

**2018-0712 IST 659 Data Admin Concepts & Database Management**

**Project Deliverable #2**

**Normalizing: Pharma Contracted Customers Database**

Contents

[The Data Problem 3](#_Toc526013908)

[The Conceptual Model 4](#_Toc526013909)

[Normalized Logical Model V1 5](#_Toc526013910)

[Normalized Logical Model V2 6](#_Toc526013911)

[Project Deliverable 2 7](#_Toc526013912)

[SQL DDL Commands 7](#_Toc526013913)

[More SQL DDL 13](#_Toc526013914)

[SQL DML INSERT Statements 15](#_Toc526013915)

[SQL DML SELECT Statements 20](#_Toc526013916)

[GUI Prototype 21](#_Toc526013917)

# The Data Problem

One of the largest pharmaceutical companies in the world writes performance-based contracts with Physician Buying Groups (PBGs) and Integrated Delivery Networks (IDNs). (Think of a PBG as a Costco for physicians, which allows physicians to achieve savings by purchasing in bulk. An IDN is oftentimes referred to as a healthcare system, think of Keiser Permanente in California or Geisinger Health System in Pennsylvania). The groups and their respective contracts allow affiliated doctors who to receive discounts for various products directly from the corporation or indirectly through a wholesaler. This data is managed and stored by the pharmaceutical company in its own database. Contracted data is sent to a vendor to determine the number of doses purchased by contracted accounts for the respective drugs in the market. Insight about the competition allows the pharma company to develop strategic initiatives. Issues often arise with typos and invalid data due to a poorly designed and managed database. The purpose of this project is to create a database that can house demographic and affiliation information for contracted physicians. Additional databases are needed for the pharmaceutical company, but this database will serve as their foundation.

Some issues with the current database are as follows:

1. There is no unique identifier for a customer
   1. The business has a field called Company\_ID but it is not unique
      1. Sometimes it is shared by different contracts within the same month and/or quarter
2. Other identifiers, which should not change, such as DEA\_ID or HIN\_ID, often change, which makes them ineffective for a candidate key
3. Names and addresses are often misspelled or spelled differently from month-to-month
   1. This makes it difficult to determine if a new customer has been added or if it is simply a misspelling that resulted in a new entry

The pharmaceutical company would like to maintain a database that tracks PBGs and IDNs and the affiliated physicians and facilities that have been under contract since December 2015. In the past, physicians could belong to multiple PBGs and IDNs, however, the business saw this created loopholes for physicians to shop around for the best discount. This led to the implementation of a “one-buyer” rule. This restricts a physician or facility to purchase through one PBG or IDN. However, each PBG and IDN can be comprised of potentially thousands of physicians or facilities. In the future, the business would like to add dose information, so it can understand purchasing patterns, such as seasonality.

Once the database is normalized and restructured, the business would like the database to answer questions such as the following:

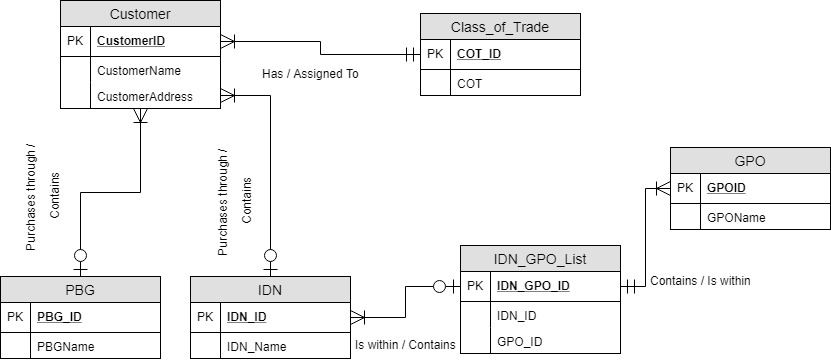
* Which PBG and IDN has the most current members
* Where are most of the contracts located (by state and region)?
* What class of trade (COT) are they engaging with the most?

The business will likely expand upon the proposed database. If it does, future questions would be:

* How many members did we have per month for each PBG and IDN
* Determine the number of doses for each PBG, IDN, physician and facility during a specific timeline
* Are there seasonality patterns appearing in the market?
  + This could be measured by total doses

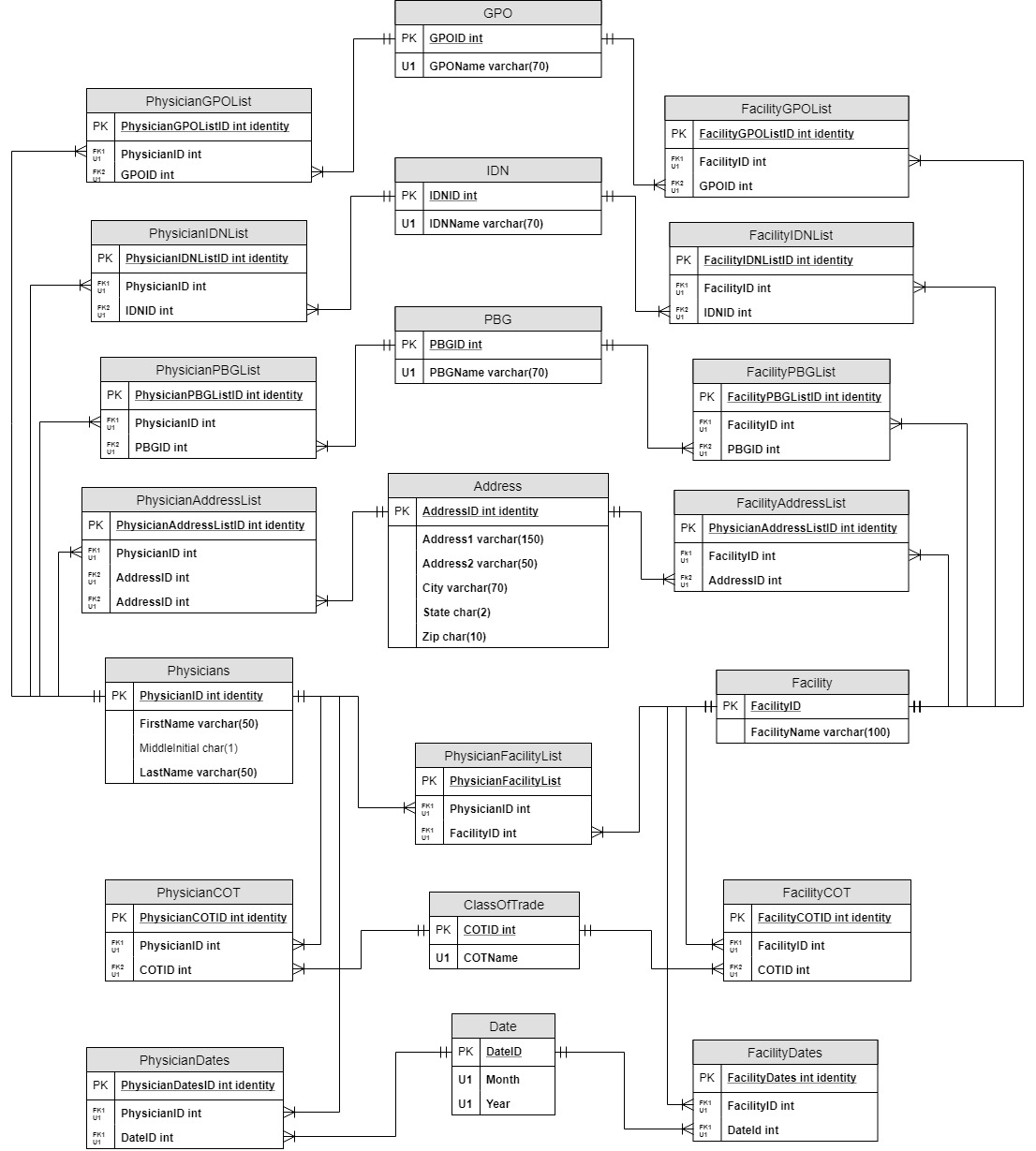
To answer the additional questions, it would require combining databases with an outside vendor. There are potential issues with this solution but the design of this database can help with the creation of additional databases.

# The Conceptual Model



# Normalized Logical Model V1

When normalizing, changes were made to the conceptual model. The first figure represents the initial design, which was submitted for the first deliverable. The second figure includes the proposed edits and was used as the design for the submitted database. (Notes were made an integrated into the final version).



20

19

21

Specialty

Physician\_Specialty

16

3

14

5

1

18

17

15

13

12

11

10

9

8

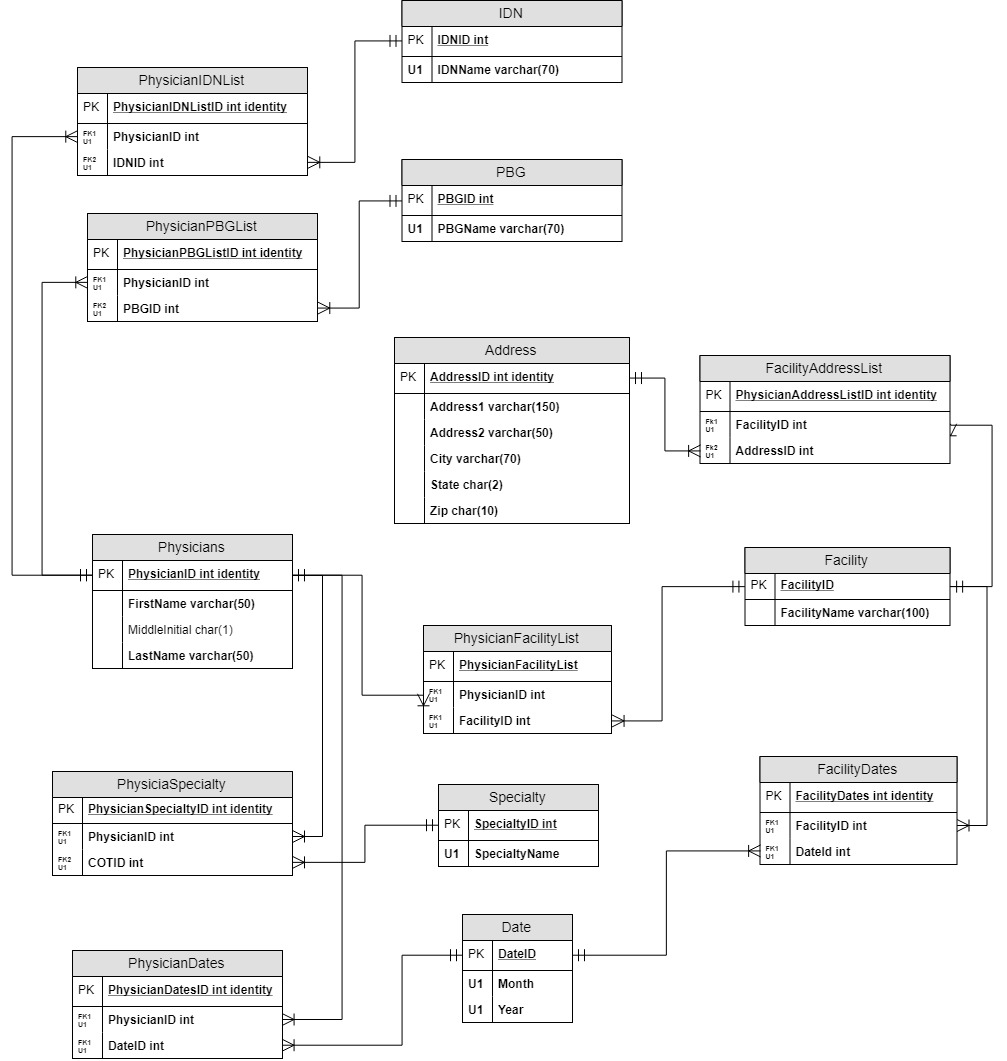
7

6

4

2

# Normalized Logical Model V2



# Project Deliverable 2

## SQL DDL Commands

The SQL CREATE TABLE statements needed to CREATE all tables in the database. Should be executable from top to bottom. Ideally, this would be repeatable by including DROP TABLE statements at the start of your script, but this is not required.

The code for this project can be found in the Appendix. A CREATE statement is available for all tables, but only 10 are listed here to help for grading purposes.

/\*TABLE 1

DROP AND CREATE PHYSICIANS TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIANS'

)

BEGIN

DROP TABLE PHYSICIANS

END

CREATE TABLE PHYSICIANS (

PhysicianID int identity,

FirstName varchar(50) not null,

MiddleInitial char(1),

LastName varchar(50) not null,

CONSTRAINT PK\_PHYSICIANS PRIMARY KEY (PhysicianID)

)

/\*TABLE 2

DROP AND CREATE FACILITY TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'FACILITY'

)

BEGIN

DROP TABLE FACILITY

END

CREATE TABLE FACILITY (

FacilityID int identity,

FacilityName varchar(100) not null,

CONSTRAINT PK\_FACILITY PRIMARY KEY (FacilityID)

)

/\*TABLE 3

DROP AND CREATE PHYSICIAN\_FACILITY\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_FACILITY\_LIST'

)

BEGIN

DROP TABLE PHYSICIAN\_FACILITY\_LIST

END

CREATE TABLE PHYSICIAN\_FACILITY\_LIST (

PhysicianFacilityListID int identity,

PhysicianID int not null,

FacilityID int not null,

CONSTRAINT PK\_PHYSICIAN\_FACILITY\_LIST PRIMARY KEY (PhysicianFacilityListID),

CONSTRAINT FK1\_PHYSICIAN\_FACILITY\_LIST FOREIGN KEY (PhysicianID) REFERENCES PHYSICIANS (PhysicianID),

CONSTRAINT FK2\_PHYSICIAN\_FACILITY\_LIST FOREIGN KEY (FacilityID) REFERENCES FACILITY (FacilityID),

CONSTRAINT U1\_PHYSICIAN\_FACILITY\_LIST UNIQUE (PhysicianID, FacilityID)

)

/\*TABLE 4

DROP AND CREATE ADDRESSINFO TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'ADDRESS\_INFO'

)

BEGIN

DROP TABLE ADDRESS\_INFO

END

CREATE TABLE ADDRESS\_INFO (

AddressID int identity,

Address1 varchar(100) not null,

Address2 varchar(50) not null,

City varchar(70) not null,

ST char(2) not null,

ZIP char(10) not null,

CONSTRAINT PK\_ADDRESS\_INFO PRIMARY KEY (AddressID)

)

/\*TABLE 5

DROP AND CREATE PHYSICIAN\_ADDRESS\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_ADDRESS\_LIST'

)

BEGIN

DROP TABLE PHYSICIAN\_ADDRESS\_LIST

END

CREATE TABLE PHYSICIAN\_ADDRESS\_LIST (

PhysicianAddressListID int identity,

PhysicianID int not null,

AddressID int not null,

CONSTRAINT PK\_PHYSICIAN\_ADDRESS\_LIST PRIMARY KEY (PhysicianAddressListID),

CONSTRAINT FK1\_PHYSICIAN\_ADDRESS\_LIST FOREIGN KEY (PhysicianID) REFERENCES PHYSICIANS (PhysicianID),

CONSTRAINT FK2\_PHYSICIAN\_ADDRESS\_LIST FOREIGN KEY (AddressID) REFERENCES ADDRESS\_INFO (AddressID),

CONSTRAINT U1\_PHYSICIAN\_ADDRESS\_LIST UNIQUE (PhysicianID, AddressID)

)

/\*TABLE 6

DROP AND CREATE FACILITY\_ADDRESS\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'FACILITY\_ADDRESS\_LIST'

)

BEGIN

DROP TABLE FACILITY\_ADDRESS\_LIST

END

CREATE TABLE FACILITY\_ADDRESS\_LIST (

FacilityAddressListID int identity,

FacilityID int not null,

AddressID int not null,

CONSTRAINT PK\_FACILITY\_ADDRESS\_LIST PRIMARY KEY (FacilityAddressListID),

CONSTRAINT FK1\_FACILITY\_ADDRESS\_LIST FOREIGN KEY (FacilityID) REFERENCES FACILITY(FacilityID),

CONSTRAINT FK2\_FACILITY\_ADDRESS\_LIST FOREIGN KEY (AddressID) REFERENCES ADDRESS\_INFO(AddressID),

CONSTRAINT U1\_FACILITY\_ADDRESS\_LIST UNIQUE (FacilityID, AddressID)

)

/\*TABLE 7

DROP AND CREATE PBG TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PBG'

)

BEGIN

DROP TABLE PBG

END

CREATE TABLE PBG (

PBGID int identity PRIMARY KEY,

PBGName varchar(70) not null UNIQUE,

)

/\*TABLE 8

DROP AND CREATE PHYSICIAN\_PBG\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_PBG\_LIST'

)

BEGIN

DROP TABLE PHYSICIAN\_PBG\_LIST

END

CREATE TABLE PHYSICIAN\_PBG\_LIST (

PhysicianPBGListID int identity PRIMARY KEY,

PhysicianID int not null FOREIGN KEY REFERENCES PHYSICIANS(PhysicianID),

PBGID int not null FOREIGN KEY REFERENCES PBG(PBGID),

)

/\*TABLE 9

DROP AND CREATE IDN TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'IDN'

)

BEGIN

DROP TABLE IDN

END

CREATE TABLE IDN (

IDNID int identity PRIMARY KEY,

IDNName varchar(70) not null UNIQUE

)

/\*TABLE 10

DROP AND CREATE PHYSICIAN\_IDN\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_IDN\_LIST'

)

BEGIN

DROP TABLE PHYSICIAN\_IDN\_LIST

END

CREATE TABLE PHYSICIAN\_IDN\_LIST (

PhysicianIDNListID int identity PRIMARY KEY,

PhysicianID int not null FOREIGN KEY REFERENCES PHYSICIANS (PhysicianID),

IDNID int not null FOREIGN KEY REFERENCES IDN(IDNID)

)

## More SQL DDL

CREATE Statements for at least five programming objects from among Views, User-defined Functions, and Stored Procedures. You may split this up in whatever way makes sense for your problem space, but be sure to include one of each.

You may also code more than five if it makes sense for your database.

/\*CREATE VIEW - 1

Identify full list to indicate where physicians are practicing. Physicians can practice in multiple locations.

Due to this being test date, actual locations are not to be interpreted as realistic\*/

CREATE VIEW PHYSICIAN\_LOCATIONS AS

SELECT P.FirstName, P.LastName, F.FacilityName, A.Address1, A.Address2, A.City, A.ST, A.ZIP

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_FACILITY\_LIST AS PF ON P.PhysicianID = PF.PhysicianID

JOIN FACILITY AS F ON PF.FacilityID = F.FacilityID

JOIN FACILITY\_ADDRESS\_LIST AS FA ON F.FacilityID = FA.FacilityID

JOIN ADDRESS\_INFO AS A ON FA.AddressID = A.AddressID

GO

/\*CREATE VIEW - 2

Identify list of PBGs that physicians have been associated to\*/

CREATE VIEW PHYSICIAN\_PBG\_AFFILIATION AS

SELECT P.FirstName, P.LastName, P.Suffix, PBG.PBGName

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_PBG\_LIST AS PBL ON P.PhysicianID = PBL.PBGID

JOIN PBG ON PBL.PBGID = PBG.PBGID

GO

/\*CREATE VIEW - 3

Provides listing of when Physicians contracts were added and renewed\*/

CREATE VIEW PHYSICIAN\_ADDED AS

SELECT P.FirstName, P.LastName, P.Suffix, D.Months, D.Years

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_DATES AS PD ON P.PhysicianID = PD.PhysicianId

JOIN DATES AS D ON PD.DateID = D.DateID

ORDER BY P.LastName OFFSET 0 ROWS

GO

/\*CREATE FUNCTION - 1

Allows user to quickly identify the Physician ID, which can be useful when populating other tables

or for a WHERE clause\*/

CREATE FUNCTION dbo.PhysicianIDLookup(@FirstName varchar(70), @LastName varchar(70), @Suffix varchar(5))

RETURNS int AS

BEGIN

DECLARE @returnvalue int

SELECT @returnvalue = PhysicianID FROM PHYSICIANS

WHERE FirstName = @FirstName AND LastName = @LastName AND Suffix = @Suffix

RETURN @returnvalue

END

GO

SELECT dbo.PhysicianIDLookup ('William', 'Holt', 'III')

/\*CREATE STORED PROCEDURE - 1

Instead of using Insert Into, create a procedure to end the information needed\*/

CREATE PROCEDURE AddPhysician (@FirstName varchar(50), @MiddleInitial char(1), @LastName varchar(50), @Suffix varchar(4))

AS

BEGIN

INSERT INTO PHYSICIANS

(FirstName, MiddleInitial, LastName, Suffix)

VALUES

(@FirstName, @MiddleInitial, @LastName, @Suffix)

END

GO

## SQL DML INSERT Statements

Code at least ten (10) SQL INSERT Statements for data in your database. You should have INSERTS for at least two different tables, but approximately 10 statements altogether.

Feel free to load the rest of the data in any way you prefer (Import from Excel, hand-keyed data entry from the UI, etc), but we’ll need to see the proper INSERT statements.

NOTE: If you abstracted your INSERT statements using a stored procedure, don’t rewrite those statements here. Instead, show the EXECUTE statement to call them. Also, nice job!

For this part, you can optionally code UPDATE and DELETE statements to count toward the 10 statements.

All tables were update through Insert statements, which can be found in the Appendix. Listed below are 10 Insert statements to help for grading purposes:

/\*INSERT INTO - 1\*/

INSERT INTO PHYSICIANS

(FirstName, MiddleInitial, LastName, Suffix)

VALUES

('Chris', 'M', 'Fandl', ''),

('Mike', 'C', 'Trescavage', ''),

('Lindsay', 'E', 'Holt', ''),

('Joshua', 'W', 'Laird', ''),

('Jennifer', '', 'Hufton', ''),

('Maynard', 'J', 'Keenan', ''),

('Jimmy', '', 'Page', ''),

('Robert', '', 'Plant', ''),

('Reggie', 'H', 'White', ''),

('Jerome', 'W', 'Brown', 'III')

GO

/\*INSERT INTO - 2\*/

INSERT INTO ADDRESS\_INFO

(Address1, Address2, City, ST, ZIP)

VALUES

('328 Elizabeth Drive', '', 'Kennett Square', 'PA', '19348'),

('808 Outter Space Blvd', '', 'Camden', 'NJ', '01342'),

('8 Plum Street', 'STE 1', 'Marcus Hook', 'GA', '68763'),

('28 Oragne Drive', '', 'Jackson', 'CO', '89173'),

('78 Brookline Road', '', 'Lateralus', 'CA', '90232'),

('156 Whispers Way', '', 'Deptford', 'MO', '56943'),

('1029 Walnut Lane', 'STE 2', 'Vicarious', 'MD', '27378'),

('67 Arch Street', '', 'Maryton', 'FL', '43143'),

('9256 Coastline Drive', '', 'Hilton Head', 'SC', '45221')

GO

/\*INSERT INTO - 3\*/

INSERT INTO PHYSICIAN\_ADDRESS\_LIST

(PhysicianID, AddressID)

VALUES

('1', '3'),

('2', '4'),

('2', '5'),

('2', '8'),

('3', '1'),

('4', '3'),

('4', '7'),

('4', '6'),

('5', '8'),

('5', '2'),

('6', '2'),

('6', '5'),

('7', '4'),

('7', '7'),

('7', '8'),

('8', '2'),

('9', '5'),

('9', '6'),

('9', '4'),

('10', '3'),

('10', '4'),

('11', '5'),

('11', '3'),

('11', '4'),

('11', '2'),

('12', '7'),

('12', '2'),

('13', '6'),

('13', '8'),

('14', '4')

GO

/\*INSERT INTO - 4\*/

INSERT INTO FACILITY

(FacilityName)

VALUES

('Saint Christopher''s of Camden'),

('First Response Care'),

('Children''s Health of Florida'),

('Family Awareness Practice'),

('Hopewell Family Practice'),

('Penn Medicine Peds'),

('Rothman Institute'),

('Careville Kids'),

('The Pediatric Center')

GO

/\*INSERT INTO - 5\*/

INSERT INTO PHYSICIAN\_FACILITY\_LIST

(PhysicianID, FacilityID)

VALUES

('4','7'),

('14','1'),

('13','8'),

('12','7'),

('13','3'),

('1','3'),

('12','1'),

('8','1'),

('2','6'),

('9','7'),

('3','1'),

('8','7'),

('3','6'),

('5','2'),

('2','3'),

('10','3'),

('1','1'),

('3','5'),

('11','3'),

('5','4'),

('14','7'),

('5','7'),

('11','7'),

('7','1'),

('6','1')

GO

/\*INSERT INTO - 6\*/

INSERT INTO FACILITY\_ADDRESS\_LIST

(FacilityID, AddressID)

VALUES

('1', '2'),

('2', '9'),

('2', '3'),

('3', '8'),

('4', '8'),

('5', '7'),

('6', '1'),

('7', '4'),

('7', '6'),

('8', '5'),

('9', '9')

GO

/\*INSERT INTO -7\*/

INSERT INTO PBG

(PBGName)

VALUES

('Catholic Health Partners'),

('CareSense'),

('Practice True'),

('Southern Hills'),

('EasternShore')

GO

/\*INSERT INTO - 8\*/

INSERT INTO PHYSICIAN\_PBG\_LIST

(PhysicianID, PBGID)

VALUES

('13','4'),

('2','2'),

('1','4'),

('2','3'),

('1','4'),

('9','4'),

('4','4'),

('11','2'),

('10','5'),

('14','3'),

('10','4'),

('7','1'),

('8','3'),

('10','5'),

('13','3'),

('3','1'),

('1','2'),

('5','3'),

('3','3'),

('1','5'),

('2','2'),

('7','2'),

('13','3'),

('12','2'),

('6','1'),

('5','1')

GO

/\*INSERT INTO - 9\*/

INSERT INTO IDN

(IDNName)

VALUES

('Lord''s Health System'),

('Vision Health'),

('Grandview Healthcare'),

('Einstein'),

('West Falls Healthcare'),

('Burbank Medicine')

GO

/\*INSERT INTO - 10\*/

INSERT INTO PHYSICIAN\_PBG\_LIST

(PhysicianID, PBGID)

VALUES

('1','3'),

('2','4'),

('2','5'),

('4','1'),

('5','2'),

('7','5'),

('8','1'),

('10','4'),

('10','3'),

('13','1'),

('14','2'),

('14','3')

GO

## SQL DML SELECT Statements

In Project Deliverable 1, you should have thought of three to five data questions you hoped your data would answer. Write the SQL SELECT statements that show the data you would need to answer those questions.

If you already coded these into Views or Functions, for this part, write the code that would use those Views and Functions here.

The Views mentioned above were created to answer some of the questions. The code without the View part has been removed and is below, but not included in the complete code found in the appendix.

/\*SELECT – 1

Retrieves physician practicing locations\*/

SELECT P.FirstName, P.LastName, F.FacilityName, A.Address1, A.Address2, A.City, A.ST, A.ZIP

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_FACILITY\_LIST AS PF ON P.PhysicianID = PF.PhysicianID

JOIN FACILITY AS F ON PF.FacilityID = F.FacilityID

JOIN FACILITY\_ADDRESS\_LIST AS FA ON F.FacilityID = FA.FacilityID

JOIN ADDRESS\_INFO AS A ON FA.AddressID = A.AddressID

/\*SELECT – 2

Retrieves the physicians and the PBGs he or she are affiliated to\*/

SELECT P.FirstName, P.LastName, P.Suffix, PBG.PBGName

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_PBG\_LIST AS PBL ON P.PhysicianID = PBL.PBGID

JOIN PBG ON PBL.PBGID = PBG.PBGID

GO

/\*SELECT – 3

Retrieves when physicians were added and renewed\*/

SELECT P.FirstName, P.LastName, P.Suffix, D.Months, D.Years

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_DATES AS PD ON P.PhysicianID = PD.PhysicianId

JOIN DATES AS D ON PD.DateID = D.DateID

ORDER BY P.LastName OFFSET 0 ROWS

GO

## GUI Prototype

Prototype at least three GUI screens for your database. You are free to use any technology you prefer, but be clear about what tables are in use on your screens.

## Appendix

All relevant code used to build the database is found below

CREATE DATABASE IST659\_PROJECT

/\*TABLE 1

DROP AND CREATE PHYSICIANS TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIANS'

)

BEGIN

DROP TABLE PHYSICIANS

END

CREATE TABLE PHYSICIANS (

PhysicianID int identity,

FirstName varchar(50) not null,

MiddleInitial char(1),

LastName varchar(50) not null,

CONSTRAINT PK\_PHYSICIANS PRIMARY KEY (PhysicianID)

)

/\*TABLE 2

DROP AND CREATE FACILITY TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'FACILITY'

)

BEGIN

DROP TABLE FACILITY

END

CREATE TABLE FACILITY (

FacilityID int identity,

FacilityName varchar(100) not null,

CONSTRAINT PK\_FACILITY PRIMARY KEY (FacilityID)

)

/\*TABLE 3

DROP AND CREATE PHYSICIAN\_FACILITY\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_FACILITY\_LIST'

)

BEGIN

DROP TABLE PHYSICIAN\_FACILITY\_LIST

END

CREATE TABLE PHYSICIAN\_FACILITY\_LIST (

PhysicianFacilityListID int identity,

PhysicianID int not null,

FacilityID int not null,

CONSTRAINT PK\_PHYSICIAN\_FACILITY\_LIST PRIMARY KEY (PhysicianFacilityListID),

CONSTRAINT FK1\_PHYSICIAN\_FACILITY\_LIST FOREIGN KEY (PhysicianID) REFERENCES PHYSICIANS (PhysicianID),

CONSTRAINT FK2\_PHYSICIAN\_FACILITY\_LIST FOREIGN KEY (FacilityID) REFERENCES FACILITY (FacilityID),

CONSTRAINT U1\_PHYSICIAN\_FACILITY\_LIST UNIQUE (PhysicianID, FacilityID)

)

/\*TABLE 4

DROP AND CREATE ADDRESSINFO TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'ADDRESS\_INFO'

)

BEGIN

DROP TABLE ADDRESS\_INFO

END

CREATE TABLE ADDRESS\_INFO (

AddressID int identity,

Address1 varchar(100) not null,

Address2 varchar(50) not null,

City varchar(70) not null,

ST char(2) not null,

ZIP char(10) not null,

CONSTRAINT PK\_ADDRESS\_INFO PRIMARY KEY (AddressID)

)

/\*TABLE 5

DROP AND CREATE PHYSICIAN\_ADDRESS\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_ADDRESS\_LIST'

)

BEGIN

DROP TABLE PHYSICIAN\_ADDRESS\_LIST

END

CREATE TABLE PHYSICIAN\_ADDRESS\_LIST (

PhysicianAddressListID int identity,

PhysicianID int not null,

AddressID int not null,

CONSTRAINT PK\_PHYSICIAN\_ADDRESS\_LIST PRIMARY KEY (PhysicianAddressListID),

CONSTRAINT FK1\_PHYSICIAN\_ADDRESS\_LIST FOREIGN KEY (PhysicianID) REFERENCES PHYSICIANS (PhysicianID),

CONSTRAINT FK2\_PHYSICIAN\_ADDRESS\_LIST FOREIGN KEY (AddressID) REFERENCES ADDRESS\_INFO (AddressID),

CONSTRAINT U1\_PHYSICIAN\_ADDRESS\_LIST UNIQUE (PhysicianID, AddressID)

)

/\*TABLE 6

DROP AND CREATE FACILITY\_ADDRESS\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'FACILITY\_ADDRESS\_LIST'

)

BEGIN

DROP TABLE FACILITY\_ADDRESS\_LIST

END

CREATE TABLE FACILITY\_ADDRESS\_LIST (

FacilityAddressListID int identity,

FacilityID int not null,

AddressID int not null,

CONSTRAINT PK\_FACILITY\_ADDRESS\_LIST PRIMARY KEY (FacilityAddressListID),

CONSTRAINT FK1\_FACILITY\_ADDRESS\_LIST FOREIGN KEY (FacilityID) REFERENCES FACILITY(FacilityID),

CONSTRAINT FK2\_FACILITY\_ADDRESS\_LIST FOREIGN KEY (AddressID) REFERENCES ADDRESS\_INFO(AddressID),

CONSTRAINT U1\_FACILITY\_ADDRESS\_LIST UNIQUE (FacilityID, AddressID)

)

/\*TABLE 7

DROP AND CREATE PBG TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PBG'

)

BEGIN

DROP TABLE PBG

END

CREATE TABLE PBG (

PBGID int identity PRIMARY KEY,

PBGName varchar(70) not null UNIQUE,

)

/\*TABLE 8

DROP AND CREATE PHYSICIAN\_PBG\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_PBG\_LIST'

)

BEGIN

DROP TABLE PHYSICIAN\_PBG\_LIST

END

CREATE TABLE PHYSICIAN\_PBG\_LIST (

PhysicianPBGListID int identity PRIMARY KEY,

PhysicianID int not null FOREIGN KEY REFERENCES PHYSICIANS(PhysicianID),

PBGID int not null FOREIGN KEY REFERENCES PBG(PBGID),

)

/\*TABLE 9

DROP AND CREATE IDN TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'IDN'

)

BEGIN

DROP TABLE IDN

END

CREATE TABLE IDN (

IDNID int identity PRIMARY KEY,

IDNName varchar(70) not null UNIQUE

)

/\*TABLE 10

DROP AND CREATE PHYSICIAN\_IDN\_LIST TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_IDN\_LIST'

)

BEGIN

DROP TABLE PHYSICIAN\_IDN\_LIST

END

CREATE TABLE PHYSICIAN\_IDN\_LIST (

PhysicianIDNListID int identity PRIMARY KEY,

PhysicianID int not null FOREIGN KEY REFERENCES PHYSICIANS (PhysicianID),

IDNID int not null FOREIGN KEY REFERENCES IDN(IDNID)

)

/\*TABLE 11

DROP AND CREATE SPECIALTY\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'SPECIALTY'

)

BEGIN

DROP TABLE SPECIALTY

END

CREATE TABLE SPECIALTY (

SpecialtyID int identity PRIMARY KEY,

SpecialtyName varchar(70) not null UNIQUE

)

/\*TABLE 12

DROP AND CREATE PHYSICIAN\_SPECIALTY TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_SPECIALTY'

)

BEGIN

DROP TABLE PHYSICIAN\_SPECIALTY

END

CREATE TABLE PHYSICIAN\_SPECIALTY (

PhysicianSpecialtyID int identity PRIMARY KEY,

PhysicianID int not null FOREIGN KEY REFERENCES PHYSICIANS(PhysicianID),

SpecialtyID int not null FOREIGN KEY REFERENCES SPECIALTY(SpecialtyID)

CONSTRAINT U1\_PHYSICIAN\_COT UNIQUE (PhysicianID, SpecialtyID)

)

/\*TABLE 13

DROP AND CREATE DATES TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'DATES'

)

BEGIN

DROP TABLE DATES

END

CREATE TABLE DATES (

DateID int identity PRIMARY KEY,

Months varchar(9) not null,

Years smallint not null,

CONSTRAINT U1\_DATES UNIQUE (Months, Years)

)

/\*TABLE 14

DROP AND CREATE PHYSICIAN\_DATES TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PHYSICIAN\_DATES'

)

BEGIN

DROP TABLE PHYSICIAN\_DATES

END

CREATE TABLE PHYSICIAN\_DATES (

PhysicianDatesID int identity PRIMARY KEY,

PhysicianId int not null FOREIGN KEY REFERENCES PHYSICIANS(PhysicianID),

DateID int not null FOREIGN KEY REFERENCES DATES(DateID)

)

/\*TABLE 15

DROP AND CREATE FACILITY\_DATES TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'FACILITY\_DATES'

)

BEGIN

DROP TABLE FACILITY\_DATES

END

CREATE TABLE FACILITY\_DATES (

FacilityDatesID int identity PRIMARY KEY,

FacilityID int not null FOREIGN KEY REFERENCES FACILITY(FacilityID),

DateID int not null FOREIGN KEY REFERENCES DATES(DateID)

)

/\*TABLE 16

DROP AND CREATE PBG\_DATES TABLE\*/

IF EXISTS (

SELECT \*

FROM INFORMATION\_SCHEMA.TABLES

WHERE TABLE\_NAME = 'PBG\_DATES'

)

BEGIN

DROP TABLE PBG\_DATES

END

CREATE TABLE PBG\_DATES (

PBGDatesID int identity PRIMARY KEY,

PBGID int not null FOREIGN KEY REFERENCES PBG(PBGID),

DateID int not null FOREIGN KEY REFERENCES DATES(DateID)

)

/\*CREATE VIEW - 1

Identify full list to indicate where physicians are practicing. Physicians can practice in multiple locations.

Due to this being test date, actual locations are not to be interpreted as realistic\*/

CREATE VIEW PHYSICIAN\_LOCATIONS AS

SELECT P.FirstName, P.LastName, F.FacilityName, A.Address1, A.Address2, A.City, A.ST, A.ZIP

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_FACILITY\_LIST AS PF ON P.PhysicianID = PF.PhysicianID

JOIN FACILITY AS F ON PF.FacilityID = F.FacilityID

JOIN FACILITY\_ADDRESS\_LIST AS FA ON F.FacilityID = FA.FacilityID

JOIN ADDRESS\_INFO AS A ON FA.AddressID = A.AddressID

GO

/\*CREATE VIEW - 2

Identify list of PBGs that physicians have been associated to\*/

CREATE VIEW PHYSICIAN\_PBG\_AFFILIATION AS

SELECT P.FirstName, P.LastName, P.Suffix, PBG.PBGName

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_PBG\_LIST AS PBL ON P.PhysicianID = PBL.PBGID

JOIN PBG ON PBL.PBGID = PBG.PBGID

GO

/\*CREATE VIEW - 3

Provides listing of when Physicians contracts were added and renewed\*/

CREATE VIEW PHYSICIAN\_ADDED AS

SELECT P.FirstName, P.LastName, P.Suffix, D.Months, D.Years

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_DATES AS PD ON P.PhysicianID = PD.PhysicianId

JOIN DATES AS D ON PD.DateID = D.DateID

ORDER BY P.LastName OFFSET 0 ROWS

GO

/\*CREATE FUNCTION - 1

Allows user to quickly identify the Physician ID, which can be useful when populating other tables

or for a WHERE clause\*/

CREATE FUNCTION dbo.PhysicianIDLookup(@FirstName varchar(70), @LastName varchar(70), @Suffix varchar(5))

RETURNS int AS

BEGIN

DECLARE @returnvalue int

SELECT @returnvalue = PhysicianID FROM PHYSICIANS

WHERE FirstName = @FirstName AND LastName = @LastName AND Suffix = @Suffix

RETURN @returnvalue

END

GO

/\*CREATE STORED PROCEDURE - 1

Instead of using Insert Into, create a procedure to end the information needed\*/

CREATE PROCEDURE AddPhysician (@FirstName varchar(50), @MiddleInitial char(1), @LastName varchar(50), @Suffix varchar(4))

AS

BEGIN

INSERT INTO PHYSICIANS

(FirstName, MiddleInitial, LastName, Suffix)

VALUES

(@FirstName, @MiddleInitial, @LastName, @Suffix)

END

GO

/\*INSERT INTO - 1\*/

INSERT INTO PHYSICIANS

(FirstName, MiddleInitial, LastName, Suffix)

VALUES

('Chris', 'M', 'Fandl', ''),

('Mike', 'C', 'Trescavage', ''),

('Lindsay', 'E', 'Holt', ''),

('Joshua', 'W', 'Laird', ''),

('Jennifer', '', 'Hufton', ''),

('Maynard', 'J', 'Keenan', ''),

('Jimmy', '', 'Page', ''),

('Robert', '', 'Plant', ''),

('Reggie', 'H', 'White', ''),

('Jerome', 'W', 'Brown', 'III')

GO

/\*INSERT INTO - 2\*/

INSERT INTO ADDRESS\_INFO

(Address1, Address2, City, ST, ZIP)

VALUES

('328 Elizabeth Drive', '', 'Kennett Square', 'PA', '19348'),

('808 Outter Space Blvd', '', 'Camden', 'NJ', '01342'),

('8 Plum Street', 'STE 1', 'Marcus Hook', 'GA', '68763'),

('28 Oragne Drive', '', 'Jackson', 'CO', '89173'),

('78 Brookline Road', '', 'Lateralus', 'CA', '90232'),

('156 Whispers Way', '', 'Deptford', 'MO', '56943'),

('1029 Walnut Lane', 'STE 2', 'Vicarious', 'MD', '27378'),

('67 Arch Street', '', 'Maryton', 'FL', '43143'),

('9256 Coastline Drive', '', 'Hilton Head', 'SC', '45221')

GO

/\*INSERT INTO - 3\*/

INSERT INTO PHYSICIAN\_ADDRESS\_LIST

(PhysicianID, AddressID)

VALUES

('1', '3'),

('2', '4'),

('2', '5'),

('2', '8'),

('3', '1'),

('4', '3'),

('4', '7'),

('4', '6'),

('5', '8'),

('5', '2'),

('6', '2'),

('6', '5'),

('7', '4'),

('7', '7'),

('7', '8'),

('8', '2'),

('9', '5'),

('9', '6'),

('9', '4'),

('10', '3'),

('10', '4'),

('11', '5'),

('11', '3'),

('11', '4'),

('11', '2'),

('12', '7'),

('12', '2'),

('13', '6'),

('13', '8'),

('14', '4')

GO

/\*INSERT INTO - 4\*/

INSERT INTO FACILITY

(FacilityName)

VALUES

('Saint Christopher''s of Camden'),

('First Response Care'),

('Children''s Health of Florida'),

('Family Awareness Practice'),

('Hopewell Family Practice'),

('Penn Medicine Peds'),

('Rothman Institute'),

('Careville Kids'),

('The Pediatric Center')

GO

/\*INSERT INTO - 5\*/

INSERT INTO PHYSICIAN\_FACILITY\_LIST

(PhysicianID, FacilityID)

VALUES

('4','7'),

('14','1'),

('13','8'),

('12','7'),

('13','3'),

('1','3'),

('12','1'),

('8','1'),

('2','6'),

('9','7'),

('3','1'),

('8','7'),

('3','6'),

('5','2'),

('2','3'),

('10','3'),

('1','1'),

('3','5'),

('11','3'),

('5','4'),

('14','7'),

('5','7'),

('11','7'),

('7','1'),

('6','1')

GO

/\*INSERT INTO - 6\*/

INSERT INTO FACILITY\_ADDRESS\_LIST

(FacilityID, AddressID)

VALUES

('1', '2'),

('2', '9'),

('2', '3'),

('3', '8'),

('4', '8'),

('5', '7'),

('6', '1'),

('7', '4'),

('7', '6'),

('8', '5'),

('9', '9')

GO

/\*INSERT INTO -7\*/

INSERT INTO PBG

(PBGName)

VALUES

('Catholic Health Partners'),

('CareSense'),

('Practice True'),

('Southern Hills'),

('EasternShore')

GO

/\*INSERT INTO - 8\*/

INSERT INTO PHYSICIAN\_PBG\_LIST

(PhysicianID, PBGID)

VALUES

('13','4'),

('2','2'),

('1','4'),

('2','3'),

('1','4'),

('9','4'),

('4','4'),

('11','2'),

('10','5'),

('14','3'),

('10','4'),

('7','1'),

('8','3'),

('10','5'),

('13','3'),

('3','1'),

('1','2'),

('5','3'),

('3','3'),

('1','5'),

('2','2'),

('7','2'),

('13','3'),

('12','2'),

('6','1'),

('5','1')

GO

/\*INSERT INTO - 9\*/

INSERT INTO IDN

(IDNName)

VALUES

('Lord''s Health System'),

('Vision Health'),

('Grandview Healthcare'),

('Einstein'),

('West Falls Healthcare'),

('Burbank Medicine')

GO

/\*INSERT INTO - 10\*/

INSERT INTO PHYSICIAN\_PBG\_LIST

(PhysicianID, PBGID)

VALUES

('1','3'),

('2','4'),

('2','5'),

('4','1'),

('5','2'),

('7','5'),

('8','1'),

('10','4'),

('10','3'),

('13','1'),

('14','2'),

('14','3')

GO

SELECT dbo.PhysicianIDLookup ('William', 'Holt', 'III')

/\*Added Suffix to Physicians Table\*/

ALTER TABLE PHYSICIANS

ADD Suffix varchar(4);

/\*Additional physicians added\*/

INSERT INTO PHYSICIANS

(FirstName, MiddleInitial, LastName)

VALUES

('William', 'F', 'Holt'),

('Nicholas', '', 'Woody')

INSERT INTO PHYSICIAN\_IDN\_LIST

(PhysicianID, IDNID)

VALUES

('3','2'),

('5','4'),

('6','3'),

('9','1'),

('9','6'),

('11','5'),

('12','5'),

('13','1'),

('13','5'),

('14','6')

SELECT \* FROM PHYSICIAN\_ADDRESS\_LIST

SELECT \* FROM FACILITY\_PBG\_LIST

SELECT \* FROM FACILITY\_IDN\_LIST

SELECT \* FROM FACILITY\_GPO\_LIST

SELECT \* FROM PHYSICIAN\_GPO\_LIST

SELECT \* FROM GPO

SELECT \*

FROM PHYSICIAN\_DATES

SELECT \*

FROM DATES

ORDER BY DATEID

INSERT INTO DATES

(Months, Years)

VALUES

('December','2015'),

('January','2016'),

('Febaruary','2016'),

('March','2016'),

('April','2016'),

('May','2016'),

('June','2016'),

('July','2016'),

('August','2016'),

('September','2016'),

('Ocotober','2016'),

('November','2016'),

('December','2016'),

('January','2017'),

('Febaruary','2017'),

('March','2017'),

('April','2017'),

('May','2017'),

('June','2017'),

('July','2017'),

('August','2017'),

('September','2017'),

('Ocotober','2017'),

('November','2017'),

('December','2017'),

('January','2018'),

('Febaruary','2018'),

('March','2018'),

('April','2018'),

('May','2018'),

('June','2018'),

('July','2018'),

('August','2018'),

('September','2018'),

('Ocotober','2018'),

('November','2018'),

('December','2018')

INSERT INTO PHYSICIAN\_DATES

(PhysicianId, DateID)

VALUES

('1','1'),

('2','1'),

('3','3'),

('4','3'),

('5','4'),

('6','4'),

('1','4'),

('2','4'),

('7','5'),

('8','5'),

('3','6'),

('4','6'),

('2','7'),

('3','8'),

('9','10'),

('5','10'),

('10','12'),

('11','12'),

('1','13'),

('8','13'),

('9','14'),

('10','15'),

('11','15'),

('6','16'),

('8','19'),

('9','19'),

('4','20'),

('12','21'),

('13','22'),

('14','22')

/\*CREATE VIEW OF DATES WHEN PHYSICIANS ADDED\*/

SELECT P.FirstName, P.LastName, D.Months, D.Years

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_DATES AS PD ON P.PhysicianID = PD.PhysicianId

JOIN DATES AS D ON PD.DateID = D.DateID

SELECT \*

FROM PBG

INSERT INTO PBG\_DATES

(PBGID, DateID)

VALUES

('1','3'),

('2', '4'),

('3','1'),

('4', '20'),

('5','10')

CREATE TABLE IDN\_DATES (

IDNDatesID int identity PRIMARY KEY,

IDNID int not null FOREIGN KEY REFERENCES IDN(IDNID),

DateID int not null FOREIGN KEY REFERENCES DATES(DateID)

)

SELECT P.FirstName, P.LastName, D.Months, D.Years

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_DATES AS PD ON P.PhysicianID = PD.PhysicianId

JOIN DATES AS D ON PD.DateID = D.DateID

/\*CREATE VIEW OF PHYSICIAN AND IDN NAMES\*/

SELECT P.FirstName, P.LastName, IDN.IDNName

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_IDN\_LIST AS PIL ON P.PhysicianID = PIL.PhysicianID

JOIN IDN ON PIL.IDNID = IDN.IDNID

SELECT \*

FROM IDN

INSERT INTO IDN\_DATES

(IDNID, DateID)

VALUES

('1','10'),

('2', '1'),

('3','4'),

('4', '4'),

('5','12'),

('6','10')

SELECT \*

FROM FACILITY\_DATES

SELECT P.FirstName, P.LastName, F.FacilityName

FROM PHYSICIANS AS P

JOIN PHYSICIAN\_FACILITY\_LIST AS PFL ON P.PhysicianID = PFL.PhysicianID

JOIN FACILITY AS F ON PFL.FacilityID = F.FacilityID

SELECT \*

FROM FACILITY

INSERT INTO FACILITY\_DATES

(FacilityID, DateID)

VALUES

('1','1'),

('2','4'),

('3','3'),

('4','4'),

('5','1'),

('6','1'),

('7','1'),

('8','10')

INSERT INTO CLASS\_OF\_TRADE

(COTName)

VALUES

('Family Medicine'),

('Pediatrics'),

('Oncology'),

('Home Health'),

('Med Surg'),

('Cardiology')

EXEC sp\_rename 'CLASS\_OF\_TRADE', 'SPECIALTY'

DROP TABLE FACILITY\_COT

EXEC sp\_rename 'PHYSICIAN\_COT', 'PHYSICIAN\_SPECIALTY'

EXEC sp\_RENAME 'PHYSICIAN\_SPECIALTY.COTID' , 'SpecialtyID', 'COLUMN'

EXEC sp\_RENAME 'PHYSICIAN\_SPECIALTY.PhysicianCOTID' , 'PhysicianSpecialtyID', 'COLUMN'

SELECT \*

FROM PHYSICIAN\_SPECIALTY

DROP TABLE PHYSICIAN\_SPECIALTY

SELECT \*

FROM SPECIALTY

SELECT \*

FROM PHYSICIANS

INSERT INTO PHYSICIAN\_SPECIALTY

(PhysicianID, SpecialtyID)

VALUES

('1','1'),

('2','2'),

('3','2'),

('4','2'),

('4','3'),

('5','2'),

('6','2'),

('7','6'),

('8','1'),

('9','4'),

('10','2'),

('10','5'),

('11','1'),

('11','2'),

('12','4'),

('13','5'),

('14','2')

SELECT \*

FROM PHYSICIAN\_DATES

SELECT \*

FROM PHYSICIAN\_DATES