Contents

Database Schema Description

Tables and their Structures

1. **COUNTRY Table:**
   - **Purpose**: Stores information about countries.
   - **Columns**:
     - `COUNTRY_NAME`: The name of the country (VARCHAR(15)).
     - `COUNTRY_CODE`: The unique code of the country (VARCHAR(2), Primary Key).
2. **SUPPLIERS Table:**
   - **Purpose**: Stores supplier details.
   - **Columns**:
     - `SPL_ID`: Unique identifier for the supplier (CHAR(9), Primary Key).
     - `SUPL_NAME`: Name of the supplier (VARCHAR(15), Not Null).
     - `SUPL_REG_COUNTRY`: Country code of supplier's registered country (VARCHAR(2), Foreign Key referencing `COUNTRY`).
3. **PRODUCTS Table:**
   - **Purpose**: Contains information about products.
   - **Columns**:
     - `PROD_ID`: Unique identifier for the product (CHAR(9), Primary Key).
     - `PROD_CATEGORY`: Category of the product (VARCHAR(15), Not Null).
     - `PROD_NAME`: Name of the product (VARCHAR(15), Not Null).
     - `UNIT`: Measurement unit of the product (VARCHAR(5)).
     - `MAN_UNIT_PRICE`: Manufacturer's unit price (DECIMAL(10, 2)).
     - `ORIGIN_COUNTRY`: Country code of product's origin (VARCHAR(2), Foreign Key referencing `COUNTRY`).
     - `SPL_ID`: Supplier ID (CHAR(9), Foreign Key referencing `SUPPLIERS`).
4. **RETAILERS Table:**
   - **Purpose**: Stores retailer details.
   - **Columns**:
     - `STORE_ID`: Unique identifier for the retailer (CHAR(9), Primary Key).
     - `STORE_NAME`: Name of the retailer store (VARCHAR(20), Not Null).
     - `STORE_COUNTRY`: Country code of retailer's location (VARCHAR(2), Foreign Key referencing `COUNTRY`).
     - `CHANNEL_LEVEL`: Level of the retail channel (VARCHAR(15)).
5. **ORDERS Table:**
   - **Purpose**: Contains order details.
   - **Columns**:
     - `ORDER_DATE`: Date of the order (DATE, Not Null).
     - `ORDER_ID`: Unique identifier for the order (CHAR(9), Primary Key).
     - `STORE_ID`: Store ID of the retailer placing the order (CHAR(9), Foreign Key referencing `RETAILERS`).

        - ORDER_AMOUNT: Total amount for the order (DECIMAL(10, 2)).

6. **ORDER_DETAILS Table:**
    - **Purpose**: Details of each order including individual products.
    - **Columns**:
        - ORDER_DETAIL_ID: Unique identifier for the order detail (CHAR(9), Primary Key).
        - ORDER_ID: Identifier for the related order (CHAR(9), Foreign Key referencing ORDERS).
        - PRODUCT_ID: Identifier for the product (CHAR(9), Foreign Key referencing PRODUCTS).
        - UNIT_PRICE: Price per unit of the product (DECIMAL(10, 2)).
        - ORDERED_QTITY: Quantity of the product ordered (INT).
        - TOTAL_AMOUNT: Computed column as (ORDERED_QTITY * UNIT_PRICE), PERSISTED.

Relationships

- COUNTRY is referenced by SUPPLIERS, PRODUCTS, and RETAILERS through COUNTRY_CODE.
- SUPPLIERS is referenced by PRODUCTS through SPL_ID.
- RETAILERS is referenced by ORDERS through STORE_ID.
- ORDERS is referenced by ORDER_DETAILS through ORDER_ID.
- PRODUCTS is referenced by ORDER_DETAILS through PRODUCT_ID.

Trigger

- **UpdateOrderAmount Trigger**:
    - **Purpose**: Automatically updates the ORDER_AMOUNT in the ORDERS table whenever there are changes (INSERT, UPDATE, DELETE) in the ORDER_DETAILS table.
    - **Logic**:
        - The trigger recalculates the ORDER_AMOUNT for each affected order by summing the TOTAL_AMOUNT from the ORDER_DETAILS table.
        - It ensures that the ORDER_AMOUNT in the ORDERS table reflects the sum of all related TOTAL_AMOUNT values from the ORDER_DETAILS.

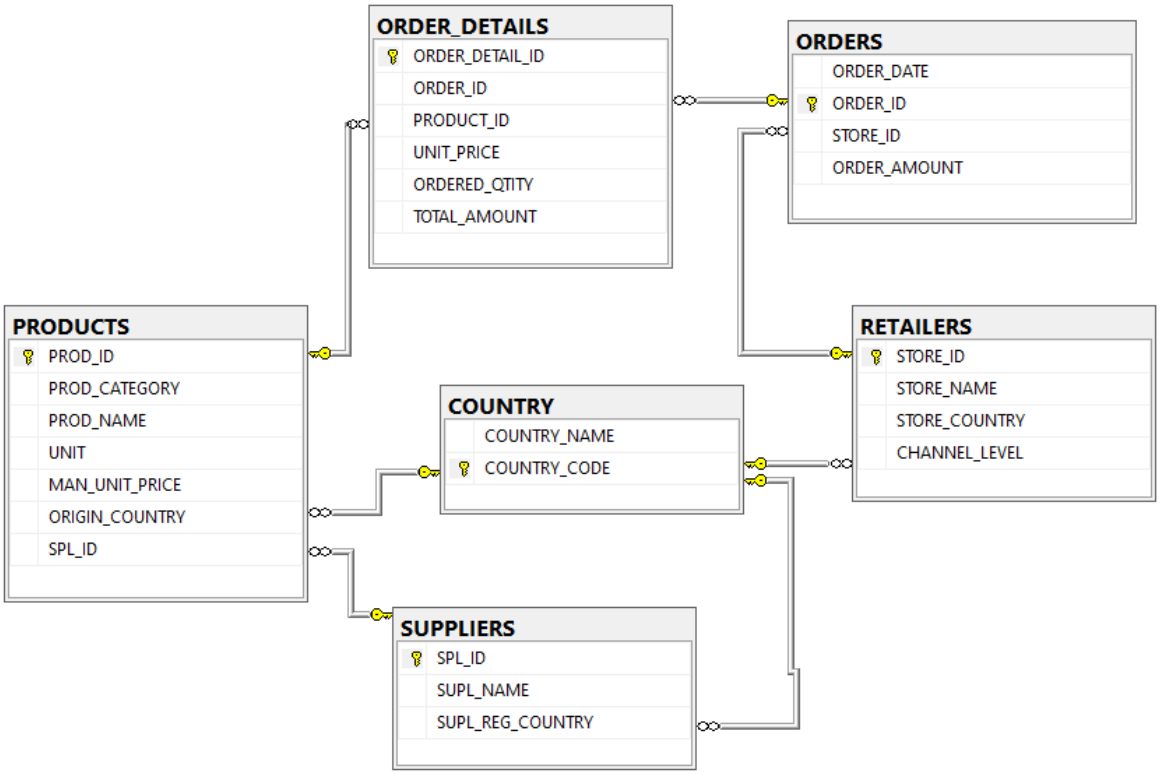
QUERIES

--CREATING TABLES

```
DROP TABLE IF EXISTS COUNTRY;
DROP TABLE IF EXISTS SUPPLIERS;
DROP TABLE IF EXISTS PRODUCTS;
DROP TABLE IF EXISTS RETAILERS;
DROP TABLE IF EXISTS ORDERS;
DROP TABLE IF EXISTS ORDER_DETAILS;

CREATE TABLE COUNTRY (
    COUNTRY_NAME VARCHAR(15),
```

```
    COUNTRY_CODE VARCHAR(2),
    PRIMARY KEY (COUNTRY_CODE)
);

CREATE TABLE SUPPLIERS (
    SPL_ID CHAR(9) NOT NULL,
    SUPL_NAME VARCHAR(15) NOT NULL,
    SUPL_REG_COUNTRY VARCHAR(2),
    PRIMARY KEY (SPL_ID),
);

CREATE TABLE PRODUCTS (
    PROD_ID CHAR(9) NOT NULL,
    PROD_CATEGORY VARCHAR(15) NOT NULL,
    PROD_NAME VARCHAR(15) NOT NULL,
    UNIT VARCHAR(5),
    MAN_UNIT_PRICE DECIMAL(10, 2),
    ORIGIN_COUNTRY VARCHAR(2),
    SPL_ID CHAR(9) NOT NULL,
    PRIMARY KEY (PROD_ID),
);


CREATE TABLE RETAILERS (
    STORE_ID CHAR(9) NOT NULL,
    STORE_NAME VARCHAR(20) NOT NULL,
    STORE_COUNTRY VARCHAR(2),
    CHANNEL_LEVEL VARCHAR(15),
    PRIMARY KEY (STORE_ID),
);

CREATE TABLE ORDERS (
    ORDER_DATE DATE NOT NULL,
    ORDER_ID CHAR(9) PRIMARY KEY,
    STORE_ID CHAR(9) NOT NULL,
    ORDER_AMOUNT DECIMAL(10, 2),
);

CREATE TABLE ORDER_DETAILS (
    ORDER_DETAIL_ID CHAR(9) PRIMARY KEY,
    ORDER_ID CHAR(9),
    PRODUCT_ID CHAR(9),
    UNIT_PRICE DECIMAL(10, 2),
    ORDERED_QTITY INT,
    TOTAL_AMOUNT AS (ORDERED_QTITY * UNIT_PRICE) PERSISTED,
);
```

--Database Diagram



--CREATING TRIGGER

```
--creating trigger to update UpdateOrderAmount in ORDERS table based on ORDERS_DETAILS
table
CREATE TRIGGER UpdateOrderAmount
ON ORDER_DETAILS
AFTER INSERT, UPDATE, DELETE
AS
BEGIN

    UPDATE ORDERS
    SET ORDER_AMOUNT = (
        SELECT SUM(TOTAL_AMOUNT)
        FROM ORDER_DETAILS
        WHERE ORDER_DETAILS.ORDER_ID = ORDERS.ORDER_ID
    )
    FROM ORDERS
    INNER JOIN (
        SELECT DISTINCT ORDER_ID
        FROM INSERTED
        UNION
        SELECT DISTINCT ORDER_ID
        FROM DELETED
    ) AS OrderIds ON ORDERS.ORDER_ID = OrderIds.ORDER_ID;
END;
```

--Inserting values into TABLES

```
INSERT INTO COUNTRY (COUNTRY_NAME, COUNTRY_CODE)
VALUES
('Italy', 'IT'),
('France', 'FR'),
('Poland', 'PL'),
('Germany', 'DE'),
```

```
('Netherlands', 'NL'),
('Latvia', 'LV'),
('Spain', 'ES'),
('Portugal', 'PT'),
('Slovenia', 'SI'),
('Greece', 'GR');

INSERT INTO SUPPLIERS (SPL_ID, SUPL_NAME, SUPL_REG_COUNTRY) VALUES
('MAN001', 'Realbuzz', 'IT'),
('MAN002', 'Kanoodle', 'FR'),
('MAN003', 'Photobug', 'PL'),
('MAN004', 'Zazio', 'DE'),
('MAN005', 'Vipe', 'NL'),
('MAN006', 'Midel', 'LV'),
('MAN007', 'Dynabox', 'ES'),
('MAN008', 'Jabbersphere', 'PT'),
('MAN009', 'Oba', 'SI'),
('MAN010', 'Leexo', 'GR');

INSERT INTO PRODUCTS (PROD_ID, PROD_CATEGORY, PROD_NAME, UNIT, MAN_UNIT_PRICE,
ORIGIN_COUNTRY, SPL_ID) VALUES
('PROD1',  'Vegetables', 'Broccoli',        'kg', '2.35', 'IT', 'MAN001'),
('PROD2',  'Vegetables', 'Yellow tomatoes', 'kg', '1.21', 'NL', 'MAN003'),
('PROD3',  'Vegetables', 'Cherry tomatoes', 'kg', '1.98', 'NL', 'MAN005'),
('PROD4',  'Vegetables', 'Red tomatoes',    'kg', '2.21', 'GR', 'MAN005'),
('PROD5',  'Vegetables', 'Cucumbers',       'kg', '0.75', 'LV', 'MAN006'),
('PROD6',  'Vegetables', 'Eggplants',       'kg', '2.31', 'GR', 'MAN002'),
('PROD7',  'Vegetables', 'Garlic',          'kg', '3.51', 'SI', 'MAN005'),
('PROD8',  'Vegetables', 'Carrot',          'kg', '0.50', 'PT', 'MAN002'),
('PROD9',  'Vegetables', 'Potato',          'kg', '0.35', 'NL', 'MAN009'),
('PROD10', 'Fruits',     'Banana',          'kg', '0.99', 'GR', 'MAN001'),
('PROD11', 'Fruits',     'Apple',           'kg', '0.35', 'PL', 'MAN003'),
('PROD12', 'Fruits',     'Citron',          'kg', '0.71', 'GR', 'MAN008'),
('PROD13', 'Fruits',     'Grapefruit',      'kg', '0.36', 'IT', 'MAN001'),
('PROD14', 'Fruits',     'Pears',           'kg', '2.21', 'SI', 'MAN007'),
('PROD15', 'Fruits',     'Nectarine',       'kg', '2.25', 'PT', 'MAN002'),
('PROD16', 'Fruits',     'Orange',          'kg', '3.35', 'IT', 'MAN009'),
('PROD17', 'Beverage',   'Water',           'btl', '0.55', 'LV', 'MAN006'),
('PROD18', 'Beverage',   'Appel juice',     'Pack', '4.25', 'DE', 'MAN004'),
('PROD19', 'Beverage',   'Orange juice',    'Pack', '5.71', 'GR', 'MAN009'),
('PROD20', 'Beverage',  'Beer',             'Pack',    '21.21', 'DE', 'MAN004'),
('PROD21', 'Beverage',  'Red wine',         'btl', '15', 'PT', 'MAN008'),
('PROD22', 'Beverage',  'White wine',       'btl', '12.21', 'ES', 'MAN007');

INSERT INTO RETAILERS (STORE_ID, STORE_NAME, STORE_COUNTRY, CHANNEL_LEVEL) VALUES
('ST001', 'Plambee', 'PL', 'Main'),
('ST002', 'Feedfish', 'IT', 'Secondary'),
('ST003', 'Kazu', 'DE', 'Rural'),
('ST004', 'Gabspot', 'DE', 'Rural'),
('ST005', 'Kaymbo', 'PL', 'Secondary'),
('ST006', 'Voonix', 'PL', 'Rural'),
('ST007', 'Blogtags', 'IT', 'Secondary'),
('ST008', 'Realpoint', 'IT', 'Secondary'),
('ST009', 'Dabshots', 'DE', 'Rural'),
('ST010', 'Jabbersphere', 'PL', 'Main'),
('ST011', 'Feedfire', 'DE', 'Main');

INSERT INTO ORDERS (ORDER_DATE, ORDER_ID, STORE_ID) VALUES
('2024-01-10', 'ORD001', 'ST001'),
('2024-01-23', 'ORD002', 'ST002'),
('2024-01-31', 'ORD003', 'ST003'),
('2024-02-03', 'ORD004', 'ST003'),
('2024-02-27', 'ORD005', 'ST005'),
('2024-03-10', 'ORD006', 'ST006'),
('2024-03-11', 'ORD007', 'ST007'),
```

```
('2024-03-15', 'ORD008', 'ST001'),
('2024-03-20', 'ORD009', 'ST002'),
('2024-03-26', 'ORD010', 'ST003'),
('2024-03-28', 'ORD011', 'ST005'),
('2024-03-29', 'ORD012', 'ST008'),
('2024-03-30', 'ORD013', 'ST009'),
('2024-04-01', 'ORD014', 'ST009'),
('2024-04-04', 'ORD015', 'ST001'),
('2024-04-06', 'ORD016', 'ST005'),
('2024-04-09', 'ORD017', 'ST006'),
('2024-04-12', 'ORD018', 'ST010'),
('2024-04-15', 'ORD019', 'ST011'),
('2024-04-18', 'ORD020', 'ST005'),
('2024-04-20', 'ORD021', 'ST006'),
('2024-04-27', 'ORD022', 'ST010'),
('2024-05-01', 'ORD023', 'ST011'),
('2024-05-02', 'ORD024', 'ST005'),
('2024-05-02', 'ORD025', 'ST001'),
('2024-05-05', 'ORD026', 'ST002'),
('2024-05-15', 'ORD027', 'ST003'),
('2024-05-26', 'ORD028', 'ST005'),
('2024-05-28', 'ORD029', 'ST009'),
('2024-05-30', 'ORD030', 'ST001'),
('2024-05-31', 'ORD031', 'ST005');

INSERT INTO ORDER_DETAILS (ORDER_DETAIL_ID, ORDER_ID, PRODUCT_ID, UNIT_PRICE,
ORDERED_QTITY) VALUES
('OZ298', 'ORD001', 'PROD4', '2.75', '845'),
('ZS978', 'ORD001', 'PROD22', '17.64', '716'),
('JW778', 'ORD002', 'PROD7', '5.01', '948'),
('GW322', 'ORD002', 'PROD13', '0.95', '612'),
('LV887', 'ORD002', 'PROD13', '0.96', '905'),
('YS772', 'ORD002', 'PROD12', '2.15', '982'),
('RY197', 'ORD003', 'PROD9', '0.75', '761'),
('WY766', 'ORD003', 'PROD19', '7.21', '642'),
('GZ897', 'ORD004', 'PROD16', '5.52', '624'),
('CZ422', 'ORD004', 'PROD16', '5.78', '988'),
('MQ630', 'ORD004', 'PROD1', '3.01', '674'),
('XM332', 'ORD005', 'PROD11', '0.89', '780'),
('SG197', 'ORD005', 'PROD2', '2.01', '526'),
('OL980', 'ORD005', 'PROD9', '0.98', '595'),
('PE782', 'ORD005', 'PROD11', '1.15', '515'),
('RI732', 'ORD006', 'PROD10', '1.75', '947'),
('OB413', 'ORD006', 'PROD22', '17.86', '578'),
('NY331', 'ORD006', 'PROD1', '2.95', '903'),
('XJ718', 'ORD007', 'PROD10', '1.68', '692'),
('AH748', 'ORD007', 'PROD13', '1.25', '655'),
('IY623', 'ORD007', 'PROD10', '1.98', '625'),
('SQ969', 'ORD008', 'PROD10', '1.85', '955'),
('MF668', 'ORD008', 'PROD5', '2.01', '775'),
('KQ574', 'ORD009', 'PROD16', '5.86', '870'),
('HT628', 'ORD010', 'PROD15', '4.25', '785'),
('UF719', 'ORD011', 'PROD2', '1.95', '515'),
('YO698', 'ORD011', 'PROD2', '1.32', '701'),
('VY148', 'ORD011', 'PROD15', '3.76', '605'),
('HR030', 'ORD011', 'PROD11', '1.25', '775'),
('QH568', 'ORD011', 'PROD3', '3.5', '892'),
('DP444', 'ORD011', 'PROD12', '2.75', '657'),
('DE452', 'ORD012', 'PROD5', '1.75', '924'),
('LZ731', 'ORD012', 'PROD4', '3.25', '921'),
('YH298', 'ORD012', 'PROD7', '4.95', '700'),
('ML472', 'ORD012', 'PROD10', '1.35', '597'),
('AZ403', 'ORD013', 'PROD7', '5.25', '908'),
('SD906', 'ORD014', 'PROD19', '8.15', '945'),
('SJ713', 'ORD014', 'PROD5', '2.31', '552'),
```

```
('DQ375', 'ORD014', 'PROD4', '3.01', '878'),
('UJ188', 'ORD014', 'PROD20', '29.61', '730'),
('UH800', 'ORD015', 'PROD7', '6.21', '643'),
('YN940', 'ORD015', 'PROD22', '19.78', '515'),
('AS367', 'ORD015', 'PROD21', '20.35', '697'),
('OD311', 'ORD016', 'PROD21', '19.65', '579'),
('QV478', 'ORD016', 'PROD5', '2.31', '824'),
('CC314', 'ORD016', 'PROD11', '1.34', '888'),
('WI053', 'ORD017', 'PROD22', '21.86', '878'),
('NJ712', 'ORD017', 'PROD19', '7.99', '903'),
('UQ892', 'ORD017', 'PROD19', '8.99', '530'),
('UM038', 'ORD017', 'PROD11', '0.78', '637'),
('CL123', 'ORD017', 'PROD8', '0.75', '917'),
('DO963', 'ORD018', 'PROD8', '1.15', '721'),
('CF370', 'ORD019', 'PROD18', '0.99', '638'),
('ID720', 'ORD020', 'PROD12', '1.99', '904'),
('MI144', 'ORD020', 'PROD10', '1.42', '716'),
('YG176', 'ORD020', 'PROD19', '13.51', '854'),
('AX590', 'ORD021', 'PROD6', '3.45', '954'),
('LY401', 'ORD021', 'PROD7', '4.32', '865'),
('PG703', 'ORD021', 'PROD3', '2.95', '933'),
('HC816', 'ORD022', 'PROD19', '10.25', '806'),
('AD514', 'ORD023', 'PROD6', '5.21', '838'),
('CM585', 'ORD023', 'PROD16', '4.99', '605'),
('ZA754', 'ORD023', 'PROD10', '1.36', '757'),
('LR728', 'ORD023', 'PROD1', '3.51', '905'),
('VC886', 'ORD023', 'PROD16', '4.35', '948'),
('UY784', 'ORD023', 'PROD11', '0.82', '738'),
('EW929', 'ORD024', 'PROD1', '2.75', '943'),
('EO433', 'ORD024', 'PROD17', '5.86', '920'),
('VA714', 'ORD024', 'PROD2', '2.15', '674'),
('CR656', 'ORD025', 'PROD1', '3.21', '939'),
('YN401', 'ORD025', 'PROD18', '1.15', '800'),
('SI401', 'ORD025', 'PROD11', '0.72', '642'),
('TH561', 'ORD025', 'PROD13', '1.35', '672'),
('YQ864', 'ORD026', 'PROD5', '1.95', '909'),
('OO301', 'ORD026', 'PROD21', '18', '762'),
('OQ287', 'ORD026', 'PROD18', '0.75', '701'),
('RL634', 'ORD026', 'PROD6', '4.32', '836'),
('WY773', 'ORD027', 'PROD17', '5.98', '989'),
('NK195', 'ORD027', 'PROD3', '3.21', '534'),
('VG896', 'ORD027', 'PROD1', '2.61', '759'),
('DC351', 'ORD028', 'PROD21', '19.69', '635'),
('FC684', 'ORD028', 'PROD5', '1.25', '710'),
('FA179', 'ORD028', 'PROD7', '6.05', '832'),
('PK621', 'ORD028', 'PROD9', '1.15', '859'),
('BG415', 'ORD028', 'PROD10', '1.89', '993'),
('SH691', 'ORD028', 'PROD22', '17.62', '859'),
('OX550', 'ORD028', 'PROD10', '1.56', '827'),
('IA208', 'ORD029', 'PROD22', '19.32', '944'),
('HU264', 'ORD029', 'PROD1', '4.11', '503'),
('PI981', 'ORD029', 'PROD10', '1.66', '760'),
('CJ518', 'ORD030', 'PROD1', '3.45', '673'),
('IE027', 'ORD030', 'PROD18', '0.98', '601'),
('DI990', 'ORD030', 'PROD10', '1.98', '1000'),
('HV540', 'ORD030', 'PROD16', '4.81', '586'),
('OP301', 'ORD030', 'PROD16', '5.17', '980'),
('QK648', 'ORD030', 'PROD17', '4.65', '886'),
('LH939', 'ORD030', 'PROD15', '4.29', '742'),
('MD245', 'ORD030', 'PROD11', '0.99', '759'),
('TJ342', 'ORD031', 'PROD4', '3.05', '846'),
('IW166', 'ORD031', 'PROD2', '3.15', '598');
```

```
--COUNT
-- Detecting products in Category

SELECT PROD_CATEGORY, COUNT(*) AS CountOfProducts_InCategory
FROM PRODUCTS
GROUP BY PROD_CATEGORY;

--Min, MAX, AVG

SELECT O.STORE_ID, O.ORDER_ID, O.ORDER_AMOUNT, AVG(O.ORDER_AMOUNT) OVER () AS
AvgOrderAmount
FROM ORDERS O
WHERE O.ORDER_AMOUNT  > (SELECT AVG(ORDER_AMOUNT) FROM ORDERS);

SELECT MAX(ORDER_AMOUNT) AS MaxOrder, MIN (ORDER_AMOUNT) AS MinOrder,
ROUND(AVG(ORDER_AMOUNT),2) AS AvgOrder
FROM ORDERS;

--CASE WHEN
--Orders' classification

SELECT ORDER_DATE, ORDER_ID, ORDER_AMOUNT,
 CASE
       WHEN ORDER_AMOUNT <= 6000 THEN 'SmallOrder'
       WHEN ORDER_AMOUNT > 6500 AND ORDER_AMOUNT <= 12000 THEN 'AverageOrder'
       ELSE 'LargeOrder'
END AS OrderClass
FROM ORDERS;
--Count of orders in each class

SELECT OrderClass, COUNT(*) AS OrderCount
FROM (
    SELECT
    CASE
        WHEN ORDER_AMOUNT <= 6000 THEN 'SmallOrder'
        WHEN ORDER_AMOUNT > 6000 AND ORDER_AMOUNT <= 12000 THEN 'AverageOrder'
        ELSE 'LargeOrder'
    END AS OrderClass
    FROM ORDERS
) AS OrderClassTable
GROUP BY OrderClass
ORDER BY OrderCount DESC;
--SELECT AND JOIN TABLES
-- selecting fruits, fruits' suppliers and their countries for Fruits

SELECT P.PROD_NAME, S.SUPL_NAME, C.COUNTRY_NAME
FROM PRODUCTS P
LEFT JOIN SUPPLIERS S on P.SPL_ID = S.SPL_ID
LEFT JOIN COUNTRY C on S.SUPL_REG_COUNTRY = C.COUNTRY_CODE
WHERE P.PROD_CATEGORY = 'Fruits';

--  calculating total order amount for each Reteiler

SELECT R.STORE_NAME, SUM(O.ORDER_AMOUNT) AS TOTAL
FROM RETAILERS R
RIGHT JOIN ORDERS O on R.STORE_ID = O.STORE_ID
GROUP BY R.STORE_NAME
ORDER BY TOTAL DESC;
```

```sql
-- Calculating TotalOrdersAmount for every Months

SELECT MONTH(ORDER_DATE) AS OrderMonth, SUM(ORDER_AMOUNT) AS TotalOrderAmount
FROM ORDERS
GROUP BY MONTH(ORDER_DATE);


--CALCULATION AND JOINING TABLES

-- Products Ordered Quantity and Orders Amount

SELECT P.PROD_ID,
       P.PROD_NAME,
       SUM(OD.ORDERED_QTITY) AS TotalOrderedQuantity,
       P.UNIT,
       SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount
FROM ORDER_DETAILS OD
LEFT JOIN PRODUCTS P on OD.PRODUCT_ID = P.PROD_ID
GROUP BY P.PROD_ID, P.PROD_NAME, P.UNIT
ORDER BY SUM(OD.ORDERED_QTITY) DESC;

-- Revenue for each Product by Month

SELECT P.PROD_ID,
       MONTH(O.ORDER_DATE) AS ORDER_MONTH,
       P.PROD_NAME,
       SUM(OD.ORDERED_QTITY) AS TotalOrderedQuantity,
       P.UNIT,
       SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount,
       SUM(OD.ORDERED_QTITY*P.MAN_UNIT_PRICE) AS ProdExpences,
       SUM(OD.TOTAL_AMOUNT) - SUM(OD.ORDERED_QTITY*P.MAN_UNIT_PRICE) AS Revenue
FROM ORDER_DETAILS OD
       LEFT JOIN PRODUCTS P on OD.PRODUCT_ID = P.PROD_ID
       LEFT JOIN ORDERS O on OD.ORDER_ID = O.ORDER_ID
GROUP BY P.PROD_ID,
       MONTH(O.ORDER_DATE),
       P.PROD_NAME,
       P.UNIT
ORDER BY ORDER_MONTH ASC;

-- Revenue FOR every MONTH

SELECT
    MONTH(O.ORDER_DATE) AS ORDER_MONTH,
    SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount,
    SUM(OD.ORDERED_QTITY * P.MAN_UNIT_PRICE) AS TotalExpenses,
    SUM(OD.TOTAL_AMOUNT) - SUM(OD.ORDERED_QTITY * P.MAN_UNIT_PRICE) AS Revenue
FROM
    ORDER_DETAILS AS OD
LEFT JOIN
    PRODUCTS AS P ON OD.PRODUCT_ID = P.PROD_ID
LEFT JOIN
    ORDERS AS O ON OD.ORDER_ID = O.ORDER_ID
GROUP BY
    MONTH(O.ORDER_DATE)
ORDER BY
    ORDER_MONTH;


-- Revenue for each Product

SELECT P.PROD_ID,
```

```
       P.PROD_NAME,
       SUM(OD.ORDERED_QTITY) AS TotalOrderedQuantity,
       P.UNIT,
       SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount,
       SUM(OD.ORDERED_QTITY*P.MAN_UNIT_PRICE) AS ProdExpences,
       SUM(OD.TOTAL_AMOUNT) - SUM(OD.ORDERED_QTITY *P.MAN_UNIT_PRICE) AS Revenue
FROM ORDER_DETAILS OD
       LEFT JOIN PRODUCTS P on OD.PRODUCT_ID = P.PROD_ID
GROUP BY P.PROD_ID,
       P.PROD_NAME,
       P.UNIT
ORDER BY Revenue DESC;
```

-- Revenue for each Reteiler

```
SELECT R.STORE_NAME,
              SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount,
              SUM(OD.ORDERED_QTITY*P.MAN_UNIT_PRICE) AS ProdExpences,
              SUM(OD.TOTAL_AMOUNT) - SUM(OD.ORDERED_QTITY *P.MAN_UNIT_PRICE) AS Revenue
FROM RETAILERS R
              LEFT JOIN ORDERS O on R.STORE_ID = O.STORE_ID
              LEFT JOIN ORDER_DETAILS OD on O.ORDER_ID = OD.ORDER_ID
              LEFT JOIN PRODUCTS P on OD.PRODUCT_ID = P.PROD_ID
GROUP BY R.STORE_NAME
ORDER BY Revenue DESC;
```

-- Revenue and Turnover for each Country

```
SELECT C.COUNTRY_NAME,
       SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount,
       SUM(OD.ORDERED_QTITY * P.MAN_UNIT_PRICE) AS ProdExpences,
       SUM(OD.TOTAL_AMOUNT) - SUM(OD.ORDERED_QTITY*P.MAN_UNIT_PRICE) AS Revenue
FROM RETAILERS R
       LEFT JOIN ORDERS O on R.STORE_ID = O.STORE_ID
       LEFT JOIN ORDER_DETAILS OD on O.ORDER_ID = OD.ORDER_ID
       LEFT JOIN PRODUCTS P on OD.PRODUCT_ID = P.PROD_ID
       LEFT JOIN COUNTRY C on R.STORE_COUNTRY = c.COUNTRY_CODE
GROUP BY C.COUNTRY_NAME
ORDER BY Revenue DESC;
```

--Revenue for each channel level

```
SELECT R.CHANNEL_LEVEL,
       SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount,
       SUM(OD.ORDERED_QTITY * P.MAN_UNIT_PRICE) AS ProdExpences,
       SUM(OD.TOTAL_AMOUNT) - SUM(OD.ORDERED_QTITY*P.MAN_UNIT_PRICE) AS Revenue
FROM RETAILERS R
       LEFT JOIN ORDERS O on R.STORE_ID = O.STORE_ID
       LEFT JOIN ORDER_DETAILS OD on O.ORDER_ID = OD.ORDER_ID
       LEFT JOIN PRODUCTS P on OD.PRODUCT_ID = P.PROD_ID
GROUP BY R.CHANNEL_LEVEL
ORDER BY Revenue DESC;
```

--PIVOT TABLES

--Revenue in PIVOT table for each channel level and month

```
SELECT
    R.CHANNEL_LEVEL,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 1 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Jan,
```

```
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 2 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Feb,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 3 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Mar,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 4 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Apr,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 5 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_May,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 6 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Jun,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 7 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Jul,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 8 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Aug,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 9 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Sep,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 10 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Oct,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 11 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Nov,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 12 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Revenue_Dec,
    SUM(OD.TOTAL_AMOUNT - OD.ORDERED_QTITY * P.MAN_UNIT_PRICE) AS Revenue_AllMonths

FROM
    RETAILERS R
LEFT JOIN
    ORDERS O ON R.STORE_ID = O.STORE_ID
RIGHT JOIN
    ORDER_DETAILS OD ON O.ORDER_ID = OD.ORDER_ID
LEFT JOIN
    PRODUCTS P ON OD.PRODUCT_ID = P.PROD_ID
GROUP BY
    R.CHANNEL_LEVEL
ORDER BY
    R.CHANNEL_LEVEL;
```

-- Orders Amount in PIVOT table for each channel level and month

```
SELECT
    R.CHANNEL_LEVEL,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 1 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Jan,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 2 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Feb,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 3 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Mar,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 4 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Apr,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 5 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_May,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 6 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Jun,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 7 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Jul,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 8 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Aug,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 9 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Sep,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 10 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Oct,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 11 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Nov,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 12 THEN OD.TOTAL_AMOUNT ELSE 0 END) AS
TotalOrdersAmount_Dec,
    SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount_AllMonths
```

```
FROM
    RETAILERS R
LEFT JOIN
    ORDERS O ON R.STORE_ID = O.STORE_ID
RIGHT JOIN
    ORDER_DETAILS OD ON O.ORDER_ID = OD.ORDER_ID
LEFT JOIN
    PRODUCTS P ON OD.PRODUCT_ID = P.PROD_ID
GROUP BY
    R.CHANNEL_LEVEL
ORDER BY
    R.CHANNEL_LEVEL;
```

-- Expences in PIVOT table for each channel level and month

```
SELECT
    R.CHANNEL_LEVEL,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 1 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_Jan,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 2 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_Feb,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 3 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_Mar,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 4 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_Apr,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 5 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_May,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 6 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_Jun,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 7 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_Jul,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 8 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_Aug,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 9 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE 0
END) AS ProdExpenses_Sep,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 10 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE
0 END) AS ProdExpenses_Oct,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 11 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE
0 END) AS ProdExpenses_Nov,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 12 THEN OD.ORDERED_QTITY * P.MAN_UNIT_PRICE ELSE
0 END) AS ProdExpenses_Dec,
    SUM(OD.ORDERED_QTITY * P.MAN_UNIT_PRICE) AS ProdExpenses_AllMonths
FROM
    RETAILERS R
LEFT JOIN
    ORDERS O ON R.STORE_ID = O.STORE_ID
RIGHT JOIN
    ORDER_DETAILS OD ON O.ORDER_ID = OD.ORDER_ID
LEFT JOIN
    PRODUCTS P ON OD.PRODUCT_ID = P.PROD_ID
GROUP BY
    R.CHANNEL_LEVEL
ORDER BY
    R.CHANNEL_LEVEL;
```

--VIEW and UNION

--Creation of view for each retailer by month in 2024

```
CREATE VIEW RevenueByReteilers2024 AS
SELECT
    R.STORE_NAME,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 1 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Jan,
```

```sql
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 2 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Feb,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 3 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Mar,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 4 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Apr,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 5 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS May,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 6 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Jun,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 7 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Jul,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 8 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Aug,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 9 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Sep,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 10 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Oct,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 11 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Nov,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 12 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Dec,
    SUM(OD.TOTAL_AMOUNT - OD.ORDERED_QTITY * P.MAN_UNIT_PRICE) AS TOTAL_YEAR
FROM
    RETAILERS R
LEFT JOIN
    ORDERS O ON R.STORE_ID = O.STORE_ID
RIGHT JOIN
    ORDER_DETAILS OD ON O.ORDER_ID = OD.ORDER_ID
LEFT JOIN
    PRODUCTS P ON OD.PRODUCT_ID = P.PROD_ID
WHERE YEAR(O.ORDER_DATE)=2024
GROUP BY
    R.STORE_NAME

UNION ALL

SELECT
    'TOTAL_REVENUE' AS STORE_NAME,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 1 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Jan,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 2 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Feb,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 3 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Mar,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 4 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Apr,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 5 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS May,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 6 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Jun,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 7 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Jul,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 8 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Aug,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 9 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Sep,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 10 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Oct,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 11 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Nov,
    SUM(CASE WHEN MONTH(O.ORDER_DATE) = 12 THEN OD.TOTAL_AMOUNT - OD.ORDERED_QTITY *
P.MAN_UNIT_PRICE ELSE 0 END) AS Dec,
    SUM(OD.TOTAL_AMOUNT - OD.ORDERED_QTITY * P.MAN_UNIT_PRICE) AS TOTAL_REVENUE
FROM
```

```
    RETAILERS R
LEFT JOIN
    ORDERS O ON R.STORE_ID = O.STORE_ID
RIGHT JOIN
    ORDER_DETAILS OD ON O.ORDER_ID = OD.ORDER_ID
LEFT JOIN
    PRODUCTS P ON OD.PRODUCT_ID = P.PROD_ID
WHERE YEAR(O.ORDER_DATE)=2024;

--Select from view
SELECT * FROM RevenueByReteilers2024;

--Delete view
DROP VIEW RevenueByReteilers2024



--STORED PROCEDURE

-- sored procedure to retrieve data for specific order
CREATE PROCEDURE OrderDetails
    @OrderID VARCHAR(9)
AS
BEGIN

SELECT O.ORDER_DATE,
       OD.ORDER_ID,
       R.STORE_NAME,
       OD.PRODUCT_ID,
       P.PROD_NAME,
       OD.ORDERED_QTITY,
       OD.UNIT_PRICE,
       P.UNIT,
       OD.TOTAL_AMOUNT
FROM ORDER_DETAILS OD
       INNER JOIN PRODUCTS P on OD.PRODUCT_ID = P.PROD_ID
       LEFT JOIN ORDERS O on OD.ORDER_ID =O.ORDER_ID
       LEFT JOIN RETAILERS R on O.STORE_ID = R.STORE_ID

WHERE
        OD.ORDER_ID = @OrderID
END


-- executing procedure
EXEC OrderDetails @OrderID = '@OrderID'


-- EXAML

EXEC OrderDetails @OrderID = 'ORD001'


--stored procedure to retrieve data for specific product by month

CREATE PROCEDURE Product_Revenue
    @ProdID CHAR(9)
AS
BEGIN
SELECT P.PROD_ID,
       MONTH(O.ORDER_DATE) AS ORDER_MONTH,
       P.PROD_NAME,
```

```
       SUM(OD.ORDERED_QTITY) AS TotalOrderedQuantity,
       P.UNIT,
       SUM(OD.TOTAL_AMOUNT) AS TotalOrdersAmount,
       SUM(OD.ORDERED_QTITY*P.MAN_UNIT_PRICE) AS ProdExpences,
       SUM(OD.TOTAL_AMOUNT) - SUM(OD.ORDERED_QTITY*P.MAN_UNIT_PRICE) AS Revenue
FROM ORDER_DETAILS OD
       LEFT JOIN PRODUCTS P on OD.PRODUCT_ID = P.PROD_ID
       LEFT JOIN ORDERS O on OD.ORDER_ID = O.ORDER_ID
WHERE P.PROD_ID = @ProdID
GROUP BY P.PROD_ID,
       MONTH(O.ORDER_DATE),
       P.PROD_NAME,
       P.UNIT
ORDER BY ORDER_MONTH ASC
END;
```

```
-- executing procedure

EXEC Product_Revenue @ProdID = '@ProdID'



-- EXAML
EXEC Product_Revenue @ProdID = 'PROD1'
```

--Procedure to update the manufacturer’s price for a specific product by 20%

```
CREATE PROCEDURE Product_Price_Update
    @ProdID CHAR(9)
AS
BEGIN
UPDATE PRODUCTS
SET MAN_UNIT_PRICE = MAN_UNIT_PRICE * 1.2
WHERE PROD_ID = @ProdID
END;
```

-- executing procedure

```
EXEC Product_Price_Update @ProdID = '@ProdID'



--EXEMPL

EXEC Product_Price_Update @ProdID = 'PROD1'
```