

DBMS PROJECT

Name : AMRUTHA U
Section : SECTION F
SRN : PES1201801879
Semester : 4th SEMESTER

Blood Bank Management System

Table of Contents

1. Description	4
2. Technology Stack	5
3. Tables	6
4. ER Diagram	7
5. Schema	8
6. Normalisation	9
7. Functional Dependencies	11
8. Testing Lossless Join Property	14
9. Check Constrains	16
10. Referential Integrity Constraints	17
11. Triggers	19
12. Complex Queries	21
13. Front – End	24
14. Conclusion	36

DESCRIPTION

Blood bank management system is a database system aimed at managing all the transactions that occur in a blood bank. The main transactions involve maintaining the stock of blood and its components, records of the people who donate blood and those who receive blood. This database implementation is a simplified version of the real world blood bank.

- It stores records of all the donors and their health conditions. Every time a donor intends to donate blood his health conditions are updated and checked for the eligibility to donate.
- It stores records of all those who have donated blood ,the blood groups ,the amount donated and the date of donation.
- It has a record of all the employees working in the blood bank and also has one manager.
- It has records of the hospitals that have ordered for blood.
- For every order placed by a hospital for blood for a particular patient, those patient details are also stored.
- The details of the people who receive the blood on behalf of the patient are also stored and are given two modes of payment – pay or replacement.
- There is table that has the stock of how much blood of each type is present at any state and the cost.
- There is a table that has all compatible types of blood for each blood group ,so incase of shortage of one group, there is an alternative to give.
- There is a bill that is generated every time a receiver chooses to pay for the blood received.

TECHNOLOGY STACK

- FRONT-END : HTML, CSS, BOOTSTRAP
- BACK-END : MySQL , php, MariaDB
- SERVER :Xampp-Apache,phpMyAdmin

TABLES

bloodbank temp
blood_group : varchar(3)
volume/250 : decimal(14,4)
cost : float

bloodbank bill
bill_id : int(11)
cost : int(11)
rec_id : varchar(5)
patient_id : varchar(8)
h_pid : varchar(10)
name : varchar(15)
reason : varchar(50)
blood_group : varchar(3)
bottles : int(11)
payment : varchar(15)
received_on : date
emp_id : varchar(5)

bloodbank stock
blood_group : varchar(3)
volume : int(11)
refill : int(11)
cost : float

bloodbank compatible
blood_group : varchar(3)
c : varchar(3)

bloodbank health
donar_id : varchar(5)
smoking : int(11)
alcohol : int(11)
cancer : int(11)
anemia : int(11)
surgery : int(11)
last_donated : date

bloodbank blood
donar_id : varchar(5)
volume : int(11)
donated_date : date
id : int(11)
blood_group : varchar(3)

bloodbank employee
emp_id : varchar(5)
e_name : varchar(15)
address : varchar(30)
phone_no : bigint(20)
email_id : varchar(20)

bloodbank donar
donar_id : varchar(5)
donar_name : varchar(15)
address : varchar(30)
phone_no : bigint(20)
email_id : varchar(20)
emp_id : varchar(5)
blood_group : varchar(3)
age : int(11)
height : float
weight : float

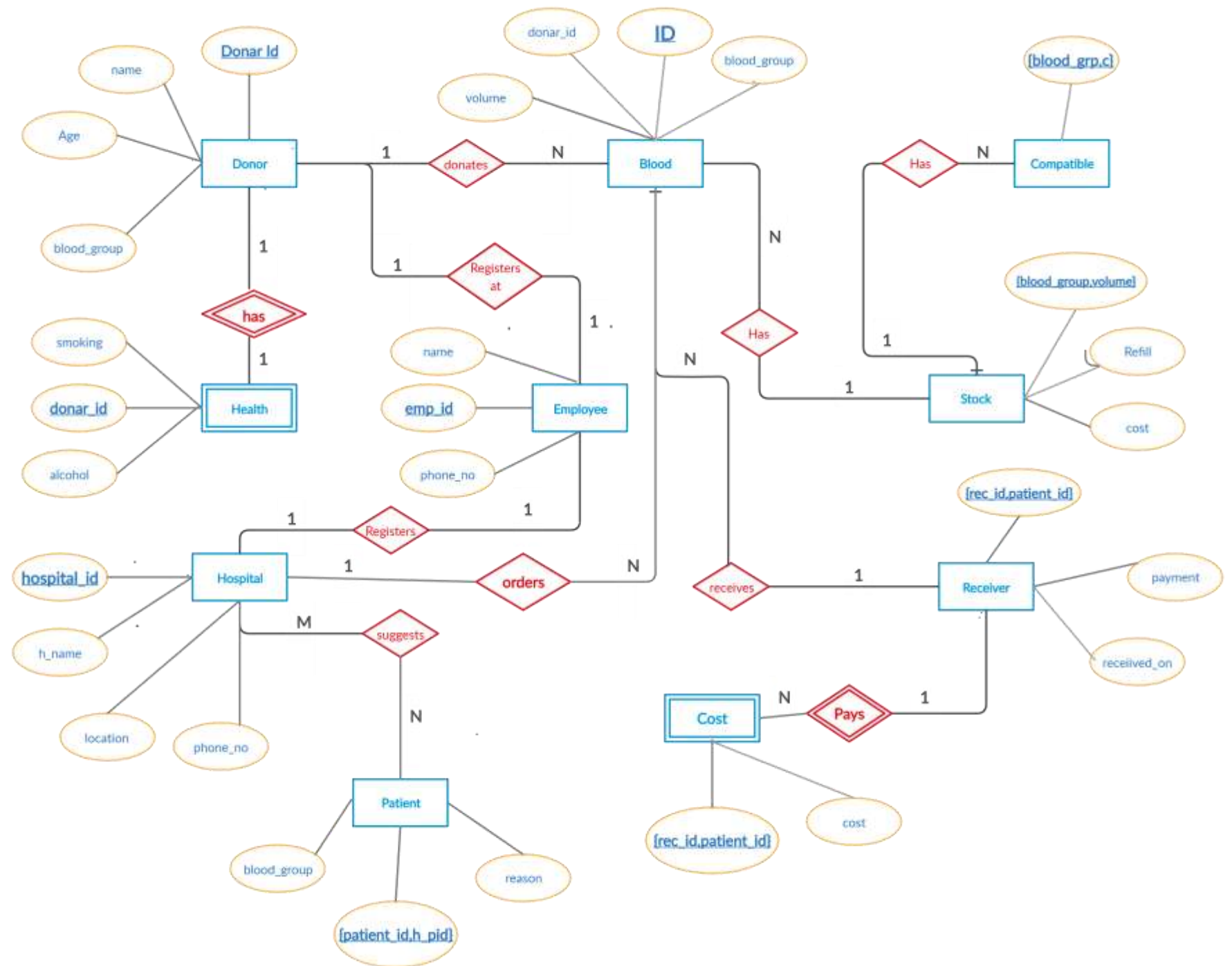
bloodbank receiver
rec_id : varchar(5)
patient_id : varchar(8)
payment : varchar(15)
received_on : date
emp_id : varchar(5)

bloodbank cost
rec_id : varchar(5)
patient_id : varchar(8)
cost : float

bloodbank hospital
hospital_id : varchar(8)
h_name : varchar(30)
location : varchar(30)
phone_no : bigint(20)

bloodbank patient
patient_id : varchar(8)
hospital_id : varchar(8)
h_pid : varchar(10)
name : varchar(15)
reason : varchar(50)
blood_group : varchar(3)
bottles : int(11)
date : date

ER DIAGRAM



SCHEMA

- **Donar** (donar_id, donar_name, address, phone_no, email_id, blood_group, age, height, weight, emp_id)
- **Health**(donar_id, smoking, alcohol, cancer, anemia, surgery, last_donated)
- **Blood**(donar_id, blood_group, volume, donated_date)
- **Stock**(blood_group, volume, refill, cost)
- **Compatible**(blood_group, c)
- **Temp**(blood_group, volume/250, cost)
- **Cost**(rec_id, patient_id, cost)

- **Hospital**(hospital_id, h_name, location, phone_no)
- **Patient**(patient_id, hospital_id, h_pid, name, reason, blood_group, bottles, date)
- **Receiver**(rec_id, patient_id, payment, received_on, emp_id)
- **Employee**(emp_id, e_name, address, phone_no, email_id)
- **Bill**(bill_id, rec_id, patient_id, h_pid, name, reason, blood_group, bottles, payment, cost, received_on, emp_id)

Bold – Relations ; **Grey** – Decomposed Relations ; **Blue** – Foreign keys ; Underline – Primary keys

NORMALISATION

1. First Normal Form-

1. Created a new table called “health” because health risk is a multivalued attribute.
2. Created a new table called “compatible” because it was a multivalued attribute in stock.

2. Second Normal Form-

1. The attribute “compatible” in stock had partial dependency on the primary key, it depended on only “blood_group”, 2NF was achieved by the decomposition done to achieve 1NF.

3. Third Normal Form –

- 1.The attribute “cost” in the receiver table had transitive dependence on the attribute payment. Created a table called “cost” to achieve 3NF.

NORMALISATION

1. Donar(donar_id, donar_name, address, phone_no, email_id, blood_group, age, height, weight, health_risk, emp_id)
The attribute “health_risk” is a multivalued attribute. To achieve 1NF, the table was decomposed into ‘donar’ and ‘health’.
 1. Donar (donar_id, donar_name, address, phone_no, email_id, blood_group, age, height, weight, emp_id)
 2. Health(donar_id, smoking, alcohol, cancer, anemia, surgery, last_donated)

2. Stock(blood_group, volume, refill, compatible, cost)
The attribute “compatible” is a multivalued attribute and had partial dependency on primary key, it didn’t depend on volume. To achieve 1NF, the table was decomposed into ‘stock’ and ‘compatible’.
 1. Stock(blood_group, volume, refill, cost)
 2. Compatible(blood_group, c)

3. Receiver(rec_id, patient_id, payment, cost, received_on, emp_id)
The attribute “cost” has a transitive dependence on a non-prime attribute “payment”. To achieve 3NF, the table was decomposed into “receiver” and “cost”.
 1. Receiver(rec_id, patient_id, payment, received_on, emp_id)
 2. Cost(rec_id, patient_id, cost)

FUNCTIONAL DEPENDENCIES

1.DONAR

1. Primary Key – donar_id
- 2.Candidate Key - { donar_id,phone_no,email_id}
- 3.{donar_id}⁺ = {donar_id, donar_name, address, phone_no, email_id, blood_group, age, height,weight, emp_id}
- 4.Foreign Key – emp_id

2. HEALTH (weak entity)

- 1.Primary key – donar_id (foreign key reference)
- 2.{donar_id}⁺ = {donar_id, smoking, alcohol, cancer, anemia, surgery, last_donated}

3.BLOOD

- 1.Primary key – Id
2. {donar_id}⁺ = {donar_id, blood_group, volume, donated_date}
- 3.Foreign Key – donar_id

4. STOCK

- 1.Primary Key – blood_group,volume
2. {blood_group,volume}⁺ = {blood_group,volume,refill,cost}

FUNCTIONAL DEPENDENCIES

5.COMPATIBLE

- 1.Primary Key – blood_group,c
- 2.{blood_group}⁺ = {blood_group,c}
- 3.Foreign Key – blood_group

6.HOSPITAL

- 1.Primary Key: hospital_id
- 2.Candidate Key: {hospital_id,phone_no}
3. {hospital_id}⁺ = {hospital_id, h_name, location, phone_no}

7. PATIENT

- 1.Primary Key – {patient_id,h_pid}
- 2.{patient_id.h_pid}⁺ = {patient_id, hospital_id, h_pid, name, reason, blood_group, bottles, date}

8.RECEIVER

- 1.Primary Key – {rec_id,patient_id}
- 2.{rec_id, patient_id}⁺ = {rec_id, patient_id, payment, received_on, emp_id}

FUNCTIONAL DEPENDENCIES

9.EMPLOYEE

- 1.Primary Key - emp_id
- 2.{emp_id}⁺ = {emp_id, e_name, address, phone_no, email_id}

10. COST(weak entity)

- 1.Primary Key – {rec_id,patient_id}
- 2.{rec_id,patient_id}⁺ = {rec_id,patient_id,cost}

11.BILL (temporary table for frontend)

- 1.Primary Key - bill_id
- 2.Candidate Key – {bill_id,patient_id,h_pid}
- 3.FD
 - 1.{patient_id}⁺ = {patient_id,name,reason,blood_group,bottles}
 - 2.{rec_id}⁺ = {payment,cost,received_on}
- 4.Foreign Key – emp_id

12.TEMP (temporary table for frontend)

- 1.{blood_group,volume}⁺ = {blood_group,volume,cost}

TESTING LOSSLESS JOIN PROPERTY

The donar table was decomposed into two tables donor and health to bring it to 1NF and it obeys the lossless join property.

```
SELECT donar.donar_id,donar_name,address,phone_no,email_id,emp_id,age,height,weight,smoking,alcohol,cancer,anemia surgery,last_donated FROM donar INNER join health ON donar.donar_id=health.donar_id
```

donar_id	donar_name	address	phone_no	email_id	emp_id	age	height	weight	smoking	alcohol	cancer	surgery	last_donated
D0001	Donar1	BSK	9156209876	don1@gmail.com	E1000	25	5.5	65	1	0	0	0	2019-05-01
D0002	Donar2	INR	9156209856	don2@gmail.com	E2000	26	5.6	55	0	0	0	0	2019-05-25
D0003	Donar3	HSR	9156202360	don3@gmail.com	E3000	20	5.1	55	0	0	0	0	2019-12-25
D0004	Donar4	CV Raman	9236209876	don4@gmail.com	E1000	49	5.4	70	0	0	0	0	2019-08-10
D0005	Donar5	Vijayanagar	8156209876	don5@gmail.com	E1000	34	4.1	50	0	0	0	0	2019-04-01
D0006	Donar6	Rajajinagar	7156209876	don6@gmail.com	E2000	21	5.2	52	0	0	0	0	2019-09-01
D0007	Donar7	Basaveshwarnagar	9136209876	don7@gmail.com	E2000	25	5.7	55	0	0	0	0	2017-01-05
D0008	Donar8	Hebbal	9156293786	don8@gmail.com	E3000	55	5.3	63	1	0	0	0	2019-10-13
D0009	Donar9	BSK	9156209126	don9@gmail.com	E4000	23	5.5	65	1	1	0	0	2019-11-25
D0010	Donar10	BSK	8756209876	don10@gmail.com	E5000	45	5.5	65	0	0	0	0	2019-08-10
D0011	Donar11	MG Road	7123412345	don11@gmail.com	E9000	20	53	5.4	0	0	0	0	2017-05-01
D0012	Donar12	Whitefield	9012436781	don12@gmail.com	E9000	45	5.8	70	0	0	0	0	2018-04-04
D0013	Donar13	HSR	9123412345	don13@gmail.com	E6000	21	54	5.7	0	0	0	0	2016-08-08

TESTING LOSSLESS JOIN PROPERTY

The receiver table was decomposed into receiver and cost to achieve 3NF and remove transitive dependence of attribute cost on payment. This decomposition obeyed lossless join property.

```
SELECT * FROM receiver INNER JOIN cost ON cost.rec_id=receiver.rec_id AND cost.patient_id=receiver.patient_id
```

rec_id	patient_id	payment	received_on	emp_id	rec_id	patient_id	cost
D0001	P1100000	REPLACEMENT	2020-01-24	E1000	D0001	P1100000	0
D0001	P2100000	PAY	2018-01-02	E1000	D0001	P2100000	0
D0006	P1600000	REPLACEMENT	2019-01-14	E1000	D0006	P1600000	0
D0002	P1200000	REPLACEMENT	2020-01-01	E2000	D0002	P1200000	0
D0003	P1300000	REPLACEMENT	2020-01-21	E2000	D0003	P1300000	0
D0009	P1900000	REPLACEMENT	2018-06-30	E2000	D0009	P1900000	0
D0002	P1600000	PAY	2020-02-25	E3000	D0002	P1600000	0
D0008	P1800000	REPLACEMENT	2020-09-24	E5000	D0008	P1800000	0
D0005	P1500000	REPLACEMENT	2018-10-12	E5000	D0005	P1500000	0
D0007	P1700000	REPLACEMENT	2017-01-13	E6000	D0007	P1700000	0
D0004	P1400000	PAY	2020-02-03	E8000	D0004	P1400000	2000
D0010	P2100000	PAY	2017-02-01	E8000	D0010	P2100000	0

CHECK CONSTRAINTS

Check constraints were implemented for-

- To check if the donor's age is greater than 18.
- To check if the date on which a donor donates is more than 3 months since his last donation.
- To check if the blood group entered is valid.
- To check if the value entered for the health conditions are binary values.
- To check if the value for refill was binary.

```
create table stock(id INT PRIMARY KEY AUTO INCREMENT,blood_group VARCHAR(3),CHECK(blood_group in ("A+","A","B+","B","O+","O","AB+","AB-")),volume INT,PRIMARY KEY(blood_group,volume),refill INT ,CHECK(refill=0 or refill=1),cost FLOAT);
```

```
create table health(donar_id VARCHAR(5), FOREIGN KEY (donar_id) REFERENCES donar(donar_id) on DELETE cascade,smoking INT ,CHECK (smoking=0 or smoking=1), alcohol INT ,CHECK ( alcohol=0 or alcohol=1),cancer INT ,CHECK (cancer=0 or cancer=1),anemia INT ,CHECK (anemia=0 or anemia=1),surgery INT ,CHECK (surgery=0 or surgery=1),last_donated DATE, CHECK (last_donated<=NOW()),PRIMARY KEY(donar_id) );
```

```
create table donar(donar_id VARCHAR(5) PRIMARY KEY , donar_name VARCHAR(15) ,address VARCHAR(30),phone_no BIGINT NOT NULL UNIQUE,email_id VARCHAR(20) NOT NULL UNIQUE ,age INT NOT NULL,CHECK (age>=18),height FLOAT NOT NULL,weight FLOAT NOT NULL,emp_id VARCHAR(5),FOREIGN KEY (emp_id) REFERENCES employee(emp_id) ON DELETE SET NULL,blood_group VARCHAR(3) NOT NULL);
```


REFERENTIAL INTEGRITY CONSTRAINTS

Referential Integrity constraints were implemented for-

- If an employee's information is deleted, the value in donor and receiver table is set to **NULL**.
- If a donor's information is deleted, his information in the health table will also be deleted using "**delete cascade**".
- If a donor's information is deleted, the id in the table blood will be set to **NULL**.
- Most of donor's details like age, weight cannot be NULL using "**not null**" constraint.
- If the patient's info is deleted, then it will be deleted from receiver's table also using "**delete cascade**".

REFERENTIAL INTEGRITY CONSTRAINTS

```
create table blood(donar_id VARCHAR(5), FOREIGN KEY (donar_id) REFERENCES donar(donar_id) on DELETE set NULL,blood_group VARCHAR(3),FOREIGN KEY (blood_group) REFERENCES donar(blood_group) ,CHECK (blood_group in ('A+','A-','B+','B-','O+','O-','AB+','AB-'))),volume INT NOT NULL,donated_date DATE NOT NULL);
```

```
ALTER TABLE receiver ADD FOREIGN KEY(emp_id) REFERENCES employee(emp_id) ON DELETE SET NULL;
```


























```
create table donar(donar_id VARCHAR(5) PRIMARY KEY , donar_name VARCHAR(15) ,address VARCHAR(30),phone_no BIGINT NOT NULL UNIQUE,email_id VARCHAR(20) NOT NULL UNIQUE ,age INT NOT NULL,CHECK (age>=18),height FLOAT NOT NULL,weight FLOAT NOT NULL,emp_id VARCHAR(5),FOREIGN KEY (emp_id) REFERENCES employee(emp_id) ON DELETE SET NULL,blood_group VARCHAR(3) NOT NULL);
```

```
create table receiver(rec_id VARCHAR(5) NOT NULL ,FOREIGN KEY (rec_id) REFERENCES donar(donar_id) , patient_id VARCHAR(8) NOT NULL,FOREIGN KEY (patient_id) REFERENCES patient(patient_id) ON DELETE CASCADE,payment VARCHAR(15), received_on DATE ,CHECK (received_on<=NOW()),emp_id VARCHAR(5), FOREIGN KEY(emp_id) references employee(emp_id)ON DELETE SET NULL,PRIMARY KEY(rec_id,patient_id));
```

TRIGGERS

Triggers are used to-

- Implement check conditions because mysql-mariaDb version ignores check conditions.
- To update the last_donated date in the health table after the donar donates to the blood bank.
- To update the stock volume when a donar donates blood.
- To update the refill based on stock volume after a donar donates blood.

Triggers 					
Name	Table	Action			Time Event
<input type="checkbox"/> check_bloodgrp	blood	 Edit	 Export	 Drop	BEFORE INSERT
<input type="checkbox"/> check_age	donar	 Edit	 Export	 Drop	BEFORE INSERT
<input type="checkbox"/> check_blood_group	compatible	 Edit	 Export	 Drop	BEFORE INSERT
<input type="checkbox"/> check_date	health	 Edit	 Export	 Drop	BEFORE UPDATE
<input type="checkbox"/> update_stockvolume	blood	 Edit	 Export	 Drop	AFTER INSERT
<input type="checkbox"/> check_bg	patient	 Edit	 Export	 Drop	BEFORE INSERT
<input type="checkbox"/> check_last_donated	health	 Edit	 Export	 Drop	BEFORE INSERT
 <input type="checkbox"/> Check all With selected:  Export  Drop					

TRIGGERS

Edit trigger

Details

Trigger name

update_stockvolume

Table

blood

Time

AFTER

Event

INSERT

Definition

```
1 BEGIN
2
3     UPDATE stock,blood SET  stock.volume=stock.volume+new.volume WHERE
      stock.blood_group=new.blood_group;
4
5     UPDATE stock,blood SET  stock.refill=0 WHERE
      stock.blood_group=new.blood_group and stock.volume>=500;
6
7     UPDATE stock,blood SET  stock.refill=1 WHERE
      stock.blood_group=new.blood_group and stock.volume<500;
8     UPDATE health set health.last_donated=new.donated_date WHERE
      health.donar_id=new.donar_id;
9
10
11 END
```

Definer

root@localhost

Go

Close

COMPLEX QUERIES

Check if the stock volume is equal to the volume donated before any transaction.

- `SELECT stock.blood_group,stock.volume,blood.blood_group ,SUM(blood.volume) FROM stock inner join blood WHERE stock.blood_group=blood.blood_group GROUP by blood.blood_group`

blood_group	volume	blood_group	SUM(blood.volume)
A-	250	A-	250
A+	500	A+	500
AB-	350	AB-	350
AB+	650	AB+	650
B-	500	B-	500
B+	500	B+	500
O-	600	O-	600
O+	650	O+	650

Create a table of compatible blood type and display the number of units that can be given for the donor='P1400000'.

- `DROP TABLE temp;`
- `CREATE TABLE temp as SELECT blood_group,volume/250,cost from stock WHERE blood_group in (SELECT c FROM compatible WHERE blood_group in(SELECT blood_group FROM patient WHERE blood_group='B+' and patient.patient_id='P1400000'))ORDER BY volume DESC`

blood_group	volume/250	cost
O+	2.6000	500
O-	2.4000	1000
B-	2.0000	800
B+	2.0000	700

COMPLEX QUERIES

Display the amount of blood donated by each eligible donar.

- `SELECT donar.donar_id,donar.blood_group,SUM(blood.volume)
FROM blood,donar WHERE donar.donar_id IN (SELECT donar_id
FROM health WHERE smoking=0 and alcohol=0 and cancer=0 and
surgery=0 and anemia=0) AND blood.donar_id=donar.donar_id
GROUP BY donar.donar_id ORDER BY SUM(volume) DESC`

donar_id	blood_group	SUM(volume)
D0013	O-	600
D0011	AB+	500
D0012	B+	500
D0003	B-	700
D0006	A-	500
D0009	A+	500
D0005	O+	650
D0010	B-	150
D0004	AB+	150

Display the donars who have donated blood twice.

- `SELECT donar_id,blood_group,SUM(volume),COUNT(donar_id)
FROM blood GROUP BY donar_id HAVING COUNT(donar_id)=2`

donar_id	blood_group	SUM(volume)	COUNT(donar_id)
D0003	B-	700	2
D0005	O+	650	2
D0006	A-	500	2
D0009	A+	500	2
D0012	B+	800	2

COMPLEX QUERIES

Display the blood groups with the number of compatible blood groups in ascending order.

- `SELECT blood_group,count(c) FROM compatible group by (blood_group) ORDER BY count(c) ASC`

blood_group	count(c)	1
O-	1	
B-	2	
O+	2	
A-	2	
B+	4	
A+	4	
AB-	4	
AB+	8	

Display the bill for all the receivers who have opted the payment mode as “PAY”.

- `CREATE TABLE bill (bill_id INT PRIMARY KEY NOT NULL AUTO_INCREMENT,cost INT NOT NULL) SELECT R.rec_id, R.patient_id,P.h_pid,P.name,P.reason,P.blood_group,P.bottles,R.payment,R.received_on,R.emp_id FROM receiver as R JOIN patient as P ON R.patient_id=P.patient_id WHERE R.payment='PAY';`

bill_id	cost	rec_id	patient_id	h_pid	name	reason	blood_group	bottles	payment	received_on	emp_id
1	0	D0001	P2100000	S23456789	patient10	thalassemia	AB-	5	PAY	2018-01-02	E1000
2	0	D0002	P1600000	12347890	patient6	accident	O+	5	PAY	2020-02-25	E3000
3	0	D0004	P1400000	34567890	patient4	Anemia	B+	2	PAY	2019-11-02	E8000
4	0	D0010	P2100000	S23456789	patient10	thalassemia	AB-	5	PAY	2017-06-02	E8000

FRONT - END

[Home](#) [Database](#) [Employee](#) [Register](#) [Health](#) [Donate](#) [Orders](#) [Hospitals](#) [Patients](#) [Payment](#)

Blood-Bank Management System



Done by Amrutha U

FRONT - END

Home Database

STOCK

#	Blood Group	Volume	Refill	Cost
1	A-	500	0	800
2	A+	500	0	700
3	AB-	350	1	500
4	AB+	650	0	500
5	B-	500	0	800
6	B+	500	0	700
7	O-	100	1	1000
8	O+	250	1	500

Snapshot of
the database
from the
front-end.

FRONT - END

[Home](#) [Database](#)

PAYMENT

Receiver Id	<input type="text" value="D0002"/>
Patient Id	<input type="text" value="P1600000"/>
Payment Mode	<input type="text" value="PAY"/>
Date	<input type="text" value="2020:02:25"/>

CHECK

**Insert – Update
Receiver makes
the payment for
the patient who
has requirement
for blood.
(Both of them
must have
registered).**

FRONT - END

Home Database

BILL

#	Receiver Id	Receiver Name	Patient Id	H_Pid	Patient Name	Reason	Blood Group	Units	Cost	Date	Employee
1	D0002	Donar2	P1600000	12347890	patient6	accident	O+	1	500	2020-02-25	E3000

A bill is generated for the receiver.

FRONT - END

COST

#	Receiver ID	Patient ID	Cost
1	D0001	P1100000	0
2	D0001	P2100000	0
3	D0006	P1600000	0
4	D0002	P1200000	0
5	D0003	P1300000	0
6	D0009	P1900000	0
7	D0002	P1600000	500

The amount to be paid is updated in the COST table.

FRONT - END

STOCK				
#	Blood Group	Volume	Refill	Cost
1	A-	500	0	800
2	A+	500	0	700
3	AB-	350	1	500
4	AB+	650	0	500
5	B-	500	0	800
6	B+	500	0	700
7	O-	100	1	1000
8	O+	0	1	500

The stock is updated – the volume of O+ blood has become 0 which was initially 250, corresponding to the blood type and the amount of units that was given.

FRONT - END

12	D0012	Donar12	Whitefield	9012436781	don12@gmail.com	45	5.8	70	B+
----	-------	---------	------------	------------	-----------------	----	-----	----	----

12	D0012	0	0	0	0	0	0	2019-04-04
----	-------	---	---	---	---	---	---	------------

[Home](#) [Database](#) [Register](#) [Health](#)

DONATE

Donar Id

D0012

Blood Group

B+

Volume

300

Date

2019:04:04

ADD

A donor registered with the Id D0012 and has no health complications donates 300mL of B+ blood.

FRONT - END

12	D0012	0	0	0	0	0	2019-04-04
----	-------	---	---	---	---	---	------------

STOCK

#	Blood Group	Volume	Refill	Cost
1	A-	500	0	800
2	A+	500	0	700
3	AB-	350	1	500
4	AB+	650	0	500
5	B-	500	0	800
6	B+	800	0	700
7	O-	100	1	1000
8	O+	0	1	500

The donor's last_donated is updated to the entered date. The volume of B+ blood in the stock is increased by 300.(Refer previous stock table to view the update).

FRONT - END

A donar with Id D0001 cannot donate as the person smokes.

Donar_Health							
#	Donar Id	Smoking	Drinking	Cancer Treatment	Anemia	Operated Recently	Last Donation
1	D0001	1	0	0	0	0	2019-05-01
2	D0002	0	0	0	0	0	2019-05-25
3	D0003	0	0	0	0	0	2019-12-25
4	D0004	0	0	0	0	0	2019-08-10

DONATE

Donar Id

D0001

Blood Group

A+

Volume

100

Date

2020:03:03

ADD

localhost says

You are not eligible to donate!

OK

FRONT - END

Registering a new donar

donar_id	donar_name	address	phone_no	email_id	emp_id	blood_group	age	height	weight
D0001	Donar1	BSK	9156209876	don1@gmail.com	E1000	A+	25	5.5	65
D0002	Donar2	INR	9156209856	don2@gmail.com	E2000	B+	26	5.6	55
D0003	Donar3	HSR	9156202360	don3@gmail.com	E3000	B-	20	5.1	55
D0004	Donar4	CV Raman	9236209876	don4@gmail.com	E1000	AB+	49	5.4	70
D0005	Donar5	Vijayanagar	8156209876	don5@gmail.com	E1000	O+	34	4.1	50
D0006	Donar6	Rajajinagar	7156209876	don6@gmail.com	E2000	A-	21	5.2	52
D0007	Donar7	Basaveshwarnagar	9136209876	don7@gmail.com	E2000	O-	25	5.7	55
D0008	Donar8	Hebbal	9156293786	don8@gmail.com	E3000	AB-	55	5.3	63
D0009	Donar9	BSK	9156209126	don9@gmail.com	E4000	A+	23	5.5	65
D0010	Donar10	BSK	8756209876	don10@gmail.com	E5000	B-	45	5.5	65
D0011	Donar11	MG Road	7123412345	don11@gmail.com	E9000	AB+	20	53	5.4
D0012	Donar12	Whitefield	9012436781	don12@gmail.com	E9000	B+	45	5.8	70
D0013	Donar13	HSR	9123412345	don13@gmail.com	E6000	O-	21	54	5.7
D0014	Donar14	Vijayanagar	4567812345	don14@gmail.com	E2000	AB-	24	6.1	81

donar_id	donar_name	address	phone_no	email_id	emp_id	blood_group	age	height	weight
D0001	Donar1	BSK	9156209876	don1@gmail.com	E1000	A+	25	5.5	65
D0002	Donar2	INR	9156209856	don2@gmail.com	E2000	B+	26	5.6	55
D0003	Donar3	HSR	9156202360	don3@gmail.com	E3000	B-	20	5.1	55
D0004	Donar4	CV Raman	9236209876	don4@gmail.com	E1000	AB+	49	5.4	70
D0005	Donar5	Vijayanagar	8156209876	don5@gmail.com	E1000	O+	34	4.1	50
D0006	Donar6	Rajajinagar	7156209876	don6@gmail.com	E2000	A-	21	5.2	52
D0007	Donar7	Basaveshwarnagar	9136209876	don7@gmail.com	E2000	O-	25	5.7	55
D0008	Donar8	Hebbal	9156293786	don8@gmail.com	E3000	AB-	55	5.3	63
D0009	Donar9	BSK	9156209126	don9@gmail.com	E4000	A+	23	5.5	65
D0010	Donar10	BSK	8756209876	don10@gmail.com	E5000	B-	45	5.5	65
D0011	Donar11	MG Road	7123412345	don11@gmail.com	E9000	AB+	20	53	5.4
D0012	Donar12	Whitefield	9012436781	don12@gmail.com	E9000	B+	45	5.8	70
D0013	Donar13	HSR	9123412345	don13@gmail.com	E6000	O-	21	54	5.7
D0014	Donar14	Vijayanagar	4567812345	don14@gmail.com	E2000	AB-	24	6.1	81
D0015	Donar15	ITPL	6754390234	don15@gmail.com	E2000	O+	45	5.8	70

NEW DONAR

Donar Id	<input type="text" value="D0015"/>
Name	<input type="text" value="Donar15"/>
Address	<input type="text" value="ITPL"/>
Phone No	<input type="text" value="6754390234"/>
Email ID	<input type="text" value="don15@gmail.com"/>
Age	<input type="text" value="45"/>
Weight	<input type="text" value="70"/>
Height	<input type="text" value="5.8"/>
Blood Group	<input type="text" value="O+"/>
Emp ID	<input type="text" value="E2000"/>

ADD

FRONT - END

Deleting the Employee with Id E9000

DELETE FROM employee WHERE emp_id='E9000';

emp_id	e_name	address	phone_no	email_id
E1000	Emp1	BSK	1234567890	emp1@gmail.com
E1100	Emp11	Frazer Town	1098234567	emp11@gmail.com
E2000	Emp2	HSR	1287612344	emp2@gmail.com
E3000	Emp3	INR	1287612343	emp3@gmail.com
E4000	Emp4	RPC Layout	1227612345	emp4@gmail.com
E5000	Emp5	Jalahalli	1281112345	emp5@gmail.com
E6000	Emp7	Peenya	9812342678	emp7@gmail.com
E7000	Emp8	RT Nagar	9812309678	emp8@gmail.com
E8000	Emp9	Malleshwaram	9811145678	emp9@gmail.com
E9000	Emp10	MG Road	9002135678	emp10@gmail.com
M1000	Emp6	Hebbal	9812345678	emp6@gmail.com

donar_id	donar_name	address	phone_no	email_id	emp_id	blood_group	age	height	weight
D0001	Donar1	BSK	9156209876	don1@gmail.com	E1000	A+	25	5.5	65
D0002	Donar2	INR	9156209856	don2@gmail.com	E2000	B+	26	5.6	55
D0003	Donar3	HSR	9156202360	don3@gmail.com	E3000	B-	20	5.1	55
D0004	Donar4	CV Raman	9236209876	don4@gmail.com	E1000	AB+	49	5.4	70
D0005	Donar5	Vijayanagar	8156209876	don5@gmail.com	E1000	O+	34	4.1	50
D0006	Donar6	Rajajinagar	7156209876	don6@gmail.com	E2000	A-	21	5.2	52
D0007	Donar7	Basaveshwarnagar	9136209876	don7@gmail.com	E2000	O-	25	5.7	55
D0008	Donar8	Hebbal	9156293786	don8@gmail.com	E3000	AB-	55	5.3	63
D0009	Donar9	BSK	9156209126	don9@gmail.com	E4000	A+	23	5.5	65
D0010	Donar10	BSK	8756209876	don10@gmail.com	E5000	B-	45	5.5	65
D0011	Donar11	MG Road	7123412345	don11@gmail.com	E9000	AB+	20	53	5.4
D0012	Donar12	Whitefield	9012436781	don12@gmail.com	E9000	B+	45	5.8	70
D0013	Donar13	HSR	9123412345	don13@gmail.com	E6000	O-	21	54	5.7
D0014	Donar14	Vijayanagar	4567812345	don14@gmail.com	E2000	AB-	24	6.1	81

FRONT - END

Employee Id “emp_id” in donar table is set to “NULL” using referential integrity constrains.

emp_id	e_name	address	phone_no	email_id
E1000	Emp1	BSK	1234567890	emp1@gmail.com
E1100	Emp11	Frazer Town	1098234567	emp11@gmail.com
E2000	Emp2	HSR	1287612344	emp2@gmail.com
E3000	Emp3	INR	1287612343	emp3@gmail.com
E4000	Emp4	RPC Layout	1227612345	emp4@gmail.com
E5000	Emp5	Jalahalli	1281112345	emp5@gmail.com
E6000	Emp7	Peenya	9812342678	emp7@gmail.com
E7000	Emp8	RT Nagar	9812309678	emp8@gmail.com
E8000	Emp9	Malleshwaram	9811145678	emp9@gmail.com
M1000	Emp6	Hebbal	9812345678	emp6@gmail.com

donar_id	donar_name	address	phone_no	email_id	emp_id	blood_group	age	height	weight
D0001	Donar1	BSK	9156209876	don1@gmail.com	E1000	A+	25	5.5	65
D0002	Donar2	INR	9156209856	don2@gmail.com	E2000	B+	26	5.6	55
D0003	Donar3	HSR	9156202360	don3@gmail.com	E3000	B-	20	5.1	55
D0004	Donar4	CV Raman	9236209876	don4@gmail.com	E1000	AB+	49	5.4	70
D0005	Donar5	Vijayanagar	8156209876	don5@gmail.com	E1000	O+	34	4.1	50
D0006	Donar6	Rajajinagar	7156209876	don6@gmail.com	E2000	A-	21	5.2	52
D0007	Donar7	Basaveshwarnagar	9136209876	don7@gmail.com	E2000	O-	25	5.7	55
D0008	Donar8	Hebbal	9156293786	don8@gmail.com	E3000	AB-	55	5.3	63
D0009	Donar9	BSK	9156209126	don9@gmail.com	E4000	A+	23	5.5	65
D0010	Donar10	BSK	8756209876	don10@gmail.com	E5000	B-	45	5.5	65
D0011	Donar11	MG Road	7123412345	don11@gmail.com	NULL	AB+	20	53	5.4
D0012	Donar12	Whitefield	9012436781	don12@gmail.com	NULL	B+	45	5.8	70
D0013	Donar13	HSR	9123412345	don13@gmail.com	E6000	O-	21	54	5.7
D0014	Donar14	Vijayanagar	4567812345	don14@gmail.com	E2000	AB-	24	6.1	81
D0015	Donar15	ITPL	6754390234	don15@gmail.com	E2000	O+	45	5.8	70

CONCLUSION

The current project is a smaller version of the real world blood bank. It can be made scalable to a real world scenario and implemented by adding more features like branches, blood donation camps, connections with hospitals.