ISYS 864 - Spring 2023

Inventory Management Database

Project Report

Submitted by-

**Saksham Motwani**

# Under the supervision of -

**Professor Guillaume Faddoul**

# 

**Contents**

Information about the organization..................................................................................................3

List of Entities and their Attributes..................................................................................................3

Business Rules.................................................................................................................................5

Entity Relationship Diagram............................................................................................................6

3NF Relational Model......................................................................................................................6

SQL Code.........................................................................................................................................7

Creating Tables

Creating Triggers

Loading Data in Tables

Data Analysis ................................................................................................................................17

8 Retrieval queries

Output of the queries

Purpose of the queries

Conclusion.....................................................................................................................................20

**Information about the organization**



Name: SmartShopTech

Business Sector: E-commerce (Consumer Electronics)

About SmartShopTech: We are an early-stage online retailer specializing in consumer electronics and household appliances. Headquartered in the City of Industry, California, currently we only service addresses in the US west coast (Washington, Oregon, California, Nevada, Arizona), and as of now, only have one warehouse. Our USP is that we offer competitive pricing (and price matching), offer a large selection of products with many brands, and guarantee delivery in 24 hours.

We created a simple Inventory Management Database to store data about our suppliers, products, customers, and orders, and manage the flow of goods from our suppliers to our warehouse, and from our warehouse to the customer. The database will help synchronize orders with inventory and generate sales reports that offer insights for better business decisions.

Reasons for picking this focus area: Inventory Management Systems widely used in businesses, and hence this project will be relevant to real-world scenarios. Moreover, all our group members were comfortable with and had some knowledge about this topic, and contributed fairly based on our understanding.

**List of entities and their attributes**

|  |  |  |
| --- | --- | --- |
| **Sr No.** | **Entities** | **Attributes** |
| 1. | Supplier | **Supplier ID (PK)**  Supplier Name  Contact Name  Phone Number  Email Address  Address (Street, City, State, Zip, Country) |
| 2. | Purchasing Order | **Purchasing Order ID (PK)**  Supplier ID  Purchase Date  Shipped Date  Received Date  Total  Discount  Tax  Grand Total |
| 3. | PO Details (Associative Entity) | **Purchasing Order ID, Product ID (PK)**  Purchase Price  Quantity Purchased |
| 4. | Product | **Product ID (PK)**  Product Category  Brand  Product Name  Product Description  Purchase Price  Sales Price  Quantity Available |
| 5. | Customer | **Customer ID (PK)**  Customer Name  Phone Number  Email Address  Address (Street, City, State, Zip, Country) |
| 6. | Orders | **Order ID (PK)**  Customer ID  Order Date  Shipped Date  Total  Discount  Tax  Grand Total |
| 7. | Order Details (Associative Entity) | **Order ID, Product ID (PK)**  Sales Price  Quantity Sold |

**Business Rules**

**(1:N)**

**Each supplier can fulfill any number of purchasing orders.**

In other words, a supplier can exist in the database irrespective of having fulfilled a purchasing order with us.

**Each purchasing order must be fulfilled by only one supplier.**

In other words, a purchasing order cannot be fulfilled by multiple suppliers.

**(M:N)**

**Each purchasing order must contain at least one product (bought from the supplier).**

In other words, there cannot be a purchasing order without any product.

**Each product can be a part of any number of purchasing orders.**

In other words, a product can exist in our database whether it is a part of a purchasing order (i.e. products we have already bought), or whether it is not a part of a purchasing order (i.e. products we haven’t yet bought but intend to buy).

**(M:N)**

**Each order must contain at least one product (sold to the customer).**

In other words, there cannot be a customer order without any product.

**Each product can be a part of any number of orders.**

In other words, a product can exist in our database whether it is a part of a customer order (i.e. if any customer has bought it), or whether it is not a part of a customer order (i.e. if it has not been bought by any customer).

**(1:N)**

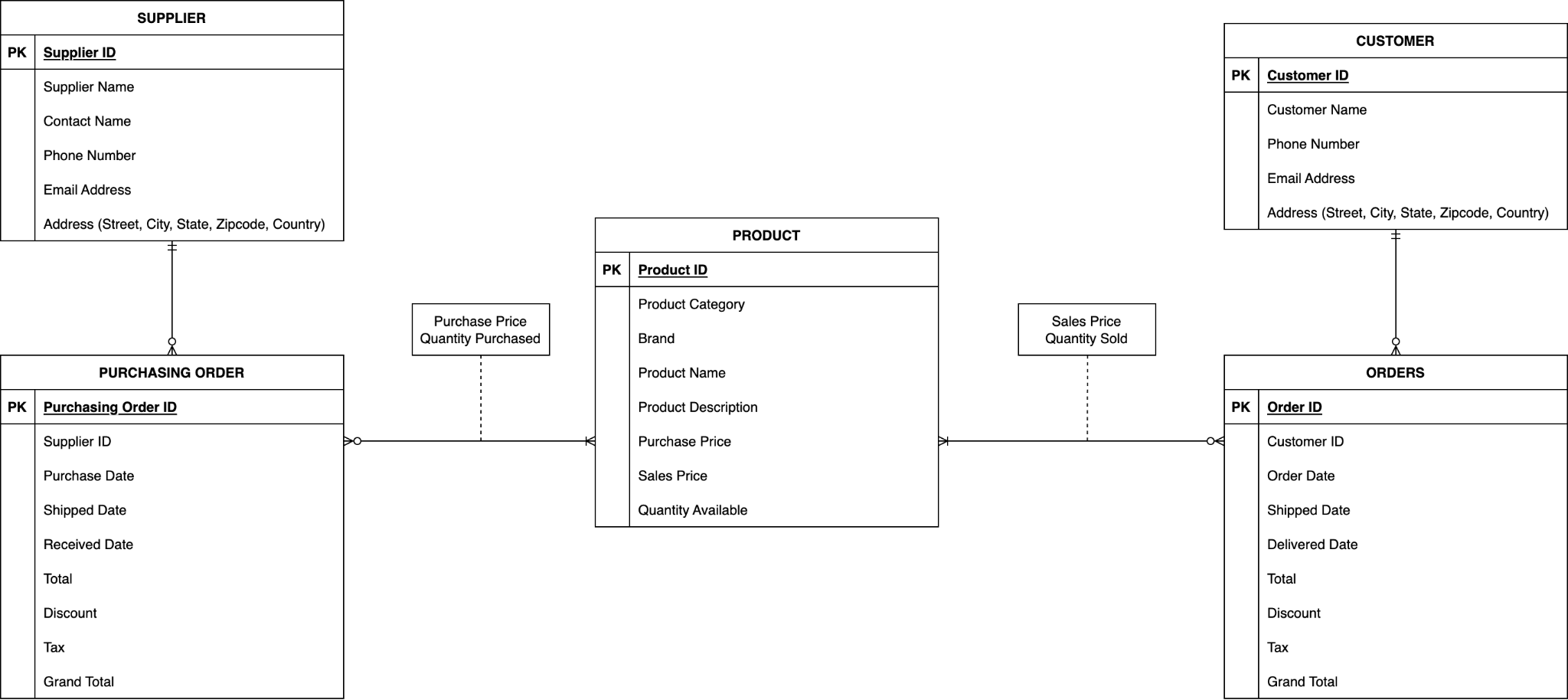
**Each customer can place any number of orders.**

In other words, a customer can exist in the database whether or not they have made a purchase. (For example: People who only create an account and not order anything)

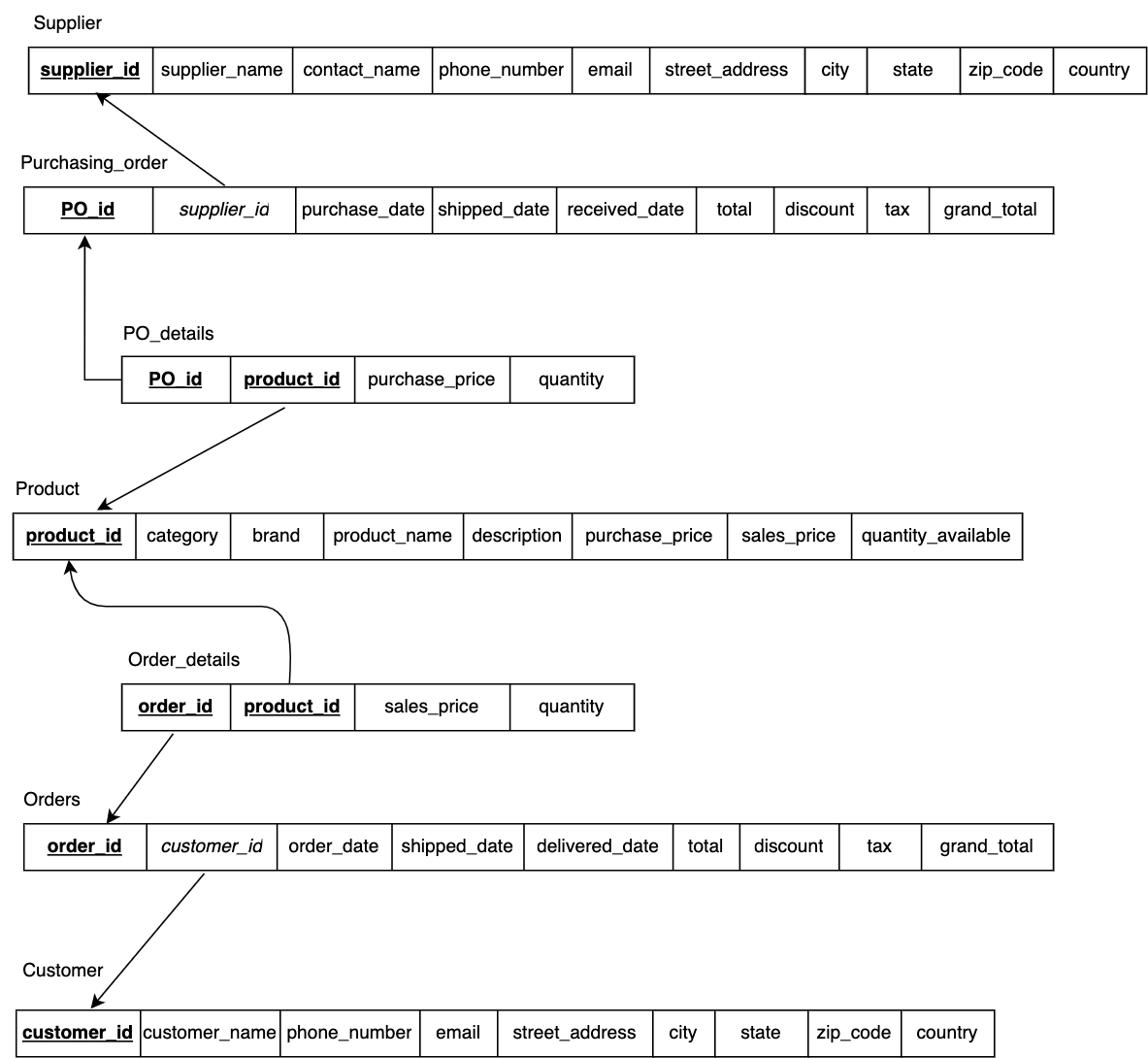
**Each order must be placed by only one customer.**

In other words, a customer order cannot be placed by multiple customers.

**Entity Relationship Diagram**



**3NF Relational Model**



**SQL Code**

**Creating Tables**

CREATE TABLE Supplier

(

supplier\_id INT NOT NULL,

supplier\_name VARCHAR(100) NOT NULL,

contact\_name VARCHAR(100) NOT NULL,

phone\_number BIGINT UNIQUE NOT NULL,

email VARCHAR(100) NOT NULL,

street\_address VARCHAR(255) NOT NULL,

city VARCHAR(50) NOT NULL,

state VARCHAR(50) NOT NULL,

zip\_code INT NOT NULL,

country VARCHAR(50) NOT NULL,

CONSTRAINT Supplier\_PK PRIMARY KEY (supplier\_id)

);

CREATE TABLE Purchasing\_order

(

PO\_id INT NOT NULL,

supplier\_id INT NOT NULL,

purchase\_date DATETIME DEFAULT CURRENT\_TIMESTAMP,

shipped\_date DATE,

received\_date DATE,

total DECIMAL(10,2),

discount DECIMAL(10,2),

tax DECIMAL(10,2),

grand\_total DECIMAL(10,2) GENERATED ALWAYS AS (total - discount + tax),

CONSTRAINT Purchasing\_order\_PK PRIMARY KEY (PO\_id),

CONSTRAINT Purchasing\_order\_FK FOREIGN KEY (supplier\_id) REFERENCES Supplier(supplier\_id) ON UPDATE CASCADE

);

CREATE TABLE Product

(

product\_id INT NOT NULL,

category VARCHAR(50) NOT NULL,

brand VARCHAR(50) NOT NULL,

product\_name VARCHAR(100) NOT NULL,

description TEXT NOT NULL,

purchase\_price DECIMAL(10,2),

sales\_price DECIMAL(10,2),

quantity\_available INT,

CONSTRAINT Product\_PK PRIMARY KEY (product\_id)

);

CREATE TABLE PO\_details

(

PO\_id INT NOT NULL,

product\_id INT NOT NULL,

purchase\_price DECIMAL(10,2) NOT NULL,

quantity INT NOT NULL,

CONSTRAINT PO\_details\_PK PRIMARY KEY (PO\_id, product\_id),

CONSTRAINT PO\_details\_FK1 FOREIGN KEY (PO\_id) REFERENCES Purchasing\_order(PO\_id) ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT PO\_details\_FK2 FOREIGN KEY (product\_id) REFERENCES Product(product\_id) ON UPDATE CASCADE ON DELETE CASCADE

);

CREATE TABLE Customer

(

customer\_id INT NOT NULL,

customer\_name VARCHAR(100),

phone\_number BIGINT UNIQUE NOT NULL,

email VARCHAR(100) NOT NULL,

street\_address VARCHAR(255) NOT NULL,

city VARCHAR(50) NOT NULL,

state VARCHAR(50) NOT NULL,

zip\_code INT NOT NULL,

country VARCHAR(50) NOT NULL,

CONSTRAINT Customer\_PK PRIMARY KEY (customer\_id)

);

CREATE TABLE Orders

(

order\_id INT NOT NULL,

customer\_id INT NOT NULL,

order\_date DATETIME DEFAULT CURRENT\_TIMESTAMP,

shipped\_date DATE,

delivered\_date DATE,

total DECIMAL(10,2),

discount DECIMAL(10,2),

tax DECIMAL(10,2),

grand\_total DECIMAL(10,2) GENERATED ALWAYS AS (total - discount + tax),

CONSTRAINT Orders\_PK PRIMARY KEY (order\_id),

CONSTRAINT Orders\_FK FOREIGN KEY (customer\_id) REFERENCES Customer(customer\_id) ON UPDATE CASCADE

);

CREATE TABLE Order\_details

(

order\_id INT NOT NULL,

product\_id INT NOT NULL,

sales\_price DECIMAL(10,2),

quantity INT NOT NULL,

CONSTRAINT Order\_details\_PK PRIMARY KEY (order\_id, product\_id),

CONSTRAINT Order\_details\_FK1 FOREIGN KEY (order\_id) REFERENCES Orders(order\_id) ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT Order\_details\_FK2 FOREIGN KEY (product\_id) REFERENCES Product(product\_id) ON UPDATE CASCADE ON DELETE CASCADE

);

**Creating Triggers**

DELIMITER $$

CREATE TRIGGER update\_inventory\_increase

AFTER INSERT ON PO\_details

FOR EACH ROW

BEGIN

UPDATE Product

SET quantity\_available = quantity\_available + NEW.quantity

WHERE product\_id = NEW.product\_id;

END $$

DELIMITER ;

This trigger is executed after every insertion on the PO\_details table. It updates (increases) the inventory count of a particular product when it is purchased from a supplier.

DELIMITER $$

CREATE TRIGGER update\_inventory\_decrease

AFTER INSERT ON Order\_details

FOR EACH ROW

BEGIN

UPDATE Product

SET quantity\_available = quantity\_available - NEW.quantity

WHERE product\_id = NEW.product\_id;

END $$

DELIMITER ;

Similarly, this trigger is executed after every insertion on the Order\_details table to update (decrease) the inventory count of a particular product when it is purchased by a customer. Both triggers help maintain the inventory levels in the Product table by adjusting the quantity\_available column based on the quantities of products being added or removed in the PO\_details and Order\_details tables.

**Loading Data in Tables**

INSERT INTO Supplier VALUES

(1, 'Dell', 'Oliver Montgomery', 15551234567, 'oliver.montgomery@dell.com', '123 Main Street', 'Dallas', 'Texas', 75052, 'United States'),

(2, 'Apple Inc.', 'Ellis Villegas', 14085512150, 'ellis.villegas@apple.com', '1 Apple Park Way', 'Cupertino', 'California', 95014, 'United States'),

(3, 'Samsung Electronics', 'Lee Joon-ho', 82234567890, 'lee.joonho@samsung.com', '456 Seoul Street', 'Seoul', 'Seoul', 04567, 'South Korea'),

(4, 'Beijing Tech', 'Zhang Xiaojie', 861098765432, 'zhang.xiaojie@beijingtech.com', '789 Nanjing Road', 'Beijing', 'Hebei', 100000, 'China'),

(5, 'Sony', 'Yuki Tanaka', 81345678901, 'yuki.tanaka@sony.com', '123 Shibuya Street', 'Tokyo', 'Tokyo', 1080075, 'Japan'),

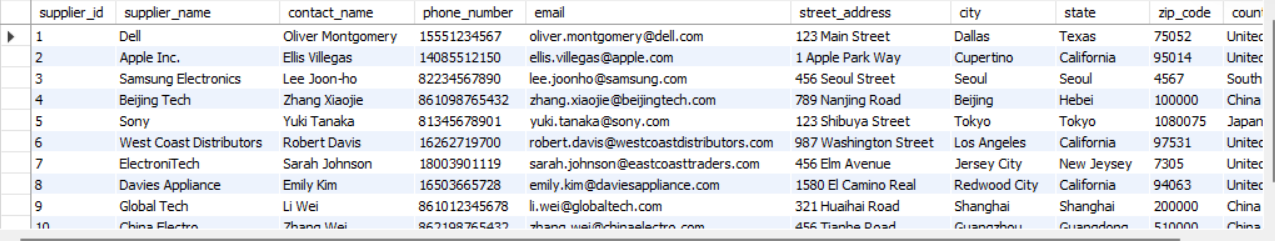
(6, 'West Coast Distributors', 'Robert Davis', 16262719700, 'robert.davis@westcoastdistributors.com', '987 Washington Street', 'Los Angeles', 'California', 97531, 'United States'),

(7, 'ElectroniTech', 'Sarah Johnson', 18003901119, 'sarah.johnson@eastcoasttraders.com', '456 Elm Avenue', 'Jersey City', 'New Jeysey', 07305, 'United States'),

(8, 'Davies Appliance', 'Emily Kim', 16503665728, 'emily.kim@daviesappliance.com', '1580 El Camino Real', 'Redwood City', 'California', 94063, 'United States'),

(9, 'Global Tech', 'Li Wei', 861012345678, 'li.wei@globaltech.com', '321 Huaihai Road', 'Shanghai', 'Shanghai', 200000, 'China'),

(10, 'China Electro', 'Zhang Wei', 862198765432, 'zhang.wei@chinaelectro.com', '456 Tianhe Road', 'Guangzhou', 'Guangdong', 510000, 'China');



Supplier Table

INSERT INTO Product VALUES

(1, 'Smartphones', 'Apple', 'iPhone 12 Pro', 'A powerful phone with 5G capabilities and advanced camera features', NULL, 1300.00, 0),

(2, 'Smartphones', 'Samsung', 'Galaxy S21', 'A high-end phone with a large screen and excellent camera quality', NULL, 1000.00, 0),

(3, 'Headphones', 'Sony', 'WH-1000XM4 Wireless Headphones', 'Premium noise-canceling headphones with excellent sound quality', NULL, 400.00, 0),

(4, 'TV', 'LG', 'OLED55CXPUA 4K TV', 'A stunning OLED TV with amazing picture quality and smart features', NULL, 2000.00, 0),

(5, 'Laptops', 'Microsoft', 'Surface Laptop 4', 'A powerful laptop with a sleek design and long battery life', NULL, 1300.00, 0),

(6, 'Tablets', 'Samsung', 'Galaxy Tab S7', 'A high-end tablet with a large screen and powerful processor', NULL, 750.00, 0),

(7, 'Gaming', 'Sony', 'PlayStation 5', 'The latest gaming console from Sony with advanced graphics and fast load times', NULL, 600.00, 0),

(8, 'Laptops', 'Apple', 'Macbook Air M2', 'Redesigned around the next-generation M2 chip, MacBook Air is strikingly thin and brings exceptional speed and power efficiency within its durable all‑aluminum enclosure.', NULL, 1300.00, 0),

(9, 'Headphones', 'Bose', 'Bose QuietComfort 45', 'The Bose QuietComfort 45 is a premium noise-canceling headphone that offers exceptional sound quality and comfort.', NULL, 350.00, 0),

(10, 'Smartphones', 'OnePlus', '9 Pro 5G', 'A flagship phone with high-end specs and a fast-charging battery', NULL, 1100.00, 0),

(11, 'TV', 'TCL', 'R655 2022 QLED', 'The TCL R655 2022 QLED is a feature-packed smart TV with a brilliant QLED display, delivering vibrant colors and immersive viewing experiences.', NULL, 750.00, 0),

(12, 'Gaming', 'Microsoft', 'Xbox Series X', 'The latest gaming console from Microsoft with advanced graphics and fast load times', NULL, 600.00, 0),

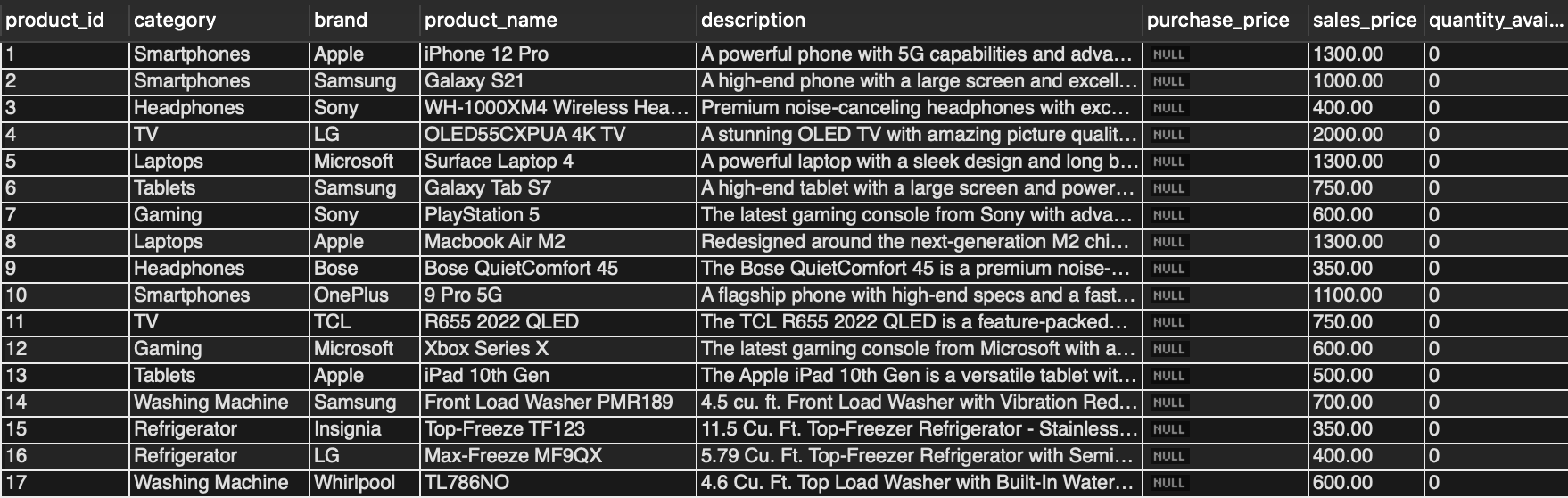
(13, 'Tablets', 'Apple', 'iPad 10th Gen', 'The Apple iPad 10th Gen is a versatile tablet with a powerful processor and a stunning Retina display, perfect for productivity and entertainment.', NULL, 500.00, 0),

(14, 'Washing Machine', 'Samsung', 'Front Load Washer PMR189', '4.5 cu. ft. Front Load Washer with Vibration Reduction Technology+ in Brushed Black', NULL, 700.00, 0),

(15, 'Refrigerator', 'Insignia', 'Top-Freeze TF123', '11.5 Cu. Ft. Top-Freezer Refrigerator - Stainless steel', NULL, 350.00, 0),

(16, 'Refrigerator', 'LG', 'Max-Freeze MF9QX', '5.79 Cu. Ft. Top-Freezer Refrigerator with Semi Auto Defrost - Platinum Silver', NULL, 400.00, 0),

(17, 'Washing Machine', 'Whirlpool', 'TL786NO', '4.6 Cu. Ft. Top Load Washer with Built-In Water Faucet - White', NULL, 600.00, 0);



Product Table

# Purchase order of 10 iphones

INSERT INTO Purchasing\_order (PO\_id, supplier\_id, purchase\_date, shipped\_date, received\_date, total, discount, tax) VALUES (1, 2, '2023-01-01', '2023-01-03', '2023-01-05', 12000.00, 1200.00, 1200.00);

INSERT INTO PO\_details (PO\_id, product\_id, purchase\_price, quantity) VALUES (1, 1, 1200.00, 10);

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 1) WHERE product\_id = 1;

# Purchase order of 10 samsung phones

INSERT INTO Purchasing\_order (PO\_id, supplier\_id, purchase\_date, shipped\_date, received\_date, total, discount, tax) VALUES (2, 3, '2023-01-01', '2023-01-05', '2023-01-10', 8000.00, 0.00, 800.00);

INSERT INTO PO\_details (PO\_id, product\_id, purchase\_price, quantity) VALUES (2, 2, 800.00, 10);

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 2) WHERE product\_id = 2;

# Purchase order of 10 bose headphones

INSERT INTO Purchasing\_order (PO\_id, supplier\_id, purchase\_date, shipped\_date, received\_date, total, discount, tax) VALUES (3, 6, '2023-01-01', '2023-01-03', '2023-01-04', 2500.00, 200.00, 250.00);

INSERT INTO PO\_details (PO\_id, product\_id, purchase\_price, quantity) VALUES (3, 9, 250.00, 10);

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 9) WHERE product\_id = 9;

# Purchase order of 5 LG TVs

INSERT INTO Purchasing\_order (PO\_id, supplier\_id, purchase\_date, shipped\_date, received\_date, total, discount, tax) VALUES (4, 4, '2023-01-15', '2023-01-20', '2023-01-27', 7500.00, 750.00, 750.00);

INSERT INTO PO\_details (PO\_id, product\_id, purchase\_price, quantity) VALUES (4, 4, 1500.00, 5);

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 4) WHERE product\_id = 4;

# Purchase order of 10 Playstation 5

INSERT INTO Purchasing\_order (PO\_id, supplier\_id, purchase\_date, shipped\_date, received\_date, total, discount, tax) VALUES (5, 5, '2023-02-01', '2023-02-10', '2023-02-15', 4000.00, 500.00, 400.00);

INSERT INTO PO\_details (PO\_id, product\_id, purchase\_price, quantity) VALUES (5, 7, 400.00, 10);

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 7) WHERE product\_id = 7;

# Purchase order of 10 Samsung Tablets

INSERT INTO Purchasing\_order (PO\_id, supplier\_id, purchase\_date, shipped\_date, received\_date, total, discount, tax) VALUES (6, 3, '2023-02-10', '2023-02-15', '2023-02-20', 6000.00, 0.00, 600.00);

INSERT INTO PO\_details (PO\_id, product\_id, purchase\_price, quantity) VALUES (6, 6, 600.00, 10);

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 6) WHERE product\_id = 6;

# Purchase order of 5 Microsoft Xbox + 10 Microsoft Laptops

INSERT INTO Purchasing\_order (PO\_id, supplier\_id, purchase\_date, shipped\_date, received\_date, total, discount, tax) VALUES (7, 10, '2023-02-11', '2023-02-21', '2023-02-28', 12500.00, 750.00, 1250.00);

INSERT INTO PO\_details (PO\_id, product\_id, purchase\_price, quantity) VALUES

(7, 12, 500.00, 5),

(7, 5, 1000.00, 10);

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 12) WHERE product\_id = 12;

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 5) WHERE product\_id = 5;

# Purchase order of 10 Macbook Air M2

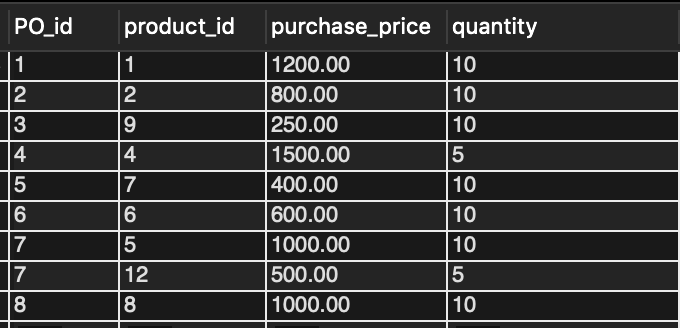
INSERT INTO Purchasing\_order (PO\_id, supplier\_id, purchase\_date, shipped\_date, received\_date, total, discount, tax) VALUES (8, 8, '2023-03-02', '2023-03-04', '2023-03-06', 10000.00, 1000, 1000.00);

INSERT INTO PO\_details (PO\_id, product\_id, purchase\_price, quantity) VALUES (8, 8, 1000.00, 10);

UPDATE Product SET purchase\_price = (SELECT purchase\_price FROM PO\_details WHERE product\_id = 8) WHERE product\_id = 8;



Purchasing\_order table after the above inserts



PO\_details table after the above inserts

INSERT INTO Customer (customer\_id, customer\_name, phone\_number, email, street\_address, city, state, zip\_code, country) VALUES

(1, 'John Doe', 2065551234, 'johndoe@example.com', '123 Main St', 'Seattle', 'Washington', 98101, 'United States'),

(2, 'Jane Smith', 5035555678, 'janesmith@example.com', '456 Elm St', 'Portland', 'Oregon', 97204, 'United States'),

(3, 'Michael Johnson', 4155558765, 'michaeljohnson@example.com', '789 Oak St', 'San Francisco', 'California', 94102, 'United States'),

(4, 'Emily Davis', 7025553456, 'emilydavis@example.com', '321 Pine St', 'Las Vegas', 'Nevada', 89101, 'United States'),

(5, 'David Wilson', 4805557890, 'davidwilson@example.com', '987 Cedar St', 'Phoenix', 'Arizona', 85001, 'United States'),

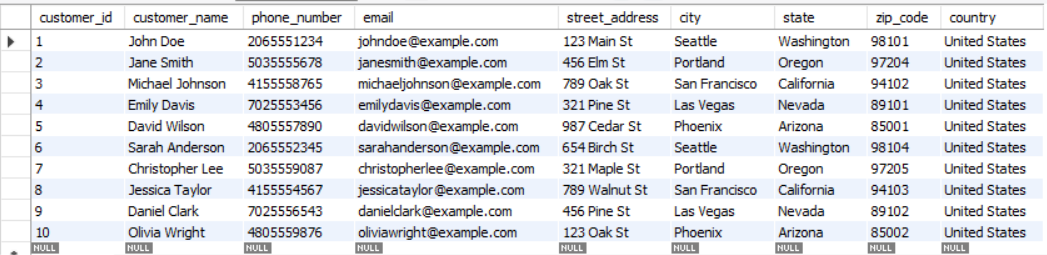
(6, 'Sarah Anderson', 2065552345, 'sarahanderson@example.com', '654 Birch St', 'Seattle', 'Washington', 98104, 'United States'),

(7, 'Christopher Lee', 5035559087, 'christopherlee@example.com', '321 Maple St', 'Portland', 'Oregon', 97205, 'United States'),

(8, 'Jessica Taylor', 4155554567, 'jessicataylor@example.com', '789 Walnut St', 'San Francisco', 'California', 94103, 'United States'),

(9, 'Daniel Clark', 7025556543, 'danielclark@example.com', '456 Pine St', 'Las Vegas', 'Nevada', 89102, 'United States'),

(10, 'Olivia Wright', 4805559876, 'oliviawright@example.com', '123 Oak St', 'Phoenix', 'Arizona', 85002, 'United States');



Customer Table

# Customer no. 1 order of 5 iphones

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (1, 1, '2023-01-06', '2023-01-06', '2023-01-07', 6500.00, 0.00, 650.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES (1, 1, 5);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 1) WHERE product\_id = 1;

# Customer no. 2 order of 3 samsung phones

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (2, 2, '2023-01-11', '2023-01-11', '2023-01-12', 3000.00, 0.00, 300.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES (2, 2, 3);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 2) WHERE product\_id = 2;

# Customer no. 3 order of 4 bose headphones and 1 samsung phone

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (3, 3, '2023-01-11', '2023-01-11', '2023-01-12', 2400.00, 0.00, 240.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES

(3, 9, 4),

(3, 2, 1);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 9) WHERE product\_id = 9;

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 2) WHERE product\_id = 2;

# Customer no. 2 again orders 1 iphone

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (4, 2, '2023-01-21', '2023-01-21', '2023-01-22', 1300.00, 0.00, 130.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES (4, 1, 1);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 1) WHERE product\_id = 1;

# Customer no. 4 orders 2 LG TVs

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (5, 4, '2023-01-28', '2023-01-28', '2023-01-29', 4000.00, 0.00, 400.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES (5, 4, 2);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 4) WHERE product\_id = 4;

# Customer no. 2 again orders 1 more iphone

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (6, 2, '2023-02-15', '2023-02-15', '2023-02-16', 1300.00, 0.00, 130.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES (6, 1, 1);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 1) WHERE product\_id = 1;

# Customer no. 5 orders 2 playstation 5 and 2 LG Tvs

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (7, 5, '2023-02-17', '2023-02-17', '2023-01-18', 5200.00, 0.00, 520.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES

(7, 7, 2),

(7, 4, 2);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 7) WHERE product\_id = 7;

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 4) WHERE product\_id = 4;

# Customer no. 6 orders 7 samsung tablets

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (8, 6, '2023-02-21', '2023-02-21', '2023-01-22', 5250.00, 0.00, 525.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES (8, 6, 7);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 6) WHERE product\_id = 6;

# Customer no. 6 again orders 5 playstation 5's and 3 microsoft Xboxs

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (9, 6, '2023-02-28', '2023-02-28', '2023-03-01', 4800.00, 0.00, 480.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES

(9, 7, 5),

(9, 12, 3);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 7) WHERE product\_id = 7;

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 12) WHERE product\_id = 12;

# Customer no. 7 orders 5 macbook air and 5 microsoft laptops

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (10, 7, '2023-03-06', '2023-03-06', '2023-03-07', 13000.00, 750.00, 1300.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES

(10, 8, 5),

(10, 5, 5);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 8) WHERE product\_id = 8;

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 5) WHERE product\_id = 5;

# Customer no. 2 again orders 3 samsung phones

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (11, 2, '2023-03-06', '2023-03-06', '2023-03-07', 3000.00, 150.00, 300.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES (11, 2, 3);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 2) WHERE product\_id = 2;

# Customer no. 8 orders 4 bose headphones and 4 macbook air

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (12, 8, '2023-03-16', '2023-03-16', '2023-03-17', 6600.00, 600.00, 660.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES

(12, 9, 4),

(12, 8, 4);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 9) WHERE product\_id = 9;

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 8) WHERE product\_id = 8;

# Customer no. 9 orders 1 iphone and 1 microsoft laptop

INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (13, 9, '2023-03-21', '2023-03-21', '2023-03-22', 2600.00, 0.00, 260.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES

(13, 1, 1),

(13, 5, 1);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 1) WHERE product\_id = 1;

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 5) WHERE product\_id = 5;

# Customer no. 10 orders 2 samsung phones

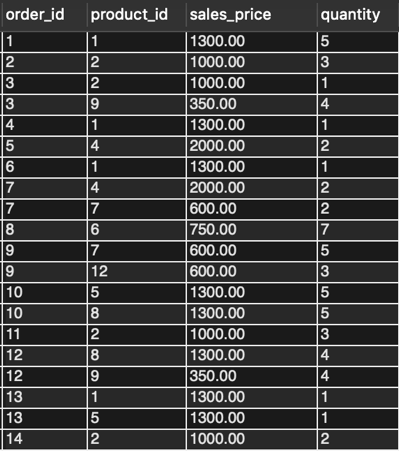
INSERT INTO Orders (order\_id, customer\_id, order\_date, shipped\_date, delivered\_date, total, discount, tax) VALUES (14, 10, '2023-03-25', '2023-03-25', '2023-03-26', 2000.00, 0.00, 200.00);

INSERT INTO Order\_details (order\_id, product\_id, quantity) VALUES (14, 2, 2);

UPDATE Order\_details SET sales\_price = (SELECT sales\_price FROM Product WHERE product\_id = 2) WHERE product\_id = 2;



Orders table after the above inserts



order\_details table after the above inserts

**Data Analysis:**

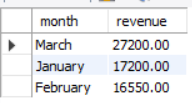
1) Total Revenue Per Month

SELECT DATE\_FORMAT(order\_date, '%M') AS month, SUM(quantity \* sales\_price) AS revenue

FROM order\_details o inner join orders a on a.order\_id = o.order\_id

GROUP BY month

ORDER BY revenue DESC;



The purpose of this query is to have an idea about the total sales per month and see the months with the highest sales. It shows whether the business is experiencing consistent growth, seasonal fluctuations, or any significant changes in revenue generation. This information is crucial for understanding the business's overall financial trajectory.

2) Historic spending for each customer

SELECT c.customer\_name as Name , o.grand\_total as Total\_order\_cost

from orders o inner join customer c on c.customer\_id=o.customer\_id

order by Total\_order\_cost desc;



The purpose of this query is to see which customer is the top buyer (in terms of total sales) from our company. It may be useful for businesses to offer some discounts to such customers (Personalization) as well Customer Segmentation (identifying high value customers and low value customers)

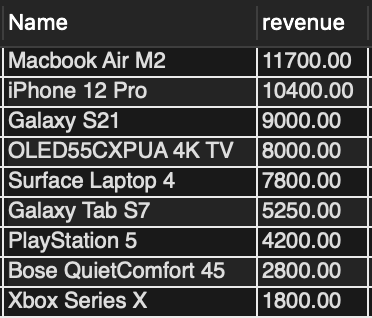
3) Total Revenue Earned for each product

SELECT product\_name as Name, SUM(quantity \* o.sales\_price) AS revenue

FROM order\_details o inner join product a on a.product\_id=o.product\_id

GROUP BY Name

ORDER BY revenue DESC;



The purpose of this query is to see which particular products earn the highest revenue. It can be useful to see which products are giving the business the most revenue/contribute the most to overall profitability.

4) Which products sold in which months, along with quantity and revenue

select DATE\_FORMAT(b.order\_date, '%M') AS month,c.Name ,c.revenue,c.quantity as Quanity from (SELECT product\_name as Name,quantity as quantity, SUM(quantity \* o.sales\_price) AS revenue,o.order\_id as order\_id

FROM order\_details o inner join product a on a.product\_id=o.product\_id

GROUP BY order\_id,Name,quantity ) c inner join orders b on b.order\_id=c.order\_id

GROUP BY month,Name,c.revenue,c.quantity

ORDER BY c.revenue DESC;

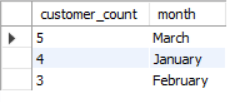


The purpose of this query is to see a breakdown of which products sold in which months. It can give an idea if there is a particular trend of sales of any product in a particular month.

5) Distinct Customers Per Month

select count(customer\_id) as customer\_count ,DATE\_FORMAT(order\_date, '%M') AS month

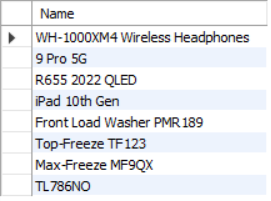
FROM orders GROUP BY month ORDER BY customer\_count DESC;



The purpose of this query is to see whether we have a growing customer base month on month. It can also help assess the customer engagement with the business.

6) Products that need to be ordered

SELECT product\_name as Name from product WHERE purchase\_price IS NULL;

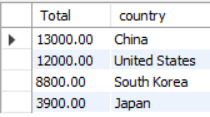


The purpose of this query is to have a look at the products which we have NEVER ordered from any supplier.

7) Country-wise purchase order totals

select Max(a.grand\_total)as Total , c.country from purchasing\_order a inner join supplier c on a.supplier\_id=c.supplier\_id

group by c.country order by Total desc;



The purpose of this query is to identify the countries from which we source the most products revealing the strength and depth of supplier relationships in different countries.

8) Count of products by category that we currently sell/intend to sell

select count(product\_id) as Product\_units , category from Product group by category order by Product\_units desc;

****

The purpose of this query is to highlight the level of product diversity offered by the business. It shows the range of different product categories, indicating whether our business specializes in selling a niche category of electronic items or offers a broad selection.

**Conclusion**

To summarize, we were able to create a simple inventory management database which tracked inventory counts for different products available with our business. There is room for improvement - in terms of tracking other quantities such as reorder point, safety stock, and a process of returns/reverse logistics. The analysis we did can definitely be used to identify opportunities and challenges for the business, and to scale our business further.

Learnings from this project:

1. We were able to use concepts taught in class (ER Diagrams, Relational Models, SQL statements) and apply them to a real-life scenarios. The project helped in strengthening the learning we got throughout the semester apart from the lectures and the homework.
2. Querying an already existing database v/s building your own database is much easier. There are many intricacies when you create your own database that you do not think about when you start working on the same.
3. Knowing how to code is important, knowing how to code to make it work the way we want it to is more important, and knowing how to present our code to make it easy to read is even more important.

Challenges from this project:

1. We wanted to make our system more dynamic in terms of updating purchase\_price and sales\_price but we were facing an error related to deadlock triggers. We tried to solve this using stored procedures and creating new triggers but we couldn’t so we went ahead and used a large number of update statements instead (kind of like a brute force way).

Things we would have done if we had more time:

1. Stored Procedures - To minimize efforts and time by automated insert/update statements. And to minimize human error while doing data entry