Kanning Wu

IST659 Project Deliverable 2

Project Deliverable 2

Summary

Begin with a summary of your project. Carefully describe your project and its importance. It is here that you will introduce the reader to your project and set up the business case, high level rules about the data, its stakeholders, and expectations of the final outcome. Edit this summary whenever necessary throughout the project to reflect any changes in scope or content. You should have a brief working version of this summary by Week 2 and this text will initially serve as your project proposal, gradually gaining more detail as you progress through the process.

In Jan. 2020, the n-CoV 2019 outbreak has affected thousands of people’s lives. Massive information of this new type of virus is missing, but misleading news has flooded the internet. In this project, the database can be divided into following parts. 1. The virus information 2. Infected patients 3. Drugs availability 4. Locations 5. Journals. This database can provide information for people to keep tracking the current situation and understand the prevention of being infected.

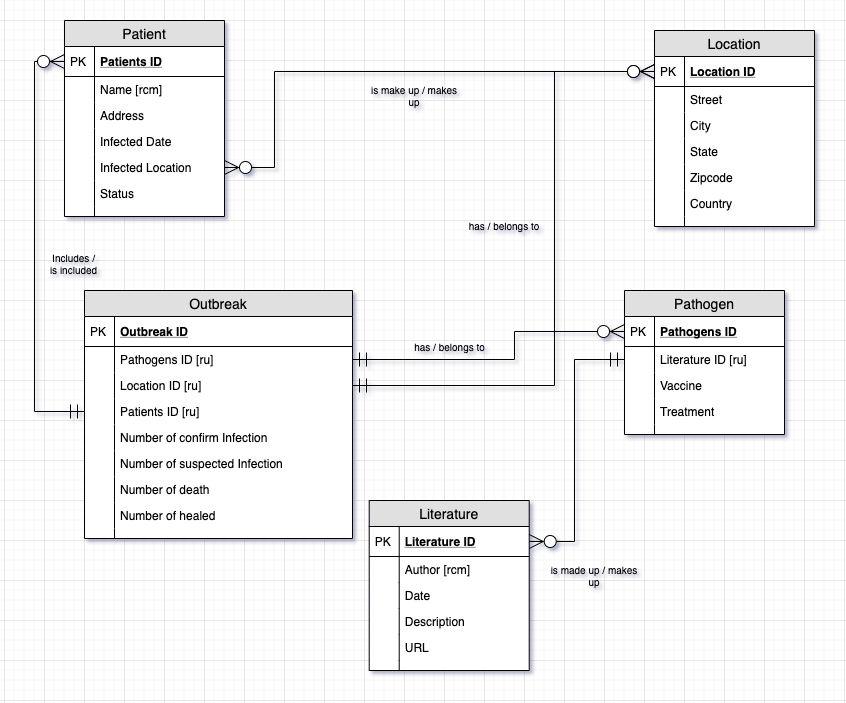
This database is able to answer following questions:

* What is the current number of infected people?
* How many people are healed/dead?
* What location should be avoided going to?
* What are the methods can be used to prevent being infected?
* Are quarantined patients’ actions being tracked?
* Is the outbreak caused by virus, bacteria or other pathogens?

Conceptual Model

Fully dress your business rules and identify all of your stakeholders. Gather and provide unrefined representative examples of your data and provide an Entity-Relationship Diagram (ERD) with a glossary describing in readable text the entities, attributes, and relaitionships in your model.

There are two stakeholders for this project. The owner and the public who can view the information.



Business Rules

The outbreak is caused by one pathogen

Literatures are made from the information of the pathogen

The outbreak has many patients

Outbreak has one or more locations

Glossary

Patients are infected by pathogen

Outbreak is an event caused by pathogens

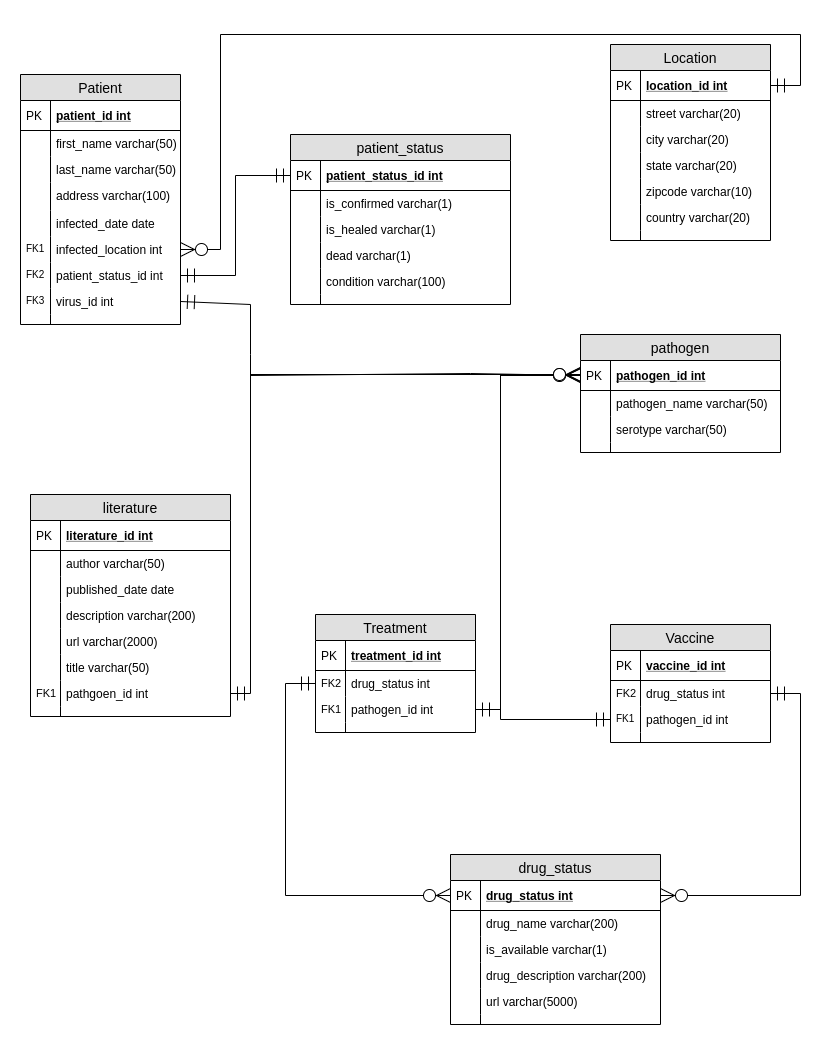
Literature is information about certain pathogen

The literature URL is the external site for the information

Patients status is the current health status of patients

Normalized

Logical Model Decompose your conceptual model into relations and normalize your relations. Provide a database diagram of your normalized relations. Be specific about how the logical model flows from the conceptual model and explain any additional attributes that arise (surrogate keys, etc). It is in this step that you will make choices about data types. Be clear as to what influenced these decisions. It may not be necessary to elaborate on each and every instance of the choice if you implement database-wide rules. For instance, if you choose to implement all primary keys as surrogate keys using the SQL Server identity property, or any date or time attribute will be implemented using the datetime data type, you do not need to provide that information at the table level. Simply state any such decision at the top of this section.



SQL DDL

-- Drop tables if exist

DROP TABLE IF EXISTS Patient

DROP TABLE IF EXISTS Outbreak

DROP TABLE IF EXISTS Patient\_status

DROP TABLE IF EXISTS Locations

DROP TABLE IF EXISTS Pathogen

DROP TABLE IF EXISTS Literature

DROP TABLE IF EXISTS Drug\_status

DROP TABLE IF EXISTS Treatment

DROP TABLE IF EXISTS Vaccine

-- Creating the Drug\_status table

CREATE TABLE Drug\_status (

-- Columns for the Drug\_status table

drug\_status\_id int identity,

drug\_name varchar(200),

is\_available varchar(1) default 'F',

drug\_description varchar(2000),

url varchar(5000),

-- Constraints on the Drug\_status table

CONSTRAINT PK\_drug\_status\_id PRIMARY KEY (drug\_status\_id)

)

-- End creating the Drug\_status table

-- Creating the Patient\_status table

CREATE TABLE Patient\_status (

-- Columns for the Patient\_status table

patient\_status\_id int identity,

is\_confirmed varchar(1),

is\_healed varchar(1),

dead varchar(1),

condition varchar(500),

-- Constraints on the Patient\_status table

CONSTRAINT PK\_patient\_status\_id PRIMARY KEY (patient\_status\_id)

)

-- End creating the Patient\_status table

-- Creating the Location table

CREATE TABLE Locations (

-- Columns for the Location table

location\_id int identity,

street varchar(20),

city varchar(20),

state varchar(20),

zipcode varchar(10),

country varchar(20),

-- Constraints on the Location table

CONSTRAINT PK\_location\_id PRIMARY KEY (location\_id)

)

-- End creating the Location table

-- Creating the Pathogen table

CREATE TABLE Pathogen (

-- Columns for the Pathogens table

pathogen\_id int identity,

pathogen\_name varchar(50),

serotype varchar(50),

-- Constraints on the Pathogens table

CONSTRAINT PK\_pathogen\_id PRIMARY KEY (pathogen\_id)

)

-- End creating the Pathogens table

-- Creating the Literature table

CREATE TABLE Literature (

-- Columns for the Literature table

literature\_id int identity,

title varchar(200),

pathogen\_id int,

author varchar(50),

published\_date datetime,

description varchar(2000),

url varchar(2000),

-- Constraints on the Literature table

CONSTRAINT PK\_literature\_id PRIMARY KEY (literature\_id),

CONSTRAINT FK1\_pathogen\_id FOREIGN KEY (pathogen\_id) REFERENCES Pathogen (pathogen\_id)

)

-- End creating the Literature table

-- Creating the Vaccine table

CREATE TABLE Vaccine (

-- Columns for the Vaccine table

vaccine\_id int identity,

pathogen\_id int,

drug\_status\_id int,

-- Constraints on the Vaccine table

CONSTRAINT PK\_vaccine\_id PRIMARY KEY (vaccine\_id),

CONSTRAINT FK2\_pathogen\_id FOREIGN KEY (pathogen\_id) REFERENCES Pathogen (pathogen\_id),

CONSTRAINT FK1\_drug\_status\_id FOREIGN KEY (drug\_status\_id) REFERENCES Drug\_status (drug\_status\_id)

)

-- End creating the Vaccine table

-- Creating the Treatment table

CREATE TABLE Treatment (

-- Columns for the Treatment table

treatment\_id int identity,

pathogen\_id int,

drug\_status\_id int,

-- Constraints on the Treatment table

CONSTRAINT PK\_treatment\_id PRIMARY KEY (treatment\_id),

CONSTRAINT FK3\_pathogen\_id FOREIGN KEY (pathogen\_id) REFERENCES Pathogen (pathogen\_id),

CONSTRAINT FK2\_drug\_status\_id FOREIGN KEY (drug\_status\_id) REFERENCES Drug\_status (drug\_status\_id)

)

-- End creating the Treatment table

-- Creating the Patient table

CREATE TABLE Patient (

--Columns for the Patient table

patient\_id int identity,

first\_name varchar(50),

last\_name varchar(50),

address varchar(100),

infected\_date datetime,

infected\_location int,

patient\_status int,

pathogen\_id int,

-- Constraints on the User table

CONSTRAINT PK\_patient PRIMARY KEY (patient\_id),

CONSTRAINT FK1\_infected\_location FOREIGN KEY (infected\_location) REFERENCES Locations(location\_id),

CONSTRAINT FK2\_patient\_status FOREIGN KEY (patient\_status) REFERENCES Patient\_status(patient\_status\_id),

CONSTRAINT FK4\_pathgoen\_id FOREIGN KEY (pathogen\_id) REFERENCES Pathogen(pathogen\_id)

)

--End Creating the User table

-- Find the number patients from a specific location

GO

CREATE FUNCTION dbo.LocationCount(@locationID int)

RETURNS int AS -- COUNT() is an integer value, so return it as an int

BEGIN

DECLARE @returnValue int -- matches the function's return type

SELECT @returnValue = COUNT(patient\_id) FROM Patient

WHERE Patient.infected\_location = @locationID

-- Return @returnValue to the calling code

RETURN @returnValue

END

GO

--

SELECT TOP 10

\*,

dbo.LocationCount(infected\_location) as locationCount

FROM Patient

ORDER BY LocationCount DESC

-- Look up pathogen by serotype

GO

-- Function to retrieve the availability

CREATE FUNCTION dbo.drugLookup(@name varchar(50))

RETURNS varchar(1) AS

BEGIN

DECLARE @returnValue varchar(1)-- Matches the function's return type

SELECT @returnValue = is\_available

FROM Drug\_status

WHERE Drug\_status.drug\_name = @name

RETURN @returnValue

END

GO

SELECT dbo.drugLookup('v1')

SELECT dbo.drugLookup('cerveza corona') -- jk

SELECT dbo.drugLookup('t2')

-- Views

-- Views for most infected location

GO

CREATE VIEW MostInfectedLocations AS

SELECT TOP 10

city,

[state],

dbo.LocationCount(infected\_location) as LocationCount

FROM Patient

JOIN Locations ON Locations.location\_id = patient\_id

ORDER BY LocationCount DESC

GO

-- Test the VIEW

SELECT \* FROM MostInfectedLocations

-- Procedures

-- Update patients status

GO

CREATE PROCEDURE UpdatePatientStatus(@patientID int, @confirmed varchar(1), @healed varchar(1), @dead varchar(1),@description varchar(400))

AS

BEGIN

UPDATE Patient\_status SET is\_confirmed = @confirmed, is\_healed = @healed, dead = @dead, condition = @description

WHERE patient\_status\_id = @patientID

--TODO use join to join patient and patient status

END

GO

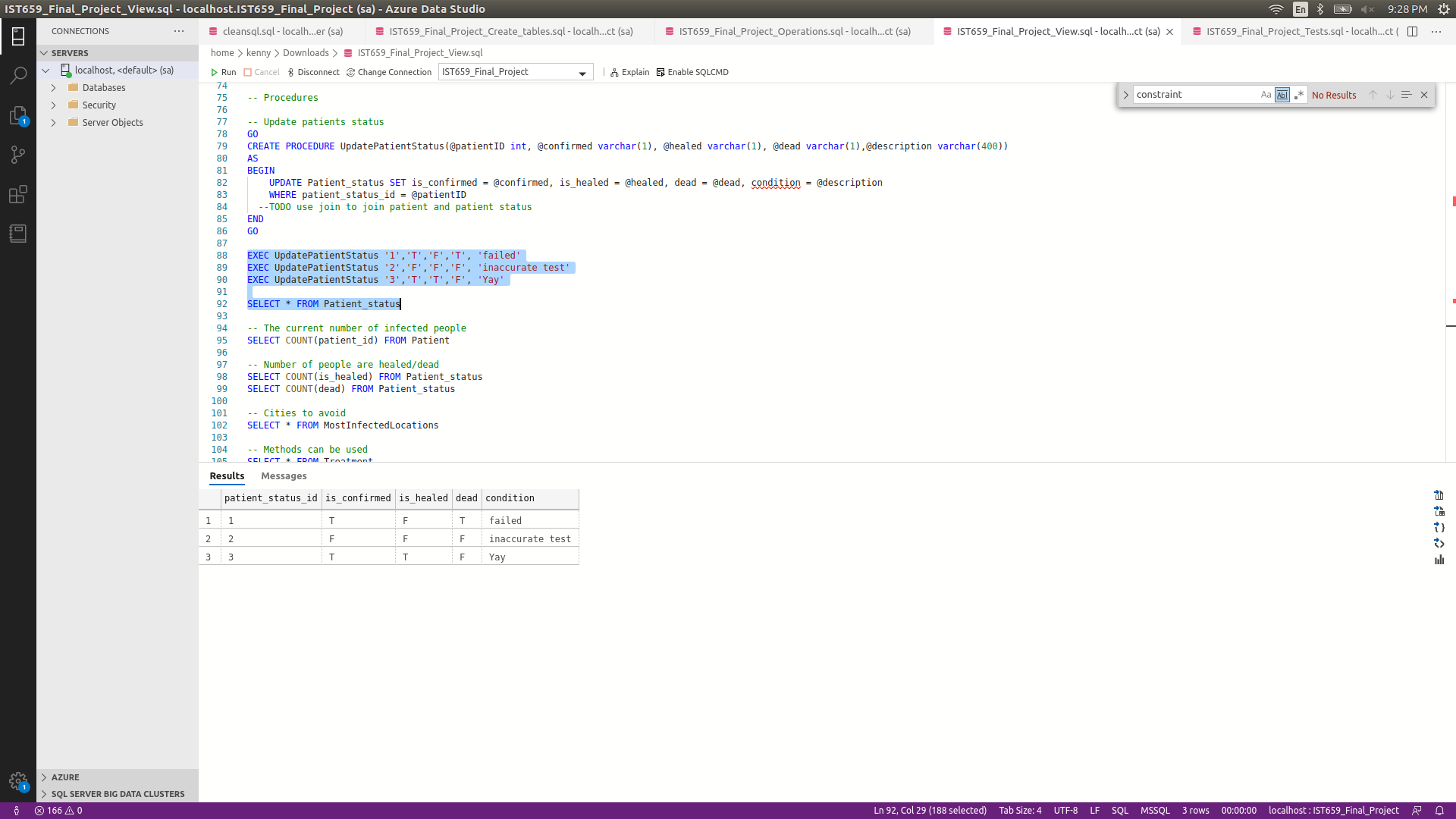
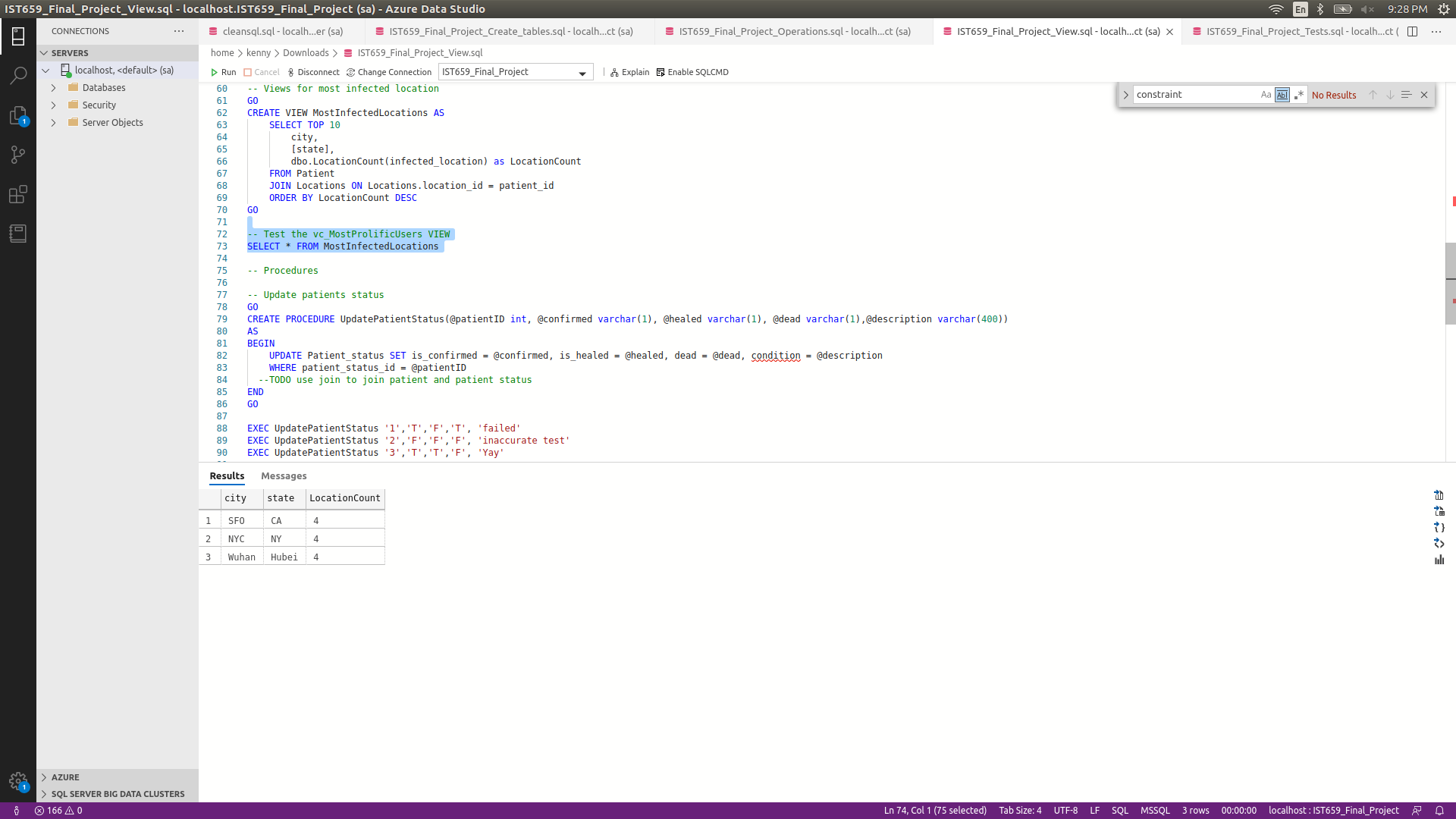
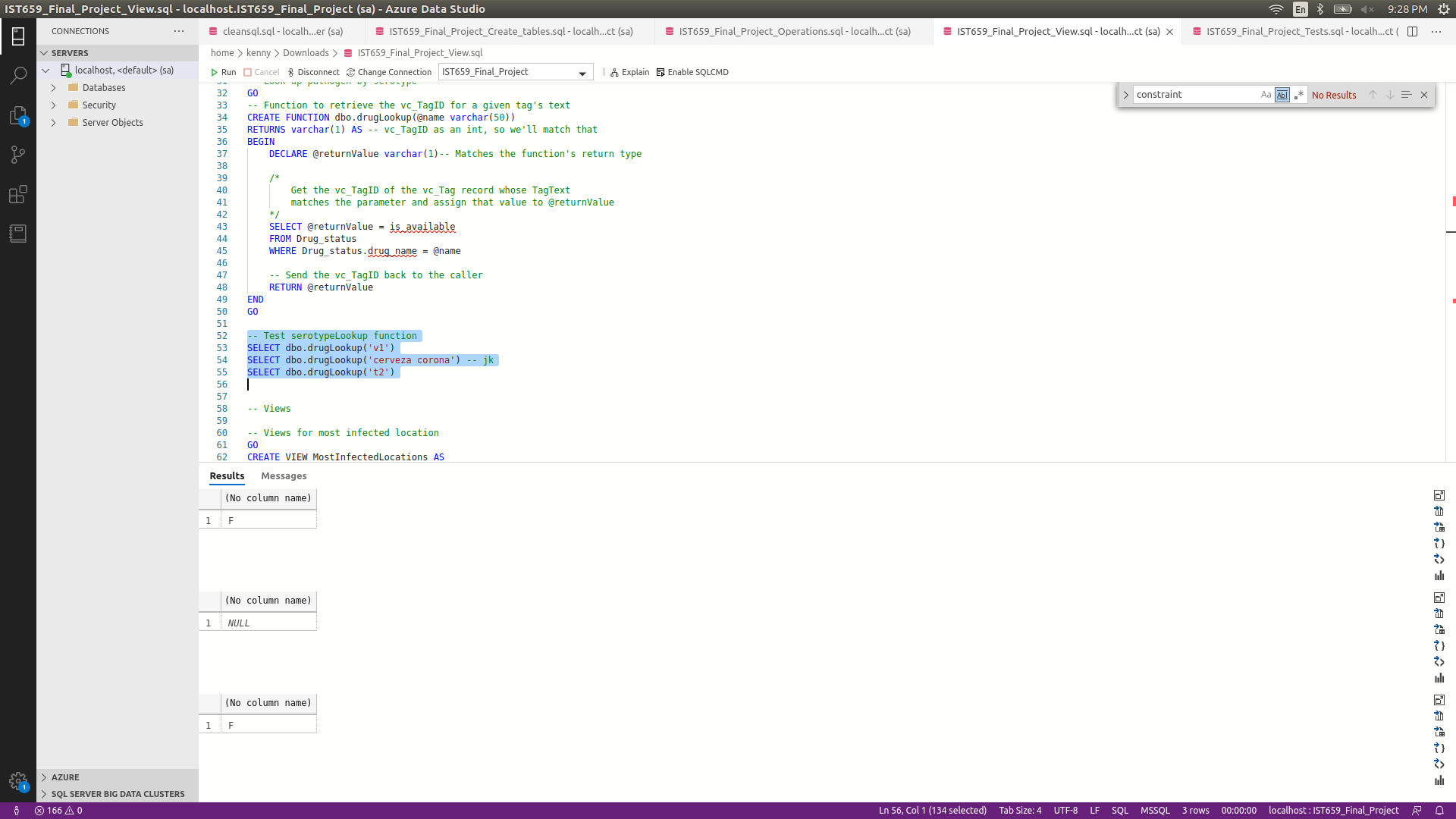
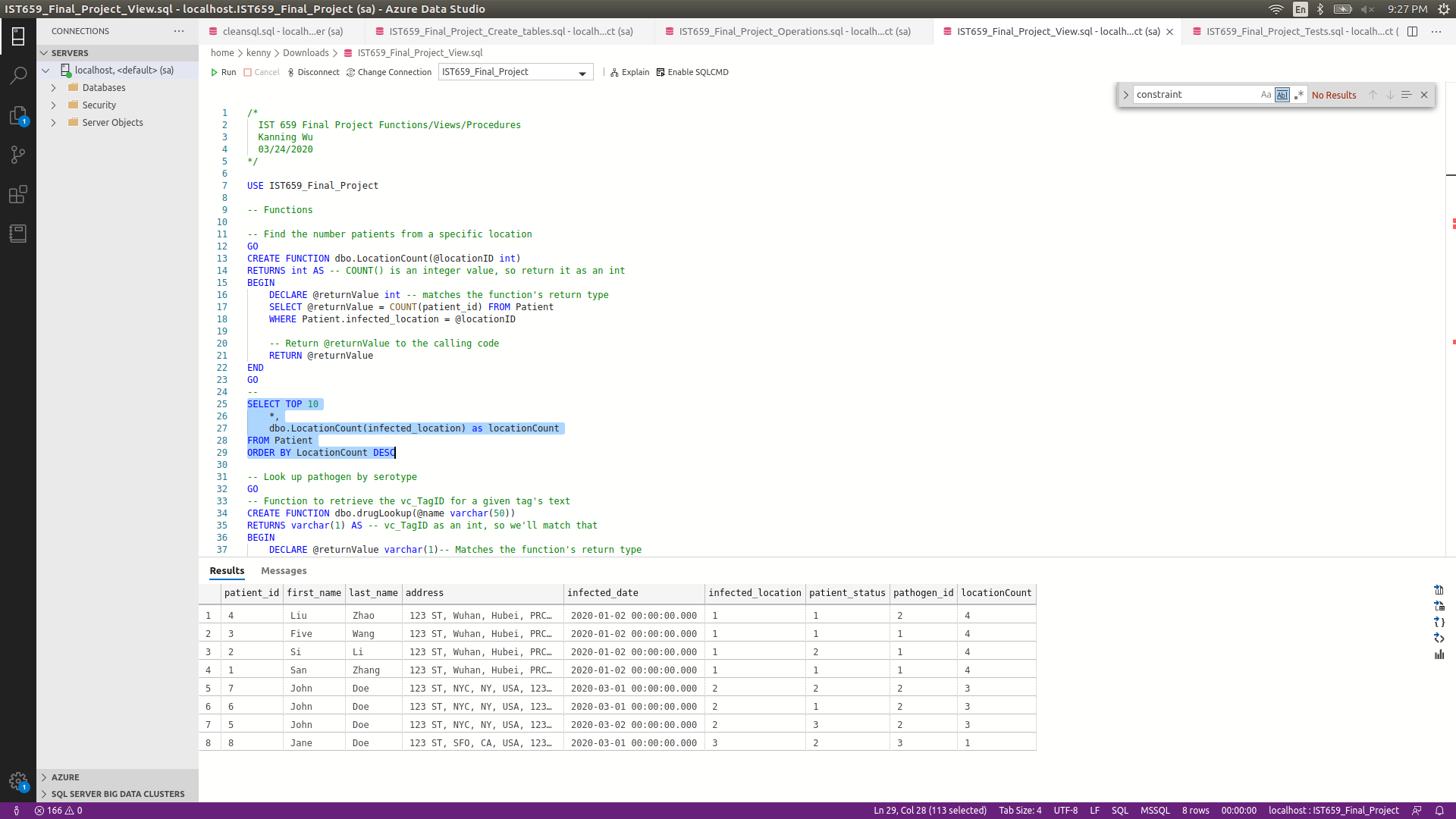
EXEC UpdatePatientStatus '1','T','F','T', 'failed'

EXEC UpdatePatientStatus '2','F','F','F', 'inaccurate test'

EXEC UpdatePatientStatus '3','T','T','F', 'Yay'

SELECT \* FROM Patient\_status

Result from Function, View and Procedure



SQL DML

-- Insert into Drug\_status

INSERT INTO Drug\_status (drug\_name, drug\_description, url)

VALUES ('v1', 'vaccine1', 'http://www.emro.who.int/health-topics/corona-virus/questions-and-answers.html'),

('v2', 'vaccine2', 'www.testurl.com'),

('t1', 'treatment3', '404'),

('t2', 'treatment2', 'google.com')

-- Insert into Patient\_status

INSERT INTO Patient\_status

VALUES ('T','F','F', 'affected, severe symptom')

INSERT INTO Patient\_status

VALUES ('T','T','F', 'very light symptom')

INSERT INTO Patient\_status

VALUES ('T','F','T', 'DEAD')

-- Insert into Location

INSERT INTO Locations

VALUES ('1234 st', 'Wuhan', 'Hubei', '123456', 'PRC')

INSERT INTO Locations

VALUES ('123 st', 'NYC', 'NY', '12345', 'USA')

INSERT INTO Locations

VALUES ('123 st', 'SFO', 'CA', '12345', 'USA')

-- Insert into Pathogen

INSERT INTO Pathogen (pathogen\_name, serotype)

VALUES ('corona virus', 'virus'),

('ccp virus', 'virus'),

('bubonic plague','bacteria')

-- Insert into Vaccine

INSERT INTO Vaccine(pathogen\_id, drug\_status\_id)

VALUES ((SELECT pathogen\_id FROM Pathogen WHERE pathogen\_name = 'corona virus'),

(SELECT drug\_status\_id FROM Drug\_status WHERE drug\_name = 'v1')),

((SELECT pathogen\_id FROM Pathogen WHERE pathogen\_name = 'ccp virus'),

(SELECT drug\_status\_id FROM Drug\_status WHERE drug\_name = 'v2')),

((SELECT pathogen\_id FROM Pathogen WHERE pathogen\_name = 'corona virus'),

(SELECT drug\_status\_id FROM Drug\_status WHERE drug\_name = 'v3'))

-- Insert into Treatment

INSERT INTO Treatment(pathogen\_id, drug\_status\_id)

VALUES ((SELECT pathogen\_id FROM Pathogen WHERE pathogen\_name = 'corona virus'),

(SELECT drug\_status\_id FROM Drug\_status WHERE drug\_name = 't1')),

((SELECT pathogen\_id FROM Pathogen WHERE pathogen\_name = 'ccp virus'),

(SELECT drug\_status\_id FROM Drug\_status WHERE drug\_name = 't2')),

((SELECT pathogen\_id FROM Pathogen WHERE pathogen\_name = 'corona virus'),

(SELECT drug\_status\_id FROM Drug\_status WHERE drug\_name = 't3'))

-- Insert into Literature

INSERT INTO Literature (title, author, published\_date, description, url, pathogen\_id)

VALUES ('Some title',

'CDC',

'3/18/2020',

'How to Protect Yourself',

'https://www.cdc.gov/coronavirus/2019-ncov/community/index.html',

(SELECT pathogen\_id FROM Pathogen WHERE pathogen\_name = 'corona virus')),

('Some title',

'CDC',

'3/21/2020',

'Schools, Workplaces & Community Locations',

'https://www.cdc.gov/coronavirus/2019-ncov/prepare/prevention.html',

(SELECT pathogen\_id FROM Pathogen WHERE pathogen\_name = 'corona virus'))

-- Insert into Patient

INSERT INTO Patient (first\_name, last\_name, address, infected\_date, infected\_location, patient\_status, pathogen\_id)

VALUES ('San', 'Zhang', '123 ST, Wuhan, Hubei, PRC, 123456', '01/02/2020',

(SELECT location\_id FROM Locations WHERE city = 'Wuhan'),1,1),

('Si', 'Li', '123 ST, Wuhan, Hubei, PRC, 123456', '01/02/2020',

(SELECT location\_id FROM Locations WHERE city = 'Wuhan'),2,1),

('Five', 'Wang', '123 ST, Wuhan, Hubei, PRC, 123456', '01/02/2020',

(SELECT location\_id FROM Locations WHERE city = 'Wuhan'),1,1),

('Liu', 'Zhao', '123 ST, Wuhan, Hubei, PRC, 123456', '01/02/2020',

(SELECT location\_id FROM Locations WHERE city = 'Wuhan'),1,2),

('John', 'Doe', '123 ST, NYC, NY, USA, 12345', '03/02/2020',

(SELECT location\_id FROM Locations WHERE city = 'NYC'),3,2),

('John', 'Doe', '123 ST, NYC, NY, USA, 12345', '03/01/2020',

(SELECT location\_id FROM Locations WHERE city = 'NYC'),1,2),

('John', 'Doe', '123 ST, NYC, NY, USA, 12345', '03/01/2020',

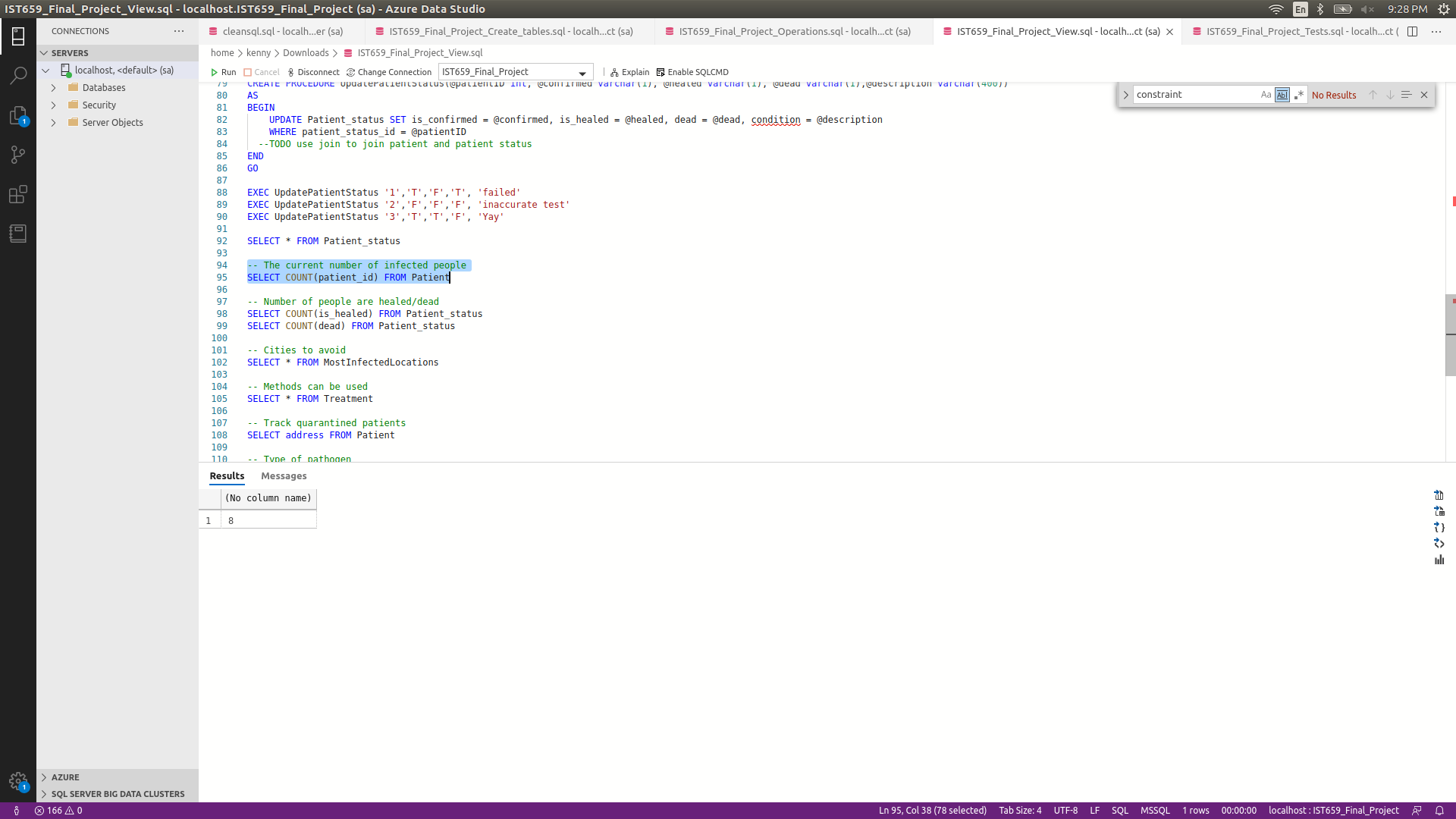
(SELECT location\_id FROM Locations WHERE city = 'NYC'),2,2),

('Jane', 'Doe', '123 ST, SFO, CA, USA, 12345', '03/01/2020',

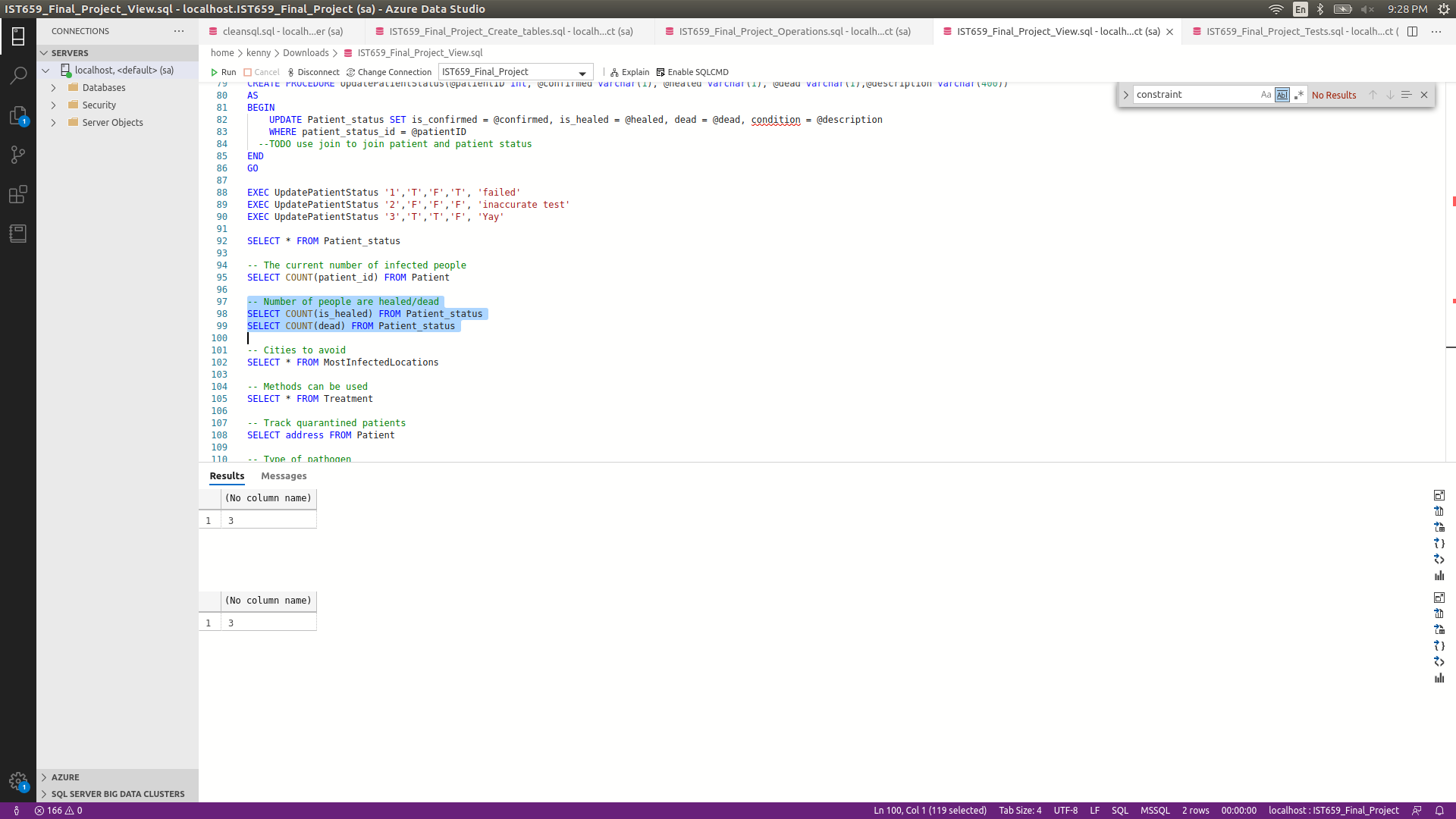
(SELECT location\_id FROM Locations WHERE city = 'SFO'),2,3)

Answer to data questions

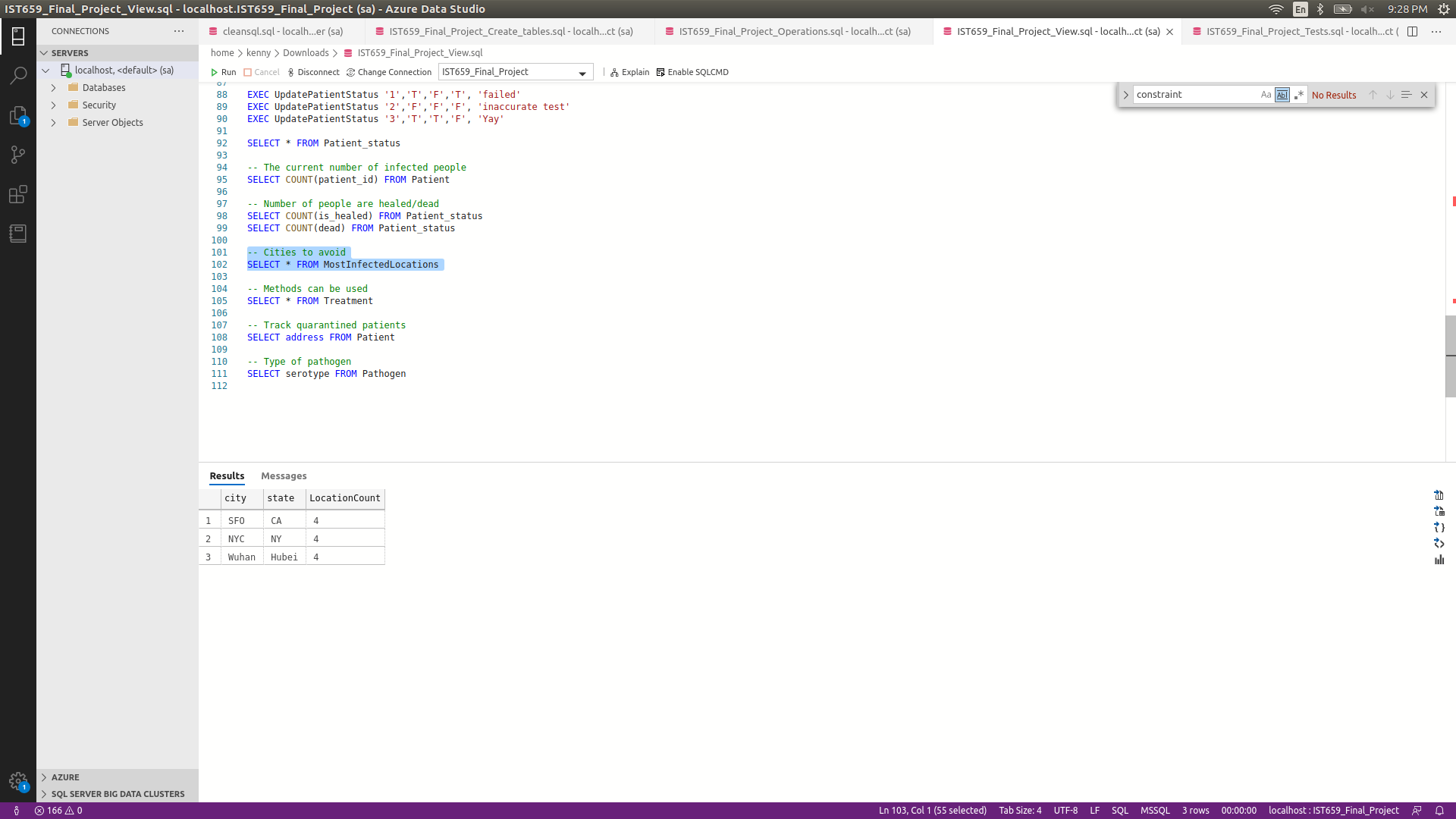
What is the current number of infected people?

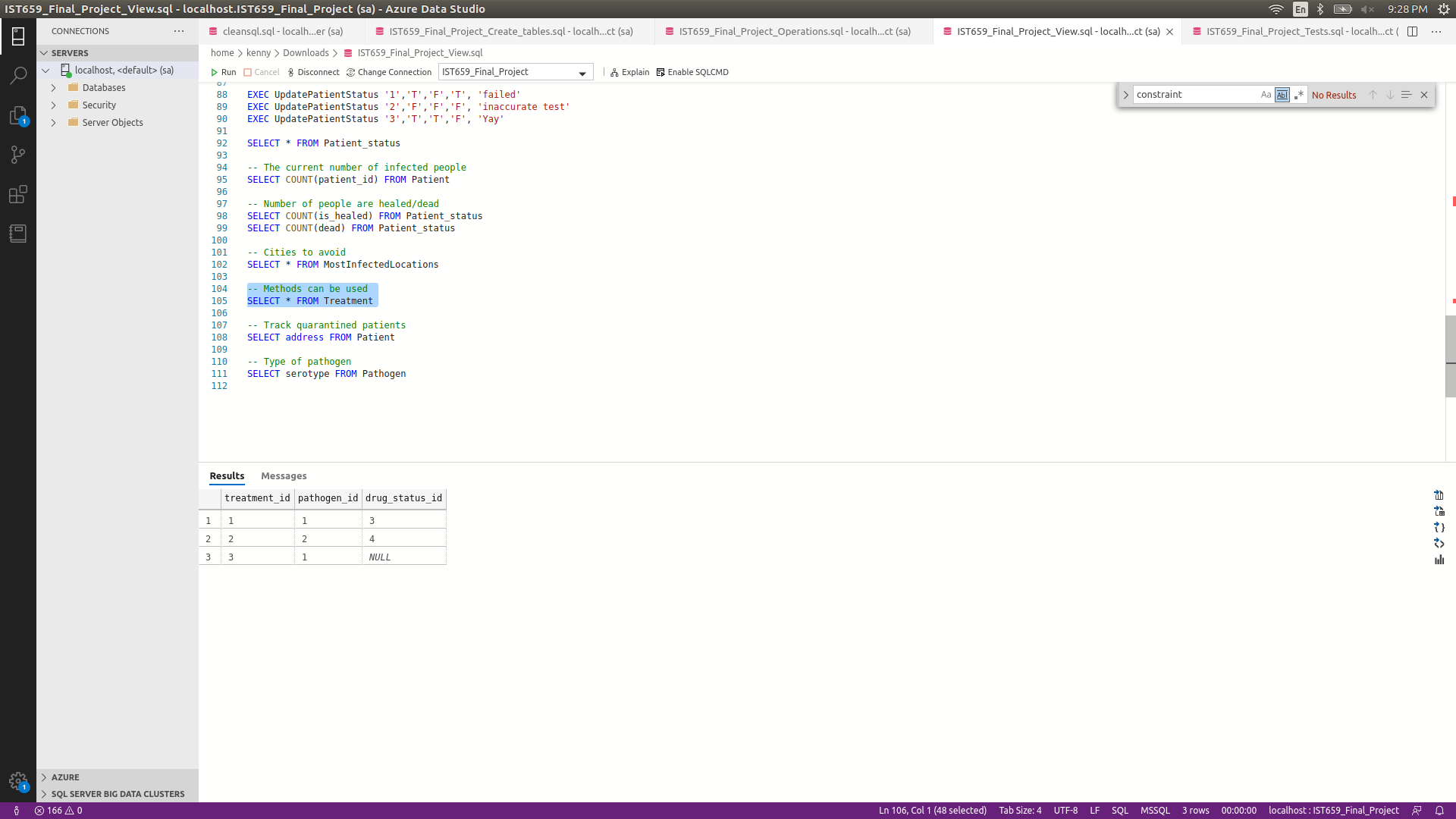


How many people are healed/dead?

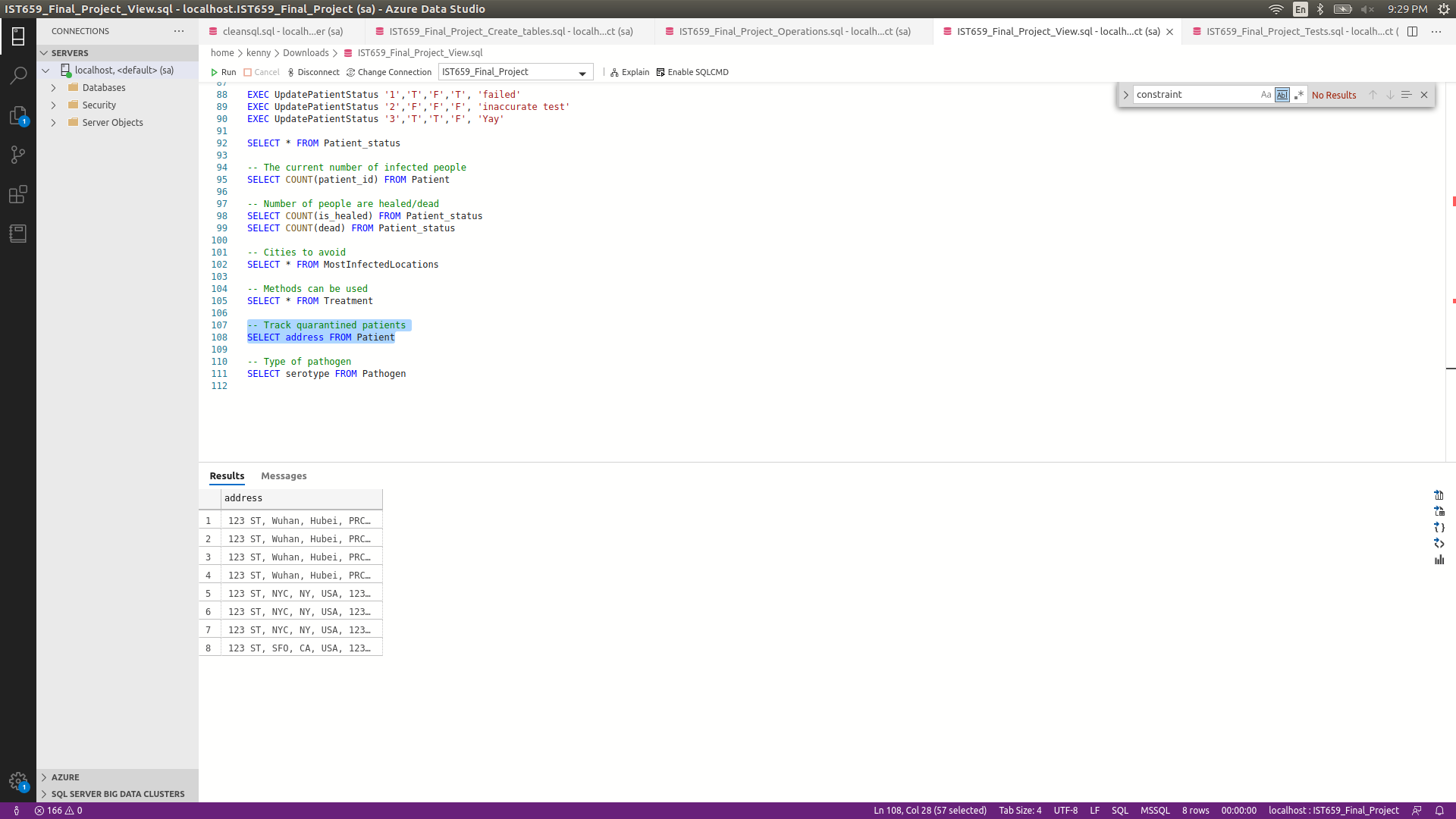


What location should be avoided going to?

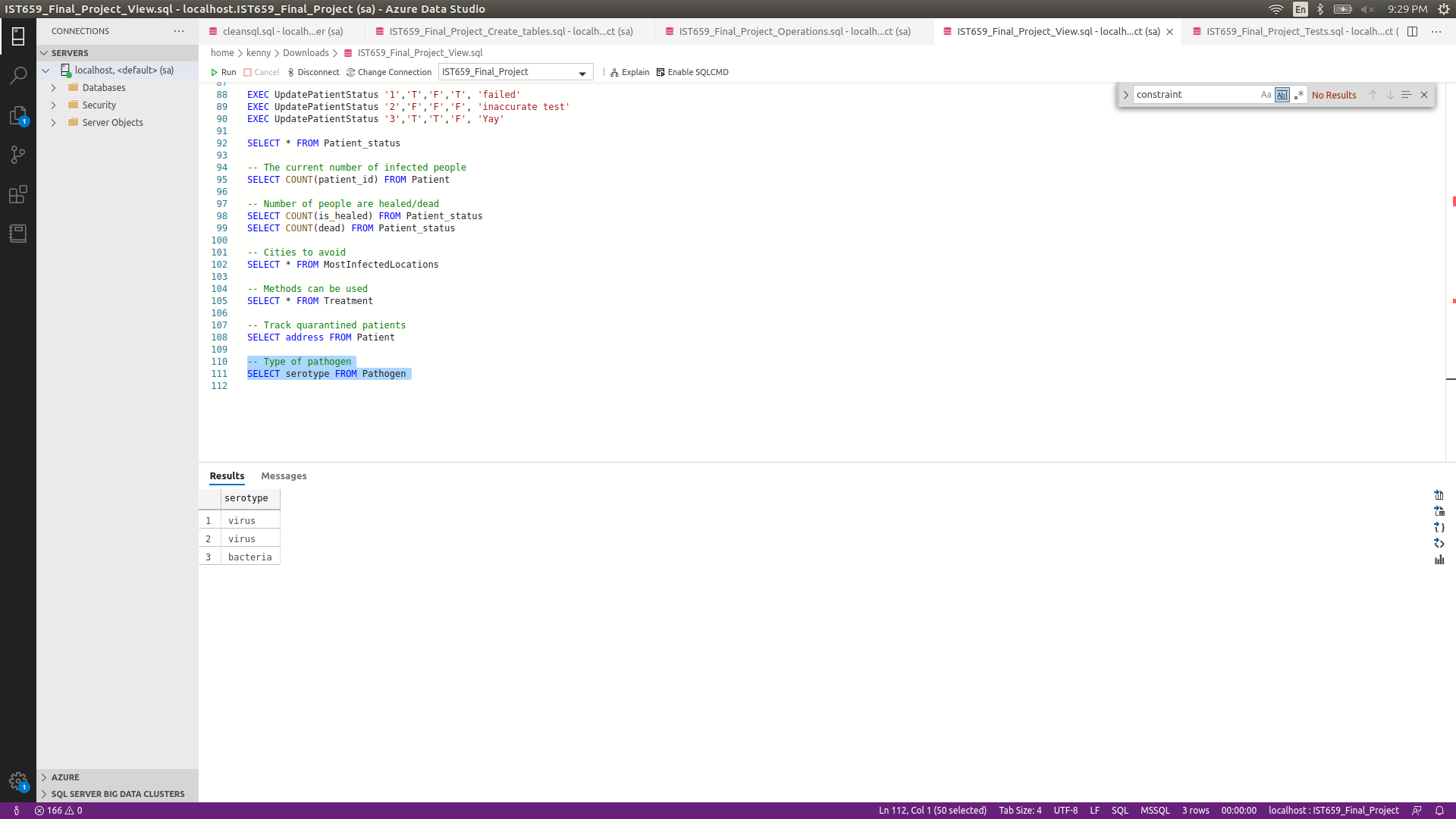


What are the methods can be used to prevent being infected? 

Are quarantined patients’ actions being tracked?



Is the outbreak caused by virus, bacteria or other pathogens?



Reflection

Throughout the project, there were a few mistakes I made in the first deliverable and I made some changes in this final deliverable. The most difficult parts for me was to find out the relationship between each table and make sure the features for the database are well designed. When testing the database, I didn’t find enough resources to test the database. So, I used some meaningless strings as attributes in the table. If there are more time permitted, this database can be even better.