| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| CustomerID | INT (PK) | Unique identifier |
| Name | VARCHAR(100) | Full name of customer |
| Email | VARCHAR(100) | Unique email |
| JoinDate | DATE | Date customer joined |

1. **Customers**
2. **Books**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| BookID | INT (PK) | Unique identifier |
| Title | VARCHAR(150) | Title of the book |
| Author | VARCHAR(100) | Author name |
| Price | DECIMAL(10,2) | Selling price |
| StockQty | INT | Quantity in stock |

1. **Orders**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| OrderID | INT (PK) | Unique order ID |
| CustomerID | INT (FK) | Who placed the order |
| OrderDate | DATE | When the order was placed |

1. **OrderItems**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| OrderItemID | INT (PK) | Unique item ID |
| OrderID | INT (FK) | Reference to Orders table |
| BookID | INT (FK) | Which book was ordered |
| Quantity | INT | How many copies |

Create the following **stored procedures**:

**1. PlaceOrder**

* **Input:** CustomerID, List of BookIDs with Quantities.
* **Behavior:**
  + Inserts into Orders and OrderItems.
  + Updates Books.StockQty (reduce stock).
  + Rollback if stock is insufficient.
  + Return new OrderID.

**2. GetCustomerOrderHistory**

* **Input:** CustomerID
* **Output:** List of all orders by the customer, including book titles, quantities, prices, and total for each order.

**3. GetLowStockBooks**

* **Input:** Threshold quantity.
* **Output:** List of books where StockQty is less than threshold.

**4. UpdateBookPrice**

* **Input:** BookID, NewPrice.
* **Behavior:** Updates the price and returns old and new price for logging.

1. **Cancel Order**

**Problem:** Create a stored procedure CancelOrder that takes an OrderID as input and cancels the order by:

* Deleting all related records from the OrderItems table.
* Deleting the order from the Orders table.
* Restoring the ordered quantities back to the respective Books.StockQty.
* If the order does not exist, return a meaningful message.

1. **. Top Selling Books**

**Problem:** Create a stored procedure GetTopSellingBooks that accepts an optional parameter TopN (default is 5). It should:

* Return the top N books based on total quantity sold from the OrderItems table.
* Include BookID, Title, and total quantity sold.

1. **Customer Spending Report**

**Problem:** Write a procedure GetCustomerSpending that takes CustomerID as input and:

* Returns the total amount that customer has spent across all orders.
* Include breakdown: OrderID, OrderDate, and Amount.

1. **Books by Author**

**Problem:** Create a procedure GetBooksByAuthor that accepts AuthorName and:

* Returns all books written by the specified author with fields like BookID, Title, Price, and StockQty.

1. **Monthly Sales Report**

**Problem:** Develop MonthlySalesReport procedure that takes Year and Month as inputs and:

* Returns total number of orders, total revenue generated, and the best-selling book in that month.

1. **Add New Book**

**Problem:** Implement AddNewBook procedure that accepts book details (Title, Author, Price, StockQty) and:

* Inserts the book into the Books table.
* Returns the newly generated BookID.

1. **Update Customer Email**

**Problem:** Write a procedure UpdateCustomerEmail that takes CustomerID and NewEmail. The procedure should:

* Validate that NewEmail is not already in use.
* Update the email if valid.
* Return success or error message accordingly.

1. **. Frequent Customers**

**Problem:** Create a stored procedure GetFrequentCustomers that takes MinOrders as input and:

* Returns all customers who have placed more than MinOrders.

1. **. Pending Orders Report**

**Problem:** Given an updated schema where Orders has a Status column, create GetPendingOrders procedure to:

* Return all orders with status 'Pending'.
* Include OrderID, CustomerName, OrderDate, and book details.

1. **Restock Low Inventory**

**Problem:** Implement RestockLowInventory that takes two parameters: Threshold and RestockAmount. It should:

* Identify all books where StockQty < Threshold.
* Increase their StockQty by RestockAmount.
* Return the list of updated books.