Department of Computer Science and Engineering

CSE 2211: Database Management System - 1

Project Title: Blood Bank management system

Submitted to:
Dr. Md. Mustafizur Rahman
Dr. Muhammad Ibrahim

Submitted by:
Md. Sadman Sakib
Roll: 49

Project Title

Blood bank management system

Project Area Major

Health

Project Area Minor

Blood Bank

Brief Description of the Database

The blood bank database was designed to effectively manage the operations and supply chain of a network of blood banks in a city. As blood transfusions are a critical healthcare need, the key purpose is to ensure blood supply are available on demand for patients in nearby hospitals.

This centralized database allows donors, blood bank inventory, patient requests, and hospital requirements to be tracked in an integrated system accessible across all locations. Key entities captured include donors, the registration team, blood bank managers, technical analysts, blood supply, and patient receivers.

Some unique aspects include:
- Linking donors to blood supply to enable efficient matching based on blood groups
- Tracking patient transfusion requests to analyze demand and supply needs
- Connecting hospitals to the blood banks for direct ordering and fulfillment
- Detailed donor eligibility screening as per regulatory guidelines on age, health, etc.

By interconnecting donors, inventory, patients, and hospitals, this database supports the core function of making life-saving blood available when needed. Automated tracking and analysis allow faster identification of rare or unusual blood groups, targeted donor drives, and optimal cross-location inventory allocation.

Overall, this centralized city-wide system enables each blood bank to operate independently while coordinating effectively as a network to maximize blood availability. The integrated database is essential for the efficient, responsive, and life-critical operations of the blood bank system.

Detailed Description of the Database

A Blood Bank is a place where blood is collected and stored for use by others who need it. it due to health emergencies or a lack of blood. There is an urgent need for a centralized, city-wide blood bank management system. Currently, each blood bank in the city operates individually with manual and disjointed processes. This leads to critical inefficiencies:

- Difficulty matching patient blood requests to supply across banks
- Sub-optimal blood inventory management and shortages
- Delayed disaster response and patient outcomes
- Donor drives are not coordinated, leading to blood group imbalance;
- High pricing due to artificial storage and out of date blood

There is no unified view of the blood inventory levels and donor base across the city. Hospital orders take too long to fulfill due to manual paperwork and records. Valuable tested blood units expire frequently due to a lack of visibility.

The BloodBank database is designed to address various problems and challenges related to the management of blood donation, distribution, and related healthcare processes.

Problem: Inefficient tracking and management of blood units and their associated information.
Solution: The Blood and Blood_Patient tables help manage the inventory of blood units, their types, and the patients receiving them. This facilitates effective tracking and organization of available blood resources.
Donor Information and Health Screening:

Problem: Lack of organized information about blood donors, including their health status and donation history.
Solution: The Donor table captures detailed information about donors, including health conditions, allowing for efficient screening and management of eligible donors.
Blood Requests and Distribution:

Problem: Ineffective coordination and tracking of blood requests from hospitals and patients.
Solution: The Requests, Patient_Requests, Hospital_Requests tables facilitate the systematic recording and monitoring of blood requests, ensuring a more organized and responsive distribution process.

Patient Information and Registration:

Problem: Challenges in managing patient records and their blood-related information.
Solution: The Patient table addresses this issue by storing comprehensive details about patients, including their contact information, blood type, and the registration team managing their records.
Registration Team and Technical Analysts:

Problem: Lack of organized information about the teams responsible for donor registration and technical analysts involved in blood processing.
Solution: The Registration_Team and Technical_Analyst tables help manage information about the teams and analysts, facilitating better coordination and communication in the blood donation and processing workflow.
Blood Bank and Hospital Management:

Problem: Inadequate organization and tracking of blood bank and hospital details.
Solution: The BloodBank, Manager_BloodBank, Hospital, and associated tables help manage information about blood banks and hospitals, ensuring that the right personnel are connected to the relevant entities.
Managerial Oversight and Accountability:

Problem: Challenges in ensuring managerial oversight and accountability in the blood donation and distribution process.
Solution: The use of the Manager table and its connections to other tables ensures that managers are associated with specific blood banks, hospitals, and other relevant entities, enhancing accountability and efficient management.
Many-to-Many Relationships:

Problem: Difficulty in representing and managing complex relationships, such as those between managers and blood banks, patients and blood requests, and technical analysts and blood units.
Solution: The use of associative tables like Manager_BloodBank, Patient_Requests, TechnicalAnalyst_Blood, and others helps handle many-to-many relationships efficiently.
In summary, the BloodBank database aims to solve problems related to the efficient management, tracking, and coordination of various aspects of the blood donation and healthcare process, ultimately contributing to the improvement of blood bank operations and patient care.

Project scopes:

Donor Management
- Registering and tracking details and donation eligibility over time
- Managing donor drives and campaigns

Blood Inventory Control
- Recording entire lifecycle from collection to component separation to testing
- Inventory and storage monitoring with expiration tracking
- Distribution and transportation to hospitals

Patient and Hospital Management
- Registration of patient transfusion requests
- Matching of patient to donated blood
- Hospital blood orders and fulfillment

Centralized Access and Reporting
- Searchable central database accessible by all banks
- Dashboards and reports on demand, supply, stock levels

Donor table: Records details about individuals who donate blood, including unique donor id, name, contact info, eligibility criteria like age/health, and donation dates. Enforces 30-day minimum interval between donations.

Blood table: Maintains information about donated blood units, linking to donor sources. Includes unique blood id, blood group, and manager overseeing the collection.

Patient table: Captures patient details requiring blood transfusions, including unique ids, names, blood groups needed, contact and location details, registration dates, and linking to registration staff.

Request table: Logs requests for blood, containing the blood group and quantity needed. Links patient needs to hospital requirements.

Hospital table: Stores information on hospitals placing blood requests, including id, name, address, and manager contact.

Blood bank table: Central entity for blood bank centers, holding id, name, location details.

Registration team table: Has staff responsible for patient registrations at each blood bank branch, with ids, names and parent blood bank links.

Technical analyst table: Maintains technicians testing blood samples at each bank, tied to parent bank.

The system interconnects blood donors, inventory, patients and hospitals for managing the end-to-end blood supply chain. Key functions cover donor drives, testing/storage, patient matching and orders, inventory control, issuance and replenishment - coordinated across the network.

functionality of each table in the BloodBank database:

Manager Table (manager):

Purpose: This table stores information about the managers responsible for overseeing the operations of blood banks.
Fields:
idManager (Primary Key): Unique identifier for each manager.
nameM: Manager's first name.
surnameM: Manager's last name.

BloodBank Table (bloodbank):

Purpose: This table contains details about individual blood banks, including their location.
Fields:
idBloodBank (Primary Key): Unique identifier for each blood bank.
nameBB: Name of the blood bank.
avenue_street: Street or avenue where the blood bank is located.
building_number: Building number of the blood bank.
neighborhood: Neighborhood where the blood bank is situated.
city: City where the blood bank is located.

Manager_BloodBank Table (manager_bloodbank):

Purpose: This table establishes a many-to-many relationship between managers and blood banks, indicating which managers are responsible for which blood banks.
Fields:
idManager (Foreign Key referencing manager): Connects to the Manager Table.
idBloodBank (Foreign Key referencing bloodbank): Connects to the BloodBank Table.

Registration_Team Table (registration_team):

Purpose: Records information about registration teams involved in the blood donation process, such as those responsible for donor registration.
Fields:
idRT (Primary Key): Unique identifier for each registration team.
nameER: First name of the registration team member.
surnameER: Last name of the registration team member.
idBloodBank (Foreign Key referencing bloodbank): Connects to the BloodBank Table.

Technical_Analyst Table (technical_analyst):

Purpose: This table stores information about technical analysts associated with blood banks, providing expertise in analyzing and processing blood.
Fields:
idTA (Primary Key): Unique identifier for each technical analyst.
nameTA: First name of the technical analyst.
surnameAT: Last name of the technical analyst.
idBloodBank (Foreign Key referencing bloodbank): Connects to the BloodBank Table.

Blood Table (blood):

Purpose: Manages information about individual blood units, including the blood type and the manager responsible for the blood bank.
Fields:
idBlood (Primary Key): Unique identifier for each blood unit.
bloodGroup: Blood type of the unit.
idManager (Foreign Key referencing manager): Connects to the Manager Table.

Patient Table (patient):

Purpose: Records details about patients, including their contact information, address, and the registration team and manager overseeing their records.
Fields:
idPatient (Primary Key): Unique identifier for each patient.
nameP: First name of the patient.
surnameP: Last name of the patient.
gender: Gender of the patient.
bloodGroup: Blood type of the patient.
contact: Contact information for the patient.
avenue_street: Street or avenue where the patient resides.
building_number: Building number of the patient's residence.
neighborhood: Neighborhood where the patient resides.
city: City where the patient resides.

dateRegisters: Date when the patient's information was registered.
idRT (Foreign Key referencing registration_team): Connects to the Registration_Team Table.
idManager (Foreign Key referencing manager): Connects to the Manager Table.

Requests Table (requests):

Purpose: Manages requests for blood, specifying the blood type and the required quantity.
Fields:
idRequests (Primary Key): Unique identifier for each blood request.
bloodGroup: Blood type requested.
amountBlood: Quantity of blood requested.

Patient_Requests Table (patient_requests):

Purpose: Establishes a many-to-many relationship between patients and blood requests, indicating which patients have requested which types and quantities of blood.
Fields:
idPatient (Foreign Key referencing patient): Connects to the Patient Table.
idRequests (Foreign Key referencing requests): Connects to the Requests Table.

Hospital Table (hospital):

Purpose: Stores information about hospitals, including their location and the manager responsible for the hospital.
Fields:
idHospital (Primary Key): Unique identifier for each hospital.
nameH: Name of the hospital.
avenue_street: Street or avenue where the hospital is located.
building_number: Building number of the hospital.
neighborhood: Neighborhood where the hospital is situated.
city: City where the hospital is located.
idManager (Foreign Key referencing manager): Connects to the Manager Table.

Hospital_Requests Table (hospital_requests):

Purpose: Establishes a many-to-many relationship between hospitals and blood requests, indicating which hospitals have requested which types and quantities of blood.
Fields:
idHospital (Foreign Key referencing hospital): Connects to the Hospital Table.
idRequests (Foreign Key referencing requests): Connects to the Requests Table.

Blood_Patient Table (blood_patient):

Purpose: Manages the relationship between blood units and patients, indicating which patients have received which blood units.
Fields:
idBlood (Foreign Key referencing blood): Connects to the Blood Table.
bloodGroup: Blood type of the unit.
idPatient (Foreign Key referencing patient): Connects to the Patient Table.

TechnicalAnalyst_Blood Table (technicalAnalyst_blood):

Purpose: Establishes a many-to-many relationship between technical analysts and blood units, indicating which technical analysts are associated with which blood units.
Fields:
idTA (Foreign Key referencing technical_analyst): Connects to the Technical_Analyst Table.
idBlood (Foreign Key referencing blood): Connects to the Blood Table.
bloodGroup: Blood type of the unit.

Donor Table (donor):

Purpose: Records information about blood donors, including personal details, health information, and the blood donation process.
Fields:
idDonor (Primary Key): Unique identifier for each donor.
nameD: First name of the donor.
surnameD: Last name of the donor.
gender: Gender of the donor.
bloodGroup: Blood type of the donor.
contact: Contact information for the donor.
avenue_street: Street or avenue where the donor resides.
building_number: Building number of the donor's residence.
neighborhood: Neighborhood where the donor resides.
city: City where the donor resides.
age: Age of the donor.
disease: Indicates whether the donor has any health conditions (0 for no, 1 for yes).
idBlood (Foreign Key referencing blood): Connects to the Blood Table.
bloodGroupBlood: Blood type of the donor's donated blood.
idRT (Foreign Key referencing registration_team): Connects to the Registration_Team Table.
dateRegisters: Date when the donor's information was registered.

Out of scope for now but will be worked on in the future:

Introducing tables for managing user accounts, roles, and permissions.
Implementing authentication mechanisms for different types of users (e.g., donors, hospital staff, administrators).

Enhanced the blood inventory management system with more details on blood storage conditions Integrating tracking for blood component levels (plasma, platelets, etc.) if applicable.

Implementing a notification system to alert administrators, managers, or relevant personnel about critical events (e.g., low blood inventory, urgent blood requests, upcoming donor eligibility).

Creating tables and views to support analytical queries and reporting, enabling insights into donation trends, blood usage patterns, and operational efficiency.
Implement stored procedures or functions for generating custom reports.

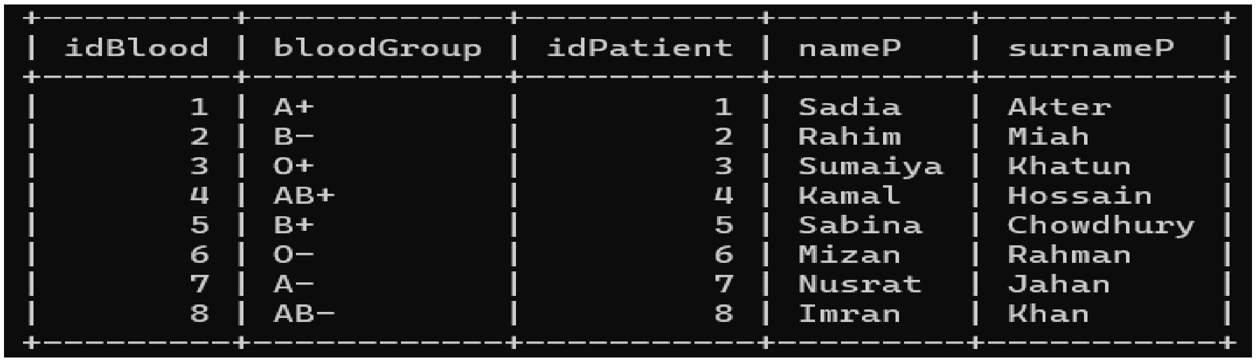
A system spaning multiple locations, consider incorporating geospatial data for mapping blood banks, hospitals, and donor locations.
This can aid in optimizing blood transportation routes and emergency response planning.

Expected Queries

- **Retrieve Blood Units and Associated Patient Information:**

Get details about blood units and the patients who received them.

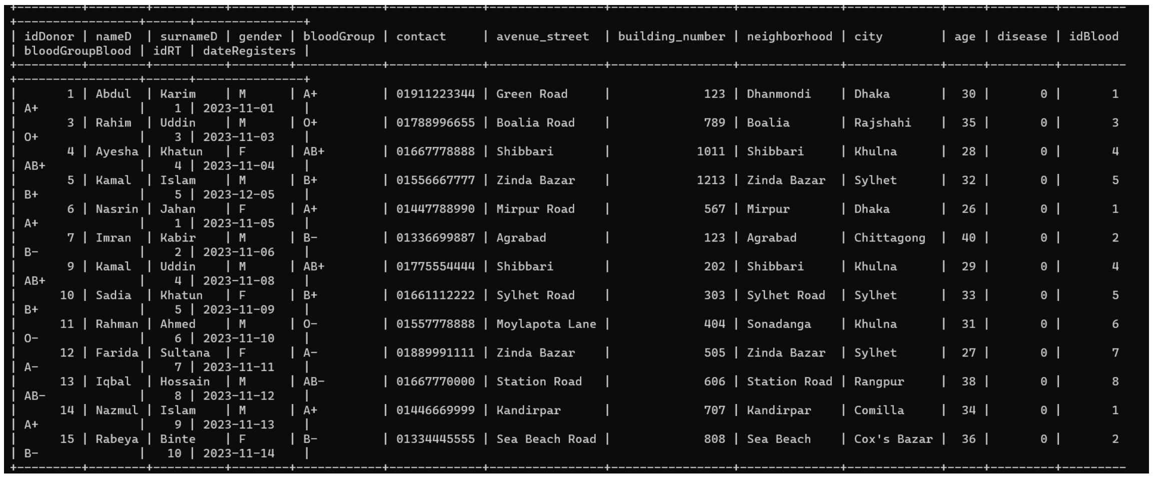
```
  SELECT Blood.idBlood, Blood.bloodGroup, Patient.idPatient, Patient.nameP,
Patient.surnameP
  FROM Blood
  JOIN Blood_Patient ON Blood.idBlood = Blood_Patient.idBlood
  JOIN Patient ON Blood_Patient.idPatient = Patient.idPatient;
```



```
+---------+------------+-----------+---------+-----------+
| idBlood | bloodGroup | idPatient | nameP   | surnameP  |
+---------+------------+-----------+---------+-----------+
|       1 | A+         |         1 | Sadia   | Akter     |
|       2 | B-         |         2 | Rahim   | Miah      |
|       3 | O+         |         3 | Sumaiya | Khatun    |
|       4 | AB+        |         4 | Kamal   | Hossain   |
|       5 | B+         |         5 | Sabina  | Chowdhury |
|       6 | O-         |         6 | Mizan   | Rahman    |
|       7 | A-         |         7 | Nusrat  | Jahan     |
|       8 | AB-        |         8 | Imran   | Khan      |
+---------+------------+-----------+---------+-----------+
```

- **List Blood Donors Eligible for Donation:**

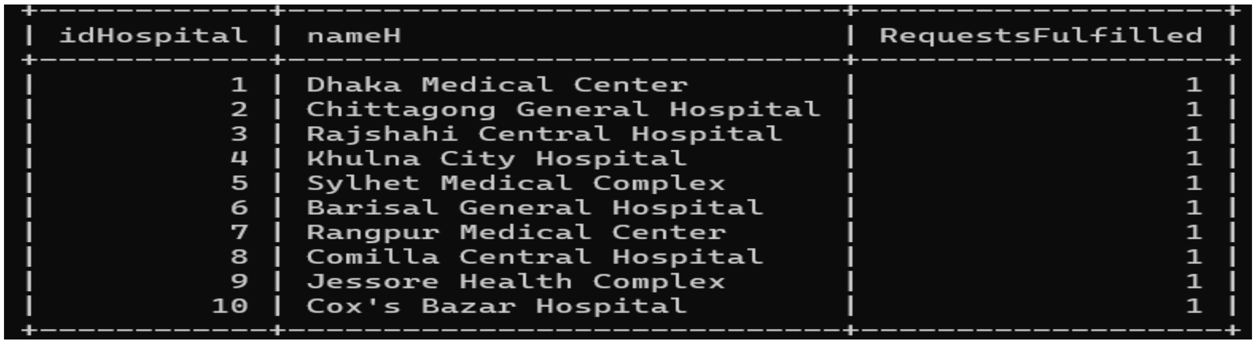
SELECT * FROM Donor
WHERE disease = 0 AND age BETWEEN 18 AND 60;



```
+---------+--------+----------+--------+------------+-------------+----------------+-----------------+--------------+-------------+-----+---------+---------
+-----------------+------+---------------+
| idDonor | nameD  | surnameD | gender | bloodGroup | contact     | avenue_street  | building_number | neighborhood | city        | age | disease | idBlood
| bloodGroupBlood | idRT | dateRegisters |
+---------+--------+----------+--------+------------+-------------+----------------+-----------------+--------------+-------------+-----+---------+---------
+-----------------+------+---------------+
|       1 | Abdul  | Karim    | M      | A+         | 01911223344 | Green Road     |             123 | Dhanmondi    | Dhaka       |  30 |       0 |       1
| A+              |    1 | 2023-11-01    |
|       3 | Rahim  | Uddin    | M      | O+         | 01788996655 | Boalia Road    |             789 | Boalia       | Rajshahi    |  35 |       0 |       3
| O+              |    3 | 2023-11-03    |
|       4 | Ayesha | Khatun   | F      | AB+        | 01667778888 | Shibbari       |            1011 | Shibbari     | Khulna      |  28 |       0 |       4
| AB+             |    4 | 2023-11-04    |
|       5 | Kamal  | Islam    | M      | B+         | 01556667777 | Zinda Bazar    |            1213 | Zinda Bazar  | Sylhet      |  32 |       0 |       5
| B+              |    5 | 2023-12-05    |
|       6 | Nasrin | Jahan    | F      | A+         | 01447788990 | Mirpur Road    |             567 | Mirpur       | Dhaka       |  26 |       0 |       1
| A+              |    1 | 2023-11-05    |
|       7 | Imran  | Kabir    | M      | B-         | 01336699887 | Agrabad        |             123 | Agrabad      | Chittagong  |  40 |       0 |       2
| B-              |    2 | 2023-11-06    |
|       9 | Kamal  | Uddin    | M      | AB+        | 01775554444 | Shibbari       |             202 | Shibbari     | Khulna      |  29 |       0 |       4
| AB+             |    4 | 2023-11-08    |
|      10 | Sadia  | Khatun   | F      | B+         | 01661112222 | Sylhet Road    |             303 | Sylhet Road  | Sylhet      |  33 |       0 |       5
| B+              |    5 | 2023-11-09    |
|      11 | Rahman | Ahmed    | M      | O-         | 01557778888 | Moylapota Lane |             404 | Sonadanga    | Khulna      |  31 |       0 |       6
| O-              |    6 | 2023-11-10    |
|      12 | Farida | Sultana  | F      | A-         | 01889991111 | Zinda Bazar    |             505 | Zinda Bazar  | Sylhet      |  27 |       0 |       7
| A-              |    7 | 2023-11-11    |
|      13 | Iqbal  | Hossain  | M      | AB-        | 01667770000 | Station Road   |             606 | Station Road | Rangpur     |  38 |       0 |       8
| AB-             |    8 | 2023-11-12    |
|      14 | Nazmul | Islam    | M      | A+         | 01446669999 | Kandirpar      |             707 | Kandirpar    | Comilla     |  34 |       0 |       1
| A+              |    9 | 2023-11-13    |
|      15 | Rabeya | Binte    | F      | B-         | 01334445555 | Sea Beach Road |             808 | Sea Beach    | Cox's Bazar |  36 |       0 |       2
| B-              |   10 | 2023-11-14    |
+---------+--------+----------+--------+------------+-------------+----------------+-----------------+--------------+-------------+-----+---------+---------
```

- **Find Blood Requests Fulfilled by Each Hospital:**

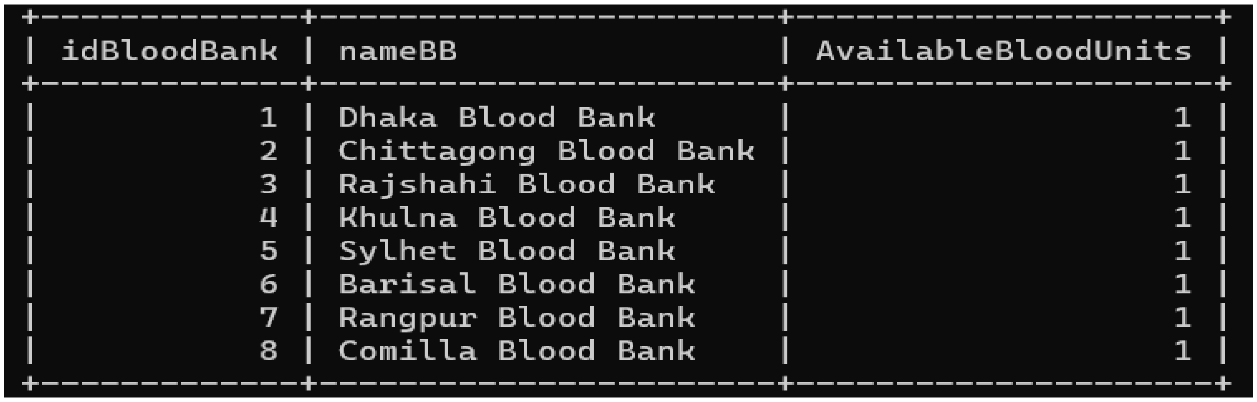
SELECT Hospital.idHospital, Hospital.nameH, COUNT(*) AS RequestsFulfilled
FROM Hospital
JOIN Hospital_Requests ON Hospital.idHospital = Hospital_Requests.idHospital
GROUP BY Hospital.idHospital;



```
+------------+-----------------------------+-------------------+
| idHospital | nameH                       | RequestsFulfilled |
+------------+-----------------------------+-------------------+
|          1 | Dhaka Medical Center        |                 1 |
|          2 | Chittagong General Hospital |                 1 |
|          3 | Rajshahi Central Hospital   |                 1 |
|          4 | Khulna City Hospital        |                 1 |
|          5 | Sylhet Medical Complex      |                 1 |
|          6 | Barisal General Hospital    |                 1 |
|          7 | Rangpur Medical Center      |                 1 |
|          8 | Comilla Central Hospital    |                 1 |
|          9 | Jessore Health Complex      |                 1 |
|         10 | Cox's Bazar Hospital        |                 1 |
+------------+-----------------------------+-------------------+
```

- **Identify Blood Banks with Critical Blood Shortages:**

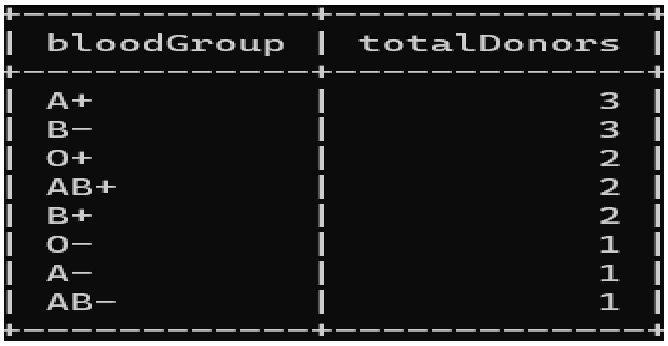
SELECT idBloodBank, nameBB, COUNT(*) AS AvailableBloodUnits
FROM BloodBank
LEFT JOIN Blood ON BloodBank.idBloodBank = Blood.idBlood
GROUP BY BloodBank.idBloodBank
HAVING AvailableBloodUnits < 10;



```
+-------------+-----------------------+---------------------+
| idBloodBank | nameBB                | AvailableBloodUnits |
+-------------+-----------------------+---------------------+
|           1 | Dhaka Blood Bank      |                   1 |
|           2 | Chittagong Blood Bank |                   1 |
|           3 | Rajshahi Blood Bank   |                   1 |
|           4 | Khulna Blood Bank     |                   1 |
|           5 | Sylhet Blood Bank     |                   1 |
|           6 | Barisal Blood Bank    |                   1 |
|           7 | Rangpur Blood Bank    |                   1 |
|           8 | Comilla Blood Bank    |                   1 |
+-------------+-----------------------+---------------------+
```

- **Find the total number of donors for each blood group:**

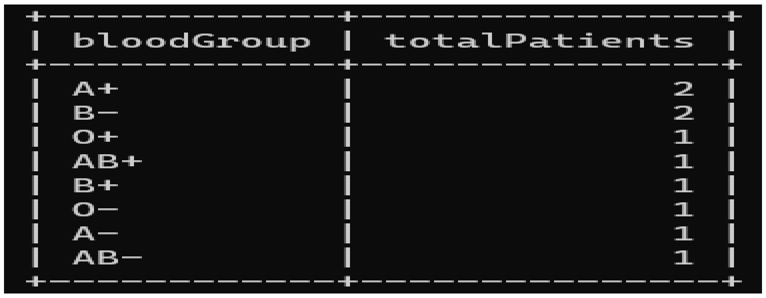
SELECT bloodGroup, COUNT(*) as totalDonors
FROM donor
GROUP BY bloodGroup;



```
+------------+-------------+
| bloodGroup | totalDonors |
+------------+-------------+
| A+         |           3 |
| B-         |           3 |
| O+         |           2 |
| AB+        |           2 |
| B+         |           2 |
| O-         |           1 |
| A-         |           1 |
| AB-        |           1 |
+------------+-------------+
```

- **Find the total number of patients for each blood group:**

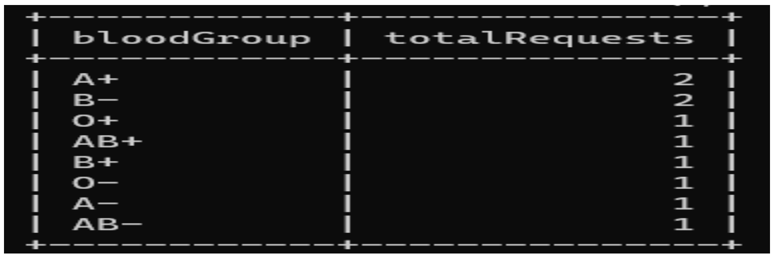
```
SELECT bloodGroup, COUNT(*) as totalPatients
FROM patient
GROUP BY bloodGroup;
```



```
+------------+---------------+
| bloodGroup | totalPatients |
+------------+---------------+
| A+         |             2 |
| B-         |             2 |
| O+         |             1 |
| AB+        |             1 |
| B+         |             1 |
| O-         |             1 |
| A-         |             1 |
| AB-        |             1 |
+------------+---------------+
```

- **Find the total number of requests for each blood group:**

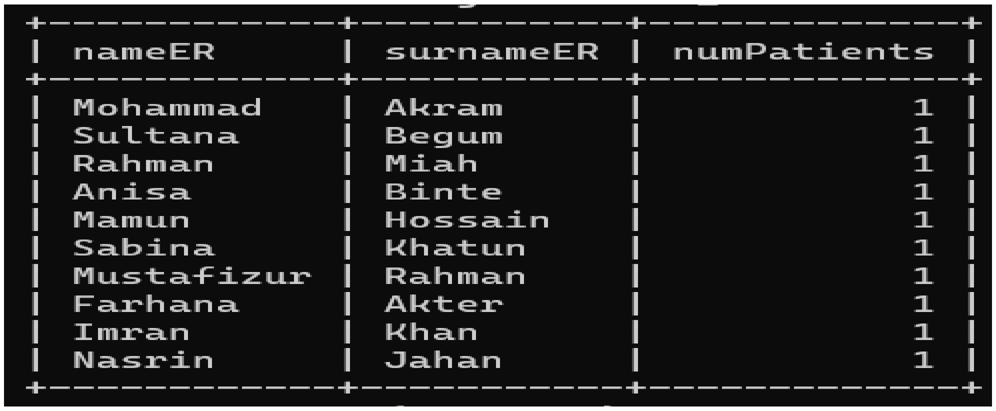
```
SELECT bloodGroup, COUNT(*) as totalRequests
FROM requests
GROUP BY bloodGroup;
```



```
+------------+---------------+
| bloodGroup | totalRequests |
+------------+---------------+
| A+         |             2 |
| B-         |             2 |
| O+         |             1 |
| AB+        |             1 |
| B+         |             1 |
| O-         |             1 |
| A-         |             1 |
| AB-        |             1 |
+------------+---------------+
```

- **Find the number of patients registered by each registration team.**

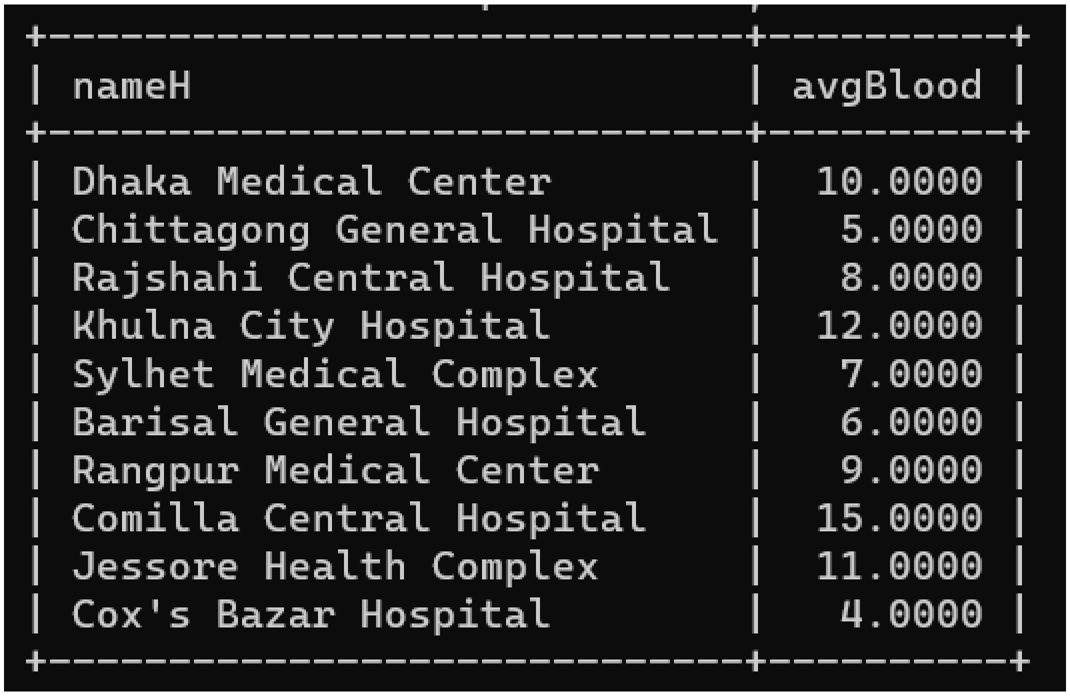
```
SELECT nameER, surnameER, COUNT(*) AS numPatients
FROM patient
JOIN registration_team ON patient.idRT = registration_team.idRT
GROUP BY registration_team.nameER, registration_team.surnameER;
```



```
+------------+-----------+-------------+
| nameER     | surnameER | numPatients |
+------------+-----------+-------------+
| Mohammad   | Akram     |           1 |
| Sultana    | Begum     |           1 |
| Rahman     | Miah      |           1 |
| Anisa      | Binte     |           1 |
| Mamun      | Hossain   |           1 |
| Sabina     | Khatun    |           1 |
| Mustafizur | Rahman    |           1 |
| Farhana    | Akter     |           1 |
| Imran      | Khan      |           1 |
| Nasrin     | Jahan     |           1 |
+------------+-----------+-------------+
```

- **Find the max amount of blood requested by each hospital.**

SELECT nameH, AVG(amountBlood) AS avgBlood
FROM hospital
JOIN hospital_requests ON hospital.idHospital = hospital_requests.idHospital
JOIN requests ON hospital_requests.idRequests = requests.idRequests
GROUP BY hospital.nameH;

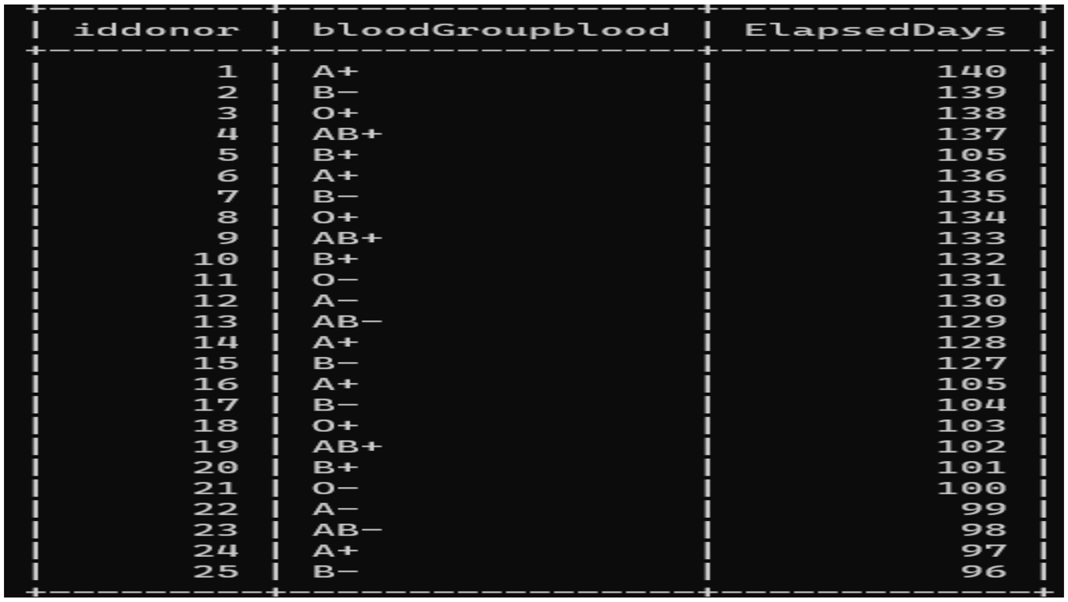


```
+----------------------------+----------+
| nameH                      | avgBlood |
+----------------------------+----------+
| Dhaka Medical Center       |  10.0000 |
| Chittagong General Hospital|   5.0000 |
| Rajshahi Central Hospital  |   8.0000 |
| Khulna City Hospital       |  12.0000 |
| Sylhet Medical Complex     |   7.0000 |
| Barisal General Hospital   |   6.0000 |
| Rangpur Medical Center     |   9.0000 |
| Comilla Central Hospital   |  15.0000 |
| Jessore Health Complex     |  11.0000 |
| Cox's Bazar Hospital       |   4.0000 |
+----------------------------+----------+
```

- **find relapsed time of all donated blood**

SELECT
    iddonor,bloodGroupblood,

```
    (DATEDIFF(CURDATE(), dateRegisters)) AS ElapsedDays
FROM
    Donor;
```

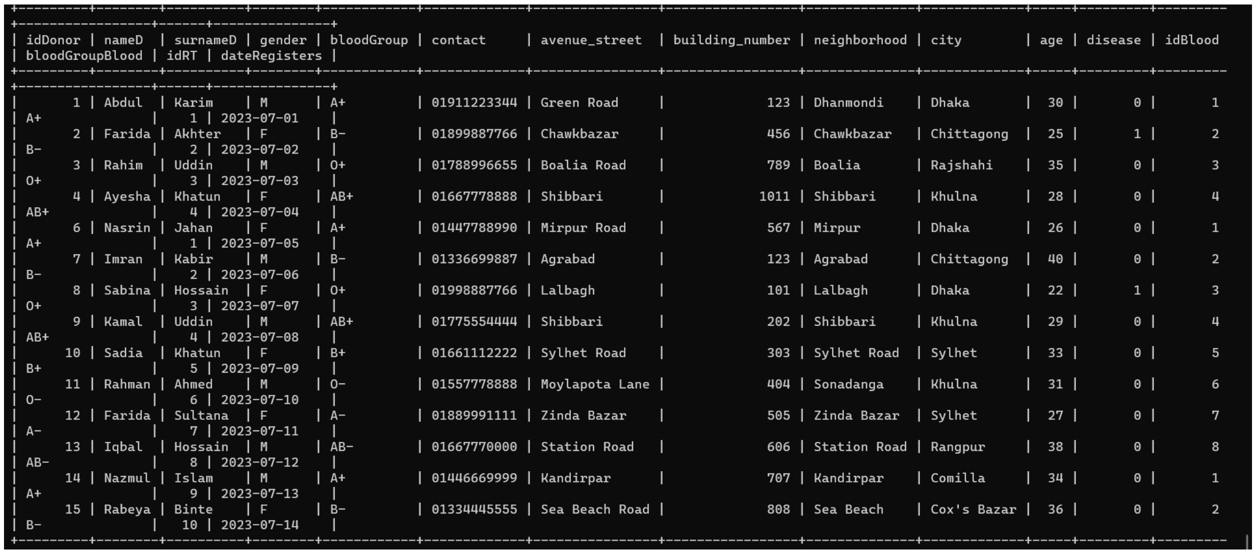


```
+---------+-----------------+-------------+
| iddonor | bloodGroupblood | ElapsedDays |
+---------+-----------------+-------------+
|       1 | A+              |         140 |
|       2 | B-              |         139 |
|       3 | O+              |         138 |
|       4 | AB+             |         137 |
|       5 | B+              |         105 |
|       6 | A+              |         136 |
|       7 | B-              |         135 |
|       8 | O+              |         134 |
|       9 | AB+             |         133 |
|      10 | B+              |         132 |
|      11 | O-              |         131 |
|      12 | A-              |         130 |
|      13 | AB-             |         129 |
|      14 | A+              |         128 |
|      15 | B-              |         127 |
|      16 | A+              |         105 |
|      17 | B-              |         104 |
|      18 | O+              |         103 |
|      19 | AB+             |         102 |
|      20 | B+              |         101 |
|      21 | O-              |         100 |
|      22 | A-              |          99 |
|      23 | AB-             |          98 |
|      24 | A+              |          97 |
|      25 | B-              |          96 |
+---------+-----------------+-------------+
```

- **find all expired donated blood**

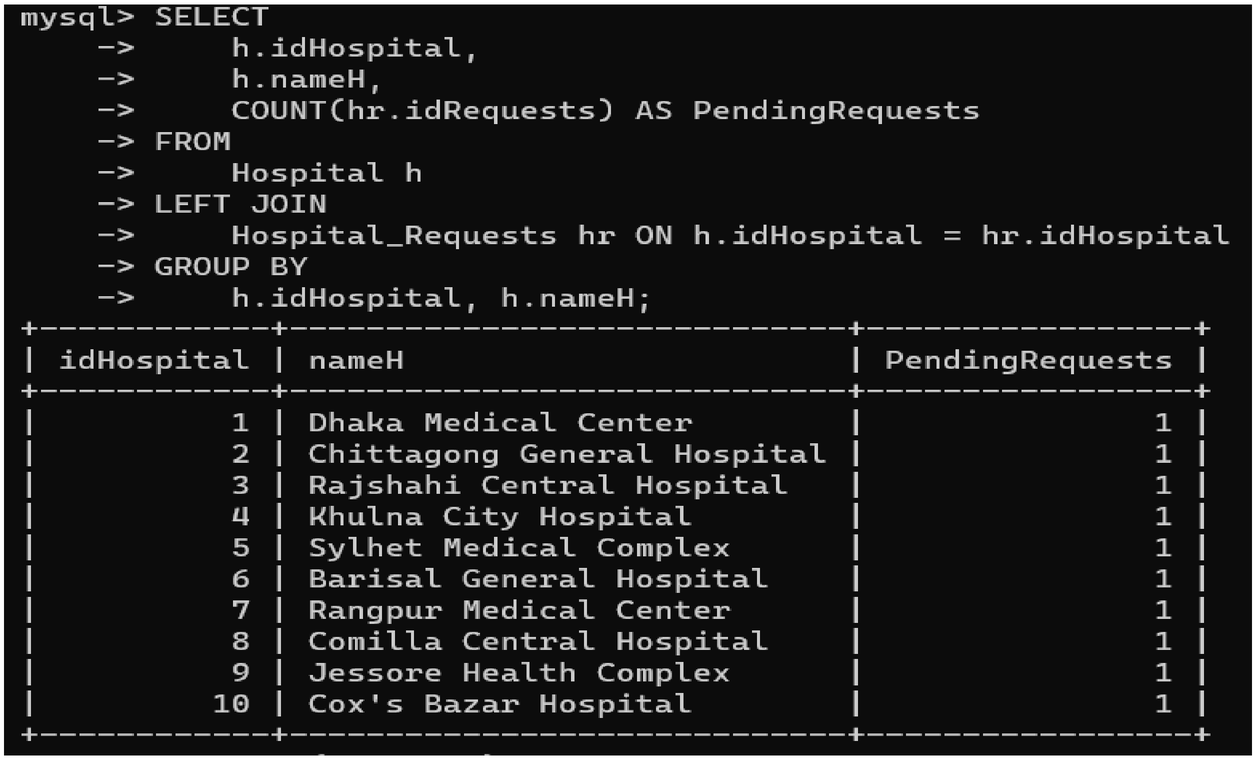
  SELECT * FROM donor

  WHERE DATEDIFF(CURDATE(), dateRegisters) >= 120;



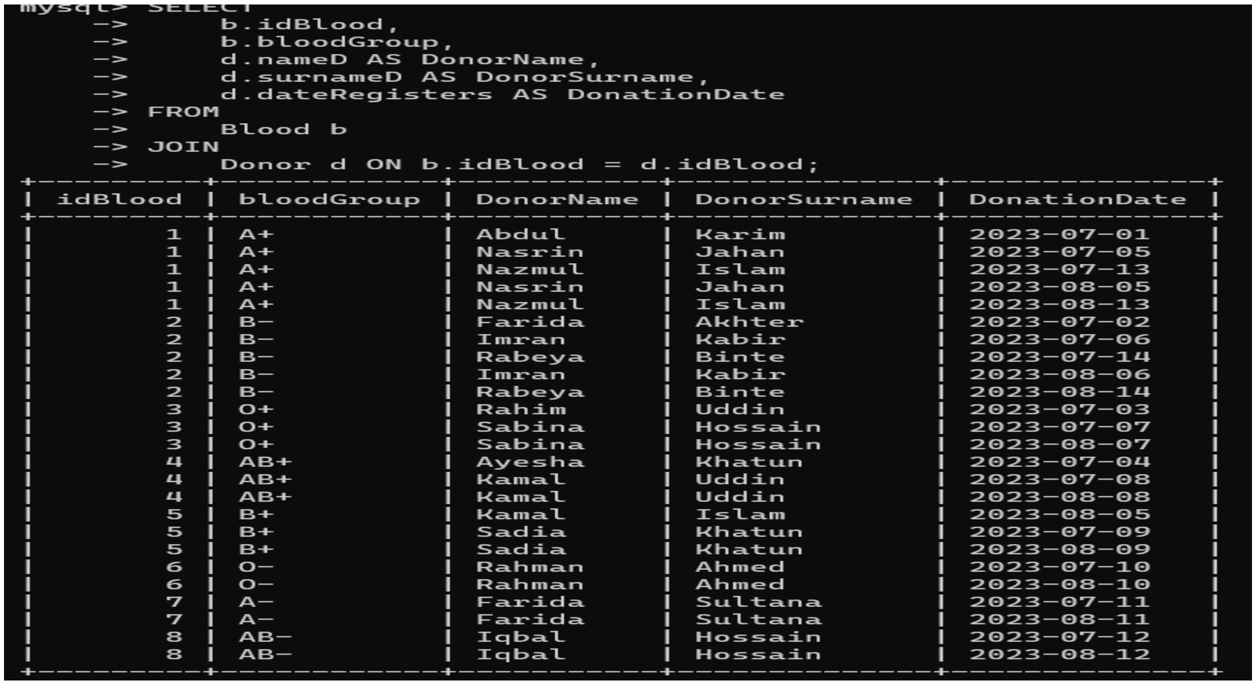
```
+---------+--------+----------+--------+------------+-------------+----------------+-----------------+--------------+-------------+-----+---------+---------
+-----------------+------+---------------+
| idDonor | nameD  | surnameD | gender | bloodGroup | contact     | avenue_street  | building_number | neighborhood | city        | age | disease | idBlood
| bloodGroupBlood | idRT | dateRegisters |
+---------+--------+----------+--------+------------+-------------+----------------+-----------------+--------------+-------------+-----+---------+---------
+-----------------+------+---------------+
|       1 | Abdul  | Karim    | M      | A+         | 01911223344 | Green Road     |             123 | Dhanmondi    | Dhaka       |  30 |       0 |       1
| A+              |    1 | 2023-07-01    |
|       2 | Farida | Akhter   | F      | B-         | 01899887766 | Chawkbazar     |             456 | Chawkbazar   | Chittagong  |  25 |       1 |       2
| B-              |    2 | 2023-07-02    |
|       3 | Rahim  | Uddin    | M      | O+         | 01788996655 | Boalia Road    |             789 | Boalia       | Rajshahi    |  35 |       0 |       3
| O+              |    3 | 2023-07-03    |
|       4 | Ayesha | Khatun   | F      | AB+        | 01667778888 | Shibbari       |            1011 | Shibbari     | Khulna      |  28 |       0 |       4
| AB+             |    4 | 2023-07-04    |
|       6 | Nasrin | Jahan    | F      | A+         | 01447788990 | Mirpur Road    |             567 | Mirpur       | Dhaka       |  26 |       0 |       1
| A+              |    1 | 2023-07-05    |
|       7 | Imran  | Kabir    | M      | B-         | 01336699887 | Agrabad        |             123 | Agrabad      | Chittagong  |  40 |       0 |       2
| B-              |    2 | 2023-07-06    |
|       8 | Sabina | Hossain  | F      | O+         | 01998887766 | Lalbagh        |             101 | Lalbagh      | Dhaka       |  22 |       1 |       3
| O+              |    3 | 2023-07-07    |
|       9 | Kamal  | Uddin    | M      | AB+        | 01775554444 | Shibbari       |             202 | Shibbari     | Khulna      |  29 |       0 |       4
| AB+             |    4 | 2023-07-08    |
|      10 | Sadia  | Khatun   | F      | B+         | 01661112222 | Sylhet Road    |             303 | Sylhet Road  | Sylhet      |  33 |       0 |       5
| B+              |    5 | 2023-07-09    |
|      11 | Rahman | Ahmed    | M      | O-         | 01557778888 | Moylapota Lane |             404 | Sonadanga    | Khulna      |  31 |       0 |       6
| O-              |    6 | 2023-07-10    |
|      12 | Farida | Sultana  | F      | A-         | 01889991111 | Zinda Bazar    |             505 | Zinda Bazar  | Sylhet      |  27 |       0 |       7
| A-              |    7 | 2023-07-11    |
|      13 | Iqbal  | Hossain  | M      | AB-        | 01667770000 | Station Road   |             606 | Station Road | Rangpur     |  38 |       0 |       8
| AB-             |    8 | 2023-07-12    |
|      14 | Nazmul | Islam    | M      | A+         | 01446669999 | Kandirpar      |             707 | Kandirpar    | Comilla     |  34 |       0 |       1
| A+              |    9 | 2023-07-13    |
|      15 | Rabeya | Binte    | F      | B-         | 01334445555 | Sea Beach Road |             808 | Sea Beach    | Cox's Bazar |  36 |       0 |       2
| B-              |   10 | 2023-07-14    |
+---------+--------+----------+--------+------------+-------------+----------------+-----------------+--------------+-------------+-----+---------+---------
```

- **Find Hospitals and Their Pending Blood Requests:**



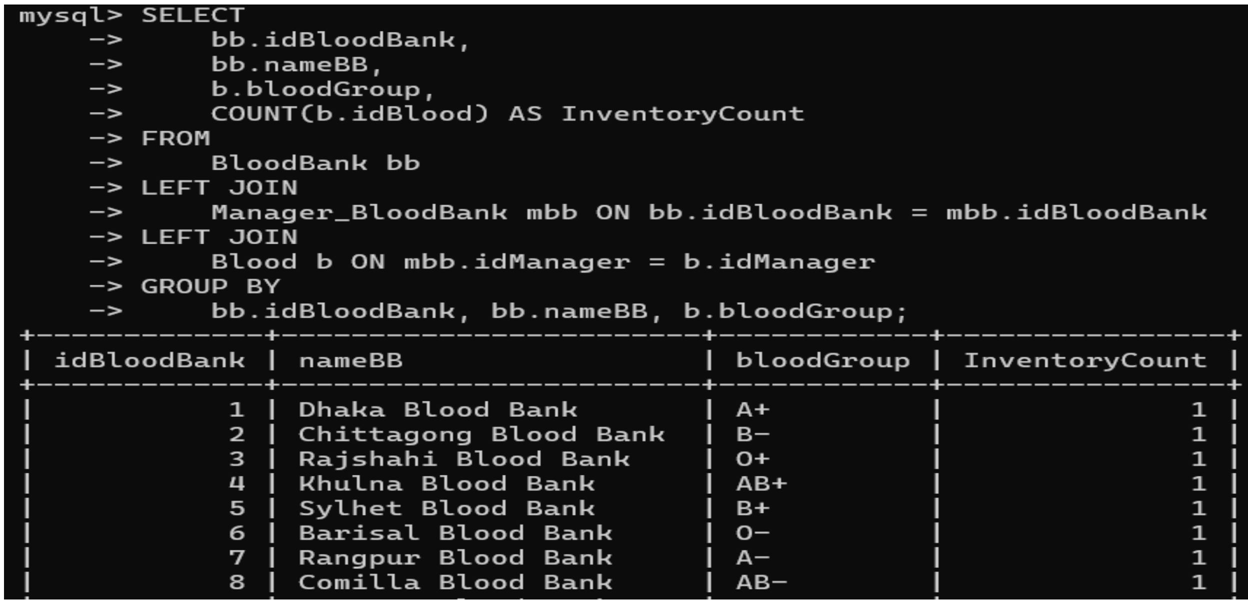
```
mysql> SELECT
    ->     h.idHospital,
    ->     h.nameH,
    ->     COUNT(hr.idRequests) AS PendingRequests
    -> FROM
    ->     Hospital h
    -> LEFT JOIN
    ->     Hospital_Requests hr ON h.idHospital = hr.idHospital
    -> GROUP BY
    ->     h.idHospital, h.nameH;
+------------+-----------------------------+-----------------+
| idHospital | nameH                       | PendingRequests |
+------------+-----------------------------+-----------------+
|          1 | Dhaka Medical Center        |               1 |
|          2 | Chittagong General Hospital |               1 |
|          3 | Rajshahi Central Hospital   |               1 |
|          4 | Khulna City Hospital        |               1 |
|          5 | Sylhet Medical Complex      |               1 |
|          6 | Barisal General Hospital    |               1 |
|          7 | Rangpur Medical Center      |               1 |
|          8 | Comilla Central Hospital    |               1 |
|          9 | Jessore Health Complex      |               1 |
|         10 | Cox's Bazar Hospital        |               1 |
+------------+-----------------------------+-----------------+
```

- **Retrive blood with donation details**



```
mysql> SELECT
    ->     b.idBlood,
    ->     b.bloodGroup,
    ->     d.nameD AS DonorName,
    ->     d.surnameD AS DonorSurname,
    ->     d.dateRegisters AS DonationDate
    -> FROM
    ->     Blood b
    -> JOIN
    ->     Donor d ON b.idBlood = d.idBlood;
+---------+------------+-----------+--------------+--------------+
| idBlood | bloodGroup | DonorName | DonorSurname | DonationDate |
+---------+------------+-----------+--------------+--------------+
|       1 | A+         | Abdul     | Karim        | 2023-07-01   |
|       1 | A+         | Nasrin    | Jahan        | 2023-07-05   |
|       1 | A+         | Nazmul    | Islam        | 2023-07-13   |
|       1 | A+         | Nasrin    | Jahan        | 2023-08-05   |
|       1 | A+         | Nazmul    | Islam        | 2023-08-13   |
|       2 | B-         | Farida    | Akhter       | 2023-07-02   |
|       2 | B-         | Imran     | Kabir        | 2023-07-06   |
|       2 | B-         | Rabeya    | Binte        | 2023-07-14   |
|       2 | B-         | Imran     | Kabir        | 2023-08-06   |
|       2 | B-         | Rabeya    | Binte        | 2023-08-14   |
|       3 | O+         | Rahim     | Uddin        | 2023-07-03   |
|       3 | O+         | Sabina    | Hossain      | 2023-07-07   |
|       3 | O+         | Sabina    | Hossain      | 2023-08-07   |
|       4 | AB+        | Ayesha    | Khatun       | 2023-07-04   |
|       4 | AB+        | Kamal     | Uddin        | 2023-07-08   |
|       4 | AB+        | Kamal     | Uddin        | 2023-08-08   |
|       5 | B+         | Kamal     | Islam        | 2023-08-05   |
|       5 | B+         | Sadia     | Khatun       | 2023-07-09   |
|       5 | B+         | Sadia     | Khatun       | 2023-08-09   |
|       6 | O-         | Rahman    | Ahmed        | 2023-07-10   |
|       6 | O-         | Rahman    | Ahmed        | 2023-08-10   |
|       7 | A-         | Farida    | Sultana      | 2023-07-11   |
|       7 | A-         | Farida    | Sultana      | 2023-08-11   |
|       8 | AB-        | Iqbal     | Hossain      | 2023-07-12   |
|       8 | AB-        | Iqbal     | Hossain      | 2023-08-12   |
+---------+------------+-----------+--------------+--------------+
```

- **Get info on current inventory**



```
mysql> SELECT
    ->     bb.idBloodBank,
    ->     bb.nameBB,
    ->     b.bloodGroup,
    ->     COUNT(b.idBlood) AS InventoryCount
    -> FROM
    ->     BloodBank bb
    -> LEFT JOIN
    ->     Manager_BloodBank mbb ON bb.idBloodBank = mbb.idBloodBank
    -> LEFT JOIN
    ->     Blood b ON mbb.idManager = b.idManager
    -> GROUP BY
    ->     bb.idBloodBank, bb.nameBB, b.bloodGroup;
+-------------+-------------------------+------------+----------------+
| idBloodBank | nameBB                  | bloodGroup | InventoryCount |
+-------------+-------------------------+------------+----------------+
|           1 | Dhaka Blood Bank        | A+         |              1 |
|           2 | Chittagong Blood Bank   | B-         |              1 |
|           3 | Rajshahi Blood Bank     | O+         |              1 |
|           4 | Khulna Blood Bank       | AB+        |              1 |
|           5 | Sylhet Blood Bank       | B+         |              1 |
|           6 | Barisal Blood Bank      | O-         |              1 |
|           7 | Rangpur Blood Bank      | A-         |              1 |
|           8 | Comilla Blood Bank      | AB-        |              1 |
```

List of tables with schema:

1. Donor = (idDonor, nameD, surnameD, gender, bloodGroup, contact, avenue_street, building_number, neighborhood, city, age, disease, idBlood, bloodGroupBlood, idRT, dateRegisters)

2. Blood =(idBlood, bloodGroup, idManager)

3. Patient=(idPatient, nameP, surnameP, gender, bloodGroup, contact, avenue_street, building_number, neighborhood, city, dateRegisters, idRT, idManager)

4. Requests=(idRequests, bloodGroup, amountBlood)

5. Hospital=(idHospital, nameH, avenue_street, building_number, neighborhood, city, idManager)

6. bloodbank=(idBloodBank, nameBB, avenue_street, building_number, neighborhood, city)

7. Registration team=(idRT, nameER, surnameER, idBloodBank)

8. Technical analyst=(idTA, nameTA, surnameTA, idBloodBank)

9. Manager = (idManager, nameM, surnameM)

10. manager_bloodbank=(idManager, idBloodBank)

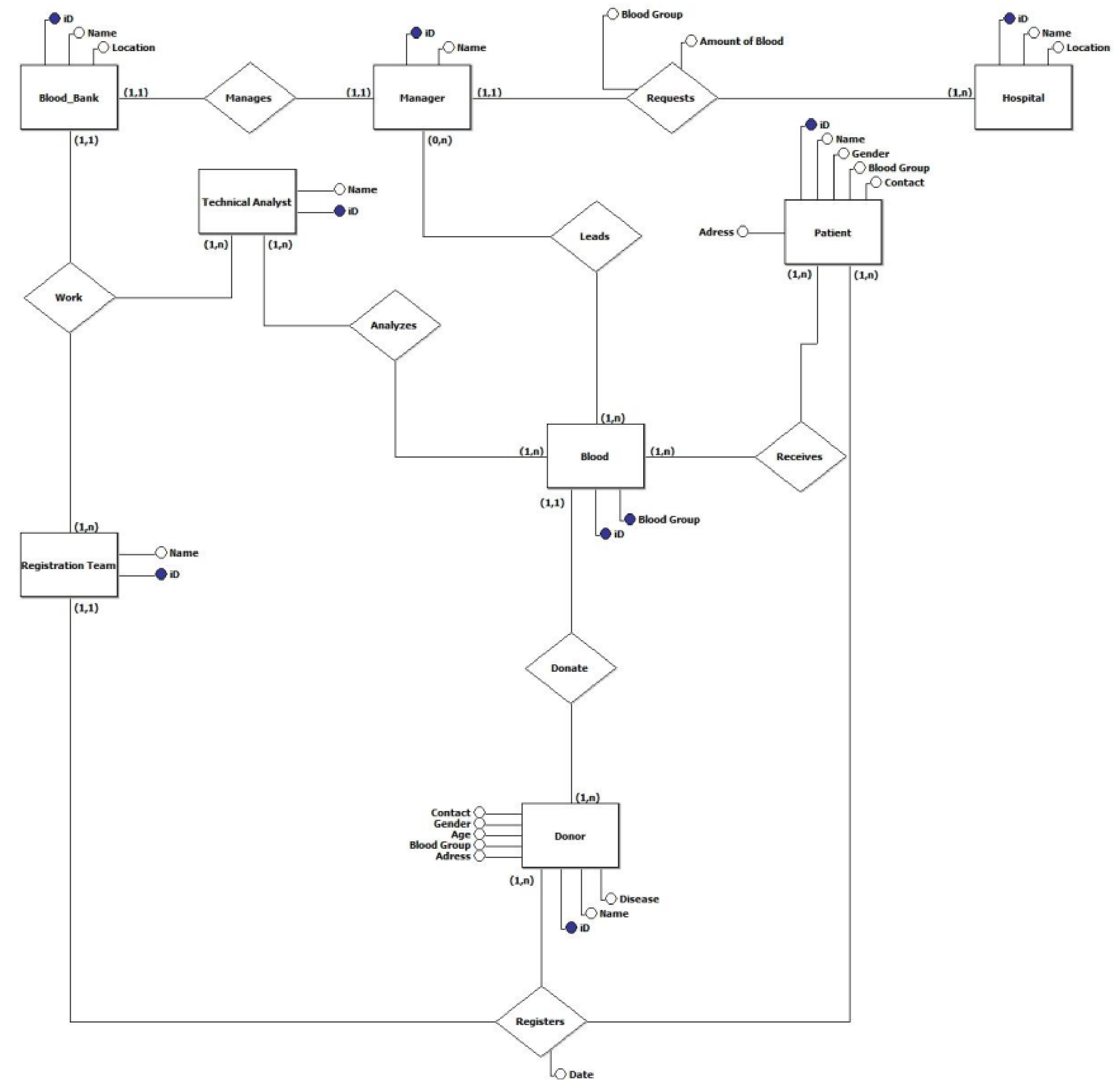
11. patient_requests=(idPatient, idRequests)

12. hospital_requests=(idHospital, idRequests)

13. blood_patient= (idBlood,bloodGroup,idPatient)

14. technicalAnalyst_blood=(idTA, idBlood, bloodGroup)

ER DIAGRAM



Blood_Bank
iD
Name
Location
Manages
Manager
Name
Requests
Blood Group
Amount of Blood
Hospital
Location
Patient
Gender
Contact
Adress
Technical Analyst
Work
Analyzes
Leads
Blood
Receives
Registration Team
Donate
Donor
Age
Disease
Registers
Date
(1,1)
(1,n)
(0,n)

List of functional dependencies to be maintained:

1. Donor (idDonor) -> (nameD, surnameD, gender, bloodGroup, contact, avenue_street, building_number, neighborhood, city, age, disease, bloodGroupBlood, dateRegisters)

   - Donor ID uniquely identifies all other donor details

2. Blood (idBlood, bloodGroup) -> idManager

   - Blood ID and group determines the managing employee

3. Patient (idPatient) -> (nameP, surnameP, gender, contact, avenue_street, building_number, neighborhood, city, dateRegisters)

   - Patient ID functionally determines their personal details

4. Requests (idRequests) -> bloodGroup, amountBlood

   - Request ID uniquely decides blood group and quantity requested

5. Hospital (idHospital) -> (nameH, avenue_street, building_number, neighborhood, city)

   - Hospital ID determines hospital details

6. Blood bank (idBloodBank) -> (nameBB, avenue_street, building_number, neighborhood, city)

   - Blood bank ID determines blood bank details

7. Registration team (idRT) -> (nameER, surnameER)

   - Registration team member ID determines their name

8. Technical analyst (idTA) -> (nameTA, surnameTA)

   - Technical analyst ID determines their name

9. Manager (idManager) -> (nameM, surnameM)

   - Manager ID functionally identifies name details

The keys in each FD are candidate keys that uniquely identify the non-key attributes.

Table schema

1. **Donor**

S/N	Attribute	Datatype	Constraint	Comment
1	idDonor	INT	PK, NOT NULL	Unique donor id
2	nameD	VARCHAR(30)	NOT NULL	Donor first name
3	surnameD	VARCHAR(30)	NOT NULL	Donor last name
4	gender	ENUM	NOT NULL	Gender values M, F, N
5	bloodGroup	VARCHAR(3)	NOT NULL	Blood group of donor
6	contact	VARCHAR(20)	NOT NULL	Contact phone/mobile number
7	avenue_street	VARCHAR(30)	NOT NULL	Address avenue/street
8	building_number	INT	NOT NULL	Address building number
9	neighborhood	VARCHAR(30)	NOT NULL	Neighborhood of address
10	city	VARCHAR(30)		City of address
11	age	INT	NOT NULL	Age in years
12	disease	INT	DEFAULT 0	1 if donor has disease, 0 otherwise
13	idBlood	INT	NOT NULL	Blood sample id donated
14	bloodGroupBlood	VARCHAR(3)	NOT NULL	Blood group of donated blood

15	idRT	INT	NOT NULL	Registering team member id
16	dateRegisters	DATE	NOT NULL	Donor registration date

Purpose: Records information about blood donors, including personal details, health information, and the blood donation process.
Information stored: Stores donor personal details like name, contact info, eligibility criteria
Captures donation instances with blood group donated and registration staff
Used to manage donor drives, contact donors, enforce eligibility checks before collecting donations

2. **Blood**

S/N	Attribute	Datatype	Constraint	Comment
1	idBlood	INT	PK, NOT NULL	Unique blood sample id
2	bloodGroup	VARCHAR(3)	PK, NOT NULL	Blood group categorization
3	idManager	INT	NOT NULL	Managing employee id

Purpose: Manages information about individual blood units, including the blood type and the manager responsible for the blood bank.
Information stored: Records blood units collected from donors with the unique id and blood group
Links to the managing employee overseeing this blood sample
Used to track blood inventory, match to patient needs, manage shelf-life before expiry

3. **Patient**

S/N	Attribute	Datatype	Constraint	Comment
1	idPatient	INT	PK, NOT NULL	Unique patient id
2	nameP	VARCHAR(30)	NOT NULL	First name
3	surnameP	VARCHAR(30)	NOT NULL	Last name

4	gender	ENUM	NOT NULL	Gender values M, F, N
5	bloodGroup	VARCHAR(30)	NOT NULL	Patient's blood group
6	contact	VARCHAR(20)	NOT NULL	Contact phone/mobile number
7	avenue_street	VARCHAR(30)	NOT NULL	Address avenue/street
8	building_number	INT	NOT NULL	Address building number
9	neighborhood	VARCHAR(30)	NOT NULL	Neighborhood of address
10	city	VARCHAR(30)	NOT NULL	City of address
11	dateRegisters	DATE	NOT NULL	Date when registered
12	idRT	INT	NOT NULL	Registering team member id
13	idManager	INT	NOT NULL	Managing employee id

Purpose: Records details about patients, including their contact information, address, and the registration team and manager overseeing their records.
Information stored: Captures patient personal and contact details requiring blood
Links to registration staff member and managing employee
Used to identify patients, contact them, match to suitable blood groups in inventory

4. **requests**

S/N	Attribute	Datatype	Constraint	Comment
1	idRequests	INT	PK, NOT NULL	Unique request id
2	bloodGroup	VARCHAR(3)	NOT NULL	Requested blood group
3	amountBlood	INT	NOT NULL	Units of blood

				requested

Purpose: Manages requests for blood, specifying the blood type and the required quantity.
Information stored: Logs every blood request with type and quantity needed
Can be linked to a patient or hospital
Used to track overall demand, analyze shortages, manage priority requests

5. **hospital**

S/N	Attribute	Datatype	Constraint	Comment
1	idHospital	INT	PK, NOT NULL	Unique hospital id
2	nameH	VARCHAR(30)	NOT NULL	Hospital name
3	avenue_street	VARCHAR(30)	NOT NULL	Address avenue/street
4	building_number	INT	NOT NULL	Address building number
5	neighborhood	VARCHAR(30)	NOT NULL	Neighborhood of address
6	city	VARCHAR(30)	NOT NULL	City of address
7	idManager	INT	NOT NULL	Managing employee id

Purpose: Stores information about hospitals, including their location and the manager responsible for the hospital.
Information stored: Stores hospital address, contact details and managing employee
Used to coordinate orders and blood unit deliveries to service requesting hospitals

6. **Blood bank**

S/N	Attribute	Datatype	Constraint	Comment
1	idBloodBank	INT	PK, NOT NULL	Unique blood bank id
2	nameBB	VARCHAR(30)	NOT NULL	Blood bank name

3	avenue_street	VARCHAR(30)	NOT NULL	Address avenue/street
4	building_number	INT	NOT NULL	Address building number
5	neighborhood	VARCHAR(30)	NOT NULL	Neighborhood of address
6	city	VARCHAR(30)	NOT NULL	City of address

Purpose: This table contains details about individual blood banks, including their location.
Information stored: Stores address and contact details for each blood bank branch
Used to manage operations across the network of bank locations

7. **manager**

S/N	Attribute	Datatype	Constraint	Comment
1	idManager	INT	PK, NOT NULL	Unique manager id
2	nameM	VARCHAR(30)	NOT NULL	First name
3	surnameM	VARCHAR(30)	NOT NULL	Last name

Purpose: This table stores information about the managers responsible for overseeing the operations of blood banks.
Information stored: Maintains employee details of managers
Managers oversee different operations and locations
Used to link accountable personnel to blood units, hospitals etc.

8. **registration_team**

S/N	Attribute	Datatype	Constraint	Comment
1	idRT	INT	PK, NOT NULL	Unique registration team id
2	nameER	VARCHAR(30)	NOT NULL	First name
3	surnameER	VARCHAR(30)	NOT NULL	Last name
4	idBloodBank	INT	NOT NULL	Associated blood bank id

Purpose: Records information about registration teams involved in the blood donation process, such as those responsible for donor registration.
Information stored: Captures staff members responsible for donor/patient registrations
Links staff members to their blood bank branch
Used to track registration operations to branches

9. **Technical analyst**

S/N	Attribute	Datatype	Constraint	Comment
1	idTA	INT	PK, NOT NULL	Unique technical analyst id
2	nameTA	VARCHAR(30)	NOT NULL	First name
3	surnameTA	VARCHAR(30)	NOT NULL	Last name
4	idBloodBank	INT	NOT NULL	Associated blood bank id

Purpose: This table establishes a many-to-many relationship between managers and blood banks, indicating which managers are responsible for which blood banks.
Information stored: Stores technicians testing blood samples at each bank branch
Used to link testing personnel to bank locations
Ensures tracking of testing by technical staff

10. **manager_bloodbank**

S/N	Attribute	Datatype	Constraint	Comment
1	idManager	INT	PK, NOT NULL	Manager id
2	idBloodBank	INT	PK, NOT NULL	Blood bank id

Purpose: This table establishes a many-to-many relationship between managers and blood banks, indicating which managers are responsible for which blood banks.

Information stored: Associates managers to multiple blood bank branches
Used to identify accountability for each bank

11. patient_requests

S/N	Attribute	Datatype	Constraint	Comment
1	idPatient	INT	PK, NOT NULL	Patient id
2	idRequests	INT	PK, NOT NULL	Request id

Purpose: Establishes a many-to-many relationship between patients and blood requests, indicating which patients have requested which types and quantities of blood.
Information stored: Links patients to their requested blood orders
Used to track patient-specific requests in the system

12. hospital_requests

S/N	Attribute	Datatype	Constraint	Comment
1	idHospital	INT	PK, NOT NULL	Hospital id
2	idRequests	INT	PK, NOT NULL	Request id

Purpose: Establishes a many-to-many relationship between hospitals and blood requests, indicating which hospitals have requested which types and quantities of blood.
Information stored: Links hospitals with their blood orders
Used to track blood requests tied to each hospital

13. blood_patient

S/N	Attribute	Datatype	Constraint	Comment
1	idBlood	INT	PK, NOT NULL	Blood unit id
2	bloodGroup	VARCHAR(3)	PK, NOT NULL	Blood group
3	idPatient	INT	PK, NOT NULL	Patient id

Purpose: Manages the relationship between blood units and patients, indicating which patients have received which blood units.
Information stored: Associates donated blood units to patients
Captures which patient received which donated unit
Used to record complete cycle from blood donation to transfusion into patients

14. technical_analyst_blood

S/N	Attribute	Datatype	Constraint	Comment
1	idTA	INT	PK, NOT NULL	Technical analyst id
2	idBlood	INT	PK, NOT NULL	Blood unit id
3	bloodGroup	VARCHAR(3)	PK, NOT NULL	Blood group

Purpose: Establishes a many-to-many relationship between technical analysts and blood units, indicating which technical analysts are associated with which blood units.
Information stored: Links technicians with the blood units they tested
Tracks the personnel who certified each donated unit is safe for use
Ensures traceability from the analytical testing phase

Table-level and project level expected constraints

Donor
- CHECK constraint on age 18-60 years
- NOT NULL constraints on essential fields like name, blood group etc.
- UNIQUE key on contact number
- FOREIGN KEY links to blood donated, registration team etc.

Blood
- NOT NULL constraints on id, group
- FOREIGN KEY to link manager overseeing this blood

Patient
- NOT NULL constraints on name, contact fields
- FOREIGN KEYs linking registration staff and manager

Hospital
- NOT NULL constraints on name and address
- FOREIGN KEY linking managing employee

Registration Team
- NOT NULL constraints on names and blood bank foreign key

Blood Bank
- NOT NULL constraints on name and address fields

Manager

- NOT NULL constraint on name
- UNIQUE constraint on manager id

Requests
- NOT NULL constraints on blood group and quantity

Technical Analyst
- NOT NULL constraints on names and blood bank key
- UNIQUE constraint on analyst id

Association Tables
- PRIMARY KEY constraints on composite keys
- FOREIGN KEY constraints linking to parent tables

Project-Level
- Entity integrity constraints on tables via PK-FK relationships
- Referential integrity across donor-blood-patient cycle data
- Domain integrity checks on invalid data values
- Column integrity via NOT NULL, CHECK etc.

Used triggers:

before_insert_donor
Purpose:
The trigger was created on the donor table to enforce a rule that donors can donate blood only once every 30 days.

Implementation:

The trigger fires before every INSERT operation on the donor table (before a new donor record is inserted).

Inside the trigger, it first fetches the last donation date for that donor from the table.

Next, it checks if the difference between the new donation date (being inserted) and previous date is less than 30 days.

If the difference is less than 30 days, the trigger raises an application error with a custom message that the minimum interval between donations is not met.

So effectively, it prevents a new donation registration for a donor within 30 days of their last donation.

Benefits:

- Enforces domain rules directly in database

- Avoid invalid data being stored
- Centralized control through database itself

```
DELIMITER //

CREATE TRIGGER before_insert_donor
BEFORE INSERT ON donor
FOR EACH ROW
BEGIN
    DECLARE last_donation_date DATE;

    -- Find the last donation date for the current donor
    SELECT MAX(dateRegisters) INTO last_donation_date
    FROM donor
    WHERE idDonor = NEW.idDonor;

    -- Check if the donation interval is at least 30 days
    IF last_donation_date IS NOT NULL AND DATEDIFF(NEW.dateRegisters, last_donation_date) <
30 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Donor can only donate once every 30 days';
    END IF;
END //

DELIMITER ;
```

SQLs to implement the project with example outputs

Table creation:

```
CREATE TABLE manager(
idManager INT  NOT NULL ,
nameM varchar(30) NOT NULL,
surnameM varchar(30) NOT NULL,
CONSTRAINT PK_manager PRIMARY KEY (idManager));



CREATE TABLE bloodbank (
idBloodBank INT  NOT NULL ,
nameBB varchar(30) NOT NULL,
avenue_street varchar(30) NOT NULL,
building_number INT(4) NOT NULL,
neighborhood varchar(30) NOT NULL,
city varchar(30) NOT NULL,
CONSTRAINT PK_bloodbank PRIMARY KEY(idBloodBank));
```

```
CREATE TABLE manager_bloodbank (
idManager INT   NOT NULL,
idBloodBank INT   NOT NULL,
CONSTRAINT PK_manager_bloodBank PRIMARY KEY (idManager,
idBloodBank),
CONSTRAINT FK_manager_bloodBank1 FOREIGN KEY (idManager)
REFERENCES manager (idManager) ON UPDATE CASCADE ON DELETE
CASCADE,
CONSTRAINT FK_manager_bloodBank2 FOREIGN KEY (idBloodBank)
REFERENCES bloodbank (idBloodBank) ON UPDATE CASCADE ON
DELETE CASCADE);




CREATE TABLE registration_team (
idRT INT   NOT NULL ,
nameER varchar(30) NOT NULL,
surnameER varchar(30) NOT NULL,
idBloodBank INT    NOT NULL,
CONSTRAINT PK_registration_team PRIMARY KEY (idRT),
CONSTRAINT FK_registration_team FOREIGN KEY (idBloodBank)
REFERENCES bloodbank (idBloodBank) ON UPDATE CASCADE ON
DELETE CASCADE);


CREATE TABLE technical_analyst(
idTA INT   NOT NULL ,
nameTA varchar(30) NOT NULL,
surnameAT varchar(30) NOT NULL,
idBloodBank INT    NOT NULL,
CONSTRAINT PK_technical_analyst PRIMARY KEY (idTA),
CONSTRAINT FK_technical_analyst FOREIGN KEY (idBloodBank)
REFERENCES bloodbank (idBloodBank) ON UPDATE CASCADE ON
DELETE CASCADE);


CREATE TABLE blood (
idBlood INT   NOT NULL ,
bloodGroup varchar(3) NOT NULL,
idManager INT    NOT NULL,
CONSTRAINT PK_blood PRIMARY KEY (idBlood, bloodGroup),
CONSTRAINT FK_blood1 FOREIGN KEY (idManager) REFERENCES
manager (idManager) ON UPDATE CASCADE ON DELETE CASCADE);


CREATE TABLE patient (
idPatient INT   NOT NULL ,
nameP varchar(30) NOT NULL,
surnameP varchar(30) NOT NULL,
gender ENUM('M', 'F', 'N') NOT NULL,
bloodGroup varchar(30) NOT NULL,
contact varchar(20) NOT NULL,
avenue_street varchar(30) NOT NULL,
building_number INT(4) NOT NULL,
neighborhood varchar(30) NOT NULL,
city varchar(30) NOT NULL ,
dateRegisters date NOT NULL,
idRT INT    NOT NULL,
idManager INT    NOT NULL,
CONSTRAINT PK_patient PRIMARY KEY(idPatient),
CONSTRAINT FK_patient1 FOREIGN KEY (idRT) REFERENCES
registration_team (idRT) ON UPDATE CASCADE ON DELETE CASCADE,
CONSTRAINT FK_patient2 FOREIGN KEY (idManager) REFERENCES
manager (idManager) ON UPDATE CASCADE ON DELETE CASCADE);


CREATE TABLE requests (
idRequests INT   NOT NULL ,
bloodGroup varchar(3) NOT NULL,
amountBlood INT(10) NOT NULL,
CONSTRAINT PK_requests PRIMARY KEY(idRequests));
```

```
CREATE TABLE patient_requests (
idPatient INT    NOT NULL,
idRequests INT    NOT NULL,
CONSTRAINT PK_patient_requests PRIMARY KEY (idPatient,
idRequests),
CONSTRAINT FK_patient_requests1 FOREIGN KEY (idPatient)
REFERENCES patient (idPatient) ON UPDATE CASCADE ON DELETE
CASCADE,
CONSTRAINT FK_patient_requests2 FOREIGN KEY (idRequests)
REFERENCES requests (idRequests) ON UPDATE CASCADE ON DELETE
CASCADE);



CREATE TABLE hospital (
idHospital INT   NOT NULL ,
nameH varchar(30) NOT NULL,
avenue_street varchar(30) NOT NULL,
building_number INT(4) NOT NULL,
neighborhood varchar(30) NOT NULL,
city varchar(30) NOT NULL ,
idManager INT   NOT NULL,
CONSTRAINT PK_hospital PRIMARY KEY (idHospital),
```

```sql
CONSTRAINT FK_hospital FOREIGN KEY (idManager) REFERENCES
manager (idManager) ON UPDATE CASCADE ON DELETE CASCADE);



CREATE TABLE hospital_requests (
idHospital INT   NOT NULL,
idRequests INT   NOT NULL,
CONSTRAINT PK_hospital_requests PRIMARY KEY (idHospital,
idRequests),
CONSTRAINT FK_hospital_requests1 FOREIGN KEY (idHospital)
REFERENCES hospital (idHospital) ON UPDATE CASCADE ON DELETE
CASCADE,
CONSTRAINT FK_hospital_requests2 FOREIGN KEY (idRequests)
REFERENCES requests (idRequests) ON UPDATE CASCADE ON DELETE
CASCADE);


CREATE TABLE blood_patient (
idBlood INT   NOT NULL,
bloodGroup varchar(3) NOT NULL,
idPatient INT   NOT NULL,
CONSTRAINT PK_blood_patient PRIMARY KEY (idBlood, bloodGroup,
idPatient),
CONSTRAINT FK_blood_patient1 FOREIGN KEY (idBlood, bloodGroup)
REFERENCES blood (idBlood, bloodGroup) ON UPDATE CASCADE ON
DELETE CASCADE,
CONSTRAINT FK_blood_patient2 FOREIGN KEY (idPatient)
REFERENCES patient (idPatient) ON UPDATE CASCADE ON DELETE
CASCADE);



CREATE TABLE technicalAnalyst_blood(
idTA INT   NOT NULL,
idBlood INT   NOT NULL,
bloodGroup varchar(3) NOT NULL,
CONSTRAINT PK_techinalAnalyst_blood PRIMARY KEY (idTA, idBlood,
bloodGroup),
CONSTRAINT FK_techinalAnalyst_blood1 FOREIGN KEY (idTA)
REFERENCES technical_analyst (idTA) ON UPDATE CASCADE ON
DELETE CASCADE,
CONSTRAINT FK_techinalAnalyst_blood2 FOREIGN KEY (idBlood,
bloodGroup) REFERENCES blood (idBlood, bloodGroup) ON UPDATE
CASCADE ON DELETE CASCADE);



CREATE TABLE donor(
idDonor INT   NOT NULL  ,
nameD varchar(30) NOT NULL,
surnameD varchar(30) NOT NULL,
gender ENUM('M', 'F', 'N') NOT NULL,
bloodGroup varchar(3) NOT NULL,
contact varchar(20) NOT NULL,
avenue_street varchar(30) NOT NULL,
building_number INT(4) NOT NULL,
neighborhood  varchar(30) NOT NULL,
city varchar(30) DEFAULT NULL,
age INT(3) NOT NULL,
disease INT(1) DEFAULT 0,
idBlood INT(10) NOT NULL,
bloodGroupBlood varchar(3) NOT NULL,
idRT INT(10) NOT NULL,
dateRegisters date NOT NULL,
CONSTRAINT PK_donor1 PRIMARY KEY(idDonor),
CONSTRAINT CHK_donor_age CHECK (age >= 18 AND age <= 60),
CONSTRAINT FK_donor1 FOREIGN KEY (idBlood, bloodGroupBlood)
REFERENCES blood (idBlood, bloodGroup) ON UPDATE CASCADE ON
DELETE CASCADE,
CONSTRAINT FK_donor2 FOREIGN KEY (idRT) REFERENCES
registration_team (idRT) ON UPDATE CASCADE ON DELETE CASCADE);
```

Triggers:

```
DELIMITER //
```

```
CREATE TRIGGER before_insert_donor
BEFORE INSERT ON donor
FOR EACH ROW
BEGIN
    DECLARE last_donation_date DATE;

    -- Find the last donation date for the current donor
    SELECT MAX(dateRegisters) INTO last_donation_date
    FROM donor
    WHERE idDonor = NEW.idDonor;

    -- Check if the donation interval is at least 30 days
    IF last_donation_date IS NOT NULL AND
DATEDIFF(NEW.dateRegisters, last_donation_date) < 30 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Donor can only donate once every 30 days';
    END IF;
END //

DELIMITER ;
```

Insert statements

```
INSERT INTO manager (idManager,nameM, surnameM) VALUES
(1,'Abdul', 'Rahman'),
(2,'Fatima', 'Begum'),
(3,'Mohammad', 'Ali'),
(4,'Ayesha', 'Akhtar'),
(5,'Kamal', 'Hossain'),
(6,'Nazia', 'Islam'),
(7,'Rahim', 'Khan'),
(8,'Nasreen', 'Binte'),
(9,'Iqbal', 'Ahmed'),
(10,'Farida', 'Sultana');


INSERT INTO bloodbank (idBloodBank,nameBB, avenue_street,
building_number, neighborhood, city) VALUES
(1,'Dhaka Blood Bank', 'Shahbagh Road', 1234, 'Shahbagh',
'Dhaka'),
(2,'Chittagong Blood Bank', 'GEC Circle', 5678, 'Agrabad',
'Chittagong'),
(3,'Rajshahi Blood Bank', 'Boalia Road', 91011, 'Boalia', 'Rajshahi'),
(4,'Khulna Blood Bank', 'Moylapota Lane', 1213, 'Sonadanga',
'Khulna'),
(5,'Sylhet Blood Bank', 'Zinda Bazar', 1415, 'Zinda Bazar',
'Sylhet'),
(6,'Barisal Blood Bank', 'Sadarghat Road', 1617, 'Sadarghat',
'Barisal'),
(7,'Rangpur Blood Bank', 'Station Road', 1819, 'Station Road',
'Rangpur'),
(8,'Comilla Blood Bank', 'Kandirpar', 2021, 'Kandirpar', 'Comilla'),
(9,'Jessore Blood Bank', 'M.K. Road', 2223, 'M.K. Road',
'Jessore'),
(10,'Cox''s Bazar Blood Bank', 'Sea Beach Road', 2425, 'Sea
Beach', 'Cox''s Bazar');


INSERT INTO manager_bloodbank (idManager, idBloodBank) VALUES
(1, 1),
(2, 2),
(3, 3),
(4, 4),
(5, 5),
(6, 6),
(7, 7),
(8, 8),
(9, 9),
(10, 10);


INSERT INTO registration_team (idRT,nameER, surnameER,
idBloodBank) VALUES
(1,'Mohammad', 'Akram', 1),
(2,'Sultana', 'Begum', 2),
(3,'Rahman', 'Miah', 3),
(4,'Anisa', 'Binte', 4),
(5,'Mamun', 'Hossain', 5),
(6,'Sabina', 'Khatun', 6),
(7,'Mustafizur', 'Rahman', 7),
(8,'Farhana', 'Akter', 8),
(9,'Imran', 'Khan', 9),
(10,'Nasrin', 'Jahan', 10);


INSERT INTO technical_analyst (idTA,nameTA, surnameAT,
idBloodBank) VALUES
(1,'Sohel', 'Islam', 1),
(2,'Shabnam', 'Chowdhury', 2),
(3,'Aminul', 'Haque', 3),
(4,'Fahmida', 'Khatun', 4),
(5,'Rakib', 'Ahmed', 5),
(6,'Tasnim', 'Akter', 6),
(7,'Rafiq', 'Khan', 7),
(8,'Nahida', 'Binte', 8),
(9,'Jamal', 'Uddin', 9),
(10,'Salma', 'Sultana', 10);


INSERT INTO blood (idBlood,bloodGroup, idManager) VALUES
(1,'A+', 1),
(2,'B-', 2),
(3,'O+', 3),
```

```sql
(4,'AB+', 4),
(5,'B+', 5),
(6,'O-', 6),
(7,'A-', 7),
(8,'AB-', 8);


INSERT INTO patient (idPatient,nameP, surnameP, gender,
bloodGroup, contact, avenue_street, building_number, neighborhood,
city, dateRegisters, idRT, idManager) VALUES
(1,'Sadia', 'Akter', 'F', 'A+', '01711234567', 'Shahbagh Road',
567, 'Shahbagh', 'Dhaka', '2023-01-01', 1, 1),
(2,'Rahim', 'Miah', 'M', 'B-', '01987654321', 'GEC Circle', 123,
'Agrabad', 'Chittagong', '2023-02-01', 2, 2),
(3,'Sumaiya', 'Khatun', 'F', 'O+', '01897654321', 'Boalia Road',
456, 'Boalia', 'Rajshahi', '2023-03-01', 3, 3),
(4,'Kamal', 'Hossain', 'M', 'AB+', '01676543210', 'Moylapota
Lane', 789, 'Sonadanga', 'Khulna', '2023-04-01', 4, 4),
(5,'Sabina', 'Chowdhury', 'F', 'B+', '01543210987', 'Zinda Bazar',
1011, 'Zinda Bazar', 'Sylhet', '2023-05-01', 5, 5),
(6,'Mizan', 'Rahman', 'M', 'O-', '01789012345', 'Sadarghat Road',
1213, 'Sadarghat', 'Barisal', '2023-06-01', 6, 6),
(7,'Nusrat', 'Jahan', 'F', 'A-', '01987654321', 'Station Road',
1415, 'Station Road', 'Rangpur', '2023-07-01', 7, 7),
(8,'Imran', 'Khan', 'M', 'AB-', '01876543210', 'Kandirpar', 1617,
'Kandirpar', 'Comilla', '2023-08-01', 8, 8),
(9,'Sadia', 'Sultana', 'F', 'A+', '01654321098', 'M.K. Road',
1819, 'M.K. Road', 'Jessore', '2023-09-01', 9, 9),
(10,'Rahman', 'Uddin', 'M', 'B-', '01789012345', 'Sea Beach
Road', 2021, 'Sea Beach', 'Cox''s Bazar', '2023-10-01', 10, 10);


INSERT INTO requests (idRequests,bloodGroup, amountBlood)
VALUES
(1,'A+', 10),
(2,'B-', 5),
(3,'O+', 8),
(4,'AB+', 12),
(5,'B+', 7),
(6,'O-', 6),
(7,'A-', 9),
(8,'AB-', 15),
(9,'A+', 11),
(10,'B-', 4);


INSERT INTO patient_requests (idPatient, idRequests) VALUES
(1, 1),
(2, 2),
(3, 3),
(4, 4),
(5, 5),
```

```sql
(6, 6),
(7, 7),
(8, 8),
(9, 9),
(10, 10);


INSERT INTO hospital (idHospital,nameH, avenue_street,
building_number, neighborhood, city, idManager) VALUES
(1,'Dhaka Medical Center', 'Green Road', 123, 'Dhanmondi',
'Dhaka', 1),
(2,'Chittagong General Hospital', 'Chawkbazar', 456, 'Chawkbazar',
'Chittagong', 2),
(3,'Rajshahi Central Hospital', 'Boalia Road', 789, 'Boalia',
'Rajshahi', 3),
(4,'Khulna City Hospital', 'Shibbari', 1011, 'Shibbari', 'Khulna', 4),
(5,'Sylhet Medical Complex', 'Zinda Bazar', 1213, 'Zinda Bazar',
'Sylhet', 5),
(6,'Barisal General Hospital', 'Sadarghat Road', 1415, 'Sadarghat',
'Barisal', 6),
(7,'Rangpur Medical Center', 'Station Road', 1617, 'Station Road',
'Rangpur', 7),
(8,'Comilla Central Hospital', 'Kandirpar', 1819, 'Kandirpar',
'Comilla', 8),
(9,'Jessore Health Complex', 'M.K. Road', 2021, 'M.K. Road',
'Jessore', 9),
(10,'Cox''s Bazar Hospital', 'Sea Beach Road', 2223, 'Sea Beach',
'Cox''s Bazar', 10);


INSERT INTO hospital_requests (idHospital, idRequests) VALUES
(1, 1),
(2, 2),
(3, 3),
(4, 4),
(5, 5),
(6, 6),
(7, 7),
(8, 8),
(9, 9),
(10, 10);


INSERT INTO blood_patient (idBlood, bloodGroup, idPatient) VALUES
(1, 'A+', 1),
(2, 'B-', 2),
(3, 'O+', 3),
(4, 'AB+', 4),
(5, 'B+', 5),
(6, 'O-', 6),
(7, 'A-', 7),
(8, 'AB-', 8);
```

```
INSERT INTO technicalAnalyst_blood (idTA, idBlood, bloodGroup)
VALUES
(1, 1, 'A+'),
(2, 2, 'B-'),
(3, 3, 'O+'),
(4, 4, 'AB+'),
(5, 5, 'B+'),
(6, 6, 'O-'),
(7, 7, 'A-'),
(8, 8, 'AB-');


INSERT INTO donor (idDonor,nameD, surnameD, gender, bloodGroup,
contact, avenue_street, building_number, neighborhood, city, age,
disease, idBlood, bloodGroupBlood, idRT, dateRegisters) VALUES
(1,'Abdul', 'Karim', 'M', 'A+', '01911223344', 'Green Road', 123,
'Dhanmondi', 'Dhaka', 30, 0, 1, 'A+', 1, '2023-11-01'),
(2,'Farida', 'Akhter', 'F', 'B-', '01899887766', 'Chawkbazar',
456, 'Chawkbazar', 'Chittagong', 25, 1, 2, 'B-', 2, '2023-11-02'),
(3,'Rahim', 'Uddin', 'M', 'O+', '01788996655', 'Boalia Road', 789,
'Boalia', 'Rajshahi', 35, 0, 3, 'O+', 3, '2023-11-03'),
(4,'Ayesha', 'Khatun', 'F', 'AB+', '01667778888', 'Shibbari',
1011, 'Shibbari', 'Khulna', 28, 0, 4, 'AB+', 4, '2023-11-04'),
(5,'Kamal', 'Islam', 'M', 'B+', '01556667777', 'Zinda Bazar',
1213, 'Zinda Bazar', 'Sylhet', 32, 0,5,'B+',5,'2023-12-05');

INSERT INTO donor (idDonor,nameD, surnameD, gender, bloodGroup,
contact, avenue_street, building_number, neighborhood, city, age,
disease, idBlood, bloodGroupBlood, idRT, dateRegisters) VALUES
(6,'Nasrin', 'Jahan', 'F', 'A+', '01447788990', 'Mirpur Road',
567, 'Mirpur', 'Dhaka', 26, 0, 1, 'A+', 1, '2023-11-05'),
(7,'Imran', 'Kabir', 'M', 'B-', '01336699887', 'Agrabad', 123,
'Agrabad', 'Chittagong', 40, 0, 2, 'B-', 2, '2023-11-06'),
(8,'Sabina', 'Hossain', 'F', 'O+', '01998887766', 'Lalbagh', 101,
'Lalbagh', 'Dhaka', 22, 1, 3, 'O+', 3, '2023-11-07'),
(9,'Kamal', 'Uddin', 'M', 'AB+', '01775554444', 'Shibbari', 202,
'Shibbari', 'Khulna', 29, 0, 4, 'AB+', 4, '2023-11-08'),
(10,'Sadia', 'Khatun', 'F', 'B+', '01661112222', 'Sylhet Road',
303, 'Sylhet Road', 'Sylhet', 33, 0, 5, 'B+', 5, '2023-11-09'),
(11,'Rahman', 'Ahmed', 'M', 'O-', '01557778888', 'Moylapota
Lane', 404, 'Sonadanga', 'Khulna', 31, 0, 6, 'O-', 6,
'2023-11-10'),
(12,'Farida', 'Sultana', 'F', 'A-', '01889991111', 'Zinda Bazar',
505, 'Zinda Bazar', 'Sylhet', 27, 0, 7, 'A-', 7, '2023-11-11'),
(13,'Iqbal', 'Hossain', 'M', 'AB-', '01667770000', 'Station Road',
606, 'Station Road', 'Rangpur', 38, 0, 8, 'AB-', 8, '2023-11-12'),
(14,'Nazmul', 'Islam', 'M', 'A+', '01446669999', 'Kandirpar', 707,
'Kandirpar', 'Comilla', 34, 0, 1, 'A+', 9, '2023-11-13'),
(15,'Rabeya', 'Binte', 'F', 'B-', '01334445555', 'Sea Beach Road',
808, 'Sea Beach', 'Cox''s Bazar', 36, 0, 2, 'B-', 10,
'2023-11-14');
```

Finding the Normal Form (1st/2nd/3rd/3+ or BC-NF)

Donor

- Atomic fields, no repetitive groups
- Primary key (idDonor) determines all attributes
- No partial dependencies -> Satisfies 3NF

Blood

- Atomic fields
- Primary key (idBlood, bloodGroup) determines all attributes
- No transitive dependencies -> Satisfies 3NF

Patient

- Atomic fields
- Primary key (idPatient) determines all attributes
- No partial or transitive dependencies -> Satisfies 3NF

Requests

- Atomic fields
- Primary key (idRequests) determines attributes
- No additional dependencies
  -> Satisfies 3NF

Hospital

- Atomic fields

- Primary key (idHospital) determines attributes
- No partial or transitive dependencies -> Satisfies 3NF

Blood Bank

- Structure similar to Hospital -> Satisfies 3NF

Manager

- Atomic fields
- Primary key (idManager) determines attributes
- No additional dependencies -> Satisfies 3NF

Registration Team

- Atomic fields
- Primary key (idRT) determines attributes
- No partial or transitive dependencies -> Satisfies 3NF

Technical Analyst

- Structure similar to Registration Team -> Satisfies 3NF

Manager_BloodBank

- Composite primary key mapping manager to blood bank
- No repetitive groups
- Satisfies at least 2NF

Patient_Requests

- Composite primary key mapping requests to patients
- No repetitive groups
- Satisfies at least 2NF

Hospital_Requests

- Composite primary key mapping hospitals to requests
- No repetitive groups
- Satisfies at least 2NF

Blood_Patient

- Composite primary key mapping blood units to patients
- No repetitive groups
- Satisfies at least 2NF

TechnicalAnalyst_Blood

- Composite primary key mapping analysts to blood units
- No repetitive groups
- Satisfies at least 2NF

The association tables satisfy 2NF or higher since they act as mapping tables between two entities.

Boyce-Codd Normal Form (BCNF):

Donor, Blood, Patient, Requests, Hospital, Blood Bank, Manager, Registration Team, Technical Analyst

- These tables are already in 3NF
- Their candidate keys determine all attributes

- No non-prime attributes are transitively dependent on candidate keys → Satisfy BCNF

Association Tables

- Manager_BloodBank
    - Candidate keys are (idManager, idBloodBank)
    - No transitive dependencies, hence in BCNF
- Patient_Requests
    - Candidate keys are (idPatient, idRequests)
    - No transitive dependencies, hence in BCNF
- Hospital_Requests
    - Candidate keys are (idHospital, idRequests)
    - No transitive dependencies, hence in BCNF
- Blood_Patient
    - Candidate keys are (idBlood, bloodGroup, idPatient)
    - No transitive dependencies, hence in BCNF
- TechnicalAnalyst_Blood
    - Candidate keys are (idTA, idBlood, bloodGroup)
    - No transitive dependencies, hence in BCNF

all tables satisfy BCNF since their candidate keys determine all attributes in their schema. There are no partial or transitive dependencies involving non-prime attributes.

Future Works and Conclusions

Expanding on BloodBank database,I want to consider incorporating additional features or tables to enhance the functionality and address potential requirements. Introducing tables for managing user accounts, roles, and permissions.Implementing authentication mechanisms for different types of users (e.g., donors, hospital staff, administrators).Enhanced the blood inventory management system with more details on blood storage conditions Integrating tracking for blood component levels (plasma, platelets, etc.) if applicable. Include tables to track quality control measures for blood testing and screening.Implement mechanisms to flag and manage blood units that do not meet quality standards.Implement a robust backup and disaster recovery plan to safeguard critical data.Regularly backup the database and implement mechanisms for quick recovery in case of data loss.Overall this was an excellent experience. I have learnt a substantial amount of knowledge on the topic