Database Management System 1
CPS 542 – M3

Database Management System
of
a Blood Bank

Term Project Final Report

By:
Sriram Sandilya Kambhampati
Teja Molaganuru

Under the Guidance of:

DR.CEMIL KIRBAS
CPS 542
DEPARTMENT OF COMPUTER SCIENCE

Database Management System of a Blood Bank

Introduction:

In this innovative concept, a sophisticated database management system plays a pivotal role in enhancing the efficiency and accuracy of managing blood supply levels, donor registrations, recipient records, and the seamless distribution of life-saving blood products. This comprehensive database comprises eight distinct entities and seven intricate relationships, forming a robust foundation for streamlining the entire process. The overarching objective of this system is to provide invaluable support in meticulously tracking and organizing every facet of blood donation, storage, and distribution. Anticipated outcomes include a substantial improvement in donor registration procedures, a finely tuned inventory management system, and a real-time tracking mechanism for blood stock levels. Ultimately, this visionary system is poised to expedite the critical process of delivering safe and timely blood to patients in need, thereby significantly enhancing healthcare outcomes.

The Proposed database consists of the following entities and relationships:

Entities:

Donor: A module used to register new donors and record their personal details and medical background. Information on donor eligibility is also included.

Donor (Name, Age, Gender, Phone Number, date of last donation, Blood type, **Don id**)

Patient: A module for registering patients, including their names, contact information, necessary blood type, and quantity of blood.

Patient (Name, Age, Gender, Blood type, Phone Number, **Pat id**)

Blood Bank: A module for registering all blood banks, together with their current blood supply, blood needs, blood types, and blood volume.

BBank (Bank name, Phone Number, **BBank id**)

Inventory: A module for keeping track of the various blood kinds and quantities that are available for donation.

Inventory (Blood Type, expiry date, Quantity, **Inven Id**)

Facility: A module that is superset for two additional entities to conduct transactions with blood banks and donors is referred to as a facility.

Facility (Name, Phone number, Operating hours, **Fac id**)

Hospital: A subset of facility entity that conducts hospital business.

Hospital (request date, blood type, request Id, quantity)

Blood Camp: A subdivision of a facilities entity that does blood camp business.

BCamp (SDate, EDate)

Employee: A module for managing tasks in the blood bank, facility, and inventory.

Employee (Name, gender, age, phone number, **Emp id**)

Relations:

1. Donor donates to BC to Blood camp.
2. Donor donates to BB to Blood bank.
3. Patients admit to Hospital.
4. Facility requests Blood bank.
5. Blood bank maintains Inventory.
6. Blood bank manages Employee.
7. Employee works at Facility.
8. Employee handles Inventory.

Transactions:

1. Get a list of all blood banks along with the details of employees working there and the facilities they are linked to, including blood banks without any employees.
2. Display all blood type requests from hospitals, including those that have not yet been fulfilled.
3. List all donors and their donation dates at blood camps, including those who have not yet donated.
4. Update contact information for a donor in the Donor table.
5. Find the total quantity of each blood type available in the inventory.
6. Get the names of donors who are older than the average age of all donors.
7. List each blood bank along with the count of donations it has received.
8. Increase the age of all employees working at a specific facility by 1 year.
9. Find the names of all donors who have donated at facilities where a blood camp was held.
10. Retrieve each donor's most recent donation date.

ER Diagram:

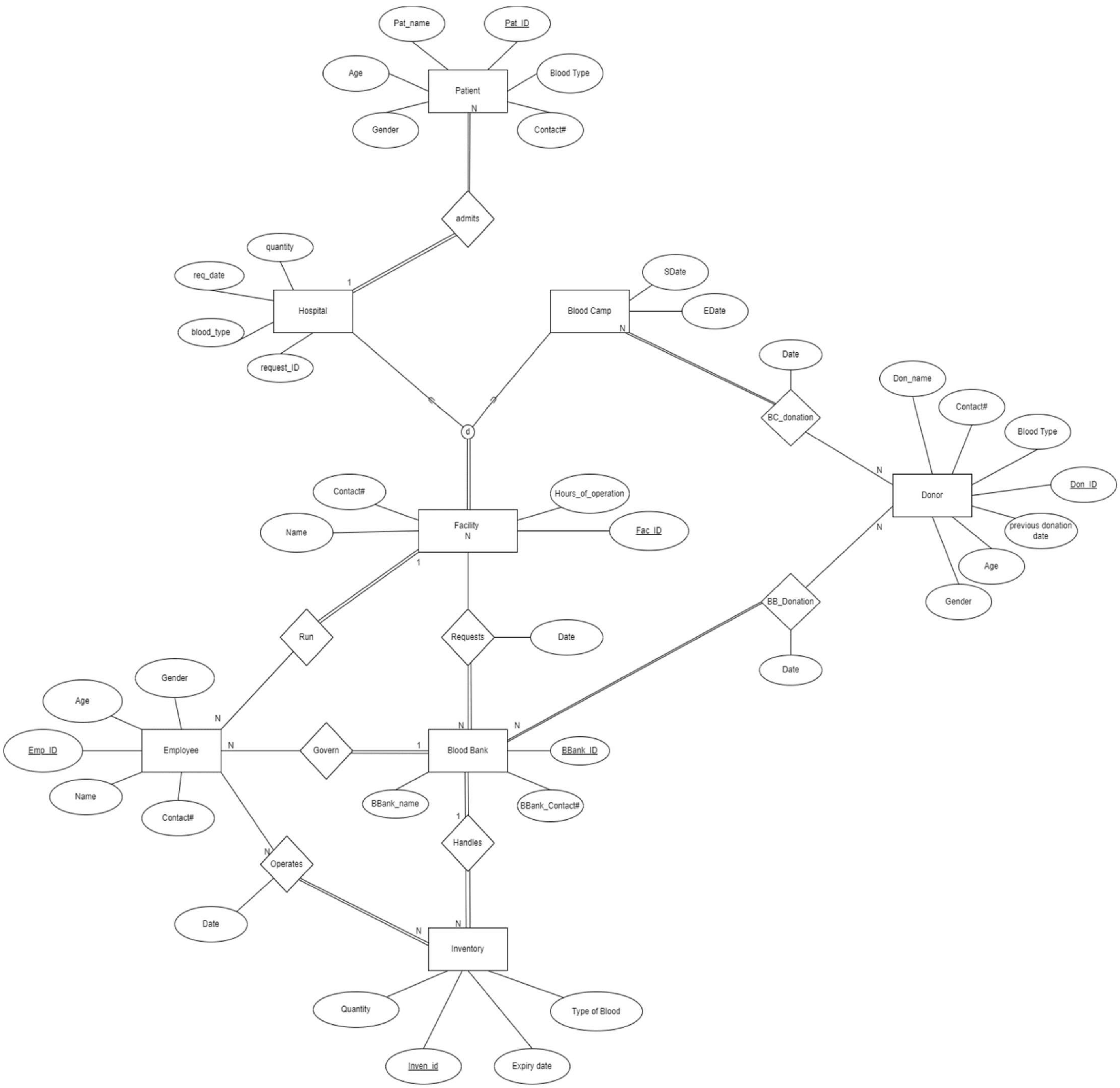
Description:

An Entity-Relationship (ER) diagram in Database Management Systems (DBMS) is a visual representation used to model and describe the structure of a database. It consists of entities, which are objects or concepts in the real world, and the relationships between them. Entities are represented as rectangles, and relationships are depicted as lines connecting these rectangles. Each entity is defined by attributes that describe its properties. ER diagrams help in understanding the database's schema, illustrating how different entities are related, and serving as a blueprint for designing the database schema. They are a vital tool for database designers and developers to ensure data integrity, efficient data retrieval, and a clear understanding of the database's structure.

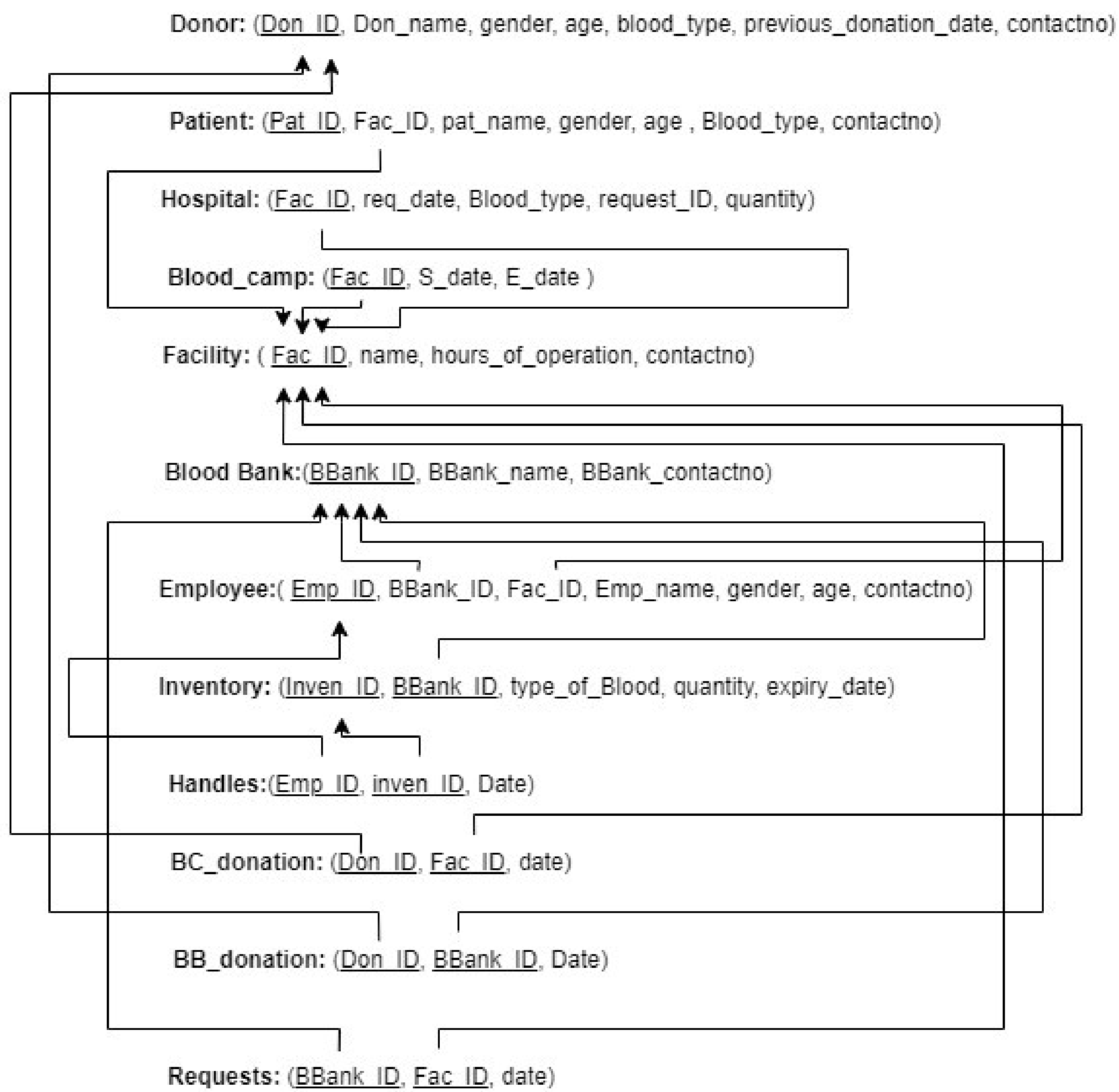
In our database management system for a blood bank, we have the following entities:

- Donor (Name, Age, Gender, Contact Number, date of last donation, Blood type, **Don id**)
- Patient (Name, Age, Gender, Blood type, Contact Number, **Pat id**)
- Blood Bank (Bank name, Contact Number, **BBank id**)
- Inventory (Blood Type, expiry date, Quantity, **Inven Id**)
- Facility (Name, Contact number, Operating hours, **Fac id**)
- Hospital (request date, blood type, request Id, quantity)
- Blood Camp (SDate, EDate)
- Employee (Name, gender, age, Contact number, **Emp id**)

ER Model:


Pat_name
Pat_ID
Age
Patient
Blood Type
N
Gender
Contact#
admits
quantity
req_date
1
Hospital
blood_type
request_ID
SDate
Blood Camp
EDate
N
Date
BC_donation
Don_name
Contact#
Blood Type
d
N
Don_ID
Donor
Contact#
Hours_of_operation
Name
Facility
N
Fac_ID
previous donation date
N
Age
1
BB_Donation
Gender
Run
Requests
Date
Date
Gender
Age
N
N
N
Emp_ID
Employee
N
Govern
1
Blood Bank
BBank_ID
Name
Contact#
BBank_name
BBank_Contact#
1
Handles
N
Operates
Date
N
N
Inventory
Quantity
Type of Blood
Inven_id
Expiry date

Relation Schema of the Database:



Normalization of the Relational Schema:

Normalization is done for minimizing the data redundancy in the database model of the Database Management System of a Blood Bank.

For normalization you need to check if the system is in 1NF, 2NF, 3NF and BCNF.

- **1NF:** A relation is in 1NF if it contains an atomic value. For each multi-valued attribute, create a new table, in which you place the key to the original table and the mutli-valued attribute. Keep the original table, with its key.
- **2NF:** A relation will be 2NF if it is in 1NF, and all non-key attributes are fully functional and dependent on the primary key.
- **3NF:** As it should be in 2NF and every non-prime attribute of the relation is non-transitively dependent on every key in the relation also whenever a non-trivial functional dependency X—A exists, then either X is asuper key or A is a member of some candidate key.
- **BCNF (Boyce codd's normal form):** Stronger than 3NF, it should be in 3NF and whenever a non-trivialfunctional dependency X→A exists, X should be a super key.

Donor Table:

Donor:(<u>Don_ID</u>,Don_name,gender,age,blood_type,Previous_donation_date, contactno)

The above relation's attributes equal the letters {ABCDEFGH} in the following order R (A|B|C|D|E|F|G)

The List of Functional Dependency: A -->BCDEFG

First Normal Form (1NF):

- The above relational model violates 1NF as attribute contactno is a multi-valued attribute. In order to make it atomicwe split the contactno to contactno1 and contactno2.
  Donor**:**(<u>Don_ID</u>,Don_name,gender,age,Blood_Type,contactno1,contactno2, Previous_donation_date, contactno)
- The above relational model is in **1NF**. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Patient Table:

Patient:(Pat_Id, Fac_ID, Pat_name, gender, age, Blood_Type, contactno)
The above relation's attributes equal the letters {ABCDEFG} in the following order R (A|B|C|D|E|F|G)
The List of Functional Dependency: A -->BCDEFG

First Normal Form (1NF):
- The above relational model violates 1NF as attribute contactno is a multi-valued attribute. Inorder to make it atomicwe split the contactno to contactno1 and contactno2.
- Patient:(Pat_Id, Fac_ID,Pat_Name, gender, age, Blood_Type, contactno1,contactno2)
- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):
- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):
- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd’s normal form):
- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Hospital Table:

Hospital:(Fac_ID, req_date, Blood_type, request_ID, quantity)
The above relation's attributes equal the letters {ABCDE} in the following order R (A|B|C|D|E).
The List of Functional Dependency: A -->BCDE

First Normal Form (1NF):
- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):
- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):
- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):
- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Blood Camp Table:

Blood_Camp:(Fac_ID, S_date, E_date)
The above relation's attributes equal the letters {ABCD} in the following order R (A|B|C).
The List of Functional Dependency: A -->BC

First Normal Form (1NF):

- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Facility Table:

Facility:(Fac_ID ,name, hours_of_operation, contactno)
The above relation's attributes equal the letters {ABCD} in the following order R (A|B|C|D).
The List of Functional Dependency: A -->BCD

First Normal Form (1NF):

- The above relational model violates 1NF as attribute contactno is a multi-valued attribute. Inorder to make it atomicwe split the contactno to contactno1 and contactno2
- Facility:(Fac_ID ,name, hours_of_operation, contactno1, contactno2)
- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd’s normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Blood Bank Table:

Blood Bank:(BBank_ID, BBank _name, BBank_contactno, Fac_ID)
The above relation's attributes equal the letters {ABCD} in the following order R (A|B|C|D).
The List of Functional Dependency: A -->BCD

First Normal Form (1NF):

- The above relational model violates 1NF as attribute BB_contatcno is a multi-valued attribute. Inorder to make it atomic we split the BB_contactno to BB_contactno1 and BB_contactno2.
- Blood Bank:(BBank_ID, BBank _name, BBank_contactno1, BBank_contactno2, Fac_ID)
- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes dependon the primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Employee Table:

Employee:(Emp_ID, BBank_ID, Fac_ID, Emp_name, gender, age, contactno)
The above relation's attributes equal the letters {ABCDEFG} in the following order R (A|B|C|D|E|F|G)
The List of Functional Dependency: A -->BCDEFG

First Normal Form (1NF):

- The above relational model violates 1NF as attribute contactno is a multi-valued attribute. Inorder to make it atomicwe split the contactno to contactno1 and contactno2.
- Employee:(Emp_ID, BBank_ID, Fac_ID, Emp_name, gender, age, contactno1, contactno2)
- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Inventory Table:

Inventory:(Inven_ID, BBank_ID, type_of_Blood, quantity, expiry_date)
The above relation's attributes equal the letters {ABCD} in the following order R (A|B|C|D|E)
The List of Functional Dependency: A -->BCDE

First Normal Form (1NF):

- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Handles Table:

Handles:(Emp_ID, invent_ID, Date)
The above relation's attributes equal the letters {ABC} in the following order R (AB|C)
The List of FunctionalDependency: A B-->C

First Normal Form (1NF):

- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

BC_donation Table:

BC_donation:(Don_ID, Fac_ID, Date)
The above relation's attributes equal the letters {ABC} in the following order R (AB|C)
The List of Functional Dependency: A B-->C

First Normal Form (1NF):

- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

BB_donation Table:

BB_donation: (Don_ID, BBank_ID, Date)
The above relation's attributes equal the letters {ABC} in the following order R (AB|C)
The List of Functional Dependency: A B-->C

First Normal Form (1NF):

- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functionaldependencies.

Requests Table:

Requests:(BBank_ID, Fac_ID, Date)
The above relation's attributes equal the letters {ABC} in the following order R (AB|C)
The List of Functional Dependency: A B-->C

First Normal Form (1NF):

- The above relational model is in 1NF. This is because every relation listed above has no multivalued attributes(i.e., atomic) and each relation has a primary key as well as a correct domain type.

Second Normal Form (2NF):

- The above relational model is in 2NF because it has a single candidate key, and all non-key attributes depend onthe primary key.

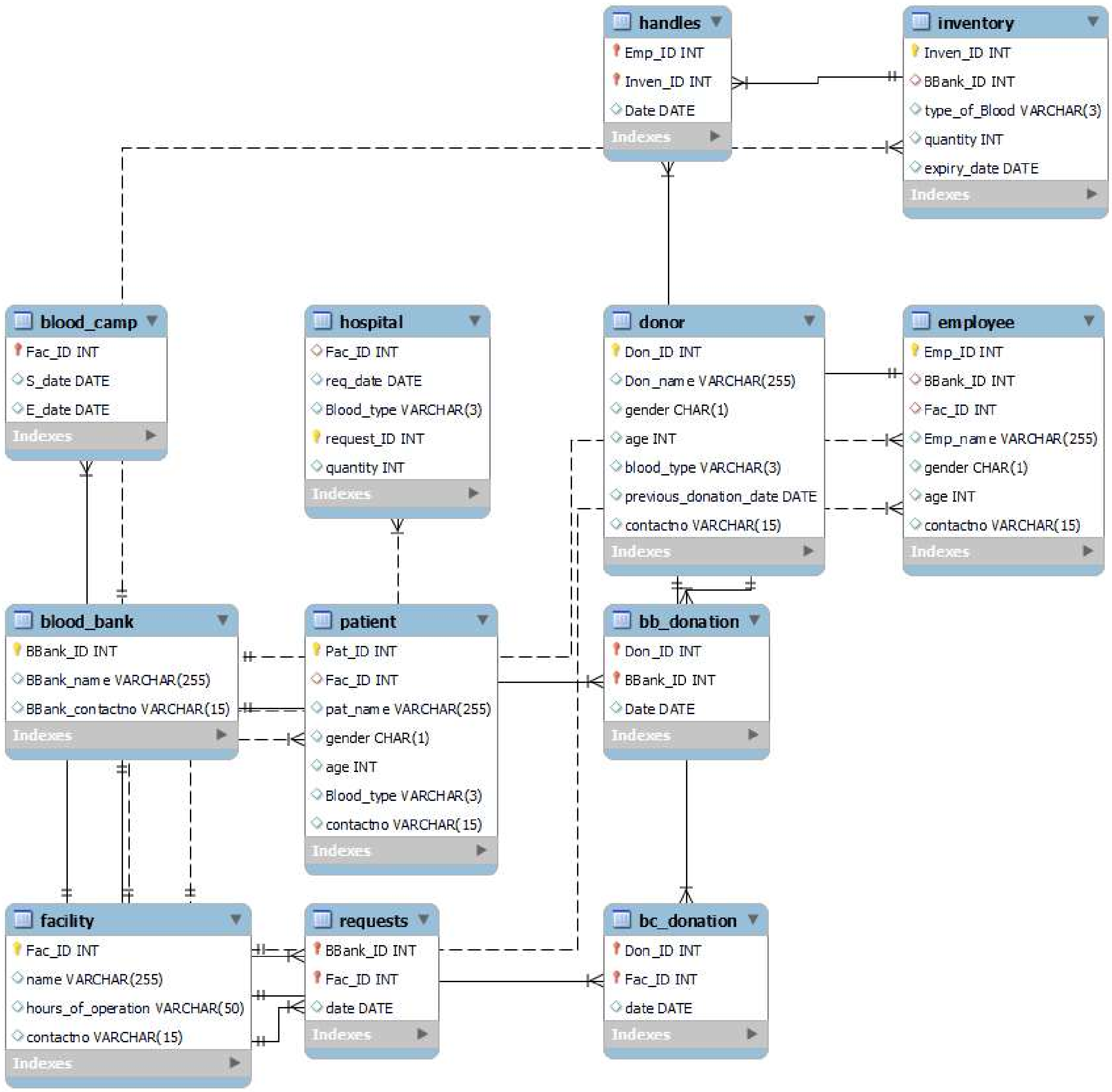
Third Normal Form (3NF):

- Now the Above model is in 3NF as it is in 2NF and every non-prime attribute of the relation is non-transitivelydependent on every key in the relation also whenever a non-trivial functional dependency X→A exists, then either X is a super key or A is a member of some candidate key.

BCNF (Boyce Codd's normal form):

- The above relational model is in BCNF because there are no overlapping candidate keys or functional dependencies.

Logical Model:



Physical Model:

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS,
UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZER
O_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENG
INE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema new_blood_bank
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema new_blood_bank
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `new_blood_bank` DEFAULT
CHARACTER SET utf8 ;
-- -----------------------------------------------------
-- Schema blood_bank
-- -----------------------------------------------------

-- -----------------------------------------------------
```

```
-- Schema blood_bank

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `blood_bank` DEFAULT CHARACTER
SET utf8mb4 COLLATE utf8mb4_0900_ai_ci ;

USE `new_blood_bank` ;


-- -----------------------------------------------------

-- Table `new_blood_bank`.`facility`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `new_blood_bank`.`facility` (

  `Fac_ID` INT NOT NULL AUTO_INCREMENT,

  `name` VARCHAR(45) NULL,

  `hours_of_operation` VARCHAR(45) NULL,

  `contactno1` VARCHAR(45) NULL,

  `contactno2` VARCHAR(45) NULL,

  PRIMARY KEY (`Fac_ID`))

ENGINE = InnoDB;



-- -----------------------------------------------------

-- Table `new_blood_bank`.`Hospital`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Hospital` (

  `req_date` VARCHAR(45) NULL,

  `Blood_type` VARCHAR(45) NULL,

  `request_ID` INT NULL,
```

```
  `quantity` VARCHAR(45) NULL,
  `facility_Fac_ID` INT NOT NULL,
  PRIMARY KEY (`facility_Fac_ID`),
  INDEX `fk_Hospital_facility1_idx` (`facility_Fac_ID` ASC) VISIBLE,
  CONSTRAINT `fk_Hospital_facility1`
    FOREIGN KEY (`facility_Fac_ID`)
    REFERENCES `new_blood_bank`.`facility` (`Fac_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `new_blood_bank`.`patient`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`patient` (
  `pat_ID` INT NOT NULL AUTO_INCREMENT,
  `Pat_Name` VARCHAR(45) NULL,
  `Gender` VARCHAR(45) NULL,
  `Age` INT NULL,
  `Blood_Type` VARCHAR(45) NULL,
  `contactno1` VARCHAR(45) NULL,
  `contactno2` VARCHAR(45) NULL,
  `Fac_ID` INT NOT NULL,
  `Hospital_facility_Fac_ID` INT NOT NULL,
```

```sql
  PRIMARY KEY (`pat_ID`),
  INDEX `fk_patient_Hospital1_idx` (`Hospital_facility_Fac_ID` ASC) VISIBLE,
  CONSTRAINT `fk_patient_Hospital1`
    FOREIGN KEY (`Hospital_facility_Fac_ID`)
    REFERENCES `new_blood_bank`.`Hospital` (`facility_Fac_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `new_blood_bank`.`Donor`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Donor` (
  `Don_ID` INT NOT NULL AUTO_INCREMENT,
  `Don_name` VARCHAR(45) NULL,
  `gender` VARCHAR(45) NULL,
  `age` INT NULL,
  `blood_type` VARCHAR(45) NULL,
  `contactno1` VARCHAR(45) NULL,
  `contactno2` VARCHAR(45) NULL,
  `Previous_donation_date` VARCHAR(45) NULL,
  PRIMARY KEY (`Don_ID`))
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `new_blood_bank`.`Facility`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Facility` (
  `Fac_ID` INT NOT NULL AUTO_INCREMENT,
  `Name` VARCHAR(45) NULL,
  `Hours_of_operation` VARCHAR(45) NULL,
  `Contactno1` VARCHAR(45) NULL,
  `Contactno2` VARCHAR(45) NULL,
  PRIMARY KEY (`Fac_ID`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `new_blood_bank`.`Blood_Bank`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Blood_Bank` (
  `BBank_ID` INT NOT NULL AUTO_INCREMENT,
  `BBank_name` VARCHAR(45) NULL,
  `BBank_contactno1` VARCHAR(45) NULL,
  `BBank_contactno2` VARCHAR(45) NULL,
  PRIMARY KEY (`BBank_ID`))
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `new_blood_bank`.`Employee`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Employee` (
  `Emp_ID` INT NOT NULL AUTO_INCREMENT,
  `Emp_name` VARCHAR(45) NULL,
  `gender` VARCHAR(45) NULL,
  `age` INT NULL,
  `contactno1` VARCHAR(45) NULL,
  `contactno2` VARCHAR(45) NULL,
  `facility_Fac_ID` INT NOT NULL,
  `Blood_Bank_BBank_ID` INT NOT NULL,
  PRIMARY KEY (`Emp_ID`),
  INDEX `fk_Employee_facility1_idx` (`facility_Fac_ID` ASC) VISIBLE,
  INDEX `fk_Employee_Blood_Bank1_idx` (`Blood_Bank_BBank_ID` ASC)
VISIBLE,
  CONSTRAINT `fk_Employee_facility1`
    FOREIGN KEY (`facility_Fac_ID`)
    REFERENCES `new_blood_bank`.`facility` (`Fac_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Employee_Blood_Bank1`
    FOREIGN KEY (`Blood_Bank_BBank_ID`)
    REFERENCES `new_blood_bank`.`Blood_Bank` (`BBank_ID`)
    ON DELETE NO ACTION
```

```
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `new_blood_bank`.`Inventory`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Inventory` (
  `Inven_ID` INT NOT NULL AUTO_INCREMENT,
  `type_of_Blood` VARCHAR(45) NULL,
  `quantity` INT NULL,
  `expiry_date` DATE NULL,
  `Blood_Bank_BBank_ID` INT NOT NULL,
  PRIMARY KEY (`Inven_ID`),
  INDEX `fk_Inventory_Blood_Bank1_idx` (`Blood_Bank_BBank_ID` ASC)
VISIBLE,
  CONSTRAINT `fk_Inventory_Blood_Bank1`
    FOREIGN KEY (`Blood_Bank_BBank_ID`)
    REFERENCES `new_blood_bank`.`Blood_Bank` (`BBank_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `new_blood_bank`.`Donor_Blood_Bank`
```

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Donor_Blood_Bank` (
  `Donor_ID` INT NOT NULL,
  `BBank_ID` INT NOT NULL,
  `Date` DATE NULL,
  PRIMARY KEY (`Donor_ID`, `BBank_ID`),
  INDEX `fk_Donor_has_Blood_Bank_Blood_Bank1_idx` (`BBank_ID` ASC)
VISIBLE,
  INDEX `fk_Donor_has_Blood_Bank_Donor1_idx` (`Donor_ID` ASC) VISIBLE,
  CONSTRAINT `fk_Donor_has_Blood_Bank_Donor1`
    FOREIGN KEY (`Donor_ID`)
    REFERENCES `new_blood_bank`.`Donor` (`Don_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Donor_has_Blood_Bank_Blood_Bank1`
    FOREIGN KEY (`BBank_ID`)
    REFERENCES `new_blood_bank`.`Blood_Bank` (`BBank_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `new_blood_bank`.`Requests`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Requests` (
```

```
  `facility_ID` INT NOT NULL,
  `Blood_Bank_BBank_ID` INT NOT NULL,
  `Date` DATE NULL,
  PRIMARY KEY (`facility_ID`, `Blood_Bank_BBank_ID`),
  INDEX `fk_facility_has_Blood_Bank_Blood_Bank1_idx`
(`Blood_Bank_BBank_ID` ASC) VISIBLE,
  INDEX `fk_facility_has_Blood_Bank_facility1_idx` (`facility_ID` ASC)
VISIBLE,
  CONSTRAINT `fk_facility_has_Blood_Bank_facility1`
    FOREIGN KEY (`facility_ID`)
    REFERENCES `new_blood_bank`.`facility` (`Fac_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_facility_has_Blood_Bank_Blood_Bank1`
    FOREIGN KEY (`Blood_Bank_BBank_ID`)
    REFERENCES `new_blood_bank`.`Blood_Bank` (`BBank_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `new_blood_bank`.`Blood_camp`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Blood_camp` (
  `S_date` VARCHAR(45) NULL,
```

```
  `E_date` VARCHAR(45) NULL,
  `facility_Fac_ID` INT NOT NULL,
  PRIMARY KEY (`facility_Fac_ID`),
  INDEX `fk_Blood_camp_facility1_idx` (`facility_Fac_ID` ASC) VISIBLE,
  CONSTRAINT `fk_Blood_camp_facility1`
    FOREIGN KEY (`facility_Fac_ID`)
    REFERENCES `new_blood_bank`.`facility` (`Fac_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `new_blood_bank`.`BC_donation`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`BC_donation` (
  `Date` DATE NULL,
  `Donor_Don_ID` INT NOT NULL,
  `Blood_camp_facility_Fac_ID` INT NOT NULL,
  PRIMARY KEY (`Donor_Don_ID`, `Blood_camp_facility_Fac_ID`),
  INDEX `fk_BC_donation_Donor1_idx` (`Donor_Don_ID` ASC) VISIBLE,
  INDEX `fk_BC_donation_Blood_camp1_idx` (`Blood_camp_facility_Fac_ID`
ASC) VISIBLE,
  CONSTRAINT `fk_BC_donation_Donor1`
    FOREIGN KEY (`Donor_Don_ID`)
    REFERENCES `new_blood_bank`.`Donor` (`Don_ID`)
```

```
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_BC_donation_Blood_camp1`
    FOREIGN KEY (`Blood_camp_facility_Fac_ID`)
    REFERENCES `new_blood_bank`.`Blood_camp` (`facility_Fac_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `new_blood_bank`.`Handles`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `new_blood_bank`.`Handles` (
  `Date` DATE NULL,
  `Employee_Emp_ID` INT NOT NULL,
  `Inventory_Inven_ID` INT NOT NULL,
  PRIMARY KEY (`Employee_Emp_ID`, `Inventory_Inven_ID`),
  INDEX `fk_Handles_Employee1_idx` (`Employee_Emp_ID` ASC) VISIBLE,
  INDEX `fk_Handles_Inventory1_idx` (`Inventory_Inven_ID` ASC) VISIBLE,
  CONSTRAINT `fk_Handles_Employee1`
    FOREIGN KEY (`Employee_Emp_ID`)
    REFERENCES `new_blood_bank`.`Employee` (`Emp_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
```

```sql
  CONSTRAINT `fk_Handles_Inventory1`
    FOREIGN KEY (`Inventory_Inven_ID`)
    REFERENCES `new_blood_bank`.`Inventory` (`Inven_ID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

USE `blood_bank` ;

-- -----------------------------------------------------
-- Table `blood_bank`.`donor`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`donor` (
  `Don_ID` INT NOT NULL,
  `Don_name` VARCHAR(255) NULL DEFAULT NULL,
  `gender` CHAR(1) NULL DEFAULT NULL,
  `age` INT NULL DEFAULT NULL,
  `blood_type` VARCHAR(3) NULL DEFAULT NULL,
  `previous_donation_date` DATE NULL DEFAULT NULL,
  `contactno` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`Don_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
```

```
-- -----------------------------------------------------
-- Table `blood_bank`.`blood_bank`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`blood_bank` (
  `BBank_ID` INT NOT NULL,
  `BBank_name` VARCHAR(255) NULL DEFAULT NULL,
  `BBank_contactno` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`BBank_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`bb_donation`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`bb_donation` (
  `Don_ID` INT NOT NULL,
  `BBank_ID` INT NOT NULL,
  `Date` DATE NULL DEFAULT NULL,
  PRIMARY KEY (`Don_ID`, `BBank_ID`),
  INDEX `BBank_ID` (`BBank_ID` ASC) VISIBLE,
  CONSTRAINT `bb_donation_ibfk_1`
    FOREIGN KEY (`Don_ID`)
```

```
    REFERENCES `blood_bank`.`donor` (`Don_ID`),
  CONSTRAINT `bb_donation_ibfk_2`
    FOREIGN KEY (`BBank_ID`)
    REFERENCES `blood_bank`.`blood_bank` (`BBank_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`facility`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`facility` (
  `Fac_ID` INT NOT NULL,
  `name` VARCHAR(255) NULL DEFAULT NULL,
  `hours_of_operation` VARCHAR(50) NULL DEFAULT NULL,
  `contactno` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`Fac_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`bc_donation`
```

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`bc_donation` (
  `Don_ID` INT NOT NULL,
  `Fac_ID` INT NOT NULL,
  `date` DATE NULL DEFAULT NULL,
  PRIMARY KEY (`Don_ID`, `Fac_ID`),
  INDEX `Fac_ID` (`Fac_ID` ASC) VISIBLE,
  CONSTRAINT `bc_donation_ibfk_1`
    FOREIGN KEY (`Don_ID`)
    REFERENCES `blood_bank`.`donor` (`Don_ID`),
  CONSTRAINT `bc_donation_ibfk_2`
    FOREIGN KEY (`Fac_ID`)
    REFERENCES `blood_bank`.`facility` (`Fac_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`blood_camp`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`blood_camp` (
  `Fac_ID` INT NOT NULL,
  `S_date` DATE NULL DEFAULT NULL,
  `E_date` DATE NULL DEFAULT NULL,
```

```
  PRIMARY KEY (`Fac_ID`),
  CONSTRAINT `blood_camp_ibfk_1`
    FOREIGN KEY (`Fac_ID`)
    REFERENCES `blood_bank`.`facility` (`Fac_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;



-- -----------------------------------------------------
-- Table `blood_bank`.`employee`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`employee` (
  `Emp_ID` INT NOT NULL,
  `BBank_ID` INT NULL DEFAULT NULL,
  `Fac_ID` INT NULL DEFAULT NULL,
  `Emp_name` VARCHAR(255) NULL DEFAULT NULL,
  `gender` CHAR(1) NULL DEFAULT NULL,
  `age` INT NULL DEFAULT NULL,
  `contactno` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`Emp_ID`),
  INDEX `BBank_ID` (`BBank_ID` ASC) VISIBLE,
  INDEX `Fac_ID` (`Fac_ID` ASC) VISIBLE,
  CONSTRAINT `employee_ibfk_1`
    FOREIGN KEY (`BBank_ID`)
```

```
    REFERENCES `blood_bank`.`blood_bank` (`BBank_ID`),
  CONSTRAINT `employee_ibfk_2`
    FOREIGN KEY (`Fac_ID`)
    REFERENCES `blood_bank`.`facility` (`Fac_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`inventory`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`inventory` (
  `Inven_ID` INT NOT NULL,
  `BBank_ID` INT NULL DEFAULT NULL,
  `type_of_Blood` VARCHAR(3) NULL DEFAULT NULL,
  `quantity` INT NULL DEFAULT NULL,
  `expiry_date` DATE NULL DEFAULT NULL,
  PRIMARY KEY (`Inven_ID`),
  INDEX `BBank_ID` (`BBank_ID` ASC) VISIBLE,
  CONSTRAINT `inventory_ibfk_1`
    FOREIGN KEY (`BBank_ID`)
    REFERENCES `blood_bank`.`blood_bank` (`BBank_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
```

```sql
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`handles`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`handles` (
  `Emp_ID` INT NOT NULL,
  `Inven_ID` INT NOT NULL,
  `Date` DATE NULL DEFAULT NULL,
  PRIMARY KEY (`Emp_ID`, `Inven_ID`),
  INDEX `Inven_ID` (`Inven_ID` ASC) VISIBLE,
  CONSTRAINT `handles_ibfk_1`
    FOREIGN KEY (`Emp_ID`)
    REFERENCES `blood_bank`.`employee` (`Emp_ID`),
  CONSTRAINT `handles_ibfk_2`
    FOREIGN KEY (`Inven_ID`)
    REFERENCES `blood_bank`.`inventory` (`Inven_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`hospital`
```

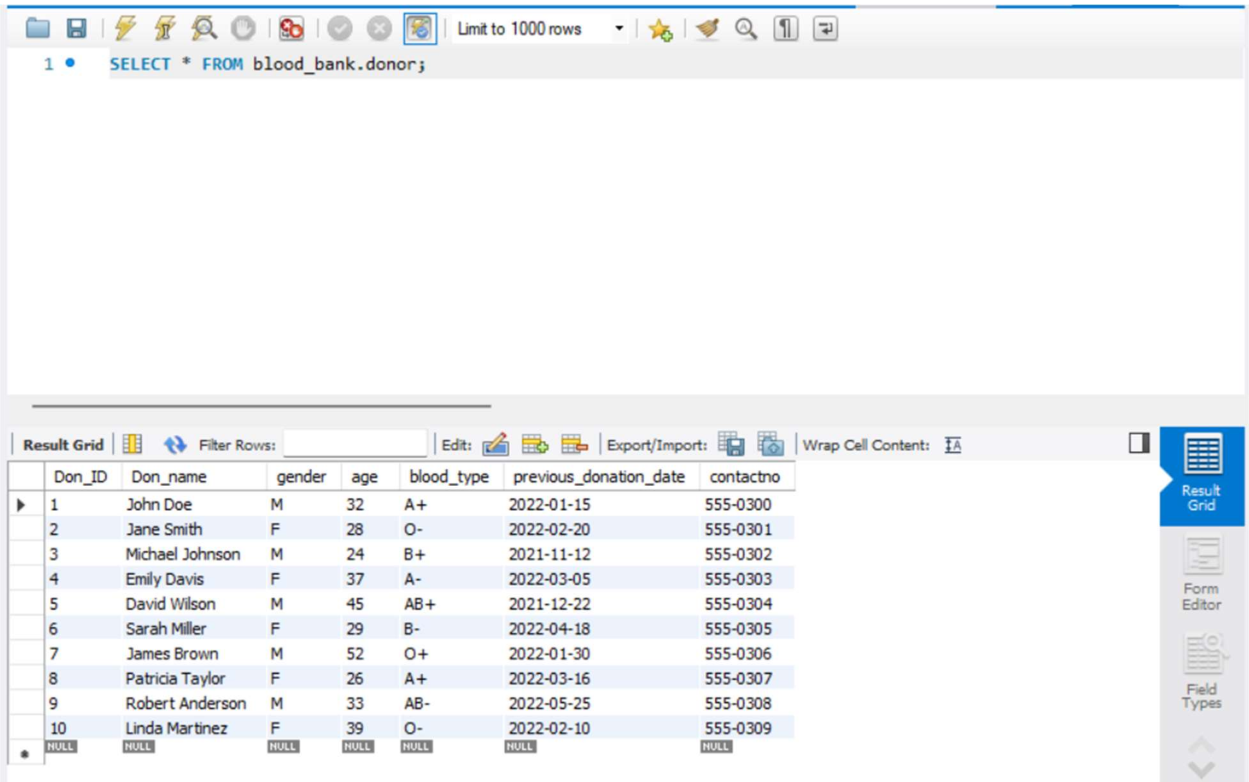
```sql
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`hospital` (
  `Fac_ID` INT NULL DEFAULT NULL,
  `req_date` DATE NULL DEFAULT NULL,
  `Blood_type` VARCHAR(3) NULL DEFAULT NULL,
  `request_ID` INT NOT NULL,
  `quantity` INT NULL DEFAULT NULL,
  PRIMARY KEY (`request_ID`),
  INDEX `Fac_ID` (`Fac_ID` ASC) VISIBLE,
  CONSTRAINT `hospital_ibfk_1`
    FOREIGN KEY (`Fac_ID`)
    REFERENCES `blood_bank`.`facility` (`Fac_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`patient`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`patient` (
  `Pat_ID` INT NOT NULL,
  `Fac_ID` INT NULL DEFAULT NULL,
  `pat_name` VARCHAR(255) NULL DEFAULT NULL,
  `gender` CHAR(1) NULL DEFAULT NULL,
```

```
  `age` INT NULL DEFAULT NULL,
  `Blood_type` VARCHAR(3) NULL DEFAULT NULL,
  `contactno` VARCHAR(15) NULL DEFAULT NULL,
  PRIMARY KEY (`Pat_ID`),
  INDEX `Fac_ID` (`Fac_ID` ASC) VISIBLE,
  CONSTRAINT `patient_ibfk_1`
    FOREIGN KEY (`Fac_ID`)
    REFERENCES `blood_bank`.`facility` (`Fac_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


-- -----------------------------------------------------
-- Table `blood_bank`.`requests`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `blood_bank`.`requests` (
  `BBank_ID` INT NOT NULL,
  `Fac_ID` INT NOT NULL,
  `date` DATE NULL DEFAULT NULL,
  PRIMARY KEY (`BBank_ID`, `Fac_ID`),
  INDEX `Fac_ID` (`Fac_ID` ASC) VISIBLE,
  CONSTRAINT `requests_ibfk_1`
    FOREIGN KEY (`BBank_ID`)
    REFERENCES `blood_bank`.`blood_bank` (`BBank_ID`),
```

```
  CONSTRAINT `requests_ibfk_2`
    FOREIGN KEY (`Fac_ID`)
    REFERENCES `blood_bank`.`facility` (`Fac_ID`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

TABLE VIEWS:

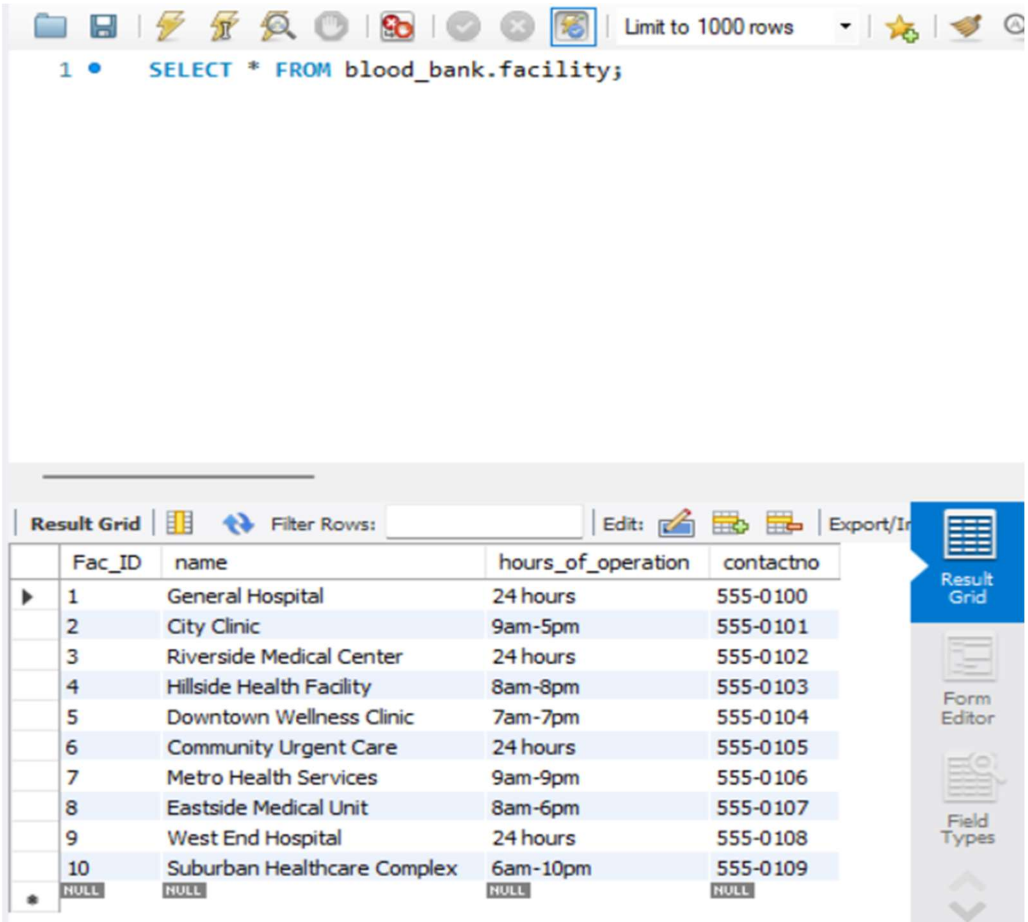
DONOR




```
1 SELECT * FROM blood_bank.donor;
```

Don_ID	Don_name	gender	age	blood_type	previous_donation_date	contactno
1	John Doe	M	32	A+	2022-01-15	555-0300
2	Jane Smith	F	28	O-	2022-02-20	555-0301
3	Michael Johnson	M	24	B+	2021-11-12	555-0302
4	Emily Davis	F	37	A-	2022-03-05	555-0303
5	David Wilson	M	45	AB+	2021-12-22	555-0304
6	Sarah Miller	F	29	B-	2022-04-18	555-0305
7	James Brown	M	52	O+	2022-01-30	555-0306
8	Patricia Taylor	F	26	A+	2022-03-16	555-0307
9	Robert Anderson	M	33	AB-	2022-05-25	555-0308
10	Linda Martinez	F	39	O-	2022-02-10	555-0309
NULL	NULL	NULL	NULL	NULL	NULL	NULL

FACILITY



```sql
SELECT * FROM blood_bank.facility;
```

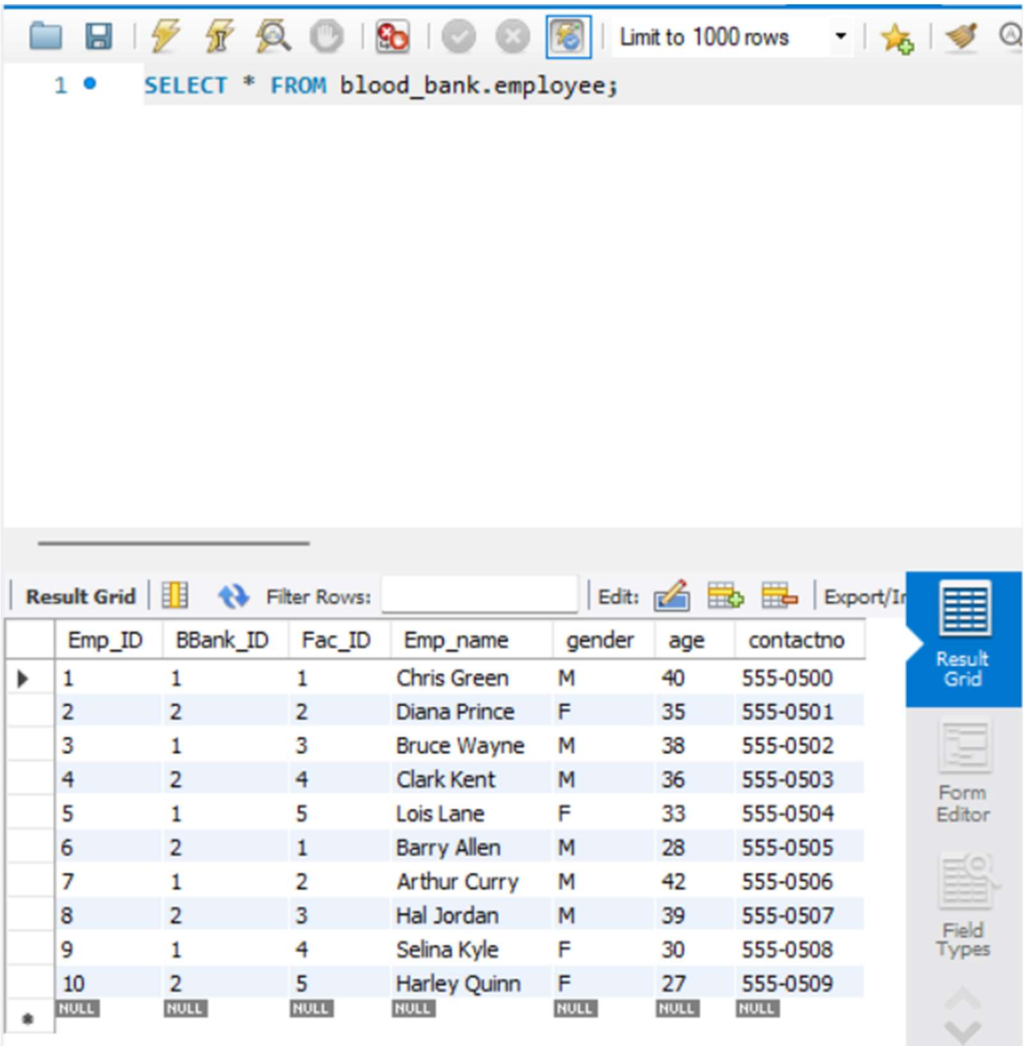
Fac_ID	name	hours_of_operation	contactno
1	General Hospital	24 hours	555-0100
2	City Clinic	9am-5pm	555-0101
3	Riverside Medical Center	24 hours	555-0102
4	Hillside Health Facility	8am-8pm	555-0103
5	Downtown Wellness Clinic	7am-7pm	555-0104
6	Community Urgent Care	24 hours	555-0105
7	Metro Health Services	9am-9pm	555-0106
8	Eastside Medical Unit	8am-6pm	555-0107
9	West End Hospital	24 hours	555-0108
10	Suburban Healthcare Complex	6am-10pm	555-0109
NULL	NULL	NULL	NULL

BLOOD BANK

```
SELECT * FROM blood_bank.blood_bank;
```

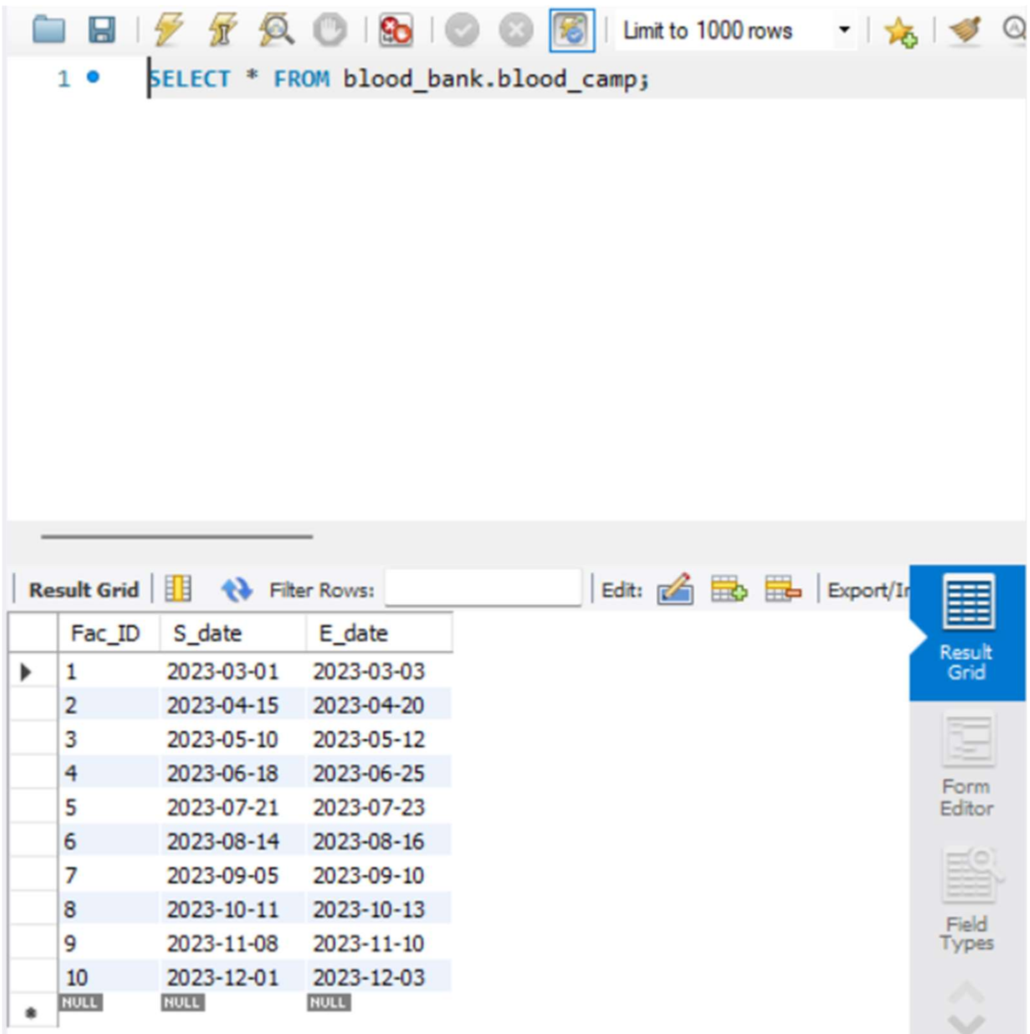
BBank_ID	BBank_name	BBank_contactno
1	Red Hope Blood Center	555-0200
2	LifeStream Blood Bank	555-0201
3	Vital Blood Services	555-0202
4	Harmony Blood Foundation	555-0203
5	Pioneer Bloodworks	555-0204
6	Guardian Blood Care	555-0205
7	Trust Blood Bank	555-0206
8	Unity Blood Network	555-0207
9	Legacy Blood Institute	555-0208
10	Caring Blood Donors	555-0209
NULL	NULL	NULL

EMPLOYEE



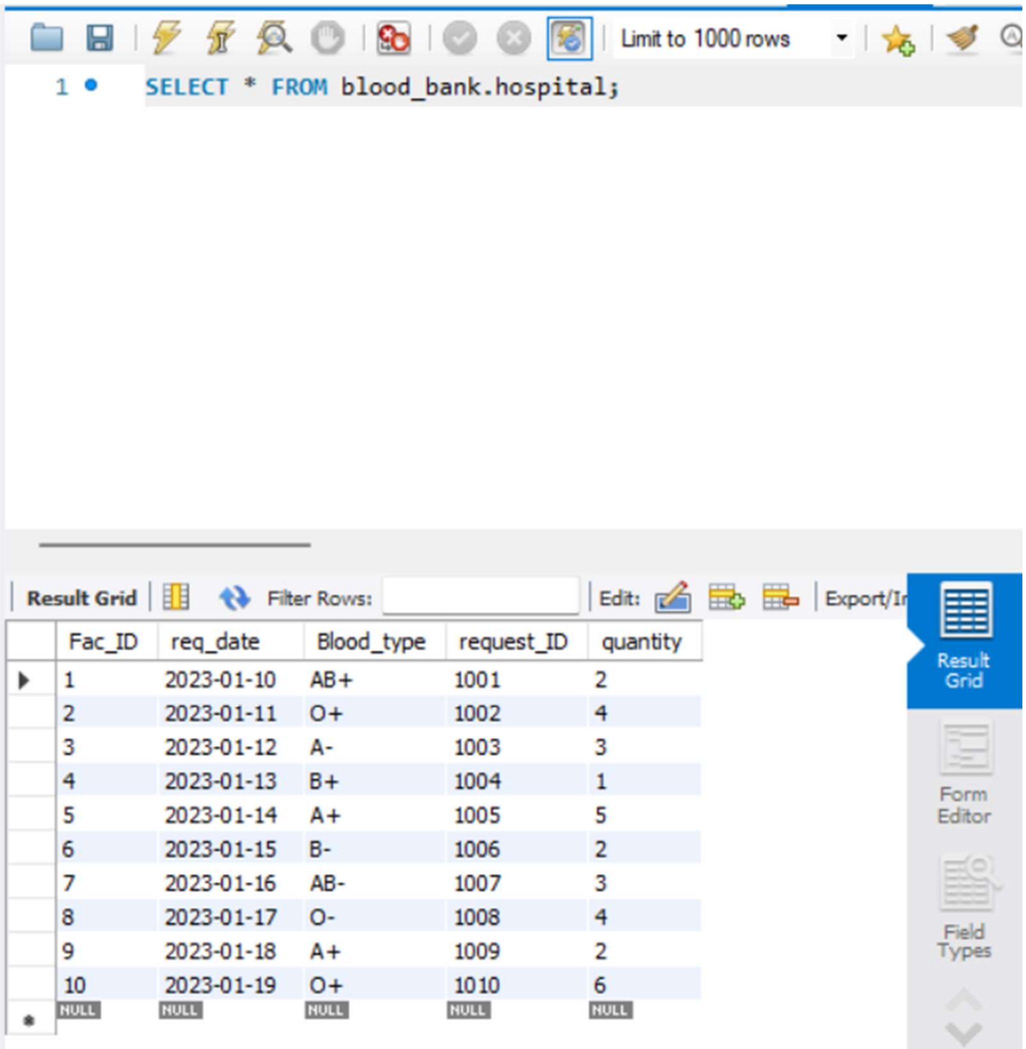
Emp_ID	BBank_ID	Fac_ID	Emp_name	gender	age	contactno
1	1	1	Chris Green	M	40	555-0500
2	2	2	Diana Prince	F	35	555-0501
3	1	3	Bruce Wayne	M	38	555-0502
4	2	4	Clark Kent	M	36	555-0503
5	1	5	Lois Lane	F	33	555-0504
6	2	1	Barry Allen	M	28	555-0505
7	1	2	Arthur Curry	M	42	555-0506
8	2	3	Hal Jordan	M	39	555-0507
9	1	4	Selina Kyle	F	30	555-0508
10	2	5	Harley Quinn	F	27	555-0509
NULL	NULL	NULL	NULL	NULL	NULL	NULL

BLOOD CAMP

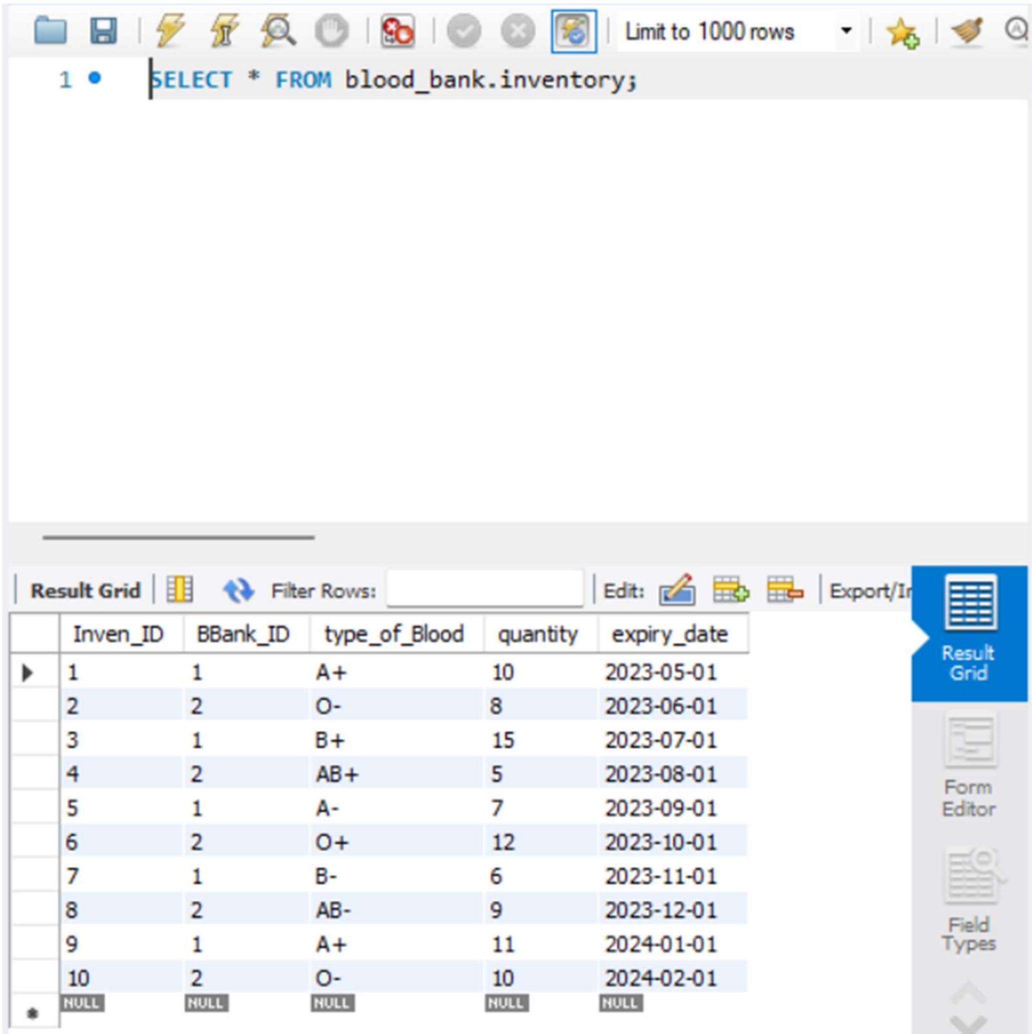


Fac_ID	S_date	E_date
1	2023-03-01	2023-03-03
2	2023-04-15	2023-04-20
3	2023-05-10	2023-05-12
4	2023-06-18	2023-06-25
5	2023-07-21	2023-07-23
6	2023-08-14	2023-08-16
7	2023-09-05	2023-09-10
8	2023-10-11	2023-10-13
9	2023-11-08	2023-11-10
10	2023-12-01	2023-12-03
NULL	NULL	NULL

HOSPITAL

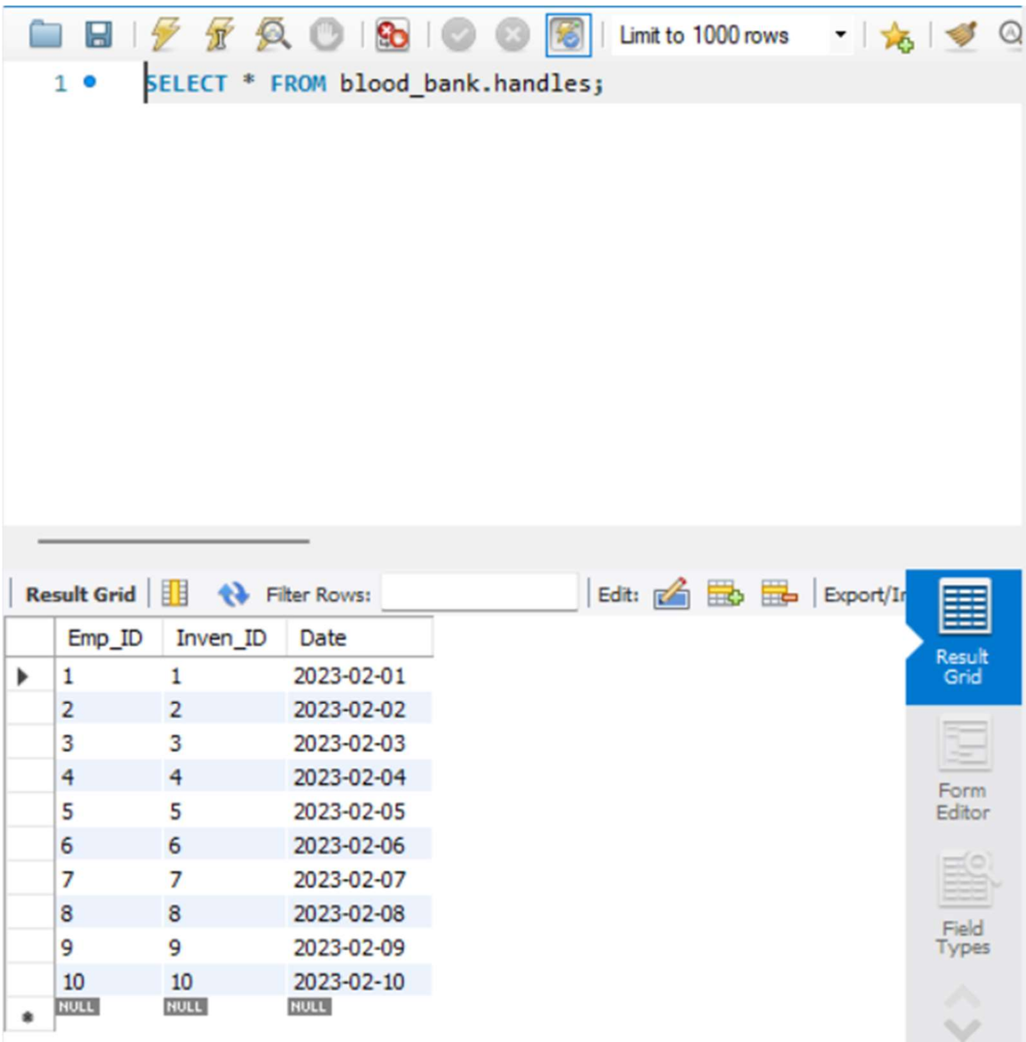

Limit to 1000 rows
1 SELECT * FROM blood_bank.hospital;
Result Grid
Filter Rows:
Edit:
Export/I
Fac_ID req_date Blood_type request_ID quantity
1 2023-01-10 AB+ 1001 2
2 2023-01-11 O+ 1002 4
3 2023-01-12 A- 1003 3
4 2023-01-13 B+ 1004 1
5 2023-01-14 A+ 1005 5
6 2023-01-15 B- 1006 2
7 2023-01-16 AB- 1007 3
8 2023-01-17 O- 1008 4
9 2023-01-18 A+ 1009 2
10 2023-01-19 O+ 1010 6
NULL NULL NULL NULL NULL
Result Grid
Form Editor
Field Types

INVENTORY



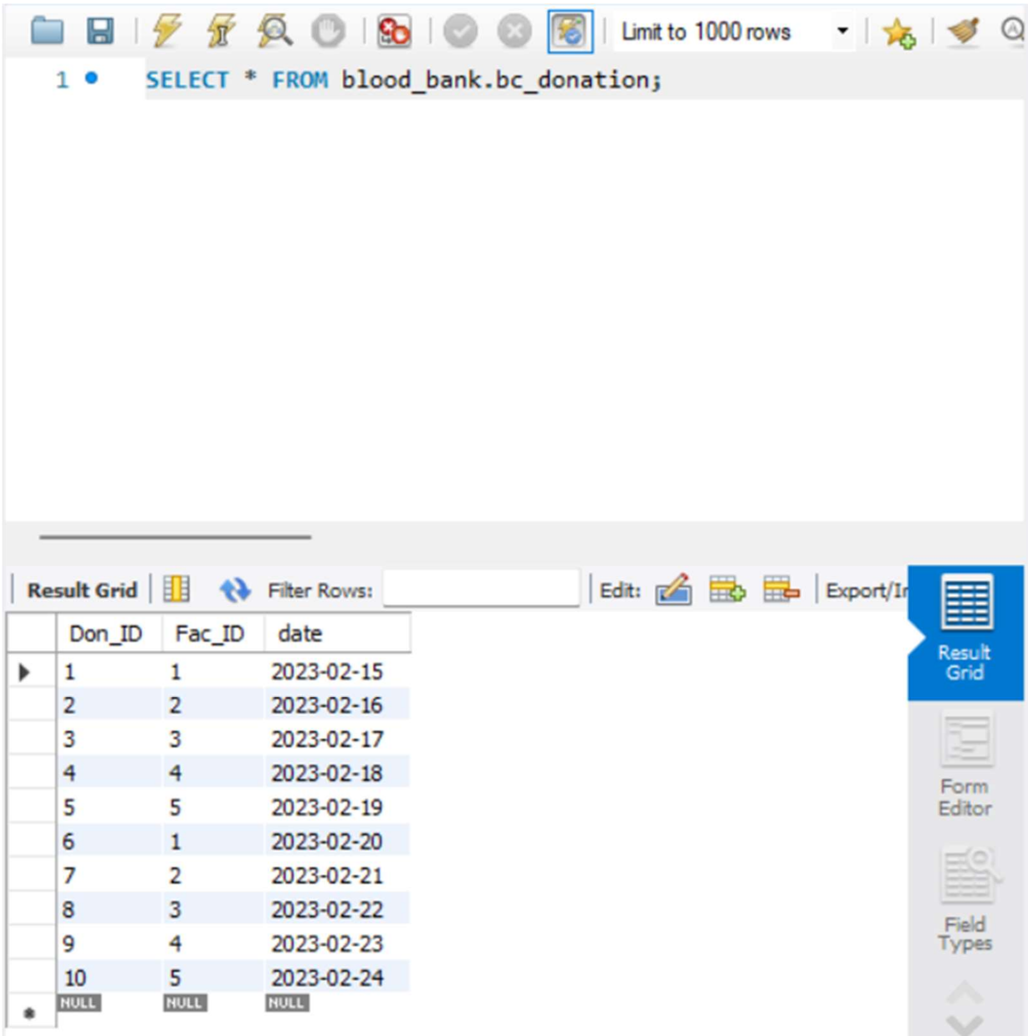
Inven_ID	BBank_ID	type_of_Blood	quantity	expiry_date
1	1	A+	10	2023-05-01
2	2	O-	8	2023-06-01
3	1	B+	15	2023-07-01
4	2	AB+	5	2023-08-01
5	1	A-	7	2023-09-01
6	2	O+	12	2023-10-01
7	1	B-	6	2023-11-01
8	2	AB-	9	2023-12-01
9	1	A+	11	2024-01-01
10	2	O-	10	2024-02-01
NULL	NULL	NULL	NULL	NULL

HANDLES



Emp_ID	Inven_ID	Date
1	1	2023-02-01
2	2	2023-02-02
3	3	2023-02-03
4	4	2023-02-04
5	5	2023-02-05
6	6	2023-02-06
7	7	2023-02-07
8	8	2023-02-08
9	9	2023-02-09
10	10	2023-02-10
NULL	NULL	NULL

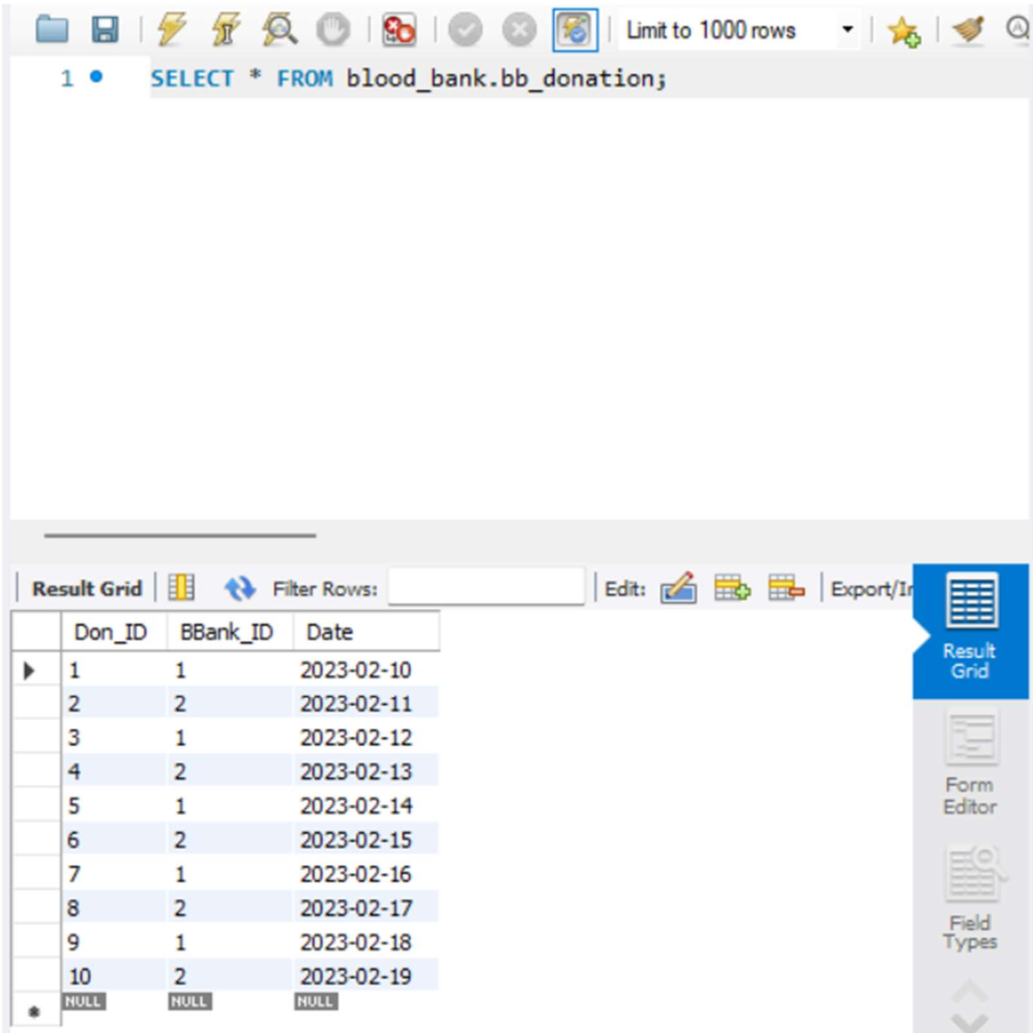
DONATES_TO_BC



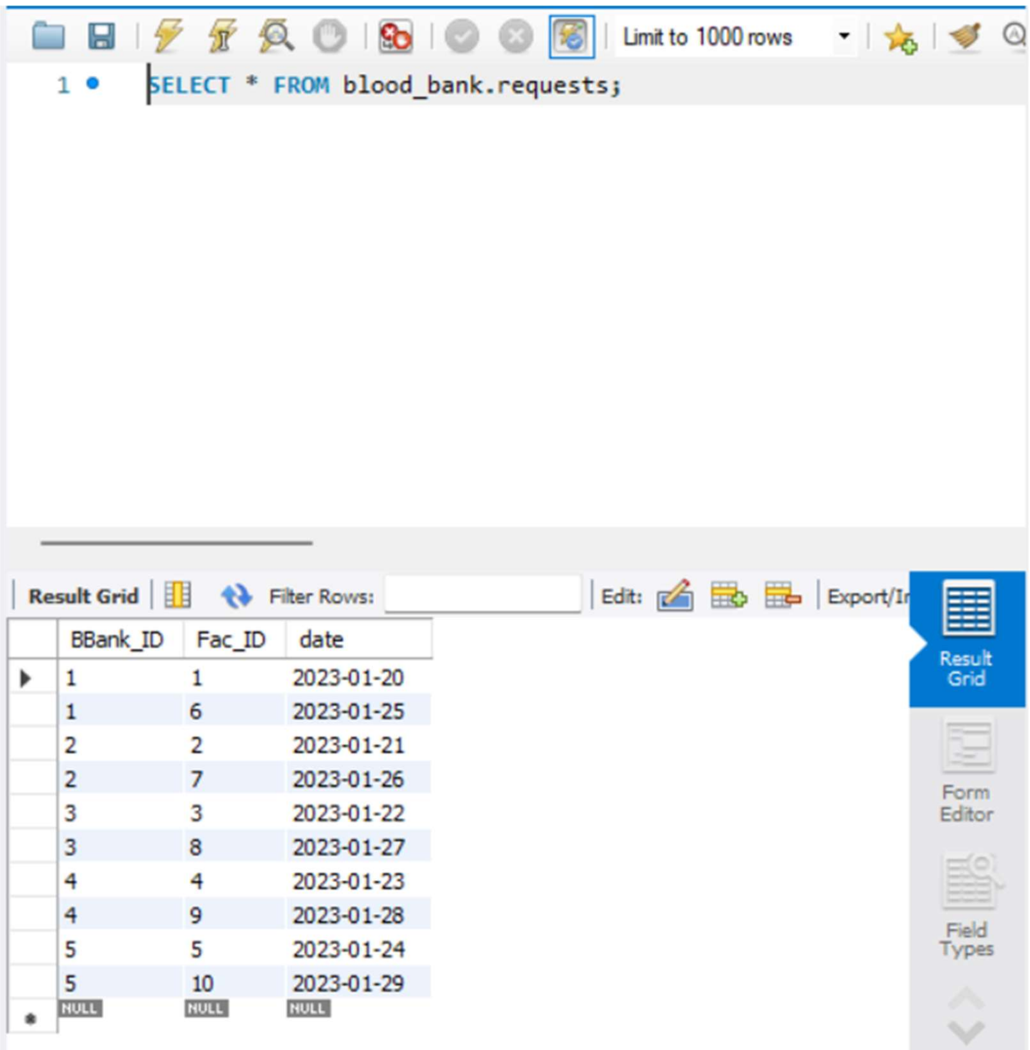

```
1  SELECT * FROM blood_bank.bc_donation;
```

Don_ID	Fac_ID	date
1	1	2023-02-15
2	2	2023-02-16
3	3	2023-02-17
4	4	2023-02-18
5	5	2023-02-19
6	1	2023-02-20
7	2	2023-02-21
8	3	2023-02-22
9	4	2023-02-23
10	5	2023-02-24
NULL	NULL	NULL

DONATES_TO_BB

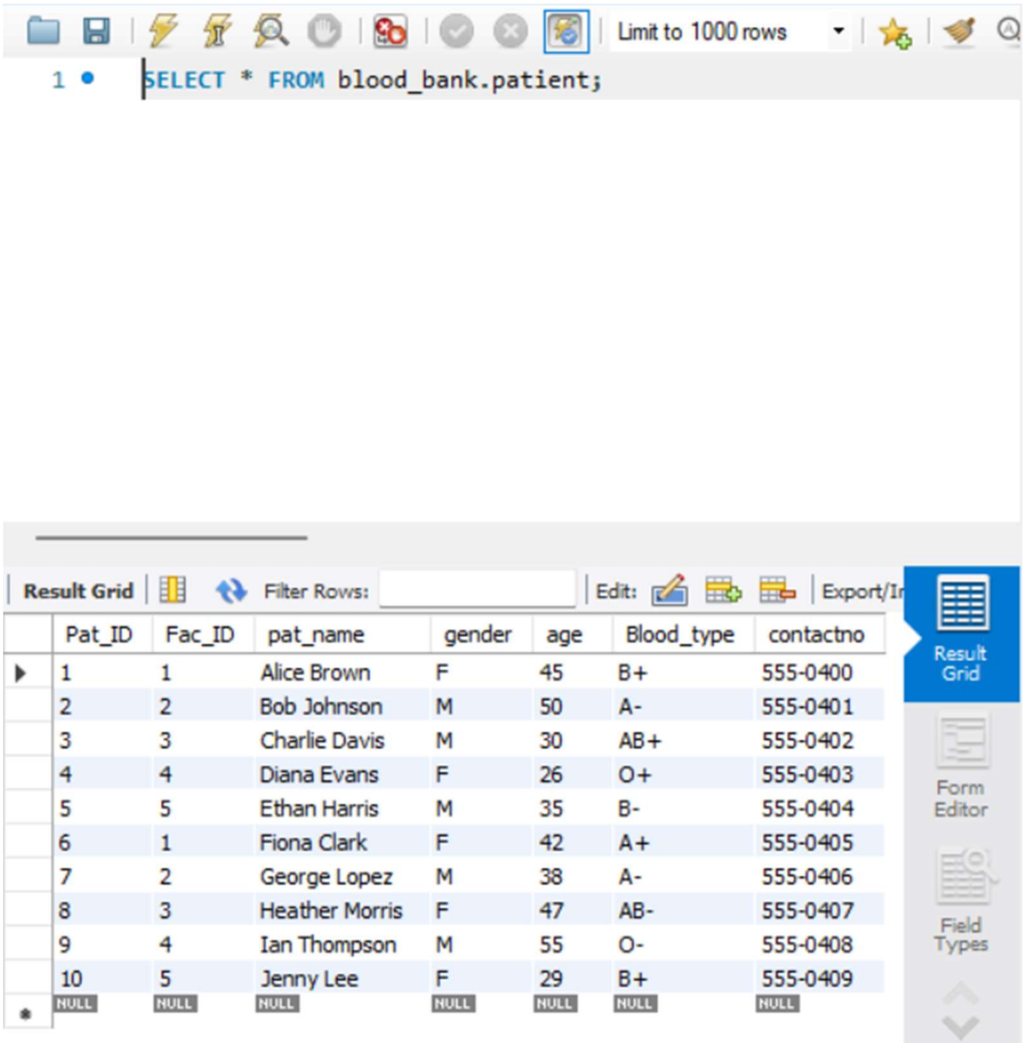


REQUESTS



BBank_ID	Fac_ID	date
1	1	2023-01-20
1	6	2023-01-25
2	2	2023-01-21
2	7	2023-01-26
3	3	2023-01-22
3	8	2023-01-27
4	4	2023-01-23
4	9	2023-01-28
5	5	2023-01-24
5	10	2023-01-29
NULL	NULL	NULL

PATIENT



```
1 SELECT * FROM blood_bank.patient;
```

Pat_ID	Fac_ID	pat_name	gender	age	Blood_type	contactno
1	1	Alice Brown	F	45	B+	555-0400
2	2	Bob Johnson	M	50	A-	555-0401
3	3	Charlie Davis	M	30	AB+	555-0402
4	4	Diana Evans	F	26	O+	555-0403
5	5	Ethan Harris	M	35	B-	555-0404
6	1	Fiona Clark	F	42	A+	555-0405
7	2	George Lopez	M	38	A-	555-0406
8	3	Heather Morris	F	47	AB-	555-0407
9	4	Ian Thompson	M	55	O-	555-0408
10	5	Jenny Lee	F	29	B+	555-0409
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Transactions:

1) Get a list of all blood banks along with the details of employees working there and the facilities they are linked to, including blood banks without any employees.

Query:

```
SELECT
  BB.BBank_name,
  E.Emp_name,
  F.name AS 'Facility_Name'
FROM
  Blood_Bank BB
LEFT JOIN
  Employee E ON BB.BBank_ID = E.BBank_ID
LEFT JOIN
  Facility F ON E.Fac_ID = F.Fac_ID;
```

BBank_name	Emp_name	Facility_Name
Red Hope Blood Center	Chris Green	General Hospital
Red Hope Blood Center	Bruce Wayne	Riverside Medical Center
Red Hope Blood Center	Lois Lane	Downtown Wellness Clinic
Red Hope Blood Center	Arthur Curry	City Clinic
Red Hope Blood Center	Selina Kyle	Hillside Health Facility
LifeStream Blood Bank	Diana Prince	City Clinic
LifeStream Blood Bank	Clark Kent	Hillside Health Facility
LifeStream Blood Bank	Barry Allen	General Hospital
LifeStream Blood Bank	Hal Jordan	Riverside Medical Center
LifeStream Blood Bank	Harley Quinn	Downtown Wellness Clinic
Vital Blood Services	NULL	NULL
Harmony Blood Found...	NULL	NULL
Pioneer Bloodworks	NULL	NULL
Guardian Blood Care	NULL	NULL
Trust Blood Bank	NULL	NULL
Unity Blood Network	NULL	NULL
Legacy Blood Institute	NULL	NULL
Caring Blood Donors	NULL	NULL

2) Display all blood type requests from hospitals, including those that have not yet been fulfilled.

Query:

```
SELECT
  H.Blood_type,
  H.req_date,
  F.name AS 'Facility_Name'
FROM
  Hospital H
RIGHT JOIN
  Facility F ON H.Fac_ID = F.Fac_ID;
```

Result Grid | Filter Rows: | Export:

Blood_type	req_date	Facility_Name
AB+	2023-01-10	General Hospital
O+	2023-01-11	City Clinic
A-	2023-01-12	Riverside Medical Center
B+	2023-01-13	Hillside Health Facility
A+	2023-01-14	Downtown Wellness Clinic
B-	2023-01-15	Community Urgent Care
AB-	2023-01-16	Metro Health Services
O-	2023-01-17	Eastside Medical Unit
A+	2023-01-18	West End Hospital
O+	2023-01-19	Suburban Healthcare Complex

3) List all donors and their donation dates at blood camps, including those who have not yet donated.

Query:

```
SELECT
  D.Don_name,
  BC.date
FROM
```

Donor D

LEFT JOIN

BC_donation BC ON D.Don_ID = BC.Don_ID;

Result Grid | Filter Rows:

Don_name	date
John Doe	2023-02-15
Jane Smith	2023-02-16
Michael Johnson	2023-02-17
Emily Davis	2023-02-18
David Wilson	2023-02-19
Sarah Miller	2023-02-20
James Brown	2023-02-21
Patricia Taylor	2023-02-22
Robert Anderson	2023-02-23
Linda Martinez	2023-02-24

4) Update contact information for a donor in the Donor table:

Query:

UPDATE Donor

SET contactno = '555-0333'

WHERE Don_ID = 1;

9 18:46:12 UPDATE Donor SET contactno = '555-0333' WHERE Don_ID = 1 1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0

5) Find the total quantity of each blood type available in the inventory.

Query:

SELECT type_of_Blood, SUM(quantity) AS Total_Quantity

FROM Inventory

GROUP BY type_of_Blood;

Result Grid Filter Rows:

type_of_Blood	Total_Quantity
A+	21
O-	18
B+	15
AB+	5
A-	7
O+	12
B-	6
AB-	9

6) Get the names of donors who are older than the average age of all donors.

Query:

SELECT Don_name, age

FROM Donor

WHERE age > (SELECT AVG(age) FROM Donor);

Result Grid Filter Ro

Don_name	age
Emily Davis	37
David Wilson	45
James Brown	52
Linda Martinez	39

7) List each blood bank along with the count of donations it has received.

Query:

SELECT BB.BBank_name, COUNT(BD.BBank_ID) AS Donation_Count

FROM Blood_Bank BB

LEFT JOIN BB_donation BD ON BB.BBank_ID = BD.BBank_ID

GROUP BY BB.BBank_name;

Result Grid | Filter Rows:

BBank_name	Donation_Count
Red Hope Blood Center	5
LifeStream Blood Bank	5
Vital Blood Services	0
Harmony Blood Foundation	0
Pioneer Bloodworks	0
Guardian Blood Care	0
Trust Blood Bank	0
Unity Blood Network	0
Legacy Blood Institute	0
Caring Blood Donors	0

8) Increase the age of all employees working at a specific facility by 1 year.

Query:

UPDATE Employee

SET age = age + 1

WHERE Fac_ID IN (SELECT Fac_ID FROM Facility WHERE name = 'City Clinic');

18 18:58:29 UPDATE Employee SET age = age + 1 WHERE Fac_ID IN (SELECT Fac_ID FROM Facility WHERE name ... 2 row(s) affected Rows matched: 2 Changed: 2 Warnings: 0

9) Find the names of all donors who have donated at facilities where a blood camp was held.

Query:

SELECT DISTINCT D.Don_name

FROM Donor D

JOIN BC_donation BCD ON D.Don_ID = BCD.Don_ID

JOIN Blood_camp BC ON BCD.Fac_ID = BC.Fac_ID;

Don_name
John Doe
Jane Smith
Michael Johnson
Emily Davis
David Wilson
Sarah Miller
James Brown
Patricia Taylor
Robert Anderson
Linda Martinez

10) Retrieve each donor's most recent donation date.

Query:

SELECT D.Don_ID, D.Don_name, MAX(BD.Date) AS Last_Donation_Date

FROM Donor D

LEFT JOIN BB_donation BD ON D.Don_ID = BD.Don_ID

GROUP BY D.Don_ID, D.Don_name;

Don_ID	Don_name	Last_Donation_Date
1	John Doe	2023-02-10
2	Jane Smith	2023-02-11
3	Michael Johnson	2023-02-12
4	Emily Davis	2023-02-13
5	David Wilson	2023-02-14
6	Sarah Miller	2023-02-15
7	James Brown	2023-02-16
8	Patricia Taylor	2023-02-17
9	Robert Anderson	2023-02-18
10	Linda Martinez	2023-02-19

Conclusion:

In conclusion, this project offers a comprehensive exploration of the intricacies involved in developing a database, with a specific focus on creating a blood bank database—a relevant and practical application in the modern world. Throughout the development process, the project team encountered various challenges, including the complexities of designing an effective ER diagram and establishing relationships between different entities. One particular challenge involved managing subsets within the facility entity, which demanded a nuanced understanding of specialization and generalization concepts. Consequently, the final blood bank database stands as a robust and efficient solution that meets all the project's initial criteria.

In summary, this project provides a valuable and informative overview of the multifaceted nature of database development. From the challenges encountered to the effective solutions devised, it offers a wealth of insights and lessons applicable to a wide array of real-world scenarios. Given the increasing importance of data management and analysis in today's world, the skills and techniques demonstrated in this project are likely to be in high demand among students and professionals alike.