

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

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Project Objective</b>	<b>3</b>
<b>3</b>	<b>ER Diagram</b>	<b>3</b>
<b>4</b>	<b>Relational Schema</b>	<b>4</b>
<b>5</b>	<b>SQL Implementation and its Snapshots</b>	<b>5</b>
<b>6</b>	<b>Key Operations</b>	<b>6</b>
<b>7</b>	<b>Sample Queries and Outputs along with Snapshots</b>	<b>6</b>
<b>8</b>	<b>SQL Queries</b>	<b>7</b>
<b>9</b>	<b>Challenges Faced</b>	<b>11</b>
<b>10</b>	<b>Conclusion</b>	<b>11</b>
<b>11</b>	<b>Project's Learning Outcome</b>	<b>11</b>
<b>12</b>	<b>References</b>	<b>11</b>

## 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 ??.

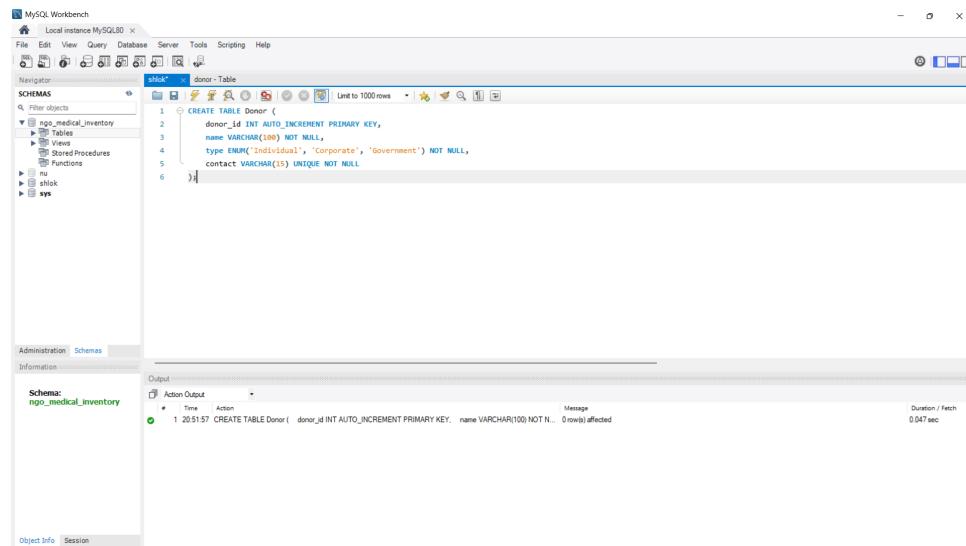
## 4. Relational Schema

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



The screenshot shows the MySQL Workbench interface with the 'donor - Table' editor open. The 'Script' tab contains the SQL code for creating the 'Donor' table:

```

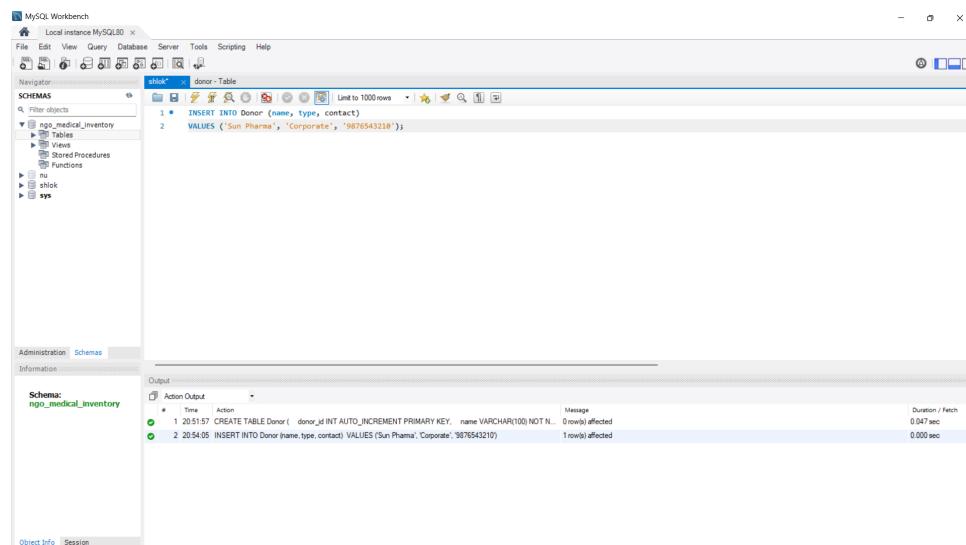
CREATE TABLE Donor (
    donor_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    type ENUM('Individual', 'Corporate', 'Government') NOT NULL,
    contact VARCHAR(15) UNIQUE NOT NULL
);

```

The 'Output' tab shows the execution results:

Action	Time	Message	Duration / Fetch
CREATE TABLE Donor ( donor_id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(100) NOT N... )	1 20:51:57	0 rows affected	0.047 sec

Figure 1: Creating the Donor Table



The screenshot shows the MySQL Workbench interface with the 'donor - Table' editor open. The 'Script' tab contains the SQL code for inserting data into the 'Donor' table:

```

INSERT INTO Donor (name, type, contact)
VALUES ('Sun Pharma', 'Corporate', '9876543210');

```

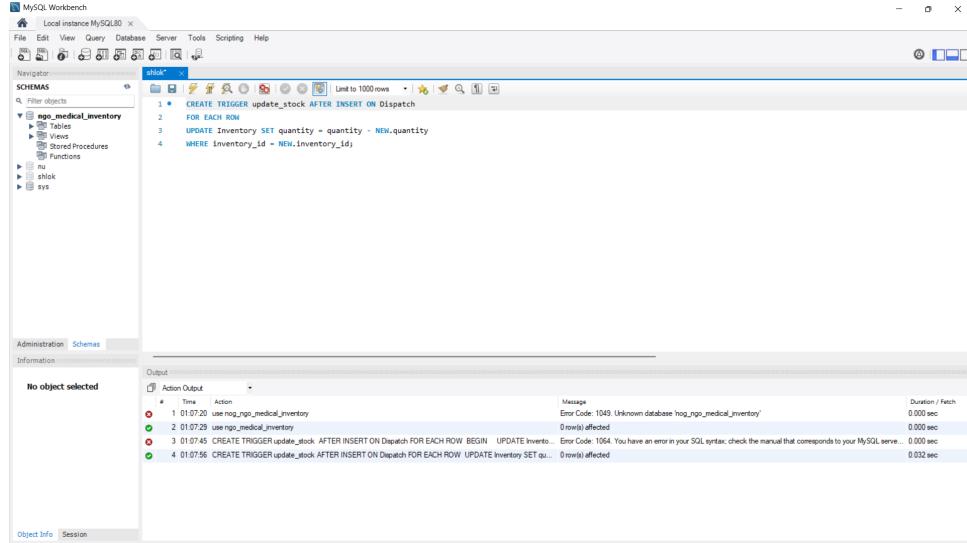
The 'Output' tab shows the execution results:

Action	Time	Message	Duration / Fetch
CREATE TABLE Donor ( donor_id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(100) NOT N... )	1 20:51:57	0 rows affected	0.047 sec
INSERT INTO Donor (name, type, contact) VALUES ('Sun Pharma', 'Corporate', '9876543210')	2 20:54:05	1 row(s) affected	0.000 sec

Figure 2: Inserting Data in Table

## 6. Key Operations

### Stock Update Trigger



The screenshot shows the MySQL Workbench interface with the 'Schemas' tree on the left containing the 'npo\_medical\_inventory' database. In the central query editor, the following SQL code is written:

```

1 • CREATE TRIGGER update_stock AFTER INSERT ON Dispatch
2   FOR EACH ROW
3     UPDATE Inventory SET quantity = quantity - NEW.quantity
4   WHERE inventory_id = NEW.inventory_id

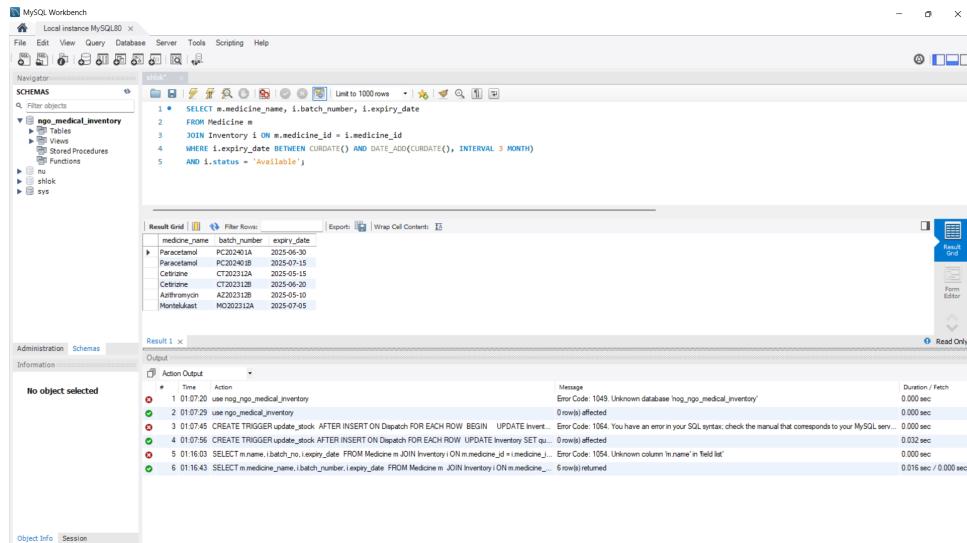
```

The 'Output' pane at the bottom shows the execution results for each line of the script. Lines 1, 2, and 3 execute successfully, but line 4 fails with the error: "Error Code: 1049 Unknown database 'npo\_npo\_medical\_inventory'". Line 5 shows the creation of the trigger successfully.

Figure 3: Key Operations – Stock Update Trigger

## 7. Sample Queries and Outputs along with Snapshots

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



The screenshot shows the MySQL Workbench interface with the 'Schemas' tree on the left containing the 'npo\_medical\_inventory' database. In the central query editor, the following SQL query is written:

```

1 • SELECT m.medicine_name, i.batch_number, i.expiry_date
2   FROM Medicine m
3   JOIN Inventory i ON m.medicine_id = i.medicine_id
4   WHERE i.expiry_date BETWEEN CURDATE() AND DATE_ADD(CURDATE(), INTERVAL 3 MONTH)
5   AND i.status = 'Available'

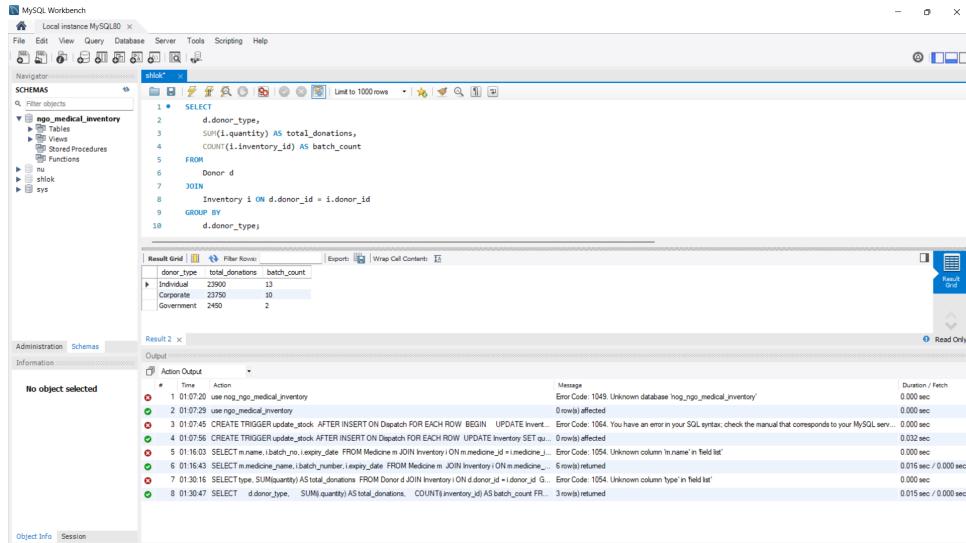
```

The 'Result Grid' pane displays the results of the query, listing medicines like Paracetamol, Paracetamol, Cetirizine, Cetirizine, Aftershave, and Monalista with their batch numbers and expiry dates.

The 'Output' pane at the bottom shows the execution results for each line of the query. Lines 1 through 5 execute successfully, and line 6 shows the result of the query itself, which returns 6 rows.

Figure 4: Medicines expiring within 3 months

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



```

1 • SELECT
2     d.donor_type,
3     SUM(i.quantity) AS total_donations,
4     COUNT(i.inventory_id) AS batch_count
5   FROM
6     Donor d
7   JOIN
8     Inventory i ON d.donor_id = i.donor_id
9   GROUP BY
10    d.donor_type;

```

The screenshot shows the MySQL Workbench interface. The top pane displays the SQL query. The bottom pane shows the results of the query execution. The results table has three columns: donor\_type, total\_donations, and batch\_count. The data is as follows:

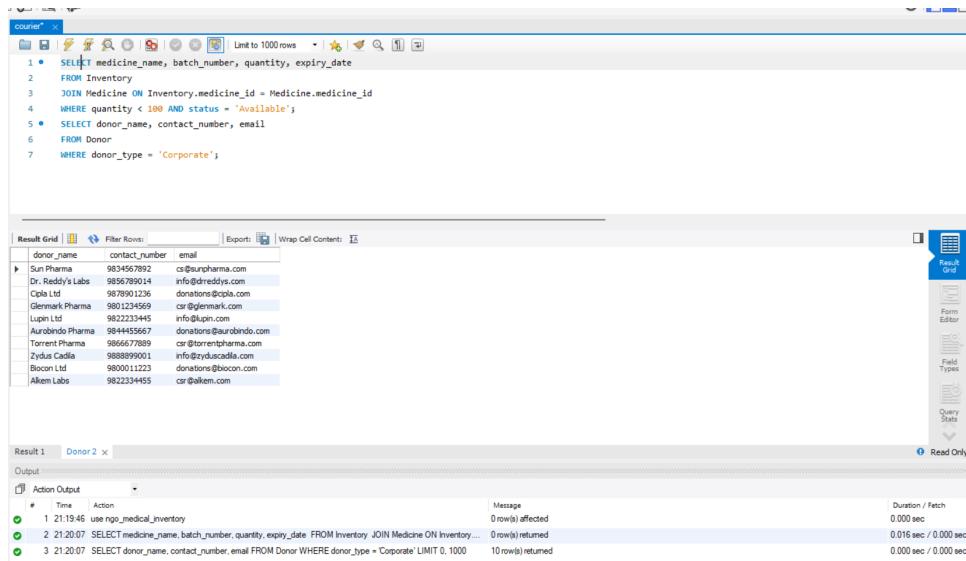
donor_type	total_donations	batch_count
Individual	23900	13
Corporate	23790	10
Government	2460	2

The status bar at the bottom right indicates "Read Only".

Figure 5: Total Donations by Donor Type

## 8. SQL Queries

- Selection and Projection



```

1 • SELECT medicine_name, batch_number, quantity, expiry_date
2   FROM Inventory
3   JOIN Medicine ON Inventory.medicine_id = Medicine.medicine_id
4   WHERE quantity < 100 AND status = 'Available'
5 •   SELECT donor_name, contact_number, email
6   FROM Donor
7   WHERE donor_type = 'Corporate';

```

The screenshot shows the MySQL Workbench interface. The top pane displays the SQL query. The bottom pane shows the results of the query execution. The results table has four columns: donor\_name, contact\_number, email, and medicine\_name. The data is as follows:

donor_name	contact_number	email	medicine_name
Sir Pharma	9834578992	cs@pharma.com	
Dr. Reddy's Labs	9856789014	info@drredlys.com	
Cipla Ltd	9878901236	donations@cipla.com	
Glenmark Pharma	9801234569	car@glenmark.com	
Lupin Ltd	9822233445	info@lupin.com	
Aurobindo Pharma	984445567	donations@aurobindo.com	
Torrent Pharma	9866677889	cr@torrentpharma.com	
Zydus Cadila	9888899001	info@zyduscadila.com	
Biocon Ltd	9800011223	donations@biocon.com	
Aklem Labs	9822334455	cr@aklem.com	

The status bar at the bottom right indicates "Read Only".

Figure 6: Selection and Projection

- Joins

The screenshot shows a MySQL Workbench interface with a query editor and a results grid. The query is:

```

1 •  SELECT m.medicine_name, c.clinic_name, d.dispatch_date, d.quantity
2   FROM Dispatch d
3   JOIN Clinic c ON d.clinic_id = c.clinic_id
4   JOIN Inventory i ON d.inventory_id = i.inventory_id
5   JOIN Medicine m ON i.medicine_id = m.medicine_id
6   WHERE c.state = 'Maharashtra';
7 •  SELECT d.donor_name, m.medicine_name, i.quantity, i.donation_date
8   FROM Donor d
9   JOIN Inventory i ON d.donor_id = i.donor_id
10  JOIN Medicine m ON i.medicine_id = m.medicine_id
    
```

The results grid displays the following data:

donor_name	medicine_name	quantity	donation_date
Alken Labs	Montelukast	1000	2023-12-12
Amit Kumar	Atorvastatin	1000	2024-01-25
Ananya Singh	Cetirizine	2800	2023-12-18
Arjun Mehta	Ibuprofen	1700	2024-01-05
Biocon Pharma	Mefenamic acid	2000	2024-02-09
Biocon Ltd	Pantoprazole	1900	2024-02-02
Cipla Ltd	Amoxicillin	1800	2024-02-08
Dr. Reddy's Labs	Paracetamol	4500	2024-01-12
Glenmark Pharma	Ibuprofen	1800	2024-02-10
Kavita Reddy	Azithromycin	110	2023-12-22
Lupin Ltd	Azithromycin	120	2023-12-20
Ministry of Health	Omeprazole	1500	2023-11-20
Neha Gupta	Metformin	2500	2023-12-05
Novartis Pharma	Paracetamol	1600	2024-01-30
Pratap Choudhary	Montelukast	1100	2023-12-10

Output panel shows the execution log:

```

# Time Action Message Duration / Fetch
1 3 21:20:07 SELECT donor_name, contact_number, email FROM Donor WHERE donor_type = 'Corporate' LIMIT 0, 1000 10 row(s) returned 0.000 sec / 0.000 sec
2 4 21:21:17 SELECT m.medicine_name, c.clinic_name, d.dispatch_date, d.quantity FROM Dispatch d JOIN Clinic c ON d.clinic_id = c.clinic_id 2 row(s) returned 0.032 sec / 0.000 sec
3 5 21:21:17 SELECT d.donor_name, m.medicine_name, i.quantity, i.donation_date FROM Donor d JOIN Inventory i ON d.donor_id = i.donor_id 25 row(s) returned 0.000 sec / 0.000 sec
    
```

Figure 7: Join Query

- Aggregation and Grouping

The screenshot shows a MySQL Workbench interface with a query editor and a results grid. The query is:

```

1 •  SELECT medicine_type, COUNT(*) AS medicine_count
2   FROM Medicine
3   GROUP BY medicine_type;
4 •  SELECT c.clinic_name, SUM(d.quantity) AS total_medicines_dispatched
5   FROM Dispatch d
6   JOIN Clinic c ON d.clinic_id = c.clinic_id
7   GROUP BY c.clinic_name
8   ORDER BY total_medicines_dispatched DESC;
    
```

The results grid displays the following data:

clinic_name	total_medicines_dispatched
Rural Health Center	500
Gramin Swasthya Kendra	450
Kisan Swasthya Seva	400
Jan Arogya Clinic	380
Jan Arogya Clinic	300
Adviser Swasthya Seva	280
Swasthya Sahayata	250
Jan Kalyan Clinic	230
Kisan Health Center	200
Tribal Medical Unit	180
Gramin Hospital	180
Jan Arogya Kendra	170
Gramen Hospital	160
Gramen Arogya Kendra	150
Advices Health Center	150

Output panel shows the execution log:

```

# Time Action Message Duration / Fetch
1 5 21:21:17 SELECT d.donor_name, m.medicine_name, i.quantity, i.donation_date FROM Donor d JOIN Inventory i ON d.donor_id = i.donor_id 25 row(s) returned 0.000 sec / 0.000 sec
2 6 21:21:54 SELECT medicine_type, COUNT(*) AS medicine_count FROM Medicine GROUP BY medicine_type LIMIT 0, 1.. 2 row(s) returned 0.000 sec / 0.000 sec
3 7 21:21:54 SELECT c.clinic_name, SUM(d.quantity) AS total_medicines_dispatched FROM Dispatch d JOIN Clinic c ON d.clinic_id = c.clinic_id 25 row(s) returned 0.000 sec / 0.000 sec
    
```

Figure 8: Aggregation and Grouping

- Nested Queries

The screenshot shows a database interface with a query editor and a results grid.

```

1 • SELECT medicine_name
  FROM Medicine
  WHERE medicine_id NOT IN (
    SELECT DISTINCT medicine_id
    FROM Inventory
    JOIN Dispatch ON Inventory.inventory_id = Dispatch.inventory_id
  );
1 • SELECT donor_name, SUM(quantity) as total_donated
  FROM Donor
  JOIN Inventory ON Donor.donor_id = Inventory.donor_id
  GROUP BY donor_name
  HAVING total_donated > (
    SELECT AVG(quantity)
    FROM Inventory
  );
  
```

**Result Grid:**

donor_name	total_donated
Rajesh Sharma	3000
Sun Pharma	5000
Amrit Kaur & Sons Labs	4500
Ananya Singh	2800
Neha Gupta	2500
Rahul Verma	4000
Aurobindo Pharma	2300
Torrent Pharma	3800

**Action Output:**

#	Time	Action	Message	Duration / Fetch
1	7 21:21:54	SELECT c.clinic_name, SUM(d.quantity) as total_medicines_dispatched FROM Dispatch d JOIN Clinic c ON ... 25 row(s) returned	0.000 sec / 0.000 sec	
2	8 21:22:37	SELECT medicine_name FROM Medicine WHERE medicine_id NOT IN ( ... ) 0 row(s) returned	0.000 sec / 0.000 sec	
3	9 21:22:37	SELECT donor_name, SUM(quantity) as total_donated FROM Donor JOIN Inventory ON Donor.donor_id = Inv... 8 row(s) returned	0.000 sec / 0.000 sec	

Figure 9: Nested Queries

- Views

The screenshot shows a database interface with a query editor and a results grid.

```

1 • SELECT * FROM Expiring_Soon;
2 • SELECT * FROM Donor_Contributions;
  
```

**Result Grid:**

donor_name	donor_type	donations_count	total_units_donated
Rajesh Sharma	Individual	1	3000
Priva Patel	Individual	2	100
Sun Pharma	Corporate	1	4950
Ministry of Health	Government	1	1500
Amit Kumar	Individual	1	1000
Amrit Kaur & Sons Labs	Corporate	1	4500
Ananya Singh	Individual	1	2800
Cipla Ltd	Corporate	1	1800
Vikram Joshi	Individual	1	1400
State Medical Corp	Government	1	950
Neha Gupta	Individual	1	2500
Glenmark Pharma	Corporate	1	1800
Rahul Verma	Individual	1	4000
Smita Desai	Corporate	1	1200
Aurobindo Pharma	Corporate	1	900
spiring_Soon_1	Donor_Contributions_2		

**Action Output:**

#	Time	Action	Message	Duration / Fetch
1	22:07:31	use ngo_medical_inventory	0 row(s) affected	0.000 sec
2	22:08:21	SELECT * FROM Expiring_Soon LIMIT 0, 1000	8 row(s) returned	0.000 sec / 0.000 sec
3	22:08:21	SELECT * FROM Donor_Contributions LIMIT 0, 1000	25 row(s) returned	0.000 sec / 0.000 sec

Figure 10: Views

- Indexing

The screenshot shows the MySQL Workbench interface with the 'courier' database selected. The 'Result Grid' pane displays the output of the 'SHOW INDEX' command for the 'Inventory' and 'Donor' tables. The 'Inventory' table has one primary key index ('PRIMARY') and one regular index ('idx\_inventory\_expiry'). The 'Donor' table has four indexes: 'PRIMARY' (primary key), 'contact\_number' (regular), 'email' (regular), and two secondary indexes ('idx\_donor\_search' and 'idx\_donor\_type'). The 'Action Output' pane shows the execution history of the commands.

Table	Non_Unique	Key_name	Seq_in_Index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
donor	0	PRIMARY	1	donor_id	A	25				BTREE			YES	
donor	0	contact_number	1	contact_number	A	25				BTREE			YES	
donor	0	contact_number	2	email	A	25				BTREE			YES	
donor	1	idx_donor_search	1	donor_name	A	25				BTREE			YES	
donor	1	idx_donor_search	2	donor_type	A	25				BTREE			YES	

**Action Output:**

#	Time	Action	Message	Duration / Fetch
1	21:36:13	CREATE INDEX idx_inventory_expiry ON Inventory(expiry_date)	Error Code: 1061. Duplicate key name 'idx_inventory_expiry'	0.000 sec
2	21:36:25	SHOW INDEX FROM Inventory	5 row(s) returned	0.000 sec / 0.000 sec
3	21:36:25	SHOW INDEX FROM Donor	5 row(s) returned	0.000 sec / 0.000 sec

Figure 11: Indexing

- Stored Procedures

The screenshot shows the MySQL Workbench interface with the 'user' database selected. The 'Result Grid' pane displays the output of 'SELECT \*' queries for the 'Inventory' and 'Dispatch' tables. The 'Action Output' pane shows the execution history of the 'SHOW INDEX' command for the 'Donor' table.

dispatch_id	inventory_id	donor_id	staff_id	dispatch_date	quantity	purpose
26	1	2	3	2024-03-26	20	Routine clinic supply
25	25	25	25	2024-03-30	80	Inflammation treatment
24	24	24	24	2024-03-28	100	Respiratory health initiative
23	23	23	23	2024-03-25	110	Asthma prevalence area
22	22	22	22	2024-03-22	150	Digestive health program
21	21	21	21	2024-03-20	160	Acidity issues in region
20	20	20	20	2024-03-18	85	Blood pressure control
19	19	19	19	2024-03-16	95	Hypertension management
18	18	18	18	2024-03-12	110	Cough and cold relief
17	17	17	17	2024-03-10	120	Respiratory infection treatment
16	16	16	16	2024-03-08	380	Seasonal illness prevention
15	15	15	15	2024-03-05	400	Fever season preparation
14	14	14	14	2024-03-01	170	Joint pain relief program
13	13	13	13	2024-02-28	180	Pain management for farmers
12	12	12	12	2024-02-25	230	Blood sugar management
11	11	11	11	2024-02-22	250	Diabetes awareness program
10	10	10	10	2024-02-20	95	Cardiac health initiative
9	9	9	9	2024-02-18	100	Cholesterol management
8	8	8	8	2024-02-15	140	Arthritis problems in region
7	7	7	7	2024-03-10	150	Gastric issues in rural area
6	6	6	6	2024-02-05	180	Infection treatment in tribal area
5	5	5	5	2024-02-01	200	Bacterial infection outbreak
4	4	4	4	2024-01-28	280	Allergy treatment in tribal area

**Action Output:**

#	Time	Action	Message	Duration / Fetch
3	21:36:25	SHOW INDEX FROM Donor	5 row(s) returned	0.000 sec / 0.000 sec

Figure 12: Stored Procedures

- Triggers

The screenshot shows the MySQL Workbench interface. At the top, there's a toolbar with various icons. Below it is a query editor window containing the SQL command: `SELECT inventory_id, quantity, status FROM Inventory WHERE Inventory_id = 5;`. The result grid below the query shows one row of data: inventory\_id 5, quantity 0, and status Dispatched. To the right of the result grid is a vertical panel with tabs for 'Result Grid', 'Form Editor', 'Field Types', and 'Query Stats'. Below the result grid is an 'Output' tab which displays the execution log:

#	Time	Action	Message	Duration / Fetch
1	21:36:13	CREATE INDEX idx_inventory_expiry ON Inventory(expiry_date)	Error Code: 1061. Duplicate key name 'idx_inventory_expiry'	0.000 sec
2	21:36:25	SHOW INDEX FROM Inventory	5 row(s) returned	0.000 sec / 0.000 sec
3	21:36:25	SHOW INDEX FROM Donor	5 row(s) returned	0.000 sec / 0.000 sec
4	21:37:20	CREATE PROCEDURE AddDonation( IN p_medicine_id INT, IN p_batch_number VARCHAR(50), IN ... )	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL se...	0.000 sec

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

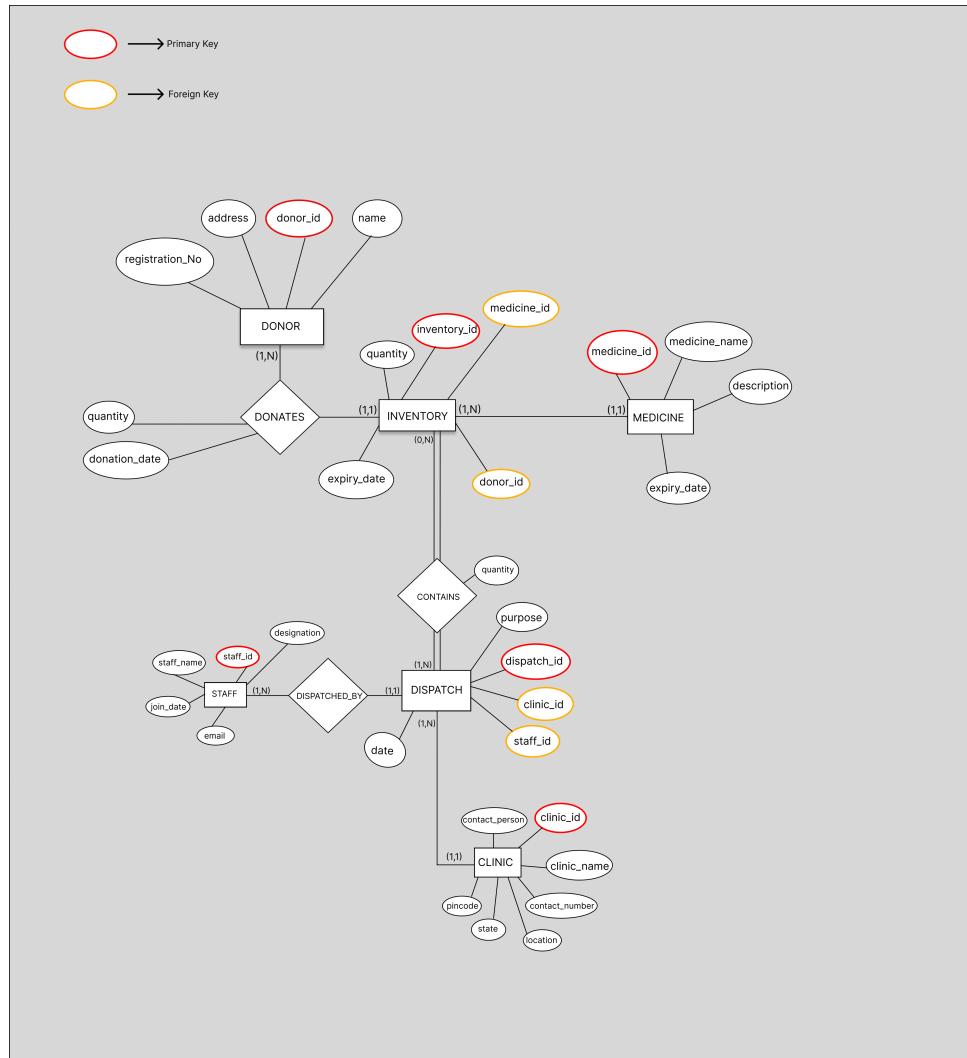


Figure 14: ER Diagram