Blood Bank Database

Mike Miller, Nolan Wacker, Yeray Lopez

The Team

Nolan Wacker

Database Design

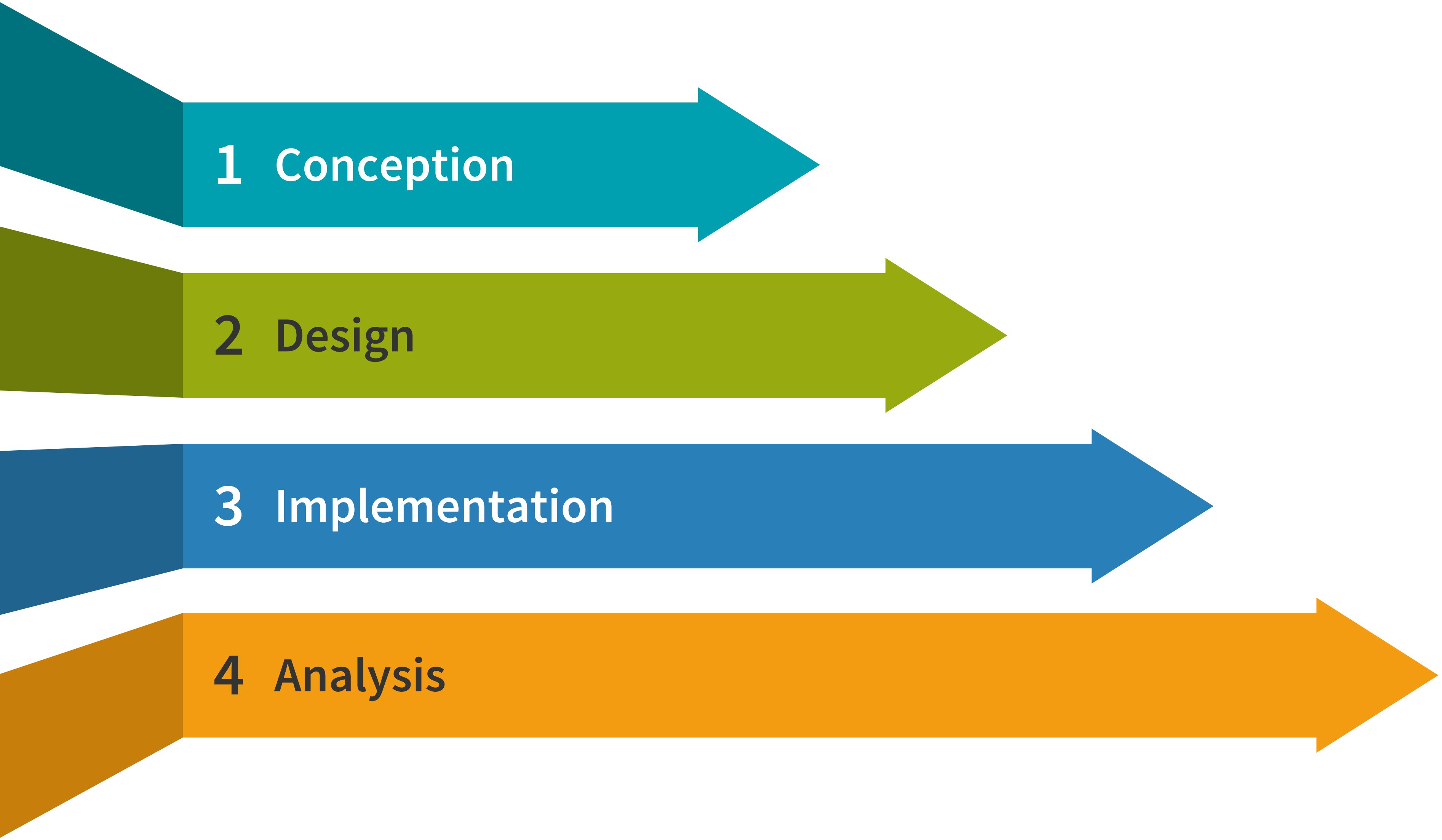
Yeray Lopez

Database Implementation

Mike Miller

Database Analysis

Project Overview



1 Conception

2 Design

3 Implementation

4 Analysis

Conception

1. Practical Application
2. Interesting Dataset
3. Variety of Characteristics
4. Realistic Use Case
5. Easy to comprehend

MADE WITH
beautiful.ai

Design

- **Rules and Restrictions**

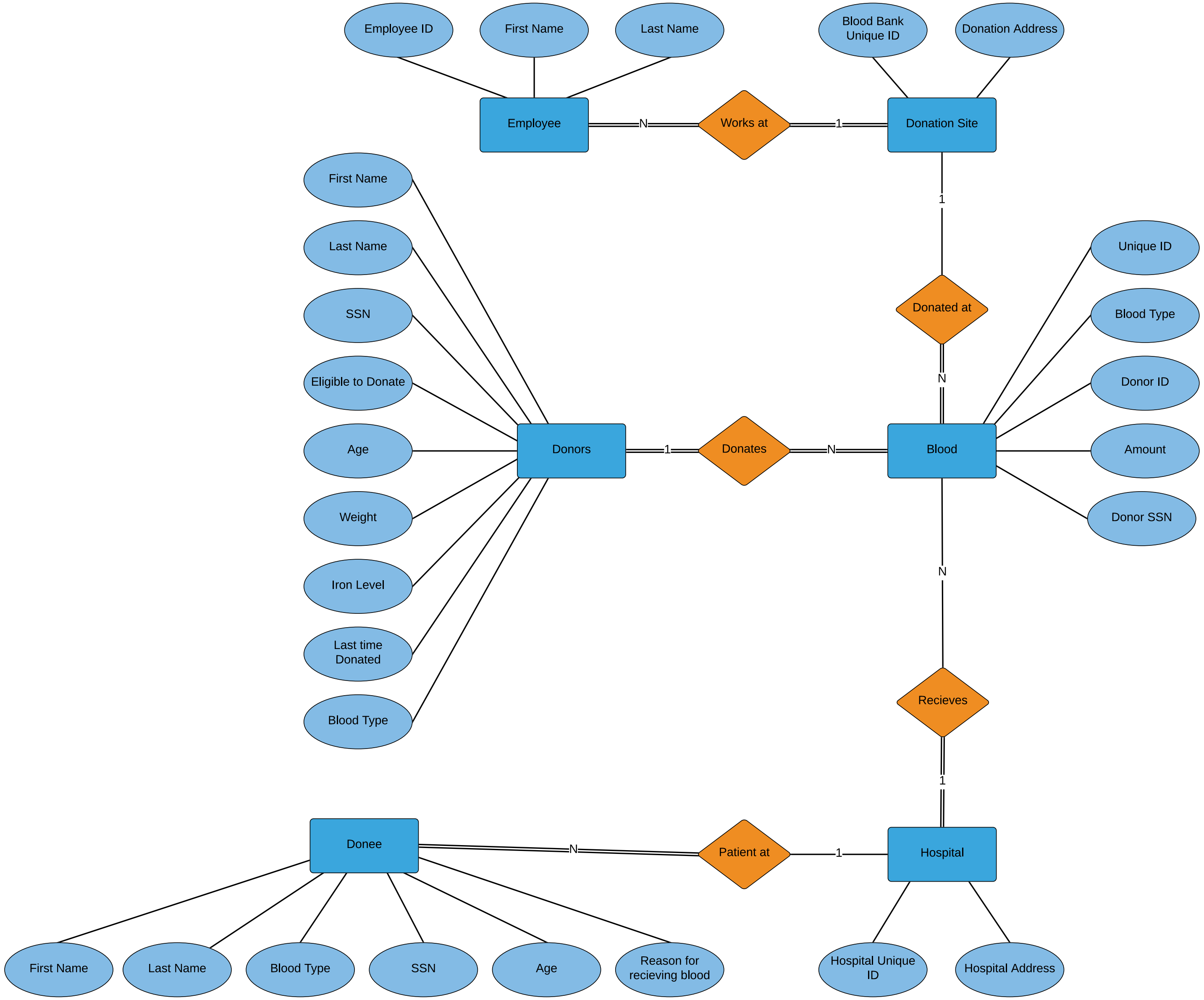
  Main rule the social security numbers and IDs cannot be null

  Many of the values for the donor table cannot be null due to that information impacting eligibility

- **Relations and Foreign Keys**

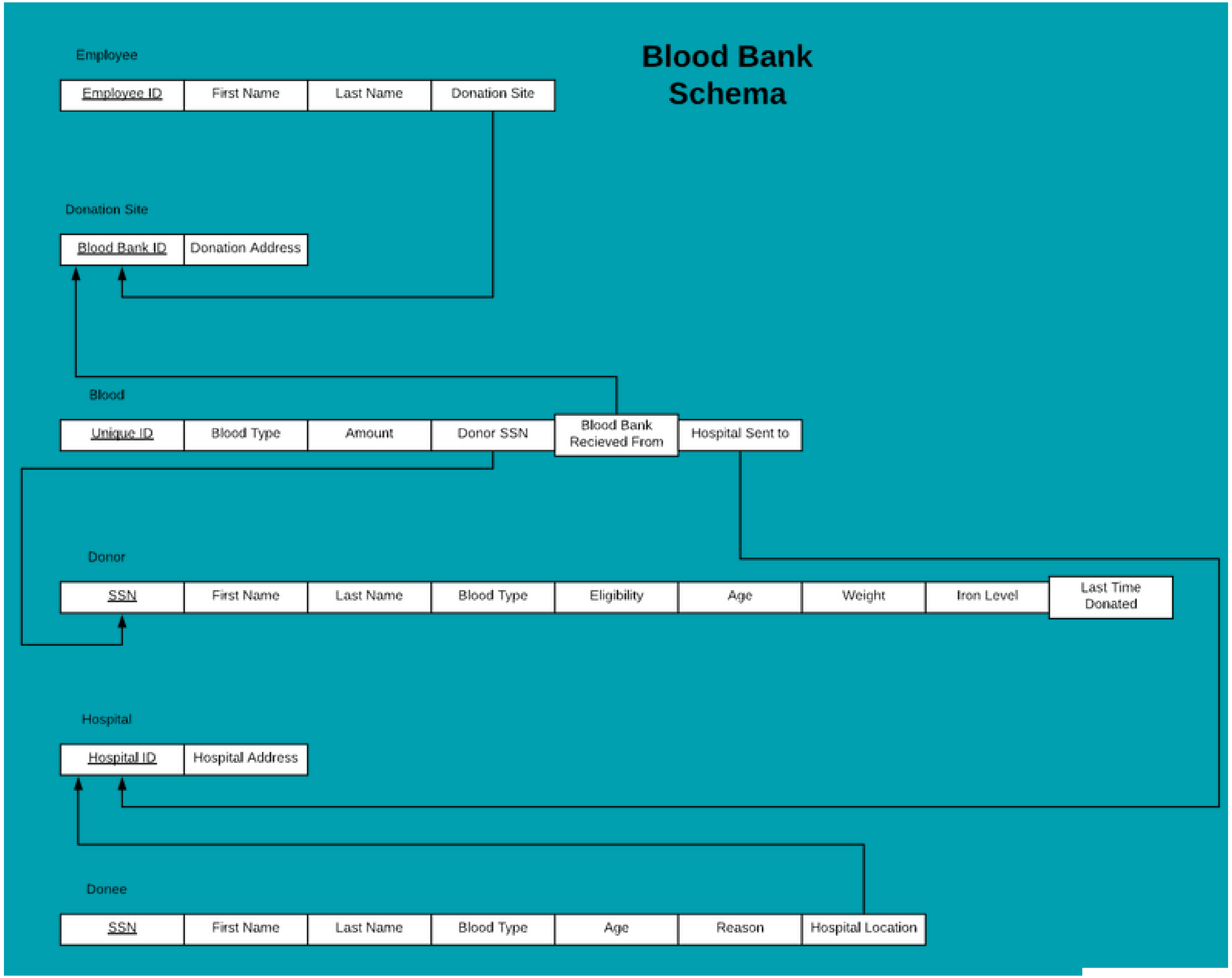
  Need to have foreign keys to represents the relations

Blood Bank ERD



MADE WITH beautiful.ai

Schema



Blood Bank Schema

Employee

Employee ID	First Name	Last Name	Donation Site

Donation Site

Blood Bank ID	Donation Address

Blood

Unique ID	Blood Type	Amount	Donor SSN	Blood Bank Recieved From	Hospital Sent to

Donor

SSN	First Name	Last Name	Blood Type	Eligibility	Age	Weight	Iron Level	Last Time Donated

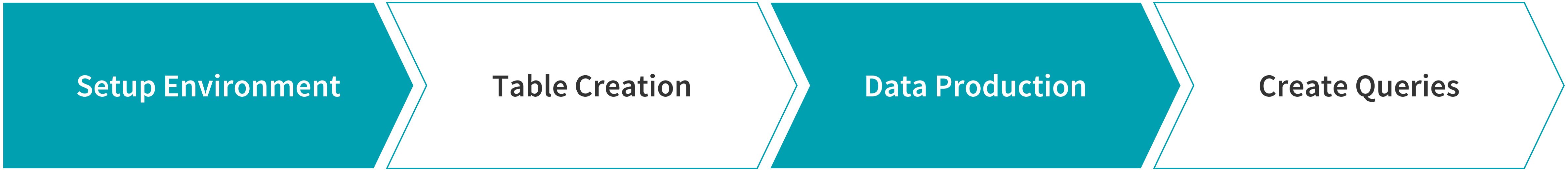
Hospital

Hospital ID	Hospital Address

Donee

SSN	First Name	Last Name	Blood Type	Age	Reason	Hospital Location

Implementation



Setup Environment

Table Creation

Data Production

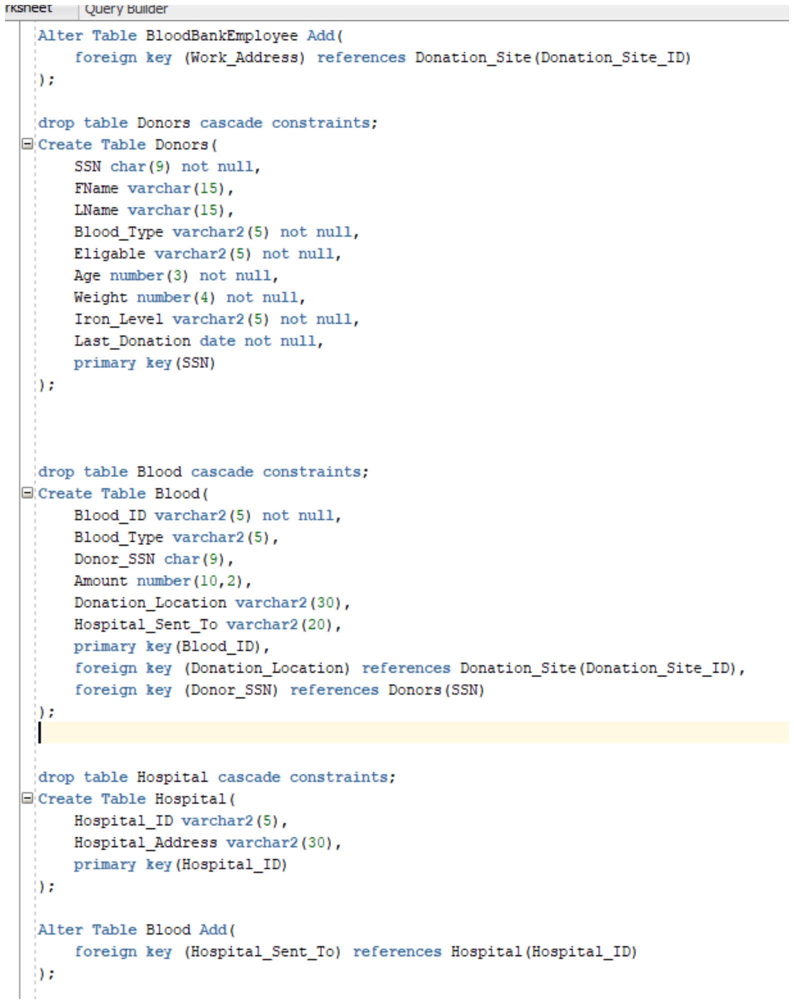
Create Queries

Environment

Windows

Oracle SQL Developer

Table Creation



```
Alter Table BloodBankEmployee Add(
    foreign key (Work_Address) references Donation_Site(Donation_Site_ID)
);

drop table Donors cascade constraints;
Create Table Donors(
    SSN char(9) not null,
    FName varchar(15),
    LName varchar(15),
    Blood_Type varchar2(5) not null,
    Eligable varchar2(5) not null,
    Age number(3) not null,
    Weight number(4) not null,
    Iron_Level varchar2(5) not null,
    Last_Donation date not null,
    primary key(SSN)
);



drop table Blood cascade constraints;
Create Table Blood(
    Blood_ID varchar2(5) not null,
    Blood_Type varchar2(5),
    Donor_SSN char(9),
    Amount number(10,2),
    Donation_Location varchar2(30),
    Hospital_Sent_To varchar2(20),
    primary key(Blood_ID),
    foreign key (Donation_Location) references Donation_Site(Donation_Site_ID),
    foreign key (Donor_SSN) references Donors(SSN)
);


drop table Hospital cascade constraints;
Create Table Hospital(
    Hospital_ID varchar2(5),
    Hospital_Address varchar2(30),
    primary key(Hospital_ID)
);

Alter Table Blood Add(
    foreign key (Hospital_Sent_To) references Hospital(Hospital_ID)
);
```

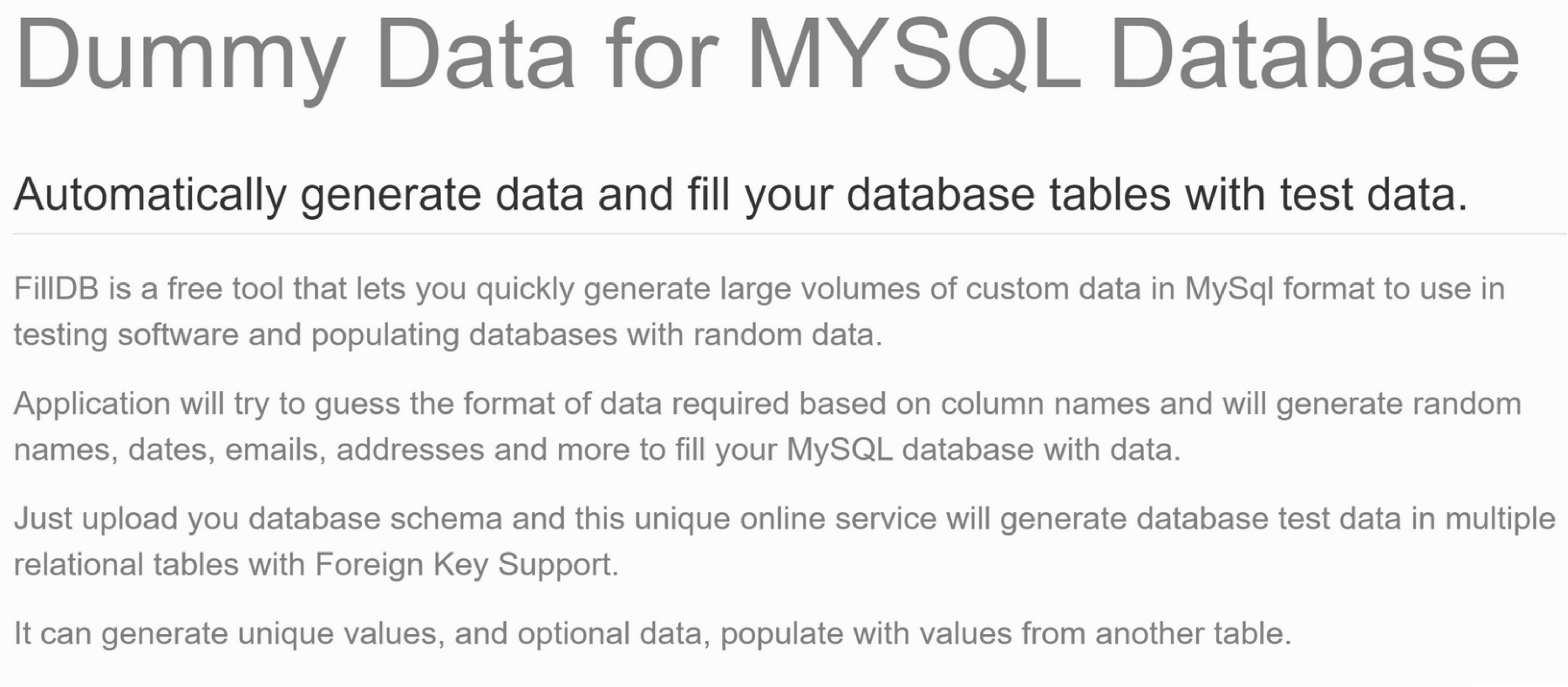
- **Main variable types are number and varchar2**

  Varchar2 used for names and addresses that are not standard

  Number used for more standardized amounts like SSN

  Blood table attributes are not null

Data Production - Fill Database



Dummy Data for MYSQL Database

Automatically generate data and fill your database tables with test data.

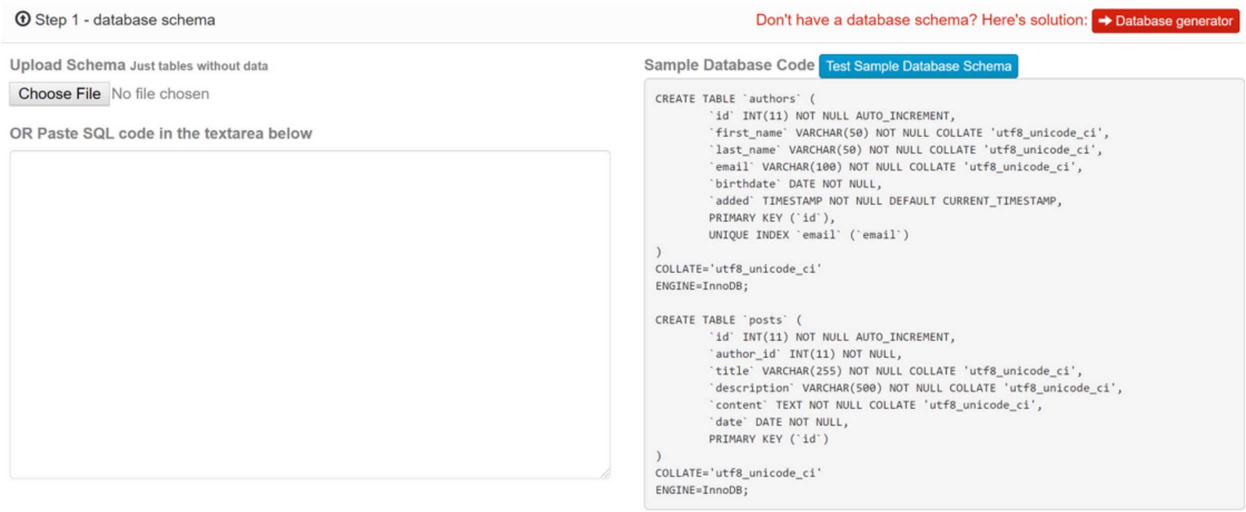
FillDB is a free tool that lets you quickly generate large volumes of custom data in MySql format to use in testing software and populating databases with random data.

Application will try to guess the format of data required based on column names and will generate random names, dates, emails, addresses and more to fill your MySQL database with data.

Just upload you database schema and this unique online service will generate database test data in multiple relational tables with Foreign Key Support.

It can generate unique values, and optional data, populate with values from another table.

Fill DB Workflow



Step 1 - database schema

Don't have a database schema? Here's solution: ➔ Database generator

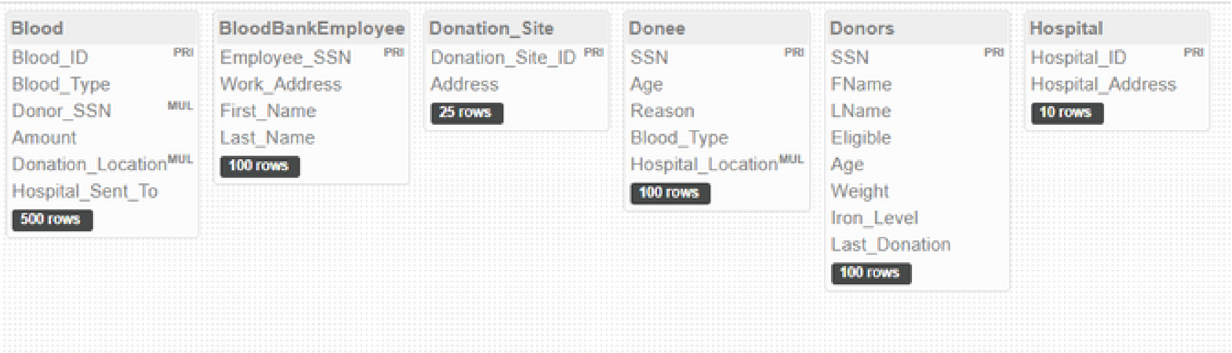
Upload Schema Just tables without data

Choose File No file chosen

OR Paste SQL code in the textarea below

Sample Database Code Test Sample Database Schema

```
CREATE TABLE `authors` (
        `id` INT(11) NOT NULL AUTO_INCREMENT,
        `first_name` VARCHAR(50) NOT NULL COLLATE 'utf8_unicode_ci',
        `last_name` VARCHAR(50) NOT NULL COLLATE 'utf8_unicode_ci',
        `email` VARCHAR(100) NOT NULL COLLATE 'utf8_unicode_ci',
        `birthdate` DATE NOT NULL,
        `added` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
        PRIMARY KEY (`id`),
        UNIQUE INDEX `email` (`email`)
)
COLLATE='utf8_unicode_ci'
ENGINE=InnoDB;

CREATE TABLE `posts` (
        `id` INT(11) NOT NULL AUTO_INCREMENT,
        `author_id` INT(11) NOT NULL,
        `title` VARCHAR(255) NOT NULL COLLATE 'utf8_unicode_ci',
        `description` VARCHAR(500) NOT NULL COLLATE 'utf8_unicode_ci',
        `content` TEXT NOT NULL COLLATE 'utf8_unicode_ci',
        `date` DATE NOT NULL,
        PRIMARY KEY (`id`)
)
COLLATE='utf8_unicode_ci'
ENGINE=InnoDB;
```



Blood
Blood_ID PRI
Blood_Type
Donor_SSN MUL
Amount
Donation_Location MUL
Hospital_Sent_To
500 rows

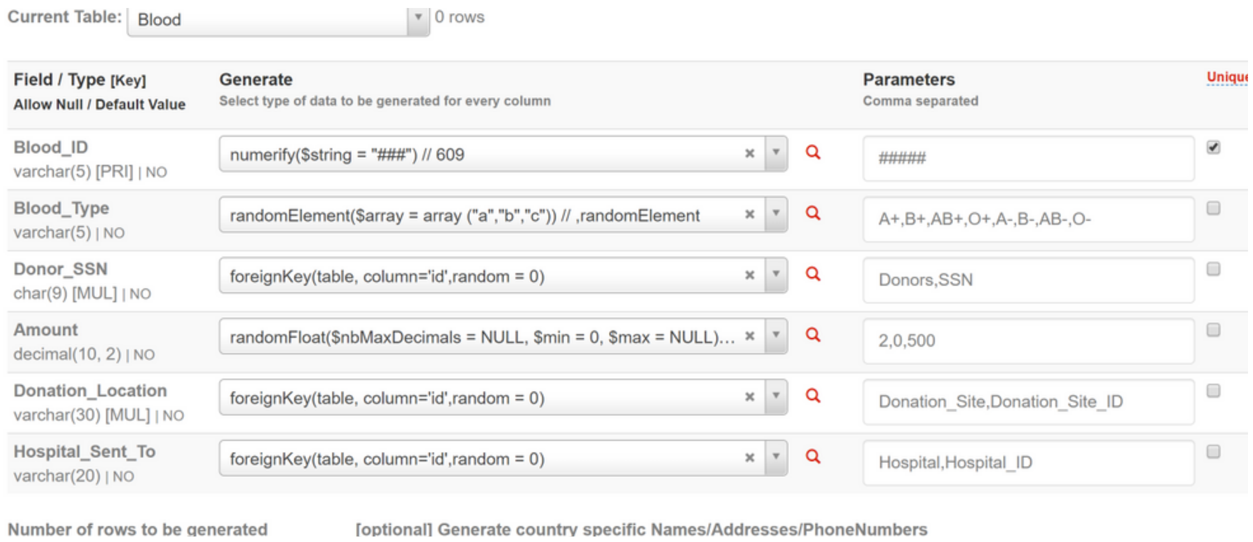
BloodBankEmployee
Employee_SSN PRI
Work_Address
First_Name
Last_Name
100 rows

Donation_Site
Donation_Site_ID PRI
Address
25 rows

Donee
SSN PRI
Age
Reason
Blood_Type
Hospital_Location MUL
100 rows

Donors
SSN PRI
FName
LName
Eligible
Age
Weight
Iron_Level
Last_Donation
100 rows

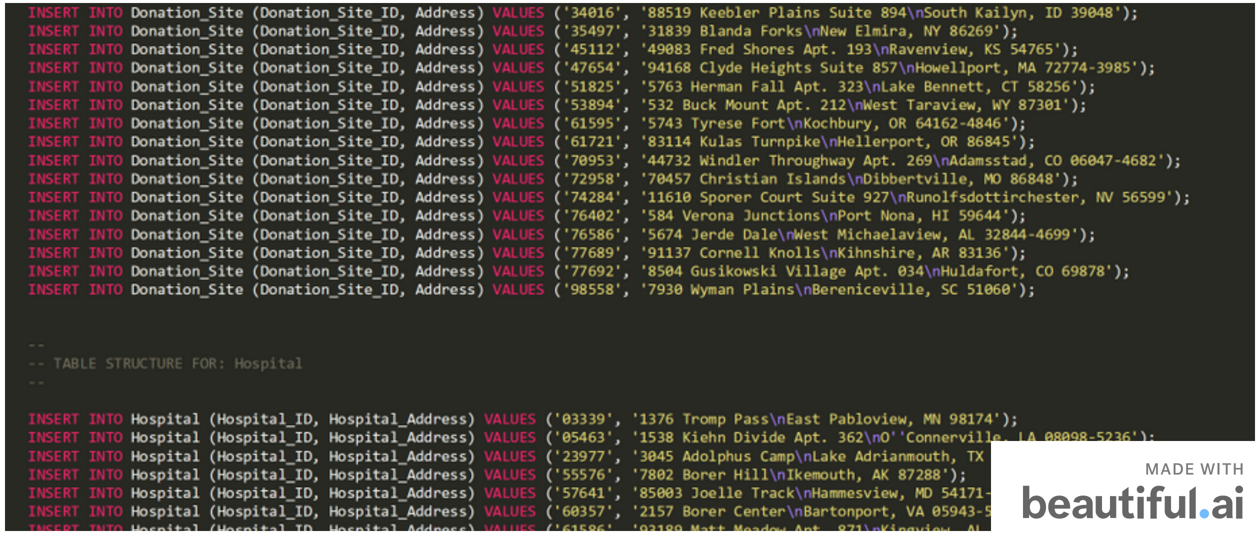
Hospital
Hospital_ID PRI
Hospital_Address
10 rows



Current Table: Blood ▾ 0 rows

Field / Type [Key] Allow Null / Default Value	Generate Select type of data to be generated for every column	Parameters Comma separated	Unique
Blood_ID varchar(5) [PRI] | NO	numerify($string = "###") // 609	#####	☑
Blood_Type varchar(5) | NO	randomElement($array = array ("a","b","c")) // ,randomElement	A+,B+,AB+,O+,A-,B-,AB-,O-	☐
Donor_SSN char(9) [MUL] | NO	foreignKey(table, column='id',random = 0)	Donors,SSN	☐
Amount decimal(10, 2) | NO	randomFloat($nbMaxDecimals = NULL, $min = 0, $max = NULL)…	2,0,500	☐
Donation_Location varchar(30) [MUL] | NO	foreignKey(table, column='id',random = 0)	Donation_Site,Donation_Site_ID	☐
Hospital_Sent_To varchar(20) | NO	foreignKey(table, column='id',random = 0)	Hospital,Hospital_ID	☐

Number of rows to be generated [optional] Generate country specific Names/Addresses/PhoneNumbers



```
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('34016', '88519 Keebler Plains Suite 894\nSouth Kailyn, ID 39048');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('35497', '31839 Blanda Forks\nNew Elmira, NY 86269');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('45112', '49083 Fred Shores Apt. 193\nRavenview, KS 54765');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('47654', '94168 Clyde Heights Suite 857\nHowellport, MA 72774-3985');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('51825', '5763 Herman Fall Apt. 323\nLake Bennett, CT 58256');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('53894', '532 Buck Mount Apt. 212\nWest Taraview, WY 87301');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('61595', '5743 Tyrese Fort\nKochbury, OR 64162-4846');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('61721', '83114 Kulas Turnpike\nHellerport, OR 86845');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('70953', '44732 Windler Throughway Apt. 269\nAdamsstad, CO 06047-4682');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('72958', '70457 Christian Islands\nDibbertville, MO 86848');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('74284', '11610 Sporer Court Suite 927\nRunolfsdottirchester, NV 56599');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('76402', '584 Verona Junctions\nPort Nona, HI 59644');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('76586', '5674 Jerde Dale\nWest Michaelaview, AL 32844-4699');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('77689', '91137 Cornell Knolls\nKihnshire, AR 83136');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('77692', '8504 Gusikowski Village Apt. 034\nHuldafort, CO 69878');
INSERT INTO Donation_Site (Donation_Site_ID, Address) VALUES ('98558', '7930 Wyman Plains\nBereniceville, SC 51060');


--
-- TABLE STRUCTURE FOR: Hospital
--

INSERT INTO Hospital (Hospital_ID, Hospital_Address) VALUES ('03339', '1376 Tromp Pass\nEast Pabloview, MN 98174');
INSERT INTO Hospital (Hospital_ID, Hospital_Address) VALUES ('05463', '1538 Kiehn Divide Apt. 362\nO''Connerville, LA 08098-5236');
INSERT INTO Hospital (Hospital_ID, Hospital_Address) VALUES ('23977', '3045 Adolphus Camp\nLake Adrianmouth, TX
INSERT INTO Hospital (Hospital_ID, Hospital_Address) VALUES ('55576', '7802 Borer Hill\nIkemouth, AK 87288');
INSERT INTO Hospital (Hospital_ID, Hospital_Address) VALUES ('57641', '85003 Joelle Track\nHammesview, MD 54171-
INSERT INTO Hospital (Hospital_ID, Hospital_Address) VALUES ('60357', '2157 Borer Center\nBartonport, VA 05943-5
INSERT INTO Hospital (Hospital_ID, Hospital_Address) VALUES ('61586', '93189 Matt Meadow Apt. 871\nKingview, AL
```

MADE WITH beautiful.ai

Fill Database Experience

Written for MySQL

Convertible to SQL

Learned MySQL Syntax

Reusable

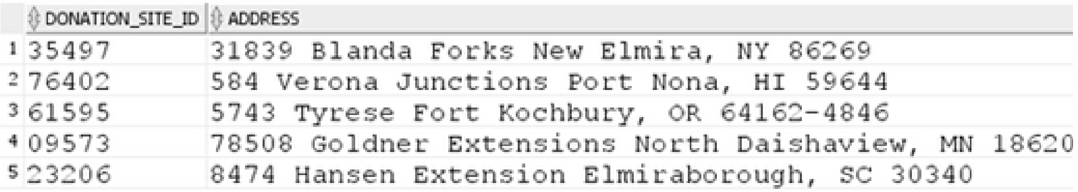
Ease of Use

Analysis - Example Queries

Donation Sites that Sent Blood to Hospitals in Minnesota

SELECT DISTINCT(d.donation_site_id), d.address
FROM donation_site d, hospital h, blood b
 WHERE b.hospital_sent_to = (SELECT h.hospital_id
FROM hospital h WHERE h.hospital_address LIKE
'%MN%')
 AND d.donation_site_id = b.donation_location;

NESTED QUERIES



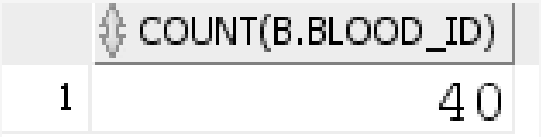
	DONATION_SITE_ID	ADDRESS
1	35497	31839 Blanda Forks New Elmira, NY 86269
2	76402	584 Verona Junctions Port Nona, HI 59644
3	61595	5743 Tyrese Fort Kochbury, OR 64162-4846
4	09573	78508 Goldner Extensions North Daishaview, MN 18620
5	23206	8474 Hansen Extension Elmiraborough, SC 30340

Analysis - Example Queries

Number of Donations in Oregon

```
SELECT COUNT(b.blood_id) FROM donation_site d,
blood b
 WHERE b.donation_location = d.donation_site_id
 AND d.address LIKE '%OR%';
```

AGGREGATE FUNCTIONS



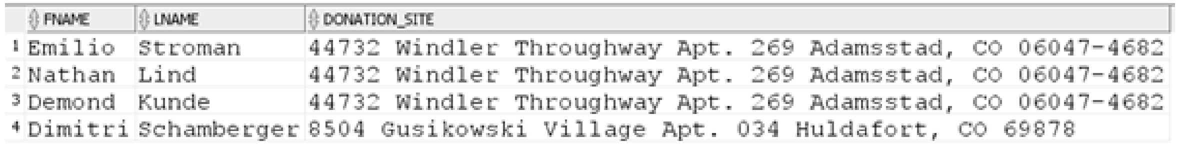
	COUNT(B.BLOOD_ID)
1	40

Analysis - Example Queries

Eligible Donors Who Donated in Colorado

```
SELECT DISTINCT d.fname, d.lname,
ds.address as Donation_Site
 FROM (donors d INNER JOIN blood b on d.ssn
= b.donor_ssn)
 INNER JOIN donation_site ds on
b.donation_location = ds.donation_site_id
 WHERE ds.address LIKE '%CO%'
 AND d.eligible = 'Yes';
```

JOINS



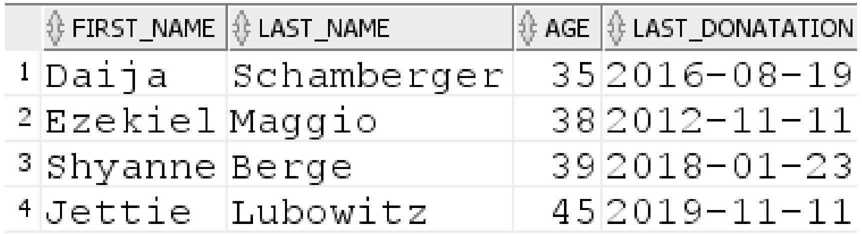
	FNAME	LNAME	DONATION_SITE
1	Emilio	Stroman	44732 Windler Throughway Apt. 269 Adamsstad, CO 06047-4682
2	Nathan	Lind	44732 Windler Throughway Apt. 269 Adamsstad, CO 06047-4682
3	Demond	Kunde	44732 Windler Throughway Apt. 269 Adamsstad, CO 06047-4682
4	Dimitri	Schamberger	8504 Gusikowski Village Apt. 034 Huldafort, CO 69878

Analysis - Example Queries

Young (25-50) who are Active Donors (<7 Years)

```
Create View young_active_donors (first_name, last_name, age,
last_donatation)
 AS
 Select d.fname, d.lname, d.age, d.last_donation FROM donors d
 WHERE d.age > 25
 AND d.age < 50
 AND d.last_donation > '2012';

 --order by increasing age
 SELECT * FROM young_active_donors
 ORDER BY age ASC;

 DROP VIEW young_active_donors;
```

CREATING VIEWS



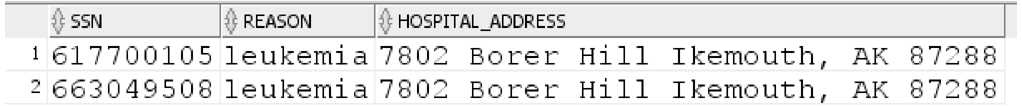
	FIRST_NAME	LAST_NAME	AGE	LAST_DONATATION
1	Daija	Schamberger	35	2016-08-19
2	Ezekiel	Maggio	38	2012-11-11
3	Shyanne	Berge	39	2018-01-23
4	Jettie	Lubowitz	45	2019-11-11

Analysis - Example Queries

Donees in Arkansas Receiving Blood Because of Leukemia

```
SELECT de.ssn, de.reason, h.hospital_address FROM
donee de, hospital h
 WHERE de.hospital_location = h.hospital_id
 AND h.hospital_address LIKE '%AK%'
 AND de.reason = 'leukemia';
```

IMPLICIT JOIN



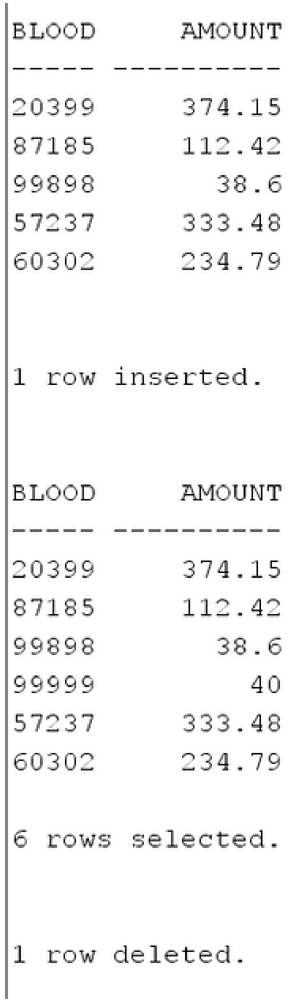
	SSN	REASON	HOSPITAL_ADDRESS
1	617700105	leukemia	7802 Borer Hill Ikemouth, AK 87288
2	663049508	leukemia	7802 Borer Hill Ikemouth, AK 87288

Analysis - Example Queries

Adding a New Donation

```
SELECT blood_id, amount FROM blood
 WHERE donor_ssn = '796881404';

 INSERT INTO blood
 VALUES ('99999', 'B+', '796881404', '40.00', '23206', '03339');

 SELECT blood_id, amount FROM blood
 WHERE donor_ssn = '796881404';

 DELETE FROM blood
 WHERE blood_id = '99999';
```

INSERT AND DELETE



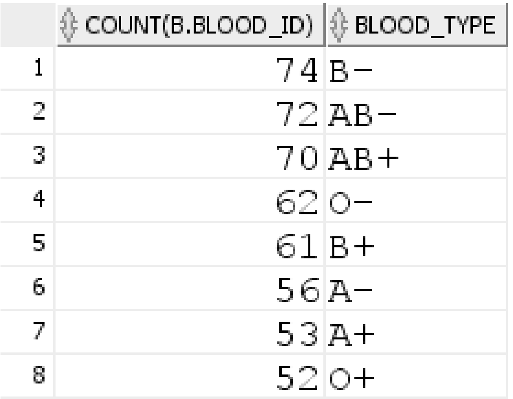
```
BLOOD     AMOUNT
----- ----------
20399     374.15
87185     112.42
99898       38.6
57237     333.48
60302     234.79


1 row inserted.


BLOOD     AMOUNT
----- ----------
20399     374.15
87185     112.42
99898       38.6
99999         40
57237     333.48
60302     234.79

6 rows selected.


1 row deleted.
```

Practical Example - Blood Types

Number of Blood Donations of Each Type

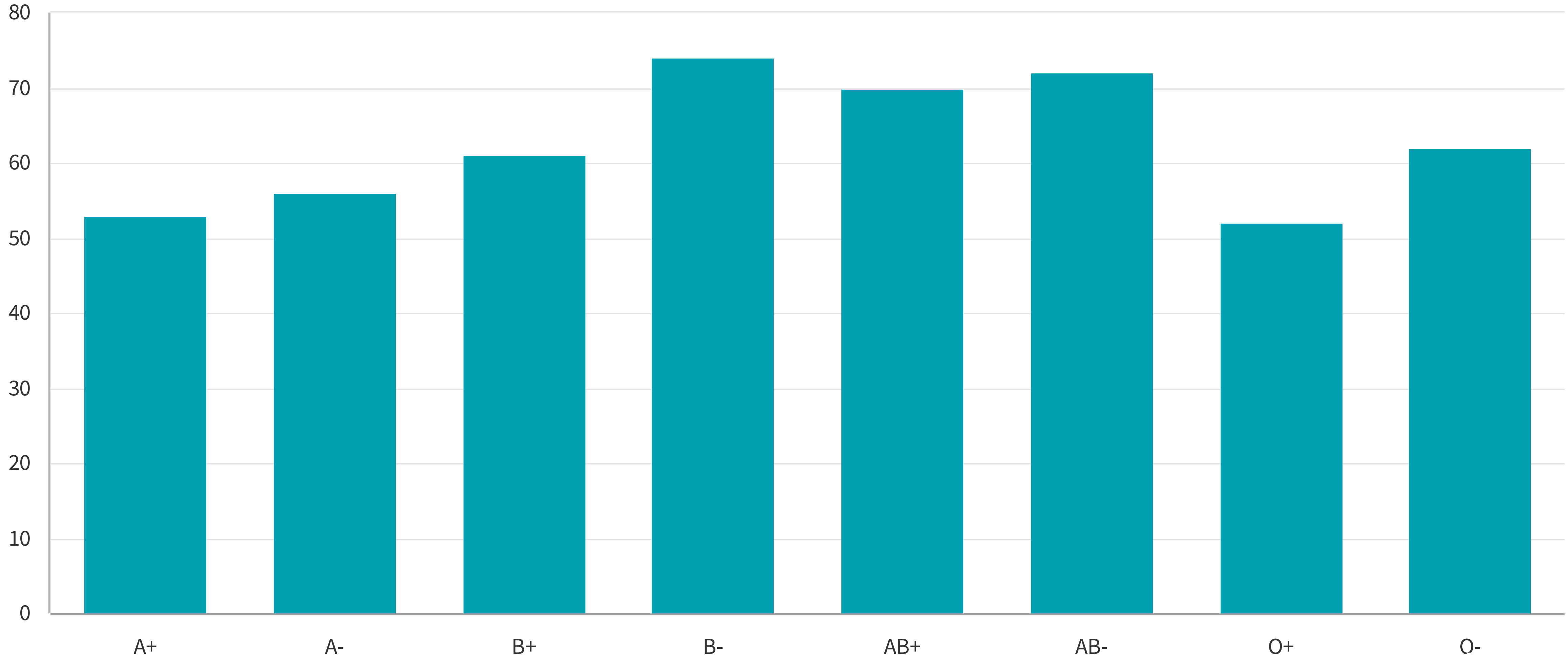
```
SELECT COUNT(b.blood_id), b.blood_type FROM
blood b
 GROUP BY b.blood_type
 ORDER BY COUNT(b.blood_id) DESC;
```

GROUP BY



	COUNT(B.BLOOD_ID)	BLOOD_TYPE
1	74	B-
2	72	AB-
3	70	AB+
4	62	O-
5	61	B+
6	56	A-
7	53	A+
8	52	O+

Blood Donations by Type



MADE WITH beautiful.ai

Final Thoughts

- **Learned**

  Importance of having fully filled out tables

- **Challenges**

  Planning the database

  Lots of factors to consider in tables and how they are related

- **Success**

  Overall Yes

  Gain insights on donors and donees and allocate resources where they can be best used