# DATABASE MANAGEMENT SYSTEM - CSA0593

ASSIGNMENT 2

B.LAKSHMI ANJALI

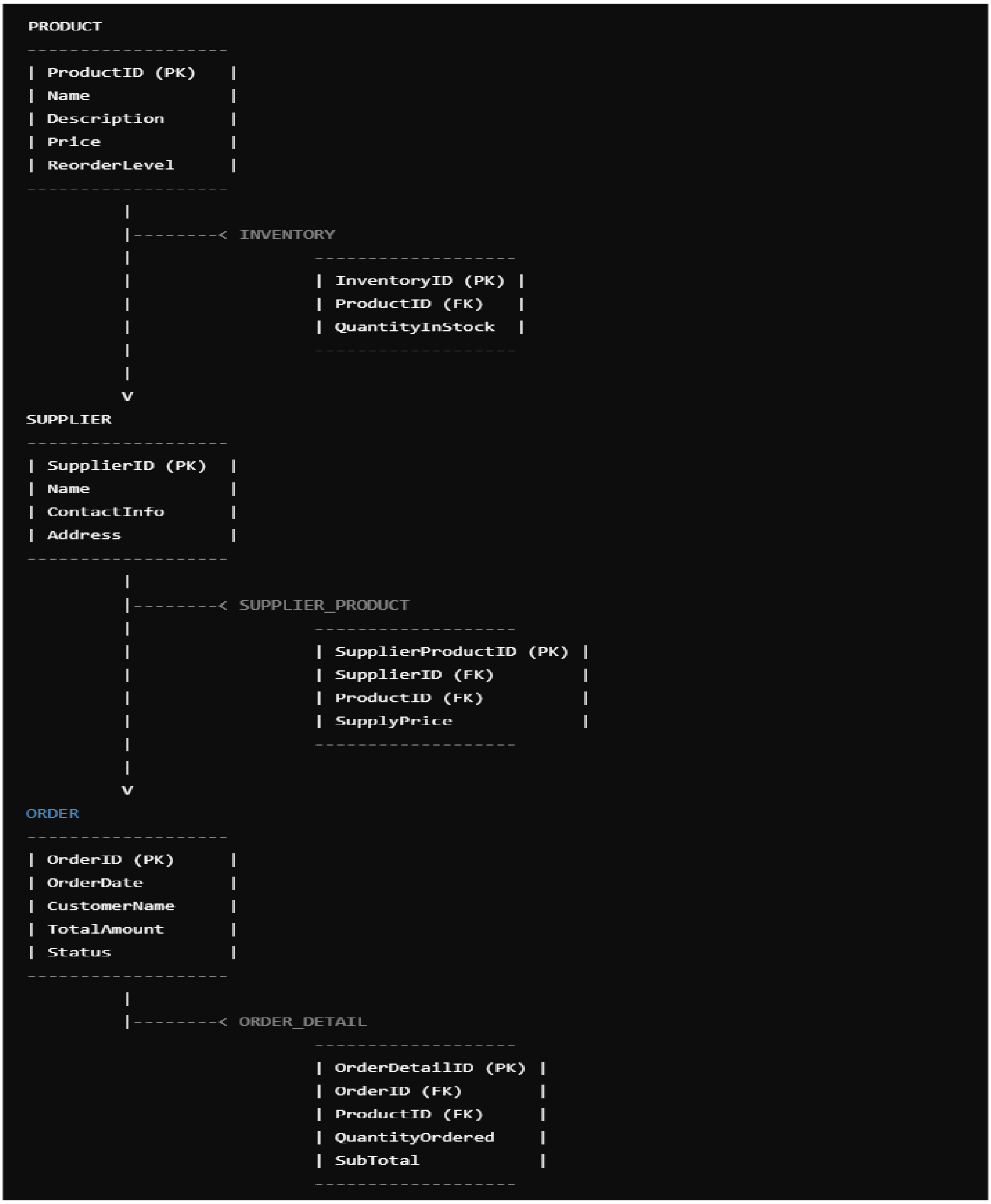
192311344

QUESTION:

- Design a database to manage products, suppliers, inventory, and orders.
  - Model tables for products, suppliers, inventory levels, and orders.
  - Write stored procedures to place and cancel orders.
  - Implement triggers to update inventory levels when an order is placed or cancelled.
  - Write SQL queries to track low stock and generate supplier order recommendations.

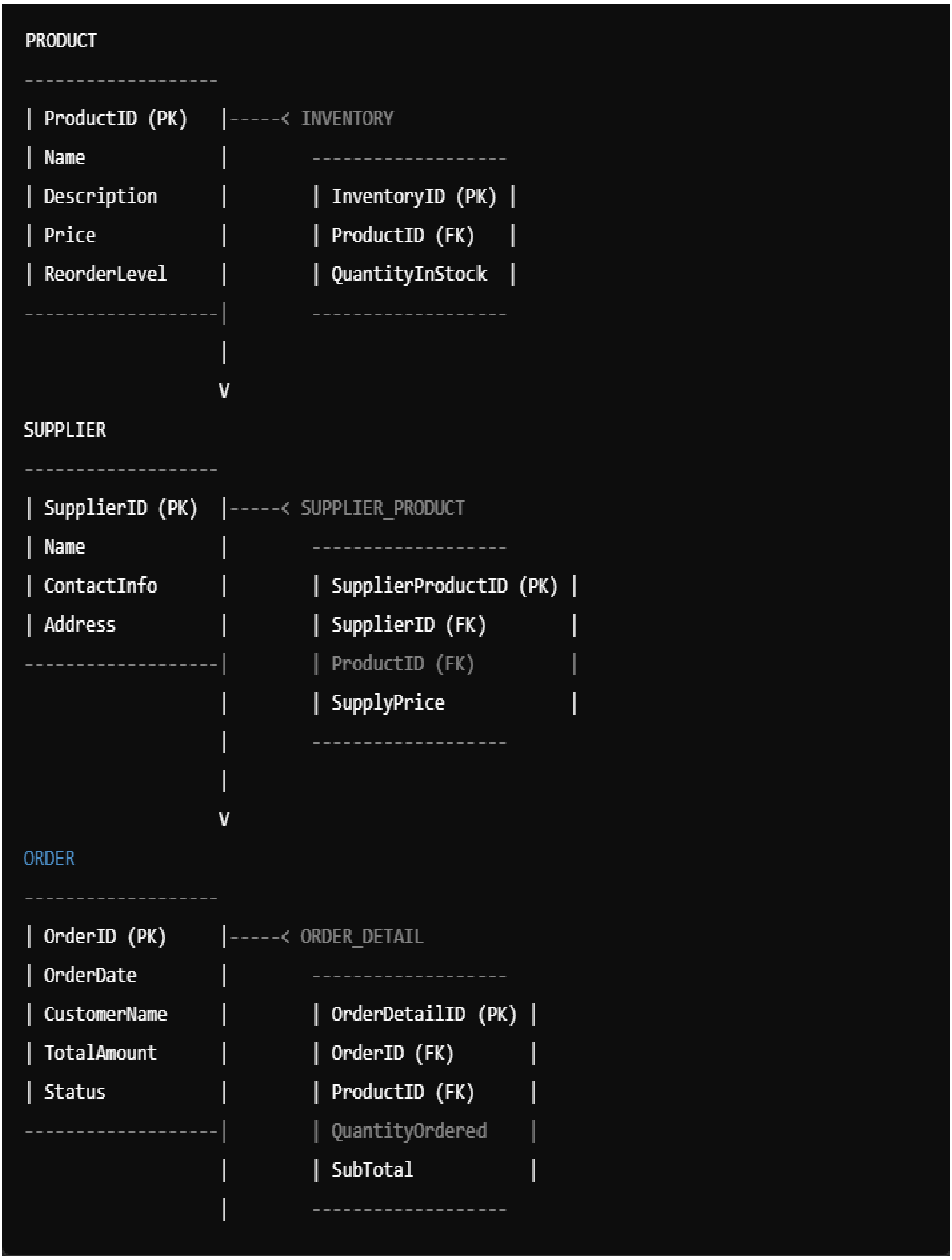
ANSWER:

CONCEPTUAL E.R.DIAGRAM:



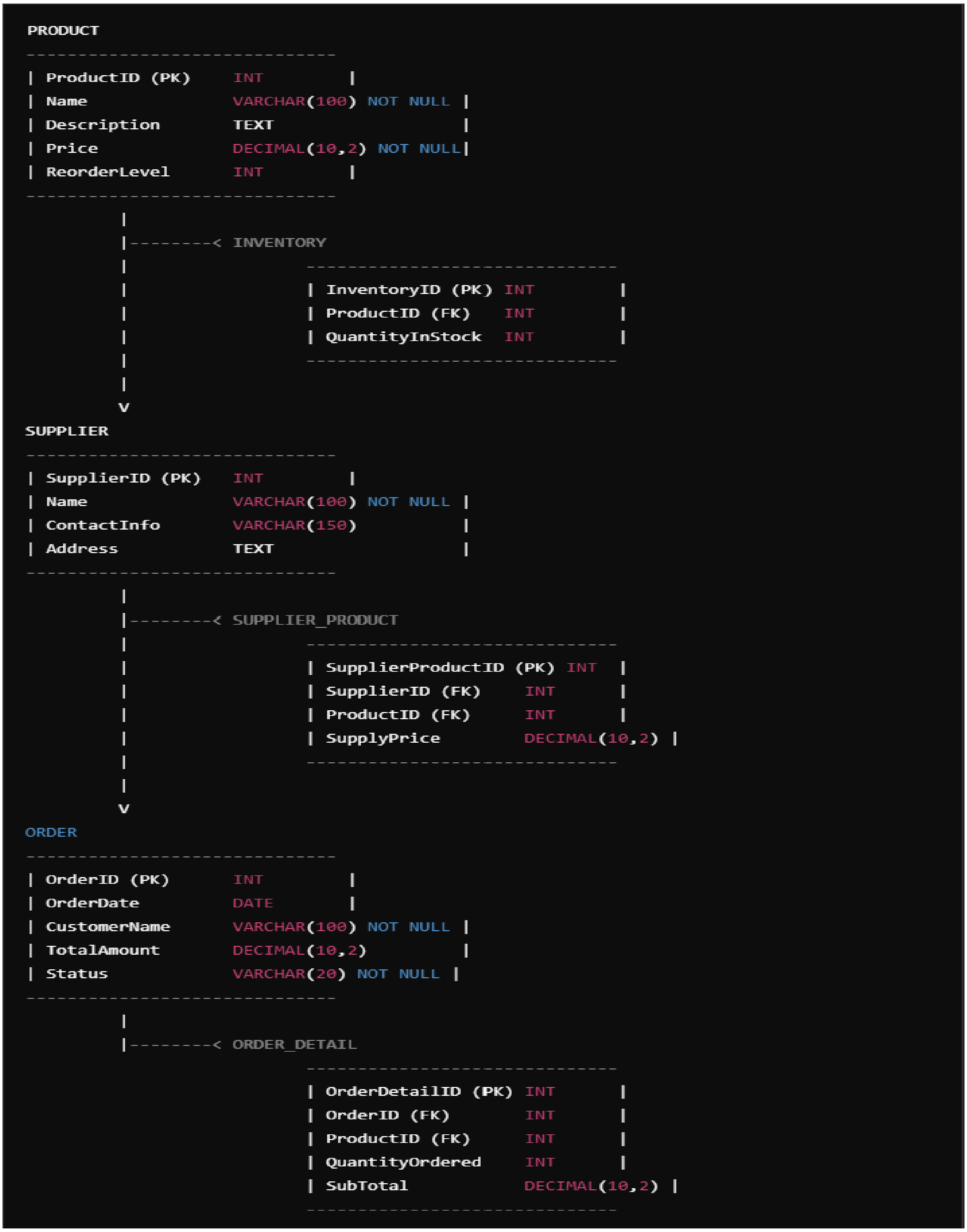
```
PRODUCT
-------------------
| ProductID (PK)    |
| Name              |
| Description       |
| Price             |
| ReorderLevel      |
-------------------
         |
         |--------< INVENTORY
         |
         |                -------------------
         |                | InventoryID (PK) |
         |                | ProductID (FK)   |
         |                | QuantityInStock  |
         |                -------------------
         |
         v
SUPPLIER
-------------------
| SupplierID (PK)   |
| Name              |
| ContactInfo       |
| Address           |
-------------------
         |
         |--------< SUPPLIER_PRODUCT
         |
         |                -------------------
         |                | SupplierProductID (PK) |
         |                | SupplierID (FK)        |
         |                | ProductID (FK)         |
         |                | SupplyPrice            |
         |                -------------------
         |
         v
ORDER
-------------------
| OrderID (PK)      |
| OrderDate         |
| CustomerName      |
| TotalAmount       |
| Status            |
-------------------
         |
         |--------< ORDER_DETAIL
                          -------------------
                          | OrderDetailID (PK) |
                          | OrderID (FK)       |
                          | ProductID (FK)     |
                          | QuantityOrdered    |
                          | SubTotal           |
                          -------------------
```

LOGICAL E.R DIAGRAM:



```
PRODUCT
-------------------
| ProductID (PK)   |-----< INVENTORY
| Name             |        -------------------
| Description      |        | InventoryID (PK) |
| Price            |        | ProductID (FK)   |
| ReorderLevel     |        | QuantityInStock  |
-------------------|        -------------------
                   |
                   v
SUPPLIER
-------------------
| SupplierID (PK)  |-----< SUPPLIER_PRODUCT
| Name             |        -------------------
| ContactInfo      |        | SupplierProductID (PK) |
| Address          |        | SupplierID (FK)        |
-------------------|        | ProductID (FK)         |
                   |        | SupplyPrice            |
                   |        -------------------
                   |
                   v
ORDER
-------------------
| OrderID (PK)     |-----< ORDER_DETAIL
| OrderDate        |        -------------------
| CustomerName     |        | OrderDetailID (PK) |
| TotalAmount      |        | OrderID (FK)       |
| Status           |        | ProductID (FK)     |
-------------------|        | QuantityOrdered    |
                   |        | SubTotal           |
                   |        -------------------
```

PHYSICAL E.R.DIAGRAM:



```
PRODUCT
------------------------------
| ProductID (PK)     INT        |
| Name               VARCHAR(100) NOT NULL |
| Description        TEXT                  |
| Price              DECIMAL(10,2) NOT NULL|
| ReorderLevel       INT        |
------------------------------
         |
         |--------< INVENTORY
         |
         |                 ------------------------------
         |                 | InventoryID (PK) INT        |
         |                 | ProductID (FK)   INT        |
         |                 | QuantityInStock  INT        |
         |                 ------------------------------
         |
         v
SUPPLIER
------------------------------
| SupplierID (PK)    INT        |
| Name               VARCHAR(100) NOT NULL |
| ContactInfo        VARCHAR(150)          |
| Address            TEXT                  |
------------------------------
         |
         |--------< SUPPLIER_PRODUCT
         |
         |                 ------------------------------
         |                 | SupplierProductID (PK) INT  |
         |                 | SupplierID (FK)      INT    |
         |                 | ProductID (FK)       INT    |
         |                 | SupplyPrice          DECIMAL(10,2) |
         |                 ------------------------------
         |
         |
         v
ORDER
------------------------------
| OrderID (PK)       INT        |
| OrderDate          DATE       |
| CustomerName       VARCHAR(100) NOT NULL |
| TotalAmount        DECIMAL(10,2)         |
| Status             VARCHAR(20) NOT NULL |
------------------------------
         |
         |--------< ORDER_DETAIL
                           ------------------------------
                           | OrderDetailID (PK) INT      |
                           | OrderID (FK)       INT      |
                           | ProductID (FK)     INT      |
                           | QuantityOrdered    INT      |
                           | SubTotal           DECIMAL(10,2) |
                           ------------------------------
```

MYSQL STATEMENTS:

```
mysql
CREATE DATABASE InventoryManagement;

USE InventoryManagement;

CREATE TABLE Suppliers (
  SupplierID INT AUTO_INCREMENT PRIMARY KEY,
  SupplierName VARCHAR(100),
  SupplierAddress VARCHAR(255),
  SupplierPhone VARCHAR(20)
);

CREATE TABLE Products (
  ProductID INT AUTO_INCREMENT PRIMARY KEY,
  ProductName VARCHAR(100),
  ProductDescription VARCHAR(255),
  UnitPrice DECIMAL(10, 2)
);
```

```
CREATE TABLE Inventory (
  InventoryID INT AUTO_INCREMENT PRIMARY KEY,
  ProductID INT,
  Quantity INT,
  ReorderLevel INT,
  FOREIGN KEY (ProductID) REFERENCES
Products(ProductID)
);

CREATE TABLE Orders (
  OrderID INT AUTO_INCREMENT PRIMARY KEY,
  SupplierID INT,
  OrderDate DATE,
  Status VARCHAR(50),
  FOREIGN KEY (SupplierID) REFERENCES
Suppliers(SupplierID)
);

CREATE TABLE OrderItems (
  OrderItemID INT AUTO_INCREMENT PRIMARY KEY,
  OrderID INT,
  ProductID INT,
  Quantity INT,
  FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
  FOREIGN KEY (ProductID) REFERENCES
Products(ProductID)
);
```

Stored Procedures:

```
mysql
DELIMITER //

CREATE PROCEDURE sp_PlaceOrder(
  IN orderID INT,
  IN supplierID INT,
  IN orderDate DATE
)
```

```
BEGIN
  INSERT INTO Orders (OrderID, SupplierID, OrderDate,
Status)
  VALUES (orderID, supplierID, orderDate, 'Pending');
END //

CREATE PROCEDURE sp_CancelOrder(
  IN orderID INT
)
BEGIN
  UPDATE Orders
  SET Status = 'Cancelled'
  WHERE OrderID = orderID;
END //

DELIMITER;
```

Triggers:

```
mysql
DELIMITER //

CREATE TRIGGER tr_UpdateInventoryOnOrder
AFTER INSERT ON OrderItems
FOR EACH ROW
BEGIN
  UPDATE Inventory
  SET Quantity = Quantity - NEW.Quantity
  WHERE ProductID = NEW.ProductID;
END //

CREATE TRIGGER tr_UpdateInventoryOnCancel
AFTER UPDATE ON Orders
FOR EACH ROW
BEGIN
  IF NEW.Status = 'Cancelled' THEN
    UPDATE Inventory
    SET Quantity = Quantity + (SELECT Quantity FROM OrderItems WHERE OrderID = NEW.OrderID)
    WHERE ProductID = (SELECT ProductID FROM OrderItems WHERE OrderID = NEW.OrderID);
  END IF;
END //

DELIMITER;
```

SQL Queries:

```
mysql
-- Track Low Stock
SELECT
  ProductName,
  Quantity,
  ReorderLevel
FROM
  Products
  JOIN Inventory ON Products.ProductID =
Inventory.ProductID
WHERE
  Quantity <= ReorderLevel;

-- Generate Supplier Order Recommendations
SELECT
  SupplierName,
  ProductName,
  Quantity
FROM
  Suppliers
  JOIN Products ON Suppliers.SupplierID =
Products.SupplierID
  JOIN Inventory ON Products.ProductID =
Inventory.ProductID
WHERE
  Quantity <= ReorderLevel;
```

## Conclusion:

This database design provides a comprehensive foundation for managing products, suppliers, inventory levels, and orders. The stored procedures simplify order placement and cancellation, while the triggers ensure data consistency and accuracy. The SQL queries enable tracking of low stock and generation of supplier order recommendations.