- 將多個商品資訊分拆成獨立的行，確保每一行只存儲單一的商品資訊。

● 第 2 正規化 (2NF) :

將 Returns 資料表分出：

Return_ID	Order_ID	Return_Date
R_0001	O_0002	2023-12-17 19:20:50
R_0002	O_0004	2023-12-14 00:24:43
R_0003	O_0028	2023-12-21 01:37:31
R_0004	O_0031	2023-12-24 00:25:34
R_0005	O_0038	2023-12-21 22:06:12

因先前已建立 Customer 表，所以引用即可：

Customer_ID	Customer_Name	Customer_Phone	Customer_Email	Country	City	Address
CU_0001	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	China	Gongchang Zhe	884 Scott Circle
CU_0002	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	Bolivia	Nueva Manoa	744 Bashford Street
CU_0003	Bamby Craker	715-166-5128	bcraker2@comsenz.com	Poland	Barlinek	691 Green Ridge Way
CU_0004	Leah Saunder	484-415-6859	lsaunder3@g.co	Philippines	President Roxas	5 Dwight Way
CU_0005	Pip Eiler	150-716-3464	peiler4@ocn.ne.jp	Mexico	Providencia	0 Briar Crest Way

因先前已建立 OrderDetails 表，所以引用即可：

Order_ID	Inventory_ID	Quantity	Unit_Price
O_0001	I_0019	1	\$3.99
O_0002	I_0013	3	\$9.99
O_0002	I_0209	5	\$399.00
O_0003	I_0090	4	\$39.99

● 第 3 正規化 (3NF)：

表格已經滿足 3NF，因為...

在 Customers 表 和 OrderDetails 表 中，所有非主鍵欄位都已直接依賴於主鍵，沒有傳遞依賴。

6. PaymentInfo：

● 未正規化：

Payment_ID	Order_ID	Payment_Method	Payment_Date	Payment_State	Customer_ID	Customer_Name	Customer_Phone	Customer_Email	Location	Product_Name	Quantity	Unit_Price	Total_Amount
P_0001	O_0001	cash	2023-12-16 13:17:02	completed	CU_0359	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	884 Scott Circle	Garnier Shampoo	1	\$3.99	\$3.99
P_0002	O_0002	transferred	2023-12-15 19:20:50	returned	CU_0317	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashford St	Generic Item 1, Premium Item	3, 5	\$9.99, \$399.00	\$29.97, \$1,995.00
P_0003	O_0003	transferred	2023-12-14 20:44:30	completed	CU_0390	Bamby Craker	715-166-5128	bcraker2@comsenz.com	691 Green Ridge Way	Nike Air Force 1	4	\$39.99	\$159.96
P_0004	O_0004	cash	2023-12-15 0:24:43	returned	CU_0340	John Smith	801-222-3333	jsmith@email.com	123 Main St	Levi's Jeans	2	\$79.99	\$159.98
P_0005	O_0005	transferred	2023-12-17 17:23:55	completed	CU_0405	Jane Doe	810-555-8888	janedoe@email.com	456 Oak Lane	iPhone 15	1	\$1,199.00	\$1,199.00

● 第 1 正規化 (1NF)：

Payment_ID	Order_ID	Payment_Method	Payment_Date	Payment_State	Customer_ID	Customer_Name	Customer_Phone	Customer_Email	Location	Product_Name	Quantity	Unit_Price	Total_Amount
P_0001	O_0001	cash	2023-12-16 13:17:02	completed	CU_0359	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	884 Scott Circle	Garnier Shampoo	1	\$3.99	\$3.99
P_0002	O_0002	transferred	2023-12-15 19:20:50	returned	CU_0317	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashford St	Generic Item 1	3	\$9.99	\$29.97
P_0002	O_0002	transferred	2023-12-15 19:20:50	returned	CU_0317	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	744 Bashford St	Premium Item	5	\$399.00	\$1,995.00
P_0003	O_0003	transferred	2023-12-14 20:44:30	completed	CU_0390	Bamby Craker	715-166-5128	bcraker2@comsenz.com	691 Green Ridge Way	Nike Air Force 1	4	\$39.99	\$159.96
P_0004	O_0004	cash	2023-12-15 0:24:43	returned	CU_0340	John Smith	801-222-3333	jsmith@email.com	123 Main St	Levi's Jeans	2	\$79.99	\$159.98
P_0005	O_0005	transferred	2023-12-17 17:23:55	completed	CU_0405	Jane Doe	810-555-8888	janedoe@email.com	456 Oak Lane	iPhone 15	1	\$1,199.00	\$1,199.00

說明：

- 欄位 Product_Name、Quantity 和 Unit_Price 在 P_0002 的記錄中包含多個值，違反了 1NF 的原子性原則。

● 第 2 正規化 (2NF)：

將 PaymentInfo 拆分出：

PaymentInfo : Payment_ID 、 Order_ID 、 Payment_Method 、
Payment_Date 、 Payment_state (completed, processed, failed, returned)

Payment_ID	Order_ID	Payment_Method	Payment_Date	Payment_state(completed, processed, failed, returned)
P_0001	O_0001	transferred	2023-12-16 13:17:02	completed
P_0002	O_0002	cash	2023-12-13 19:20:50	returned
P_0003	O_0003	cash	2023-12-16 20:44:30	completed
P_0004	O_0004	transferred	2023-12-16 00:24:43	returned
P_0005	O_0005	transferred	2023-12-18 17:23:55	completed

因先前已建立 Customer 表，所以引用即可：

Customer_ID	Customer_Name	Customer_Phone	Customer_Email	Country	City	Address
CU_0001	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	China	Gongchang Zhe	884 Scott Circle
CU_0002	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	Bolivia	Nueva Manoa	744 Bashford Street
CU_0003	Bamby Craker	715-166-5128	bcraker2@comsenz.com	Poland	Barlinek	691 Green Ridge Way
CU_0004	Leah Saunder	484-415-6859	lsaunder3@g.co	Philippines	President Roxas	5 Dwight Way
CU_0005	Pip Eiler	150-716-3464	peiler4@ocn.ne.jp	Mexico	Providencia	0 Briar Crest Way

因先前已建立 OrderDetails 表，所以引用即可：

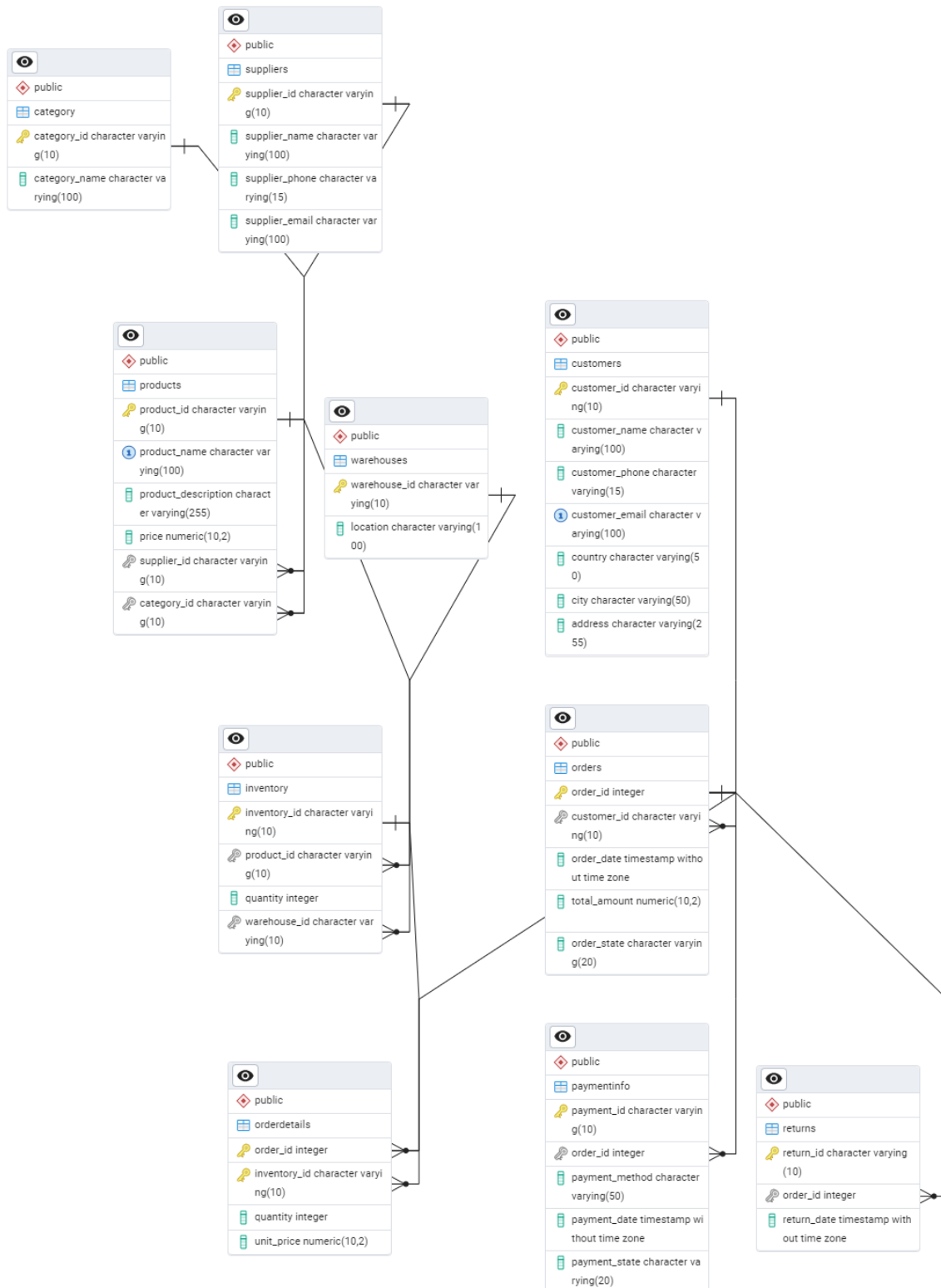
Order_ID	Inventory_ID	Quantity	Unit_Price
O_0001	I_0019	1	\$3.99
O_0002	I_0013	3	\$9.99
O_0002	I_0209	5	\$399.00
O_0003	I_0090	4	\$39.99

● 第 3 正規化 (3NF)：

表格已經滿足 3NF，因為...

在 Customers 表和 OrderDetails 表中，所有非主鍵欄位都已直接依賴於主鍵，
沒有傳遞依賴。

4.2 正規化後的關聯網要



五、SQL 撰寫 - DDL 和 DML

5.1 DDL 腳本

5.1.1 創建資料表 (Tables)

以創建 Product 資料表為例：

```
-- Table1 Products

CREATE TABLE Products (

    Product_ID VARCHAR(10) PRIMARY KEY,          -- 產品編號

    Product_Name VARCHAR(100) NOT NULL,          -- 產品名稱

    Product_Description VARCHAR(255),             -- 產品描述

    Price DECIMAL(10, 2) NOT NULL,                -- 產品價格

    Supplier_ID VARCHAR(10),                      -- 外鍵：供應商編號

    Category_ID VARCHAR(10)                      -- 外鍵：產品類型編號

);

-- 設定 Product_ID 遞增

CREATE SEQUENCE product_id_seq

    START WITH 1

    INCREMENT BY 1

    MINVALUE 0

    NO MAXVALUE

    CACHE 1;
```

5.1.2 資料表之間的關聯 (Relationships)

```
-- Products

ALTER TABLE Products

ADD CONSTRAINT supplies

FOREIGN KEY (Supplier_ID) REFERENCES Suppliers (Supplier_ID)

ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE Products
```

```

ADD CONSTRAINT belongs_to_category
FOREIGN KEY (Category_ID) REFERENCES Category (Category_ID)
ON DELETE RESTRICT ON UPDATE RESTRICT;

```

5.1.3 創建索引 (Indexes)

```

-- 為產品表的 product_ID 建立索引
CREATE INDEX idx_product_primary ON products (Product_id);

-- 為產品表的 Supplier_ID 建立索引
CREATE INDEX idx_supplier_id ON Products(Supplier_ID);

-- 為庫存表的 Product_ID 建立索引
CREATE INDEX idx_inventory_product_id ON Inventory(Product_ID);

-- 為訂單表的 Customer_ID 建立索引
CREATE INDEX idx_customer_id ON Orders(Customer_ID);

-- 為支付資訊表的 Order_ID 建立索引
CREATE INDEX idx_payment_order_id ON PaymentInfo(Order_ID);

-- 為訂單明細表的 Order_ID 和 Inventory_ID 建立索引
CREATE INDEX idx_order_inventory_id ON OrderDetails(Order_ID, Inventory_ID);

-- 為訂單日期建立索引
CREATE INDEX idx_order_date ON Orders(Order Date);

```

5.1.4 功能 (Functions)

1. ID 格式自動化，自動將序列的當前值格式化為 Product_ID : P_XXX

```

-- Product_id 格式:P_XXXX
CREATE FUNCTION set_product_id()
RETURNS TRIGGER AS $$
BEGIN
    NEW.product_id := 'P_' || my_lpad(nextval('product_id_seq'), 4, '0');
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

```

2. Orderdetail.total_amount 查表加總

```

CREATE FUNCTION update_order_total()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE orders
    SET total_amount = (
        SELECT SUM(quantity * unit_price)
        FROM orderdetails
        WHERE order_id = NEW.order_id
    )
    WHERE order_id = NEW.order_id;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

```

3. Orderdetails.unit_price 查表 Products.price

```

CREATE FUNCTION update_unit_price() RETURNS TRIGGER AS $$
DECLARE
    product_price NUMERIC(10, 2);
BEGIN
    SELECT price INTO product_price
    FROM products
    WHERE product_id = (
        SELECT product_id
        FROM inventory
        WHERE inventory_id = NEW.inventory_id
    );
    NEW.unit_price := product_price;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

```

4. 在 Returns 表紀錄一筆 Order_ID 之後，此筆 order_state 紀錄為 returned

```

CREATE FUNCTION update_order_status_on_return() RETURNS TRIGGER AS $$
BEGIN

    UPDATE orders

    SET order_state = 'returned'

    WHERE order_id = NEW.order_id;

    RETURN NEW;

END;

$$ LANGUAGE plpgsql;

```

5. 在 Returns 表紀錄一筆 Order_ID 之後，此筆 payment_state 紀錄為 returned

```

CREATE FUNCTION update_payment_state_on_return() RETURNS TRIGGER AS $$
BEGIN

    IF EXISTS (

        SELECT 1

        FROM paymentinfo

        WHERE order_id = NEW.order_id

    ) THEN

        UPDATE paymentinfo

        SET payment_state = 'returned'

        WHERE order_id = NEW.order_id;

    END IF;

    RETURN NEW;

END;

$$ LANGUAGE plpgsql;

```

6. Order_state 紀錄為 cancelled 之後，Payment_state 同步紀錄為 cancelled

```

CREATE FUNCTION update_payment_state_on_order_cancel() RETURNS TRIGGER AS $$
BEGIN

    IF NEW.order_state = 'cancelled' THEN

        UPDATE paymentinfo

        SET payment_state = 'cancelled'

```

```

        WHERE order_id = NEW.order_id ;

    END IF;

    RETURN NEW;

END;

$$ LANGUAGE plpgsql;

```

7. 在更新 Orders 表的 order_state 之前，檢查 paymentinfo 表中的狀態

```

CREATE FUNCTION check_payment_state_before_update() RETURNS trigger AS $$
BEGIN

    IF NEW.order_state = 'completed' THEN

        IF NOT EXISTS (

            SELECT 1 FROM paymentinfo

            WHERE NEW.order_id = paymentinfo.order_id

            AND paymentinfo.payment_state = 'completed'

        ) THEN

            RAISE EXCEPTION 'Payment state must be completed before updating
order state to completed.';

        END IF;

    END IF;

    RETURN NEW;

END;

$$ LANGUAGE plpgsql;

```

5.1.5 限制（Constraints）

1. 確定產品名稱的唯一性

```

-- 確保每個產品的名稱唯一

ALTER TABLE Products

ADD CONSTRAINT unique product name UNIQUE (Product Name);

```

2. 確定退貨紀錄只出現一次，避免重複退貨的問題

```

-- 確保退貨紀錄中訂單號碼只出現一次(不重複退貨)

ALTER TABLE Returns

ADD CONSTRAINT unique return order UNIQUE (order id);

```


3. 確定產品價格及庫存數量大於或等於 0，避免價錢或數量出現負數的情況發生

```
-- 確保產品價格大於或等於 0

ALTER TABLE Products

ADD CONSTRAINT chk_product_price CHECK (Price >= 0);

-- 確保庫存數量大於或等於 0

ALTER TABLE Inventory

ADD CONSTRAINT chk_inventory_quantity CHECK (Quantity >= 0);
```

4. CHECK Order_state("Confirmed", "Returned", "Cancelled", "Completed")

```
-- 確保訂單狀態只會是 'confirmed', 'returned', 'cancelled', 'completed'

ALTER TABLE Orders

ADD CONSTRAINT chk_order_state CHECK (Order_State IN ('confirmed', 'returned', 'cancelled', 'completed'));
```

5. CHECK Payment_state("Processed", "Returned", "Cancelled", "Completed")

```
-- 確保付款狀態只會是 'processed', 'returned', 'cancelled', 'completed'

ALTER TABLE PaymentInfo

ADD CONSTRAINT chk_payment_state CHECK (Payment state IN ('processed', 'returned', 'cancelled', 'completed'));
```

6. UNIQUE (order_id, inventory_id)

orderdetails 紀錄中，一筆訂單對應一筆 inventory_id

```
-- 同一件商品同一筆訂單只會出現一次

ALTER TABLE OrderDetails

ADD CONSTRAINT unique_order_item

UNIQUE (order id, inventory id);
```

5.1.6 觸發 (TRIGGERS)

1. 插入新紀錄後，自動將序列的當前值格式化為 Product_ID : P_XXX

```
-- 插入新記錄時，自動將序列的當前值格式化為 Product_ID

CREATE TRIGGER trigger_set_product_id
```

```
BEFORE INSERT ON Products
FOR EACH ROW
EXECUTE PROCEDURE set_product_id();
```

2. 在 Returns 表紀錄一筆 Order_ID 之後，此筆 order_state 紀錄為 returned

```
-- #3 插入新的 Returns 紀錄後，更新 Orders.Order_state
CREATE TRIGGER update_order_status_trigger
AFTER INSERT ON returns
FOR EACH ROW
EXECUTE FUNCTION update_order_status_on_return();
```

3. 在 Returns 表紀錄一筆 Order_ID 之後，若有建立 paymentinfo，此筆 payment_state 紀錄為 returned

```
-- #4 插入新的 Returns 紀錄後，更新 Paymentinfo.Payment_state
CREATE TRIGGER update_payment_state_trigger
AFTER INSERT ON returns
FOR EACH ROW
EXECUTE FUNCTION update_payment_state_on_return();
```

4. 在 Order_state 紀錄為 Cancelled 之後，payment_state 紀錄為 cancelled

```
-- #5 更新訂單狀態之後，檢查訂單狀態是否為 cancelled，若是則同步更新 payment_state
CREATE TRIGGER update_payment_state_trigger
AFTER UPDATE ON orders
FOR EACH ROW
EXECUTE FUNCTION update_payment_state_on_order_cancel();
```

5. 確保 payment_state 為 completed 之前，order_state 不會變成 completed

```
-- #6 檢查訂單更新之前付款是否完成
CREATE TRIGGER check_payment_state_trigger
BEFORE UPDATE ON orders
FOR EACH ROW
EXECUTE FUNCTION check_payment_state_before_update();
```

5.1.7 建立檢視

1. monthlysales :

- 目的：分析各項產品每個月的銷售表現。
- 字段：product_id, product_name, count, sales_quantity, sales_amount。
- 應用場景：對產品銷售量進行分析，統計各月各產品銷售狀況。

```
-- 產品月報表

CREATE VIEW monthlysales AS

SELECT

    TO_CHAR(o.order_date, 'YYYY-MM') AS selling_period,

    p.product_id,

    p.product_name,

    COUNT(o.order_id),

    SUM(od.quantity) AS sales_quantity,

    SUM(od.quantity * p.price) AS sales_amount

FROM

    orders AS o

INNER JOIN orderdetails AS od ON o.order_id = od.order_id

INNER JOIN inventory AS i ON od.inventory_id = i.inventory_id

INNER JOIN products AS p ON i.product_id = p.product_id

GROUP BY

    p.product_id, TO_CHAR(o.order_date, 'YYYY-MM')

ORDER BY

    selling_period, product id ;
```

2. categorysales :

- 目的：分析不同產品類別的銷售表現，以便調整產品策略。
- 字段：category_id, category_name, total_sales_quantity, total_sales_amount, selling_period。
- 應用場景：找出最賺錢的產品類別，以及哪些類別需要加強推廣。

```
-- 各分類月報表

CREATE VIEW category_sales AS

SELECT
```

```

        TO_CHAR(o.order_date, 'YYYY-MM') AS selling_period,
        c.category_id,
        c.category_name,
        SUM(od.quantity) AS total_sales_quantity,
        SUM(od.quantity * p.price) AS total_sales_amount
FROM
    orders AS o
INNER JOIN orderdetails AS od ON o.order_id = od.order_id
INNER JOIN inventory AS i ON od.inventory_id = i.inventory_id
INNER JOIN products AS p ON i.product_id = p.product_id
INNER JOIN category AS c ON p.category_id = c.category_id
GROUP BY
    c.category_id, c.category_name, TO_CHAR(o.order_date, 'YYYY-MM')
ORDER BY
selling_period, category_id;

```

3. inventory_state :

- 目的： 監控庫存水平，避免缺貨或庫存積壓。
- 字段： product_id, product_name, quantity_on_hand, reorder_level。
- 應用場景： 進行庫存預測，制定補貨計劃，避免缺貨導致的銷售損失。

```

-- 庫存情況表

CREATE VIEW inventory_status AS
SELECT
    i.product_id,
    p.product_name,
    i.quantity,
    CASE WHEN i.quantity <= 20 THEN 'low'
        ELSE 'sufficient.'
    END AS status
FROM
    inventory AS i
INNER JOIN products AS p ON i.product_id = p.product_id;

```

5.2 DML 腳本

1. 產品記錄手續

```
-- 新增產品

BEGIN TRANSACTION;

INSERT INTO Products (Product Name, Product Description, Price, Supplier ID,
Category_ID)

VALUES ('Fuji Apple', 'Apples picked directly from the source.', 92.29,
'S_0005', 'CA_0008');

INSERT INTO Inventory (Product_ID, Quantity, Warehouse_ID)

VALUES ((SELECT Product id FROM Products WHERE Product Name = 'Fuji Apple'),
100, 'W_0001');

COMMIT;

-- 修改產品資訊

BEGIN TRANSACTION;

UPDATE Products

SET Product_Name = '新的產品名稱', Price = 29.99

WHERE Product_ID = 'product_id';

COMMIT;

-- 刪除產品

BEGIN TRANSACTION;

DELETE FROM Products

WHERE Product_ID = 'product_id';

COMMIT;

BEGIN TRANSACTION;

-- 更新庫存

UPDATE Inventory

SET Quantity = Quantity + 10

WHERE Product_ID = 'product_id' AND Warehouse_ID = 'warehouse_id';

COMMIT;
```

2. 退貨手續

```
BEGIN;

-- 新增退貨記錄
```

```

INSERT INTO Returns (Order ID)
VALUES (998);

-- 更新庫存數量
UPDATE Inventory I
SET Quantity = Quantity + (
    SELECT COALESCE(SUM(OD.Quantity), 0)
    FROM OrderDetails OD
    WHERE OD.Inventory ID = I.Inventory ID AND OD.Order ID = 998
)
WHERE EXISTS (
    SELECT 1
    FROM OrderDetails OD
    WHERE I.Inventory ID = OD.Inventory ID AND OD.Order ID = 998
);

-- 訂單以及付款狀態已透過觸發改成'returned'
COMMIT;

```

3. 銷貨手續

```

BEGIN TRANSACTION;

-- 新增資料
INSERT INTO Orders (Customer_ID, Order_State)
VALUES ('CU_0033', 'confirmed');

INSERT INTO OrderDetails (Order_ID, Inventory_ID, Quantity)
VALUES (
    LASTVAL(),
    'I_0028',
    5
);

-- 更新資料
UPDATE Inventory
SET Quantity = Quantity - 5
WHERE Inventory_ID = 'I_0028';

INSERT INTO PaymentInfo (Order_ID, Payment_Method, Payment_State)
VALUES (
    LASTVAL(),
    'transferred',
    'processed'
);

UPDATE Orders

```



```
SET Order_State = 'confirmed'

WHERE Order_ID = LASTVAL();

COMMIT;
```

4. 顧客資料更新相關手續

```
-- 新增顧客資料

BEGIN;

INSERT INTO Customers (Customer Name, Customer Phone, Customer Email, Country,
City, Address)

VALUES ('A DUDE', '568-464-8688', 'a.dude@example.com', 'Taiwan', 'Taipei',
'No.15, Sec. 1, Jianguo N. Rd., Zhongshan Dist.');
```

```
COMMIT;
```

```
-- 更新顧客資料

BEGIN;

UPDATE Customers

SET Customer_Phone = '568-464-8688'

WHERE Customer_ID = 'CU_0001'; --/Customer_Name = 'A DUDE'

COMMIT;
```

```
-- 刪除顧客資料

BEGIN;

DELETE FROM Customers

WHERE Customer_ID = 'CU_0001';

COMMIT;
```

```
/*取消訂單手續*/

BEGIN;

UPDATE Orders

SET Payment_state = 'cancelled',

    Order_state = 'cancelled'

WHERE Order_ID = 123;

COMMIT;
```

5.3 關鍵查詢的 SQL 腳本

1. 查詢每個顧客的訂單總金額：

說明：此查詢展示每位顧客的所有訂單的總金額，使用了 JOIN 和 SUM()聚合函數來計算總額。

```
SELECT

    c.Customer_Name,

    SUM(o.Total_Amount) AS Total_Spent

FROM

    Customers c

JOIN Orders o ON c.Customer_ID = o.Customer_ID

GROUP BY

    c.Customer_Name;
```

2. 查詢每個產品的銷售總金額：

說明：計算每個產品的銷售總額，通過 JOIN 結合訂單明細、訂單紀錄、庫存資料和產品資料，過濾已退貨或已取消的值，並使用 SUM()和 COUNT()聚合銷售金額以及計算銷售筆數。

```
SELECT

    p.product_id,

    p.Product_Name,

    COUNT(od.order_id) AS OrderCount,

    SUM(od.Quantity * od.Unit_Price) AS Total_Sales

FROM

    OrderDetails od

JOIN inventory i ON od.Inventory_ID = i.Inventory_ID

JOIN Products p ON i.Product_ID = p.Product_ID

JOIN Orders o ON od.Order_ID = o.Order_ID

WHERE

    o.Order_State NOT IN ('cancelled', 'returned')

GROUP BY

    p.product_id, p.Product_Name
```

```
ORDER BY  
  
    p.product_id;
```

3. 查詢庫存數量少於 5 的產品：

說明：此查詢顯示庫存數量少於 5 的產品，通過 JOIN 結合庫存與產品資料進行過濾。

```
SELECT  
  
    p.Product_Name,  
  
    i.Quantity  
  
FROM  
  
    Inventory i  
  
JOIN Products p ON i.Product_ID = p.Product_ID  
  
WHERE  
  
    i.Quantity < 5;
```

4. 查詢顧客未支付的訂單：

說明：這個查詢顯示顧客未支付的訂單，使用了多表 JOIN 並過濾支付狀態為 "Pending" 的記錄。

```
SELECT  
  
    o.Order_ID,  
  
    o.Order_Date,  
  
    c.Customer_Name,  
  
    p.Payment_Status  
  
FROM  
  
    Orders o  
  
JOIN Customers c ON o.Customer_ID = c.Customer_ID  
  
JOIN PaymentInfo p ON o.Order_ID = p.Order_ID  
  
WHERE  
  
    p.Payment_Status = 'Pending';
```

5. 查詢所有訂單及其退換貨情況：

說明：此查詢展示所有訂單中涉及的退換貨情況，通過 JOIN 關聯返回資料。

```
SELECT
```

```

        o.Order_ID,

        p.Product_Name,

        r.Return_Date

FROM

    Returns r

JOIN Products p ON r.Inventory_ID = p.Product_ID

JOIN Orders o ON r.Order_ID = o.Order_ID;

```

6. 查詢某個顧客的所有訂單與支付情況：

說明：查詢顧客的所有訂單及其支付狀態，使用 JOIN 來連接訂單和支付資料。

```

SELECT

    o.Order_ID,

    o.Order_Date,

    p.Payment_Method,

    p.Payment_Status

FROM

    Orders o

JOIN PaymentInfo p ON o.Order_ID = p.Order_ID

WHERE

    o.Customer_ID = 1;

```

7. 查詢每個倉庫的庫存情況：

說明：顯示每倉庫的庫存情況，通過 JOIN 連接倉庫、庫存和產品資料。

```

SELECT

    w.Location,

    p.Product_Name,

    i.Quantity

FROM

    Warehouses w

JOIN Inventory i ON w.Warehouse_ID = i.Warehouse_ID

JOIN Products p ON i.Product_ID = p.Product_ID;

```

8. 查詢顧客訂單中訂單明細的產品及數量：

說明：查詢每位顧客的訂單明細，顯示訂單中的產品名稱和數量，通過多表 JOIN 來組合資料。

```
SELECT

    c.Customer_Name,

    o.Order_ID,

    p.Product_Name,

    od.Quantity

FROM

    Customers c

JOIN Orders o ON c.Customer_ID = o.Customer_ID

JOIN OrderDetails od ON o.Order_ID = od.Order_ID

JOIN Products p ON od.Inventory_ID = p.Product_ID;
```

9. 查詢特定月份的銷售數量和金額：

說明：查詢特定月份的銷售數量和金額，使用 EXTRACT() 函數提取月份並進行分組。

```
SELECT

    p.Product_Name,

    SUM(od.Quantity) AS Total_Quantity,

    SUM(od.Quantity * od.Unit_Price) AS Total_Sales

FROM

    OrderDetails od

JOIN Products p ON od.Inventory_ID = p.Product_ID

WHERE

    EXTRACT(MONTH FROM o.Order_Date) = 12  -- 例如，查詢 12 月的數據

GROUP BY

    p.Product Name;
```

10. 查詢每個供應商提供的所有產品及其庫存數量：

說明：查詢每個供應商提供的產品及其庫存數量，通過 JOIN 關聯供應商、產品和庫存資料。

```

SELECT

    s.Supplier_Name,

    p.Product_Name,

    i.Quantity

FROM

    Suppliers s

JOIN Products p ON s.Supplier_ID = p.Supplier_ID

JOIN Inventory i ON p.Product_ID = i.Product_ID;

```

11. 查詢某個產品的庫存數量及其所在倉庫位置：

說明：查詢某個產品的庫存數量及其所在倉庫位置，通過多表 JOIN 來獲取所需資料。

```

SELECT

    p.Product_Name,

    i.Quantity,

    w.Location

FROM

    Products p

JOIN Inventory i ON p.Product_ID = i.Product_ID

JOIN Warehouses w ON i.Warehouse_ID = w.Warehouse_ID

WHERE

    p.Product_ID = 1;  -- 例如查詢 Product_ID = 1 的產品

```

12. 查詢訂單明細中數量最多的產品：

說明：查詢在訂單明細中數量最多的產品，使用 SUM() 聚合函數並限制結果返回數量最多的產品。

```

SELECT

    p.Product_Name,

    SUM(od.Quantity) AS Total_Quantity

FROM

    OrderDetails od

JOIN Products p ON od.Inventory_ID = p.Product_ID

GROUP BY

```



```
p.Product_Name  
  
ORDER BY  
  
Total_Quantity DESC  
  
LIMIT 1;
```

13. 查詢顧客的訂單及支付情況，並按訂單日期排序：

說明：查詢顧客的訂單及支付情況，並按訂單日期倒序排列。

```
SELECT  
  
o.Order_ID,  
  
o.Order_Date,  
  
p.Payment_Status  
FROM  
  
Orders o  
JOIN PaymentInfo p ON o.Order_ID = p.Order_ID  
WHERE  
  
o.Customer_ID = 1  
ORDER BY  
  
o.Order Date DESC;
```

14. 查詢某產品的退換貨情況：

說明：查詢每個產品的退換貨次數，通過 COUNT() 聚合函數來計算退換貨的數量。

```
SELECT  
  
p.Product_Name,  
  
COUNT(r.Return_ID) AS Return_Count  
FROM  
  
Returns r  
JOIN Products p ON r.Inventory_ID = p.Product_ID  
GROUP BY  
  
p.Product Name;
```

15. 查詢每個產品類別的銷售總額：

說明：計算每個產品類別的銷售總額，使用 JOIN 結合產品與類別資料，並使用 SUM() 聚合。

```
SELECT
    c.Category_Name,
    SUM(od.Quantity * od.Unit_Price) AS Total_Sales
FROM
    OrderDetails od
JOIN Products p ON od.Inventory_ID = p.Product_ID
JOIN Category c ON p.Category_ID = c.Category_ID
GROUP BY
    c.Category_Name;
```

六、資料載入

6.1 資料載入腳本

1. Category data :

```
INSERT INTO category (category_id, category_name) VALUES ('CA_0001',  
'Electronics');  
  
INSERT INTO category (category_id, category_name) VALUES ('CA_0002',  
'Clothing');  
  
INSERT INTO category (category_id, category_name) VALUES ('CA_0003', 'Shoes');  
  
INSERT INTO category (category_id, category_name) VALUES ('CA_0004',  
'Accessories');  
  
INSERT INTO category (category_id, category_name) VALUES ('CA_0005', 'Beauty');  
  
..... (共有 20 筆資料)
```

2. Warehouses data :

```
INSERT INTO warehouses (Warehouse ID, Location) VALUES ('W_0001', '217 Willow  
Creek Drive');  
  
INSERT INTO warehouses (Warehouse ID, Location) VALUES ('W_0002', '218 Willow  
Creek Drive');  
  
INSERT INTO warehouses (Warehouse ID, Location) VALUES ('W_0003', '219 Willow  
Creek Drive');  
  
..... (共有 5 筆資料)
```

3. Suppliers data :

```
INSERT INTO suppliers (supplier name, supplier phone, supplier email) VALUES  
('Kanoodle', '895-783-2524', 'gbutchers0@hostgator.com');  
  
INSERT INTO suppliers (supplier name, supplier phone, supplier email) VALUES  
('Lego Group', '483-849-9508', 'tgrimsdykel@spiegel.de');  
  
INSERT INTO suppliers (supplier name, supplier phone, supplier email) VALUES  
('Brooks Running Company', '315-374-9120', 'kpriscott2@go.com');  
  
INSERT INTO suppliers (supplier name, supplier phone, supplier email) VALUES  
('Luxottica Group', '254-231-7491', 'hblundin3@sohu.com');  
  
INSERT INTO suppliers (supplier name, supplier phone, supplier email) VALUES  
('L'Oreal', '140-896-7254', 'gblaske4@infoseek.co.jp');  
  
INSERT INTO suppliers (supplier name, supplier phone, supplier email) VALUES  
('Dyson Ltd', '516-373-3554', 'rcrowcher5@hostgator.com');  
  
..... (共有 50 筆資料)
```

4. Products data :

```

INSERT INTO
products( product name,product description,price,supplier ID,category ID )
VALUES ('Garnier Fructis Shampoo','Revitalizing shampoo for all hair
types',18.56,'S_0005','CA_0005');

INSERT INTO
products( product name,product description,price,supplier ID,category ID )
VALUES ('Apple iPhone 15 Pro Max','Latest flagship
smartphone',1199,'S_0022','CA_0001');

INSERT INTO
products( product name,product description,price,supplier ID,category ID )
VALUES ('Nike Air Force 1','Classic sneakers',99.99,'S_0025','CA_0003');

INSERT INTO
products( product name,product description,price,supplier ID,category ID )
VALUES ('Levi's 501 Jeans','Iconic denim jeans',79.99,'S_0031','CA_0002');

INSERT INTO
products( product name,product description,price,supplier ID,category ID )
VALUES ('Ray-Ban Wayfarer Sunglasses','Classic
sunglasses',159.99,'S_0040','CA_0004');

..... (共有 243 筆資料)

```

5. Inventory data :

```

INSERT INTO Inventory( Product ID,Quantity,Warehouse ID ) VALUES
('P_0001',385,'W_0001');

INSERT INTO Inventory( Product ID,Quantity,Warehouse ID ) VALUES
('P_0002',143,'W_0002');

INSERT INTO Inventory( Product ID,Quantity,Warehouse ID ) VALUES
('P_0003',179,'W_0002');

INSERT INTO Inventory( Product ID,Quantity,Warehouse ID ) VALUES
('P_0004',53,'W_0005');

INSERT INTO Inventory( Product ID,Quantity,Warehouse ID ) VALUES
('P_0005',212,'W_0001');

..... (共有 243 筆資料)

```

6. Customer data :

```

INSERT INTO customers (Customer Name, Customer Phone, Customer Email, Country,
City, Address) VALUES ('Tymon Piscotti', '810-721-8788',
'tpiscotti0@wikipedia.org', 'China', 'Gongchang Zhen', '884 Scott Circle');

INSERT INTO customers (Customer Name, Customer Phone, Customer Email, Country,
City, Address) VALUES ('Abdel Jephcott', '941-908-9999',
'ajephcott1@goodreads.com', 'Bolivia', 'Nueva Manoa', '744 Bashford Street');

INSERT INTO customers (Customer Name, Customer Phone, Customer Email, Country,
City, Address) VALUES ('Bamby Craker', '715-166-5128', 'bcraker2@comsenz.com',
'Poland', 'Barlinek', '691 Green Ridge Way');

INSERT INTO customers (Customer Name, Customer Phone, Customer Email, Country,
City, Address) VALUES ('Leah Saunder', '484-415-6859', 'lsaunder3@g.co',
'Philippines', 'President Roxas', '5 Dwight Way');

```

```
INSERT INTO customers (Customer Name, Customer Phone, Customer Email, Country, City, Address) VALUES ('Pip Eiler', '150-716-3464', 'peiler4@ocn.ne.jp', 'Mexico', 'Providencia', '0 Briar Crest Way');
```

..... (共有 500 筆資料)

7. Orders data :

```
INSERT INTO orders (Customer ID, Order Date, Order State) VALUES ('CU 0359', '2023/12/12 13:17', 'completed');
```

```
INSERT INTO orders (Customer ID, Order Date, Order State) VALUES ('CU 0317', '2023/12/12 19:20', 'returned');
```

```
INSERT INTO orders (Customer ID, Order Date, Order State) VALUES ('CU 0390', '2023/12/12 20:44', 'completed');
```

```
INSERT INTO orders (Customer ID, Order Date, Order State) VALUES ('CU 0275', '2023/12/13 00:24', 'returned');
```

```
INSERT INTO orders (Customer ID, Order Date, Order State) VALUES ('CU 0123', '2023/12/13 17:23', 'completed');
```

```
INSERT INTO orders (Customer ID, Order Date, Order State) VALUES ('CU 0265', '2023/12/13 20:10', 'completed');
```

..... (共有 1000 筆資料)

8. OrderDetails data :

```
INSERT INTO orderdetails (Order ID, Inventory ID, Quantity) VALUES (1, 'I_0019', 1);
```

```
INSERT INTO orderdetails (Order ID, Inventory ID, Quantity) VALUES (2, 'I_0013', 3);
```

```
INSERT INTO orderdetails (Order ID, Inventory ID, Quantity) VALUES (2, 'I_0209', 5);
```

```
INSERT INTO orderdetails (Order ID, Inventory ID, Quantity) VALUES (3, 'I_0090', 4);
```

```
INSERT INTO orderdetails (Order ID, Inventory ID, Quantity) VALUES (3, 'I_0092', 1);
```

..... (共有 1000 筆資料)

9. Returns data :

```
INSERT INTO returns (Order_ID, Return_Date) VALUES (2, '2023/12/16 19:20');
```

```
INSERT INTO returns (Order_ID, Return_Date) VALUES (4, '2023/12/15 00:24');
```

```
INSERT INTO returns (Order_ID, Return_Date) VALUES (28, '2023/12/23 01:37');
```

```
INSERT INTO returns (Order_ID, Return_Date) VALUES (31, '2023/12/24 00:25');
```

```
INSERT INTO returns (Order_ID, Return_Date) VALUES (38, '2023/12/23 22:06');
```

..... (共有 208 筆資料)

10. PaymentInfo data :

```
INSERT INTO paymentinfo( Order ID,Payment Method,Payment Date,Payment State )
VALUES (1,'cash','2023/12/15 13:17','completed');

INSERT INTO paymentinfo( Order ID,Payment Method,Payment Date,Payment State )
VALUES (2,'credit card','2023/12/13 19:20','returned');

INSERT INTO paymentinfo( Order ID,Payment Method,Payment Date,Payment State )
VALUES (3,'transferred','2023/12/14 20:44','completed');

INSERT INTO paymentinfo( Order ID,Payment Method,Payment Date,Payment State )
VALUES (4,'cash','2023/12/18 00:24','returned');

INSERT INTO paymentinfo( Order ID,Payment Method,Payment Date,Payment State )
VALUES (5,'transferred','2023/12/17 17:23','completed');

..... (共有 1000 筆資料)
```








6.2 範例資料展示

1. category :

Data Output Messages Notifications		
	category_id [PK] character varying (10)	category_name character varying (100)
1	CA_0001	Electronics
2	CA_0002	Clothing
3	CA_0003	Shoes
4	CA_0004	Accessories
5	CA_0005	Beauty
6	CA_0006	Home Goods
7	CA_0007	Kitchenware
8	CA_0008	Food
9	CA_0009	Beverages
10	CA_0010	Books
11	CA_0011	Stationery
12	CA_0012	Toys
13	CA_0013	Sports Equipment
14	CA_0014	Gardening Supplies
15	CA_0015	Pet Supplies
16	CA_0016	Automotive
17	CA_0017	Hardware
18	CA_0018	Health & Wellness
19	CA_0019	Pharmacy
20	CA_0020	Grocery

Total rows: 20 of 20 Query complete 00:00:00.066 Ln 1, Col 1

2. customers :

Data Output Messages Notifications							
	      						
	customer_id [PK] character varying (10)	customer_name character varying (100)	customer_phone character varying (15)	customer_email character varying (100)	country character varying (50)	city character varying (50)	address character varying (255)
1	CU_0001	Tymon Piscotti	810-721-8788	tpiscotti0@wikipedia.org	China	Gongchang Zhen	884 Scott Circle
2	CU_0002	Abdel Jephcott	941-908-9999	ajephcott1@goodreads.com	Bolivia	Nueva Manoa	744 Bashford Street
3	CU_0003	Bamby Craker	715-166-5128	bcraker2@comsenz.com	Poland	Barlinek	691 Green Ridge Way
4	CU_0004	Leah Saunder	484-415-6859	lsaunder3@g.co	Philippines	President Roxas	5 Dwight Way
5	CU_0005	Pip Eiler	150-716-3464	peiler4@ocn.ne.jp	Mexico	Providencia	0 Briar Crest Way
6	CU_0006	Miner Shemmin	444-956-3833	mshemmin5@cpanel.net	Greece	Stalis	171 Oakridge Road
7	CU_0007	Dot Martina	477-492-7081	dmartina6@chicagotribune.com	Philippines	Sison	0871 Hallows Plaza
8	CU_0008	Care Souch	320-798-5914	csouch7@163.com	Sweden	Själavad	666 Lakewood Way
9	CU_0009	Salaith Jillett	914-126-6559	sjillett8@myspace.com	United States	Bronx	5125 American Parkway
10	CU_0010	Wain Venediktov	426-710-7163	wvenediktov9@t-online.de	China	Fubei	29 Tennessee Trail
11	CU_0011	Gillan Perkin	406-722-3393	gperkina@ow.ly	Brazil	Rio Branco do Sul	92 Fallview Avenue
12	CU_0012	Winnah MacBean	987-250-7998	wmacbeanb@vistaprint.com	Indonesia	Banyuurip	83801 School Way
13	CU_0013	Amber Leving	177-433-6501	alevingc@ebay.co.uk	Georgia	P'rimorsk'oe	058 Myrtle Street
14	CU_0014	Michal Gardener	103-622-5341	mgardenerd@hexun.com	China	Ningde	40402 Vernon Circle
15	CU_0015	Woodman Doerffer	591-526-6554	wdoerffer@cnr.com	Indonesia	Kanugrahan	67 Lakewood Drive
16	CU_0016	Cos Cotgrave	158-277-1402	ccotgravef@weather.com	Indonesia	Weetobula	1 Bunker Hill Junction
17	CU_0017	Nye Westerman	623-182-9526	nwestermang@google.de	Philippines	Mapalad	4 Fulton Circle
18	CU_0018	Ferd Scorah	464-236-2860	fscorahh@yolasite.com	Peru	San Pedro de Lloc	395 Messerschmidt Juncti...
19	CU_0019	Phyllys Tyndall	361-792-4766	ptyndall@home.pl	Indonesia	Sepanjang	3049 Shelley Parkway
20	CU_0020	Arv Millier	361-656-9585	amillierj@ted.com	Philippines	Tayasan	07571 Prairieview Alley
21	CU_0021	Sherry Eggleston	327-721-5494	segglesonk@wufoo.com	Cameroon	Dschang	2 Emmet Trail
22	CU_0022	Shelagh Shurlock	379-409-4716	sshurlockl@netvibes.com	Indonesia	Kabongan Kidul	8 Jenifer Center
23	CU_0023	Bard Cassidv	168-882-3227	bcassidvm@bloos.com	Morocco	Senada	9 Westridge Way
Total rows: 500 of 500 Query complete 00:00:00.082 Ln 1, Col 1							

3. inventory :

Data Output Messages Notifications				
	inventory_id [PK] character varying (10)	product_id character varying (10)	quantity integer	warehouse_id character varying (10)
1	I_0001	P_0001	385	W_0001
2	I_0002	P_0002	143	W_0002
3	I_0003	P_0003	179	W_0002
4	I_0004	P_0004	53	W_0005
5	I_0005	P_0005	212	W_0001
6	I_0006	P_0006	320	W_0004
7	I_0007	P_0007	118	W_0005
8	I_0008	P_0008	214	W_0003
9	I_0009	P_0009	261	W_0002
10	I_0010	P_0010	442	W_0001
11	I_0011	P_0011	162	W_0001
12	I_0012	P_0012	68	W_0002
13	I_0013	P_0013	387	W_0004
14	I_0014	P_0014	429	W_0003
15	I_0015	P_0015	346	W_0002
16	I_0016	P_0016	461	W_0004
17	I_0017	P_0017	252	W_0002
18	I_0018	P_0018	308	W_0001
19	I_0019	P_0019	466	W_0003
20	I_0020	P_0020	286	W_0001
21	I_0021	P_0021	293	W_0002
22	I_0022	P_0022	172	W_0005
23	I_0023	P_0023	307	W_0005
Total rows: 243 of 243 Query complete 00:00:00.054 Ln 1, Col 1				

4. orderdetails :

Data Output Messages Notifications				
	order_id [PK] integer	inventory_id [PK] character varying (10)	quantity integer	unit_price numeric (10,2)
1	1	I_0019	1	3.99
2	2	I_0013	3	9.99
3	2	I_0209	5	399.00
4	3	I_0090	4	39.99
5	3	I_0092	1	129.99
6	3	I_0139	2	1.99
7	3	I_0201	2	14.99
8	3	I_0243	1	79.99
9	4	I_0021	5	1299.00
10	4	I_0183	4	10.99
11	5	I_0013	3	9.99
12	5	I_0020	2	29.99
13	5	I_0088	5	24.99
14	6	I_0012	2	24.99
15	6	I_0032	4	19.99
16	6	I_0045	5	3.99
17	6	I_0158	4	39.99
18	6	I_0159	5	89.99
19	7	I_0029	2	1.00
20	7	I_0116	5	999.00
21	8	I_0046	3	1.75
22	8	I_0201	1	14.99
23	8	I_0206	3	199.99
Total rows: 1000 of 2984 Query complete 00:00:00.066 Ln 1, Col 1				

5. orders :

Data Output

Messages

Notifications














	order_id [PK] integer	customer_id character varying (10)	order_date timestamp without time zone	total_amount numeric (10,2)	order_state character varying (20)
1	1	CU_0359	2023-12-12 13:17:00	3.99	completed
2	2	CU_0317	2023-12-12 19:20:00	2024.97	returned
3	3	CU_0390	2023-12-12 20:44:00	403.90	completed
4	4	CU_0275	2023-12-13 00:24:00	6538.96	returned
5	5	CU_0123	2023-12-13 17:23:00	214.90	completed
6	6	CU_0265	2023-12-13 20:10:00	759.80	completed
7	7	CU_0319	2023-12-14 01:50:00	4997.00	completed
8	8	CU_0347	2023-12-14 11:25:00	620.21	completed
9	9	CU_0402	2023-12-14 22:52:00	345.88	completed
10	10	CU_0232	2023-12-15 12:11:00	1214.88	completed
11	11	CU_0432	2023-12-15 23:07:00	249.95	completed
12	12	CU_0466	2023-12-16 01:49:00	2719.79	completed
13	13	CU_0435	2023-12-16 03:06:00	1019.40	completed
14	14	CU_0308	2023-12-16 06:24:00	17.70	completed
15	15	CU_0184	2023-12-16 07:02:00	557.87	completed
16	16	CU_0380	2023-12-16 07:59:00	2058.89	completed
17	17	CU_0060	2023-12-16 08:46:00	199.96	completed
18	18	CU_0051	2023-12-16 17:22:00	186.94	completed
19	19	CU_0103	2023-12-16 22:57:00	7882.91	completed
20	20	CU_0400	2023-12-17 04:40:00	227.94	completed
21	21	CU_0266	2023-12-17 05:02:00	460.88	completed
22	22	CU_0354	2023-12-17 05:15:00	5026.45	completed
23	23	CU_0130	2023-12-17 23:42:00	2096.86	completed

Total rows: 1000 of 1000 Query complete 00:00:00.069 Ln 1, Col 1

6. paymentinfo :

Data Output Messages Notifications					
	payment_id [PK] character varying (10)	order_id integer	payment_method character varying (50)	payment_date timestamp without time zone	payment_state character varying (20)
1	P_0001		1 cash	2023-12-15 13:17:00	completed
2	P_0002		2 credit card	2023-12-13 19:20:00	returned
3	P_0003		3 transferred	2023-12-14 20:44:00	completed
4	P_0004		4 cash	2023-12-18 00:24:00	returned
5	P_0005		5 transferred	2023-12-17 17:23:00	completed
6	P_0006		6 cash	2023-12-16 20:10:00	completed
7	P_0007		7 credit card	2023-12-19 01:50:00	completed
8	P_0008		8 credit card	2023-12-16 11:25:00	completed
9	P_0009		9 transferred	2023-12-15 22:52:00	completed
10	P_0010		10 transferred	2023-12-18 12:11:00	completed
11	P_0011		11 transferred	2023-12-20 23:07:00	completed
12	P_0012		12 cash	2023-12-18 01:49:00	completed
13	P_0013		13 credit card	2023-12-20 03:06:00	completed
14	P_0014		14 cash	2023-12-18 06:24:00	completed
15	P_0015		15 cash	2023-12-21 07:02:00	completed
16	P_0016		16 cash	2023-12-19 07:59:00	completed
17	P_0017		17 cash	2023-12-21 08:46:00	completed
18	P_0018		18 credit card	2023-12-19 17:22:00	completed
19	P_0019		19 transferred	2023-12-19 22:57:00	completed
20	P_0020		20 cash	2023-12-19 04:40:00	completed
21	P_0021		21 cash	2023-12-19 05:02:00	completed
22	P_0022		22 credit card	2023-12-18 05:15:00	completed
23	P_0023		23 credit card	2023-12-18 23:42:00	completed
Total rows: 1000 of 1000 Query complete 00:00:00.062 Ln 1, Col 1					

7. products :

Data Output Messages Notifications						
	      					
	 product_id [PK] character varying (10)	 product_name character varying (100)	 product_description character varying (255)	 price numeric (10,2)	 supplier_id character varying (10)	 category_id character varying (10)
1	P_0001	Garnier Fructis Shampoo	Revitalizing shampoo for all hair types	18.56	S_0005	CA_0005
2	P_0002	Apple iPhone 15 Pro Max	Latest flagship smartphone	1199.00	S_0022	CA_0001
3	P_0003	Nike Air Force 1	Classic sneakers	99.99	S_0025	CA_0003
4	P_0004	Levi's 501 Jeans	Iconic denim jeans	79.99	S_0031	CA_0002
5	P_0005	Ray-Ban Wayfarer Sunglasses	Classic sunglasses	159.99	S_0040	CA_0004
6	P_0006	KitchenAid Stand Mixer	Versatile kitchen appliance	399.99	S_0007	CA_0007
7	P_0007	Nestle Chocolate Bar	Delicious chocolate treat	1.99	S_0018	CA_0008
8	P_0008	Coca-Cola	Refreshing beverage	1.50	S_0029	CA_0009
9	P_0009	Harry Potter and the Sorcerer's Stone	Classic fantasy novel	12.99	S_0037	CA_0010
10	P_0010	Bic Ballpoint Pen	Classic ballpoint pen	0.50	S_0046	CA_0011
11	P_0011	LEGO Building Blocks	Creative building toys	29.99	S_0002	CA_0012
12	P_0012	Adidas Soccer Ball	Official soccer ball	24.99	S_0011	CA_0013
13	P_0013	Gardening Gloves	Durable gardening gloves	9.99	S_0020	CA_0014
14	P_0014	Purina Pro Plan Dog Food	High-quality dog food	29.99	S_0029	CA_0015
15	P_0015	Odyssey Battery	Reliable car battery	129.99	S_0038	CA_0016
16	P_0016	Stanley Hammer	Versatile tool	14.99	S_0047	CA_0017
17	P_0017	Nature's Way Vitamin C Supplement	Immune support supplement	19.99	S_0005	CA_0018
18	P_0018	Advil Ibuprofen	Pain reliever	9.99	S_0014	CA_0019
19	P_0019	Organic Valley Milk	Essential dairy product	3.99	S_0023	CA_0020
20	P_0020	L'oreal Paris Revitalift Triple Power Anti-Ageing Moisturiz...	Anti-aging moisturizer	29.99	S_0005	CA_0005
21	P_0021	Samsung Galaxy S24 Ultra	High-end smartphone	1299.00	S_0041	CA_0001
22	P_0022	Adidas Ultraboost 23	Running shoes	179.99	S_0008	CA_0003
23	P_0023	H&M Denim Jacket	Stylish denim jacket	49.99	S_0017	CA_0002
Total rows: 243 of 243 Query complete 00:00:00.120 Ln 1, Col 1						

8. returns :

Data Output Messages Notifications			
	return_id [PK] character varying (10)	order_id integer	return_date timestamp without time zone
1	R_0001	2	2023-12-16 19:20:00
2	R_0002	4	2023-12-15 00:24:00
3	R_0003	28	2023-12-23 01:37:00
4	R_0004	31	2023-12-24 00:25:00
5	R_0005	38	2023-12-23 22:06:00
6	R_0006	43	2023-12-22 04:46:00
7	R_0007	46	2023-12-26 00:18:00
8	R_0008	50	2023-12-26 05:53:00
9	R_0009	53	2023-12-24 16:44:00
10	R_0010	54	2023-12-26 23:05:00
11	R_0011	56	2023-12-27 06:16:00
12	R_0012	58	2023-12-26 11:20:00
13	R_0013	61	2023-12-30 10:35:00
14	R_0014	63	2024-01-01 08:28:00
15	R_0015	77	2024-01-06 22:09:00
16	R_0016	79	2024-01-05 09:29:00
17	R_0017	81	2024-01-06 23:02:00
18	R_0018	104	2024-01-15 04:32:00
19	R_0019	106	2024-01-17 08:43:00
20	R_0020	112	2024-01-19 00:44:00
21	R_0021	113	2024-01-21 03:26:00
22	R_0022	124	2024-01-24 02:26:00
23	R_0023	125	2024-01-24 03:47:00
Total rows: 208 of 208 Query complete 00:00:00.050 Ln 1, Col 1			

9. warehouses :

Data Output Messages Notifications		
	warehouse_id [PK] character varying (10)	location character varying (100)
1	W_0001	217 Willow Creek Drive
2	W_0002	218 Willow Creek Drive
3	W_0003	219 Willow Creek Drive
4	W_0004	220 Willow Creek Drive
5	W_0005	221 Willow Creek Drive
Total rows: 5 of 5 Query complete 00:00:00.052 Ln 2, Col 27		

10. suppliers :

Data Output Messages Notifications				
	supplier_id [PK] character varying (10)	supplier_name character varying (100)	supplier_phone character varying (15)	supplier_email character varying (100)
1	S_0001	Kanoodle	895-783-2524	gbutchers0@hostgator.com
2	S_0002	Lego Group	483-849-9508	tgrimsdyke1@spiegel.de
3	S_0003	Brooks Running Company	315-374-9120	kpriscott2@go.com
4	S_0004	Luxottica Group	254-231-7491	hblundin3@sohu.com
5	S_0005	L'Oreal	140-896-7254	gblaske4@infoseek.co.jp
6	S_0006	Dyson Ltd	516-373-3554	rcrowcher5@hostgator.com
7	S_0007	Instant Pot Company	302-652-4937	tatton6@free.fr
8	S_0008	Houghton Mifflin Harcourt	362-951-3688	nsutch7@gmpg.org
9	S_0009	Keurig Dr Pepper	649-812-7467	edrayton8@harvard.edu
10	S_0010	Scholastic Corporation	365-607-2270	wkitchingman9@tmall.com
11	S_0011	Peloton Interactive	491-962-4722	fmdonagha@boston.com
12	S_0012	Weber-Stephen Products Co.	770-991-4539	lmccarlichb@spotify.com
13	S_0013	Royal Canin SAS	652-297-2741	bwyrec@redcross.org
14	S_0014	Odyssey Battery Company	357-878-4937	chabbershond@wiley.com
15	S_0015	Stanley Black & Decker	860-919-9095	tohoolahane@sciencedaily.com
16	S_0016	General Motors	998-401-4261	kemmaf@ebay.com
17	S_0017	Levi Strauss & Co.	659-938-5055	dmcillamg@is.gd
18	S_0018	GNC Holdings LLC	414-783-9298	lthemih@rambler.ru
19	S_0019	Pfizer Inc.	614-604-3474	jplewmani@g.co
20	S_0020	The Scotts Miracle-Gro Company	888-684-4817	sbothbiej@biglobe.ne.jp
21	S_0021	Samsung Electronics	976-745-2302	sgirardk@cisco.com
22	S_0022	Apple Inc.	285-948-5214	jroughl@blinklist.com
23	S_0023	Thrive Market	940-327-5967	mspadarom@skvpe.com
Total rows: 50 of 50 Query complete 00:00:00.075 Ln 1, Col 1				

6.3 統計報告

Products	ROWS: 243	允許空值欄位: product_description		空值比例: 0%
Category	ROWS: 20	無空值		
Inventory	ROWS: 243	無空值		
Orders	ROWS: 1000	無空值		
Customer	ROWS: 500	無空值		
Orderdetails	ROWS: 2984	無空值		
Suppliers	ROWS: 50	無空值		
Returns	ROWS: 208	無空值		
Warehouses	ROWS: 5	無空值		
PaymentInfo	ROWS: 1000	無空值		

七、測試與查詢優化

7.1 功能性測試報告

1. 測試觸發：

```
Query Query History
1  /*銷貨手續*/
2
3  BEGIN TRANSACTION;
4
5  ✓ INSERT INTO Orders (Customer_ID, Order_State)
6  VALUES ('CU_0033', 'confirmed');
7
8  ✓ INSERT INTO OrderDetails (Order_ID, Inventory_ID, Quantity)
9  VALUES (
10     LASTVAL(),
11     'I_0028',
12     5
13 );
14
15 ✓ UPDATE Inventory
16 SET Quantity = Quantity - 5
17 WHERE Inventory_ID = 'I_0028';
18
19 ✓ INSERT INTO PaymentInfo (Order_ID, Payment_Method, Payment_State)
20 VALUES (
21     LASTVAL(),
22     'transferred',
23     'processed'
24 );
25
26 COMMIT;
```

```
Data Output Messages Notifications
COMMIT

Query returned successfully in 86 msec.
```

2. 測試限制：

- 目標：測試 unique_product_name 是否確保每個產品的名稱唯一

```
Query Query History
1  INSERT INTO products( product_name,product_description,price,supplier_ID,category_ID ) VALUES ('LEGO Friends',
```

```
Data Output Messages Notifications
ERROR: Key (product_name)=(LEGO Friends) already exists.duplicate key value violates unique constraint "unique_product_name"
```

- 目標：測試 `unique_return_order` 是否確保退貨紀錄中訂單號碼只出現一次

Query Query History

```
1 INSERT INTO returns(Order_ID,Return_Date ) VALUES(419,'2024/5/9 06:26');
```

Data Output Messages Notifications

ERROR: Key (order_id)=(419) already exists.duplicate key value violates unique constraint "unique_return_order"

- 目標：測試 `chk_product_price` 是否確保產品價格大於或等於 0

Query Query History

```
1 UPDATE products SET price = -1 WHERE product_id = 'P_0001';
```

Data Output Messages Notifications

ERROR: Failing row contains (P_0001, Garnier Fructis Shampoo, Revitalizing shampoo fo

ERROR: new row for relation "products" violates check constraint "chk_product_price"

- 目標：測試 `chk_inventory_quantity` 是否確保庫存數量大於或等於 0

Query Query History

```
1 UPDATE inventory SET quantity = -1 WHERE inventory_id = 'I_0001';
```

Data Output Messages Notifications

ERROR: Failing row contains (I_0001, P_0001, -1, W_0001).new row for relation "inventory" v

ERROR: new row for relation "inventory" violates check constraint "chk_inventory_quantity"

- 目標：測試 `unique_order_item` 是否確保同一筆訂單不出現多筆相同的 `Inventory_ID` 紀錄

Query Query History

```
1 INSERT INTO orderdetails(Order_ID,Inventory_ID,Quantity ) VALUES (2,'I_0013',3);
```

Data Output Messages Notifications

ERROR: Key (order_id, inventory_id)=(2, I_0013) already exists.duplicate key value violates unique constraint "orderdetails_pkey"

ERROR: duplicate key value violates unique constraint "orderdetails_pkey"

3. 測試功能：

- 說明：可以看到不用輸入 `order_id` 跟 `total_amount` 是因為 trigger 啟動了 function：`update_order_total`，`order_id` 是使用 `DEFAULT`

功能加上 sequence

Query Query History

```
1 SELECT * FROM public.orders
2 ORDER BY order_id DESC LIMIT 100
```

Data Output Messages Notifications

	order_id [PK] integer	customer_id character varying (10)	order_date timestamp without time zone	total_amount numeric (10,2)	order_state character varying (20)
1	1003	CU_0033	2024-12-17 18:58:51.32408	74.95	confirmed

- 說明：這裡的 unit price 也是因為 trigger 啟動了自動查詢產品價格的 function: update_unit_price()

Query Query History

```
1 SELECT * FROM public.orderdetails
2 ORDER BY order_id DESC, inventory_id DESC LIMIT 100
```

Data Output Messages Notifications

	order_id [PK] integer	inventory_id [PK] character varying (10)	quantity integer	unit_price numeric (10,2)
1	1003	I_0028	5	14.99

7.2 查詢效能報告

```
a=# EXPLAIN ANALYZE SELECT * FROM orderdetails
a=# ;
                                QUERY PLAN
-----
Seq Scan on orderdetails (cost=0.00..49.84 rows=2984 width=21) (actual time=0.005..0.112 rows=2984 loops=1)
Planning Time: 0.053 ms
Execution Time: 0.167 ms
(3 筆資料)
```

```
a=# EXPLAIN ANALYZE SELECT * FROM orders;
                                QUERY PLAN
-----
Seq Scan on orders (cost=0.00..48.00 rows=1000 width=35) (actual time=0.281..0.901 rows=1000 loops=1)
Planning Time: 2.096 ms
Execution Time: 0.931 ms
(3 筆資料)
```

```
a=# EXPLAIN ANALYZE SELECT * FROM products;
                                QUERY PLAN
-----
Seq Scan on products (cost=0.00..6.43 rows=243 width=69) (actual time=0.008..0.078 rows=243 loops=1)
Planning Time: 2.118 ms
Execution Time: 0.088 ms
(3 筆資料)
```