Blood Donation Record Database for

American Red Cross

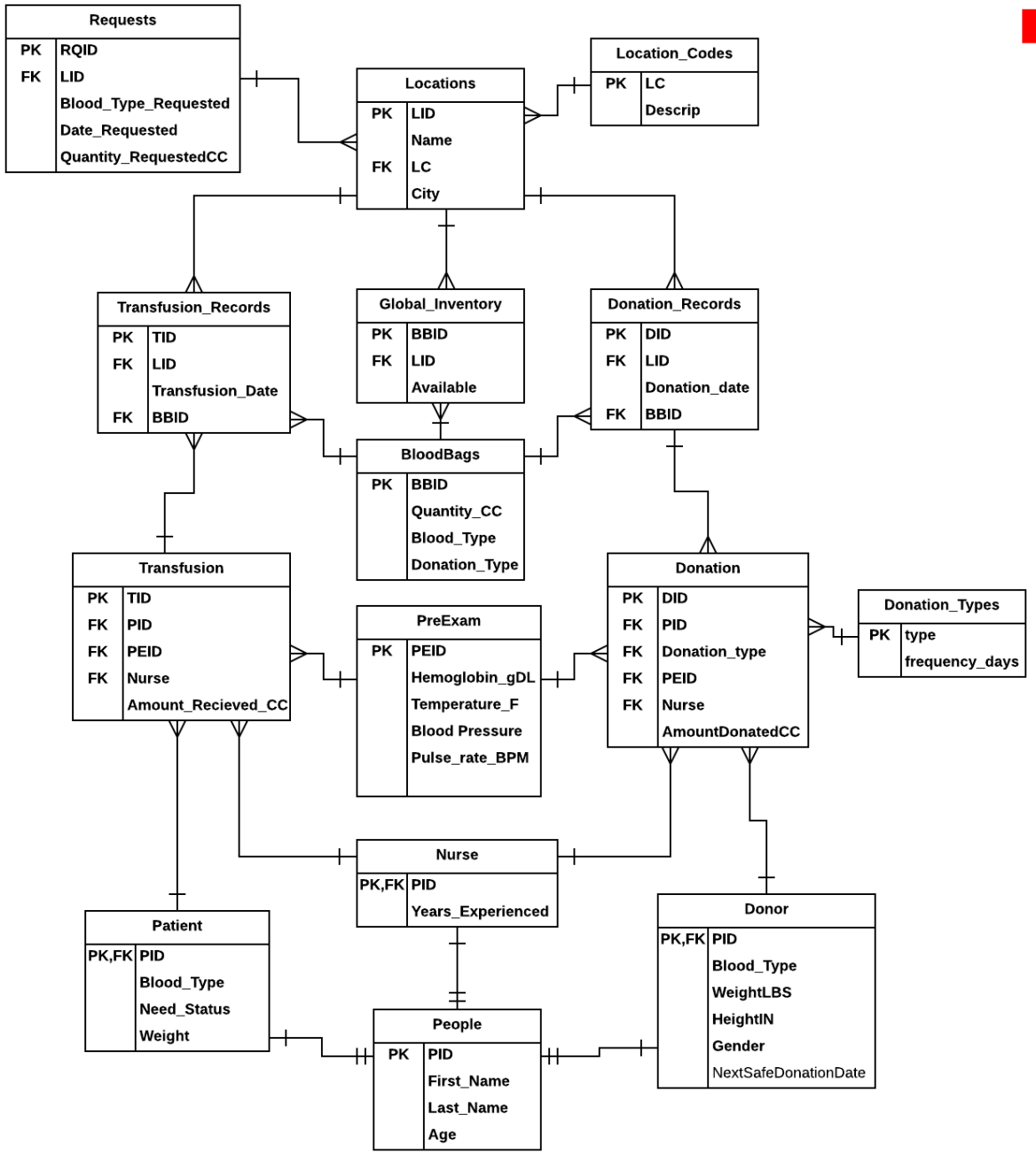
BRIAN HENDERSON

Table of Contents

Executive Summary

This document outlines the design of a database to hold all the data for the American Red Cross in regards to their blood donation division. The American Red Cross is the leading blood donation organization in the world. Distributing to about 2,600 hospitals and healthcare facilities in the United States alone, the American Red Cross collects and processes roughly 40% of the nation's blood supply. The design of this database is to show the framework for the amount of data that the American Red Cross comes across, as well as to serve as a historical reference. This database holds all the information required for each donation/transfusion, including the required pre-exam, a global inventory to show the inventory stocks across all locations, which can also be queried to narrow down to the specific location. The data implemented into this database is fictional, with some exceptions. All persons, pre-exams, records, and some locations are fictional. This database is designed to hold large scale data. The ultimate objective is to design a database that is not only fully functional, but also fully normalized in third normal form that can help serve the American Red Cross for their blood donation records.

E/R DIAGRAM


Requests
PK RQID
FK LID
Blood_Type_Requested
Date_Requested
Quantity_RequestedCC
Locations
PK LID
Name
FK LC
City
Location_Codes
PK LC
Descrip
Transfusion_Records
PK TID
FK LID
Transfusion_Date
FK BBID
Global_Inventory
PK BBID
FK LID
Available
Donation_Records
PK DID
FK LID
Donation_date
FK BBID
BloodBags
PK BBID
Quantity_CC
Blood_Type
Donation_Type
Transfusion
PK TID
FK PID
FK PEID
FK Nurse
Amount_Recieved_CC
PreExam
PK PEID
Hemoglobin_gDL
Temperature_F
Blood Pressure
Pulse_rate_BPM
Donation
PK DID
FK PID
FK Donation_type
FK PEID
FK Nurse
AmountDonatedCC
Donation_Types
PK type
frequency_days
Nurse
PK,FK PID
Years_Experienced
Patient
PK,FK PID
Blood_Type
Need_Status
Weight
People
PK PID
First_Name
Last_Name
Age
Donor
PK,FK PID
Blood_Type
WeightLBS
HeightIN
Gender
NextSafeDonationDate

Persons Table

The **people** table contains all the people and their common attributes. There are three subtypes for the people table: patient, donor, and nurse.

```
CREATE TABLE persons (
       pid          char(8)      not null unique,
       first_name   text         not null,
       last_name    text         not null,
       age          integer      not null,
    primary key(pid)
);
```

Functional Dependencies

Pid → first_name, last_name, age

Sample Data

pid character	first_name text	last_name text	age integer
p1	John	Centra	23
p2	Anne	Parker	42
p3	Ryan	Fowler	32
p4	Peter	Cruz	21
p5	Stephanie	Collins	46
p6	Harry	Bryant	62
p7	James	Bond	40
p8	David	King	34
p9	Thomas	Tank	24
p10	Jessica	Roberts	42
p11	Carol	Porter	62
p12	Emily	Murphy	21
p13	Matthew	McDerm...	37
p14	Joseph	Ruggiero	19
p15	Hannah	Taylor	17
p16	Anne	Miller	48
p17	Caroline	Powers	38
p18	Emily	McFelice	40
p19	Pattricia	Wilson	39
p20	Bill	Bowerman	26
p21	Mickey	Globe	23

Donor Table

The **donor** table contains the information required to be a donor. Blood and Platelet donors must be 110lbs and 17 years of age. Plasma donors have other requirements.

```
CREATE TABLE donor (
        pid             char(8)      not null references  persons(pid),
        blood_type      char(3)      not null,
        weightLBS       integer      not null,
        heightIN        integer      not null,
        gender          char(1)      not null,
        nextSafeDonation DATE,
    CONSTRAINT check_gender CHECK (gender = 'M' OR gender = 'F'),
    primary key(pid)
);
```

Functional Dependencies

Pid → blood_type, weightLBS, heightIN, gender

Constraints

check_gender → Checks gender input is 'M' or 'F'

Sample Data

pid character	blood_type character	weightlbs integer	heightin integer	gender character	nextsafedonation date
p4	O+	149	70	M	
p7	O-	170	74	M	
p8	AB-	148	66	M	
p9	B-	180	71	M	
p13	O+	212	76	M	
p14	B+	128	68	M	
p15	O+	104	63	F	
p20	A+	178	65	M	
p21	AB+	120	57	F	

Patient Table

The **patient** table contains all the patients and their information required before a blood transfusion. The *need status* field indicates whether their require blood on a high priority or a low priority.

```
CREATE TABLE patient (
        pid          char(8)      not null references persons(pid),
        blood_type   char(3)      not null,
        need_status  text         not null,
        weightLBS    integer      not null,
    CONSTRAINT check_status CHECK (need_status = 'high' OR need_status = 'low'),
    primary key(pid)
);
```

Sample Data

pid character	blood_type character	need_status text	weightlbs integer
p1	O+	low	172
p3	A+	low	185
p6	AB+	high	128
p11	B+	low	120
p12	A+	low	118

Functional Dependencies

Pid → blood_type, need_status, weightLBS

Constraints

chech_status → Checks need status input to be either ‘high’ or ‘low’

Nurse Table

The **nurse** table contains all the nurses, with the years of experience they have.

```
CREATE TABLE nurse (
        pid                  char(8)     not null references persons(pid),
        years_experienced    integer     not null,
    primary key(pid)
);
```

Sample Data

pid character	years_experienced integer
p2	12
p4	17
p10	5
p16	18
p17	9
p18	10
p19	12

Functional Dependencies

Pid → years_experienced

Pre Exam Table

The **pre_exam** table contains the respective information about a donor before a donation, as well as a patient before a transfusion.

```
CREATE TABLE pre_exam (
        peid              char(8)          not null,
        hemoglobin_gDL    decimal(5,2)     not null,
        temperature_F     decimal(5,2)     not null,
        blood_pressure    char(8)          not null,
        pulse_rate_BPM    integer          not null,
    primary key(peid)
);
```

Functional Dependencies

peid → hemoglobin_gDL, temperature_F, blood_pressure, pulse_rate_BMP

Sample Data

peid character	hemoglobin_gdl numeric (5,2)	temperature_f numeric (5,2)	blood_pressure character	pulse_rate_bpm integer
pe1	15.2	98.6	120/80	70
pe2	14.9	98.5	110/70	75
pe3	15.7	98.5	130/85	59
pe4	16.1	98.4	120/80	67
pe5	14.2	98.3	90/80	90
pe6	17.1	98.2	110/70	44
pe7	14.2	98.1	140/90	79
pe8	7.1	98.9	90/60	65
pe9	8	98.6	130/85	80
pe10	7.9	98.7	120/80	82
pe11	7.6	98.4	90/60	76
pe12	6.9	98.5	120/80	70
pe13	14.5	98.3	120/80	70
pe14	15.3	98.4	110/70	77
pe15	14.3	98.3	120/80	63
pe16	13.9	98.3	110/70	81
pe17	16.4	98.8	120/80	72
pe18	17.1	98.1	120/80	59

Donation Table

The **donation** table contains the basic attributes about a blood donation. The donation type references the type of blood donation.

```
CREATE TABLE donation (
        did                 char(8)       not null,
        pid                 char(8)       not null references donor(pid),
        peid                char(8)       not null references pre_exam(peid),
        nurse               char(8)       not null references nurse(pid),
        amount_donated_CC   decimal(5,2)  not null,
        donation_type       text          not null references donation_types(type),
    primary key(did)
);
```

Functional Dependencies

peid → pid, peid, nurse,
amount_donated_CC, donation_type

Sample Data

did character	pid character	peid character	nurse character	amount_donated_cc numeric (5,2)	donation... text
d1	p4	pe1	p16	946	Power Red
d2	p7	pe2	p17	473	Blood
d3	p8	pe3	p18	473	Plasma
d4	p9	pe4	p19	473	Blood
d5	p13	pe5	p16	473	Platelets
d6	p14	pe6	p16	473	Blood
d7	p15	pe7	p16	473	Blood
d8	p4	pe13	p16	473	Blood
d9	p14	pe14	p16	473	Blood
d10	p20	pe15	p17	473	Blood
d11	p15	pe16	p18	473	Blood
d12	p21	pe17	p19	473	Blood
d13	p20	pe18	p16	473	Blood

Donation Types Table

The **donation_type** table contains the four different types of blood donation types, as well as the frequency/wait time in which the donor must wait before donating that type again.

```
CREATE TABLE donation_types (
        type             text      not null unique,
        frequency_days   integer   not null,
    primary key(type)
);
```

Sample Data

type text	frequency_days integer
Blood	56
Platelets	7
Plasma	28
Power Red	112

Functional Dependencies

type → frequency_days

Transfusion Table

The **transfusion** table contains the basic attributes about a blood transfusion.

```
CREATE TABLE transfusion (
        tid                  char(8)      not null,
        pid                  char(8)      not null references patient(pid),
        peid                 char(8)      not null references pre_exam(peid),
        nurse                char(8)      not null references nurse(pid),
        amount_recieved_CC decimal(5,2)   not null,
    primary key(tid)
);
```

Sample Data

tid character	pid character	peid character	nurse character	amount_recieved_cc numeric (5,2)
t1	p1	pe8	p2	473
t2	p3	pe9	p4	946
t3	p6	pe10	p2	473
t4	p11	pe11	p10	716
t5	p12	pe12	p4	473

Functional Dependencies

tid → pid, peid, nurse, amount_recieved_CC

Blood Bags Table

The **bloodbags** table contains the basic attributes about each blood bag. Each blood bag is labeled with: the blood type, the quantity, and the type of donation.

```
CREATE TABLE bloodbags (
        bbid              char(10)       not null unique,
        donation_type     text           not null references donation_types(type),
        quantity_CC       decimal(5,2)   not null,
        blood_type        char(3)        not null,
    primary key(bbid)
);
```

Functional Dependencies

bbid → quantity_CC, blood_type, donation_type

Sample Data

bbid character	quantity_cc numeric (5...	blood_type character	donation_type text
bb1	473	O+	Blood
bb2	473	O-	Blood
bb3	946	O+	Power Red
bb4	473	AB-	Plasma
bb5	473	B-	Blood
bb6	473	B+	Blood
bb7	473	O+	Platelets
bb8	473	A+	Blood
bb9	473	AB+	Blood
bb10	473	A+	Blood
bb11	473	B+	Blood
bb12	473	O+	Blood
bb13	473	O+	Blood

Locations Table

The **locations** table contains all the locations, as well as a code to describe the type of location.

```
CREATE TABLE locations (
        lid     char(6)     not null unique,
        name    text        not null,
        lc      char(4)     not null references location_codes(lc),
        city    text        not null,
   primary key(lid)
);
```

Functional Dependencies

lid → name, lc, city

Sample Data

lid character	name text	lc character	city text
L1	Mid Hudson Regional Hospital	HSPT	Poughkeepsie
L2	Vassar Brothers Medical Center	CLIN	Vassar
L3	Marist College	BDCO	Poughkeepsie
L4	Fort Monmouth	MILT	Eatontown
L5	American Red Cross Eastern New York Chapter	ARCF	Albany
L6	Ramsey High School	BDHS	Ramsey
L7	Charlesville Emergency Clinic	CLIN	Chatstown
L8	IBM	BDOG	Poughkeepsie
L9	Poughkeepsie Galleria	BDOG	Poughkeepsie
L10	American Red Cross New York City Chapter	ARCF	New York

Location Codes Table

The **location_codes** table contains a four character code describing the type of location. Blood Drives are indicated with a BD in front.

```
CREATE TABLE location_codes (
        lc        char(4)      not null unique,
        descrip   text         not null,
   primary key(lc)
);
```

Functional Dependencies

lc→ descrip

Sample Data

lc character	descrip text
ARCF	American Red Cross Facility
BDHS	Blood Drive - High School
BDUN	Blood Drive - University
BDCO	Blood Drive - College
BDOG	Blood Drive - Orginization
MILT	Military Facility
CLIN	Clinics
HSPT	Hospital
RESR	Research Facility

Global Inventory Table

The **global_inventory** table contains the a global inventory of all blood bags with the location in which they are stored.

```
CREATE TABLE global_inventory (
        bbid        char(10)    not null references bloodbags(bbid),
        lid         char(6)     not null references locations(lid),
        available   boolean     DEFAULT TRUE,
    primary key (bbid,lid)
);
```

Functional Dependencies

bbid , lid → available

Sample Data

bbid character	lid character	available boolean
bb2	L5	true
bb3	L10	true
bb4	L5	true
bb5	L10	true
bb7	L5	true
bb11	L5	true
bb12	L10	true
bb13	L10	true
bb1	L5	false
bb8	L5	false
bb9	L10	false
bb6	L10	false
bb10	L10	false

Requests Table

The **requests** table contains attributes describing a "request" from a location. Locations can request blood from American Red Cross.

```
CREATE TABLE requests (
        rqid                    char(8)     not null unique,
        lid                     char(6)     not null references locations(lid),
        blood_type_requested    text        not null,
        date_requested          DATE        not null,
        quantity_requestedPints integer     not null,
    primary key(rqid)
);
```

Sample Data

rqid character	lid character	blood_ty... text	date_requ... date	quantity... integer
rq1	L1	A+	2016-12-08	10000
rq2	L4	A-	2016-12-08	9000
rq3	L1	A-	2016-12-08	3600
rq4	L2	O-	2016-12-08	13000
rq5	L1	AB-	2016-12-08	7000
rq6	L1	B-	2016-12-08	6000
rq7	L7	O-	2016-12-08	12000

Functional Dependencies

rqid → lid, blood_type_requested, date_requested, quantity_requestedPints

Donation Records Table

The **donation_records** table provides a more detailed record of all the donations.

```
CREATE TABLE donation_records (
        did              char(8)      not null references donation(did),
        lid              char(4)      not null references locations(lid),
        donation_date    DATE         not null,
        bbid             char(10)     not null references bloodbags(bbid),
    primary key(did)
);
```

Functional Dependencies

did → lid, donation_date, bbid

Sample Data

did character	lid character	donation_d... date	bbid character
d1	L3	2016-07-01	bb3
d2	L6	2016-07-30	bb2
d3	L8	2016-08-14	bb4
d4	L9	2016-09-18	bb5
d5	L3	2016-08-12	bb7
d6	L3	2016-12-01	bb11
d7	L3	2016-08-18	bb12
d8	L3	2016-09-03	bb13
d9	L3	2016-09-03	bb1
d10	L6	2016-08-13	bb6
d11	L6	2016-07-08	bb8
d12	L8	2016-11-20	bb9
d13	L3	2016-12-06	bb10

Transfusion Records Table

The **transfusion_records** table provides a more detailed record of all the transfusions.

```
CREATE TABLE transfusion_records (
        tid                 char(8)     not null references transfusion(tid),
        lid                 char(4)     not null references locations(lid),
        transfusion_date    date        not null,
        bbid                char(10)    not null references bloodbags(bbid),
    primary key(tid)
);
```

Sample Data

tid character	lid character	transfusion_date date	bbid character
t1	L1	2016-09-10	bb1
t2	L5	2016-11-13	bb8
t3	L1	2016-11-21	bb9
t4	L10	2016-12-02	bb6
t5	L5	2016-12-05	bb10

Functional Dependencies

tid → lid, transfusion_date, bbid

Stored Procedures get_persons_donation_records

The **get_persons_donation_records** stored procedure can be used to look up all the donation records for a donor by passing through the persons '*pid*'.

```
CREATE OR REPLACE FUNCTION get_persons_donation_records (char(8), REFCURSOR) returns refcursor as
$$
DECLARE
   personID   char(8)     := $1;
   results    REFCURSOR   := $2;
BEGIN
   OPEN results FOR
       SELECT dr.did, dr.lid, dr.donation_date, d.pid, d.peid,
              d.nurse, d.amount_donated_CC, d.donation_type
       FROM donation_records dr INNER JOIN donation d ON dr.did = d.did
       WHERE personID = d.pid;
   RETURN results;
END;
$$
language plpgsql;
```

Sample Output

```
select get_persons_donation_records('p4', 'results');
fetch all from results;
```

did character	lid character	donation... date	pid character	peid character	nurse character	amount_... numeric ...	donation... text
d1	L3	2016-07-...	p4	pe1	p16	946	Power Red
d8	L3	2016-09-...	p4	pe13	p16	473	Blood

Stored Procedures get_blood_type_inventory_percentage

The **get_blood_type_inventory_percentage** stored procedure can be used to look up how much of the global inventory that is available for use is a the passed in blood type.

```
CREATE OR REPLACE FUNCTION get_blood_type_inventory_percentage (char(3), REFCURSOR) returns refcursor as
$$
DECLARE
    reqType    char(3)    := $1;
    results    REFCURSOR  := $2;
BEGIN
OPEN results FOR
    SELECT TRUNC (
      CAST (
         ( SELECT COUNT(gi.bbid) AS selectedBBID
           FROM global_inventory gi INNER JOIN bloodbags bb ON gi.bbid = bb.bbid
           WHERE bb.blood_type = reqType
             AND gi.available = TRUE
         ) as decimal(5,2)
        )
         /
        ( SELECT COUNT(gi.bbid) AS allBBIDs
          FROM global_inventory gi
          WHERE gi.available = TRUE
        )
        * 100 )  AS BloodTypePercentage;
RETURN results;
END;
$$
language plpgsql;
```

Sample Output

```
select get_blood_type_inventory_percentage('B-', 'results');
fetch all from results;
```

bloodtypepercentage numeric
12

Stored Procedures update_inventory_status

The **update_inventory_statys** stored procedure is used to update the blood bags inventory when a blood bag is used for transfusion.

```
CREATE OR REPLACE FUNCTION update_inventory_status()
RETURNS TRIGGER AS
$$
BEGIN
   IF NEW.bbid is NOT NULL THEN
      UPDATE global_inventory
      SET available = FALSE
      WHERE NEW.bbid = global_inventory.bbid;
   END IF;
RETURN NEW;
END;
$$
LANGUAGE PLPGSQL;
```

Stored Procedure update_next_donation_date

Update_next_donation_date procedure is intended to, based of of the type of donation give, mark the next time it would be safe to donated blood from reference to the donation types table, where frequency is the amount of days needed to wait.

```
CREATE OR REPLACE FUNCTION update_next_donation_date()
RETURNS TRIGGER AS
$$
DECLARE
   waitDays integer;
   donDate DATE := NEW.donation_date;
   selectedDid char(8) := NEW.did;
BEGIN
   IF selectedDid IS NOT NULL THEN
      SELECT dt.frequency_days INTO waitDays
      FROM donation d INNER JOIN donation_types dt ON d.donation_type = dt.type
                      INNER JOIN donation_records dr ON d.did = dr.did
      WHERE dr.did = selectedDid;

  -- update next safe donation date
      UPDATE donor
        SET nextSafeDonation = donDate + waitDays
        WHERE donor.pid IN ( SELECT donor.pid
                             FROM donation d INNER JOIN donation_records dr ON d.did = dr.did
                             INNER JOIN donor ON donor.pid = d.pid
                             WHERE d.did = selectedDid
        );
      END IF;
RETURN NEW;
END;
$$
LANGUAGE PLPGSQL;
```

Trigger update_inventory_status()

When a new blood transfusion record is inserted, the trigger is called to set the blood bag used in the transfusion to be '*FALSE*'

```
CREATE TRIGGER update_inventory_status_trigger
BEFORE INSERT ON transfusion_records
FOR EACH ROW
EXECUTE PROCEDURE update_inventory_status();
```

Trigger Update_next_donation_date_trigger

When a new donation record is inserted, this trigger is called to mark the date that the donor could next donate blood safely.

```
CREATE TRIGGER update_next_safe_donation_date_trigger
BEFORE INSERT ON donation_records
FOR EACH ROW
EXECUTE PROCEDURE update_inventory_status();
```

Views AvailableBloodBags

The **AvailableBloodBags** view contains all the bloodbags that have not been used yet. This can be used to track the inventory of blood bags available.

```
DROP VIEW IF EXISTS AvailableBloodBags;
CREATE VIEW AvailableBloodBags as (
SELECT   gi.bbid,
         gi.lid,
         bb.blood_type,
         bb.donation_type,
         bb.quantity_CC
FROM global_inventory gi INNER JOIN bloodbags bb
         ON gi.bbid = bb.bbid
WHERE gi.available = TRUE
);
```

Sample Output

```
SELECT *
FROM AvailableBloodBags
WHERE blood_type = 'O+';
```

bbid character	lid character	blood_ty... character	donation... text	quantity... numeric ...
bb3	L10	O+	Power Red	946
bb7	L5	O+	Platelets	473
bb12	L10	O+	Blood	473
bb13	L10	O+	Blood	473

The example above narrows down blood bag information to display all the available blood bags for blood type O+.

Views locationInventories.

Locations can check their inventory supply by using the locationInvetories view, and based off of that report, they can make necessary determinations on if the location should request more stock, and how much.

```
DROP VIEW IF EXISTS locationInventories;
CREATE VIEW locationInventories AS (
SELECT gi.lid,
       SUM(bb.quantity_CC) AS totQuantity,
       bb.blood_type,
       bb.donation_type
FROM global_inventory gi INNER JOIN bloodbags bb ON gi.bbid = bb.bbid
                         INNER JOIN locations l  ON gi.lid  = l.lid
GROUP BY blood_type,
         donation_type,
         gi.lid
ORDER BY lid desc,
         totquantity desc
);
```

Sample Output

```
SELECT *
FROM locationInventories
WHERE lid = 'L10';
```

lid character	totquantity numeric	blood_type character	donation_type text
L10	946	O+	Blood
L10	946	O+	Power Red
L10	473	A+	Blood
L10	473	AB+	Blood
L10	473	B-	Blood
L10	473	B+	Blood

The example above shows the current inventory of the location desired, in this case *L10*

Reports / Interesting Query 1

Return to query the total times a donor has donated and the total amount of blood they have donated in units of CC. Orders by the total amount in descending order.

```
SELECT p.pid,
       p.first_name,
       p.last_name,
       COUNT(d.pid) AS TimesDonated,
       SUM(d.amount_donated_CC) AS TotalAmount
FROM persons p INNER JOIN donation d ON p.pid = d.pid
GROUP BY p.pid
ORDER by TotalAmount desc;
```

Sample Output

pid character	first_name text	last_name text	timesdo... bigint	totamount numeric
p4	Peter	Cruz	2	1419
p15	Hannah	Taylor	2	946
p20	Bill	Bowerman	2	946
p14	Joseph	Ruggiero	2	946
p13	Matthew	McDerm...	1	473
p8	David	King	1	473
p9	Thomas	Tank	1	473
p21	Mickey	Globe	1	473
p7	James	Bond	1	473

Reports / Interesting Query 2

Return to query the total amount of each blood type and donation type in the global inventory. Ordered by the total quantity descending.

```
SELECT bb.blood_type,
       bb.donation_type,
       SUM(quantity_cc) AS totQuantity
FROM bloodbags bb INNER JOIN global_inventory gi ON bb.bbid = gi.bbid
GROUP BY bb.blood_type,
         bb.donation_type
ORDER BY totQuantity desc;
```

Sample Output

blood_ty... character	donation... text	totquant... numeric
O+	Blood	1419
B+	Blood	946
O+	Power Red	946
A+	Blood	946
B-	Blood	473
O+	Platelets	473
O-	Blood	473
AB-	Plasma	473
AB+	Blood	473

Security Admin, Register, Requester

Admin - Database Administrator has full control over the DB.
Register – This role is intended for the person registering the data for each donation and transfusion.
Requester – Locations can use this role to request inventory from American Red Cross.

Admin

```
CREATE ROLE ADMIN;
GRANT ALL ON ALL TABLES IN SCHEMA PUBLIC TO ADMIN;
```

Register

```
CREATE ROLE REGISTER;
REVOKE ALL ON ALL TABLES IN SCHEMA PUBLIC FROM REGISTER;
GRANT SELECT ON ALL TABLES IN SCHEMA PUBLIC TO REGISTER;
GRANT INSERT ON PERSONS, PATIENT, NURSE, DONOR,
                PRE_EXAM, TRANSFUSION, DONATION,
                BLOODBAGS, DONATION_RECORDS,
                TRANSFUSION_RECORDS
                TO REGISTER;
GRANT UPDATE ON PERSONS, PATIENT, NURSE, DONOR,
                PRE_EXAM, TRANSFUSION, DONATION,
                BLOODBAGS, DONATION_RECORDS,
                TRANSFUSION_RECORDS
                TO REGISTER;
```

Requester

```
CREATE ROLE REQUESTER;
REVOKE ALL ON ALL TABLES IN SCHEMA PUBLIC
      FROM REQUESTER;
GRANT SELECT ON REQUESTS, LOCATIONS, LOCATION_CODES,
      locationInventories, availableBloodBags
      TO REQUESTER;
GRANT INSERT ON REQUESTS, LOCATIONS, LOCATION_CODES,
      locationInventories, availableBloodBags
      TO REQUESTER;
GRANT UPDATE ON REQUESTS, LOCATIONS, LOCATION_CODES,
      locationInventories, availableBloodBags
      TO REQUESTER;
```

Implementation Notes – Known Problems – Future Enhancements

- *Implementation Notes*
  - If the data implemented in this database was on a large scale, the inventory for the blood bags would have been better in the UNITS of Pints, rather than CC. 1 Pint is approximately 473 CC, which is the standard donation quantity.

- *Known Problems*
  - More views should be created to better target the users needs to limit interaction.
  - Height and weight should be moved from donor and patient table to the pre-exam table as donors and peoples weight can be different at different times when donating.
  - Next Safe Donation date is not updated properly from the intended stored procedure.

- *Future Enhancements:*
  - Implement checks on blood type input to make sure it is a valid blood type input
  - Implement way to make sure the donor is a valid donor, meeting any requirements or limitations for blood donors.
  - Implement a way to check the global inventory