Database Management System Project

on

Medical Inventory for NGOs

submitted to NIIT University

by

Shlok Burmi
Enrolment No.: BT23GCS155
Email ID: shlok.burmi23@st.niituniversity.in
Section: B4

Course In-Charge: Dr. Anand Kumar Mishra
Assistant Professor, CSE

Computer Science and Engineering

NIIT University, Neemrana, Rajasthan

Academic Year: 2024–25
Semester: Sem IV (Jan–June)

Reflections on DBMS Learning

Shlok Burmi

Enrolment No.: BT23GCS155

The DBMS course has greatly improved my understanding of how databases work and how data-driven systems are built. Through the Medical Inventory for NGOs project, I gained practical experience in designing databases, writing SQL queries, and solving real-world problems. This hands-on learning boosted my technical skills and confidence.

These skills will be valuable for my future career in software development or higher studies, as data management is essential in almost every tech field. Database knowledge is a core part of building efficient, scalable systems. Moving forward, I plan to explore NoSQL, cloud databases, and advanced topics to build on this strong foundation.

Contents

1. Introduction

This project develops a Medical Inventory Management System for NGOs to track medicine donations, stock levels, expiry dates, and dispatches to rural clinics. It emphasizes database design, SQL implementation, and real-world applicability in healthcare logistics.

2. Project Objective

Design and implement a system to manage medical inventory for NGOs, including:

- Tracking donations from individuals/corporations
- Monitoring stock levels and expiry dates
- Managing dispatches to rural clinics
- Generating re-order alerts and expiry notifications

3. ER Diagram

This section presents the Entity-Relationship (ER) model used in the system, describing the key entities, relationships, and constraints involved.

Entity Sets and Attributes

- **Donor** (donor_id, name, type, contact, email, address)
- **Medicine** (medicine_id, name, manufacturer, type, description)
- **Inventory** (inventory_id, medicine_id *(FK)*, donor_id *(FK)*, batch_no, quantity, expiry_date)
- **Clinic** (clinic_id, name, location, contact_person)
- **Staff** (staff_id, name, designation, contact)
- **Dispatch** (dispatch_id, inventory_id *(FK)*, clinic_id *(FK)*, staff_id *(FK)*, date, quantity)

Relationship Sets

- DONATES (Donor → Inventory): One-to-Many
- CONTAINS (Medicine → Inventory): One-to-Many
- DISPATCHES (Inventory → Dispatch): One-to-Many

- RECEIVES (Clinic → Dispatch): One-to-Many
- PROCESSES (Staff → Dispatch): One-to-Many

Keys & Constraints

- **Primary Keys**: Underlined (e.g., donor_id)
- **Foreign Keys**:
  - Used to maintain referential integrity across tables (e.g., 'medicine_id' in Inventory references the Medicine entity)
  - Italicized in the diagram (e.g., *donor_id* in Inventory)
- **Structural Constraints**:
  - Cardinality (e.g., one-to-many between Donor and Inventory)
  - Participation (e.g., total participation of Inventory in Dispatch)
  - Crow's foot notation (1:M for all relationships)
- **Assumptions**:
  - Each dispatch is linked to only one inventory item
  - Staff members are responsible for managing dispatch operations
  - A medicine batch is donated by one donor
  - Dispatches are processed by one staff member

The visual representation of the ER Diagram is shown in Figure **??**.

4. Relational Schema

The relational schema consists of:

- **Donor**(donor_id, name, type, contact, email, address, reg_date)
- **Medicine**(medicine_id, name, manufacturer, type, description)
- **Inventory**(inventory_id, *medicine_id*, *donor_id*, batch_no, quantity, expiry_date, status)
- **Clinic**(clinic_id, name, location, contact_person, district, state)
- **Staff**(staff_id, name, designation, contact, email)
- **Dispatch**(dispatch_id, *inventory_id*, *clinic_id*, *staff_id*, date, quantity, purpose)

5. SQL Implementation and its Snapshots

- Table creation SQL
- Insert and update SQL

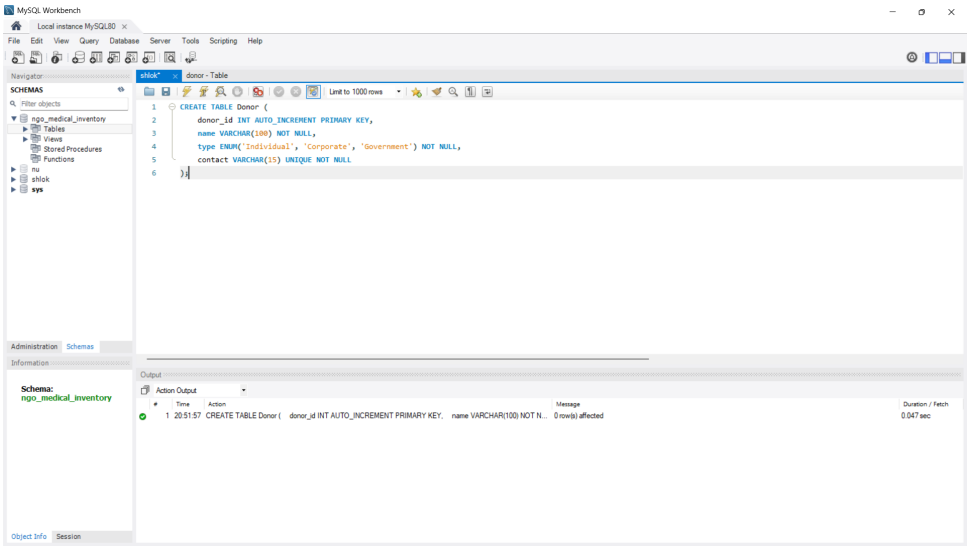

Figure 1: Creating the Donor Table

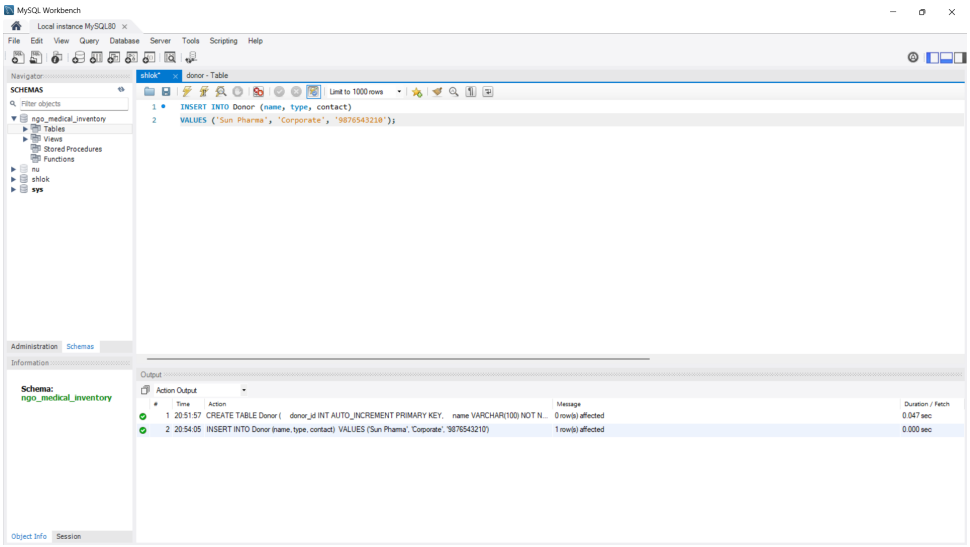

Figure 2: Inserting Data in Table

6. Key Operations

Stock Update Trigger

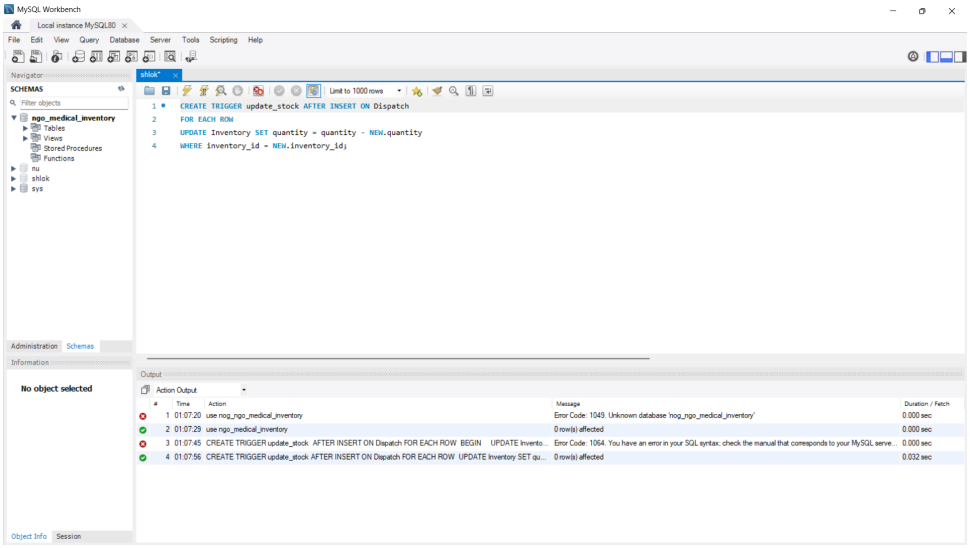

Figure 3: Key Operations – Stock Update Trigger

7. Sample Queries and Outputs along with Snapshots

– **Query:** List medicines expiring within 3 months.

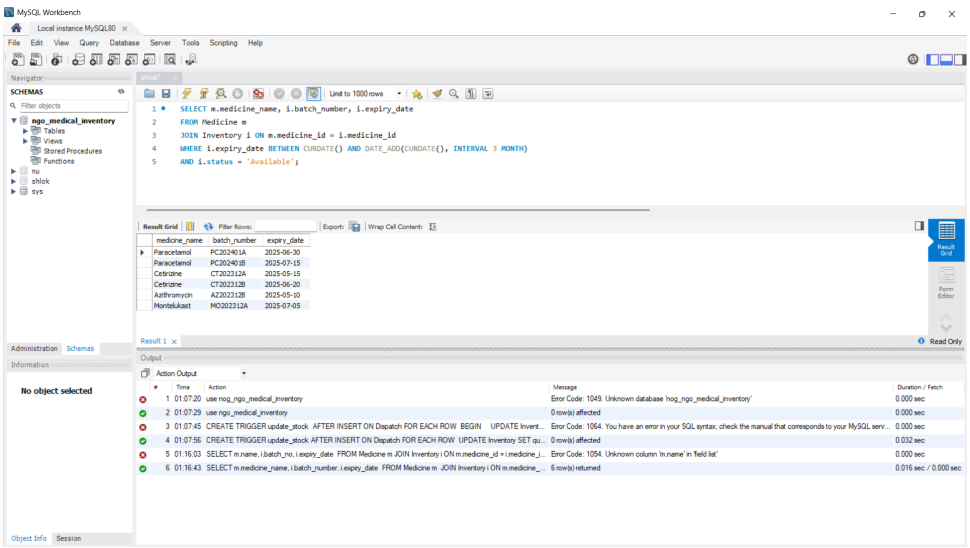

Figure 4: Medicines expiring within 3 months

– **Query:** Total Donations by Donor Type.

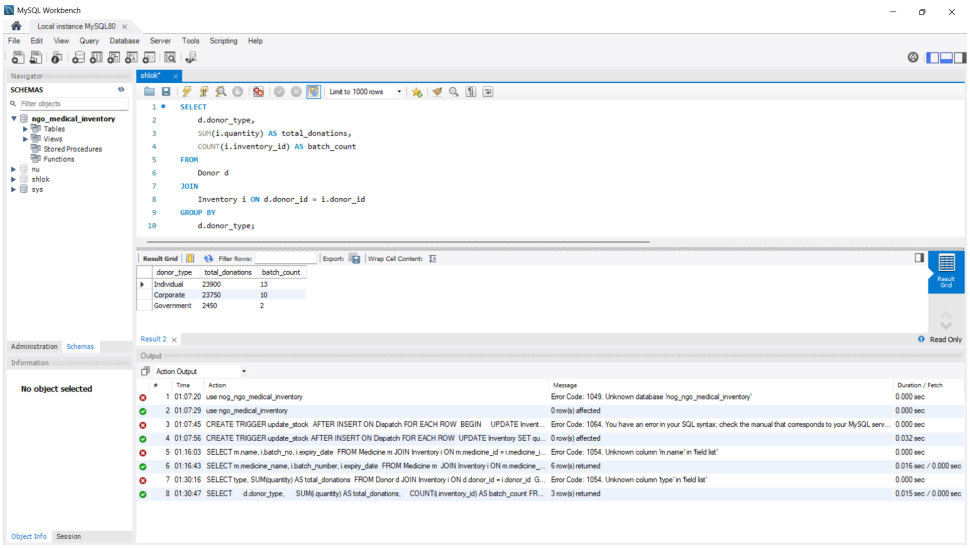

Figure 5: Total Donations by Donor Type

8. SQL Queries

- Selection and Projection

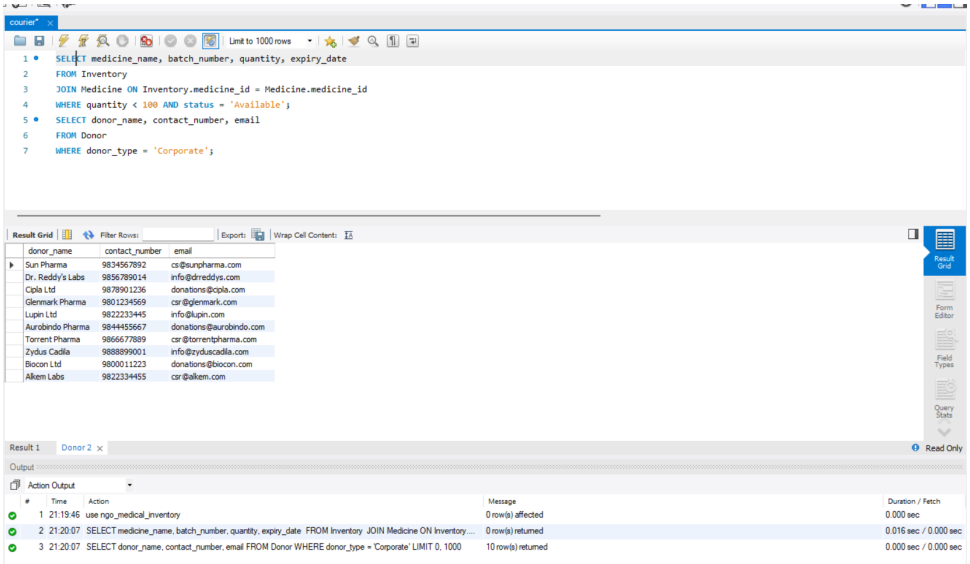

Figure 6: Selection and Projection

- Joins

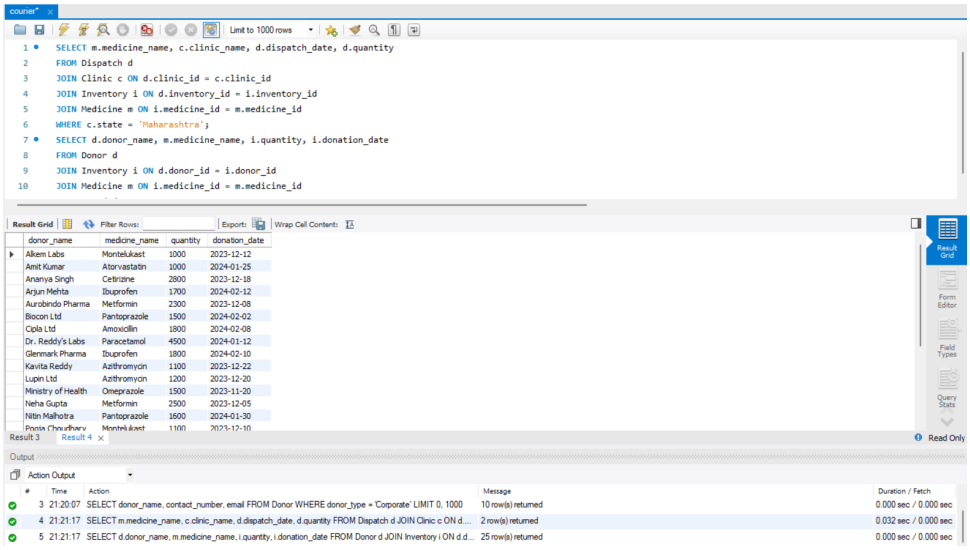

Figure 7: Join Query

- Aggregation and Grouping

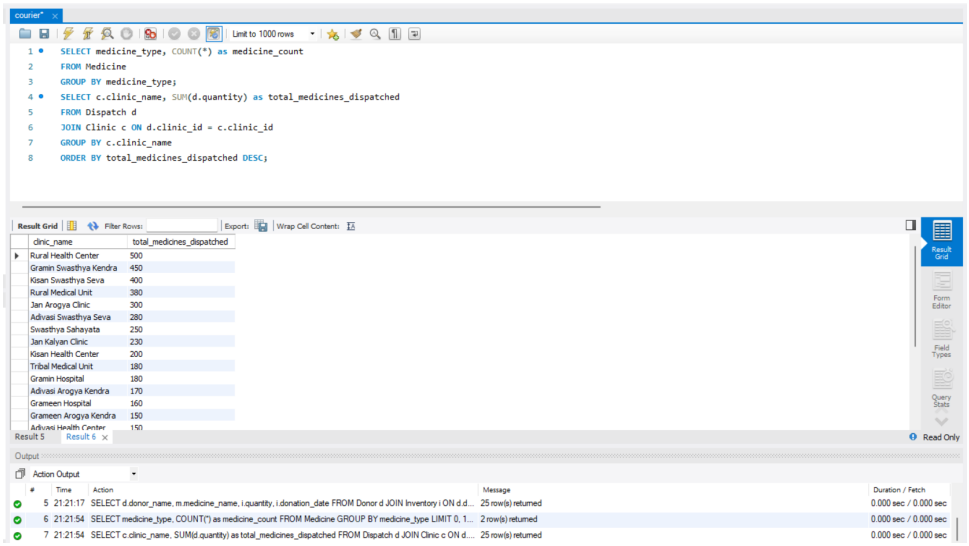

Figure 8: Aggregation and Grouping

- Nested Queries

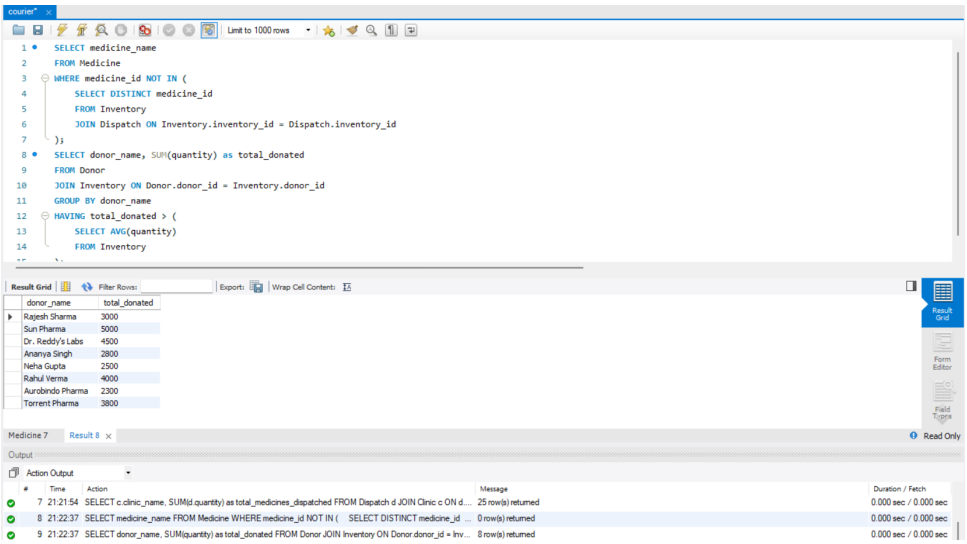

Figure 9: Nested Queries

- Views

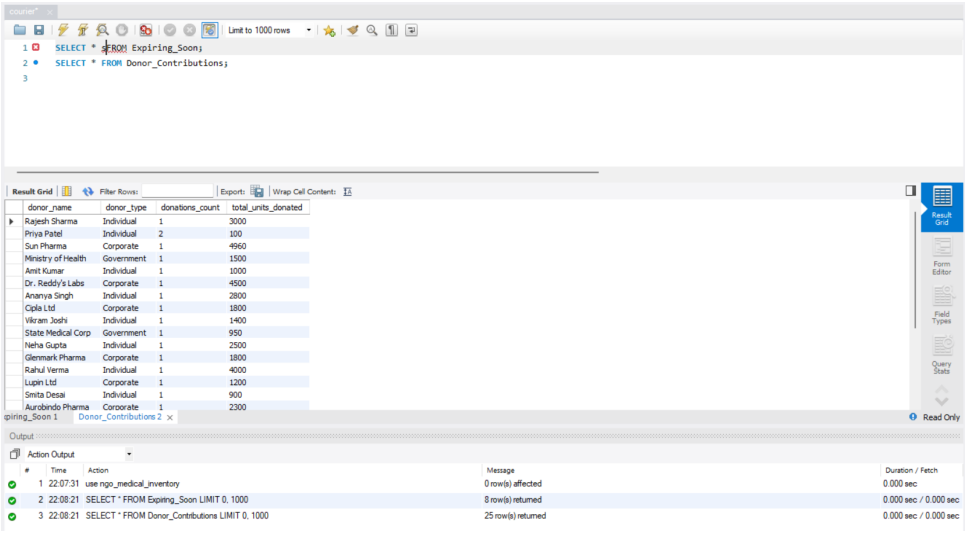

Figure 10: Views

- Indexing

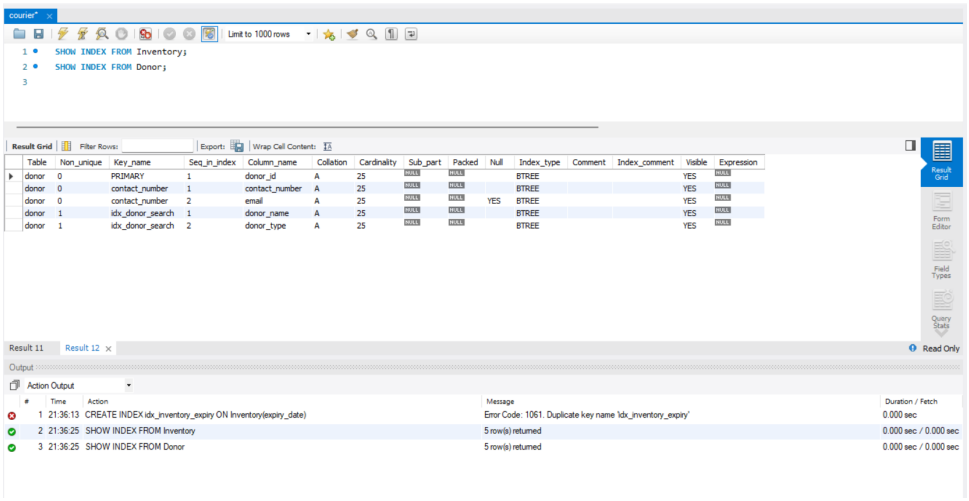

Figure 11: Indexing

- Stored Procedures

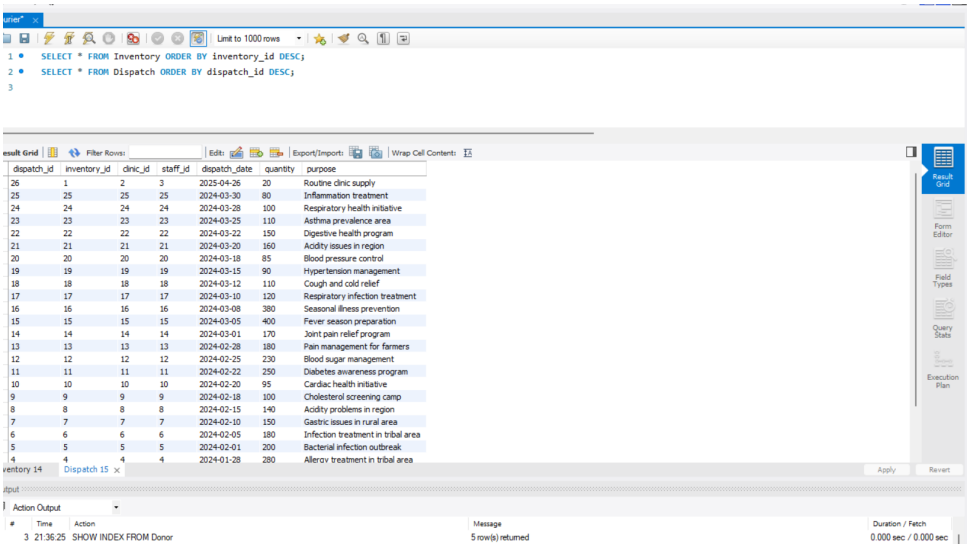

Figure 12: Stored Procedures

- Triggers

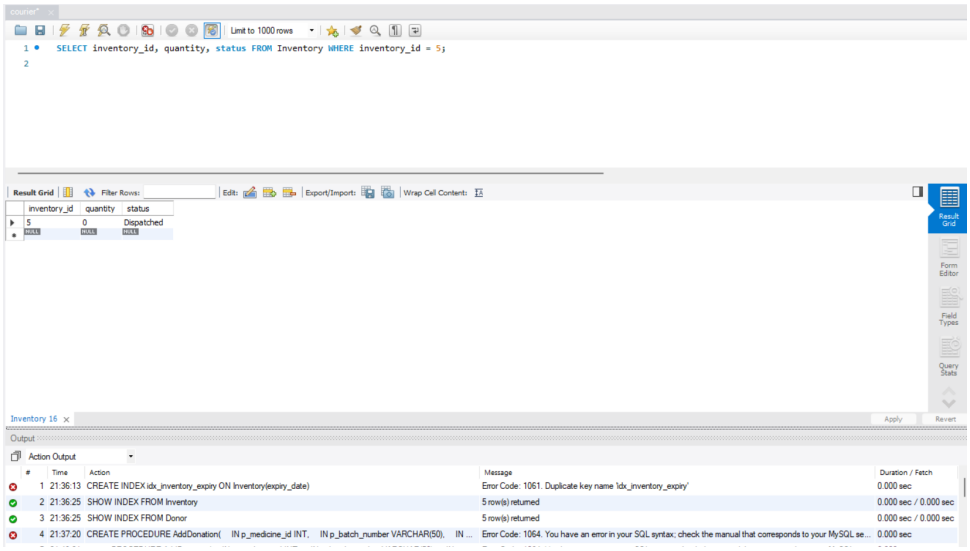

Figure 13: Triggers

9. Challenges Faced

- Complex Joins: Handling 4-table joins for dispatch reports.
- Data Integrity: Ensuring stock updates via triggers.
- Normalization: Balancing clinic locations (denormalized for simplicity).

10. Conclusion

This project demonstrated practical RDBMS design for healthcare logistics, covering schema design, SQL optimization, and real-world constraints like expiry tracking.

11. Project's Learning Outcome

- Designed an ER model with 1:M relationships.
- Implemented triggers for automated stock management.
- Optimized queries for expiry alerts and donor analytics.

12. References

- MySQL Documentation: `https://dev.mysql.com/doc/`
- Database System Concepts, Silberschatz

Appendix: SQL Code

- Full SQL File: Attached as Medical_Inventory.sql with:
  - Table definitions
  - Sample data (25+ records per main table)
  - Views/Triggers
  - All test queries

Appendix: ER Diagram

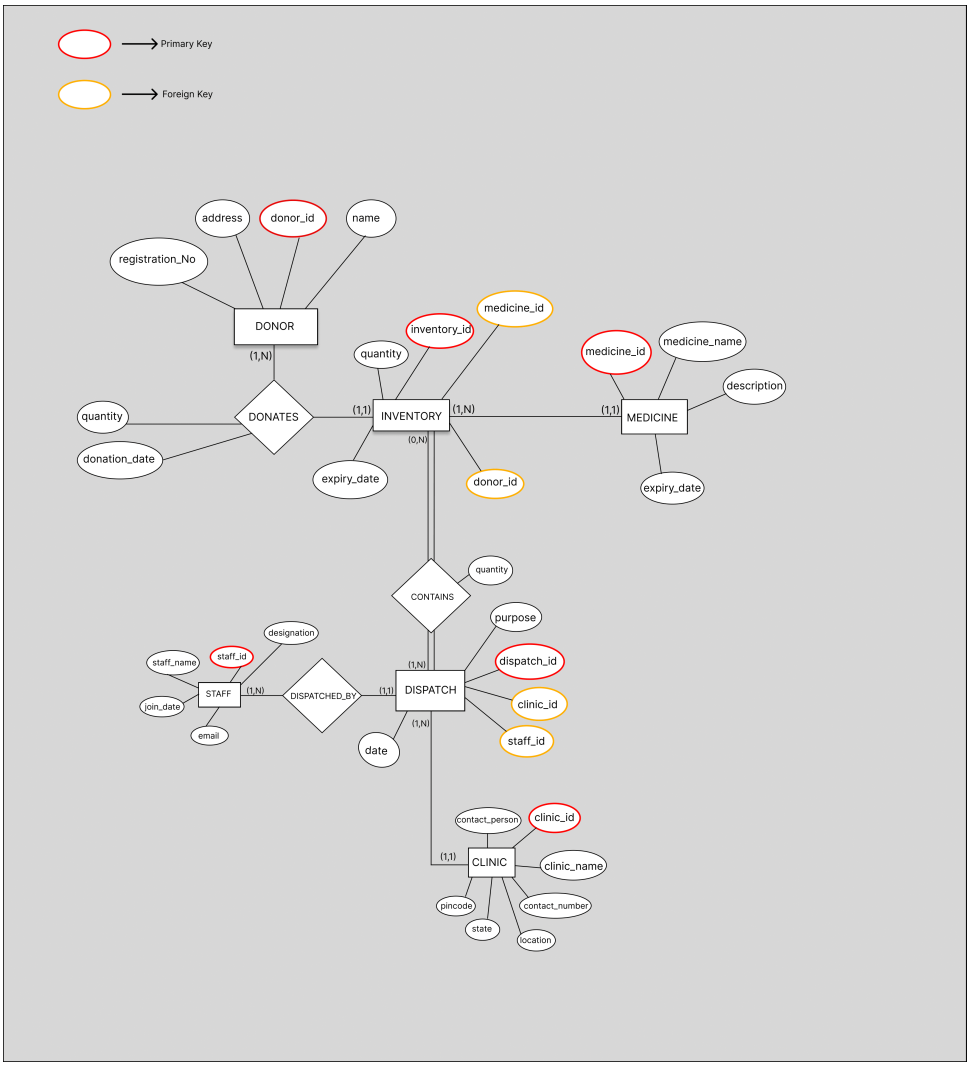

Figure 14: ER Diagram