**Final Project: Blood Bank Database**

Prathibha Bondili, CPT Steven Fowler, Xingyu Liu, Monjurul Islam, Deva Kalasani, Sree Gayatri Anusha Mylavarapu, and Cao Yuan

University of Pittsburgh

HI 2451: Database Design and Big Data Analytics

Dr. Andi Saptano and Dr. August Turano

April 26, 2024

**Introduction**

The goal of this project was to create a local database for a small blood bank organization (we utilized four individuals for the sake of showing the product but imagine something small in the realm of double-digit employees). The goal of this database would be to track the basic day to day data of the operations, to include the donors who are donating, the staff, the donations that occur, the blood that has been procured from those donations, and then tracking requests and transfers to local hospitals. Thus, in doing so, the organization would be able to track past, current, and future data, be able to monitor supply and demand, and attempt to become more effective in their operations as well as having the ability to audit their internal data and efficiency.

As stated above, blood donations are something that need to be carefully monitored and explored when donating and then transferring that blood. Again, much of the objective of this database is for the rapid and precise monitoring of current blood products on hand, requests that need to be filled, the transfers occurring, as well as the surrounding data (i.e., the donors and staff involved). Accurate transfers and accountability can mean literal life and death for patients in many cases, and thus a robust tool to monitor these facets is required.

Some problems and constraints that exist at present in looking at designing such a database. One problem would be ensuring security and privacy, especially since it would contain specifically data on the donors themselves. Ensuring accuracy and integrity is important, especially in managing user access and level of access, ensuring all data is accurately and timely an ad-pop organization, consider in a normally sized organization that, the data is going to be updated, and so forth. While we state later in this documentation that it is almost ad-pop organization, consider in normally sized organization that the data is going to potentially massive, and that scalability will be a large requirement. One aspect that is discussed later in improvements is effectively interoperability, and one would hope that all this data could be pushed to the donors’ electronic health record. If a larger organization as well, technology constraints and geographical distribution issues could furthermore be an issue. Finally, considering database backup and emergency response planning is another problem to consider. While many of these problems/concerns/constraints are standard in database development, it is still something to consider in its development.

Aspects related to the design will be given next in respective sections, such as ERD, business rules, and some example images.

**Design**

**ER Diagram**

A black screen with white text

Description automatically generated

**Business Rules**

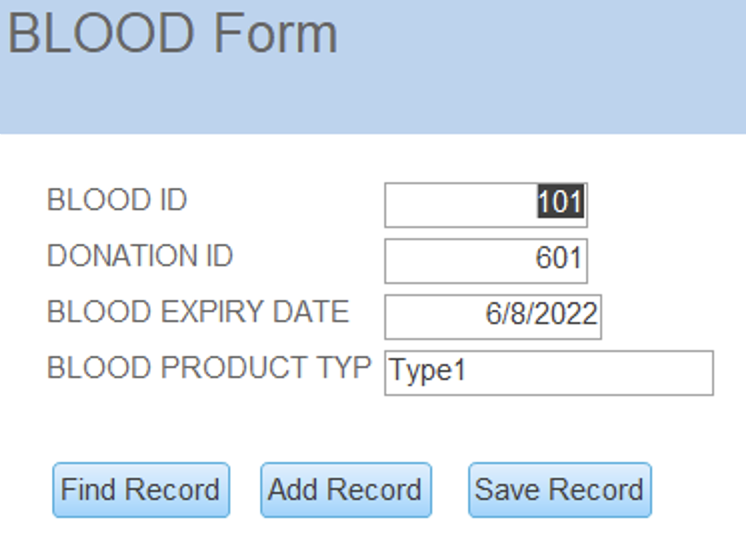
There are several determined business rules as described below:

1. We currently have several aspects related to the ERD above. First, we assume that one donor can have many donations, although the many is optional. Furthermore, for any donation, at minimum one blood product is produced, but there is an optional ability for multiple to be taken during one donation from an individual. Finally, regarding the Blood, Transfer, and Request tables, multiple Blood values can be given in a specific transfer but could be only one as well. Similarly, there must be at least one product requested for each transfer, but it can separately have many blood products requested. Finally, we assume that all staff can be involved in any type of operation and that they can oversee multiple donations and transfers. More specifically, we assume that we require staff to oversee each transfer for accuracy although that could potentially be managed on the application or SQL server level constraints to ensure that the blood being transferred is valid and accurate.
2. Currently, the database is designed such that blood types requested are 1:1 in nature. For example, if ‘A+’ is requested, it is the only type given. However, from a practical standpoint, any blood type that is compatible could be given (such as the universal donor, O-). This is thus limiting as it would make storage management an issue whereas otherwise there would be increased flexibility.
3. Staff are involved in the validation of the transfer process. Preferably there would be application or script level constraints that would confirm through the relationships that it could only give the proper blood. This does, however, allow for us to have a method to audit in case anything occurs as a reaction. This could potentially be done through adding relationships for blood type and product between more tables.
4. Presently, we do not have any health background data on the donors. Furthermore, we do not have any donors that would not have a donation. Thus, the rule is that individuals are only ever entered into this database if they are prescreened. Thus, every donor must be tied to at least one successful donation (can be more). This makes it such that we do not have excess iterative data that would not be useful.
5. We assume that every staff member can do any role. There are no present constraints that would reframe individuals from being assigned to donation or transfer regardless of their role (for example, the manager who could be strictly admin there is no constraints. Instead, you would have to make individual tables to make those relationships individually.
6. For the sake of the project, we assumed all expiring blood was used before the date. This is, however, impractical as there is a decent chance that there could be blood that expires before it is requested/used. This is especially true for something such as platelets which have a shelf life of only 5-7 days when stored. That stated, for the sake of showing the product, all data is treated that it is given in time. Otherwise, there would be data stuck at the BLOOD and DONATION level tables unused in the TRANSFER level.
7. As discussed, there is an ability to merge the Donation, Blood, and Transfer tables into one table and make the request and transfer tables have an optional relationship on the ‘many’ sides such that nulls would be allowed when the blood is currently stored but not yet transferred to a hospital. This would make the user interface easier as any blood that is not yet donated could be searched for and then be altered when prepared to transfer and makes it so manual searching through the other tables is not required and all attributes would be combined (and duplicate attributes listed just once). However, we chose the layout to not make nulls allowable presently and gives us better detail in each location and ensure that individuals do not forget to go back to change null values.

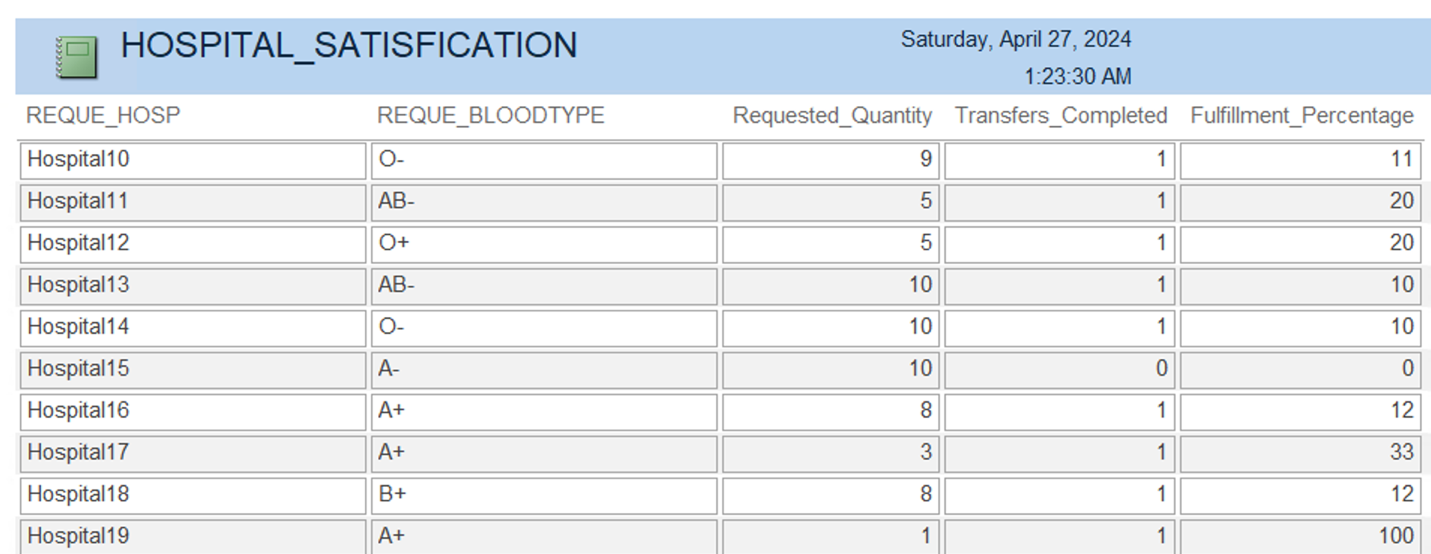
**Data Dictionary**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| -P/F Key | Field Name | Table Name | Data Type | Field Size or Method | Constraints | Notes |
| PK | DON\_ID | DONORS | INT (5,0) | 5 |  | Must use all 5 characters |
|  | DON\_NAME | DONORS | VAR\_CHAR (75) | 75 characters |  | English Language |
|  | DON\_AGE | DONORS | DATE | YYYY-MM-DD |  |  |
|  | DON\_GENDER | DONORS | BOOLEAN | Boolean data |  | 0 = F  1 = M |
|  | DON\_BLOODTYPE | DONORS | CHAR (3) | 3 | ENUM('A+', 'A-', 'B+', 'B-', 'AB+', 'AB-', 'O+', 'O-') | Enumerated accordingly during design |
|  | DON\_HIS | DONORS | VARCHAR(100) | 100 Characters |  | English Language |
| PK | BLOOD\_ID | BLOOD | INT(5,0) | 5 |  | Must use all 5 characters |
| FK | DONA\_ID | BLOOD | INT(5,0) | 5 |  | Must use all 5 characters |
|  | BLOOD\_EXP\_DATA | BLOOD | DATE | YYYY-MM-DD |  |  |
|  | BLOOD\_PRO\_TYPE | BLOOD | VARCHAR(50) | 50 | ENUM(‘BloodProType0’, ‘BloodProType01t’, “BloodProType2’) | Whole Blood =BloodProType0, Plasma = BloodProType1,\Platlets = BloodProType2, |
| PK1 | DONA\_ID | DONATION | INT (5,0) | 5 |  | Must use all 5 characters |
| FK1 | DON\_ID | DONATION | INT(5,0) | 5 |  | Must use all 5 characters |
| PK2 | STAFF\_ID | DONATION | INT(5,0) | 5 |  | Must use all 5 characters |
|  | DONA\_DATE | DONATION | DATE | YYYY-MM-DD |  |  |
| PK | STAFF\_ID | STAFF | INT(5,0) | 5 |  | Must use all 5 characters |
|  | STAFF\_NAME | STAFF | VAR\_CHAR (75) | 75 characters |  | English Language |
|  | STAFF\_PHONE\_NUMBER | STAFF | NUMBER(10) |  |  | Dashes unallowed |
|  | STAFF\_ROLE | STAFF | VARCHAR(25) | 25 Characters |  |  |
| PK | TRANSFER\_ID | TRANSFER | INT(5,0) | 5 |  | Must use all 5 characters |
| FK1 | BLOOD\_ID | TRANSFER | INT(5,0) | 5 |  |  |
| FK2 | REQUE\_ID | TRANSFER | INT(5,0) | 5 |  |  |
| FK3 | STAFF\_ID | TRANSFER | INT(5,0) | 5 |  |  |
|  | TRANS\_DATE | TRANSFER | DATE | YYYY-MM-DD |  |  |
|  | TRANS\_COMPA\_RESULT | TRANSFER | VAR\_CHAR (75) | 75 characters |  | English Language |
| PK | REQUE\_ID | REQUEST | INT(5,0) | 5 |  | Must use all 5 characters |
|  | REQUE-BLOODTYPE | REQUEST | CHAR(3) | 3 | ENUM('A+', 'A-', 'B+', 'B-', 'AB+', 'AB-', 'O+', 'O-') | Enumerated accordingly during design |
|  | REQUE\_BLOODQUAN | REQUEST | INT(2,0) | 2 |  |  |
|  | REQUE\_HOSPITAL | REQUEST | VARCHAR(50) |  |  |  |

**Draft Design User Interface (one example)**



**Draft Design of Reports (one example)**



**Detailed System Specifications**

For the database, it depends on what we would say is the size of our organization. We presently have 4 personnel, so we would maintain an internal server with SAN’s internal to us. We could do BitLocker for encryption as well as pay for a cloud service to conduct an offsite backup after we do a blood drive or a request as presumably this is small and manageable at this point, and we would not be doing an inordinate amount of these transactions. That stated, assuming we were a larger organization across a large area (a whole state, states, nationally), we would use a HIPAA (Health Insurance Portability and Accountability) compatible cloud-based system (one suggestion would be Microsoft Azure Healthcare as it has a large arrangement of security options). Encryption would be managed through encryption keys and SSL and TLS encryption. The data would be available instantaneously anywhere and backed up by the system and would be easily scalable at a cheap cost holistically.

**Maintenance and Training Plan**

There are various aspects that could be considered in creating a maintenance and training plan. For maintenance, aspects such as regular backups and security updates primarily if we use a local DBMS (cloud services would have their own updates as part of the product). We can also consider disaster recovery planning and offsite backups accordingly on a remote server. You could consider further documenting all maintenance procedures used and employee training on these matters, so the methodology and knowledge is not lost over time.

Some considerations for training could include the following. First, an important aspect would be the database designers to conduct database administration training with some of the administrative, leadership, or otherwise identified staff so they can continue to monitor and enhance the database in time. Security awareness training and HIPAA are always aspects that should be considered with any medical database annually. Furthermore, videos or training modules for user level training are of utmost importance. Finally, just like the HIPAA and security training, consider doing ongoing education on a quarterly or annual basis such that individuals do not become complacent and lose the lessons learned.

**Database Discussion**

One limitation/constraint is that donor history is not much reflected in screening at present and it is assumed individuals are already ready to donate by the time they enter the system. Specifically, any disease (take HIV as an example) would at best be captured in that free text. Ultimately, an expansion and then application level flag would be implemented to stop any potentially contaminated blood from either entering the registry or go into a separate portion (a better solution) that is tied to history with donors such that the donor is not allowed to donate in the case of certain diseases or more highly tested if the disease is one which can potentially resolve (i.e. some bloodborne pathogen that could either self-resolve or through the assistance of antibiotics). The basis for the database is here, however it is currently focused on the donation and transfusion brass tacks and not potential additional features.

Yet another constraint and potential addition would be to add the ability to check all viable donor blood types for receiving patients, not necessarily a one-to-one match as the system is currently presented. As designed presently, all data is made such that the blood type for the blood transferred to a hospital that is requesting that exact blood type (which is not necessarily wrong, but something like O- can always be used, even if not potentially most optimal). While in one specific case this is completely necessary (O- blood), every other blood type can receive blood from one or more types. For example, AB+, the universal receiver, can have a transfusion from any other blood type. Thus, the system could be improved to remove that restriction and instead verify in each blood type of the eight major categories can receive the blood designated and potentially block or create glaring warnings in the case that the blood product is inappropriate to be transfused.

Another limitation which coincides with future development and implementation would be to work to integrate this potentially into a patient’s EHR (Electronic Health Record) anytime they donated blood. Thus, the data gathered strictly within this database can be used for a far greater clinical significance. One can imagine the case to ensure it is in their record in case they appear hypovolemic after giving blood and a potential root cause could be found. The database as stands assists in the collection, distribution, and monitoring of current levels and expiration dates primarily, and all the limitations and safety aspects would be improvements to the overall database.

Another improvement that could be made would be to make application level would be to make it such that when entering the attributes in the user application that the date for expiration could be automatically created without the user having to do so and calculate manually. This could be done by making a function such that based off the type of product, the correct number of days would be added to the donation date to produce the expiration date. This would make it easier for the user and lessen their burden and effort in accomplishing data entry.

A further addition that could be added to tables such as Staff, Donors, Blood, and Requests tables would be an “active” attribute. For example, when transferred, the Blood could then be marked from active to inactive insomuch that it has now been consumed and left our inventory. This could also be done if it is found that the blood is contaminated, expires, or otherwise before being transferred so it is known that it can no longer to be used Similarly, if a staff member leaves, a request is cancelled or fulfilled, etc. They can be marked inactive such that only active requests remain, so we know what needs to be fulfilled. Another aspect this would assist could be to show audit trails such that if a transfer or request is modified you can illustrate the data trail. All of these could help monitor what is active, consumed, needs to be tabled for whatever reason, and so forth, enhancing the data's quality.

**Appendix**

**Appendix A: SQL Script**

CREATE TABLE DONORS (

DON\_ID INT,

DON\_NAME VARCHAR(75),

DON\_AGE NUMERIC(2,0),

DON\_GENDER BIT,

DON\_BLOODTYPE CHAR(3) CHECK (DON\_BLOODTYPE IN ('A+', 'A-', 'B+', 'B-', 'AB+', 'AB-', 'O+', 'O-')),

PRIMARY KEY (DON\_ID)

);

CREATE TABLE BLOOD (

BLOOD\_ID INT,

BLOOD\_EXP\_DATE DATE,

BLOOD\_PRO\_TYPE VARCHAR(20),

DON\_ID INT,

PRIMARY KEY (BLOOD\_ID),

FOREIGN KEY (DONA\_ID) REFERENCES DONATION(DONA\_ID)

);

CREATE TABLE STAFF (

STAFF\_ID INT,

STAFF\_NAME VARCHAR(75),

STAFF\_ROLE VARCHAR(25),

STAFF\_PHONE\_NUMBER NUMERIC(10,0),

PRIMARY KEY (STAFF\_ID)

);

CREATE TABLE REQUEST (

REQUE\_ID INT,

REQUE\_BLOODTYPE CHAR(3) CHECK (PAT\_BLOODTYPE IN ('A+', 'A-', 'B+', 'B-', 'AB+', 'AB-', 'O+', 'O-')),

REQUE\_BLOODQUAN NUMERIC(2,0),

REQUE\_HOSP VARCHAR(75),

PRIMARY KEY (REQUE\_ID)

);

CREATE TABLE DONATION (

DONA\_ID INT,

DON\_ID INT,

STAFF\_ID INT,

DONA\_DATE DATE,

PRIMARY KEY (DONA\_ID),

FOREIGN KEY (DON\_ID) REFERENCES DONORS(DON\_ID),

FOREIGN KEY (STAFF\_ID) REFERENCES STAFF(STAFF\_ID),

);

CREATE TABLE TRANSFER (

TRANS\_ID INT,

BLOOD\_ID INT,

REQUE\_ID INT,

STAFF\_ID INT,

TRANS\_DATE DATE,

TRANS\_COMPA\_RESULT BIT,

PRIMARY KEY (TRANS\_ID),

FOREIGN KEY (STAFF\_ID) REFERENCES STAFF(STAFF\_ID),

FOREIGN KEY (BLOOD\_ID) REFERENCES BLOOD(BLOOD\_ID),

FOREIGN KEY (REQUE\_ID) REFERENCES REQUEST(REQUE\_ID)

);

**Appendix B: Report SQL Script**

SELECT dbo\_REQUEST.REQUE\_HOSP, dbo\_REQUEST.REQUE\_BLOODTYPE, dbo\_REQUEST.REQUE\_BLOODQUAN AS Requested\_Quantity, COUNT(dbo\_TRANSFERS.TRANS\_ID) AS Transfers\_Completed, ROUND((Transfers\_Completed \* 100.0) / dbo\_REQUEST.REQUE\_BLOODQUAN) AS Fulfillment\_Percentage  
FROM dbo\_REQUEST LEFT JOIN dbo\_TRANSFERS ON dbo\_REQUEST.REQUE\_ID = dbo\_TRANSFERS.REQUE\_ID  
GROUP BY dbo\_REQUEST.REQUE\_HOSP, dbo\_REQUEST.REQUE\_BLOODTYPE, dbo\_REQUEST.REQUE\_BLOODQUAN  
ORDER BY dbo\_REQUEST.REQUE\_HOSP;

SELECT dbo\_BLOOD.BLOOD\_PRO\_TYPE, dbo\_DONORS.DON\_BLOODTYPE, COUNT(dbo\_BLOOD.BLOOD\_ID) AS Total\_Units, SUM(IIF(dbo\_BLOOD.BLOOD\_EXP\_DATE < DATE(), 1, 0)) AS Expired\_Units, SUM(IIF(dbo\_BLOOD.BLOOD\_EXP\_DATE >= DATE(), 1, 0)) AS Valid\_Units  
FROM (dbo\_BLOOD INNER JOIN dbo\_DONATION ON dbo\_BLOOD.DONA\_ID = dbo\_DONATION.DONA\_ID) INNER JOIN dbo\_DONORS ON dbo\_DONATION.DON\_ID = dbo\_DONORS.DON\_ID  
GROUP BY dbo\_DONORS.DON\_BLOODTYPE, dbo\_BLOOD.BLOOD\_PRO\_TYPE  
ORDER BY dbo\_DONORS.DON\_BLOODTYPE;

**Appendix C: Group Photo**

