

# CS202 – Database Management Systems

## Semester Project – Part 2 Report

Group 10

Students: Hüseyin Emir Bayrak, Oğuz Fergan Dal

Date: Fall 2025

## 1. Introduction

This project presents the design and implementation of an **E-Commerce Order Management System** that models the core functionality of a modern online marketplace. The system integrates a relational database with a graphical user interface and enforces business rules through database-level constraints and SQL-based operations.

The primary objective of the project is to demonstrate a comprehensive understanding of **relational database design principles**, including normalization, referential integrity, and transaction modeling, while also implementing a **fully functional database-driven application** using Java and JDBC.

The system supports multiple user roles with different access privileges and manages the complete lifecycle of an e-commerce transaction, from product browsing to order fulfillment and post-purchase feedback.

---

## 2. System Overview

The system is designed to support three main user roles:

- **Customers**, who browse products, place orders, make payments, and submit reviews.
- **Sellers**, who manage product catalogs, inventory, and incoming orders.
- **Administrators**, who oversee system-wide operations such as user management, category management, shipment control, and global statistics.

Each role interacts with the system through a dedicated interface, ensuring clarity, security, and role-based access control.

---

## 3. System Architecture

The application follows a layered architecture consisting of:

- **Graphical User Interface (GUI):**  
Implemented using Java Swing, providing user-friendly and role-specific screens.
- **Application Logic:**  
Handles user interaction flow, role validation, and coordination between the interface and database operations.
- **Database Access Layer:**  
Uses JDBC with prepared statements to execute SQL queries securely and efficiently.
- **Database Layer:**  
A MySQL relational database that stores all persistent data and enforces data integrity rules.

This architecture improves modularity, maintainability, and consistency across the system.

---

## 4. Database Design

### 4.1 Core Entities

The database schema includes the following main entities:

- Users
- Addresses
- Categories
- Catalogs
- Products
- Orders
- Order Items
- Payments
- Shipments
- Reviews

Additional entities are included to extend system functionality, such as wishlists, notifications, and coupons.

Each entity is defined with a primary key, and relationships between entities are established using foreign keys to maintain referential integrity.

---

### 4.2 Relationships and Business Rules

The database design enforces key business rules, including:

- A user may have multiple addresses.
- Each seller owns exactly one catalog.
- Each product belongs to a single category and a single catalog.
- Orders may include multiple products, but all products in an order must belong to the same seller.
- Each customer may have at most one active shopping cart at any given time.
- Orders progress through a well-defined lifecycle (shopping cart, pending, paid, shipped, delivered, canceled).
- Reviews can only be submitted for products that have been purchased and shipped.

These rules are enforced using a combination of foreign keys, unique constraints, generated columns, and database triggers.

---

## 5. Normalization and Data Integrity

The database schema is fully normalized to **Third Normal Form (3NF)**:

- All attributes are functionally dependent on their primary keys.
- Transitive dependencies are eliminated.
- Redundant data storage is avoided.

This normalization ensures data consistency, reduces redundancy, and prevents update, insertion, and deletion anomalies.

Data integrity is further strengthened through:

- Referential integrity constraints
  - Domain constraints (e.g., valid status values, non-negative prices)
  - Controlled deletion and update policies
- 

## 6. Order Lifecycle Management

A central aspect of the system is the explicit modeling of the **order lifecycle**. Orders progress through standardized states that reflect real-world e-commerce processes.

The concept of a shopping cart is modeled as an order with a dedicated status, allowing the system to enforce the rule that each customer can have only one active cart at a time.

This constraint is enforced directly at the database level, preventing invalid states regardless of application logic.

---

## 7. Inventory and Pricing Management

Inventory is managed at the product level:

- Stock quantities are updated when orders are placed or inventory is restocked.
- Orders cannot include products with insufficient stock.
- Historical pricing is preserved by storing the purchase price at the order-item level.

Subtotal and total order amounts are calculated consistently using database mechanisms, ensuring accuracy and eliminating duplication of business logic.

---

## 8. Payment and Shipment Handling

Payments are directly associated with orders and include information about payment method, amount, status, and timestamps. Each order is linked to at most one payment, providing a clear transaction model.

Shipments track the delivery process and include shipment status, shipping dates, and optional tracking information. Shipment updates are managed by administrators and follow logical dependencies on payment confirmation.

---

## 9. Review System

The review system ensures reliable and fair feedback by enforcing the following constraints:

- Reviews can only be submitted after an order has reached the shipped or delivered stage.
- Each order item may receive at most one review.
- Reviews are always associated with both the customer and the purchased product.

These rules are enforced at the database level to prevent invalid review submissions.

---

## 10. Application Functionality

### Customer Capabilities

Customers can:

- Browse product catalogs and detailed product information.
- Add products to a shopping cart subject to availability.
- Submit orders and view order history.
- Track order status and shipment progress.
- Submit reviews for delivered products.
- View personalized purchase statistics generated using SQL queries.

### Seller Capabilities

Sellers can:

- Manage products in their own catalogs.
- Update inventory levels.
- View and process incoming orders.
- Access sales performance statistics.
- View customer reviews related to their products.

### Administrator Capabilities

Administrators can:

- Manage all users and product categories.
- Monitor and update shipment statuses.
- View system-wide analytical reports.
- Oversee the overall integrity and operation of the platform.

---

## 11. Statistical Reporting

The system provides role-based statistical summaries computed entirely using SQL aggregation functions. These include:

- Monthly purchase totals
- Most purchased product categories
- Average order values
- Seller revenue summaries
- System-wide sales and popularity metrics

This approach ensures efficiency and compliance with database-driven computation requirements.

---

## 12. Exception Handling and Usability

The application includes comprehensive exception handling to manage:

- Invalid inputs
- Database connectivity issues
- Constraint violations
- Unauthorized actions

User feedback is delivered through clear and informative interface messages. The graphical interface is designed to be intuitive and role-specific, improving overall usability.

---

## 13. Conclusion

This project demonstrates a complete and coherent implementation of an e-commerce order management system that integrates **sound relational database design** with **practical application development**.

By enforcing business rules at the database level, maintaining a normalized schema, and leveraging SQL for all data processing and analysis, the system achieves robustness, consistency, and extensibility. The resulting application satisfies all project requirements while providing a solid foundation for further enhancements.