[BLOOD BANK Management System (DBMS)]

Project submitted to the
SRM University – AP, Andhra Pradesh
for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology
In
Computer Science and Engineering
School of Engineering and Sciences

Submitted by
Praneeth Bulusu(AP22110010049)
Devesh Rawat(AP22110010038)
K Rahul(AP22110010052)
M Vignesh(AP22110010016)

SRM University–AP
Neerukonda, Mangalagiri, Guntur
Andhra Pradesh – 522 240
[December, 2022]

INDEX

Abstract

The Blood Bank Management System (BBMS) uses a sophisticated database to manage all the information about donors and the blood inventory. Here's a breakdown of how it works:

- **Database Structure:** The system uses a relational database that organizes data into tables, like donors, recipients, and blood units. This setup reduces duplication and keeps the data accurate and consistent, with clear connections between different pieces of data.
- **Schema Design:** The layout of the database is carefully planned to handle the complex relationships within the blood donation system. Diagrams help visualize these relationships, making the database easier to manage and search.
- **Performance Optimization:** The system uses special indexing techniques to make searches faster, especially for common search terms like blood types or appointment dates.
- **Transaction Management:** The database ensures that all data operations, like recording a blood donation or updating the inventory, are performed correctly and safely, maintaining the accuracy and consistency of the data even when many operations are happening at once.

Abbreviations

1. BBMS: Blood Bank Management System

2.ACID: Atomicity, Consistency, Isolation, Durability

3. ER: Entity-Relationship

Aim :

Blood Bank Management System: The focus is on creating a system that manages various aspects of a blood bank, including donor information, recipient details, blood donations, and blood inventory tracking.
Using Transactions and Views: The title emphasizes the utilization of two key database concepts: transactions and views. Transactions ensure data integrity by grouping database operations into atomic units, while views provide a simplified and organized presentation of data to users.

Attributes — characteristics of an entity, and has an oval symbol.

There are different types of attributes :

- ***Key attribute:*** An attribute uniquely distinguishes the entity in an entity set.
- ***Simple attribute:*** An attribute that cannot be further subdivided into components.
- ***Composite attribute:*** An attribute that can be split into components.
- ***Single-valued attribute:*** The attribute which takes up only a single value for each entity instance.

- ***Multi-valued attribute:*** The attribute which takes up more than a single value for each entity instance.
- ***Stored attribute:*** An attribute that stores the data which can be used to get the derived attribute.
- ***Derived attribute:*** An attribute that can be derived from other attributes.

The entities in the blood bank management system database are:

Donor:
Attributes:
Donor_ID (Primary Key)
Name
Blood_Group
Contact_Number
Address
Age
Gender
Last_Donation_Date
Medical_History
Recipient:
Attributes:
Recipient_ID (Primary Key)
Name
Blood_Group
Contact_Number
Address
Age
Gender
Required_Blood_Units
Hospital_Details
Blood_Bank:
Attributes:
Bank_id (Primary Key)
Name
Location
Contact_Number
Blood_Donation:
Attributes:

Donation_ID (Primary Key)
Donor_ID (Foreign Key referencing Donor)
Recipient_ID (Foreign Key referencing Recipient)
Donation_Date
Blood_Group
Quantity_Donated

RELATIONSHIP :
A relationship is an association among several entities
Relationships:
In the blood bank management system, the relationships between entities can be defined as follows:

Donor Donates Blood:
One donor can make multiple blood donations.
One blood donation is made by one donor.
Relationship between Donor and Blood_Donation.
One-to-Many relationship.
Foreign key: Donor_ID in the Blood_Donation table references Donor_ID in the Donor table.

Blood Donation Belongs to Donor:
Each blood donation is associated with one donor.
Relationship between Blood_Donation and Donor.
Many-to-One relationship.
Foreign key: Donor_ID in the Blood_Donation table references Donor_ID in the Donor table.

Blood Donation is For Recipient:
Each blood donation is intended for one recipient.
One recipient can receive multiple blood donations.
Relationship between Blood_Donation and Recipient.
One-to-Many relationship.
Foreign key: Recipient_ID in the Blood_Donation table references Recipient_ID in the Recipient table.

Recipient Receives Blood from Donor:
One recipient can receive blood from multiple donors.
Each blood donation is received by one recipient.
Relationship between Recipient and Blood_Donation.
One-to-Many relationship.

Foreign key: Recipient_ID in the Recipient table references Recipient_ID in the Blood_Donation table.

Donor May Have Medical History:
Each donor may have a medical history.
Optional relationship between Donor and Medical_History.
One-to-One or Zero relationship.
Foreign key: None (if using a separate table for medical history, it would typically include Donor_ID as a foreign key).

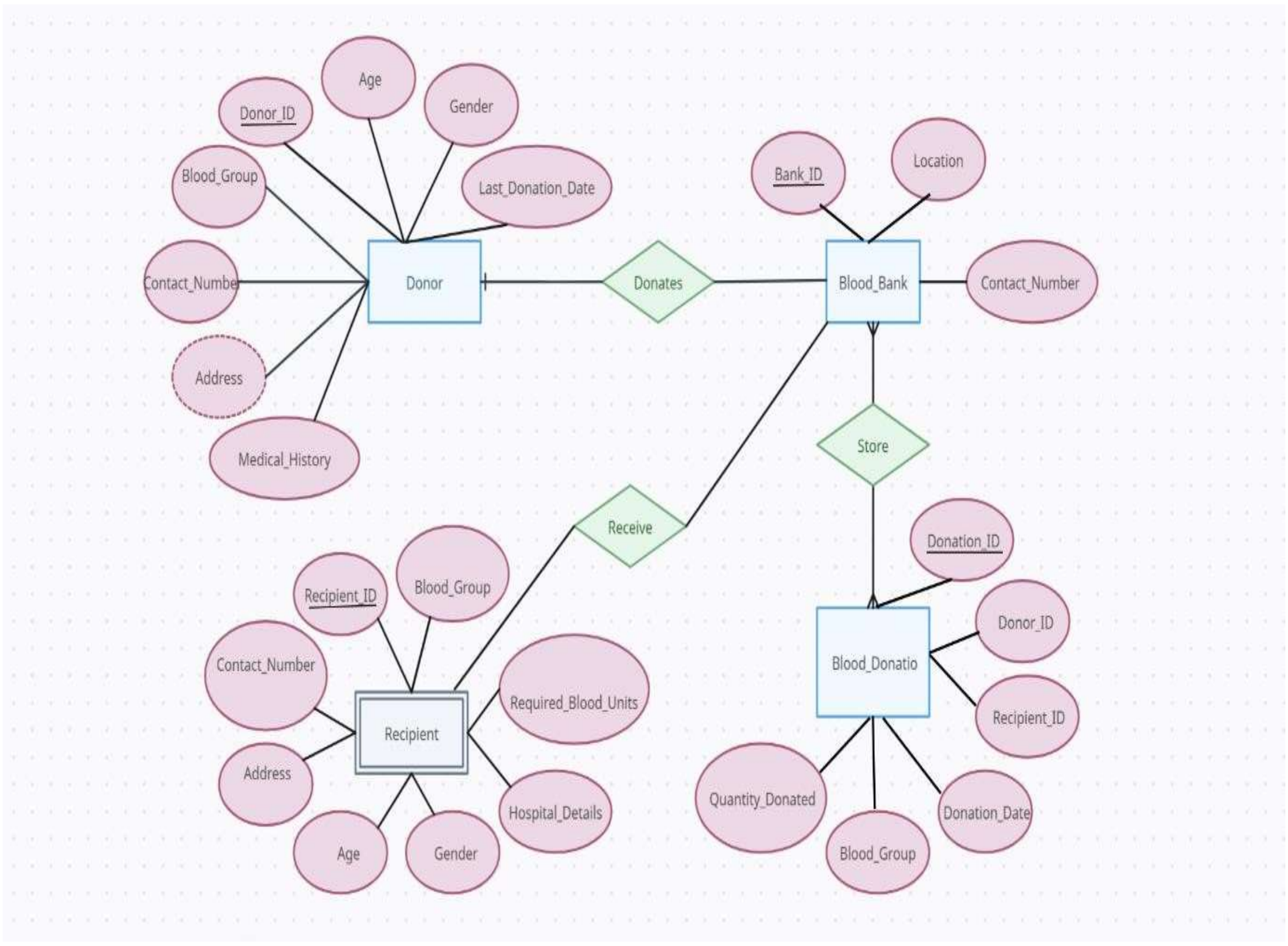
Recipient is Admitted to Blood Bank:
One recipient may be admitted to multiple blood banks.
Each blood bank may admit multiple recipients.
Relationship between Recipient and Blood_Bank.
Many-to-Many relationship.
Junction table might be needed to represent this relationship, containing Recipient_ID and Bank_ID.

These relationships define how different entities within the blood bank management system are connected and interact with each other. They are crucial for maintaining data integrity and ensuring that the database accurately represents the real-world scenario of blood donation and distribution.

E-R Model:

ER model stands for an Entity-Relationship model.
It is a high-level data model. This model is used to define the data elements and relationships for a specified system.

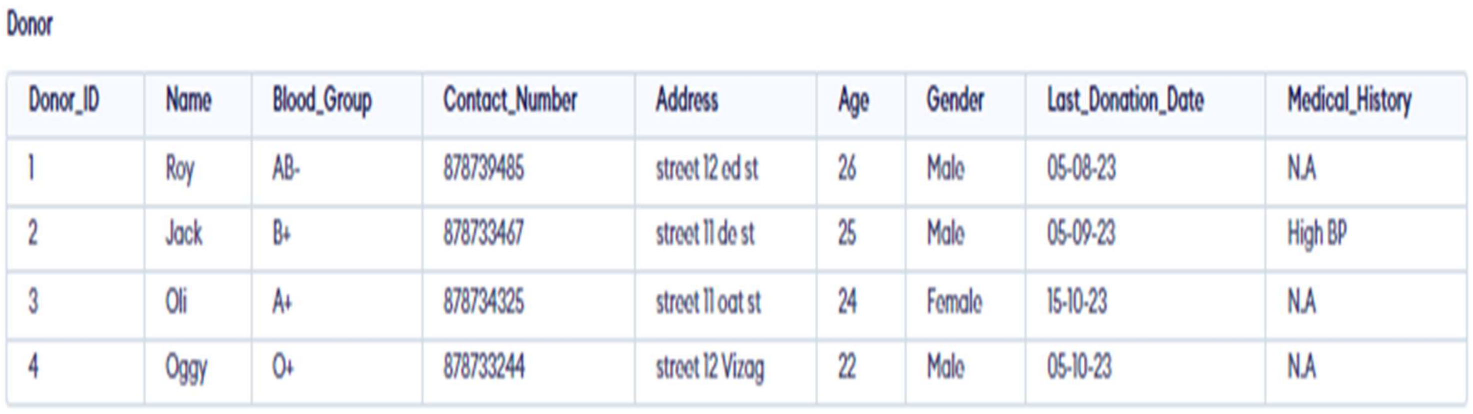
It develops a conceptual design for the database. It also develops a very simple and easy-to-design view of data.



SQL CODE :

Donor Table:

```
CREATE TABLE Donor (
 Donor_ID INT PRIMARY KEY,
 Name VARCHAR(100),
 Blood_Group VARCHAR(3),
 Contact_Number VARCHAR(15),
 Address VARCHAR(255),
 Age INT,
 Gender VARCHAR(10),
 Last_Donation_Date DATE,
 Medical_History TEXT
);
```

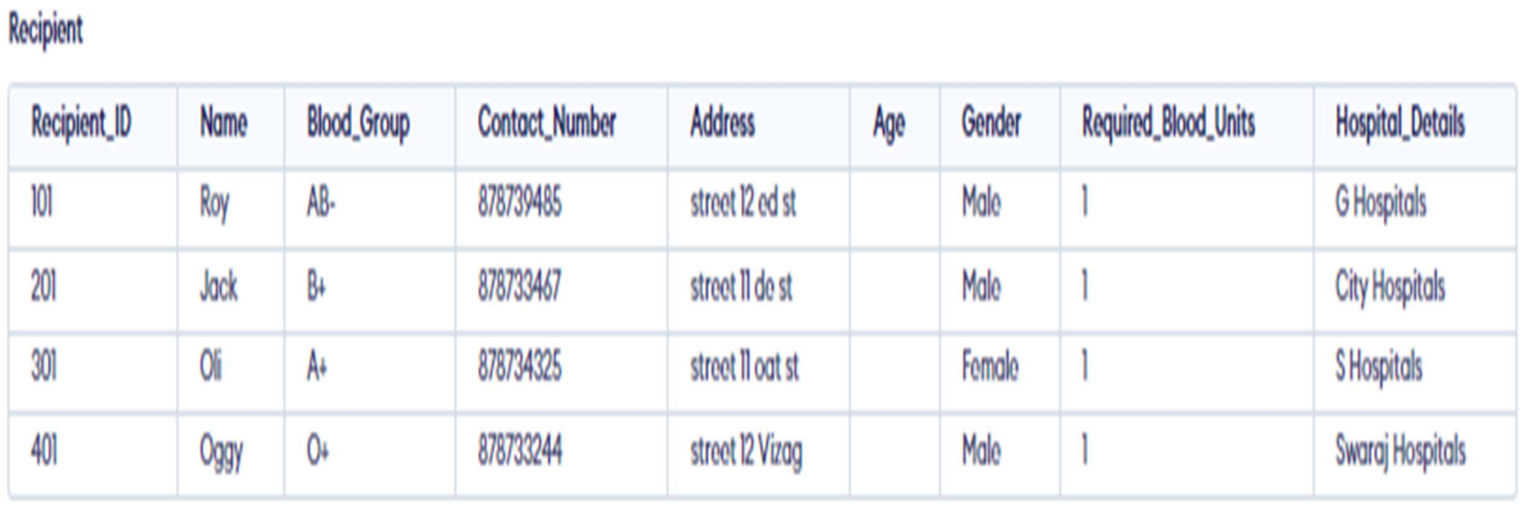


Donor

Donor_ID	Name	Blood_Group	Contact_Number	Address	Age	Gender	Last_Donation_Date	Medical_History
1	Roy	AB-	878739485	street 12 ed st	26	Male	05-08-23	N.A
2	Jack	B+	878733467	street 11 de st	25	Male	05-09-23	High BP
3	Oli	A+	878734325	street 11 oat st	24	Female	15-10-23	N.A
4	Oggy	O+	878733244	street 12 Vizag	22	Male	05-10-23	N.A

Recipient Table:

```
CREATE TABLE Recipient (
 Recipient_ID INT PRIMARY KEY,
 Name VARCHAR(100),
 Blood_Group VARCHAR(3),
 Contact_Number VARCHAR(15),
 Address VARCHAR(255),
 Age INT,
 Gender VARCHAR(10),
 Required_Blood_Units INT,
 Hospital_Details VARCHAR(255)
);
```

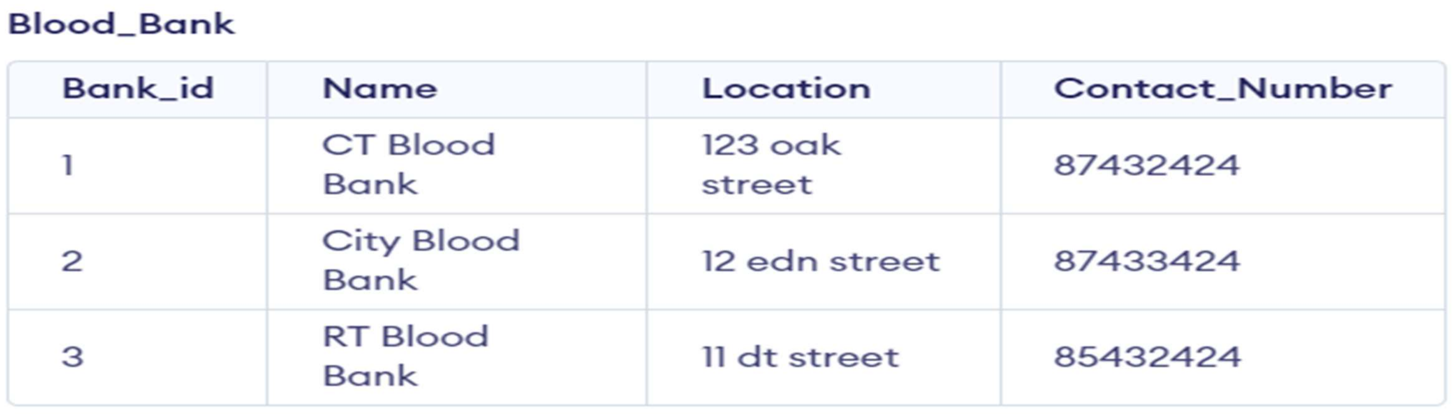


Recipient

Recipient_ID	Name	Blood_Group	Contact_Number	Address	Age	Gender	Required_Blood_Units	Hospital_Details
101	Roy	AB-	878739485	street 12 ed st		Male	1	G Hospitals
201	Jack	B+	878733467	street 11 de st		Male	1	City Hospitals
301	Oli	A+	878734325	street 11 oat st		Female	1	S Hospitals
401	Oggy	O+	878733244	street 12 Vizag		Male	1	Swaraj Hospitals

Blood Bank:

```
CREATE TABLE Blood_Bank(
Bank_id INT PRIMARY KEY,
Name VARCHAR(100),
LOCATION VARCHAR(100),
CONTACT_NUMBER VARCHAR(50)
);
```

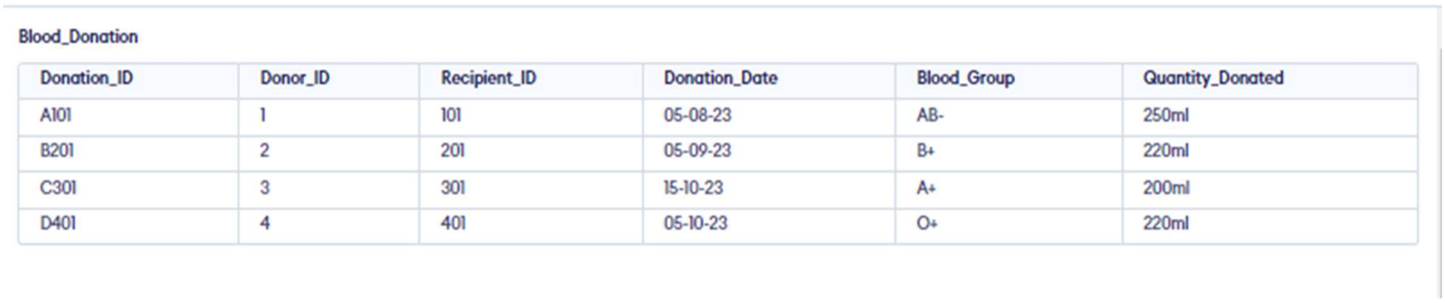


Blood_Bank

Bank_id	Name	Location	Contact_Number
1	CT Blood Bank	123 oak street	87432424
2	City Blood Bank	12 edn street	87433424
3	RT Blood Bank	11 dt street	85432424

Blood Donation:

```
CREATE TABLE Blood_Donation(
Donation_ID INT PRIMARY KEY,
Donor_ID INT,
Recipient_ID INT,
Donation_Date DATE,
Blood_Group VARCHAR(3),
Quantity_Donated INT,
FOREIGN KEY(Donor_ID) REFERENCES
Donor(Donor_ID),
FOREIGN KEY (Recipient_ID) REFERENCES
Recipient(Recipient_ID)
);
```



Blood_Donation

Donation_ID	Donor_ID	Recipient_ID	Donation_Date	Blood_Group	Quantity_Donated
A101	1	101	05-08-23	AB-	250ml
B201	2	201	05-09-23	B+	220ml
C301	3	301	15-10-23	A+	200ml
D401	4	401	05-10-23	O+	220ml

Normalization:

Normalization is the process of organizing the attributes and tables of a relational database to minimize redundancy and dependency. There are different levels of normalization, commonly referred to as normal forms (1NF, 2NF, 3NF, BCNF, etc.).

First Normal Form (1NF):

Ensure that each table has a primary key.
Eliminate repeating groups within tables.
Ensure that each attribute holds a single value.
Ensure that attribute names are unique within the table.
The provided tables already satisfy 1NF as they have primary keys, no repeating groups, and attribute names are unique.

Second Normal Form (2NF):

Be in 1NF.
All non-key attributes are fully functional dependent on the primary key.
In the provided schema:

All tables are already in 2NF because each non-key attribute is fully functionally dependent on the primary key.
Third Normal Form (3NF):

Be in 2NF.
Eliminate transitive dependencies.

To achieve 3NF, we need to ensure that there are no transitive dependencies in each table. Let's examine each table:

Donor Table:
No transitive dependencies are present.
Recipient Table:
No transitive dependencies are present.
Blood_Bank Table:
No transitive dependencies are present.
Blood_Donation Table:
Transitive dependency: Donor_ID -> Last_Donation_Date.
Solution: Remove Last_Donation_Date from this table and create a separate table for tracking donor's last donation date.

VIEWS:

View to Retrieve Donors and Their Last Donation Dates:

```
CREATE VIEW Donor_Last_Donation AS
SELECT d.Donor_ID, d.Name, d.Last_Donation_Date
FROM Donor d;
```

View to Retrieve Recipients and Their Required Blood Units:

```
CREATE VIEW Recipient_Requirements AS
SELECT r.Recipient_ID, r.Name, r.Required_Blood_Units
FROM Recipient r;
```

View to Retrieve Donors with No Medical History:

```
CREATE VIEW Donors_No_Medical_History AS
SELECT d.Donor_ID, d.Name, d.Blood_Group, d.Age, d.Gender
FROM Donor d
WHERE d.Medical_History IS NULL;
```

View to Retrieve Blood Donations with Donor and Recipient Details:

```
CREATE VIEW Blood_Donation_Details AS
SELECT bd.Donation_ID, d.Name AS Donor_Name, r.Name AS
Recipient_Name, bd.Donation_Date, bd.Blood_Group,
bd.Quantity_Donated
FROM Blood_Donation bd
JOIN Donor d ON bd.Donor_ID = d.Donor_ID
JOIN Recipient r ON bd.Recipient_ID = r.Recipient_ID;
```

View to Retrieve Blood Donations Made by Donors Under 30:

```
CREATE VIEW Blood_Donations_Under_30 AS
SELECT bd.Donation_ID, d.Name AS Donor_Name,
bd.Donation_Date, bd.Blood_Group, bd.Quantity_Donated
FROM Blood_Donation bd
JOIN Donor d ON bd.Donor_ID = d.Donor_ID
WHERE d.Age < 28;
```

THANK
YOU