Part 1: Database Setup & Basic Operations

1. Database Schema Design

Create the following tables with appropriate data types and constraints:

Authors Table:

- Author ID (primary key)
- Full name (required)
- Birth year
- Country

Books Table:

- Book ID (primary key)
- Title (required, unique)
- ISBN (unique)
- Publication year
- Price
- Stock quantity
- Author ID (foreign key)

Customers Table:

- Customer ID (primary key)
- Email (required, unique)
- Full name (required)
- Registration date
- City

Orders Table:

- Order ID (primary key)
- Customer ID (foreign key)
- Order date

- Total amount

Order_Items Table:

- Order item ID (primary key)
- Order ID (foreign key)
- Book ID (foreign key)
- Quantity
- Price at purchase

Reviews Table:

- Review ID (primary key)
- Book ID (foreign key)
- Customer ID (foreign key)
- Rating (1-5)
- Review text
- Review date

2. Data Manipulation

- Insert at least 5 authors, 10 books, 8 customers, 6 orders with multiple items, and 12 reviews
- Update the stock quantity of 3 books after orders are placed
- Delete a customer who has never placed an order

3. Queries

Write queries to:

1. Find all books published after 2020, sorted by price (descending)
2. List all customers from a specific city
3. Show all orders placed in the last 30 days with customer names
4. Find books that are out of stock (quantity = 0)
5. Display the average rating for each book (show book title and average rating)

6. Count how many orders each customer has made
7. Find the top 3 most expensive books

4. Joins & Advanced Queries

1. Show all orders with customer name, book titles, and quantities purchased
2. List all books with their author names
3. Display customers who have never left a review
4. Show all authors with their total book sales (sum of quantities sold)

Part 2: Advanced Implementation

5. Views & Optimization

1. Create a view called book_details that shows book title, author name, average rating, and total reviews
2. Create an index on the Orders table for the order_date column
3. Create an index on the Books table for the author_id column

6. Data Integrity & Transactions

1. Write a transaction that:
    - Creates a new order
    - Adds 2 order items
    - Updates the stock quantity for the purchased books
    - Commits if all steps succeed, or rolls back if any step fails
2. Add a constraint to ensure that book prices cannot be negative
3. Add a constraint to ensure review ratings are between 1 and 5

7. Stored Procedure

Create a stored procedure called place_order that:

- Takes customer_id, book_id, and quantity as parameters
- Checks if enough stock is available

- Creates the order and order item
- Updates the book stock
- Returns success or failure message

8. Security & Performance

1. Explain how you would prevent SQL injection when accepting user input for searching books by title
2. Write a prepared statement example for inserting a new customer
3. Suggest which columns should have indexes and why

9. Scaling Considerations

Answer these questions:

1. If the bookstore grows to millions of users, would you recommend vertical or horizontal scaling? Explain why.
2. How would you implement a leader-follower architecture for this database?
3. If you need to shard the database, what would be a good sharding key and why?

10. Normalization Check

Review your database design and explain:

- Is your design in 1NF and 2NF?
- Are there any redundancies in your tables?
- If you find any, how would you eliminate them?

Bonus Challenge

Create a complex query that shows:

- Customer name
- Total amount spent
- Number of books purchased
- Average rating they've given
- Only for customers who have spent more than $100

Sort by total amount spent (descending) and limit to top 5 customers.

Deliverables

1. SQL file with all CREATE TABLE statements
2. SQL file with all INSERT statements
3. SQL file with all queries and their results
4. Written explanations for questions in parts 7, 8, 9, and 10
5. Screenshots or exported results showing successful query execution

Good luck! Take your time to think through the design and implementation carefully.