

Project Implementation Report

Comprehensive Medical History Database Management System

IST 659 M005 Spring 2020

Nianyou Zhang & Yawen Zheng

Table of Contents

Project Summary **3**

Relational Data Model**4**

Data Dictionary**5**

Business Rules**13**

Database System Infrastructure**13**

SQL Codes**13**

Major Data Questions**25**

Interface Implementation-Forms and Reports**29**

1. Summary

Dr. David Larsen is a professor at the department of public health, David B. Falk College of Sport and Human Dynamics, Syracuse University. He has been working closely with SUNY Upstate Medical University, studying vector-borne diseases, and analyzing data from low-income countries. His studies focus on predicting health trends for individuals and population.

His major obstacle in analyzing healthcare data from low-income countries is inconsistency and discontinuity of the data (Larsen). Even today, a lot of those regions do not have modern databases to begin with. Handwritten tables and excel sheets are still the go-to methods. Data gathered with different methods are not connected and can be inconsistent. Surveys and medical records from different hospitals are separated from each other. Essential data are not linked, such as necessary demographic and geographic information, together with medical histories. Furthermore, despite the outside help from other countries, governments are slow in developing database systems to keep everyone in check. It is shocking to learn that several villages are still using birth/death records instead of a citizen ID system (Larsen). Immunization records, diagnosis histories and family disease records are only luxury, even though the need for healthcare tracker is imminent.

Our project aims to create a concise, easy-to-implement database management system to solve the above problems with following features:

a. Individual central. This will avoid the inconsistency from different hospital records, by tying the records to an individual's citizen ID. This will also create linkage among immunization, diagnosed diseases, symptoms, and treatments, in all the hospitals, to provide access to large set of data for better analysis.

b. The implementation of time frame. This is effective in keeping a consistent, continuous record of an individual. It will spot expired immunization and help emergency brakes in case of epidemics.

c. Treatments assessment. Treatment results are recorded to provide information whether the diseases were cured. This tells the effectiveness of certain treatments and informs the existence of certain antibodies in an individual or proportion of the population.

d. Symptoms-Disease analytics. Symptoms are included for disease diagnosis. As a result, this provide data for disease analytics, assisting doctors with diagnosis.

d. Tracking Cost. All costs are recorded and declared as funded by Federal, State or Private. This will help healthcare assess budgets and individual financial plans.

Currently the database does a good job keeping record of popular known medical histories for public health concerns. Future improvement can be made to tailor such database systems toward specific diseases to achieve disease analytics.

1. Relational Model



1. Data Dictionary

--part 1: basic information

Stores the basic demographic information of individuals.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: demographicInformation | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key | citizenId | INTEGER | NOT NULL |  | Unique identifier for individual citizen |
|  | firstName | VARCHAR (40) | NOT NULL |  | First name of the citizen |
|  | middleName | VARCHAR (40) |  |  | Middle name of the citizen |
|  | lastName | VARCHAR (40) | NOT NULL |  | Last name of the citizen |
|  | gender | VARCHAR (1) | NOT NULL |  | Check input in M(male)/F (female) |
|  | maritalStatus | VARCHAR (30) | NOT NULL |  | Check input in Single/Married/ Divorced/Widowed |
|  | ethnicity | VARCHAR (40) | NOT NULL |  | Check input in White/African American/Native American/Pacific Islander/Asian/Native Hawaiian |
|  | dateOfBirth | DATE | NOT NULL |  | Date of birth of the citizen |

Stores the current address of individuals.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: geographicInformation | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key, Foreign Key | citizenId | INTEGER | NOT NULL | Table demographicInformation (citizenId) | Unique identifier for individual citizen |
| Primary Key | startDate | DATE | NOT NULL |  | Date of the citizen began to live at this address |
|  | cStreetNo | VARCHAR (30) | NOT NULL |  | Street number of the citizen’s current address |
|  | cStreetName | VARCHAR (30) | NOT NULL |  | Street name of the citizen’s current address |
|  | cCity | VARCHAR (30) | NOT NULL |  | City of the citizen’s current address |
|  | cState | VARCHAR (30) | NOT NULL |  | State of the citizen’s current address |
|  | cZipCode | VARCHAR (10) | NOT NULL |  | Zip code of the citizen’s current address |

Stores parents’ information and birth address of individuals. Unary relationship.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: birthInformation | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key, Foreign Key 1 | childCitizenId | INTEGER | NOT NULL | Table demographicInformation (citizenId) | Unique identifier for individual citizen |
| Foreign Key 2 | fatherCitizenId | INTEGER |  | Table demographicInformation (citizenId) | Unique identifier for individual citizen |
| Foreign Key 3 | motherCitizenId | INTEGER |  | Table demographicInformation (citizenId) | Unique identifier for individual citizen |
|  | bStreetNo | VARCHAR (30) | NOT NULL |  | Street number of the citizen’s birth address |
|  | bStreetName | VARCHAR (30) | NOT NULL |  | Street name of the citizen’s birth address |
|  | bCity | VARCHAR (30) | NOT NULL |  | City of the citizen’s birth address |
|  | bState | VARCHAR (30) | NOT NULL |  | State of the citizen’s birth address |
|  | bZipCode | VARCHAR (10) | NOT NULL |  | Zip code of the citizen’s birth address |

--part 2: immunization

Stores vaccine name and description.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: vaccineDictionary | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key | vaccineId | INTEGER | NOT NULL |  | Unique identifier for vaccine |
|  | vaccineName | VARCHAR (100) | NOT NULL |  | Name of the vaccine |
|  | vaccineDescription | TEXT |  |  | Check input in Tetanus/Hepatitis/Human Papillomavirus/Mumps/Influenza/Hib |

Stores immunization clinic name/immunization cost/funding source. Allows many to many relationship between individuals and vaccines they have taken.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: immunizationRecord | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key, Foreign Key 1 | citizenId | INTEGER | NOT NULL | Table demographicInformation (citizenId) | Unique identifier for individual citizen |
| Primary Key, Foreign Key 2 | vaccineId | INTEGER | NOT NULL | Table vaccineDictionary (vaccineId) | Unique identifier for vaccine |
| Primary Key | dateAdministered | DATETIME | NOT NULL |  | Date of administering the immunization record |
|  | iClinicName | VARCHAR (80) | NOT NULL |  | Name of the clinic that the immunization shot took place |
|  | immunizationCost | DECIMAL | NOT NULL |  | Cost of the immunization |
|  | iFoundingSource | VARCHAR (1) | NOT NULL |  | Funding source of the immunization. Check input in F (Federal)/ S (State)/P (Private) |

--part 3: disease

Stores disease name and description.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: diseaseDictionary | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key | diseaseTypeId | INTEGER | NOT NULL |  | Unique identifier for disease |
|  | diseaseName | VARCHAR (100) | NOT NULL |  | Name of the disease |
|  | diseaseDescription | TEXT |  |  | Description of the disease |

Stores symptom name and description.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: symptomDictionary | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key | symptomId | INTEGER | NOT NULL |  | Unique identifier for symptom |
|  | symptomName | VARCHAR (100) | NOT NULL |  | Name of the symptom |
|  | symptomDescription | TEXT |  |  | Description of the symptom |

Links diseaseSymptom and diagnoseDisease; Stores diseaseId.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: disease | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key | diseaseId | INTEGER | NOT NULL |  | Unique identifier for diagnosed disease |
| Foreign Key | diseaseTypeId | INTEGER | NOT NULL | Table diseaseDictionary (diseaseTypeId) | Unique identifier for disease |

Allows many to many relationship between diagnosed disease and symptoms.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: diseaseSymptom | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key, Foreign Key 1 | diseaseId | INTEGER | NOT NULL | Table disease (diseaseId) | Unique identifier for diagnosed disease |
| Primary Key, Foreign Key 2 | symptomId | INTEGER | NOT NULL | Table symptomDictionary (symptomId) | Unique identifier for symptom |

--part 4: diagnose

Stores date/clinic/cost/funding of diagnosed disease.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: diagnosedDisease | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key, Foreign Key 1 | citizenId | INTEGER | NOT NULL | Table demographicInformation (citizenId) | Unique identifier for individual citizen |
| Primary Key, Foreign Key 2 | diseaseId | INTEGER | NOT NULL | Table disease (diseaseId) | Unique identifier for diagnosed disease |
| Primary Key | dateDiagnosed | DATETIME | NOT NULL |  | Date of diagnosis |
|  | dClinicName | VARCHAR (80) | NOT NULL |  | Clinic name of the diagnosis |
|  | diagnoseCost | DECIMAL | NOT NULL |  | Cost of diagnosis |
|  | dFoundingSource | VARCHAR (1) | NOT NULL |  | Funding source of the diagnose. Check input in F(Federal)/S (State)/P (Private) |

--part 5: treatment

Stores medicine name and description.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: medicineDictionary | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key | medicineId | INTEGER | NOT NULL |  | Unique identifier for medicine |
|  | medicineName | VARCHAR (100) | NOT NULL |  | Name of the medicine |
|  | medicineDescription | TEXT |  |  | Description of the medicine |

Stores surgery name and description.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: surgeryDictionary | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key | surgeryId | INTEGER | NOT NULL |  | Unique identifier for surgery |
|  | surgeryName | VARCHAR (100) | NOT NULL |  | Name of the surgery |
|  | surgeryDescription | TEXT |  |  | Description of the surgery |

Allows many to many relationship between treatments and medicines/ treatments and surgeries; Stores treatmentId.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: treatment | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key, | treatmentId | INTEGER | NOT NULL |  | Unique identifier for treatment |
| Foreign Key 1 | medicineId | INTEGER |  | Table medicineDictionary (medicineId) | Unique identifier for medicine |
| Foreign Key 2 | surgeryId | INTEGER |  | Table surgeryDictionary (surgeryId) | Unique identifier for surgery |

Stores treatment hospital/cost/result/funding.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name: diagnoseTreatment | Entity and Attributes | Field type | Nullable | Foreign Key Constraints | Description |
| Primary Key, Foreign Key 1 | citizenId | INTEGER | NOT NULL | Table diagnosedDisease (citizenId) | Unique identifier for individual citizen |
| Primary Key, Foreign Key 1 | diseaseId | INTEGER | NOT NULL | Table diagnosedDisease (diseaseId) | Unique identifier for diagnosed disease |
| Primary Key, Foreign Key 1 | dateDiagnosed | DATETIME | NOT NULL | Table diagnosedDisease (dateDiagnosed) | Date of diagnosis |
| Primary Key, Foreign Key 2 | treatmentId | INTEGER |  | treatment (treatmentId) | Unique identifier for treatment |
|  | tHospitalName | VARCHAR (80) | NOT NULL |  | Hospital name of the treatment |
|  | treatmentCost | DECIMAL | NOT NULL |  | Cost of the treatment |
|  | treatmentResult | TEXT |  |  | Result of the treatment |
|  | tFoundingSource | VARCHAR (1) | NOT NULL |  | Funding source of the diagnose. Check input in F(Federal)/S(State)/P (Private) |

1. Business Rules
2. Individuals must update their current address within 15 days of changing address.
3. Ethnicity can have 6 categories: White, African American, Native American, Pacific Islander, Asian, Native Hawaiian (Sawe).
4. Gender is M (male) or F (female).
5. Marital status can have four types: married, single, divorced, widowed.
6. Our model only captures 6 most important vaccinations: Tetanus, Hepatitis, Human Papillomavirus, Mumps, Influenza, Hib (Larsen).
7. Expired immunization must be taken again within 6 months.
8. Immunization, doctor visit and disease treatment funding sources can have three types: F (Federal), S (State), or P (Private) (CDC).
9. Database System Infrastructure

We used three tools to create our database: MS Visio, SQL, Access.

Visio:

Visio is used to construct the logical model for our database. All entities and relations are clearly shown in the table. Primary keys and Foreign keys are clearly written. Data type and is also added when SQL database is constructed.

SQL:

SQL server is the physical database that stores our tables and data. We wrote code to create all tables, set constraints and insert sample data. Queries were also written to answer our data questions, following which views were created.

Access:

Access is used to create our user interface, linked with our SQL database. Four major forms are created for users to enter data. Five reports are generated to answer our five data questions.

6. SQL Codes

--IST659M005-Project-Zhang-Nianyou-Yawen-Zheng

--create tables

--part 1: basic information

CREATE TABLE demographicInformation

(

citizenId INTEGER NOT NULL,

firstName VARCHAR (40) NOT NULL,

middleName VARCHAR (40),

lastName VARCHAR (40) NOT NULL,

gender VARCHAR (1) NOT NULL,

maritalStatus VARCHAR(30) NOT NULL,

ethnicity VARCHAR (40) NOT NULL,

dateOfBirth DATE NOT NULL,

CONSTRAINT pk\_demographicInformation PRIMARY KEY (citizenId),

CONSTRAINT chk\_gender CHECK (gender='M' OR gender='F'),

CONSTRAINT chk\_maritalStatus CHECK (maritalStatus='Single' OR maritalStatus='Married'OR maritalStatus='Divorced'OR maritalStatus='Widowed'),

CONSTRAINT chk\_ethnicity CHECK (ethnicity='White' OR ethnicity='African American' OR ethnicity='Native American' OR ethnicity='Pacific Islander' OR ethnicity='Asian' OR ethnicity='Native Hawaiian'),

);

CREATE TABLE geographicInformation

(

citizenId INTEGER NOT NULL,

startDate DATE NOT NULL,

cStreetNo VARCHAR (30) NOT NULL,

cStreetName VARCHAR (30) NOT NULL,

cCity VARCHAR (30) NOT NULL,

cState VARCHAR (30) NOT NULL,

cZipCode VARCHAR (10) NOT NULL,

CONSTRAINT pk\_geographicInformation PRIMARY KEY (citizenId, startDate),

CONSTRAINT fk\_geographicInformation FOREIGN KEY (citizenId) REFERENCES demographicInformation (citizenId),

);

CREATE TABLE birthInformation

(

childCitizenId INTEGER NOT NULL,

fatherCitizenId INTEGER,

motherCitizenId INTEGER,

bStreetNo VARCHAR (30) NOT NULL,

bStreetName VARCHAR (30) NOT NULL,

bCity VARCHAR (30) NOT NULL,

bState VARCHAR (30) NOT NULL,

bZipCode VARCHAR (10) NOT NULL,

CONSTRAINT pk\_birthInformation PRIMARY KEY (childCitizenId),

CONSTRAINT fk\_birthInformation1 FOREIGN KEY (childCitizenId) REFERENCES demographicInformation (citizenId),

CONSTRAINT fk\_birthInformation2 FOREIGN KEY (fatherCitizenId) REFERENCES demographicInformation (citizenId),

CONSTRAINT fk\_birthInformation3 FOREIGN KEY (motherCitizenId) REFERENCES demographicInformation (citizenId),

);

--part 2: immunization

CREATE TABLE vaccineDictionary

(

vaccineId INTEGER NOT NULL,

vaccineName VARCHAR(100) NOT NULL,

vaccineDescription TEXT,

CONSTRAINT pk\_vaccineDictionary PRIMARY KEY (vaccineId),

CONSTRAINT chk\_vaccineName CHECK (vaccineName='Tetanus' OR vaccineName='Hepatitis' OR vaccineName='Human Papillomavirus' OR vaccineName='Mumps' OR vaccineName='Influenza' OR vaccineName='Hib')

);

CREATE TABLE immunizationRecord

(

citizenId INTEGER NOT NULL,

vaccineId INTEGER NOT NULL,

dateAdministered DATETIME DEFAULT GETDATE() NOT NULL,

iClinicName VARCHAR(80) NOT NULL,

immunizationCost DECIMAL NOT NULL,

iFoundingSource VARCHAR(1) NOT NULL,

CONSTRAINT pk\_immunizationRecord PRIMARY KEY (citizenId, vaccineId, dateAdministered),

CONSTRAINT fk\_immunizationRecord1 FOREIGN KEY (citizenId) REFERENCES demographicInformation (citizenId),

CONSTRAINT fk\_immunizationRecord2 FOREIGN KEY (vaccineId) REFERENCES vaccineDictionary (vaccineId),

CONSTRAINT chk\_iFoundingSource CHECK (iFoundingSource='F' OR iFoundingSource='S' OR iFoundingSource='P')

);

--part 3: disease

CREATE TABLE diseaseDictionary

(

diseaseTypeId INTEGER NOT NULL,

diseaseName VARCHAR(100) NOT NULL,

diseaseDescription TEXT,

CONSTRAINT pk\_diseaseDictionary PRIMARY KEY (diseaseTypeId)

);

CREATE TABLE symptomDictionary

(

symptomId INTEGER NOT NULL,

symptomName VARCHAR(100) NOT NULL,

symptomDescription TEXT,

CONSTRAINT pk\_symptomDictionary PRIMARY KEY (symptomId)

);

CREATE TABLE disease

(

diseaseId INTEGER NOT NULL,

diseaseTypeId INTEGER NOT NULL,

CONSTRAINT pk\_disease PRIMARY KEY (diseaseId),

CONSTRAINT fk\_disease FOREIGN KEY (diseaseTypeId) REFERENCES diseaseDictionary (diseaseTypeId)

);

CREATE TABLE diseaseSymptom

(

diseaseId INTEGER NOT NULL,

symptomId INTEGER NOT NULL,

CONSTRAINT pk\_diseaseSymptom PRIMARY KEY (diseaseId,symptomId),

CONSTRAINT fk\_diseaseSymptom1 FOREIGN KEY (diseaseId) REFERENCES disease (diseaseId),

CONSTRAINT fk\_diseaseSymptom2 FOREIGN KEY (symptomId) REFERENCES symptomDictionary (symptomId)

);

--part 4: diagnose

CREATE TABLE diagnosedDisease

(

citizenId INTEGER NOT NULL,

diseaseId INTEGER NOT NULL,

dateDiagnosed DATETIME NOT NULL DEFAULT GETDATE(),

dClinicName VARCHAR (80) NOT NULL,

diagnoseCost DECIMAL NOT NULL,

dFoundingSource VARCHAR (1) NOT NULL,

CONSTRAINT pk\_diagnosedDisease PRIMARY KEY (citizenId, diseaseId, dateDiagnosed),

CONSTRAINT fk\_diagnosedDisease1 FOREIGN KEY (citizenId) REFERENCES demographicInformation (citizenId),

CONSTRAINT fk\_diagnosedDisease2 FOREIGN KEY (diseaseId) REFERENCES disease (diseaseId),

CONSTRAINT chk\_DFoundingSource CHECK (dFoundingSource='F' OR dFoundingSource='S' OR dFoundingSource='P'),

);

--part 5: treatment

CREATE TABLE medicineDictionary

(

medicineId INTEGER NOT NULL,

medicineName VARCHAR(100) NOT NULL,

medicineDescription TEXT,

CONSTRAINT pk\_medicineDictionary PRIMARY KEY (medicineId)

);

CREATE TABLE surgeryDictionary

(

surgeryId INTEGER NOT NULL,

surgeryName VARCHAR(100) NOT NULL,

surgeryDescription TEXT,

CONSTRAINT pk\_surgeryDictionary PRIMARY KEY (surgeryId)

);

CREATE TABLE treatment

(

treatmentId INTEGER NOT NULL,

medicineId INTEGER,

surgeryId INTEGER,

CONSTRAINT pk\_treatment PRIMARY KEY (treatmentId),

CONSTRAINT fk\_treatment1 FOREIGN KEY (medicineId) REFERENCES medicineDictionary (medicineId),

CONSTRAINT fk\_treatment2 FOREIGN KEY (surgeryId) REFERENCES surgeryDictionary (surgeryId)

);

CREATE TABLE diagnoseTreatment

(

citizenId INTEGER NOT NULL,

diseaseId INTEGER NOT NULL,

dateDiagnosed DATETIME NOT NULL,

treatmentId INTEGER NOT NULL,

tHospitalName VARCHAR(80) NOT NULL,

treatmentCost DECIMAL NOT NULL,

treatmentResult TEXT,

tFoundingSource VARCHAR(1) NOT NULL,

CONSTRAINT pk\_diagnoseTreatment PRIMARY KEY (citizenId, diseaseId, dateDiagnosed, treatmentId),

CONSTRAINT fk\_diagnoseTreatment1 FOREIGN KEY (citizenId, diseaseId, dateDiagnosed) REFERENCES diagnosedDisease (citizenId, diseaseId, dateDiagnosed),

CONSTRAINT fk\_diagnoseTreatment2 FOREIGN KEY (treatmentId) REFERENCES treatment (treatmentId),

CONSTRAINT chk\_tFoundingSource CHECK (tFoundingSource='F' OR tFoundingSource='S' OR tFoundingSource='P'),

);

--insert values

--part 1: basic information

INSERT INTO demographicInformation (citizenId, firstName, middleName, lastName, gender, maritalStatus, ethnicity, dateOfBirth)

VALUES (1, 'Mack', 'Meng', 'Wang', 'M', 'Married', 'Asian', '1990-05-15'),

(2, 'Hiba', 'Louise', 'Blackburn', 'F', 'Widowed', 'Pacific Islander', '1993-06-16'),

(3, 'Marta', 'Rose', 'Jennings', 'F', 'Married', 'White', '1986-11-09'),

(4, 'Yosef', 'James', 'Bender', 'M', 'Divorced', 'White', '1966-08-02'),

(5, 'Ernest', 'William', 'Aguilar', 'M', 'Single', 'African American', '1989-04-22'),

(6, 'Lily', 'Mae', 'Sweeney', 'F', 'Single', 'Native American', '1977-02-27');

INSERT INTO demographicInformation (citizenId, firstName, lastName, gender, maritalStatus, ethnicity, dateOfBirth)

VALUES (7, 'Joe', 'Johnson', 'M', 'Married', 'White', '1963-08-01'),

(8, 'Ibrar', 'Christian', 'M', 'Divorced', 'Native Hawaiian', '1999-09-05'),

(9, 'Zunaira', 'Hudson', 'F', 'Widowed', 'Asian', '2001-12-17');

INSERT INTO geographicInformation (citizenId, startDate, cStreetNo, cStreetName, cCity, cState, cZipCode)

VALUES (1, '1990-05-15', '410', 'Comstock Ave', 'Syracuse', 'New York', '13210'),

(2, '1993-06-16', '112', 'Lafayette Rd', 'Syracuse', 'New York', '13205'),

(3, '1986-11-09', '17', 'James St', 'Syracuse', 'New York', '13210'),

(4, '1966-08-02', '4301', 'Nottingham Rd', 'Syracuse', 'New York', '13244'),

(5, '1989-04-22', '3', 'Ostrom Ave', 'Syracuse', 'New York', '13225'),

(6, '1977-02-27', '137', 'Sumner Ave', 'Syracuse', 'New York', '13210'),

(7, '1963-08-01', '4248', 'Nottingham Rd', 'Syracuse', 'New York', '13244'),

(8, '1999-09-05', '7116', 'Lafayette Rd', 'Syracuse', 'New York', '13205'),

(9, '2001-12-17', '4237', 'Nottingham Rd', 'Syracuse', 'New York', '13244'),

(2, '2000-08-23', '415', 'Comstock Ave', 'Syracuse', 'New York', '13210'),

(4, '1999-05-17', '433', 'Comstock Ave', 'Syracuse', 'New York', '13210'),

(5, '2013-01-13', '23', 'James St', 'Syracuse', 'New York', '13210'),

(8, '2018-03-09', '44', 'Ostrom Ave', 'Syracuse', 'New York', '13225');

INSERT INTO birthInformation (childCitizenId, bStreetNo, bStreetName, bCity, bState, bZipCode)

VALUES (1, '410', 'Comstock Ave', 'Syracuse', 'New York', '13210'),

(2, '112', 'Lafayette Rd', 'Syracuse', 'New York', '13205'),

(3, '17', 'James St', 'Syracuse', 'New York', '13210'),

(4, '4301', 'Nottingham Rd', 'Syracuse', 'New York', '13244'),

(5, '3', 'Ostrom Ave', 'Syracuse', 'New York', '13225'),

(6, '137', 'Sumner Ave', 'Syracuse', 'New York', '13210'),

(7, '4248', 'Nottingham Rd', 'Syracuse', 'New York', '13244'),

(8, '7116', 'Lafayette Rd', 'Syracuse', 'New York', '13205'),

(9, '4237', 'Nottingham Rd', 'Syracuse', 'New York', '13244');

--part 2: immunization

INSERT INTO vaccineDictionary (vaccineId, vaccineName, vaccineDescription)

VALUES (101, 'Tetanus', 'A serious bacterial infection that causes painful muscle spasms and can lead to death'),

(102, 'Hepatitis', 'An inflammation of the liver'),

(103, 'Human Papillomavirus', 'An infection that causes warts in various parts of the body, depending on the strain'),

(104, 'Influenza', 'A common viral infection, flu attack of the lungs, nose, and throat'),

(105, 'Hib', 'Haemophilus influenzae type b'),

(106, 'Mumps', 'A viral infection that affects the salivary glands');

INSERT INTO immunizationRecord (citizenId, vaccineId, dateAdministered, iClinicName, immunizationCost, iFoundingSource)

VALUES (1, 101, '1990-05-15', 'SyrImmue', 50, 'F'),

(1, 102, '1990-05-15', 'SyrImmue', 150, 'F'),

(1, 103, '1990-05-15', 'SyrImmue', 75, 'S'),

(1, 104, '1990-05-15', 'SyrImmue', 50, 'P'),

(1, 105, '1990-05-15', 'SyrImmue', 100, 'S'),

(1, 106, '1990-05-15', 'SyrImmue', 20, 'P'),

(2, 101, '1993-06-16', 'SyrImmue', 50, 'F'),

(2, 102, '1993-06-16', 'SyrImmue', 125, 'F'),

(2, 105, '1993-06-16', 'SyrImmue', 120, 'F'),

(2, 106, '1993-06-16', 'SyrImmue', 25, 'P'),

(3, 101, '1986-11-09', 'USImmue', 35, 'F'),

(3, 102, '1986-11-09', 'USImmue', 45, 'F'),

(3, 103, '1986-11-09', 'USImmue', 60, 'F'),

(3, 105, '1986-11-09', 'USImmue', 20, 'F'),

(3, 106, '1990-11-09', 'USImmue', 25, 'F'),

(4, 101, '1966-08-02', 'USImmue', 50, 'S'),

(4, 102, '2000-08-02', 'USImmue', 50, 'S'),

(4, 103, '1966-08-02', 'USImmue', 50, 'S'),

(4, 104, '1966-08-02', 'USImmue', 50, 'S'),

(4, 105, '2000-08-02', 'USImmue', 50, 'S'),

(4, 106, '1966-08-02', 'USImmue', 50, 'S'),

(5, 101, '1989-04-22', 'USImmue', 100, 'P'),

(5, 102, '1989-04-22', 'USImmue', 90, 'P'),

(5, 103, '1989-04-22', 'USImmue', 50, 'P'),

(5, 105, '1989-04-22', 'USImmue', 80, 'P'),

(5, 106, '1989-04-22', 'USImmue', 80, 'P'),

(6, 101, '1977-02-27', 'SyrImmue', 50, 'F'),

(6, 102, '2000-02-27', 'SyrImmue', 50, 'P'),

(6, 103, '1977-02-27', 'SyrImmue', 150, 'S'),

(6, 105, '1977-02-27', 'SyrImmue', 150, 'S'),

(6, 106, '1977-02-27', 'SyrImmue', 50, 'F'),

(7, 101, '1963-08-01', 'SyrImmue', 250, 'F'),

(7, 102, '1963-08-01', 'SyrImmue', 50, 'P'),

(7, 103, '1963-08-01', 'SyrImmue', 40, 'P'),

(7, 105, '1963-08-01', 'SyrImmue', 150, 'P'),

(7, 106, '2000-08-01', 'SyrImmue', 30, 'F'),

(8, 101, '1999-09-05', 'USImmue', 350, 'F'),

(8, 102, '1999-09-05', 'USImmue', 150, 'P'),

(8, 103, '1999-09-05', 'USImmue', 50, 'P'),

(8, 104, '1999-09-05', 'USImmue', 50, 'P'),

(8, 105, '1999-09-05', 'USImmue', 150, 'F'),

(8, 106, '1999-09-05', 'USImmue', 50, 'S'),

(9, 101, '2001-12-17', 'USImmue', 50, 'S'),

(9, 102, '2001-12-17', 'USImmue', 120, 'S'),

(9, 104, '2001-12-17', 'USImmue', 80, 'F'),

(9, 105, '2001-12-17', 'USImmue', 90, 'P'),

(9, 106, '2001-12-17', 'USImmue', 20, 'P');

--part 3: disease

INSERT INTO diseaseDictionary (diseaseTypeId, diseaseName, diseaseDescription)

VALUES (1001,'atherosclerotic disease','a hardening and narrowing of arteries'),

(1002,'heart arrhythmias','improper beating of the heart, whether irregular, too fast, or too slow'),

(1003,'dilated cardiomyopathy','a disease of the heart muscle, usually starting in heart main pumping chamber (left ventricle)'),

(1004,'valvular heart disease','Valvular heart disease is characterized by damage to or a defect in one of the four heart valves'),

(1005,'heart defects','an abnormality in the heart that develops before birth'),

(1006,'meningitis','an inflammation of the lining around the brain or spinal cord'),

(1007,'hydrocephalus','an abnormally increased amount of cerebrospinal (brain) fluid inside the skull'),

(1008,'pseudotumor cerebri','increased pressure inside the skull with no apparent cause'),

(1009,'liver cancer','a type of cancer that starts in the liver');

INSERT INTO symptomDictionary (symptomId, symptomName, symptomDescription)

VALUES (2001,'chest pain',''),

(2002,'shortness of breath',''),

(2003,'fluttering in chest',''),

(2004,'bradycardia','Slow heartbeat'),

(2005,'leg muscle infection', 'swelling of the legs, ankles and feet'),

(2006,'cyanosis','pale gray or blue skin color '),

(2007,'headache',''),

(2008,'jaundice','Yellow discoloration of your skin and the whites of your eyes'),

(2009,'upper abdominal pain','');

INSERT INTO disease (diseaseId, diseaseTypeId)

VALUES (6001,1001),

(6002,1001),

(6003,1006),

(6004,1002),

(6005,1003),

(6006,1004),

(6007,1005),

(6008,1008),

(6009,1009);

INSERT INTO diseaseSymptom (diseaseId, symptomId)

VALUES (6001,2001),

(6002,2002),

(6003,2003),

(6004,2004),

(6005,2005),

(6006,2002),

(6007,2006),

(6008,2007),

(6009,2008);

--part 4: diagnose

INSERT INTO diagnosedDisease (citizenId, diseaseId, dClinicName, diagnoseCost, dFoundingSource)

VALUES (1, 6003, 'SyrImmue', 200, 'S'),

(1, 6004, 'SyrImmue', 80, 'F'),

(1, 6005, 'SyrImmue', 150, 'S'),

(1, 6007, 'USImmue', 20, 'P'),

(2,6004,'SyrImmue',55,'P'),

(2,6006,'SyrImmue',40,'S'),

(2,6001,'SyrImmue',15,'P'),

(3,6009,'USImmue',150,'P'),

(4,6002,'SyrImmue',60,'S'),

(4,6001,'SyrImmue',60,'S'),

(4,6008,'USImmue',60,'F'),

(4,6007,'USImmue',60,'S'),

(6,6003,'SyrImmue',710,'F'),

(7,6008,'USImmue',230,'F'),

(7,6004,'USImmue',130,'P'),

(9, 6007, 'USImmue', 225, 'P'),

(9, 6005, 'SyrImmue', 525, 'P');

INSERT INTO diagnosedDisease (citizenId, diseaseId, dateDiagnosed, dClinicName, diagnoseCost, dFoundingSource)

VALUES (1, 6001, '2018-09-07 21:37:48.111', 'SyrImmue', 180, 'P'),

(2,6002,'2017-05-05 10:56:56.876','SyrImmue',55,'P'),

(3,6003,'1998-12-15 16:55:33.345','USImmue',170,'P'),

(4,6004,'2019-08-13 21:51:12.333','SyrImmue',160,'S'),

(5,6005,'2012-10-16 22:24:25.776','USImmue',150,'P'),

(6,6006,'2020-06-22 08:33:21.823','SyrImmue',70,'F'),

(7,6007,'2017-09-19 07:07:07.886','USImmue',220,'F'),

(8,6008,'2014-07-31 18:45:47.234','SyrImmue',94,'P'),

(9,6009,'2008-04-27 14:14:14.624','USImmue',111,'S');

--part 5: treatment

INSERT INTO medicineDictionary (medicineId, medicineName, medicineDescription)

VALUES (3001,'atorvastatin ','statins'),

(3002,'lovastatin','statins'),

(3003,'amiodarone','This medication is used to treat certain types of serious (possibly fatal) irregular heartbeat (such as persistent ventricular fibrillation/tachycardia). It is used to restore normal heart rhythm and maintain a regular, steady heartbeat.'),

(3004,'flecainide','It is used to restore normal heart rhythm and maintain a regular, steady heartbeat. It is also used to prevent certain types of irregular heartbeat from returning (such as atrial fibrillation).'),

(3005,'digoxin','Digoxin helps make the heart beat stronger and with a more regular rhythm.'),

(3006,'warfarin','a prescription medication used to prevent harmful blood clots from forming or growing larger'),

(3007,'acetazolamide','a glaucoma drug'),

(3008,'cabozantinib','a medication used to treat medullary thyroid cancer and a second line treatment for renal cell carcinoma among others'),

(3009,'prednisone','a corticosteroid. It prevents the release of substances in the body that cause inflammation. It also suppresses the immune system');

INSERT INTO surgeryDictionary (surgeryId, surgeryName, surgeryDescription)

VALUES (4001,'appendectomy','an appendectomy is removing the appendix'),

(4002,'breast biopsy','a test used to help diagnose cancer. The surgeon removes a small sample of tissue or cells'),

(4003,'cataract surgery','Cataracts cloud the normally clear lens of the eyes.'),

(4004,'carotid endarterectomy','a surgery to remove blockage from carotid arteries'),

(4005,'coronary artery bypass grafting','a procedure to improve poor blood flow to the heart'),

(4006,'dilation and curettage','a minor surgery where the cervix is expanded (dilated)'),

(4007,'partial colectomy','A partial colectomy is removing part of the large intestine (colon).'),

(4008,'prostatectom','A prostatectomy is removing all or part of the prostate gland.'),

(4009,'tonsillectomy','A tonsillectomy is removing of one or both tonsils.');

INSERT INTO treatment (treatmentId, medicineId, surgeryId)

VALUES (5001,3001,4004),

(5002,3002,4005);

INSERT INTO treatment (treatmentId, medicineId)

VALUES (5003,3003),

(5004,3004),

(5005,3005),

(5006,3006),

(5007,3005),

(5008,3007),

(5009,3008);

INSERT INTO diagnoseTreatment (citizenId, diseaseId, dateDiagnosed, treatmentId, tHospitalName, treatmentCost, treatmentResult, tFoundingSource)

VALUES (1,6001,'2018-09-07 21:37:48.111','5001','Aultman Hospital',120000,'','P'),

(2,6002,'2017-05-05 10:56:56.876','5002','Upstate Health Care Center',200000,'','P'),

(3,6003,'1998-12-15 16:55:33.345','5003','Aultman Hospital',178,'','P'),

(4,6004,'2019-08-13 21:51:12.333','5004','Upstate Health Care Center',166,'','S'),

(5,6005,'2012-10-16 22:24:25.776','5005','Crouse Hospital',150,'','P'),

(6,6006,'2020-06-22 08:33:21.823','5006','Upstate Health Care Center',150,'','F'),

(7,6007,'2017-09-19 07:07:07.886','5007','Upstate Health Care Center',333,'','F'),

(8,6008,'2014-07-31 18:45:47.234','5008','Crouse Hospital',94,'','P'),

(9,6009,'2008-04-27 14:14:14.624','5009','Aultman Hospital',111,'','S');

--select tables

--part 1: basic information

SELECT \* FROM demographicInformation

SELECT \* FROM geographicInformation

SELECT \* FROM birthInformation

--part 2: immunization

SELECT \* FROM vaccineDictionary

SELECT \* FROM immunizationRecord

--part 3: disease

SELECT \* FROM diseaseDictionary

SELECT \* FROM symptomDictionary

SELECT \* FROM disease

SELECT \* FROM diseaseSymptom

--part 4: diagnose

SELECT \* FROM diagnosedDisease

--part 5: treatment

SELECT \* FROM medicineDictionary

SELECT \* FROM surgeryDictionary

SELECT \* FROM treatment

SELECT \* FROM diagnoseTreatment

--query questions:

--1. Query individual vaccination history (example Mack Wang's immunization history).

SELECT d.citizenId, d.firstName, d.lastName, v.vaccineName, i.dateAdministered, i.immunizationCost, i.iClinicName

FROM demographicInformation d

JOIN immunizationRecord i

ON d.citizenId=i.citizenId

JOIN vaccineDictionary v

ON i.vaccineId=v.vaccineId

WHERE d.firstName='Mack' AND d.lastName='Wang'

CREATE VIEW vaccination\_history\_Mack\_Wang AS (

SELECT d.citizenId, d.firstName, d.lastName, v.vaccineName, i.dateAdministered, i.immunizationCost, i.iClinicName

FROM demographicInformation d

JOIN immunizationRecord i

ON d.citizenId=i.citizenId

JOIN vaccineDictionary v

ON i.vaccineId=v.vaccineId

WHERE d.firstName='Mack' AND d.lastName='Wang'

);

--2. What part of the population has effective immunization against Tetanus and Influenza, and when were they vaccinated?

SELECT d.citizenId, d.firstName, d.lastName, v.vaccineName, i.dateAdministered

FROM immunizationRecord i

JOIN demographicInformation d

ON i.citizenId=d.citizenId

JOIN vaccineDictionary v

ON i.vaccineId=v.vaccineId

WHERE i.vaccineId=101 OR i.vaccineId=104

ORDER BY i.vaccineId

CREATE VIEW vaccination\_history\_tetanus\_influenza AS (

SELECT d.citizenId, d.firstName, d.lastName, v.vaccineName, i.dateAdministered

FROM immunizationRecord i

JOIN demographicInformation d

ON i.citizenId=d.citizenId

JOIN vaccineDictionary v

ON i.vaccineId=v.vaccineId

WHERE i.vaccineId=101 OR i.vaccineId=104

);

--3. What are common symptoms for atherosclerotic disease?

SELECT dd.dClinicName ,dd.dateDiagnosed, ddc.diseaseName, sdc.symptomName, sdc.symptomDescription, ddc.diseaseDescription

FROM diagnosedDisease dd

JOIN disease d

ON dd.diseaseId=d.diseaseId

JOIN diseaseSymptom ds

ON d.diseaseId=ds.diseaseId

JOIN diseaseDictionary ddc

ON d.diseaseTypeId=ddc.diseaseTypeId

JOIN symptomDictionary sdc

ON ds.symptomId=sdc.symptomId

WHERE ddc.diseaseName='atherosclerotic disease'

-- important note, this view protects patient information

CREATE VIEW symptom\_atherosclerotic\_disease AS (

SELECT dd.dClinicName ,dd.dateDiagnosed, ddc.diseaseName, sdc.symptomName, sdc.symptomDescription, ddc.diseaseDescription

FROM diagnosedDisease dd

JOIN disease d

ON dd.diseaseId=d.diseaseId

JOIN diseaseSymptom ds

ON d.diseaseId=ds.diseaseId

JOIN diseaseDictionary ddc

ON d.diseaseTypeId=ddc.diseaseTypeId

JOIN symptomDictionary sdc

ON ds.symptomId=sdc.symptomId

WHERE ddc.diseaseName='atherosclerotic disease'

);

--4. What diseases are appearing on Nottingham Rd?

SELECT dd.dClinicName, dd.dateDiagnosed, ddc.diseaseName, ddc.diseaseDescription

FROM demographicInformation d

JOIN geographicInformation g

ON d.citizenId=g.citizenId

JOIN diagnosedDisease dd

ON d.citizenId=dd.citizenId

JOIN disease di

ON dd.diseaseId=di.diseaseId

JOIN diseaseDictionary ddc

ON di.diseaseTypeId=ddc.diseaseTypeId

WHERE g.cStreetName='Nottingham Rd'

CREATE VIEW diagnoses\_Nottingham\_Rd AS (

SELECT dd.dClinicName, dd.dateDiagnosed, ddc.diseaseName, ddc.diseaseDescription

FROM demographicInformation d

JOIN geographicInformation g

ON d.citizenId=g.citizenId

JOIN diagnosedDisease dd

ON d.citizenId=dd.citizenId

JOIN disease di

ON dd.diseaseId=di.diseaseId

JOIN diseaseDictionary ddc

ON di.diseaseTypeId=ddc.diseaseTypeId

WHERE g.cStreetName='Nottingham Rd'

);

-- 5. Where did federal government funding go in diagnose and treatment?

SELECT d.citizenId, dd.diagnoseCost, dt.treatmentCost, dd.dateDiagnosed

FROM demographicInformation d

FULL OUTER JOIN diagnosedDisease dd

ON d.citizenId=dd.citizenId

FULL OUTER JOIN diagnoseTreatment dt

ON dd.citizenId=dt.citizenId AND dd.diseaseId=dt.diseaseId AND dd.dateDiagnosed=dt.dateDiagnosed

WHERE dFoundingSource='F' OR tFoundingSource='F'

CREATE VIEW federal\_funding\_diagnose\_treatment AS (

SELECT d.citizenId, dd.diagnoseCost, dt.treatmentCost, dd.dateDiagnosed

FROM demographicInformation d

FULL OUTER JOIN diagnosedDisease dd

ON d.citizenId=dd.citizenId

FULL OUTER JOIN diagnoseTreatment dt

ON dd.citizenId=dt.citizenId AND dd.diseaseId=dt.diseaseId AND dd.dateDiagnosed=dt.dateDiagnosed

WHERE dFoundingSource='F' OR tFoundingSource='F'

);

--select views

SELECT \* FROM vaccination\_history\_Mack\_Wang

SELECT \* FROM vaccination\_history\_tetanus\_influenza

SELECT \* FROM symptom\_atherosclerotic\_disease

SELECT \* FROM diagnoses\_Nottingham\_Rd

SELECT \* FROM federal\_funding\_diagnose\_treatment

--drop views

DROP VIEW vaccination\_history\_Mack\_Wang

DROP VIEW vaccination\_history\_tetanus\_influenza

DROP VIEW symptom\_atherosclerotic\_disease

DROP VIEW diagnoses\_Nottingham\_Rd

DROP VIEW federal\_funding\_diagnose\_treatment

--drop tables

--part 5: treatment

DROP TABLE diagnoseTreatment

DROP TABLE treatment

DROP TABLE surgeryDictionary

DROP TABLE medicineDictionary

--part 4: diagnose

DROP TABLE diagnosedDisease

--part 3: disease

DROP TABLE diseaseSymptom

DROP TABLE disease

DROP TABLE symptomDictionary

DROP TABLE diseaseDictionary

--part 2: immunization

DROP TABLE immunizationRecord

DROP TABLE vaccineDictionary

--part 1: basic information

DROP TABLE birthInformation

DROP TABLE geographicInformation

DROP TABLE demographicInformation

7. Data Questions:

1. Query individual vaccination history (example what is Mack Wang's immunization history?).

This is a very important purpose of our database: allowing individuals to keep track with their immunization history. This question is answered by simply connecting demographic information with immunization record.

--1. Query individual vaccination history (example Mack Wang's immunization history).

SELECT d.citizenId, d.firstName, d.lastName, v.vaccineName, i.dateAdministered, i.immunizationCost, i.iClinicName

FROM demographicInformation d

JOIN immunizationRecord i

ON d.citizenId=i.citizenId

JOIN vaccineDictionary v

ON i.vaccineId=v.vaccineId

WHERE d.firstName='Mack' AND d.lastName='Wang'

A screenshot of a social media post

Description automatically generated

CREATE VIEW vaccination\_history\_Mack\_Wang AS (

SELECT d.citizenId, d.firstName, d.lastName, v.vaccineName, i.dateAdministered, i.immunizationCost, i.iClinicName

FROM demographicInformation d

JOIN immunizationRecord i

ON d.citizenId=i.citizenId

JOIN vaccineDictionary v

ON i.vaccineId=v.vaccineId

WHERE d.firstName='Mack' AND d.lastName='Wang'

);

1. What part of the population has effective immunization against Tetanus and Influenza, and when were they vaccinated?

From healthcare worker perspective, it is important to keep track with the proportion of population that is vaccinated for certain diseases. This will allow foreseeing potential disease outbreak. Then actions such as reinforcing vaccination can be taken. This question is answered by linking demographic information with immunization records.

--2. What part of the population has effective immunization against Tetanus and Influenza, and when were they vaccinated?

SELECT d.citizenId, d.firstName, d.lastName, v.vaccineName, i.dateAdministered

FROM immunizationRecord i

JOIN demographicInformation d

ON i.citizenId=d.citizenId

JOIN vaccineDictionary v

ON i.vaccineId=v.vaccineId

WHERE i.vaccineId=101 OR i.vaccineId=104

ORDER BY i.vaccineId

A screenshot of a social media post

Description automatically generated

CREATE VIEW vaccination\_history\_tetanus\_influenza AS (

SELECT d.citizenId, d.firstName, d.lastName, v.vaccineName, i.dateAdministered

FROM immunizationRecord i

JOIN demographicInformation d

ON i.citizenId=d.citizenId

JOIN vaccineDictionary v

ON i.vaccineId=v.vaccineId

WHERE i.vaccineId=101 OR i.vaccineId=104

);

1. What are common symptoms for atherosclerotic disease?

From the doctor perspective, disease and symptom analytics is always extremely valuable. By linking diagnoses and symptom records, we can answer such questions: what symptoms do a disease cause? With large amount of data, diseases will be more accurately diagnosed.

--3. What are common symptoms for atherosclerotic disease?

SELECT dd.dClinicName ,dd.dateDiagnosed, ddc.diseaseName, sdc.symptomName, sdc.symptomDescription, ddc.diseaseDescription

FROM diagnosedDisease dd

JOIN disease d

ON dd.diseaseId=d.diseaseId

JOIN diseaseSymptom ds

ON d.diseaseId=ds.diseaseId

JOIN diseaseDictionary ddc

ON d.diseaseTypeId=ddc.diseaseTypeId

JOIN symptomDictionary sdc

ON ds.symptomId=sdc.symptomId

WHERE ddc.diseaseName='atherosclerotic disease'

A screenshot of a social media post

Description automatically generated

-- important note, this view protects patient information

CREATE VIEW symptom\_atherosclerotic\_disease AS (

SELECT dd.dClinicName ,dd.dateDiagnosed, ddc.diseaseName, sdc.symptomName, sdc.symptomDescription, ddc.diseaseDescription

FROM diagnosedDisease dd

JOIN disease d

ON dd.diseaseId=d.diseaseId

JOIN diseaseSymptom ds

ON d.diseaseId=ds.diseaseId

JOIN diseaseDictionary ddc

ON d.diseaseTypeId=ddc.diseaseTypeId

JOIN symptomDictionary sdc

ON ds.symptomId=sdc.symptomId

WHERE ddc.diseaseName='atherosclerotic disease'

);

1. What diseases are appearing on Nottingham Rd?

In case of outbreak, it is very important for healthcare workers to know the location of outbreak, and quarantine if necessary. By linking diagnose information and citizen current address, we can learn the popular diseases in a specific location, and take action to stop the spread ahead of time.

--4. What diseases are appearing on Nottingham Rd?

SELECT dd.dClinicName, dd.dateDiagnosed, ddc.diseaseName, ddc.diseaseDescription

FROM demographicInformation d

JOIN geographicInformation g

ON d.citizenId=g.citizenId

JOIN diagnosedDisease dd

ON d.citizenId=dd.citizenId

JOIN disease di

ON dd.diseaseId=di.diseaseId

JOIN diseaseDictionary ddc

ON di.diseaseTypeId=ddc.diseaseTypeId

WHERE g.cStreetName='Nottingham Rd'

A screenshot of a social media post

Description automatically generated

CREATE VIEW diagnoses\_Nottingham\_Rd AS (

SELECT dd.dClinicName, dd.dateDiagnosed, ddc.diseaseName, ddc.diseaseDescription

FROM demographicInformation d

JOIN geographicInformation g

ON d.citizenId=g.citizenId

JOIN diagnosedDisease dd

ON d.citizenId=dd.citizenId

JOIN disease di

ON dd.diseaseId=di.diseaseId

JOIN diseaseDictionary ddc

ON di.diseaseTypeId=ddc.diseaseTypeId

WHERE g.cStreetName='Nottingham Rd'

);

1. Where did federal government funding go in diagnose and treatment?

Funding is a big part of healthcare. By connecting all the diagnose, treatment records, we can calculate the total cost, and discriminate different funding sources. This will allow government to keep track with the medical founding, and plan future budgets.

-- 5. Where did federal government funding go in diagnose and treatment?

SELECT d.citizenId, dd.diagnoseCost, dt.treatmentCost, dd.dateDiagnosed

FROM demographicInformation d

FULL OUTER JOIN diagnosedDisease dd

ON d.citizenId=dd.citizenId

FULL OUTER JOIN diagnoseTreatment dt

ON dd.citizenId=dt.citizenId AND dd.diseaseId=dt.diseaseId AND dd.dateDiagnosed=dt.dateDiagnosed

WHERE dFoundingSource='F' OR tFoundingSource='F'

A screenshot of a social media post

Description automatically generated

CREATE VIEW federal\_funding\_diagnose\_treatment AS (

SELECT d.citizenId, dd.diagnoseCost, dt.treatmentCost, dd.dateDiagnosed

FROM demographicInformation d

FULL OUTER JOIN diagnosedDisease dd

ON d.citizenId=dd.citizenId

FULL OUTER JOIN diagnoseTreatment dt

ON dd.citizenId=dt.citizenId AND dd.diseaseId=dt.diseaseId AND dd.dateDiagnosed=dt.dateDiagnosed

WHERE dFoundingSource='F' OR tFoundingSource='F'

);

8. Interface Implementation-Forms and Reports

Demographic Information Form Design:

A screenshot of a computer

Description automatically generated

Demographic Information Form 1:

A screenshot of a social media post

Description automatically generated

Demographic Information Form 2:

A screenshot of a social media post

Description automatically generated

Demographic Information form allows users to create and delete citizen records. Their birth information will be entered at birth, and their current address will be updated within 15 days of moving. This will allow analysis of health trend from demographic and geographic perspective.

Immunization Form Design:

A picture containing screenshot, white, kitchen

Description automatically generated

Immunization Form:

A screenshot of a social media post

Description automatically generated

This form allows immunization records to be created and deleted. Entering the citizen ID and vaccine ID will automatically generate according patient information and vaccine information.

Diagnose Form Design:

A screenshot of a social media post

Description automatically generated

Diagnose Form:

A screenshot of a computer

Description automatically generated

Diagnose Form allows diagnose records to be created and deleted. The physician will choose or create disease ID, based on the disease type from dictionary, and according symptom type from symptom dictionary. Date diagnosed will be automatically generated. Patient information is shown on the side as quality of life.

Treatment Form Design:

A screenshot of a computer

Description automatically generated

Treatment Form:

A screenshot of a social media post

Description automatically generated

Treatment Form allows creation and deletion of treatment records. Treatment type will be created based on the medicines and surgeries used. Patient information is shown on the side as quality of life.

Data Questions Reports:

1. Query individual vaccination history (example Mack Wang's immunization history).

A screenshot of a social media post

Description automatically generated

1. What part of the population has effective immunization against Tetanus and Influenza, and when were they vaccinated?

A screenshot of a cell phone

Description automatically generated

1. What are common symptoms for atherosclerotic disease?

A screenshot of a social media post

Description automatically generated

1. What diseases are appearing on Nottingham Rd?

A screenshot of a cell phone

Description automatically generated

1. Where did federal government funding go in diagnose and treatment?

A screenshot of a cell phone

Description automatically generated