Database Management Systems

Patient Logging System

Himanshu Gupta - hg387

TABLE OF CONTENTS

1. PROBLEM DESCRIPTION:

We all have been in the Covid19 pandemic for more than a year now. The hospital-level data reveals how much the pandemic has affected hospitals, primary health care facilities, and health workers.

The risk of severity of the illness increases with age. Health care database systems play a major role in monitoring and enhancing the value of health care services leading into well being of the patients.

A medical history about a patient can identify people with a higher-than-usual chance of having certain disorders or illnesses such as certain cancers, diseases related to heart, blood pressure, diabetes, etc. Being aware of the patient's medical history allows the health care workers to reduce the risk.

A certain organization iCare is a healthcare organization that has been at full capacity week after week to take care of the covid patients. iCare encompasses a huge number of locations and healthcare practitioners, totaling four hospitals,3 urgent care centers, and a cancer center, Over 20 doctors support staff relies on the network's software tools to serve their patients satisfactorily.

With the current process of expansion by opening new facilities in several locations, giving birth to the problem of handling huge amounts of patients' history data. This problem is further elaborated in the problem statement with the solution proposed in the goals of this proposal.

a. **Problem statement:** For the health care workers at iCare to operate on the patients, it is important that they have full knowledge about the history of the patient, his/her conditions, allergies, genetic disorders, etc. Currently, there are two systems of maintaining patient records. One system records and maintains everything in excel sheets where a person manually enters every detail related to the patient. The other system does not use a centralized database to record and maintain patients' history. It is becoming difficult to keep up with the pace at which the patients need care week after week during such a pandemic. This arises several problems as follows:

    i. There is no centralized database across 4 locations allowing the patients to be able to go to any location as per their convenience, to avoid multiple data entries at various locations, and inculcate consistency in the data.
    ii. There is a lot of duplicate data in terms of multiple entries related to one person resulting in high inaccuracy.
    iii. There is missing data too. Not all fields are required to record a patient's history.
    iv. There is no family-based history division that helps health workers understand the commonalities within a family.
    v. There is no record of the doctor who attended the patient or the doctor under whose supervision the patient was admitted.
    vi. Since the excel sheets are maintained, the data across various locations being shared is not consistent.
    vii. Also, the accessibility of data is very poor given not all departments have access to each other's data, resulting in redundant difficulty in accessing the data.
    viii. Increase in efficiency and speed

b. **Goals:** The goal of creating a centralized database is to have consistency over the database at all the locations thereby avoiding duplicate data and allowing the patients to choose the location at their convenience. It will automate the process of user verification whose record exists at another

location. It will facilitate the provision of accurate and an up-to-date patient information and his/her history. It will also help the hospital as a whole to avoid medical errors if any. Using this database system allows handling missing data.

c. In Scope:

Patient - It keeps track of patient's basic(demographic) information which include name, sex, race, address, emergency contact information etc. Also, we would maintain the patient's previous medical history along with the allergies while admission into hospital for the first time.

Allergies - Drug allergies, food allergies, height and weight and medication he currently is on

Doctor - Keeps track of the doctor's information like name, age, sex and various other demographical information. Apart from this, name of the hospital and time of work, specialization, level of education, experience are also stored.

Nurse - Keeps track of the nurse's information like name, age, sex and various other demographical information. Apart from this, name of hospital and time of work, specialization, level of education, experience are also stored.

Visits - This entity stores the patient id which is connected to the doctor's id and nurse's id attending to him/her, time stamp of the visit, branch of the hospital, the reason for the visit and medication or tests prescribed in that particular visit. (I'm not sure if we should make another table connecting the visit to the medication or tests prescribed for better knowledge of the person's history in previous visits)

Branches - Name and demographic information of the hospital which is connected to the doctors, nurses and also the patient's visit.

Claims - Claims consists of claim amount and claim status associated using an unique identifier claimId. These are any kind of claims made to the hospital. Moreover, the claim is a weak entity associated with the visits, branches, and the patient as they can't exist without either one of them.

d. Out of Scope:

i. Third party integrations
ii. Visitor information
iii. Appointment scheduling
iv. Insurance

e. Any related systems and open source tools:

i. Open MRS

Open Medical Record System (OpenMRS) is an efficient electronic medical record (EMR) storage and retrieval systems for treating the millions of HIV/AIDS and tuberculosis (TB) patients in the developing world. OpenMRS is a free, open source software, which enables the system to be as widely accessible as possible by sites with limited funding.The system is based on open standards for medical data exchange such as HL7, allowing the exchange of patient data with other medical information systems.

ii. Greenway Health Prime Suite

Greenway Health Prime Suite comes with an EHR module that includes features to enhance monitoring and patient interaction for medical practices. Charting, for instance, is a dashboard that allows to access medical information easily and to understand clinical history. Another function is the entry of instructions, which enables practices to catch laboratory test requests and create e-prescriptions.

iii. InSync

InSync is a healthcare solution that bundles two modules: medical record management (EMR/EHR) and practice management. The EHR module helps practices record and organize patient-related data, such as patient summary reports, notes regarding treatment plans/progress, and e-prescriptions. The practice management module allows medical firms to view patient schedules and financial reports, such as general ledger balance, accounts payable, and accounts receivable.

iv. LiquidEHR

LiquidEHR is a medical records management solution that comes with features such as patient record repository, e-prescribing, document management, billing management, and reporting. The tool also offers a module for managing patient communications and a client portal for scheduling appointments.

v. Praxis

Praxis is an EMR/EHR solution that provides features such as monitoring of medical history, charting, e-prescribing, and a platform for patients. The "concept processing" of the instrument is a crucial capability that helps to generate medications, staff reports, patient directions, and lab orders for identical or related patient cases. Additional features include a module for record administration and integration with billing solutions.

vi. VistA

VistA, an open source EHR solution developed by the US Department of Veterans Affairs (VA), is a highly rated and free alternative to expensive, proprietary EHR software.

f. Assumptions:

- The hospital has a finite number of employees (doctors and nurses) and patients.
- The hospital has five branches.
- Any employee can work at the multiple branches given a doctor can consult for multiple branches, likewise a nurse could be required at the multiple branches.
- Each employee has a domain under which they operate.
- One or more doctors can treat one or many patients.
- Any number of nurses can supervise one or more patients.
- One patient must have at least one visit to the hospital.
- One patient can have zero or more drug allergies.
- One patient should be diagnosed with one condition at one time.
- At least one employee should work at one branch.
- One doctor can diagnose any number of illnesses and an illness should be diagnosed by at least one doctor.
- One visit can result into multiple diagnosed conditions for a patient.
- A patient can file for multiple claims per visit.
- Each patient can give up to 2 phone numbers.

2. REQUIREMENTS SPECIFICATION

1. **Data Management Requirements:**
    a. A robust data management system that can run a large amount of queries concurrently.
    b. Reliably process large volumes of real-time data at any time of the day from any location.
    c. All the data entered must be stored and be made easily accessible for the health care workers only.
    d. The database tables shall contain views that will be appropriately denormalized for ease of use.
    e. No data corruption or malware attack.
    f. Can be evolved or modified over time.
    g. Easy to delete.

2. **Hardware requirement :**
    a. System unit Monitor (VDU)
    b. Uninterrupted power supply (UPS)
    c. RAM (64)
    d. Hard disk capacity of 40GB
    e. Printer

3. **Software Requirement:**
    a. Window XP operating system/ MacOS
    b. Oracle

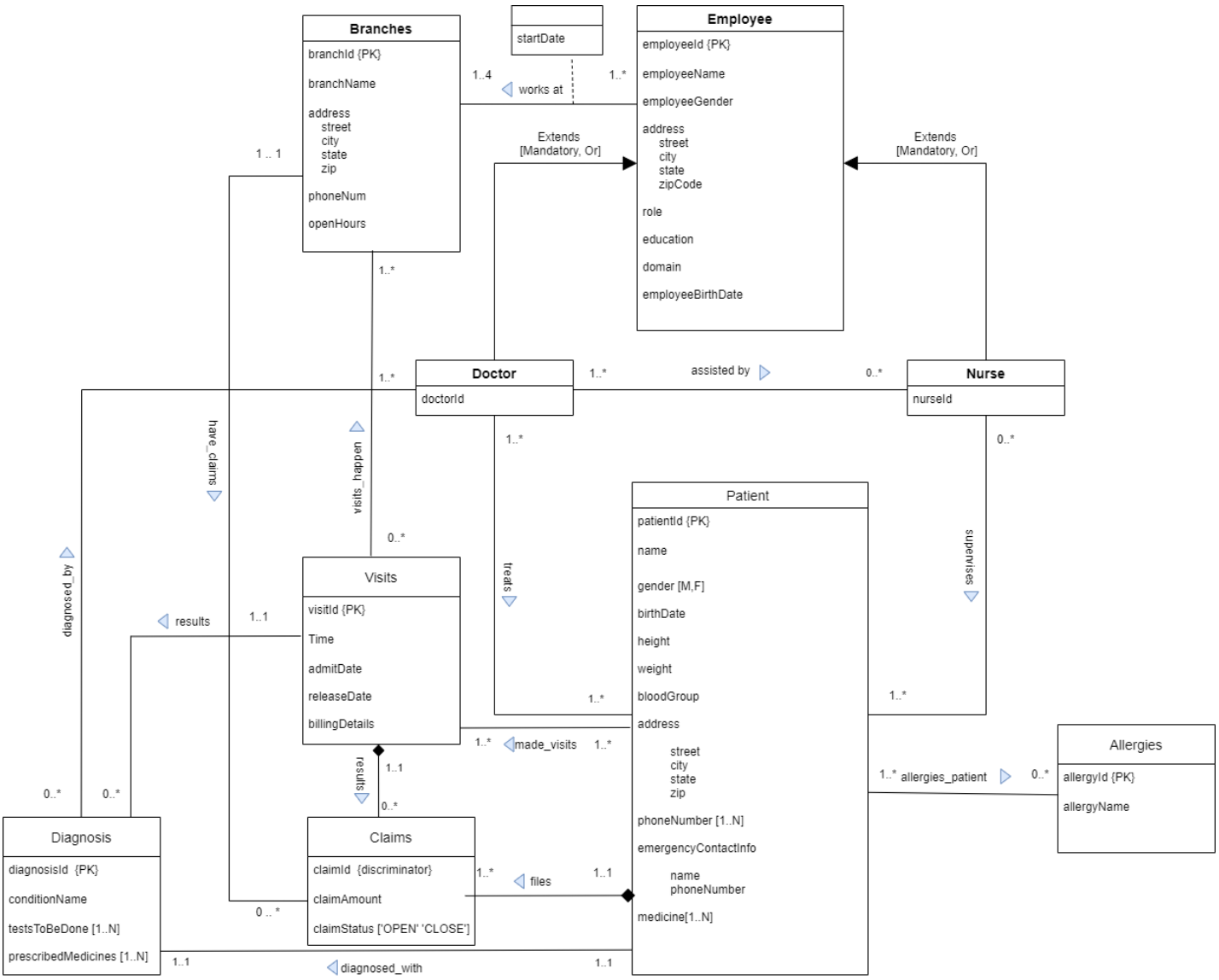
3. CONCEPTUAL DESIGN:

Expected queries

a. Individual specific queries:
    i. Fetch record of a patient, nurse or a doctor.
    ii. Fetch all the records pertaining to one patient from all the branches.
    iii. Fetch record of a doctor.
    iv. Fetch record of a nurse.
    v. Modify a record of an existing patient.
    vi. Modify a record of a doctor.
    vii. Modify a record of a nurse.
    viii. Delete a record of a patient or multiple patients.
    ix. Delete a record of a doctor or multiple doctors.
    x. Delete a record of a nurse or multiple nurses.
    xi.
c. Branch specific queries:
    i. Fetch records of all the patients at a particular Branch.
    ii. Fetch records of all the nurses at a particular Branch.
    iii. Fetch records of all the doctors at a particular Branch.
    iv. Fetch all the locations of the Branch.
    v. Modify the locations of the Branch.
    vi. Delete a Branch.
    vii.
d. Domain specific queries:
    i. Fetch all the domains in the hospital.
    ii. Fetch records of the patient's specific to a domain.
    iii. Fetch records of the doctors/nurses specific to a domain.
    iv.
e. Join queries:
    i. Fetch record of a patient attended by a particular doctor.
    ii. Fetch record of a patient supervised by a particular nurse.
    iii. Fetch record of a patient at a particular branch.
    iv. Fetch record of a patient under specific domain.
    v.

f. Visit specific queries:
    i. Fetch record of patients visited on a specific date.
    ii. Fetch record of patients visited on a specific date at a specific branch.
    iii.
g. Allergies specific queries:
    i. Fetch record of patients having a specific allergy.
    ii. Finding number of patients branch wise having a specific allergy.
    iii.
h. Claims specific queries:
    B. Fetch record of all claims present to a specific branch.
    C. Fetch record of all claims present to a patient.
    D. Fetch record of all the claims associated with a visit.
    E. Fetch record of all the claims made on a specific date.

Entity Relationship Diagram:



4. RELATIONAL SCHEMA

1. Entities:

   Branches, Employee, Doctor, Nurse, Visits, Patients, Allergies, Diagnosis, Claims

2. Special attributes:
   a. Composite attributes:

      emergencyContactInfo, address

3. Initial translation to relational schemas for all identified entities:
    a. Branches(branchId, branchName, addressStreet, addressCity, addressState, addressZip, phoneNum, openHours)
    b. Employee(employeeId, employeeName, employeeGender, addressStreet, addressCity, addressState, addressZip, role, education, domain, birthDate)
    c. Doctor(doctorId)
    d. Nurse(nurseId)
    e. Patient(patientId, name, gender, birthDate, height, weight, bloodGroup, addressStreet, addressCity, addressState, addressZip, phoneNumber, emergencyContactInfoName, emergencyContactInfoPhoneNumber, medicine)
    f. Visits(visitId, Time, admitDate, releaseDate, billingDetails)
    g. Claims(claimId, claimAmount, claimStatus)
    h. Diagnosis(diagnosisId, conditionName, testsToBeDone, prescribedMedicines)
    i. Allergies(allergyId, allergyName)

4. **Cardinality ratios** of all relationships:
    a. Many-to-Many:

        Employee : Branches

        Doctor : Nurse

        Visits : Branches

        Doctor : Patient

        Diagnosis : Doctor

        Patient : Visits

        Patient : Allergies
        Nurse : Patient

    b. One-to-Many:

        Patient : Claims
        Branch : Claims
        Visits : Diagnosis

    c. One-to-One:

        Patient : Diagnosis

5. Translated ERD

a. Many-to-many relationships:

WorksAt(brnNo, empNo, startDate)
Foreign key brnNo references Branches(branchId)
Foreign key empNo references Employee(employeeId)

AssistedBy(doctorAssistedId, nurseAssistedId)
Foreign key doctorAssistedId references Doctor(employeeId)
Foreign key nurseAssistedId references Nurse(employeeId)

Visits_Happen(branchNum, visitNum)
Foreign key branchNum references Branches(branchId)
Foreign key visitNumr references Visits(visitId)

Treats(doctorTreatsId, patientTreatedId)
Foreign key doctorTreatsId references Doctor(employeeId)
Foreign key patientTreatedId references Patient(patientId)

DiagnosedBy(diagnosisNum, docNum)
Foreign key docNum references Doctor(employeeId)
Foreign key diagnosisNum references Diagnosis(diagnosisId)

Made_Visits(patNumber, vstNumber)
Foreign key patNumber references Patient(patientId)
Foreign key vstNumber references Visits(visitId)

b. One-to-many relationships:

Claims(claimId, patNo, visitNo, branchNo, claimAmount, claimStatus)

Foreign key patNo references Patient(patientId)

Foreign key visitNo references Visits(visitId)

Foreign key branchNo references Branches(branchId)

Diagnosis(diagnosisId, conditionName, testsToBeDone, prescribedMedicines, visitNumber)

Foreign key visitNumber references Visits(visitId)

c. One-to-one relationship:

Diagnosis(diagnosisId, conditionName, testsToBeDone, prescribedMedicines, visitNumber, patientNumber)

Foreign key patientNumber references Patient(patientId)

d. Multi-valued attributes:

PatientPhone(patientNumber,phoneNumber)

Foreign key patientNumber references Patient(patientId)

TestsMedicines(diagnosisNumber, testToBeDone, prescribedMedicines)

Foreign key diagnosisNumber references Diagnosis(diagnosisId)

Patient_Med_In_Use(patientMedId, medicine)

Foreign key patientMedId references Patient(patientId)

5. DATA DICTIONARY

Employee: Contains Information about the employees in the hospital.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
EmployeeId	Identification number of the employee	Varchar(25)	All	No	Yes	No

Name	Employee Name	Varchar(25)	All	No	No	No
Gender	Employee Gender	Char(1)	‘M’,‘F’	No	No	No
AddressStreet	Employee residing street	Varchar(50)	All	No	No	No
AddressCity	Employee residing city	Varchar(50)	All	No	No	No
AddressState	Employee residing state	Varchar(2)	All	No	No	No
AddressZip	Employee location Zip code	Number(5)	All	No	No	No
Role	Employee role in hospital	Varchar(25)	All	No	No	No
Education	Employee education	Varchar(100)	All	Yes	No	No
Domain	Employee working domain in hospital	Varchar(25)	‘General’,‘Heart’,‘Primary’,‘Kidney’,‘Dermatology’	No	No	No
Birthdate	Date employee was born	Date	All	Yes	No	No

Constraint	**Type**	**On Delete**
EMPLOYEE_PK	Primary Key	-
EMPLOYEE_NAME	NOT NULL	-
EMPLOYEE_GENDER	NOT NULL	-
domain_C	CHECK	-
EMPLOYEE_DOB	NOT NULL	-
EMPLOYEE_ADDRESS_STREET	NOT NULL	-
EMPLOYEE_ADDRESS_CITY	NOT NULL	-
EMPLOYEE_ADDRESS_STATE	NOT NULL	-

EMPLOYEE_ROLE	NOT NULL	-
EMPLOYEE_ZIP_CODE	NOT NULL	-

Works_At: Contains information of employees from when and which branch they are working.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
StartDate	Date of employee when they started working	Date	All	Yes	No	No
EmployeeId	Identification number of the employee	Varchar(25)	All	No	Yes	Yes
BranchId	Identification number of the branch	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
WORKS_AT_FK_EMP	Foreign Key	-
WORKS_AT_FK_B	Foreign Key	-
PK_WORKS_AT	Primary Key	-

Branch: Contains information about the hospital branches

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
BranchId	Identification number of the branch	Varchar(25)	All	No	Yes	No
BranchName	Name of the branch	Varchar(25)	All	No	No	No
AddressStreet	Branch located Street	Varchar(50)	All	No	No	No
AddressCity	Branch located City	Varchar(50)	All	No	No	No
AddressState	Branch located State	Varchar(2)	All	No	No	No
AddressZip	Branch located Zip code	Number(5)	All	No	No	No
PhoneNumber	Branch phone number	Varchar(15)	All	No	No	No

OpenHours	Branch Open Hours	Varchar(25)	All	Yes	No	No

Constraint	Type	On Delete
BRANCHES_PK	Primary Key	-
BRANCHES_NN_BRANCHNAME	NOT NULL	
BRANCHES_NN_STREET	NOT NULL	
BRANCHES_NN_CITY	NOT NULL	
BRANCHES_NN_STATE	NOT NULL	
BRANCHES_NN_ZIP	NOT NULL	
BRANCHES_NN_PHONENUM	NOT NULL	

Treats: Contains information of patients who are treated by doctors.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
PatientId	Identification number of patient	Varchar(25)	All	No	Yes	Yes
DoctorId	Employee who is doctor	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
EMP_FK	Foreign Key	-
EMPL_FK	Foreign Key	
PK_TREATS	Primary Key	

Patient: Contains information about Patients admitted in the hospitals.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
PatientId	Identification Number of patient	Varchar(25)	All	No	Yes	No
Name	Patient Name	Varchar(50)	All	No	No	No

Gender	Patient Gender	Char(1)	‘M’,‘F’	No	No	No
Birthdate	Date Patient was born	Date	All	No	No	No
Height	Patient Height	Number	All	yes	No	No
Weight	Patient Weight	Number	All	yes	No	No
Blood Group	Patient Blood Group	Varchar(25)	All	No	No	No
AddressStreet	Patient residing street	Varchar(50)	All	No	No	No
AddressCity	Patient residing city	Varchar(50)	All	No	No	No
AddressState	Patient residing state	Varchar(2)	All	No	No	No
AddressZip	Patient residing Zip code	Number(5)	All	No	No	No
EmergencyContactName	Patient’s emergency Contact person name	Varchar(50)	All	No	No	No
EmergencyPhoneNumber	Patient’s emergency Contact Person number	Varchar(15)	All	No	No	No

Constraint	**Type**	**On Delete**
PATIENT_PK	Primary Key	
PATIENT_NAME	NOT NULL	
PATIENT_GENDER	NOT NULL	
patient_CK_gender	Check	
PATIENT_DOB	NOT NULL	
PATIENT_BG	NOT NULL	
PATIENT_ADDRESS_STREET	NOT NULL	
PATIENT_ADDRESS_CITY	NOT NULL	
PATIENT_ADDRESS_STATE	NOT NULL	
PATIENT_ZIPCODE	NOT NULL	
EMERGENCY_CONTACT_NAME	NOT NULL	

EMERGENCY_CONTACT_PHONE	NOT NULL	

Doctor: Contains information of employees who are doctors.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
DoctorId	Employee who is doctor	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
EMPLOYEE_FK	Foreign Key	CASCADE
DOCTOR_PK	Primary Key	CASCADE

Nurse: Contains information of employees who are Nurses.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
NurseId	Employee who is nurse	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
NURSE_FK	Foreign Key	CASCADE
NURSE_PK	Primary Key	CASCADE

Supervises: Contains information of patients who are supervised by nurses.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
PatientId	Identification number of patient	Varchar(25)	All	No	Yes	Yes
NurseId	Employee who is nurse	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete

NURSESUPER_FK	Foreign Key	-
PATIENTS_FK	Foreign Key	-
PK_SUPERVISES	Primary Key	-

Assisted_By: contains information of nurses who are assisting doctors.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
PatientId	Identification number of patient	Varchar(25)	All	No	Yes	Yes
NurseId	Employee who is nurse	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
DOCTORASSIG_FK	Foreign Key	-
NURSEASSIG_FK	Foreign Key	-
PK_ASSISTS	Primary Key	-

Patient_Phone_Number: contains information of phone numbers of patients.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
PatientId	Identification number of patient	Varchar(25)	All	No	Yes	Yes
PhoneNumber	Contact numbers of patient	Varchar(15)	All	No	Yes	No

Constraint	Type	On Delete
PATIENT_ID	Foreign Key	-
PHN_PK	Primary Key	-

Patient_Med_In_Use: contains information of phone numbers of patients.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK

PatientId	Identification number of patient	Varchar(25)	All	No	Yes	Yes
Medicine	Medicines of patient that they are using already	Varchar (75)	All	Yes	Yes	No

Constraint	Type	On Delete
PATIENT_MEDICINE_FK	Foreign Key	-
MEDICINE_PK	Primary Key	-

Allergies: Contains information of allergies that patients have.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
AllergyId	Identification Number of allergy	Varchar(25)	All	No	Yes	No
AllergyName	Name of the allergy	Varchar (50)	All	No	No	No

Constraint	Type	On Delete
ALLERGY_NAME	NOT NULL	-
ALLERGY_PK	Primary Key	-

Allergies_Patient: Contains information of allergies for each patient.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
AllergyId	Identification Number of allergy	Varchar(25)	All	No	Yes	Yes
PatientId	Identification number of patient	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
ALLER_FK	Foreign Key	-
PATIENTALLER_FK	Foreign Key	-
PK_ALLERGIESPATIENT	Primary Key	-

Claims: Contains information of claims of patients.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
ClaimId	Identification number of claim for a patient	Char(12)	All	No	Yes	No
ClaimAmount	Claiming amount of patient	Number	All	No	No	No
ClaimStatus	Status of claimed amount by patient	Varchar (5)	‘Open’,‘Close’	No	No	No
PatientId	Identification Number of patient	Varchar(25)	All	No	Yes	Yes
VisitId	Identification number of visit of a patient	Varchar(25)	All	No	Yes	Yes
BranchId	Identification number of the branch	Varchar(25)	All	No	No	Yes

Constraint	Type	On Delete
CLAIMS_FK_PATIENTNUMBER	Foreign Key	CASCADE
CLAIMS_FK_VISITNUMBER	Foreign Key	CASCADE
CLAIMS_FK_BRANCHNUMBER	Foreign Key	-
CLAIMS_NN_BRANCHNUMBER	NOT NULL	-
CLAIMAMOUNT_NN_CLAIMAMOUNT	NOT NULL	-
CLAIMS_CK_STATUS	Check	-
CLAIMS_PK	Primary Key	-

Visits: Contains information of visits of each patient.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK

VisitId	Identification number of visit of a patient	Varchar(25)	All	No	Yes	No
Time	The time patient visited the hospital	Varchar2(15)	All	No	No	No
AdmitDate	The date patient admitted to hospital	Date	All	No	No	No
ReleaseDate	The date patient is released from hospital	Date	All	No	Yes	No
BillingDetails	Billing data of patient on a visit	Varchar(100)	All	No	Yes	No

Constraint	Type	On Delete
VISITS_PK	Primary Key	-
VISITS_NN_TIME	NOT NULL	-
VISITS_NN_ADMITDATE	NOT NULL	-

Visits_Happen: Contains information of patients visiting specific branches.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
VisitId	Identification number of visit of a patient	Varchar(25)	All	No	Yes	Yes
BranchId	Identification number of the branch	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
VISITS_HAPPEN_FK_V	Foreign Key	-
VISITS_HAPPEN_FK_B	Foreign Key	-
PK_VISITS_HAPPEN	Primary Key	-

Diagnosis: Contains information of diagnosis done to the patient in a visit to hospital.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK

DiagnosisId	Identification Number of disease	Varchar(25)	All	No	Yes	No
ConditionName	Name of the condition patient having	Varchar(50)	All	No	No	No
PatientId	Identification Number of patient	Varchar(25)	All	No	No	Yes
VisitId	Identification number of visit of a patient	Varchar(25)	All	No	No	Yes

Constraint	Type	On Delete
DIAGNOSIS_PK	Primary Key	-
DIAGNOSIS_FK_PATIENTNUMBER	Foreign Key	-
DIAGNOSIS_NN_PATIENTID	NOT NULL	-
DIAGNOSIS_UQ_PATIENTID	UNIQUE	-
DIAGNOSIS_FK_VISITNUMBER	Foreign Key	-
DIAGNOSIS_NN_VISITNUMBER	NOT NULL	-
DIAGNOSIS_UQ_VISITNUMBER	UNIQUE	-
DIAGNOSIS_NN_CONDITIONNAME	NOT NULL	-

Test_Medicines: Contains information of tests to be done and medicines for a given diagnosis.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
DiagnosisId	Identification Number of disease	Varchar(25)	All	No	Yes	yes

TestsToBeDone	Tests to be done for the patient for given condition	Varchar(50)	All	Yes	Yes	No
PrescribedMedicines	Medicines to be taken for the condition	Varchar(50)	All	Yes	Yes	No

Constraint	Type	On Delete
PK_TESTSMEDICINES	Primary Key	-
TESTSMEDICINES_FK_ID	Foreign Key	-

Diagnosed_By: Contains information of the diagnosis done by the doctors.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
DiagnosisId	Identification Number of disease	Varchar(25)	All	No	Yes	yes
DoctorId	Employee who is doctor	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
PK_DIAGNOSIS_BY	Primary Key	-
DIAGNOSED_BY_FK_ID	Foreign Key	-
DIAGNOSED_BY_FK_EMPLOYEENUMBER	Foreign Key	-

Made_Visits: Contains information about visits of every patient.

Attribute Name	Description	Datatype	Domain	Nullable	PK	FK
PatientId	Identification Number of patient	Varchar(25)	All	No	Yes	Yes
VisitId	Identification number of visit of a patient	Varchar(25)	All	No	Yes	Yes

Constraint	Type	On Delete
PK_MADE_VISITS	Primary Key	-

MADE_VISITS_FK_P	Foreign Key	-
MADE_VISITS_FK_V	Foreign Key	-

6. DATABASE IMPLEMENTATION

Insertion queries:

- Aishwarya DDL Section Start - asp344

- 1

```
CREATE TABLE Patient(
        patientId VARCHAR(25) CONSTRAINT PATIENT_PK PRIMARY KEY,
        name VARCHAR(50) CONSTRAINT PATIENT_NAME NOT NULL,
        gender VARCHAR(1) CONSTRAINT PATIENT_GENDER NOT NULL CONSTRAINT
patient_CK_gender CHECK(gender = 'M' OR gender = 'F'),
        birthDate Date CONSTRAINT PATIENT_DOB NOT NULL,
        height NUMBER,
        weight NUMBER,
        bloodGroup VARCHAR(25) CONSTRAINT PATIENT_BG NOT NULL,
        addressStreet VARCHAR(50) CONSTRAINT PATIENT_ADDRESS_STREET NOT NULL,
        addressCity VARCHAR(50) CONSTRAINT PATIENT_ADDRESS_CITY NOT NULL,
        addressState VARCHAR(2) CONSTRAINT PATIENT_ADDRESS_STATE NOT NULL,
        zip NUMBER(5) CONSTRAINT PATIENT_ZIPCODE NOT NULL,
        emerContactName VARCHAR(50) CONSTRAINT EMERGENCY_CONTACT_NAME NOT
NULL,
        emerContactNum VARCHAR(15) CONSTRAINT EMERGENCY_CONTACT_PHONE NOT
NULL
```

-)

- Table created.

- Statement

- 2

```
CREATE TABLE Employee(
        employeeId VARCHAR(25) CONSTRAINT EMPLOYEE_PK PRIMARY KEY,
        employeeName VARCHAR(25) CONSTRAINT EMPLOYEE_NAME NOT NULL,
        employeeGender VARCHAR(1) CONSTRAINT EMPLOYEE_GENDER NOT NULL,
        role varchar(25) CONSTRAINT EMPLOYEE_ROLE NOT NULL,
        education varchar(100),
```

```
        domain VARCHAR(25) CONSTRAINT domain_C CHECK (domain in
('General','Heart','Primary','Kidney','Dermatology')),
        employeeBirthDate Date CONSTRAINT EMPLOYEE_DOB NOT NULL,
        addressStreet VARCHAR(50) CONSTRAINT EMPLOYEE_ADDRESS_STREET NOT NULL,
        addressCity VARCHAR(50) CONSTRAINT EMPLOYEE_ADDRESS_CITY NOT NULL,
        addressState VARCHAR(2) CONSTRAINT EMPLOYEE_ADDRESS_STATE NOT NULL,
        zipCode NUMBER(5) CONSTRAINT EMPLOYEE_ZIP_CODE NOT NULL
```

-)
- Table created.

- Statement

- 3

```
CREATE TABLE Doctor(
   doctorId VARCHAR(25) CONSTRAINT DOCTOR_PK PRIMARY KEY CONSTRAINT
EMPLOYEE_FK REFERENCES Employee(employeeId)
   ON DELETE CASCADE
```

-)
- Table created.

- Statement

- 4

```
CREATE TABLE Nurse(
   nurseId VARCHAR(25) CONSTRAINT NURSE_PK PRIMARY KEY CONSTRAINT NURSE_FK
REFERENCES EMPLOYEE(employeeId)
   ON DELETE CASCADE
```

-)
- Table created.

- Aishwarya DDL Section End - asp344

- 5

```
CREATE TABLE Branches(
        branchId VARCHAR(25) CONSTRAINT BRANCHES_PK PRIMARY KEY,
        branchName VARCHAR(25) CONSTRAINT BRANCHES_NN_BRANCHNAME NOT
NULL,
        addressStreet VARCHAR(50) CONSTRAINT BRANCHES_NN_STREET NOT NULL,
```

addressCity VARCHAR(50) CONSTRAINT BRANCHES_NN_CITY NOT NULL,
addressState VARCHAR(2) CONSTRAINT BRANCHES_NN_STATE NOT NULL,
addressZip NUMBER(5) CONSTRAINT BRANCHES_NN_ZIP NOT NULL,
phoneNum VARCHAR(15) CONSTRAINT BRANCHES_NN_PHONENUM NOT NULL,
openHours VARCHAR(25)

-)
- Table created.

- Himanshu DDL Section Start - hg387

- 6

```
CREATE TABLE Visits(
    visitId VARCHAR(25) CONSTRAINT VISITS_PK PRIMARY KEY,
                Time DATE CONSTRAINT VISITS_NN_TIME NOT NULL,
    admitDate DATE CONSTRAINT VISITS_NN_ADMITDATE NOT NULL,
    releaseDate DATE,
    billingDetails VARCHAR(100)
```

-)
- Table created.

- Statement

- 7

```
CREATE TABLE Allergies(
    allergyId VARCHAR(25) CONSTRAINT ALLERGY_PK PRIMARY KEY,
    allergyName VARCHAR(50) CONSTRAINT ALLERGY_NAME NOT NULL
```

-)
- Table created.

- Statement

- 8

```
CREATE TABLE Diagnosis(
    diagnosisId VARCHAR(25) CONSTRAINT DIAGNOSIS_PK PRIMARY KEY,
    patientNumber VARCHAR(25) CONSTRAINT DIAGNOSIS_FK_PATIENTNUMBER
REFERENCES PATIENT(patientId)
                CONSTRAINT DIAGNOSIS_NN_PATIENTID NOT NULL
                CONSTRAINT DIAGNOSIS_UQ_PATIENTID UNIQUE,
```

visitNumber VARCHAR(25) CONSTRAINT DIAGNOSIS_FK_VISITNUMBER REFERENCES Visits(visitId)
CONSTRAINT DIAGNOSIS_NN_VISITNUMBER NOT NULL
CONSTRAINT DIAGNOSIS_UQ_VISITNUMBER UNIQUE,
conditionName VARCHAR(50) CONSTRAINT DIAGNOSIS_NN_CONDITIONNAME NOT NULL

-)
- Table created.

- Statement

- 9

CREATE TABLE TestsMedicines(
diagnosisNumber VARCHAR(25) CONSTRAINT TESTSMEDICINES_FK_ID REFERENCES Diagnosis(diagnosisId),
testToBeDone VARCHAR(50),
prescribedMedicines VARCHAR(50),
CONSTRAINT PK_TESTSMEDICINES PRIMARY KEY(diagnosisNumber, testToBeDone, prescribedMedicines)

-)
- Table created.

- Himanshu DDL Section End - hg387

- 10

CREATE TABLE Diagnosed_By(
diagnosisNum VARCHAR(25) CONSTRAINT DIAGNOSED_BY_FK_ID REFERENCES Diagnosis(diagnosisId),
docNum VARCHAR(25) CONSTRAINT DIAGNOSED_BY_FK_EMPLOYEENUMBER REFERENCES Doctor(doctorId),
CONSTRAINT PK_DIAGNOSIS_BY PRIMARY KEY(diagnosisNum, docNum)

-)
- Table created.

- Sravya DDL Section Start - sa3648

- 11

```
CREATE TABLE Visits_Happen(
                visitNum VARCHAR(25) CONSTRAINT VISITS_HAPPEN_FK_V
REFERENCES Visits(visitId),
                branchNum VARCHAR(25) CONSTRAINT VISITS_HAPPEN_FK_B
REFERENCES Branches(branchId),
        CONSTRAINT PK_VISITS_HAPPEN PRIMARY KEY(visitNum, branchNum)
```

-)
- Table created.

- Statement

- 12

```
CREATE TABLE Made_Visits(
                patNumber VARCHAR(25) CONSTRAINT MADE_VISITS_FK_P
REFERENCES Patient(patientId),
                vstNumber VARCHAR(25) CONSTRAINT MADE_VISITS_FK_V
REFERENCES Visits(visitId),
        CONSTRAINT PK_MADE_VISITS PRIMARY KEY(patNumber, vstNumber)
```

-)
- Table created.

- Statement

- 13

```
CREATE TABLE Claims(
        claimId VARCHAR(25),
                patNo VARCHAR(25) CONSTRAINT CLAIMS_FK_PATIENTNUMBER
REFERENCES Patient(patientId) ON DELETE CASCADE,
                visitNo VARCHAR(25) CONSTRAINT CLAIMS_FK_VISITNUMBER
REFERENCES Visits(visitId) ON DELETE CASCADE,
                branchNo VARCHAR(25) CONSTRAINT CLAIMS_FK_BRANCHNUMBER
REFERENCES Branches(branchId) CONSTRAINT CLAIMS_NN_BRANCHNUMBER NOT NULL,
                claimAmount NUMBER CONSTRAINT
CLAIMAMOUNT_NN_CLAIMAMOUNT NOT NULL,
        claimStatus VARCHAR(5) CONSTRAINT CLAIMS_CK_STATUS CHECK(claimStatus =
'OPEN' OR claimStatus = 'CLOSE'),
                CONSTRAINT CLAIMS_PK PRIMARY KEY(claimId, patNo, visitNo)
```

-)
- Table created.

- Statement

- 14

```
CREATE TABLE Works_At(
        startDate DATE,
        empNo VARCHAR(25) CONSTRAINT WORKS_AT_FK_EMP REFERENCES
Employee(employeeId),
        brnNo VARCHAR(25) CONSTRAINT WORKS_AT_FK_B REFERENCES
Branches(branchId),
        CONSTRAINT PK_WORKS_AT PRIMARY KEY(empNo, brnNo)
```

-)
- Table created.

- Sravya DDL Section End - sa3648

- 15

```
CREATE TABLE Treats(
                doctorTreatsId VARCHAR(25) CONSTRAINT EMP_FK REFERENCES
Doctor(doctorId),
                patientTreatedId VARCHAR(25) CONSTRAINT EMPL_FK REFERENCES
Patient(patientId),
                CONSTRAINT PK_TREATS PRIMARY KEY(doctorTreatsId, patientTreatedId)
```

-)
- Table created.

- Devi Supraja DDL Section Start - dr986

- 16

```
CREATE TABLE Supervises(
                nurseSuperVisesId VARCHAR(25) CONSTRAINT NURSESUPER_FK
REFERENCES Nurse(nurseId),
                patientSupervisedId VARCHAR(25) CONSTRAINT PATIENTS_FK REFERENCES
Patient(patientId),
                CONSTRAINT PK_SUPERVISES PRIMARY KEY(nurseSuperVisesId,
patientSupervisedId)
```

-)

- Table created.

- Statement

- 17

```
CREATE TABLE AssistedBy(
                doctorAssistedId VARCHAR(25) CONSTRAINT DOCTORASSIG_FK
REFERENCES Doctor(doctorId),
                nurseAssistedId VARCHAR(25) CONSTRAINT NURSEASSIG_FK REFERENCES
Nurse(nurseId),
                CONSTRAINT PK_ASSISTS PRIMARY KEY(doctorAssistedId, nurseAssistedId)
```

-)
- Table created.

- Statement

- 18

```
CREATE TABLE Allergies_Patient(
                allergyPatId VARCHAR(25) CONSTRAINT ALLER_FK REFERENCES
Allergies(allergyId),
                patientAllergyId VARCHAR(25) CONSTRAINT PATIENTALLER_FK REFERENCES
Patient(patientId),
                CONSTRAINT PK_ALLERGIESPATIENT PRIMARY KEY(allergyPatId,
patientAllergyId)
```

-)
- Table created.

- Statement

- 19

```
CREATE TABLE Patient_Phone_Number(
        patientPhoneId VARCHAR(25) CONSTRAINT PATIENT_ID REFERENCES
Patient(patientId),
        phoneNumber VARCHAR(15),
        CONSTRAINT PHN_PK PRIMARY KEY(patientPhoneId, phoneNumber)
```

-)
- Table created.

- Devi Supraja DDL Section End - dr986

- 20

```
CREATE TABLE Patient_Med_In_Use(
        patientMedId VARCHAR(25) CONSTRAINT PATIENT_MEDICINE_FK REFERENCES
Patient(patientId),
        medicine VARCHAR(75),
        CONSTRAINT MEDICINE_PK PRIMARY KEY(patientMedId, medicine)
```

-)
- Table created.

7. DATA

- Aishwarya Insert Section Start - asp344

- 1
- INSERT INTO Patient(patientId, name, gender, birthDate, height, weight, bloodGroup,addressStreet, addressCity, addressState, zip, emerContactName, emerContactNum) VALUES ('p1', 'Deepika', 'F', TO_DATE('18-AUG-04', 'dd-MM-yy'), 162, 60, 'O-', '3 Shipley Ave. ', 'West Springfield', 'MA', 49509, 'Ria', '4567890456')
- 1 row(s) inserted.

- Statement

- 2
- INSERT INTO Patient(patientId, name, gender, birthDate, height, weight, bloodGroup,addressStreet, addressCity, addressState, zip, emerContactName, emerContactNum) VALUES ('p2', 'Atharva', 'M', TO_DATE('18-MAY-01', 'dd-MM-yy'), 182, 70, 'B+', '879 Lawrence St. ', 'Delevan', 'NY', 40050, 'Aarya', '1234567899')
- 1 row(s) inserted.

- Statement

- 3
- INSERT INTO Patient(patientId, name, gender, birthDate, height, weight, bloodGroup,addressStreet, addressCity, addressState, zip, emerContactName, emerContactNum) VALUES ('p3', 'Aditya', 'M', TO_DATE('08-SEP-99', 'dd-MM-yy'), 172, 65, 'B-', '1800 Spring garden St', 'Philadelphia', 'PA', 19104, 'Shelby', '0987654321')
- 1 row(s) inserted.

- Statement

- 4
- INSERT INTO Patient(patientId, name, gender, birthDate, height, weight, bloodGroup,addressStreet, addressCity, addressState, zip, emerContactName, emerContactNum) VALUES ('p4', 'Zenobia', 'F', TO_DATE('04-JAN-94', 'dd-MM-yy'), 122, 45, 'A-', '3203 Race St', 'Baltimore', 'MD', 19104, 'Irene', '6789054321')
- 1 row(s) inserted.

- Statement

- 5
- INSERT INTO Patient(patientId, name, gender, birthDate, height, weight, bloodGroup,addressStreet, addressCity, addressState, zip, emerContactName, emerContactNum) VALUES ('p5', 'Shriya', 'F', TO_DATE('18-JUL-97', 'dd-MM-yy'), 155, 57, 'B+', '16 Shelly Drive', 'Elk Grove', 'NJ', 50456, 'Pam', '1122334455')
- 1 row(s) inserted.

- Statement

- 6

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e1','John Doe','M','Doctor','MBBS', 'General', TO_DATE('01-JUN-82', 'dd-MM-yy'), '3047 Highland View Drive', 'Elk Grove', 'NJ', '08901')
- 1 row(s) inserted.

- Statement

- 7

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e2','Delena Dodson','F','Doctor','MD','Heart', TO_DATE('12-JUL-81', 'dd-MM-yy'), '4855 James Martin Circle', 'Philadelphia', 'PA', '19111')
- 1 row(s) inserted.

- Statement

- 8

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e3','Robin Borton','M','Doctor','MD','Primary', TO_DATE('29-SEP-80', 'dd-MM-yy'), '877 Bryan Street', 'Delevan', 'NY', '14042')
- 1 row(s) inserted.
- Statement
- 9

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e4','Reginia Isham','F','Doctor','MBBS','Kidney', TO_DATE('16-AUG-82', 'dd-MM-yy'), '1353 Harron Drive', 'Baltimore', 'MD', '21202')
- 1 row(s) inserted.
- Statement
- 10

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e5','Billi Yurick','M','Doctor','MD','Dermatology', TO_DATE('23-JUN-79', 'dd-MM-yy'), '2247 Levy Court', 'Cambridge', 'MA', '02141')
- 1 row(s) inserted.
- Statement
- 11

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e6','Wilbur Nickels','M','Nurse','LPN','General', TO_DATE('04-JAN-85', 'dd-MM-yy'), '2658 Davisson Street', 'Elk Grove', 'NJ', '08901')
- 1 row(s) inserted.
- Statement
- 12

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e7','Tina Rayner','F','Nurse','BS','Heart', TO_DATE('22-JUL-84', 'dd-MM-yy'), '2598 Parrill Court', 'Philadelphia', 'PA', '19112')
- 1 row(s) inserted.
- Statement
- 13

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e8','Cecilia Hampshire','F','Nurse','LPN','Primary', TO_DATE('19-MAY-87', 'dd-MM-yy'), '3378 Middleville Road', 'Delevan', 'NY', '14044')
- 1 row(s) inserted.
- Statement
- 14

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e9','Marlon Whitis','M','Nurse','ADN','Kidney', TO_DATE('21-SEP-82', 'dd-MM-yy'), '825 Vine Street', 'Baltimore', 'MD', '21203')
- 1 row(s) inserted.
- Statement
- 15

INSERT INTO Employee (employeeId, employeeName, employeeGender, role, education, domain, employeeBirthDate, addressStreet, addressCity, addressState, zipCode)

- VALUES ('e10','Carry Mirsky','M','Nurse','BS','Dermatology', TO_DATE('27-APR-87', 'dd-MM-yy'), '4436 Burwell Heights Road', 'Cambridge', 'MA', '02144')
- 1 row(s) inserted.
- Statement
- 16

INSERT INTO Doctor (doctorId)

- VALUES ('e1')
- 1 row(s) inserted.
- Statement
- 17

INSERT INTO Doctor (doctorId)

- VALUES ('e2')
- 1 row(s) inserted.
- Statement
- 18

INSERT INTO Doctor (doctorId)

- VALUES ('e3')
- 1 row(s) inserted.

- Statement
- 19

INSERT INTO Doctor (doctorId)
- VALUES ('e4')
- 1 row(s) inserted.
- Statement
- 20

INSERT INTO Doctor (doctorId)
- VALUES ('e5')
- 1 row(s) inserted.
- Statement
- 21

INSERT INTO Nurse (nurseId)
- VALUES ('e6')
- 1 row(s) inserted.
- Statement
- 22

INSERT INTO Nurse (nurseId)
- VALUES ('e7')
- 1 row(s) inserted.
- Statement
- 23

INSERT INTO Nurse (nurseId)
- VALUES ('e8')
- 1 row(s) inserted.
- Statement
- 24

INSERT INTO Nurse (nurseId)
- VALUES ('e9')
- 1 row(s) inserted.
- Statement
- 25

INSERT INTO Nurse (nurseId)
- VALUES ('e10')

- 1 row(s) inserted.
  - Statement
- 26

INSERT INTO Branches (branchId, branchName, addressStreet, addressCity, addressState, addressZip, phoneNum, openHours)

- VALUES ('b1', 'New Jersey Branch', '4476 Center Street', 'Elk Grove', 'NJ', 08911, '(432) 242-9108', '24 Hours')
- 1 row(s) inserted.
  - Statement
- 27

INSERT INTO Branches (branchId, branchName, addressStreet, addressCity, addressState, addressZip, phoneNum, openHours)

- VALUES ('b2', 'Pennsylvania Branch', '1451 Spirit Drive', 'Philadelphia', 'PA', 19122, '(378) 996-0188', '24 Hours')
- 1 row(s) inserted.
  - Statement
- 28

INSERT INTO Branches (branchId, branchName, addressStreet, addressCity, addressState, addressZip, phoneNum, openHours)

- VALUES ('b3', 'New York Branch', '988 Pick Street', 'Delevan', 'NY', 14054, '(792) 399-5976', '24 Hours')
- 1 row(s) inserted.
  - Statement
- 29

INSERT INTO Branches (branchId, branchName, addressStreet, addressCity, addressState, addressZip, phoneNum, openHours)

- VALUES ('b4', 'Maryland Branch', '43 Woodland Terrace', 'Baltimore', 'MD', 21222, '(206) 558-1560', '24 Hours')
- 1 row(s) inserted.
  - Statement
- 30

INSERT INTO Branches (branchId, branchName, addressStreet, addressCity, addressState, addressZip, phoneNum, openHours)

- VALUES ('b5', 'Massachusetts Branch', '2751 Massachusetts Avenue', 'Cambridge', 'MA', 08911, '(970) 778-0874', '24 Hours')
- 1 row(s) inserted.

- Aishwarya Insert Section End - asp344

- 31
- INSERT INTO Visits(visitId, time, admitDate, releaseDate, billingDetails) VALUES('v1', TO_DATE('03.02.2020:10:34:24','DD.MM.YYYY:HH24:MI:SS'), TO_DATE('03.02.2020:11:40:24','DD.MM.YYYY:HH24:MI:SS'),TO_DATE('07.02.2020:10:01:17','DD.MM.YYYY:HH24:MI:SS'), 'Bill Amount - 1000 Dollars(Paid)')
- 1 row(s) inserted.

- Himanshu Insert Section Start - hg387

- 32
- INSERT INTO Visits(visitId, time, admitDate, releaseDate, billingDetails) VALUES('v2', TO_DATE('03.03.2020:01:34:10','DD.MM.YYYY:HH24:MI:SS'), TO_DATE('05.03.2020:11:40:01','DD.MM.YYYY:HH24:MI:SS'),TO_DATE('07.03.2020:11:02:10','DD.MM.YYYY:HH24:MI:SS'), 'Bill Amount - 1500 Dollars(Paid)')
- 1 row(s) inserted.

- Statement

- 33
- INSERT INTO Visits(visitId, time, admitDate, releaseDate, billingDetails) VALUES('v3', TO_DATE('05.04.2020:11:40:01','DD.MM.YYYY:HH24:MI:SS'), TO_DATE('05.04.2020:12:50:01','DD.MM.YYYY:HH24:MI:SS'),TO_DATE('06.04.2020:11:02:10','DD.MM.YYYY:HH24:MI:SS'), 'Bill Amount - 2500 Dollars(Paid)')
- 1 row(s) inserted.

- Statement

- 34
- INSERT INTO Visits(visitId, time, admitDate, releaseDate, billingDetails) VALUES('v4', TO_DATE('05.05.2020:11:41:01','DD.MM.YYYY:HH24:MI:SS'), TO_DATE('05.05.2020:12:52:30','DD.MM.YYYY:HH24:MI:SS'),TO_DATE('15.05.2020:11:02:10','DD.MM.YYYY:HH24:MI:SS'), 'Bill Amount - 5500 Dollars(Paid)')
- 1 row(s) inserted.

- Statement

- 35
- INSERT INTO Visits(visitId, time, admitDate, releaseDate, billingDetails) VALUES('v5', TO_DATE('15.09.2020:09:30:10','DD.MM.YYYY:HH24:MI:SS'), TO_DATE('15.09.2020:09:30:10','DD.MM.YYYY:HH24:MI:SS'),TO_DATE('15.09.2020:12:22:15','DD.MM.YYYY:HH24:MI:SS'), 'Bill Amount - 500 Dollars(Paid)')
- 1 row(s) inserted.
- Statement
- 36

INSERT INTO allergies (allergyId, allergyName)
- VALUES ('a1', 'Penicillin')
- 1 row(s) inserted.
- Statement
- 37

INSERT INTO allergies (allergyId, allergyName)
- VALUES ('a2', 'Antibiotics')
- 1 row(s) inserted.
- Statement
- 38

INSERT INTO allergies (allergyId, allergyName)
- VALUES ('a3', 'NSAIDs')
- 1 row(s) inserted.
- Statement
- 39

INSERT INTO allergies (allergyId, allergyName)
- VALUES ('a4', 'Anticonvulsants')
- 1 row(s) inserted.
- Statement
- 40

INSERT INTO allergies (allergyId, allergyName)
- VALUES ('a5', 'Chemotherapy drugs')
- 1 row(s) inserted.
- Statement
- 41

- INSERT INTO Diagnosis(diagnosisId, patientNumber, visitNumber, conditionName) Values ('dg1','p1', 'v1', 'Viral Fever')
- 1 row(s) inserted.
- Statement
- 42
- INSERT INTO Diagnosis(diagnosisId, patientNumber, visitNumber, conditionName) Values ('dg2','p2', 'v2', 'Stroke')
- 1 row(s) inserted.
- Statement
- 43
- INSERT INTO Diagnosis(diagnosisId, patientNumber, visitNumber, conditionName) Values ('dg3','p3', 'v3', 'Kidney Stones')
- 1 row(s) inserted.
- Statement
- 44
- INSERT INTO Diagnosis(diagnosisId, patientNumber, visitNumber, conditionName) Values ('dg4','p4', 'v4', 'COVID')
- 1 row(s) inserted.
- Statement
- 45
- INSERT INTO Diagnosis(diagnosisId, patientNumber, visitNumber, conditionName) Values ('dg5','p5', 'v5', 'Skin Pigmentation')
- 1 row(s) inserted.
- Statement
- 46
- INSERT INTO TestsMedicines(diagnosisNumber, testToBeDone, prescribedMedicines) Values ('dg1','Physical Checkup','Dolo-65')
- 1 row(s) inserted.
- Statement
- 47
- INSERT INTO TestsMedicines(diagnosisNumber, testToBeDone, prescribedMedicines) Values ('dg2','MRI','tPA')

- 1 row(s) inserted.
- Statement
- 48
- INSERT INTO TestsMedicines(diagnosisNumber, testToBeDone, prescribedMedicines) Values ('dg3','CT Scan','ibuprofen')
- 1 row(s) inserted.
- Statement
- 49
- INSERT INTO TestsMedicines(diagnosisNumber, testToBeDone, prescribedMedicines) Values ('dg4','PCR test','Paracetamol')
- 1 row(s) inserted.
- Statement
- 50
- INSERT INTO TestsMedicines(diagnosisNumber, testToBeDone, prescribedMedicines) Values ('dg5','Biopsy','Clindac Spray')
- 1 row(s) inserted.
- Statement
- 51
- INSERT INTO Diagnosed_by(diagnosisNum, docNum) VALUES('dg1','e1')
- 1 row(s) inserted.
- Statement
- 52
- INSERT INTO Diagnosed_by(diagnosisNum, docNum) VALUES('dg2','e2')
- 1 row(s) inserted.
- Statement
- 53
- INSERT INTO Diagnosed_by(diagnosisNum, docNum) VALUES('dg3','e3')
- 1 row(s) inserted.
- Statement
- 54
- INSERT INTO Diagnosed_by(diagnosisNum, docNum) VALUES('dg4','e4')
- 1 row(s) inserted.
- Statement

- 55
- INSERT INTO Diagnosed_by(diagnosisNum, docNum) VALUES('dg5','e5')
- 1 row(s) inserted.
- Statement
- 56
- INSERT INTO Visits_Happen(visitNum, branchNum) Values('v1', 'b5')
- 1 row(s) inserted.
- Statement
- 57
- INSERT INTO Visits_Happen(visitNum, branchNum) Values('v2', 'b3')
- 1 row(s) inserted.
- Statement
- 58
- INSERT INTO Visits_Happen(visitNum, branchNum) Values('v3', 'b2')
- 1 row(s) inserted.
- Statement
- 59
- INSERT INTO Visits_Happen(visitNum, branchNum) Values('v4', 'b4')
- 1 row(s) inserted.
- Statement
- 60
- INSERT INTO Visits_Happen(visitNum, branchNum) Values('v5', 'b1')
- 1 row(s) inserted.
- Statement
- 61
- INSERT INTO Made_Visits(patNumber, vstNumber) Values('p1', 'v1')
- 1 row(s) inserted.
- Statement
- 62
- INSERT INTO Made_Visits(patNumber, vstNumber) Values('p2', 'v2')
- 1 row(s) inserted.
- Statement
- 63

- INSERT INTO Made_Visits(patNumber, vstNumber) Values('p3', 'v3')
- 1 row(s) inserted.

- Himanshu Insert Section End - hg387

- 64
- INSERT INTO Made_Visits(patNumber, vstNumber) Values('p4', 'v4')
- 1 row(s) inserted.

- Sravya Insert Section Start - sa3648

- 65
- INSERT INTO Made_Visits(patNumber, vstNumber) Values('p5', 'v5')
- 1 row(s) inserted.

- Statement

- 66
- INSERT INTO Claims(claimId, patNo, visitNo, branchNo, claimAmount, claimStatus) VALUES ('c1', 'p1', 'v1', 'b5',800, 'CLOSE')
- 1 row(s) inserted.

- Statement

- 67
- INSERT INTO Claims(claimId, patNo, visitNo, branchNo, claimAmount, claimStatus) VALUES ('c2', 'p2', 'v2', 'b3',1300, 'CLOSE')
- 1 row(s) inserted.

- Statement

- 68
- INSERT INTO Claims(claimId, patNo, visitNo, branchNo, claimAmount, claimStatus) VALUES ('c3', 'p3', 'v3', 'b2',1800, 'OPEN')
- 1 row(s) inserted.

- Statement

- 69
- INSERT INTO Claims(claimId, patNo, visitNo, branchNo, claimAmount, claimStatus) VALUES ('c4', 'p4', 'v4', 'b4',4800, 'CLOSE')
- 1 row(s) inserted.

- Statement

- 70

- INSERT INTO Claims(claimId, patNo, visitNo, branchNo, claimAmount, claimStatus) VALUES ('c5', 'p5', 'v5', 'b1',400, 'OPEN')
- 1 row(s) inserted.
- Statement
- 71

INSERT INTO Works_At (startDate, empNo, brnNo)
- VALUES (TO_DATE('01-JUN-19', 'dd-MM-yy'), 'e1','b1')
- 1 row(s) inserted.
- Statement
- 72

INSERT INTO Works_At (startDate, empNo, brnNo)
- VALUES (TO_DATE('15-JUL-19', 'dd-MM-yy'), 'e6','b1')
- 1 row(s) inserted.
- Statement
- 73

INSERT INTO Works_At (startDate, empNo, brnNo)
- VALUES (TO_DATE('13-JUN-19', 'dd-MM-yy'), 'e2','b2')
- 1 row(s) inserted.
- Statement
- 74

INSERT INTO Works_At (startDate, empNo, brnNo)
- VALUES (TO_DATE('20-AUG-19', 'dd-MM-yy'), 'e7','b2')
- 1 row(s) inserted.
- Statement
- 75

INSERT INTO Works_At (startDate, empNo, brnNo)
- VALUES (TO_DATE('29-AUG-19', 'dd-MM-yy'), 'e3','b3')
- 1 row(s) inserted.
- Statement
- 76

INSERT INTO Works_At (startDate, empNo, brnNo)
- VALUES (TO_DATE('12-SEP-19', 'dd-MM-yy'), 'e8','b3')
- 1 row(s) inserted.
- Statement

- 77

INSERT INTO Works_At (startDate, empNo, brnNo)

- VALUES (TO_DATE('10-JUN-19', 'dd-MM-yy'), 'e4','b4')
- 1 row(s) inserted.
- Statement
- 78

INSERT INTO Works_At (startDate, empNo, brnNo)

- VALUES (TO_DATE('11-JUL-19', 'dd-MM-yy'), 'e9','b4')
- 1 row(s) inserted.
- Statement
- 79

INSERT INTO Works_At (startDate, empNo, brnNo)

- VALUES (TO_DATE('01-JUN-19', 'dd-MM-yy'), 'e5','b5')
- 1 row(s) inserted.
- Statement
- 80

INSERT INTO Works_At (startDate, empNo, brnNo)

- VALUES (TO_DATE('15-JUN-19', 'dd-MM-yy'), 'e10','b5')
- 1 row(s) inserted.
- Statement
- 81

INSERT INTO Treats (doctorTreatsId, patientTreatedId)

- VALUES ('e1', 'p1')
- 1 row(s) inserted.
- Statement
- 82

INSERT INTO Treats (doctorTreatsId, patientTreatedId)

- VALUES ('e2', 'p2')
- 1 row(s) inserted.
- Statement
- 83

INSERT INTO Treats (doctorTreatsId, patientTreatedId)

- VALUES ('e2', 'p3')
- 1 row(s) inserted.

- Statement
- 84

INSERT INTO Treats (doctorTreatsId, patientTreatedId)
- VALUES ('e3', 'p3')
- 1 row(s) inserted.
- Statement
- 85

INSERT INTO Treats (doctorTreatsId, patientTreatedId)
- VALUES ('e4', 'p4')
- 1 row(s) inserted.
- Statement
- 86

INSERT INTO Treats (doctorTreatsId, patientTreatedId)
- VALUES ('e4', 'p5')
- 1 row(s) inserted.
- Statement
- 87

INSERT INTO Treats (doctorTreatsId, patientTreatedId)
- VALUES ('e5', 'p5')
- 1 row(s) inserted.
- Statement
- 88

INSERT INTO Supervises (nurseSuperVisesId, patientSupervisedId)
- VALUES ('e6', 'p1')
- 1 row(s) inserted.
- Statement
- 89

INSERT INTO Supervises (nurseSuperVisesId, patientSupervisedId)
- VALUES ('e6', 'p2')
- 1 row(s) inserted.
- Statement
- 90

INSERT INTO Supervises (nurseSuperVisesId, patientSupervisedId)
- VALUES ('e7', 'p2')

- 1 row(s) inserted.
- Statement
- 91

INSERT INTO Supervises (nurseSuperVisesId, patientSupervisedId)

- VALUES ('e7', 'p3')
- 1 row(s) inserted.
- Statement
- 92

INSERT INTO Supervises (nurseSuperVisesId, patientSupervisedId)

- VALUES ('e8', 'p3')
- 1 row(s) inserted.
- Statement
- 93

INSERT INTO Supervises (nurseSuperVisesId, patientSupervisedId)

- VALUES ('e9', 'p4')
- 1 row(s) inserted.
- Statement
- 94

INSERT INTO Supervises (nurseSuperVisesId, patientSupervisedId)

- VALUES ('e10', 'p5')
- 1 row(s) inserted.
- Statement
- 95

INSERT INTO assistedBy (doctorAssistedId, nurseAssistedId)

- VALUES ('e1', 'e6')
- 1 row(s) inserted.
- Sravya Insert Section End - sa3648
- 96

INSERT INTO assistedBy (doctorAssistedId, nurseAssistedId)

- VALUES ('e2', 'e7')
- 1 row(s) inserted.

- Devi Supraja Insert Section Start - dr986

- 97

INSERT INTO assistedBy (doctorAssistedId, nurseAssistedId)

- VALUES ('e3', 'e8')
- 1 row(s) inserted.
- Statement
- 98

INSERT INTO assistedBy (doctorAssistedId, nurseAssistedId)

- VALUES ('e4', 'e9')
- 1 row(s) inserted.
- Statement
- 99

INSERT INTO assistedBy (doctorAssistedId, nurseAssistedId)

- VALUES ('e5', 'e10')
- 1 row(s) inserted.
- Statement
- 100

INSERT INTO Allergies_Patient (allergyPatId, patientAllergyId)

- VALUES ('a1', 'p1')
- 1 row(s) inserted.
- Statement
- 101

INSERT INTO Allergies_Patient (allergyPatId, patientAllergyId)

- VALUES ('a2', 'p2')
- 1 row(s) inserted.
- Statement
- 102

INSERT INTO Allergies_Patient (allergyPatId, patientAllergyId)

- VALUES ('a2', 'p3')
- 1 row(s) inserted.
- Statement
- 103

INSERT INTO Allergies_Patient (allergyPatId, patientAllergyId)

- VALUES ('a3', 'p3')
- 1 row(s) inserted.

- Statement
- 104

INSERT INTO Allergies_Patient (allergyPatId, patientAllergyId)
- VALUES ('a4', 'p4')
- 1 row(s) inserted.
- Statement
- 105

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p1', '(990) 907-1865')
- 1 row(s) inserted.
- Statement
- 106

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p1', '(572) 653-3900')
- 1 row(s) inserted.
- Statement
- 107

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p2', '(525) 415-1068')
- 1 row(s) inserted.
- Statement
- 108

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p2', '(357) 207-7417')
- 1 row(s) inserted.
- Statement
- 109

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p3', '(511) 479-6598')
- 1 row(s) inserted.
- Statement
- 110

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p3', '(822) 709-8028')

- 1 row(s) inserted.
- Statement
- 111

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p4', '(273) 755-1095')
- 1 row(s) inserted.
- Statement
- 112

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p4', '(412) 541-5397')
- 1 row(s) inserted.
- Statement
- 113

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p5', '(498) 363-2062')
- 1 row(s) inserted.
- Statement
- 114

INSERT INTO Patient_Phone_Number (patientPhoneId, phoneNumber)
- VALUES ('p5', '(238) 671-6771')
- 1 row(s) inserted.
- Statement
- 115

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)
- VALUES ('p1', 'Acetaminophen')
- 1 row(s) inserted.
- Statement
- 116

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)
- VALUES ('p1', 'Cephalexin')
- 1 row(s) inserted.
- Statement
- 117

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)

- VALUES ('p2', 'Adderall')
- 1 row(s) inserted.
- Statement
- 118

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)
- VALUES ('p3', 'Fentanyl')
- 1 row(s) inserted.
- Statement
- 119

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)
- VALUES ('p3', 'Acetaminophen')
- 1 row(s) inserted.
- Statement
- 120

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)
- VALUES ('p3', 'Methadone')
- 1 row(s) inserted.
- Statement
- 121

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)
- VALUES ('p4', 'Omeprazole')
- 1 row(s) inserted.
- Statement
- 122

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)
- VALUES ('p4', 'Fentanyl')
- 1 row(s) inserted.
- Statement
- 123

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)
- VALUES ('p5', 'Acetaminophen')
- 1 row(s) inserted.
- Statement
- 124

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)

- VALUES ('p5', 'Paracetamol')
- 1 row(s) inserted.

- Statement

- 125

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)

- VALUES ('p5', 'Advil')
- 1 row(s) inserted.

- Devi Supraja Insert Section End - dr986

- 126

INSERT INTO Patient_Med_In_Use (patientMedId, medicine)

- VALUES ('p5', 'Claritin')
- 1 row(s) inserted.

Select queries:

- Aishwarya Select Section Start -asp344

- 1
- SELECT * FROM Patient
-

PATIENTID	NAME	GENDER	BIRTHDATE	HEIGHT	WEIGHT	BLOODGROUP	ADDRESSSTREET	ADDRESSCITY	ADDRESSSTATE	ZIP	EMERCONTACTNAME	EMERCONTACTNUM

p1	Deepika	F	18-AUG-04	162	60	O-	3 Shipley Ave.	West Springfield	MA	49509	Ria	4567890456
p2	Atharva	M	18-MAY-01	182	70	B+	879 Lawrence St.	Delevan	NY	40050	Aarya	1234567899
p3	Aditya	M	08-SEP-99	172	65	B-	1800 Spring garden St	Philadelphia	PA	19104	Shelby	0987654321
p4	Zenobia	F	04-JAN-94	122	45	A-	3203 Race St	Baltimore	MD	19104	Irene	6789054321
p5	Shriya	F	18-JUL-97	155	57	B+	16 Shelly Drive	Elk Grove	NJ	50456	Pam	1122334455

5 rows selected.

- Statement

- 2
- SELECT * FROM Employee
-

EMPLOYEEID	EMPLOYEENAME	EMPLOYEEGENDER	ROLE	EDUCATION	DOMAIN	EMPLOYEEBIRTHDATE	ADDRESSSTREET	ADDRESSSCITY	ADDRESSSTATE	ZIPCODE
e1	John Doe	M	Doctor	MBBS	General	01-JUN-82	3047 Highland View Drive	Elk Grove	NJ	8901
e2	Delena Dodson	F	Doctor	MD	Heart	12-JUL-81	4855 James Martin Circle	Philadelphia	PA	19111

e3	Robin Borton	M	Doctor	MD	Primary	29-SEP-80	877 Bryan Street	Delevan	NY	14042
e4	Reginia Isham	F	Doctor	MBBS	Kidney	16-AUG-82	1353 Harron Drive	Baltimore	MD	21202
e5	Billi Yurick	M	Doctor	MD	Dermatology	23-JUN-79	2247 Levy Court	Cambridge	MA	2141
e6	Wilbur Nickels	M	Nurse	LPN	General	04-JAN-85	2658 Davisson Street	Elk Grove	NJ	8901
e7	Tina Rayner	F	Nurse	BS	Heart	22-JUL-84	2598 Parrill Court	Philadelphia	PA	19112
e8	Cecilia Hampshire	F	Nurse	LPN	Primary	19-MAY-87	3378 Middleville Road	Delevan	NY	14044
e9	Marlon Whitis	M	Nurse	ADN	Kidney	21-SEP-82	825 Vine Street	Baltimore	MD	21203
e10	Carry Mirsky	M	Nurse	BS	Dermatology	27-APR-87	4436 Burwell Heights Road	Cambridge	MA	2144

10 rows selected.

- Statement

- 3
- SELECT * FROM Doctor
-

DOCTORID

e1
e2
e3
e4
e5

5 rows selected.

- Statement

- 4
- SELECT * FROM Nurse
-

NURSEID
e10
e6
e7
e8
e9

5 rows selected.

- Aishwarya Select Section End - asp344

- 5
- SELECT * FROM Branches
-

BRANCHID	BRANCHNAME	ADDRESSSTREET	ADDRESSCITY	ADDRESSSTATE	ADDRESSZIP	PHONENUM	OPENHOURS

b1	New Jersey Branch	4476 Center Street	Elk Grove	NJ	8911	(432) 242-9108	24 Hours
b2	Pennsylvania Branch	1451 Spirit Drive	Philadelphia	PA	19122	(378) 996-0188	24 Hours
b3	New York Branch	988 Pick Street	Delevan	NY	14054	(792) 399-5976	24 Hours
b4	Maryland Branch	43 Woodland Terrace	Baltimore	MD	21222	(206) 558-1560	24 Hours
b5	Massachusetts Branch	2751 Massachusetts Avenue	Cambridge	MA	8911	(970) 778-0874	24 Hours

5 rows selected.

- Himanshu Select Section Start - hg387

- 6
- SELECT * FROM Visits
-

VISITID	TIME	ADMIT DATE	RELEASEDATE	BILLINGDETAILS
v1	03-FEB-20	03-FEB-20	07-FEB-20	Bill Amount - 1000 Dollars(Paid)
v2	03-MAR-20	05-MAR-20	07-MAR-20	Bill Amount - 1500 Dollars(Paid)
v3	05-APR-20	05-APR-20	06-APR-20	Bill Amount - 2500 Dollars(Paid)

v4	05-MAY-20	05-MAY-20	15-MAY-20	Bill Amount - 5500 Dollars(Paid)
v5	15-SEP-20	15-SEP-20	15-SEP-20	Bill Amount - 500 Dollars(Paid)

5 rows selected.

- Statement

- 7
- SELECT * FROM Allergies
-

ALLERGYID	ALLERGYNAME
a1	Penicillin
a2	Antibiotics
a3	NSAIDs
a4	Anticonvulsants
a5	Chemotherapy drugs

5 rows selected.

- Statement

- 8
- SELECT * FROM Diagnosis
-

DIAGNOSISID	PATIENTNUMBER	VISITNUMBER	CONDITIONNAME
dg1	p1	v1	Viral Fever

dg2	p2	v2	Stroke
dg3	p3	v3	Kidney Stones
dg4	p4	v4	COVID
dg5	p5	v5	Skin Pigmentation

5 rows selected.

- Statement

- 9
- SELECT * FROM TestsMedicines
-

DIAGNOSISNUMBER	TESTTOBEDONE	PRESCRIBEDMEDICINES
dg1	Physical Checkup	Dolo-65
dg2	MRI	tPA
dg3	CT Scan	ibuprofen
dg4	PCR test	Paracetamol
dg5	Biopsy	Clindac Spray

5 rows selected.

- Himanshu Select Section End - hg387

- 10
- SELECT * FROM Diagnosed_By
-

DIAGNOSISNUM	DOCNUM
dg1	e1

dg2	e2
dg3	e3
dg4	e4
dg5	e5

5 rows selected.

- Sravya Select Section Start - sa3648

- 11
- SELECT * FROM Visits_Happen
-

VISITNUM	BRANCHNUM
v1	b5
v2	b3
v3	b2
v4	b4
v5	b1

5 rows selected.

- Statement

- 12
- SELECT * FROM Made_visits
-

PATNUMBER	VSTNUMBER
p1	v1

p2	v2
p3	v3
p4	v4
p5	v5

5 rows selected.

- Statement

- 13
- SELECT * FROM Claims
-

CLAIMID	PATNO	VISITNO	BRANCHNO	CLAIMAMOUNT	CLAIMSTATUS
c1	p1	v1	b5	800	CLOSE
c2	p2	v2	b3	1300	CLOSE
c3	p3	v3	b2	1800	OPEN
c4	p4	v4	b4	4800	CLOSE
c5	p5	v5	b1	400	OPEN

5 rows selected.

- Statement

- 14
- SELECT * FROM Works_At
-

STARTDATE	EMPNO	BRNNO
01-JUN-19	e1	b1

15-JUL-19	e6	b1
13-JUN-19	e2	b2
20-AUG-19	e7	b2
29-AUG-19	e3	b3
12-SEP-19	e8	b3
10-JUN-19	e4	b4
11-JUL-19	e9	b4
01-JUN-19	e5	b5
15-JUN-19	e10	b5

10 rows selected.

- Sravya Select Section End - sa3648

- 15
- SELECT * FROM Treats
-

DOCTORTREATSID	PATIENTTREATEDID
e1	p1
e2	p2
e2	p3
e3	p3
e4	p4
e4	p5
e5	p5

7 rows selected.

- Devi Supraja Select Section Start - dr986

- 16
- SELECT * FROM Supervises
-

NURSESUPERVISESID	PATIENTSUPERVISEDID
e10	p5
e6	p1
e6	p2
e7	p2
e7	p3
e8	p3
e9	p4

7 rows selected.

- Statement

- 17
- SELECT * FROM AssistedBy
-

DOCTORASSISTEDID	NURSEASSISTEDID
e1	e6
e2	e7
e3	e8

e4	e9
e5	e10

5 rows selected.

- Statement

- 18
- SELECT * FROM Allergies_Patient
-

ALLERGYPATID	PATIENTALLERGYID
a1	p1
a2	p2
a2	p3
a3	p3
a4	p4

5 rows selected.

- Statement

- 19
- SELECT * FROM Patient_Phone_Number
-

PATIENTPHONEID	PHONENUMBER
p1	(572) 653-3900
p1	(990) 907-1865
p2	(357) 207-7417

p2	(525) 415-1068
p3	(511) 479-6598
p3	(822) 709-8028
p4	(273) 755-1095
p4	(412) 541-5397
p5	(238) 671-6771
p5	(498) 363-2062

10 rows selected.

- Devi Supraja Select Section End - dr986

- 20
- SELECT * FROM Patient_Med_In_Use
-

PATIENTMEDID	MEDICINE
p1	Acetaminophen
p1	Cephalexin
p2	Adderall
p3	Acetaminophen
p3	Fentanyl
p3	Methadone
p4	Fentanyl
p4	Omeprazole
p5	Acetaminophen

p5	Advil
p5	Claritin
p5	Paracetamol

12 rows selected.

8. DATA QUERIES:

- Himanshu Data Section Start

- 1

```
SELECT Visits_Happen.branchNum as branchNum, SUM(Claims.claimAmount) as amount
FROM Visits_Happen, Claims WHERE
Visits_Happen.visitNum = Claims.visitNo AND
Claims.claimStatus = 'OPEN'
```

- GROUP BY Visits_Happen.branchNum

-

BRANCHNUM	AMOUNT
b2	1800
b1	400

-

2 rows selected.

- Statement

- 2

```
SELECT Visits_Happen.branchNum as branchNum, SUM(Claims.claimAmount) as amount
FROM Visits_Happen, Claims WHERE
Visits_Happen.visitNum = Claims.visitNo AND
Claims.claimStatus = 'CLOSE'
```

- GROUP BY Visits_Happen.branchNum

-

BRANCHNUM	AMOUNT
b4	4800
b5	800
b3	1300

- 3 rows selected.

- Himanshu Data Section End

- 3

```
SELECT Visits_Happen.branchNum as branchNum, Claims.claimId, Claims.claimStatus,
Claims.claimAmount
FROM Visits_Happen, Claims WHERE
```

- Visits_Happen.visitNum = Claims.visitNo

BRANCHNUM	CLAIMID	CLAIMSTATUS	CLAIMAMOUNT
b5	c1	CLOSE	800
b3	c2	CLOSE	1300
b2	c3	OPEN	1800
b4	c4	CLOSE	4800
b1	c5	OPEN	400

- 5 rows selected.

- Aishwarya Data Section Start

- 4

SELECT Patient.patientId, count(*) as NumOfAllergies
FROM Allergies_Patient, Patient WHERE
Allergies_Patient.patientAllergyId = Patient.patientId

- GROUP BY Patient.patientId

PATIENTID	NUMOFALLERGIES
p1	1
p2	1
p4	1
p3	2

4 rows selected.

- 5

SELECT Allergies_Patient.allergyPatId, Allergies.allergyName, count(*) as NumOfPatients
FROM Allergies_Patient, Patient, Allergies WHERE
Allergies_Patient.patientAllergyId = Patient.patientId AND
Allergies_Patient.allergyPatId = Allergies.allergyID

- GROUP BY Allergies_Patient.allergyPatId, Allergies.allergyName

ALLERGYPATID	ALLERGYNAME	NUMOFPATIENTS
a1	Penicillin	1
a2	Antibiotics	2
a3	NSAIDs	1
a4	Anticonvulsants	1

- 4 rows selected.

- Aishwarya Data Section End

- 6

```
SELECT Works_At.brnNo, Doctor.doctorId, Employee.employeeName as DoctorName
FROM Works_At, Doctor, Employee WHERE
Works_At.empNo = Doctor.doctorId AND
```

- Doctor.doctorId = Employee.employeeId

BRNNO	DOCTORID	DOCTORNAME
b1	e1	John Doe
b2	e2	Delena Dodson
b3	e3	Robin Borton
b4	e4	Reginia Isham
b5	e5	Billi Yurick

- 5 rows selected.

- Sravya Data Section Start

- 7

```
SELECT Works_At.brnNo, Nurse.nurseId, Employee.employeeName as NurseName
FROM Works_At, Nurse, Employee WHERE
Works_At.empNo = Nurse.nurseId AND
```

- Nurse.nurseId = Employee.employeeId

BRNNO	NURSEID	NURSENAME
b1	e6	Wilbur Nickels
b2	e7	Tina Rayner

b3	e8	Cecilia Hampshire
b4	e9	Marlon Whitis
b5	e10	Carry Mirsky

- 5 rows selected.

- 8

```
SELECT Patient.patientId, Patient.name, Diagnosis.conditionName
FROM Patient, Diagnosis WHERE
```

- Patient.patientId = Diagnosis.patientNumber

PATIENTID	NAME	CONDITIONNAME
p1	Deepika	Viral Fever
p2	Atharva	Stroke
p3	Aditya	Kidney Stones
p4	Zenobia	COVID
p5	Shriya	Skin Pigmentation

- 5 rows selected.

- Sravya Data Section End

- 9

```
SELECT Visits.visitId, Diagnosis.diagnosisId, Diagnosis.conditionName
FROM Visits
LEFT JOIN Diagnosis
```

- ON Visits.visitId = Diagnosis.visitNumber
-

VISITID	DIAGNOSISID	CONDITIONNAME
v1	dg1	Viral Fever
v2	dg2	Stroke
v3	dg3	Kidney Stones
v4	dg4	COVID
v5	dg5	Skin Pigmentation

-
  5 rows selected.

    - Devi Supraja Data Section Start

- 10

```
SELECT Visits.visitId, Claims.claimId, Claims.claimStatus
FROM Visits
LEFT JOIN Claims
```

- ON Claims.visitNo = Visits.VisitID
-

VISITID	CLAIMID	CLAIMSTATUS
v1	c1	CLOSE
v2	c2	CLOSE
v3	c3	OPEN
v4	c4	CLOSE
v5	c5	OPEN

-
  5 rows selected.

    -

- 11

SELECT Works_At.brnNo, Employee.employeeId, Employee.employeeName as EmployeeName
FROM Works_At, Employee WHERE

- Works_At.empNo = Employee.employeeId

BRNNO	EMPLOYEEID	EMPLOYEENAME
b1	e1	John Doe
b2	e2	Delena Dodson
b3	e3	Robin Borton
b4	e4	Reginia Isham
b5	e5	Billi Yurick
b1	e6	Wilbur Nickels
b2	e7	Tina Rayner
b3	e8	Cecilia Hampshire
b4	e9	Marlon Whitis
b5	e10	Carry Mirsky

- 10 rows selected.

- Devi Supraja Data Section End

- 12

SELECT Visits_Happen.branchNum, count(*) as NumOfPatients
FROM Visits_Happen, Made_Visits WHERE
Visits_Happen.visitNum = Made_Visits.vstNumber

- GROUP BY Visits_Happen.branchNum

BRANCHNUM	NUMOFPATIENTS
b2	1

b4	1
b1	1
b5	1
b3	1

- 5 rows selected.

9. DATA MANIPULATION

(Each member must include at least one example of UPDATE and DELETE to any table in your database in the context of your requirements. For each deletion and update command, you must display the data before and after the command to confirm the correctness of the command. That is, I want you to practice insertion/deletion/update in your projects. Write down the member's name at the end of the DML commands)

- Aishwarya DML Section Start
- 1
- SELECT * FROM PATIENT WHERE name='Deepika' AND patientId='p1'

PATIENTID	NAME	GENDER	BIRTHDATE	HEIGHT	WEIGHT	BLOODGROUP	ADDRESSSTREET	ADDRESSCITY	ADDRESSSTATE	ZIP	EMERCONTACTNAME	EMERCONTACTNUM
p1	Deepika	F	18-AUG-04	162	60	O-	3 Shipley Ave.	West Springfield	MA	49509	Ria	4567890456

- Statement

- 2
- UPDATE Patient SET weight=65 WHERE name='Deepika' AND patientId='p1'
- 1 row(s) updated.

- Statement

- 3
- SELECT * FROM PATIENT WHERE name='Deepika' AND patientId='p1'
-

PATIENTID	NAME	GENDER	BIRTHDATE	HEIGHT	WEIGHT	BLOODGROUP	ADDRESSSTREET	ADDRESSCITY	ADDRESSSTATE	ZIP	EMERCONTACTNAME	EMERCONTACTNUM
p1	Deepika	F	18-AUG-04	162	65	O-	3 Shipley Ave.	West Springfield	MA	49509	Ria	4567890456

- Statement

- 4
- SELECT * FROM Claims WHERE claimId = 'c1' AND patNo = 'p1' AND visitNo = 'v1' AND branchNo = 'b5' AND claimAmount = 800
-

CLAIMID	PATNO	VISITNO	BRANCHNO	CLAIMAMOUNT	CLAIMSTATUS
c1	p1	v1	b5	800	CLOSE

- Statement

- 5

- **UPDATE Claims SET claimStatus='OPEN' WHERE claimId = 'c1' AND patNo = 'p1' AND visitNo = 'v1' AND branchNo = 'b5' AND claimAmount = 800**
- **1 row(s) updated.**

- **Aishwarya DML Section End**

- **6**
- **SELECT * FROM Claims WHERE claimId = 'c1' AND patNo = 'p1' AND visitNo = 'v1' AND branchNo = 'b5' AND claimAmount = 800**
-

CLAIMID	PATNO	VISITNO	BRANCHNO	CLAIMAMOUNT	CLAIMSTATUS
c1	p1	v1	b5	800	OPEN

- **Himanshu DML Section Start**

- **7**
- **SELECT * FROM Supervises**
-

NURSESUPERVISESID	PATIENTSUPERVISEDID
e10	p5
e6	p1
e6	p2
e7	p2
e7	p3
e8	p3
e9	p4

7 rows selected.

- **Statement**

- 8
- UPDATE Supervises SET nurseSuperVisesId = 'e10' WHERE nurseSuperVisesId = 'e6' AND patientSupervisedId = 'p1'
- 1 row(s) updated.

- Statement

- 9
- SELECT * FROM Supervises
-

NURSESUPERVISESID	PATIENTSUPERVISEDID
e10	p1
e10	p5
e6	p2
e7	p2
e7	p3
e8	p3
e9	p4

7 rows selected.

- Statement

- 10
- SELECT * FROM assistedBy
-

DOCTORASSISTEDID	NURSEASSISTEDID
e1	e6
e2	e7

e3	e8
e4	e9
e5	e10

5 rows selected.

- Statement

- 11
- UPDATE assistedBy SET doctorAssistedId = 'e2' WHERE doctorAssistedId = 'e1' AND nurseAssistedId = 'e6'
- 1 row(s) updated.

- Himanshu DML Section End

- 12
- SELECT * FROM assistedBy
-

DOCTORASSISTEDID	NURSEASSISTEDID
e2	e6
e2	e7
e3	e8
e4	e9
e5	e10

5 rows selected.

- Sravya DML Section Start

- 13
- SELECT * FROM Patient_Phone_Number

-

PATIENTPHONEID	PHONENUMBER
p1	(572) 653-3900
p1	(990) 907-1865
p2	(357) 207-7417
p2	(525) 415-1068
p3	(511) 479-6598
p3	(822) 709-8028
p4	(273) 755-1095
p4	(412) 541-5397
p5	(238) 671-6771
p5	(498) 363-2062

10 rows selected.

- Statement

- 14
- DELETE FROM Patient_Phone_Number WHERE patientPhoneId = 'p1' AND phoneNumber = '(990) 907-1865'
- 1 row(s) deleted.

- Statement

- 15
- SELECT * FROM Patient_Phone_Number
-

PATIENTPHONEID	PHONENUMBER

p1	**(572) 653-3900**
p2	**(357) 207-7417**
p2	**(525) 415-1068**
p3	**(511) 479-6598**
p3	**(822) 709-8028**
p4	**(273) 755-1095**
p4	**(412) 541-5397**
p5	**(238) 671-6771**
p5	**(498) 363-2062**

- **9 rows selected.**

- **Statement**

- **16**
- **SELECT * FROM Patient_Med_In_Use**

PATIENTMEDID	MEDICINE
p1	**Acetaminophen**
p1	**Cephalexin**
p2	**Adderall**
p3	**Acetaminophen**
p3	**Fentanyl**
p3	**Methadone**
p4	**Fentanyl**

p4	Omeprazole
p5	Acetaminophen
p5	Advil
p5	Claritin
p5	Paracetamol

- 12 rows selected.

- Statement

- 17
- DELETE FROM Patient_Med_In_Use WHERE patientMedId = 'p1' AND medicine = 'Acetaminophen'
- 1 row(s) deleted.

- Sravya DML Section End

- 18
- SELECT * FROM Patient_Med_In_Use

PATIENTMEDID	MEDICINE
p1	Cephalexin
p2	Adderall
p3	Acetaminophen
p3	Fentanyl
p3	Methadone
p4	Fentanyl

p4	Omeprazole
p5	Acetaminophen
p5	Advil
p5	Claritin
p5	Paracetamol

- 11 rows selected.

- Devi Supraja DML Section Start

- 19
- SELECT * FROM Allergies_Patient

ALLERGYPATID	PATIENTALLERGYID
a1	p1
a2	p2
a2	p3
a3	p3
a4	p4

- 5 rows selected.

- Statement

- 20
- DELETE FROM Allergies_Patient WHERE allergyPatId = 'a1' AND patientAllergyId = 'p1'
- 1 row(s) deleted.

- Statement

- 21
- SELECT * FROM Allergies_Patient

-

ALLERGYPATID	PATIENTALLERGYID
a2	p2
a2	p3
a3	p3
a4	p4

- 4 rows selected.

- Statement

- 22
- SELECT * FROM assistedBy
-

DOCTORASSISTEDID	NURSEASSISTEDID
e2	e6
e2	e7
e3	e8
e4	e9
e5	e10

- 5 rows selected.

- Statement

- 23
- DELETE FROM assistedBy WHERE doctorAssistedId = 'e2' AND nurseAssistedId = 'e7'
- 1 row(s) deleted.

- Devi Supraja DML Section End

- **24**
- **SELECT * FROM assistedBy**
-

DOCTORASSISTEDID	NURSEASSISTEDID
e2	e6
e3	e8
e4	e9
e5	e10

- **4 rows selected.**

10. SUMMARY

The project is about handling/retrieving patient medical history by the organization iCare, which is a healthcare organization taking care of COVID patients, for identifying people with health conditions like allergies or genetic disorders etc. to reduce the risk while treatment. Having no centralized database has caused problems like duplicacy in data resulting from multiple entries relating to every patient, missing information because they could be left null, no consistent sharing of the data among various locations of iCare, etc.

The scope of our project would be to create a centralized database with consistency achieved amongst data at various locations which would help reduce the duplicate data and the patient is free to choose a location of his/her convenience. Also, this would also reduce the chances of having missing data by making such kinds of fields as 'not null'. Overall, we could say that the goal is to properly retrieve consistent medical history from various visits at the healthcare organization which helps speed up the treatment process. We would concentrate on including information about the patient, his/her previous medical history including allergies, doctors and nurses involved in the treatment, branches of the visits, diagnosis from previous visits if/any, claims data, etc.

The project was implemented in various parts. Firstly, the goal was to create an ER diagram in order to include as much information as it could and convert that into a relational schema which helps understand the relationships between various tables in order to retrieve/ insert information. Converting the relational schema into 3NF plays an important role in reducing data redundancy in the database and medical data being vast, reducing duplicacy can save out on a lot of space which could in turn reduce the storage issues. Thus, to attain all the goals related to the organization we created a 3NF of the database which is our final product from which information gets inserted/retrieved.

Through implementing the 3NF, we have achieved reliability on the data through achieving consistency as it avoids us from referring to outdated versions of data and duplicated data, and helps us understand a clear hierarchy between the components in the database. Normalization of iCare data could help the employees to refer to an always organized data that is stored without having duplicated or outdated versions that helps the organization to eliminate the constant communication among the various branches to update each other with information. The databases tend to get more secured apart from easy maintenance of the data and cost savings from storage space optimization.

If we had more time, we thought of implementing the connection between family members to retrieve a family medical history which could help the doctors diagnose the patient faster. Our plan was to connect the parent's Id to the children's Id which could help information from each other's record to retrieve information.

11. FUTURE WORK

In this paper, we have restricted our scope to Doctors and Nurses in the hospital. In the future, firstly, we intend to include other employees of the hospital. The Supervisors can also be given to access to all the Doctors', Nurses' and Patients records.

Secondly, when the medical sensors are being used to detect symptoms and allergies of the patient, the report and the results can be directly connected to the database management system and the corresponding records can be updated.

Thirdly, the patients can be given access to the system in order to view their own reports and records.

Attached Files Description: For replicating above results, run these files in this order:

Create -> insert -> select -> joins -> update_delete

1) **Create.sql:** contains all the ddl queries for table creation.
2) **Insert.sql:** contains all the insert queries for data insertion in the table.
3) **Select.sql:** contains all the select data statements for displaying data.
4) **Joins.sql:** contains all the joins statements as data queries.
5) **Update_delete.sql:** contains all the update/delete queries.
6) **ERD.png:** contains the ERD for this project.
7) **ERD.drawio:** contains the original .drawio version of the ERD.