

SCHOOL CLINIC MANAGEMENT DATABASE

**Winter 2024**

**INFO 605 – Introduction To Database Management**

**Prof. Il-Yeol Song**

**Members:**

|  |  |
| --- | --- |
| Full Name | Email |
| Adesewa Adesida | Ama547@drexel.edu |
| Donald Hattier | Dah358@drexel.edu |
| Shah Krushal | kms828@drexel.edu |
| Stephan Dupoux | Sgd45@drexel.edu |

1. **EXECUTIVE SUMMARY**

Our project focuses on developing a School Clinic Management Database System (SCMS) to streamline health services within school environments. The SCMS aims to consolidate student health records, appointment scheduling, medical inventory management, and parent communication into a unified platform. By integrating various clinic functions, the system enhances efficiency and reduces redundancies.

In the current scenario, school clinics often rely on disparate systems for managing health-related data, leading to inefficiencies and data redundancies. The SCMS addresses this challenge by providing a centralized solution for managing all relevant clinic functions. By combining data on student health, appointments, and medical inventory, the system offers administrative oversight and facilitates communication between the clinic and parents.

The project's scope includes managing student health records, appointment scheduling, medical inventory tracking, donation handling, and patient information storage. Out-of-scope elements include employee management, financial tracking, and non-health-related transactions.

While developing the SCMS, we focused on meeting specific data requirements such as physician availability, patient information, medical resources, appointments, events, blood drives, and donations. Business rules and data logic were established to ensure appointment scheduling, resource allocation, and patient management adhered to predefined criteria. After using ERDS to aid a more concise overview of our project conceptually, we got a better understanding of what we were aiming for and with review developed a more inclined Logical ERD.

The SCMS is expected to support efficient management of student health resources, evaluate medical resource allocation, output reports on clinic usage, and manage other necessary healthcare informatics. Assumptions were made regarding student membership, appointment constraints, vaccination status, and donation procedures to guide system development.

Moreover, using our system design, we developed a starting Database for this Project in Oracle SQL with technical Queries we believe highlight the efficiency and complexity of our system. Specifically, adhering to F.A.I.R data principles when possible and minding access control in line with some HIPAA regulations.

Lastly, as this was a short-term project, you will find we also included our takeaways, issues, and even slight feedback from our own reviews. Moving forward, our goal is to implement the SCMS effectively, ensuring seamless integration into school clinic operations while addressing any challenges that may arise. By leveraging the SCMS, schools can improve the delivery of healthcare services, enhance administrative oversight, and promote transparency in clinic management.

1. **THE PROBLEM STATEMENT**

**2.1 Overall Goals of the System**

This project aims to develop a comprehensive School Clinic Management Database System (SCMS) to efficiently manage and organize essential data related to health services within the school environment. The SCMS will be designed to manage student health records, appointments, medical inventory, and communication with parents or guardians. The primary goal is to create a unified platform that streamlines various clinic functions, reducing redundancies and enhancing overall efficiency.

The SCMS will serve as a centralized database for the student perspective, consolidating student health information, appointment scheduling, and medical inventory records into a single application. This integrated approach ensures that clinic staff can access all necessary information seamlessly, eliminating the need for multiple disconnected systems. Additionally, the system will facilitate communication between the clinic and parents, providing updates on student health and appointment details.

**2.2 Context and Importance of the System**

In a school clinic's context, the current scenario often involves disparate systems for managing student health records, appointment bookings, and medical inventory to avoid violating design rules. This divided approach can lead to data redundancies and inefficiencies in coordinating clinic activities. The SCMS aims to address these challenges by offering a unified solution for managing all clinic-related functions.

The importance of the SCMS lies in its potential to enhance administrative oversight and streamline communication. By combining data on student health, appointments, and medical resources, the school administration gains a complete view of the clinic's operations. This visibility allows for better budget allocation, reporting to relevant authorities, and evaluating the need for additional resources.

Moreover, the SCMS benefits students and parents by providing a centralized portal for accessing health-related information. Students can track their health records, parents can stay informed about appointments and health updates, all within a single user account. This not only simplifies the user experience but also contributes to a more efficient and transparent school clinic management system.

**2.3 Scope of the Project**

**2.3.1 IN-Scope**

The School Clinic Management Database System (SCMS) will encompass various essential components to effectively manage health-related processes within the school. The in-scope elements include:

* **Student Health Records:** The system will store health information for each student.
* **Appointment Records:** Recording and managing appointments for students seeking medical assistance.
* **Medical Inventory:** Keeping track of the clinic's medical supplies like blood supply, medical devices, and medicines.
* **Donation Handling:** Organizing blood donations drives and keeping track of the blood supply, its type, the blood expiration date among other important information.
* **Patient Information:** Storing patient information such as legal guardians, emergency contact
  + 1. **Out-Scope**

Certain aspects fall outside the scope of the SCMS, including:

* **Employee Management:** The system will not manage employee scheduling, or volunteer management.
* **Financials:** It will not track the entire school budget, or monetary donations, employee salaries, operations expenses, or building maintenance costs.
* **Student Transaction Processing System:** Transactions related to student information, grades, or other non-health-related activities.
* **Other Forms of Donations:** Donations such as donating organs or volunteer time by students or researchers are not covered by this.
  1. **Related systems and any open-source tools**

While the SCMS focuses on managing health-related aspects, other systems within the school include:

* **HR Database:** This system manages employee information, including vacation and sick leave, hours worked, wages and benefits, and scheduling.
* **Budget Software:** Tracking expenses and funds across the school system, processing payments for lost books, and managing clinic funding.

These related systems provide a clear view of the school clinic's operations, with each serving a specific purpose and contributing to the overall efficiency and functionality of the institution. The SCMS complements the existing infrastructure by focusing specifically on health-related aspects within the broader context of student information and school transactions.

1. **REQUIREMENTS SPECIFICATION**

**3.1 Data Requirements**

* **Singleton:**
  + Physician Available, Version, Last Update Date, Clinic Address, Operational Hours, Clinic Phone Number, Clinic Name.
* **Patient Information:**
  + Name, address, Phone Number, Email Address, Vaccination records, Medical Insurance, Emergency Contact, Legal Guardians if applicable, Records.
  + Vaccination record – Vaccination Name, Date Given if it is Mandatory, Administered by.
  + Insurance – Insurance Name, Policy Id, Group Id, Member Id, Contact given in insurance.
  + Emergency Contact- Phone number, Email Address, Relationship.
  + Records – Record Id, Record Date, Resources Used, Treatment, Physician Id, Medical Resource Id.
* **Medical Resources (Inventory):**
  + Types include medical supplies, equipment, and relevant health materials.
  + Details such as suppliers, date of procurement, location of storage, and availability.
  + Attributes used are as follows: Medical Resource Id, Storage Location, Item Name, Expiration Date, Supplier, Quantity, Quantity On Hold and the Quantity Available.
  + Medical Devices – Purchased Date, Maintenance Date, Purchased From.
  + Medicines- This includes the general description of the medicines.
  + Blood Supply – Blood Type, Hemoglobin Level, RH Value, Blood Amount, Date Received, Event Id, Expiration Date
* **Physician:**
  + Physician table records the information related to the Physician and connected to the appointments that the physician has with the patients.
  + Medical License, Name, (first name and last name), Specialty, Phone Number and Email Address.
* **Appointments:**
  + Appointment Id, Reserved Resources, Appointment Date
* **Events:**
  + Event name, date, time, clinic location, and topics or themes related to health e.g., blood donation drive, flu shots.
* **Blood Drive:**
  + Status HIV, Sickle cell, History of Disease, Additional Details.
* **Donations:**
  + Donation details (foundation or individual donor), donation date, donation type, and donation's contact information.
  + Allocation of donation to health events and medical resources.

**3.2 Business Rules and Data Logic**

* **Appointment Rules:**
  + Each student can have multiple appointments but not more than one at a given time.
* **Resource and Event Associations:**
  + One donation of blood can supply multiple resources of the same blood type.
* **Blood Drive Associations:**
  + The blood drive can supply multiple blood supply resources
* **Patient Rules:**
  + Patients must have a(n) emergency contact(s)
  + If a patient is under 18, they must have Parental or Guardian contact
  + A patient can set an appointment if they meet conditions

**3.3 Sample Output**

The school hopes to use this database for various activities, including:

* Efficient Management of Student Health resources.
* Evaluating medical resources to understand their allocation towards specific appointments.
* Assessing clinic usage by outputting reports on the most attended health events and their locations.
* Manage other necessary healthcare informatics.

**3.4 Other Assumptions**

* **Student:**
  + Student membership and access to health services are only for active students.
  + All active students have medical Insurance.
* **Appointment Constraints:**
  + Each student can have zero to many appointments as long as they do not overlap.
  + Overlapping appointment schedules for a specific student are not allowed.
* **Vaccination Status:**
  + Each student can have a vaccination status of failing if their vaccine has expired or if they did not take it.
* **Donation Constraints:**
  + Donators can only donate blood during blood drives.

**3.5 Data Creation Technique**

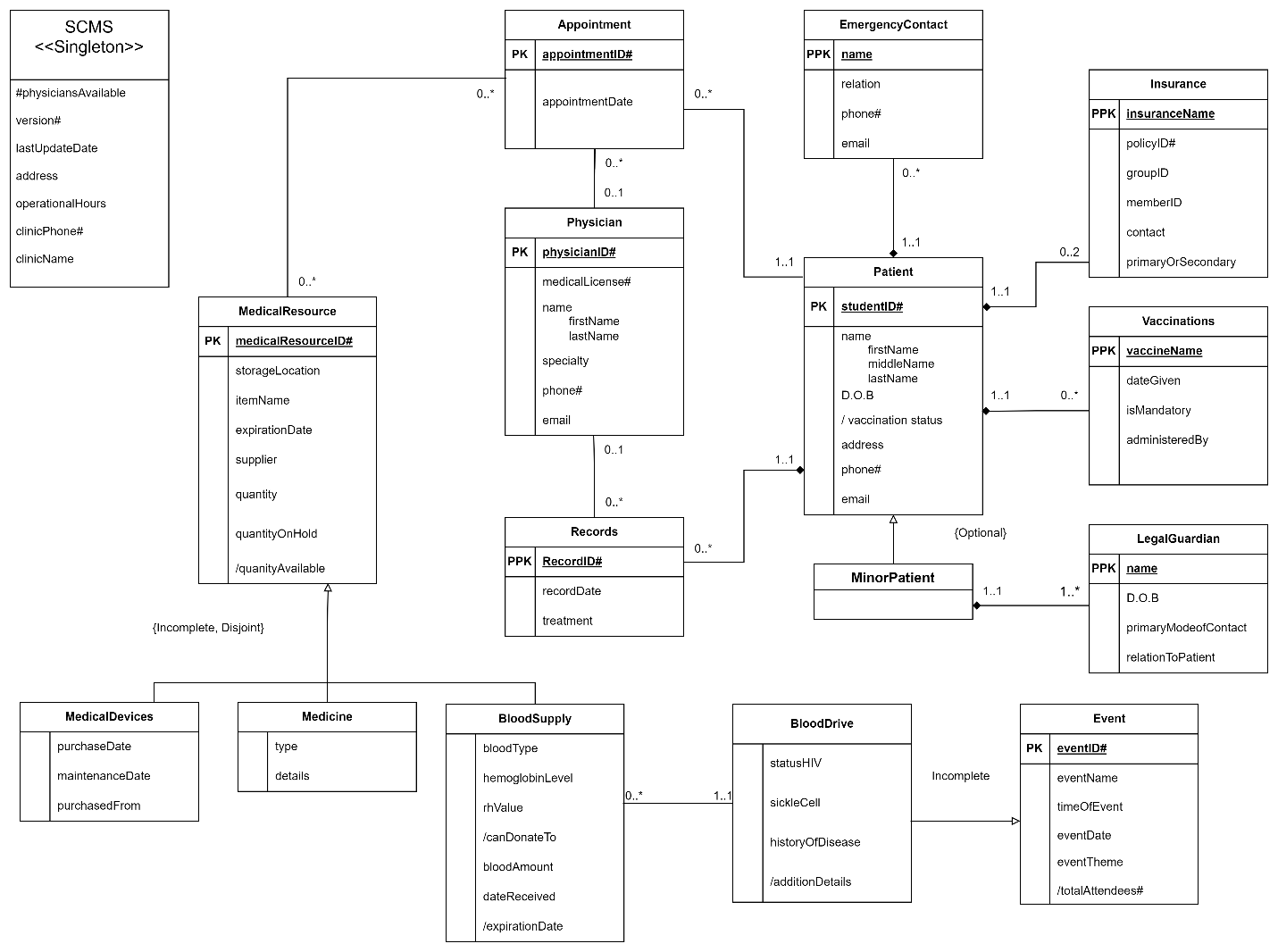
We created hypothetical random data following our data requirements. With our data creation we did our best to have a representation for most instances. For example, for blood type there are patients with various blood types encompassing all the possible blood types.

**3.6 Hardware And Software**

* + - 1. **Draw.io**
         * Link: <https://app.diagrams.net/>
         * Draw.io is a versatile online tool that enables users to create diagrams, flowcharts, and visual representations with ease. It offers a user-friendly interface and a wide range of shapes and templates for designing various types of diagrams.
      2. **Oracle SQL Server**
         * Provided by Drexel
         * Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
         * Version 19.3.0.0.0

1. **CONCEPTUAL DESIGN**

**4.1 Conceptual Model**



Digital Collections Management/A. Adesida, S. Krushal, D. Hattier, and S. Dupoux

**4.2 Explanations on the ERD**

* + We used 4 weak entities as they all belong to their owner entities with existence dependencies.
  + Event has an {incomplete} relationship with blood drive because that is not the only event clinics can host.
  + Medical Resources has an {incomplete, disjoint} relationship with Medicine, medical devices, and blood Supply because those are not the only resources a clinic may have in its inventory.
  + A patient is either an adult or a minor.
  + An appointment can require one to many physicians
  + An appointment can reserve zero to many medical resources
  + D.O.B means date of birth

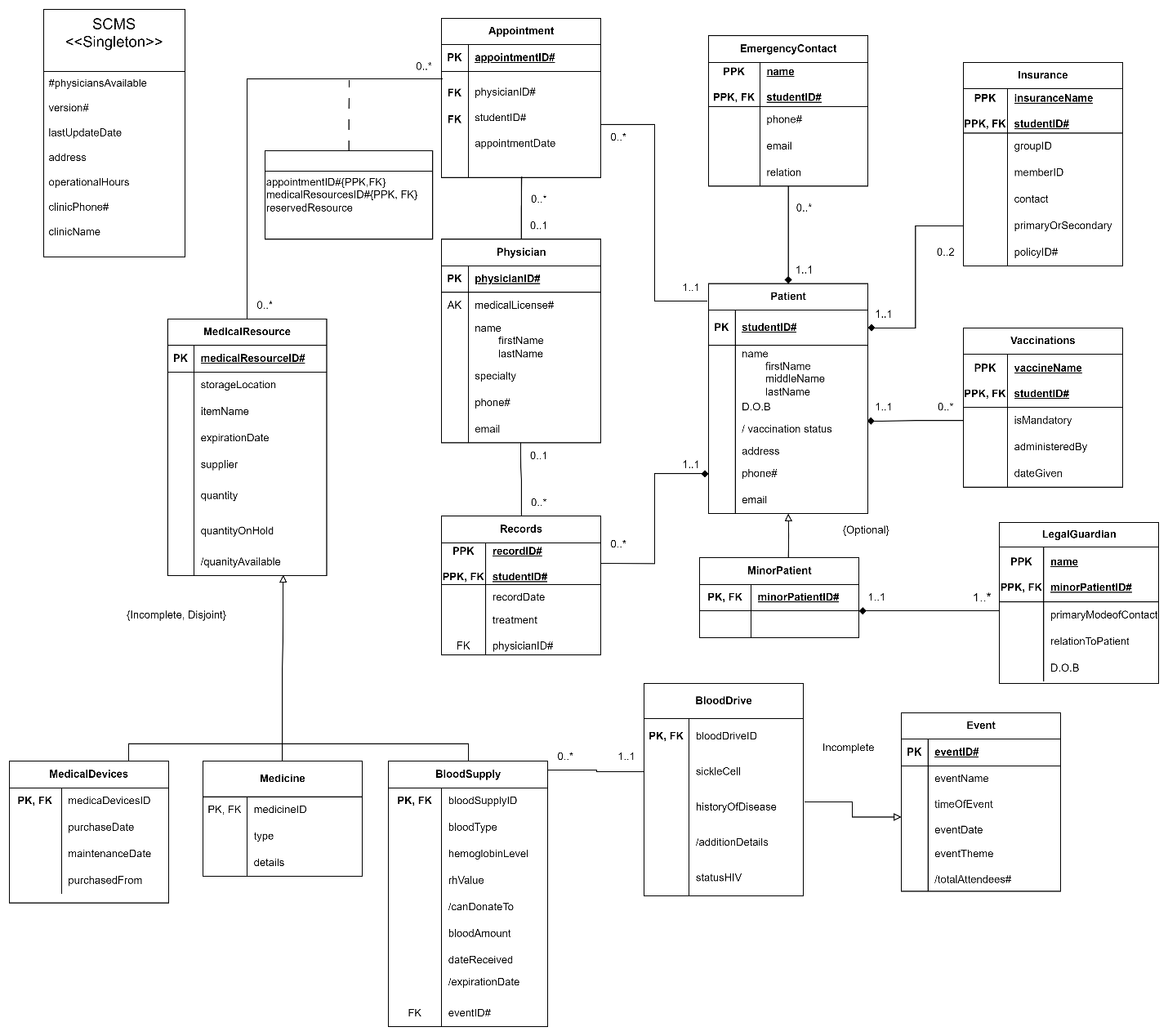
1. **RELATIONAL SCHEMA**

**5.1 Relational Schema**

* Patient (**studentID#**, firstName, middleName, lastName, D.O.B, address, phone, email)
* EmergencyContact (**studentID#**, **name**, relation, phone, email)
  + Foreign key(*studentID#*) references Patient(*studentID#)*
* Records (**studentID#** ,**recordID#**, recordDate, treatment)
  + Foreign key(*studentID#*) references Patient(*studentID#*)
* Insurance (**insuranceName**, **studentID#**, policyID, groupID, memberID, contact, primaryOrSecondary)
  + Foreign key(*studentID#*) references Patient(*studentID#)*
* Vaccinations (**vaccineName**, **studentID#** dateGiven, isMandatory, administeredBy)
  + Foreign key(*studentID#*) references Patient(*studentID#)*
* MinorPatient(**minorPatientID#**)
  + Foreign key(*minorPatientID#*) references Patient(*studentID#)*
* LegalGuardian(**minorPatientID#**, **name**, dateOfBirth, primaryModeOfContact, relationToPatient)
  + Foreign key*(minorPatientID*) references Patient(*studentID#*)
* Physician **(physicianID#,** physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email)
* Appointment (**appointmentID#**, **studentID#**, **physicianID#**, appointmentDate)
  + Foreign key *(studentID#*) references Patient(*studentID#*)
  + Foreign key *(physicianID#*) references Physician(*physicianID#*)
* AppointmentReservedResource(**appointmentID,** **medicalResourseID,** reservedResource)
  + Foreign Key (*appointmentID*) references Appointment*(appointmentID*)
  + Foreign Key (medicalResourseID) references MedicalResource(medicalResourceID)
* MedicalResource (**medicalResourceID#**, storageLocation, itemName, expirationDate, supplier, quantity, quantityOnHold)
* BloodSupply (**bloodSupplyID#**, **bloodDriveID#**, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived)
  + Foreign key (*bloodSupplyID#*) references MedicalResource(*medicalResourceID#*)
  + Foreign key (*bloodDriveID#*) references Event(*eventID#*)
* Medicine (**medicineID#**, type, details)
  + Foreign Key (*medicineID#*) references MedicalResource(*medicalResourceID#*)
* MedicalDevices (**medicalDevicesID#**, purchaseDate, purchaseFrom, maintenanceDate)
  + Foreign Key (**medicalDevicesID#**) references MedicalResource(*medicalResourceID#*)
* Event (**eventID#**, eventName, timeOfEvent, eventDate, eventTheme)
* BloodDrive(**bloodDriveID#,** statusHIV, sickleCell, historyOfDisease)
  + Foreign key (**bloodDriveID#**) references Event(*eventID#*)

**5.2) LOGICAL ERD**

Digital Collections Management/A. Adesida, S. Krushal, D. Hattier, and S. Dupoux



1. **DATA DICTIONARY (Follows relational Schema Order)**
   1. **Table Name: Patient**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **studentID#** | Unique Identifier for Student Insertion | Integer | Digit ID | Yes | Yes | No |  |
| firstName | First name of student | Varchar(40) | Alphanumeric Text | Yes | No | No |  |
| middleName | Middle name of student | Varchar(40) | Alphanumeric Text | Yes | No | No |  |
| lastName | Last name of student | Varchar(40) | Alphanumeric Text | Yes | No | No |  |
| D.O.B | Student’s date of birth | Date | YYYY-MM-DD | Yes | No | No |  |
| address | Student’s address | Varchar(200) | Alphanumeric Text | Yes | No | No |  |
| phone | Student’s phone number | Varchar(50) | Alphanumeric Text | Yes | No | No |  |
| email | Student’s email address | Varchar(50) | Alphanumeric Text | Yes | No | No |  |

* 1. **Table Name: EmergencyContact**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **studentID#** | Unique Identifier for Student Insertion | Integer | Digit ID | Yes | Yes | Yes |  |
| **name** | Emergency contact’s name | Varchar(40) | Alphanumeric Text | Yes | Yes | No |  |
| relation | Emergency contact’s relationship to the student | Varchar(50) | Alphanumeric Text | Yes | No | No |  |
| phone | Emergency contact’s phone number | Varchar(20) | Alphanumeric Text | Yes | No | No |  |
| email | Emergency contact’s phone number | Varchar(100) | Alphanumeric Text | Yes | No | No |  |

* 1. **Table Name: Records**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **studentID#** | Unique Identifier for Student Insertion | Integer | Digit ID | Yes | No | Yes |  |
| **recordID#** | Unique Identifier for Record Insertion | Integer | Digit ID | Yes | Yes | No |  |
| recordDate | Unique Record for Date of Insertion | Date | YYYY-MM-DD | Yes | No | No | Metadata |
| treatment | Instructions for Patients Post Appointment | Varchar(255) | Alphanumeric Text | Yes | No | No |  |

* 1. **Table Name: Insurance**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **insuranceName** | Name of the insurance plan | Varchar(100) | Alphanumeric Text | Yes | Yes | No |  |
| **studentID#** | Unique Identifier for Student Insertion | Integer | Digit ID | Yes | Yes | Yes |  |
| policyID | Unique Identifier for policy | Varchar(50) | Alphanumeric Text | Yes | No | No |  |
| groupID | Unique Identifier for policy group | Varchar(50) | Alphanumeric Text | Yes | No | No |  |
| memberID | Unique Identifier for policy member | Varchar(50) | Alphanumeric Text | Yes | No | No |  |
| contact | Insurance provider’s contact | Varchar(100) | Alphanumeric Text | Yes | No | No |  |
| primaryOrSecondary | Is this insurance the students primary or secondary insurance | Varchar(20) | Alphanumeric Text | Yes | No |  |  |

* 1. **Table Name: Vaccinations**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **vaccineName** | Name of Vaccine | Varchar(100) | Text | Yes | Yes | No |  |
| **studentID#** | Unique Identifier for Student Insertion | Integer | Digit  ID | Yes | Yes | Yes |  |
| dateGiven | Date the vaccination was given | Date | YYYY-MM-DD | Yes | No | No |  |
| isMandatory | If Vaccine is required before school | Char(1) | Single Alpha Variable | Yes | No | No | Either Y or N |
| administeredBy | Name of the clinic that administered the vaccination | Varchar(100) | Alphanumeric Text | Yes | No | No |  |

* 1. **Table Name: MinorPatient**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **minorPatientID#** | Unique Identifier for Student Insertion | Integer | Eight Digit ID | Yes | Yes | Yes |  |

* 1. **Table Name: LegalGuardian**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **minorPatientID#** | Unique Identifier for Student Insertion | Int | Digit ID | Yes | Yes | Yes |  |
| **name** | Legal Guardian’s name | Varchar(100) | Alphanumeric Text | Yes | Yes | No |  |
| dateOfBirth | Legal guardian’s date of birth | Date | YYYY-MM-DD | Yes | No | No |  |
| primaryModeOfContact | Legal guardian’s primary contact information | Varchar(20) | Alphanumeric Text | Yes | No | No |  |
| relationToPatient | Legal guardian's relationship to the student | Varchar(50) | Alphanumeric Text | Yes | No | No |  |

* 1. **Table Name: Physician**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **physicianID#** | Unique Identifier for Physician Insertion | Integer | Digit ID | Yes | Yes | No |  |
| physician\_firstName | Physician’s first name | Varchar(50) | Alphanumeric Text | Yes | No | No |  |
| physician\_lastName | Physician’s last name | Varchar(50) | Alphanumeric Text | Yes | No | No |  |
| medicalLicense | Physician’s license identifier | Varchar(100) | Alphanumeric Text | Yes | No | No |  |
| specialty | Physician’s license | Varchar(100) | Alphanumeric Text | Yes | No | No |  |
| phone | Physician’s phone number | Varchar(20) | Alphanumeric Text | Yes | No | No |  |
| email | Physician’s email address | Varchar(100) | Alphanumeric Text | Yes | No | No |  |

* 1. **Table Name: Appointment**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **appointmentID#** | Unique Identifier for Appointment Insertion | Integer | Digit ID | Yes |  | No |  |
| **studentID#** | Unique Identifier for Student Insertion | Integer | Digit ID | Yes | No |  |  |
| **physicianID#** | Unique Identifier for Physician Insertion | Integer | Digit ID | Yes | No |  |  |
| appointmentDate | Date of the appointment | Date | YYYY-MM-DD | Yes | No | No |  |

* 1. **Table Name: AppointmentReservedResource**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **appointmentID** | Unique Identifier for Appointment Insertion | Integer | Digit ID | Yes | Yes | Yes |  |
| **medicalResourseID** | Unique Identifier for medical resource Insertion | Integer | Digit ID | Yes | Yes | Yes |  |
| reservedResource | Whether or not the resource is reserved | CHAR(1) | Single Alpha Variable | Yes | No | No | Either ‘Y’ or ‘N’ |

* 1. **Table Name: MedicalResource**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **medicalResourceID#** | Unique Identifier for Medical Resource Insertion | Integer | Eight Digit ID | Yes | Yes | No |  |
| storageLocation |  |  |  | Yes |  |  |  |
| itemName | The name of the resource | Varchar(100) | Alphanumeric Text | Yes | No | No |  |
| expirationDate | The resources expiration date | Date | YYYY-MM-DD | Yes | No | No |  |
| supplier | Name of supplier | Varchar(100) | Alphanumeric Text | Yes | No | No |  |
| quantity | Quantity of the resource in stock | Integer | Integer | Yes | No | No |  |
| quantityOnHold | Quantity of the resource on hold | Integer | Integer | Yes | No | No |  |

* 1. **Table Name: BloodSupply**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **bloodSupplyID#** | Unique Identifier | Int | Four Digit ID | Yes | Yes | Yes |  |
| **bloodDriveID#** | Unique Identifier | Int | Four Digit ID | Yes | No | Yes |  |
| bloodType | Displaying Types of Blood | Varchar(10) | Short Text | Yes | No | No |  |
| hemoglobinLevel | Amount of Hemoglobin in Sample | Decimal(5,2) | Float Number | Yes | No | No |  |
| rhValue | What the Rh value of the blood is | Varchar(10) | Special Characters | Yes | No | No |  |
| bloodAmount | How much blood we have | Int | Integer | Yes | No | No |  |
| dateReceived | Date that was donated to the blood bank | Date | YYYY-MM-DD | Yes | No | No |  |

* 1. **Table Name: Medicine**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **medicineID#** | Unique Identifier | Integer | Integer | Yes | Yes | Yes |  |
| type | Describing types of medicine being used | Varchar(100) | Text | Yes | No | No |  |
| details | Identifier to Name use for the medicine | Varchar(255) | Text | Yes | No | No |  |

* 1. **Table Name: Medical Devices**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **medicalDevicesID#** | Unique Identifier for Each Medical Device | Int | Integer | Yes | Yes | No |  |
| purchaseDate | Date that the Device was bought | Date | YYYY-MM-DD | Yes | No | No |  |
| purchaseFrom | Vendor that the product was bought from | Varchar(100) | Text | Yes | No | No |  |
| maintenanceDate | Date maintenance was last performed | Date | YYYY-MM-DD | Yes | No | No |  |

* 1. **Table Name: Event**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **eventID#** | Unique Identifier | INT | Integer | Yes | Yes | No |  |
| eventName | Name of the Event | VARCHAR(100) | Text | Yes | No | No |  |
| timeOfEvent | The Date and Time this event Started at | TimeStamp | YYYY-MM-DD HH24:MI:SS | Yes | No | No |  |
| eventDate | The date of the event | Date | YYYY-MM-DD | Yes | No | No |  |
| eventTheme | If this | Varchar(100) | Text | Yes | No | No |  |

* 1. **Table Name: BloodDrive**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Meaning** | **Data Type** | **Domain** | **NOT NULL** | **PK** | **FK** | **Other** |
| **bloodDriveID#** | Unique identifier for each blood drive event | Integer | 701293 | Yes | Yes | No |  |
| statusHIV | Identifier to notify if a person has HIV | Char(1) | Y or N | Yes | No | No | Will Either Be Y or N |
| sickleCell | Identifier to notify if a person has Sickle Cell | Char(1) | Y or N | Yes | No | No | Will Either Be Y or N |
| historyOfDisease | Identifier to highlight history of major sicknesses | Varchar(255) | Text | Yes | No | No |  |

1. **SCHEMA AND DATA**

**7.1) Create Table commands**

--creates tables for the SCMS

--Patient

--drop table in case your db already have Patient table.

drop TABLE Patient CASCADE CONSTRAINTS;

CREATE TABLE Patient (

studentID INT PRIMARY KEY,

firstName VARCHAR(40) NOT NULL,

middleName VARCHAR(40) NOT NULL,

lastName VARCHAR(40) NOT NULL,

dob DATE NOT NULL,

address VARCHAR(200) NOT NULL,

phone VARCHAR(50) NOT NULL,

email VARCHAR(50) NOT NULL

);

--MedicalResource

--drop table in case your db already have MedicalResource table.

drop TABLE MedicalResource CASCADE CONSTRAINTS;

CREATE TABLE MedicalResource (

medicalResourceID INT PRIMARY KEY,

itemName VARCHAR(100) NOT NULL,

expirationDate DATE NOT NULL,

supplier VARCHAR(100) NOT NULL,

storageLocation VARCHAR(100) NOT NULL,

quantity INT NOT NULL,

quantityOnHold INT NOT NULL

);

--Medicine

--drop table in case your db already have Medicine table.

drop TABLE Medicine CASCADE CONSTRAINTS;

CREATE TABLE Medicine (

medicineID INT PRIMARY KEY,

type VARCHAR(100) NOT NULL,

details VARCHAR(255) NOT NULL,

FOREIGN KEY (medicineID) REFERENCES MedicalResource(medicalResourceID)

);

--MedicalDevices

--drop table in case your db already have MedicalDevices table.

drop TABLE MedicalDevices CASCADE CONSTRAINTS;

CREATE TABLE MedicalDevices (

medicalDevicesID INT PRIMARY KEY,

purchaseDate DATE NOT NULL,

purchaseFrom VARCHAR(100) NOT NULL,

maintenanceDate DATE NOT NULL,

FOREIGN KEY (medicalDevicesID) REFERENCES MedicalResource(medicalResourceID)

);

--Event

--drop table in case your db already have Event table.

drop TABLE Event CASCADE CONSTRAINTS;

CREATE TABLE Event (

eventID INT PRIMARY KEY,

eventName VARCHAR(100) NOT NULL,

timeOfEvent TIMESTAMP NOT NULL,

eventDate DATE NOT NULL,

eventTheme VARCHAR(100) NOT NULL

);

--BloodDrive

--drop table in case your db already have BloodDrive table.

drop TABLE BloodDrive CASCADE CONSTRAINTS;

CREATE TABLE BloodDrive (

bloodDriveID INT PRIMARY KEY,

statusHIV CHAR(1) NOT NULL,

sickleCell CHAR(1) NOT NULL,

historyOfDisease VARCHAR(255) NOT NULL,

FOREIGN KEY (bloodDriveID) REFERENCES Event(eventID)

);

--EmergencyContact

--drop table in case your db already have EmergencyContact table.

drop TABLE EmergencyContact CASCADE CONSTRAINTS;

CREATE TABLE EmergencyContact (

studentID INT NOT NULL,

name VARCHAR(100) NOT NULL,

relation VARCHAR(50) NOT NULL,

phone VARCHAR(20) NOT NULL,

email VARCHAR(100) NOT NULL,

FOREIGN KEY (studentID) REFERENCES Patient(studentID)

);

--Records

--drop table in case your db already have Records table.

drop TABLE Records CASCADE CONSTRAINTS;

CREATE TABLE Records (

recordID INT PRIMARY KEY,

studentID INT NOT NULL,

recordDate DATE NOT NULL,

treatment VARCHAR(255) NOT NULL,

FOREIGN KEY (studentID) REFERENCES Patient(studentID)

);

--Insurance

--drop table in case your db already have Insurance table.

drop TABLE Insurance CASCADE CONSTRAINTS;

CREATE TABLE Insurance (

insuranceName VARCHAR(100) NOT NULL,

studentID INT NOT NULL,

policyID VARCHAR(50) NOT NULL,

groupID VARCHAR(50) NOT NULL,

memberID VARCHAR(50) NOT NULL,

contact VARCHAR(100) NOT NULL,

primaryOrSecondary VARCHAR(20) NOT NULL,

FOREIGN KEY (studentID) REFERENCES Patient(studentID)

);

--Vaccinations

--drop table in case your db already have Vaccinations table.

drop TABLE Vaccinations CASCADE CONSTRAINTS;

CREATE TABLE Vaccinations (

vaccineName VARCHAR(100) NOT NULL,

studentID INT NOT NULL,

dateGiven DATE NOT NULL,

isMandatory Char(1) NOT NULL,

administeredBy VARCHAR(100) NOT NULL,

FOREIGN KEY (studentID) REFERENCES Patient(studentID)

);

--MinorPatient

--drop table in case your db already have MinorPatient table.

drop TABLE MinorPatient CASCADE CONSTRAINTS;

CREATE TABLE MinorPatient (

minorPatientID INT NOT NULL,

FOREIGN KEY (minorPatientID) REFERENCES Patient(studentID)

);

--LegalGuardian

--drop table in case your db already have LegalGuardian table.

drop TABLE LegalGuardian CASCADE CONSTRAINTS;

CREATE TABLE LegalGuardian (

minorPatientID INT NOT NULL,

name VARCHAR(100) NOT NULL,

dateOfBirth DATE NOT NULL,

primaryModeOfContact VARCHAR(20) NOT NULL,

relationToPatient VARCHAR(50) NOT NULL,

FOREIGN KEY (minorPatientID) REFERENCES Patient(studentID)

);

--Physician

--drop table in case your db already have Physician table.

drop TABLE Physician CASCADE CONSTRAINTS;

CREATE TABLE Physician (

physicianID INT PRIMARY KEY,

physician\_firstName VARCHAR(50) NOT NULL,

physician\_lastName VARCHAR(50) NOT NULL,

medicalLicense VARCHAR(100) NOT NULL,

specialty VARCHAR(100) NOT NULL,

phone VARCHAR(20) NOT NULL,

email VARCHAR(100) NOT NULL

);

--Appointment

--drop table in case your db already have Appointment table.

drop TABLE Appointment CASCADE CONSTRAINTS;

CREATE TABLE Appointment (

appointmentID INT PRIMARY KEY,

studentID INT NOT NULL,

physicianID INT NOT NULL,

appointmentDate DATE NOT NULL,

FOREIGN KEY (studentID) REFERENCES Patient(studentID),

FOREIGN KEY (physicianID) REFERENCES Physician(physicianID)

);

--AppointmentReservedResource

--drop table in case your db already have AppointmentReservedResource table.

drop TABLE AppointmentReservedResource CASCADE CONSTRAINTS;

CREATE TABLE AppointmentReservedResource (

appointmentID INT NOT NULL,

medicalResourceID INT NULL ,

reservedResource CHAR(1) NOT NULL,

FOREIGN KEY (appointmentID) REFERENCES Appointment(appointmentID),

FOREIGN KEY (medicalResourceID) REFERENCES MedicalResource(medicalResourceID)

);

--BloodSupply

--drop table in case your db already have BloodSupply table.

drop TABLE BloodSupply CASCADE CONSTRAINTS;

CREATE TABLE BloodSupply (

bloodSupplyID INT PRIMARY KEY,

bloodDriveID INT NOT NULL,

bloodType VARCHAR(10) NOT NULL,

hemoglobinLevel DECIMAL(5,2) NOT NULL,

rhValue VARCHAR(10) NOT NULL,

bloodAmount INT NOT NULL,

dateReceived DATE NOT NULL,

FOREIGN KEY (bloodSupplyID) REFERENCES MedicalResource(medicalResourceID),

FOREIGN KEY (bloodDriveID) REFERENCES Event(eventID)

);

**7.2)** **INSERT INTO Commands**

-- Insert commands for Patient table

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES (10001, 'John', 'Robert', 'Doe', DATE '2005-03-15', '123 Main St, Anytown, USA', '12125550101', 'john.doe@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10002, 'Emily', 'Anne', 'Smith', DATE '2006-07-20', '456 Elm St, Othertown, USA', '12125550102', 'emily.smith@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10003, 'Michael', 'James', 'Johnson', DATE '2007-01-10', '789 Oak St, Anycity, USA', '12125550103', 'michael.johnson@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10004, 'Sarah', 'Grace', 'Williams', DATE '2008-05-05', '321 Maple St, Otherville, USA', '12125550104', 'sarah.williams@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10005, 'Matthew', 'David', 'Brown', DATE '2009-09-18', '654 Pine St, Somecity, USA', '12125550105', 'matthew.brown@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10006, 'Olivia', 'Nicole', 'Jones', DATE '2010-11-30', '987 Cedar St, Anothertown, USA', '12125550106', 'olivia.jones@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10007, 'Daniel', 'Andrew', 'Garcia', DATE '2011-04-25', '741 Birch St, Somewhere, USA', '12125550107', 'daniel.garcia@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10008, 'Emma', 'Elizabeth', 'Martinez', DATE '2012-08-12', '852 Walnut St, Anywhere, USA', '12125550108', 'emma.martinez@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10009, 'Christopher', 'Ryan', 'Hernandez', DATE '2013-10-05', '369 Spruce St, Anothercity, USA', '12125550109', 'christopher.hernandez@example.com');

INSERT INTO Patient (studentID, firstName, middleName, lastName, dob, address, phone, email) VALUES(10010, 'Ava', 'Isabella', 'Young', DATE '2014-12-20', '159 Fir St, Someplace, USA', '12125550110', 'ava.young@example.com');

-- Insert commands for Physician table

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5001, 'John', 'Smith', 'PA12345', 'Pediatrics', '1234567890', 'drjohn@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5002, 'Emily', 'Johnson', 'GP67890', 'General Practice', '2345678901', 'dremily@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5003, 'Michael', 'Williams', 'FA12378', 'Family Medicine', '3456789012', 'drwilliams@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5004, 'Sarah', 'Brown', 'IM90783', 'Internal Medicine', '4567890123', 'drsbrown@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5005, 'Jennifer', 'Garcia', 'PS87543', 'Psychiatry', '5678901234', 'drjgarcia@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5006, 'David', 'Martinez', 'CN23456', 'Cardiology', '6789012345', 'drdmartinez@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5007, 'Jessica', 'Hernandez', 'OB45678', 'Obstetrics and Gynecology', '7890123456', 'drjhernandez@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5008, 'Daniel', 'Young', 'NS56378', 'Neurology', '8901234567', 'drdyoung@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5009, 'Amanda', 'Clark', 'SU23478', 'Surgery', '9012345678', 'dramanda@example.com');

INSERT INTO Physician (physicianID, physician\_firstName, physician\_lastName, medicalLicense, specialty, phone, email) VALUES (5010, 'Kevin', 'Lee', 'OR76543', 'Orthopedics', '1234567890', 'drklee@example.com');

-- Insert commands for MedicalResource table

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3001, 'Bandages', DATE '2025-12-31', 'Medical Supply Co.', 'Clinic Storage Room 1', 500, 50);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3002, 'Thermometers', DATE '2025-12-31', 'MedTech Supplies', 'Clinic Storage Room 2', 100, 10);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3003, 'Syringes', DATE '2025-12-31', 'Healthcare Solutions Inc.', 'Clinic Storage Room 3', 1000, 100);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3004, 'Cotton Swabs', DATE '2025-12-31', 'MediCare Distributors', 'Clinic Storage Room 1', 2000, 200);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3005, 'Gloves', DATE '2025-12-31', 'Safety Health Corp.', 'Clinic Storage Room 2', 800, 80);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3006, 'Disinfectant Solution', DATE '2025-12-31', 'Hygiene Products Ltd.', 'Clinic Storage Room 3', 300, 30);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3007, 'Medical Bench', DATE '2025-12-31', 'Pharma World Inc.', 'Clinic Storage Room 1', 400, 40);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3008, 'First Aid Kits', DATE '2025-12-31', 'Emergency Medical Supplies', 'Clinic Storage Room 2', 50, 5);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3009, 'Blood Pressure Monitors', DATE '2025-12-31', 'MediTech Innovations', 'Clinic Storage Room 3', 50, 5);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3010, 'Stethoscopes', DATE '2025-12-31', 'Medical Instruments Ltd.', 'Clinic Storage Room 1', 30, 3);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3011, 'Blood', DATE '2025-06-30', 'Donation', 'Blood Bank A', 50, 5);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3012, 'Blood', DATE '2025-09-15', 'Donation', 'Blood Bank B', 70, 2);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3013, 'Blood', DATE '2025-10-20', 'Donation', 'Blood Bank A', 60, 8);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3014, 'Blood', DATE '2025-07-10', 'Donation', 'Blood Bank D', 80, 6);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3015, 'Blood', DATE '2025-08-25', 'Donation', 'Blood Bank A', 90, 4);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3016, 'Blood', DATE '2025-11-05', 'Donation', 'Blood Bank B', 55, 3);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3017, 'Blood', DATE '2025-06-15', 'Donation', 'Blood Bank B', 75, 7);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3018, 'Blood', DATE '2025-10-30', 'Donation', 'Blood Bank C', 65, 2);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3019, 'Blood', DATE '2025-12-05', 'Donation', 'Blood Bank A', 85, 6);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3020, 'Blood', DATE '2025-07-20', 'Donation', 'Blood Bank B', 100, 10);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3021, 'Oral', DATE '2025-06-30', 'Donation', 'Pharmacy A', 50, 5);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3022, 'Topical', DATE '2025-09-15', 'Donation', 'Pharmacy B', 70, 2);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3023, 'Injectable', DATE '2025-10-20', 'Donation', 'Pharmacy C', 60, 8);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3024, 'Inhalation', DATE '2025-07-10', 'Donation', 'Pharmacy D', 80, 6);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3025, 'Rectal', DATE '2025-08-25', 'Donation', 'Pharmacy E', 90, 4);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3026, 'Intramuscular', DATE '2025-11-05', 'Donation', 'Pharmacy F', 55, 3);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3027, 'Ophthalmic', DATE '2025-06-15', 'Donation', 'Pharmacy G', 75, 7);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3028, 'Nasal', DATE '2025-10-30', 'Donation', 'Pharmacy H', 65, 2);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3029, 'Vaginal', DATE '2025-12-05', 'Donation', 'Pharmacy I', 85, 6);

INSERT INTO MedicalResource (medicalResourceID, itemName, expirationDate, supplier, storageLocation, quantity, quantityOnHold) VALUES (3030, 'Sublingual', DATE '2025-07-20', 'Donation', 'Pharmacy J', 100, 10);

-- Insert commands for Appointment table

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES (6001, 10001, 5001, DATE '2023-01-10');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6002, 10002, 5002, DATE '2023-02-15');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6003, 10003, 5003, DATE '2023-03-20');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6004, 10004, 5004, DATE '2023-04-25');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6005, 10005, 5005, DATE '2023-05-30');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6006, 10006, 5006, DATE '2023-06-05');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6007, 10007, 5007, DATE '2023-07-10');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6008, 10008, 5008, DATE '2023-08-15');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6009, 10009, 5009, DATE '2023-09-20');

INSERT INTO Appointment (appointmentID, studentID, physicianID, appointmentDate) VALUES(6010, 10010, 5010, DATE '2023-10-25');

-- Insert commands for AppointmentReservedResource table

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6001, 3001, 'Y');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6002, NULL, 'N');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6003, 3003, 'Y');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6004, NULL, 'N');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6005, 3005, 'Y');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6006, 3006, 'Y');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6007, NULL, 'N');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6008, 3008, 'Y');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6009, NULL, 'N');

INSERT INTO AppointmentReservedResource (appointmentID, medicalResourceID, reservedResource) VALUES (6010, 3010, 'Y');

-- Insert commands for Event table

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES (1001, 'Flu Vaccination Drive', TIMESTAMP '2023-10-15 10:00:00', DATE '2023-10-15', 'Preventing Flu Outbreaks');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1002, 'Blood Donation Camp', TIMESTAMP '2023-11-20 09:00:00', DATE '2023-11-20', 'Saving Lives Through Donations');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1003, 'Health Awareness Seminar', TIMESTAMP '2023-12-05 14:00:00', DATE '2023-12-05', 'Promoting Healthy Lifestyles');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1004, 'Dental Check-up Camp', TIMESTAMP '2023-12-20 11:00:00', DATE '2023-12-20', 'Oral Health Awareness');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1005, 'Eye Check-up Camp', TIMESTAMP '2024-01-10 10:30:00', DATE '2024-01-10', 'Vision Care');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1006, 'Sports Injury Prevention Workshop', TIMESTAMP '2024-02-15 15:00:00', DATE '2024-02-15', 'Safe Sports Practices');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1007, 'Nutrition and Diet Counseling Session', TIMESTAMP '2024-03-05 12:00:00', DATE '2024-03-05', 'Healthy Eating Habits');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1008, 'Mental Health Awareness Campaign', TIMESTAMP '2024-04-10 13:30:00', DATE '2024-04-10', 'Destigmatizing Mental Health');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1009, 'First Aid Training Program', TIMESTAMP '2024-05-20 09:30:00', DATE '2024-05-20', 'Emergency Preparedness');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1010, 'Cancer Screening Drive', TIMESTAMP '2024-06-15 08:00:00', DATE '2024-06-15', 'Early Detection Saves Lives');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1020, 'Blood Drive', TIMESTAMP '2024-06-15 08:00:00', DATE '2024-06-15', 'Early Detection Saves Lives');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1011, 'Blood Drive', TIMESTAMP '2024-07-20 09:00:00', DATE '2024-07-20', 'Give the Gift of Life');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1012, 'Blood Drive', TIMESTAMP '2024-08-10 10:00:00', DATE '2024-08-10', 'Promoting Healthy Lifestyles');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1013, 'Blood Drive', TIMESTAMP '2024-09-05 11:00:00', DATE '2024-09-05', 'Protecting Our Community');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1014, 'Blood Drive', TIMESTAMP '2024-10-12 12:00:00', DATE '2024-10-12', 'Vision for All');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1015, 'Blood Drive', TIMESTAMP '2024-11-18 13:00:00', DATE '2024-11-18', 'Smile Bright, Smile Healthy');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1016, 'Blood Drive', TIMESTAMP '2024-12-05 14:00:00', DATE '2024-12-05', 'Keeping Our Environment Clean');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1017, 'Blood Drive', TIMESTAMP '2025-01-15 15:00:00', DATE '2025-01-15', 'Healthy Eating Habits');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1018, 'Blood Drive', TIMESTAMP '2025-02-20 16:00:00', DATE '2025-02-20', 'Get Fit, Stay Active');

INSERT INTO Event (eventID, eventName, timeOfEvent, eventDate, eventTheme) VALUES(1019, 'Blood Drive', TIMESTAMP '2025-03-10 17:00:00', DATE '2025-03-10', 'Breaking the Stigma');

-- Insert commands for BloodSupply table

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES (3011, 1011, 'A+', 15.5, '+', 100, DATE '2023-01-10');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3012, 1012, 'B-', 13.2, '-', 150, DATE '2023-02-15');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3013, 1013, 'O+', 16.8, '+', 120, DATE '2023-03-20');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3014, 1014, 'AB-', 14.5, '-', 80, DATE '2023-04-25');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3015, 1015, 'A-', 12.6, '-', 200, DATE '2023-05-30');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3016, 1016, 'B+', 14.9, '+', 90, DATE '2023-06-05');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3017, 1017, 'O-', 11.8, '-', 110, DATE '2023-07-10');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3018, 1018, 'AB+', 17.2, '+', 70, DATE '2023-08-15');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3019, 1019, 'A-', 12.3, '-', 180, DATE '2023-09-20');

INSERT INTO BloodSupply (bloodSupplyID, bloodDriveID, bloodType, hemoglobinLevel, rhValue, bloodAmount, dateReceived) VALUES(3020, 1020, 'O-', 11.5, '-', 130, DATE '2023-10-25');

-- Insert commands for Medicine table

INSERT INTO Medicine (medicineID, type, details) VALUES (3021, 'Oral', 'Medicine for headaches');

INSERT INTO Medicine (medicineID, type, details) VALUES (3022, 'Topical', 'Skin cream for rashes');

INSERT INTO Medicine (medicineID, type, details) VALUES (3023, 'Intravenous', 'Injectable medicine for allergies');

INSERT INTO Medicine (medicineID, type, details) VALUES (3024, 'Inhalation', 'Medicine for asthma');

INSERT INTO Medicine (medicineID, type, details) VALUES (3025, 'Rectal', 'Suppository for fever');

INSERT INTO Medicine (medicineID, type, details) VALUES (3026, 'Intramuscular', 'Vaccine for flu');

INSERT INTO Medicine (medicineID, type, details) VALUES (3027, 'Ophthalmic', 'Eye drops for allergies');

INSERT INTO Medicine (medicineID, type, details) VALUES (3028, 'Nasal', 'Spray for congestion');

INSERT INTO Medicine (medicineID, type, details) VALUES (3029, 'Vaginal', 'Tablet for yeast infection');

INSERT INTO Medicine (medicineID, type, details) VALUES (3030, 'Sublingual', 'Dissolvable tablet for allergies');

-- Insert commands for MedicalDevices table

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES (3001, TO\_DATE('2025-12-31', 'YYYY-MM-DD'), 'Medical Supplies Inc.', TO\_DATE('2026-02-01', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3002, TO\_DATE('2025-12-31', 'YYYY-MM-DD'), 'Medical Equipment Corp.', TO\_DATE('2026-01-10', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3003, TO\_DATE('2025-12-31', 'YYYY-MM-DD'), 'Health Solutions Ltd.', TO\_DATE('2026-03-01', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3004, TO\_DATE('2025-12-31', 'YYYY-MM-DD'), 'MediTech Supplies', TO\_DATE('2027-03-20', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3005, TO\_DATE('2025-12-31', 'YYYY-MM-DD'), 'Pharma Equipment Co.', TO\_DATE('2026-05-01', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3006, TO\_DATE('2025-12-31', 'YYYY-MM-DD'), 'Medical Devices LLC', TO\_DATE('2026-06-10', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3008, TO\_DATE('2025-06-15', 'YYYY-MM-DD'), 'Healthcare Solutions Inc.', TO\_DATE('2026-07-01', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3009, TO\_DATE('2025-07-10', 'YYYY-MM-DD'), 'MediSupplies Corp.', TO\_DATE('2026-08-01', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3010, TO\_DATE('2025-08-05', 'YYYY-MM-DD'), 'PharmaTech Ltd.', TO\_DATE('2026-09-01', 'YYYY-MM-DD'));

INSERT INTO MedicalDevices (medicalDevicesID, purchaseDate, purchaseFrom, maintenanceDate) VALUES(3007, TO\_DATE('2025-09-20', 'YYYY-MM-DD'), 'Medical Innovations', TO\_DATE('2026-10-10', 'YYYY-MM-DD'));

-- Insert commands for BloodDrive table

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES (1011, 'N', 'N', 'No history of diseases or infections reported');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1012, 'N', 'Y', 'Sickle cell trait detected in some donors');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1013, 'Y', 'N', 'Potential risk of HIV transmission in a few cases');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1014, 'N', 'N', 'No significant medical issues reported in previous drives');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1015, 'N', 'Y', 'Some donors had a history of sickle cell anemia');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1020, 'Y', 'Y', 'Previous instances of HIV and sickle cell cases reported');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1016, 'N', 'N', 'All donors screened negative for HIV and sickle cell');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1017, 'Y', 'Y', 'High prevalence of HIV and sickle cell traits observed');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1018, 'Y', 'N', 'Potential risk of HIV infection identified');

INSERT INTO BloodDrive (bloodDriveID, statusHIV, sickleCell, historyOfDisease) VALUES(1019, 'N', 'N', 'No adverse events reported in past blood drives');

-- Insert commands for EmergencyContact table

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10001, 'Jane Doe', 'Mother', '12125550111', 'jane.doe@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10002, 'Mary Smith', 'Mother', '12125550112', 'mary.smith@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10003, 'David Johnson', 'Father', '12125550113', 'david.johnson@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10004, 'Jessica Williams', 'Mother', '12125550114', 'jessica.williams@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10005, 'Richard Brown', 'Father', '12125550115', 'richard.brown@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10006, 'Carol Jones', 'Guardian', '12125550116', 'carol.jones@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10007, 'James Garcia', 'Father', '12125550117', 'james.garcia@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10008, 'Maria Martinez', 'Mother', '12125550118', 'maria.martinez@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10009, 'Luis Hernandez', 'Father', '12125550119', 'luis.hernandez@example.com');

INSERT INTO EmergencyContact (studentID, name, relation, phone, email) VALUES (10010, 'Sophia Young', 'Mother', '12125550120', 'sophia.young@example.com');

-- Insert commands for Records table

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES (4001, 10001, DATE '2023-05-10','Routine check-up and vaccination');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4002, 10002, DATE '2023-06-20','Treatment for flu symptoms');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4003, 10003, DATE '2023-07-15','Physical examination and medication prescription');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4004, 10004, DATE '2023-08-05','Dressing change for wound care');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4005, 10005, DATE '2023-09-12','Blood pressure monitoring and health assessment');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4006, 10006, DATE '2023-10-25','Follow-up appointment for chronic condition management');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4007, 10007, DATE '2023-11-30','Treatment for allergic reaction');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4008, 10008, DATE '2023-12-18','Dental check-up and cleaning');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4009, 10009, DATE '2024-01-22','Mental health counseling session');

INSERT INTO Records (recordID, studentID, recordDate, treatment) VALUES(4010, 10010, DATE '2024-02-14','Consultation for sports injury rehabilitation');

-- Insert commands for Insurance table

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES ('Blue Cross Blue Shield', 10001, 'ABC123', 'G456', 'M789', 'insurance@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('Aetna', 10002, 'DEF456', 'H789', 'N123', 'aetna@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('Cigna', 10003, 'GHI789', 'I234', 'O456', 'cigna@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('UnitedHealthcare', 10004, 'JKL012', 'J345', 'P789', 'uhc@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('Humana', 10005, 'MNO345', 'K567', 'Q012', 'humana@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('Anthem', 10006, 'PQR678', 'L890', 'R345', 'anthem@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('Kaiser Permanente', 10007, 'STU901', 'M234', 'S678', 'kaiser@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('Molina Healthcare', 10008, 'VWX234', 'N567', 'T901', 'molina@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('MetLife', 10009, 'YZA567', 'O890', 'U234', 'metlife@provider.com', 'Primary');

INSERT INTO Insurance (insuranceName, studentID, policyID, groupID, memberID, contact, primaryOrSecondary) VALUES('GEICO', 10010, 'BCD890', 'P123', 'V567', 'geico@provider.com', 'Primary');

-- Insert commands for Vaccinations table

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES ('MMR', 10001, DATE '2023-01-05', 'Y', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('Polio', 10002, DATE '2023-02-10', 'Y', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('Hepatitis B', 10003, DATE '2023-03-15', 'Y', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('DTaP', 10004, DATE '2023-04-20', 'Y', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('Varicella', 10005, DATE '2023-05-25', 'Y', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('HPV', 10006, DATE '2023-06-30', 'N', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('Meningococcal', 10007, DATE '2023-07-30', 'N', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('Influenza', 10008, DATE '2023-08-15', 'N', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('Tdap', 10009, DATE '2023-09-25', 'N', 'School Nurse');

INSERT INTO Vaccinations (vaccineName, studentID, dateGiven, isMandatory, administeredBy) VALUES('Hib', 10010, DATE '2023-10-30', 'N', 'School Nurse');

-- Insert commands for MinorPatient table

INSERT INTO MinorPatient (minorPatientID) VALUES (10001);

INSERT INTO MinorPatient (minorPatientID) VALUES (10002);

INSERT INTO MinorPatient (minorPatientID) VALUES (10003);

INSERT INTO MinorPatient (minorPatientID) VALUES (10004);

INSERT INTO MinorPatient (minorPatientID) VALUES (10005);

INSERT INTO MinorPatient (minorPatientID) VALUES (10006);

INSERT INTO MinorPatient (minorPatientID) VALUES (10007);

INSERT INTO MinorPatient (minorPatientID) VALUES (10008);

INSERT INTO MinorPatient (minorPatientID) VALUES (10009);

INSERT INTO MinorPatient (minorPatientID) VALUES (10010);

-- Insert commands for LegalGuardian table

INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES (10001, 'John Doe', DATE '1970-05-15', 'Phone', 'Father');

INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10002, 'Susan Smith', DATE '1975-07-20', 'Email', 'Mother');

INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10003, 'Michael Johnson', DATE '1968-11-30', 'Phone', 'Father');

INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10004, 'Emily Williams', DATE '1980-03-25', 'Email', 'Mother');

INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10005, 'Christopher Brown', DATE '1972-09-10', 'Phone', 'Father');

INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10006, 'Jennifer Jones', DATE '1978-01-05', 'Email', 'Guardian');

INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10007, 'Robert Garcia', DATE '1974-06-20', 'Phone', 'Father');

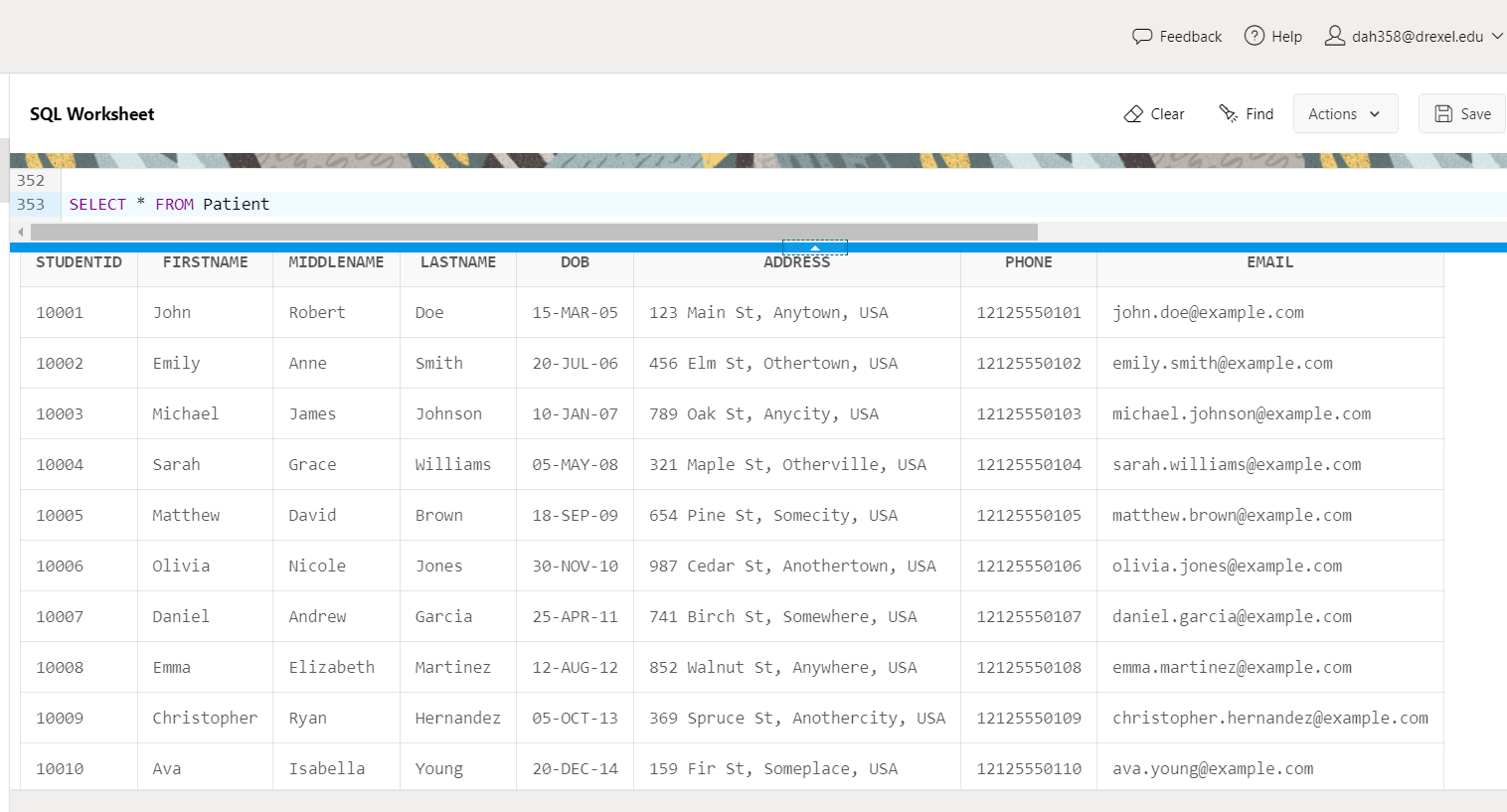
INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10008, 'Jessica Martinez', DATE '1985-04-15', 'Email', 'Mother');

INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10009, 'Daniel Hernandez', DATE '1982-08-30', 'Phone', 'Father');

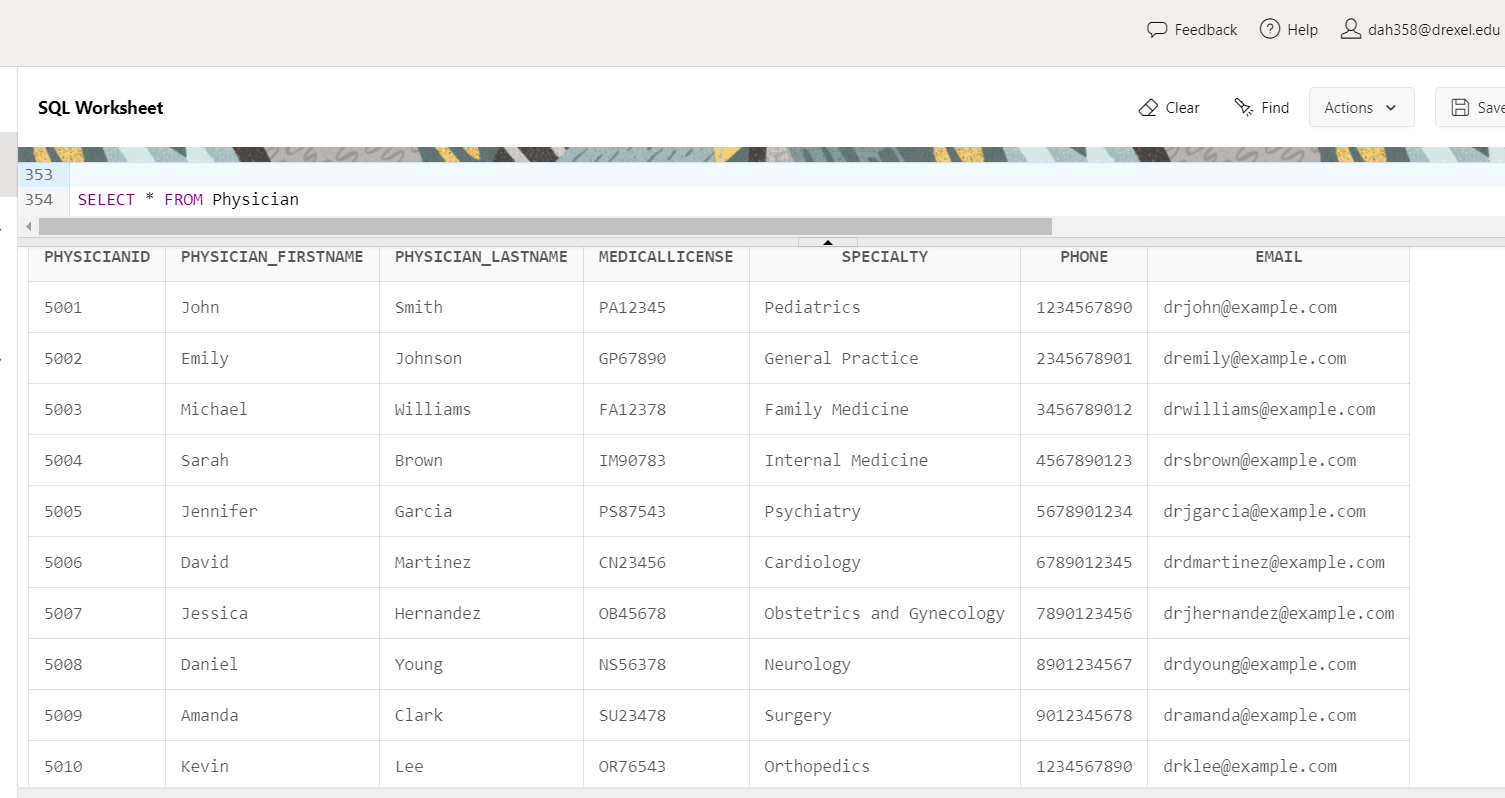
INSERT INTO LegalGuardian (minorPatientID, name, dateOfBirth, primaryModeOfContact, relationToPatient) VALUES(10010, 'Amanda Young', DATE '1977-12-10', 'Email', 'Mother');

**7.3) Snipped output from SELECT \* FROM your\_table\_name; for all tables**

1. Patient



2. MedicalResource



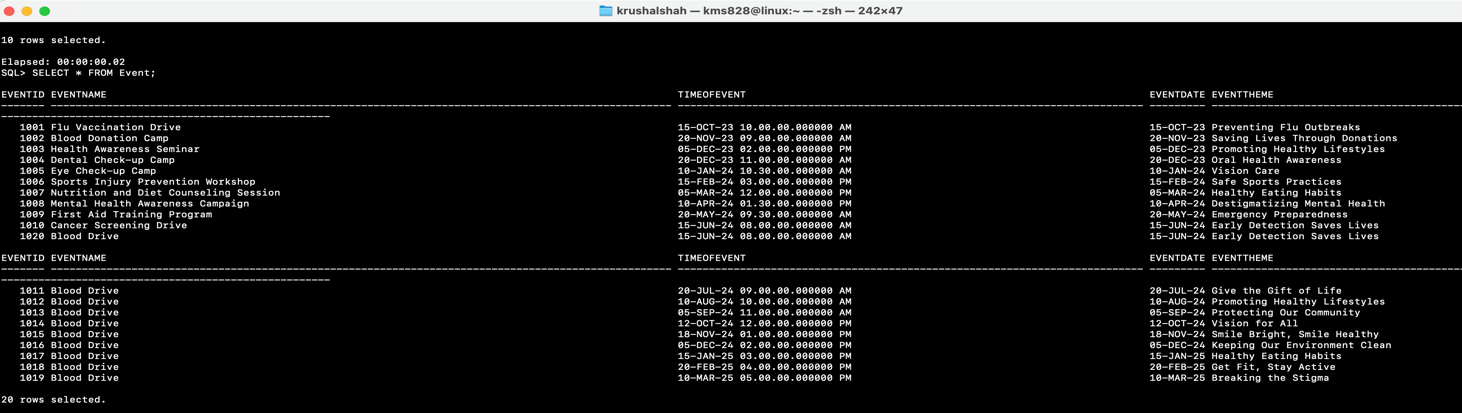
3. MedicineA screenshot of a computer

Description automatically generated

4. MedicalDevicesA screenshot of a computer

Description automatically generated

5. Event



6. BloodDrive

7. EmergencyContact



8. Records

A screenshot of a computer

Description automatically generated

9. InsuranceA screenshot of a computer

Description automatically generated

10. VaccinationsA screenshot of a computer

Description automatically generated

11. MinorPatientA black rectangular object with a black border

Description automatically generated

12. LegalGuardianA screenshot of a computer

Description automatically generated

13. PhysicianA screenshot of a computer

Description automatically generated

14. Appointment

A screenshot of a computer

Description automatically generated

15. AppointmentReservedResource

A screenshot of a computer

Description automatically generated

16. BloodSupply

**A screenshot of a computer

Description automatically generated**

1. **DATA QUERIES**

8.1) **Queries by Stephan Dupoux**

**1.1) Query to get the count of specialties that support primary healthcare data**

COLUMN medicalLicense HEADING 'Medical License' FORMAT A20

COLUMN specialty HEADING 'Specialty' FORMAT A20

COLUMN "COUNT(Doctor.specialty)" HEADING 'Specialty Count' FORMAT 999999

WITH Healthcare AS (

SELECT Ins.insuranceName, Ins.primaryOrSecondary, Ins.studentID, app.physicianID

FROM Insurance Ins

LEFT JOIN Appointment app ON app.studentID = Ins.studentID

WHERE Ins.primaryOrSecondary = 'Primary'

),

Doctor AS (

SELECT doc.physician\_lastName, doc.physician\_firstName, doc.medicalLicense, doc.specialty

FROM Physician doc

LEFT JOIN Healthcare Health ON doc.physicianID = Health.physicianID

)

SELECT Doctor.medicalLicense, Doctor.specialty, COUNT(Doctor.specialty)

FROM Doctor

GROUP BY Doctor.medicalLicense, Doctor.specialty; A screenshot of a computer

Description automatically generated

**1.2) Query to get the Requirement Treatment for the patients on this current date, get their insurance name**

COLUMN firstName HEADING 'First Name' FORMAT A20

COLUMN lastName HEADING 'Last Name' FORMAT A20

COLUMN treatment HEADING 'Treatment' FORMAT A40

COLUMN insuranceName HEADING 'Insurance Name' FORMAT A40

WITH InsuranceRecords AS (

SELECT Rec.treatment, Rec.studentID, Ins.insuranceName

FROM Records Rec

INNER JOIN Insurance Ins ON Rec.studentID = Ins.studentID

WHERE Ins.primaryOrSecondary = 'Primary'

)

SELECT Pat.firstName, Pat.lastName, InRec.treatment, InRec.insuranceName

FROM InsuranceRecords InRec

LEFT JOIN Patient Pat ON InRec.studentID = Pat.studentID;A screenshot of a computer

Description automatically generated

**1.3) Query to get a count of all medicines that are available.**

COLUMN itemName HEADING 'Item Name' FORMAT A40

COLUMN purchaseDate HEADING 'Purchase Date' FORMAT A20

COLUMN quantityOnHold HEADING 'Quantity On Hold' FORMAT 999999

WITH MedicalMetadata AS (

SELECT Vit.medicineID, MD.purchaseDate

FROM Medicine Vit

LEFT JOIN MedicalDevices MD

ON Vit.medicineID = MD.medicalDevicesID

)

SELECT MR.itemName, MM.purchaseDate, MR.quantityOnHold

FROM MedicalResource MR

INNER JOIN MedicalMetadata MM ON MR.medicalResourceID = MM.medicineID

ORDER BY MR.quantityOnHold DESC;A screenshot of a computer

Description automatically generated

8.4) **Queries by Krushal Shah**

**8.4.1)** Retrieve the total number of vaccinations administered by each physician:

SELECT p.physicianID, p.physician\_firstName, p.physician\_lastName, COUNT(v.vaccineName) AS totalVaccinations

FROM PHYSICIAN p

JOIN APPOINTMENT a ON p.physicianID = a.physicianID

JOIN VACCINATIONS v ON a.studentID = v.studentID

GROUP BY p.physicianID, p.physician\_firstName, p.physician\_lastName;

**8.4.2)** Retrieve the number of appointments and the total blood amount donated for each blood drive:

SELECT bd.bloodDriveID, bd.statusHIV, bd.sickleCell, COUNT(a.appointmentID) AS totalAppointments, SUM(bs.bloodAmount) AS totalBloodAmount

FROM BLOODDRIVE bd

LEFT JOIN EVENT e ON bd.bloodDriveID = e.eventID

LEFT JOIN APPOINTMENT a ON e.eventID = a.appointmentID

LEFT JOIN BLOODSUPPLY bs ON bd.bloodDriveID = bs.bloodDriveID

GROUP BY bd.bloodDriveID, bd.statusHIV, bd.sickleCell;

**8.4.3)** Retrieve the top 3 patients with the most appointments:

SELECT p.studentID, p.firstName, p.lastName, COUNT(a.appointmentID) AS totalAppointments

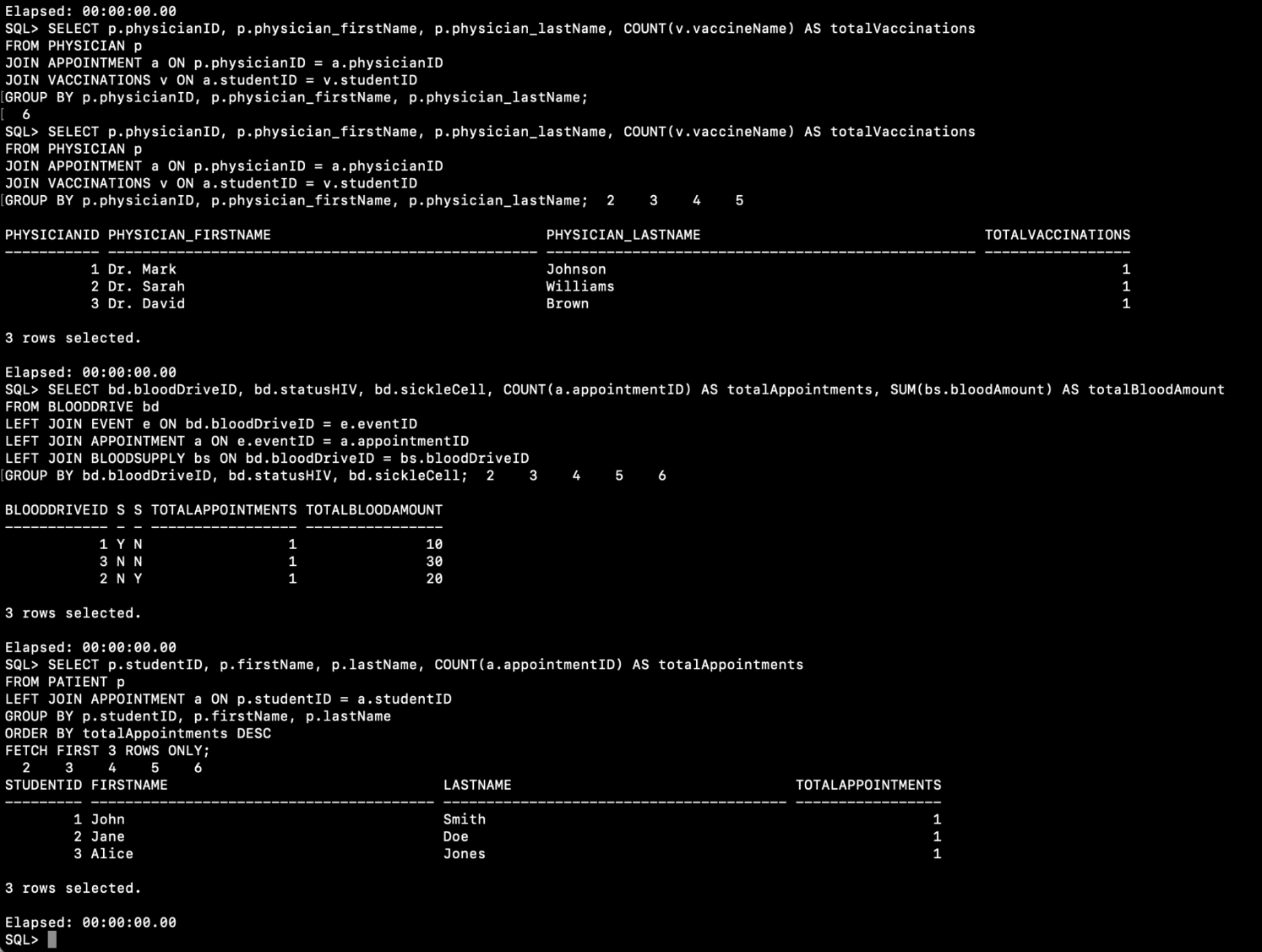
FROM PATIENT p

LEFT JOIN APPOINTMENT a ON p.studentID = a.studentID

GROUP BY p.studentID, p.firstName, p.lastName

ORDER BY totalAppointments DESC

FETCH FIRST 3 ROWS ONLY;



8.2) **Queries By Adesewa Adesida**

* 1. **Query 1 (English meaning, SQL, snipped output, #rows)**

This query retrieves the count of each blood type donated and their respective expiration dates from the Blood Supply table. Useful to know when a new blood drive needs to be held.

**COLUMN EARLYEXPIRATIONDATE FORMAT A20**

**SELECT**

**bloodType,**

**COUNT(\*) AS donationCount,**

**MIN(expirationDate) AS earlyExpirationDate**

**FROM**

**BloodSupply BS**

**INNER JOIN MedicalResource MR ON BS.bloodSupplyID = MR.medicalResourceID**

**WHERE bloodType IS NOT NULL**

**AND expirationDate >= SYSDATE**

**GROUP BY bloodType;**

A screenshot of a computer

Description automatically generated

* 1. **Query 2 (English meaning, SQL, snipped output, #rows)**

Query to retrieve the total number of appointments made by each physician, along with their specialty. Useful to know if a Physician is overbooked.

**--Query to retrieve the total number of appointments made by each physician, along with their specialty.**

**SELECT p.physicianID, p.physician\_firstName, p.physician\_lastName, p.specialty, COUNT(a.appointmentID) AS total\_appointments**

**FROM Physician p**

**LEFT JOIN Appointment a ON p.physicianID = a.physicianID**

**GROUP BY p.physicianID, p.physician\_firstName, p.physician\_lastName, p.specialty;**

A screenshot of a computer

Description automatically generated

* 1. **Query 3 (English meaning, SQL, snipped output, #rows)**

Query to find the total number of appointments for patients with primary healthcare insurance and their corresponding physicians' details.

**-- Query to find the total number of appointments for patients with primary healthcare insurance and their corresponding physicians' details.**

**WITH PrimaryHealthcareInsurance AS (**

**SELECT i.studentID, i.insuranceName, a.physicianID**

**FROM Insurance i**

**INNER JOIN Appointment a ON i.studentID = a.studentID**

**WHERE i.primaryOrSecondary = 'Primary'**

**)**

**SELECT ph.physicianID, ph.physician\_firstName, ph.physician\_lastName, COUNT(pa.studentID) AS total\_appointments**

**FROM Physician ph**

**LEFT JOIN PrimaryHealthcareInsurance pa ON ph.physicianID = pa.physicianID**

**GROUP BY ph.physicianID, ph.physician\_firstName, ph.physician\_lastName;**

**A screenshot of a computer

Description automatically generated**

* 1. **Query 4**

Query to list all patients along with their appointment details, including the physician's last name and appointment date.

**--Query to list all patients along with their appointment details, including the physician's last name and appointment date.**

**SELECT p.studentID, p.firstName, p.lastName, a.appointmentID, ph.physician\_lastName, a.appointmentDate**

**FROM Patient p**

**INNER JOIN Appointment a ON p.studentID = a.studentID**

**LEFT JOIN Physician ph ON a.physicianID = ph.physicianID;**

A screenshot of a computer

Description automatically generated

8.3) **Queries by Donald Hattier (I exceeded the limit on LiveSQl and due to time constraints teammates had to run these queries, big thanks to Krushal)**

**1.1)**

-- Query to list all patients who have not made any appointments and their corresponding insurance details

SELECT p.studentID, p.firstName, p.lastName, i.insuranceName

FROM Patient p

LEFT JOIN Insurance i ON p.studentID = i.studentID

WHERE p.studentID NOT IN (SELECT DISTINCT studentID FROM Appointment);

**1.2)**

-- Retrieves the details of patients along with their emergency contacts who are listed as parents, if available

SELECT p.studentID, p.firstName, p.lastName, ec.name AS emergencyContactName, ec.relation AS emergencyContactRelation, ec.phone AS emergencyContactPhone

FROM Patient p

LEFT JOIN EmergencyContact ec ON p.studentID = ec.studentID

WHERE ec.name IS NOT NULL

AND ec.phone IN (

SELECT phone

FROM EmergencyContact

WHERE relation = 'Parent'

)

ORDER BY p.lastName, p.firstName;

**1.3)**

-- Query retrieves the blood drive IDs and calculates the total number of donors attending each blood drive, filtering out blood drives with fewer than 50 attendees and displaying only those with a reported history of disease

SELECT bd.bloodDriveID, COUNT(\*) AS total\_donors

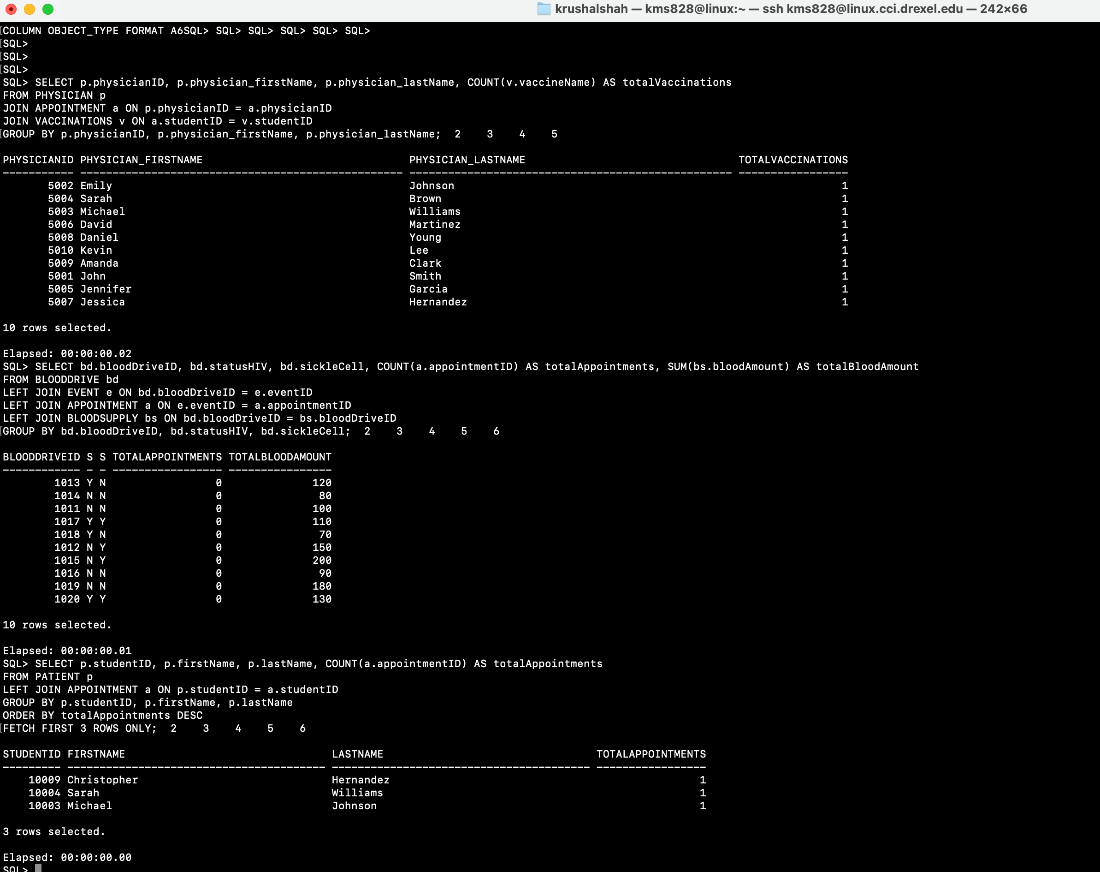
FROM BloodDrive bd

INNER JOIN Event e ON bd.bloodDriveID = e.eventID

LEFT JOIN Patient p ON e.eventID = p.studentID

WHERE bd.historyOfDisease IS NOT NULL

GROUP BY bd.bloodDriveID

HAVING COUNT(\*) >= 50;

1. **DATA MANIPULATION**

**9.1) DML by Stephan Dupoux**

**Update Data**

**Update Insurance Email to Independence@email.com**

UPDATE Insurance

SET CONTACT = 'Independence@email.com'

WHERE InsuranceName = 'Blue Cross Blue Shield';

ROLLBACK;

A screenshot of a computer

Description automatically generated

**Delete all instances of Oral Medicine**

DELETE FROM Medicine WHERE TYPE = 'Oral';

ROLLBACK;

A screenshot of a computer

Description automatically generated

For my sql update command, I wanted to update the email for a primary healthcare insurance system. This command is simple where it finds the contact in the Insurance table that has the InsuranceName of Blue Cross Blue Shield.

For my SQL delete command, we do not hold any more oral medication anymore so it is unnecessary data for us to contain within the database.

**9.2) DML by Adesewa Adesida**

an example of a DML operation for updating a patient's contact information in the "Patient" table:

2.1) **Data before the UPDATE command**

SELECT \* FROM EMERGENCYCONTACT WHERE studentID = 10006;

A screenshot of a computer

Description automatically generated

2.2) **UPDATE command** **and Data after the UPDATE command**

SET AUTOCOMMIT OFF

UPDATE EmergencyContact

SET phone = '1234567890', email = 'ama547@drexel.edu'

WHERE studentID = 10006;

A screenshot of a computer

Description automatically generated

2.4) **ROLLBACK COMMAND**

A screenshot of a computer

Description automatically generated2.5) **Data before the DELETE command**

A screenshot of a computer

Description automatically generated

2.6) **DELETE command** **and Data after the DELETE command**

SET AUTOCOMMIT OFF;

DELETE FROM Patient WHERE studentID = 10006;

A screenshot of a computer

Description automatically generated

2.7) ROLLBACK

ROLLBACK;

A screenshot of a computer

Description automatically generated

**9.2) DML by Donald Hattier (I reached the limit on LiveSQL, credit to Stephan for running my update commands)**

an example of a DML operation for updating a patient's contact information in the "Patient" table:

2.1) Data before the UPDATE command

SET AUTOCOMMIT OFF;

SELECT \*

FROM EmergencyContact

WHERE name = 'Jane Doe';

2.3) Data after the UPDATE command

UPDATE EmergencyContact

SET phone = '3025555555'

WHERE name = 'Jane Doe';

SELECT \*

FROM EmergencyContact

WHERE name = 'Jane Doe';A screen shot of a computer

Description automatically generated

2.4) ROLLBACK

ROLLBACK;

2.5) Data before the DELETE command

SELECT \* FROM Vaccinations;

2.6) DELETE command

SET AUTOCOMMIT OFF;

DELETE FROM Vaccinations

WHERE vaccineName = ‘Tdap’;

2.7) Data after the DELETE command

SELECT \* FROM Vaccinations;

2.8) ROLLBACK

ROLLBACK;

**9.3) DML by Krushal Shah**

9.3.1) Data before the UPDATE command

SET AUTOCOMMIT OFF;

SELECT \*

FROM LegalGuardian

WHERE name = 'John Doe';

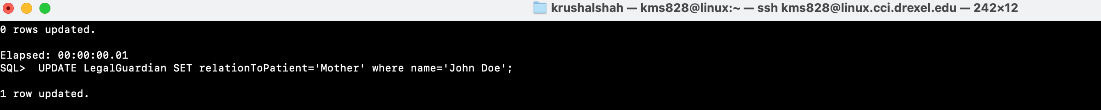


9.3.2) Data after the UPDATE command

UPDATE LegalGuardian

SET relationToPatient = 'Mother'

WHERE name = 'John Doe';



SELECT \*

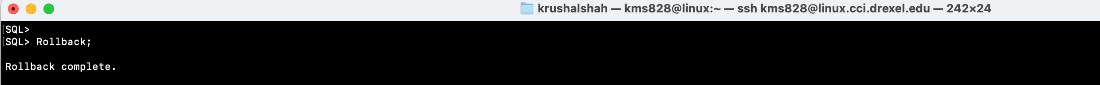
FROM LegalGuardian

WHERE name = 'John Doe';



9.3.3) ROLLBACK

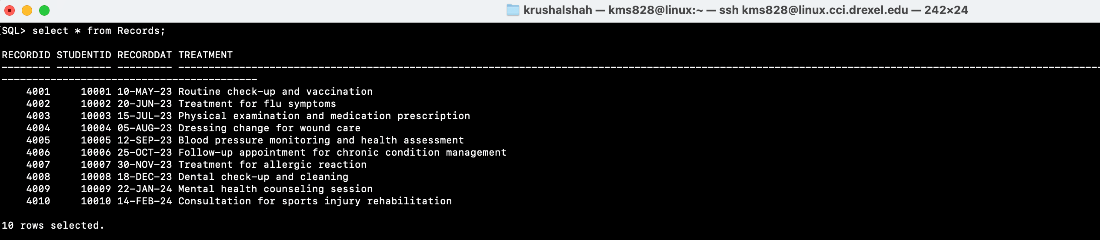
ROLLBACK;





9.3.4) Data before the DELETE command

SELECT \* FROM Records;



9.3.5) DELETE command

SET AUTOCOMMIT OFF;

DELETE FROM Records

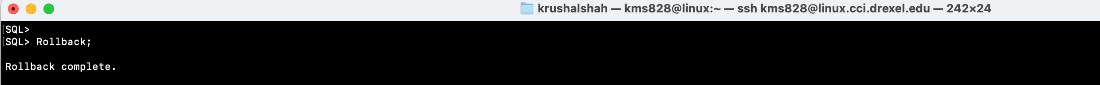
WHERE treatment = ' Treatment for flu symptoms’;

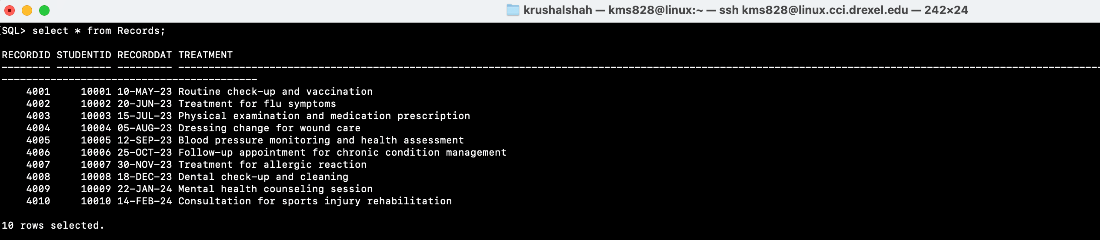


9.3.6) Data after the DELETE command

SELECT \* FROM Records;

9.3.7) ROLLBACK

ROLLBACK ;



1. **Lessons Learned**

(Stephan Dupoux)

This was a solid understanding in building and managing a database overall. I understood more about how databases work and how best to organize data at a lower level. There were choices I think that could be re-done after knowing more about the systems at hand and how better to organize the data. For example, there is a hard missing link that should have been added between medicine and patients. It would have been much easier to connect data about patients’ health statistics and the patient themselves. Because this is not for a real-world application, I think that we could have gotten far more verbose in terms of the data within the tables and the relational schema.

(Adesewa Adesida)

This experience in developing a healthcare database management system was both challenging and enlightening. We began by carefully defining the domain and scope of the project, aligning it with the organization's goals and data management needs. Developing entity-relationship diagrams (ERDs) helped me visualize the database structure, facilitating efficient database creation. Through group work I believe I have gotten better at drawing ERD’s. By Utilizing database software and SQL, we were able to implement complex queries and populate data effectively, streamlining the data management process. One of the key lessons I learned was the critical importance of data accuracy and integrity, particularly in healthcare settings where patient information is sensitive. I learnt the need for proper data validation and security measures to ensure patient confidentiality and comply with regulatory requirements like HIPAA. Challenges arose in optimizing database performance and addressing scalability issues, underscoring the importance of ongoing monitoring and maintenance. Looking forward, I aim to enhance the user interface and accessibility of the database system, while also exploring advanced analytics capabilities to extract actionable insights and improve patient outcomes. Overall, our healthcare database management system stands as an asset in supporting any tertiary level School’s mission to deliver high-quality healthcare services.

(Donald Hattier)

In this project, we tackled healthcare database management, focusing on defining its scope and aligning it with organizational needs. Creating entity-relationship diagrams (ERDs) helped visualize the database structure, making the development process smoother. We leveraged SQL to implement various queries and ensure data accuracy. Populating data emphasized the importance of meticulous validation processes. Our database system is designed to streamline data management processes and ensure compliance. Moving forward, we would aim to explore advanced analytics for better insights and patient outcomes.

(Krushal Shah)

It sounds like I have gained valuable insights from my project! Documenting my project with UML and schema diagrams is indeed crucial for understanding and communicating the structure and design of my database. Learning the syntax and usage of Oracle SQL has allowed me to effectively manage and manipulate my database. Additionally, mastering operations like update, delete, create, joins, and group by functions has enhanced my ability to work with data and extract valuable insights.

1. **Appendix (Optional)**