

CIT230 FINAL DATABASE PROJECT

Your Name:	Samantha Lavrinc
Course:	CIT230 – Spring 2020
Professor:	Nancy S. Grant, Ed.D.
Completed:	04/25/2020

NARRATIVE OF THE PURPOSE OF THE CREATED DATABASE

This database was created using Microsoft SQL Management Studio 2017.

Database Name: Lavrinc_Healthcare.mdf

Tables:

- dbo.PATIENT_INFO
- dbo.PROC_INFO
- dbo.PHYS_INFO
- dbo.INS_BRIDGE
- dbo.INS_INFO

The Lavrinc_Healthcare database was created to allow medical billing facilities and departments the ability to access multiple patient accounts in correlation with their medical history and insurance information to support the billing process within a timely manner. Most health insurance companies will deny claims that haven't been received within a 60-120 day timeline, resulting in a profit loss for physicians and healthcare groups. To promote efficiency and increase profits, it's crucial that healthcare providers utilize a database that consists of the correct data to include on a HCFA 1500 form, including Patient Information, Insurance Information, Physician Information and Procedure Information.

Insurance billing standards change annually. While this database includes the most basic data sets required to process services rendered, it also enables us to later add additional requirements if these standards change based on any insurance billing updates.

The biggest feature of this database is that it allows detection of issues that may arise by pulling patient accounts with similar parameters to rebill or adjust coding on accounts that meet these parameters, rather than comb through individual patient accounts one-by-one. As an example, if a specific insurance company changes their claim approval to require an additional pre-existing condition ICD-10 code on a certain procedure, we will be able to pull records of every patient with that specific insurance and ICD-10 service code and make these appropriate adjustments all at once.

A LISTING OF ALL OF THE DATABASE TABLE STRUCTURES

```
USE Lavrinc_Healthcare
GO
```

Table: INS_INFO

Contents: General insurance company contact information.

The primary key, Insurance_Code, is a 4-digit value assigned to each insurance company that can be easily accessed numerous times with the INS_BRIDGE table.

base type nvarchar is utilized in the Insurance Name, Address lines and City to account for the possibility that names or addresses have accents and/or need access to extended Unicode characters.

No constraints added to zip as standard rules suggest because Insurance provider may not be located within the United States, and requiring integers may prevent ability for claims to be filed to countries which utilize characters in their postal codes.

```
CREATE TABLE INS_INFO(
    Insurance_Code          Int      NOT NULL IDENTITY(1000,1) PRIMARY KEY,
    Ins_Name                nvarchar(50)      NOT NULL,
    Ins_add1                nvarchar(100)     NOT NULL,
    Ins_add2                nvarchar(100)     NULL,
    Ins_city                nvarchar(40)     NOT NULL,
    Ins_state               char(2)          NOT NULL,
    Ins_zip                 char(10)         NOT NULL,
    Ins_phone               varchar(15)      NOT NULL, /* Max possible
phone number digits in the world. */
);
```

Table: PATIENT_INFO

Contents: Patient contact information.

The primary key, Patient_ID, is a unique 6-digit integer assigned to each patient that can be linked to different procedures via PROC_INFO and the INS_BRIDGE.

base type nvarchar is utilized in the Patient Name, Address lines and City to account for the possibility that names or addresses have accents and/or need access to extended Unicode characters.

No constraints added to zip as standard rules suggest because Patient may not reside within the United States, and requiring integers may prevent ability for claims to be filed to countries which utilize characters in their postal codes.

```
CREATE TABLE PATIENT_INFO(
    Patient_ID              Int      NOT NULL IDENTITY(100000,1) PRIMARY KEY,
```

```

Pt_Last_Name          nvarchar(30)          NOT NULL,
Pt_First_Name         nvarchar(30)          NOT NULL,
Pt_Middle_Name        nvarchar(30)          NULL,
DOB                  date                   NOT NULL,
Pt_Add1              nvarchar(100)         NOT NULL,
Pt_Add2              nvarchar(100)         NULL,
Pt_City              nvarchar(40)         NOT NULL,
Pt_State             char(2)              NOT NULL,
Pt_Zip              char(10)             NOT NULL,
Pt_Phone            char(15)             NOT NULL, /* Max possible
phone number digits in the world. */

);

```

Table: INS_BRIDGE

Contents: Primary, Secondary and Tertiary Insurance information for each Patient.

Constraints include multiple foreign keys which allows this table to work as a bridge and populate insurance contact information within a filed claim.

base type nvarchar is only utilized in columns where the patient policy number may contain unknown characters.

```

CREATE TABLE INS_BRIDGE(
    Patient_ID          Int                NOT NULL,
    Prim_Code           Int                NOT NULL,
    Prim_Pt_ID         nvarchar(30)       NULL,
    Sec_Code            Int                NULL,
    Sec_Pt_ID          nvarchar(30)       NULL,
    Ter_Code            Int                NULL,
    Ter_PT_ID           nvarchar(30)       NULL,

    CONSTRAINT FK_INS_BRIDGE_PATIENT_INFO FOREIGN KEY(Patient_ID)
    REFERENCES PATIENT_INFO(Patient_ID),

    CONSTRAINT FK_INS_BRIDGE_INS_INFO FOREIGN KEY(Prim_Code)
    REFERENCES INS_INFO (Insurance_Code)
    ON UPDATE CASCADE
    ON DELETE CASCADE,

    CONSTRAINT FK_INS_BRIDGE_INS_INFO2 FOREIGN KEY(Sec_Code)
    REFERENCES INS_INFO (Insurance_Code),

    CONSTRAINT FK_INS_BRIDGE_INS_INFO3 FOREIGN KEY(Ter_Code)
    REFERENCES INS_INFO (Insurance_Code)

);

```

Table: PHYS_INFO

Contents: Physician and practice information.

Phys_ID is the Primary Key, with a unique 5 digit code assigned to each service provider.

base type nvarchar is only utilized in the columns that may contain extended characters such as names or addresses.

After working with the data used in the PHYS_INFO table, it became evident that the State and Physician First Name column was missing, please reference Other Database Project Information for additional commentary and UPDATE information.

```
CREATE TABLE PHYS_INFO(  
    Phys_ID          Int          NOT NULL IDENTITY(10000,1) PRIMARY KEY,  
    Phys_Name        nvarchar(20) NOT NULL,  
    Practice_add1    nvarchar(100) NOT NULL,  
    Practice_add2    nvarchar(100) NULL,  
    Practice_city    nvarchar(30) NOT NULL,  
    Practice_zip     char(10)     NOT NULL,  
    Practice_phone   char(10)     NOT NULL,  
  
);
```

Table: PROC_INFO

Contents: Information on the procedure and date of service.

Constraints include foreign keys Patient_ID and Phys_ID to link the procedure performed by specific physicians to the services received by the specific patients.

```
CREATE TABLE PROC_INFO(  
    Patient_ID       Int          NOT NULL,  
    Proc_ID          varchar(20)  NOT NULL,  
    ICD10_Code       varchar(20)  NOT NULL,  
    DOS              date         NOT NULL,  
    Phys_ID          Int          NOT NULL,  
  
    CONSTRAINT FK_PROC_INFO_PATIENT_INFO FOREIGN KEY(Patient_ID)  
        REFERENCES PATIENT_INFO(Patient_ID),  
  
    CONSTRAINT FK_PROC_INFO_PHYS_INFO FOREIGN KEY(Phys_ID)  
        REFERENCES PHYS_INFO(Phys_ID)  
        ON UPDATE CASCADE  
        ON DELETE CASCADE  
  
);
```

A LISTING OF THE DATABASE DIAGRAM

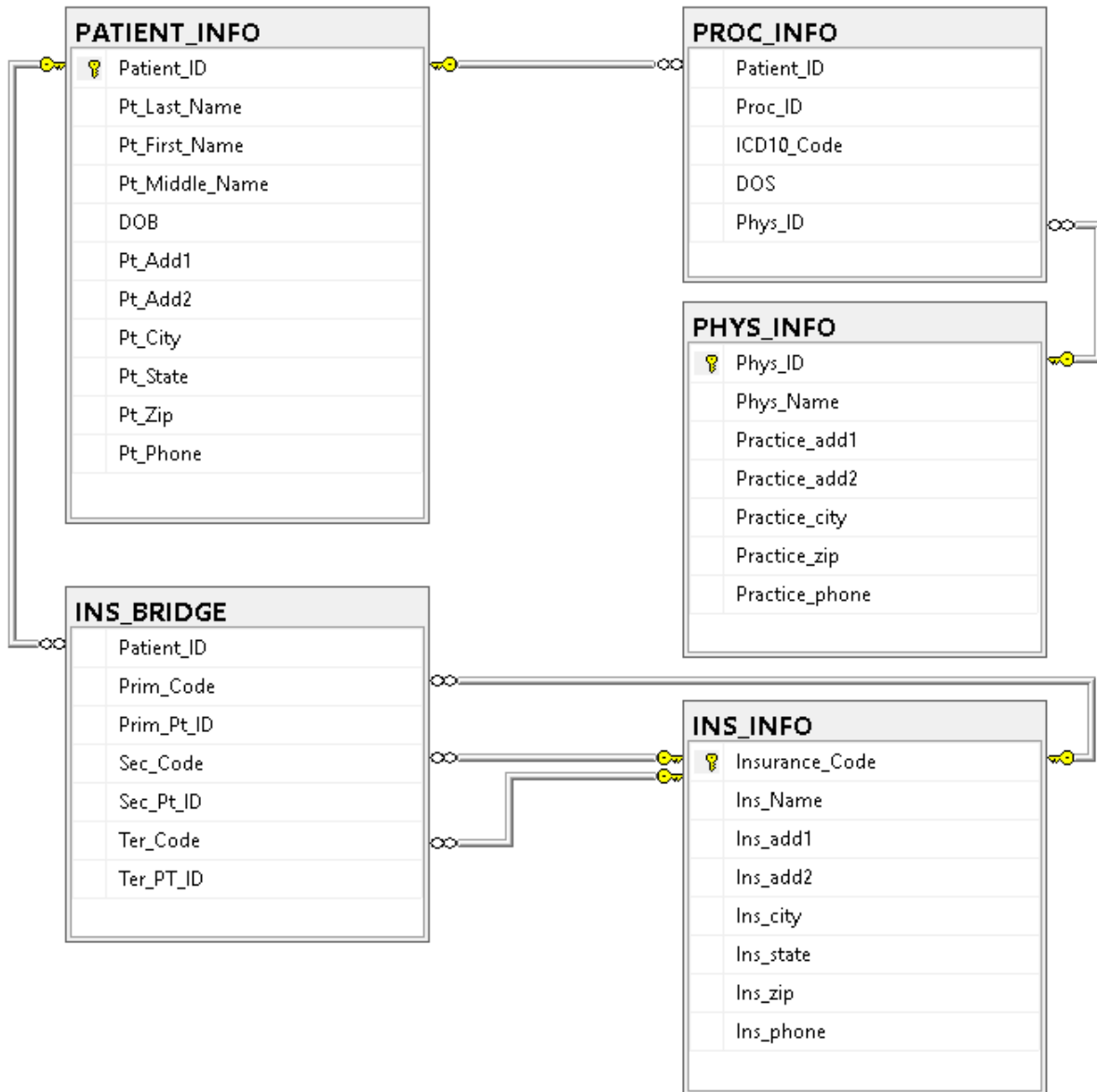


TABLE RELATIONSHIPS

PATIENT_INFO to PROC_INFO – Each patient can have multiple procedures, but only one procedure on the specific date of service can apply to each patient, so this is a one to many relationship. (1:N)

PROC_INFO to PHYS_INFO – Each physician can perform multiple procedures, but each procedure will only have one primary physician, so this is a one to many relationship. (1:N)

PATIENT INFO to INS_BRIDGE – Each patient is assigned to one insurance bridge and each insurance bridge is assigned to one patient, so this is a one to one relationship. (1:1)

INS_BRIDGE to INS_INFO – Each instance of INS_BRIDGE assigned per patient can have multiple references to INS_INFO, such as if a patient were to have both primary and secondary insurance with the same provider, and each instance of INS_INFO can be applied to multiple instances of INS_BRIDGE, therefore this relationship is many to many. (N:M)

A LISTING OF ALL OF THE DATA CONTAINED IN EACH TABLE

```
USE Lavrinc_Healthcare
GO
```

```
/* Populating tables that have a Primary Key and no Foreign Keys.
Tables: INS_INFO, PATIENT_INFO, PHYS_INFO */
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'UPMC Health Plan', 'PO Box 2999', NULL,
'Pittsburgh', 'PA', '15230-2999', '844 220 4785')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'UPMC for Life MC', 'PO Box 2997', NULL,
'Pittsburgh', 'PA', '15230-2999', '844 220 4785')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'UPMC for You MA', 'PO Box 2995', NULL,
'Pittsburgh', 'PA', '15230-2999', '844 220 4785')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'UPMC Community HealthChoices', 'PO Box
106042', NULL, 'Pittsburgh', 'PA', '15230-2999', '844 220 4785')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'Keystone Blue', 'Claims Processing', 'PO
Box 898819', 'Camp Hill', 'PA', '17089-8819', '412 544 7000')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'Highmark BCBS', 'Claims Processing', 'PO
Box 890062', 'Camp Hill', 'PA', '17089-0062', '800 241 5704')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'Medicare Part B', 'Novitas Solutions',
'PO Box 3418', 'Mechanicsburg', 'PA', '17055-1854', '877 235 8073')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'Medicaid', 'Claims Processing', 'PO Box
8042', 'Harrisburg', 'PA', '17105', '800 537 8862')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'Aetna', 'Claims Processing', 'PO Box
981106', 'El Paso', 'TX', '79998-1106', '800 784 3991')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'United Healthcare', 'PO Box 30555',
NULL, 'Salt Lake City', 'UT', '84130-0555', '866 414 1959')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'Kaiser Permanente', 'Claims Processing',
'PO Box 373150', 'Denver', 'CO', '80237-6970', '800 632 9700')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'Humana', 'PO Box 14601', NULL,
'Lexington', 'KY', '40512', '800 457 4708')
```

```
INSERT INTO INS_INFO VALUES (/*Insurance_Code,*/'Cigna', 'PO Box 182223', NULL,
'Chattanooga', 'TN', '37422-7223', '800 244 6224')
```


SELECT *
FROM INS_INFO;

100 %

Results Messages

	Insurance_Code	Ins_Name	Ins_add1	Ins_add2	Ins_city	Ins_state	Ins_zip	Ins_phone
1	1000	UPMC Health Plan	PO Box 2999	NULL	Pittsburgh	PA	15230-2999	844 220 4785
2	1001	UPMC for Life MC	PO Box 2997	NULL	Pittsburgh	PA	15230-2999	844 220 4785
3	1002	UPMC for You MA	PO Box 2995	NULL	Pittsburgh	PA	15230-2999	844 220 4785
4	1003	UPMC Community HealthChoices	PO Box 106042	NULL	Pittsburgh	PA	15230-2999	844 220 4785
5	1004	Keystone Blue	Claims Processing	PO Box 898819	Camp Hill	PA	17089-8819	412 544 7000
6	1005	Highmark BCBS	Claims Processing	PO Box 890062	Camp Hill	PA	17089-0062	800 241 5704
7	1006	Medicare Part B	Novitas Solutions	PO Box 3418	Mechanicsburg	PA	17055-1854	877 235 8073
8	1007	Medicaid	Claims Processing	PO Box 8042	Harrisburg	PA	17105	800 537 8862
9	1008	Aetna	Claims Processing	PO Box 981106	El Paso	TX	79998-1106	800 784 3991
10	1009	United Healthcare	PO Box 30555	NULL	Salt Lake City	UT	84130-0555	866 414 1959
11	1010	Kaiser Permanente	Claims Processing	PO Box 373150	Denver	CO	80237-6970	800 632 9700
12	1011	Humana	PO Box 14601	NULL	Lexington	KY	40512	800 457 4708
13	1012	Cigna	PO Box 182223	NULL	Chattanooga	TN	37422-7223	800 244 6224

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Jones', 'James', 'R', '1964-08-16', '123
Rose Lane', NULL, 'Pittsburgh', 'PA', '15227', '555 638 9576')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Robinson', 'Robert', 'M', '1982-04-10',
'487 Roady Road', 'Apt #6', 'Pittsburgh', 'PA', '15237', '555 544 8764')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Miller', 'Michael', 'K', '1973-01-27',
'8762 Ventura Street', NULL, 'Pittsburgh', 'PA', '15005', '555 876 8742')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Tabb', 'Candice', 'Diana', '1960-03-31',
'2188 Wetzel Lane', NULL, 'Grand Rapids', 'MI', '49503', '555 225 0915')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Preston', 'Melanie', 'M', '1959-12-31',
'3783 Coventry Court', NULL, 'Pass Christian', 'MS', '39571', '555 255 1656')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Elliot', 'Rodney', 'Jacob', '1974-09-
04', '4741 Mayo Street', NULL, 'Winchester', 'KY', '40391', '555 691 2471')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Manning', 'Leah', 'J', '1972-03-28',
'1273 Eagles Nest Drive', NULL, 'Oroville', 'CA', '95966', '555 589 7373')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Hendrickson', 'Kelli', 'A', '1961-05-
15', '544 Tibbs Avenue', NULL, 'Wise River', 'MT', '59762', '555 832 7796')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'McDonnell', 'Jonathan', 'B', '1982-07-
26', '684 Parkview Drive', NULL, 'Anaheim', 'CA', '92801', '555 249 5242')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Kent', 'Kenneth', 'J', '1976-01-09',
'4239 Broadcast Drive', NULL, 'Clover', 'NC', '29710', '555 228 9949')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Rogers', 'Edna', 'O', '1943-08-26',
'4984 Drainer Avenue', '#5b', 'Pensacola', 'FL', '32514', '555 602 7660')
```

```
INSERT INTO PATIENT_INFO VALUES (*Patient_ID,*/'Mendoza', 'Charles', 'Alan', '1966-10-
07', '2869 47th Avenue', NULL, 'Newbrook', 'AB', 'T0A 2P0', '555 858 571')
```

```
INSERT INTO PATIENT_INFO VALUES (/*Patient_ID,*/'Arteaga', 'Etel', 'Garrido', '1980-05-06', 'Pl. Virgen Blanca, 14', NULL, 'Les Franqueses del Vallès', '34', '08520', '555 114 065')
```

SELECT *
FROM PATIENT_INFO;

100 %

Results Messages

	Patient_ID	Pt_Last_Name	Pt_First_Name	Pt_Middle_Name	DOB	Pt_Add1	Pt_Add2	Pt_City	Pt_State	Pt_Zip	Pt_Phone
1	100000	Jones	James	R	1964-08-16	123 Rose Lane	NULL	Pittsburgh	PA	15227	555 638 9576
2	100001	Robinson	Robert	M	1982-04-10	487 Roady Road	Apt #6	Pittsburgh	PA	15237	555 544 8764
3	100002	Miller	Michael	K	1973-01-27	8762 Ventura Street	NULL	Pittsburgh	PA	15005	555 876 8742
4	100003	Tabb	Candice	Diana	1960-03-31	2188 Wetzel Lane	NULL	Grand Rapids	MI	49503	555 225 0915
5	100004	Preston	Melanie	M	1959-12-31	3783 Coventry Court	NULL	Pass Christian	MS	39571	555 255 1656
6	100005	Elliot	Rodney	Jacob	1974-09-04	4741 Mayo Street	NULL	Winchester	KY	40391	555 691 2471
7	100006	Manning	Leah	J	1972-03-28	1273 Eagles Nest Drive	NULL	Oroville	CA	95966	555 589 7373
8	100007	Hendrickson	Kelli	A	1961-05-15	544 Tibbs Avenue	NULL	Wise River	MT	59762	555 832 7796
9	100008	McDonnell	Jonathan	B	1982-07-26	684 Parkview Drive	NULL	Anaheim	CA	92801	555 249 5242
10	100009	Kent	Kenneth	J	1976-01-09	4239 Broadcast Drive	NULL	Clover	NC	29710	555 228 9949
11	100010	Rogers	Edna	O	1943-08-26	4984 Drainer Avenue	#5b	Pensacola	FL	32514	555 602 7660
12	100011	Mendoza	Charles	Alan	1966-10-07	2869 47th Avenue	NULL	Newbrook	AB	TQA 2P0	555 858 571
13	100012	Arteaga	Etel	Garrido	1980-05-06	Pl. Virgen Blanca, 14	NULL	Les Franqueses del Vallès	34	08520	555 114 065

```
INSERT INTO PHYS_INFO VALUES (/*Phys_ID,*/'Otero', '1882 Hummingbird Ct', 'Suite 5', 'Pittsburgh', '15220', '5558654251', 'Olivia', 'PA')
```

```
INSERT INTO PHYS_INFO VALUES (/*Phys_ID,*/'Timmons', '2142 Maryland Avenue', '# 1682', 'Pittsburgh', '15846', '5558423751', 'John', 'PA')
```

```
INSERT INTO PHYS_INFO VALUES (/*Phys_ID,*/'Taylor', '297 Bloomfield Way', NULL, 'Wexford', '15116', '5558423751', 'Christine', 'PA')
```

```
INSERT INTO PHYS_INFO VALUES (/*Phys_ID,*/'Young', '1566 Ashwood Drive', NULL, 'Cranberry', '15222', '5556422153', 'Andrew', 'PA')
```

```
INSERT INTO PHYS_INFO VALUES (/*Phys_ID,*/'Espinoza', '3279 Christie Way', NULL, 'Westborough', '01581', '5558431267', 'George', 'MA')
```

SELECT *
FROM PHYS_INFO;

100 %

Results Messages

	Phys_ID	Phys_Name	Practice_add1	Practice_add2	Practice_city	Practice_zip	Practice_phone	Phys_FirstName	Practice_State
1	10005	Otero	1882 Hummingbird Ct	Suite 5	Pittsburgh	15220	5558654251	Olivia	PA
2	10006	Timmons	2142 Maryland Avenue	# 1682	Pittsburgh	15846	5558423751	John	PA
3	10007	Taylor	297 Bloomfield Way	NULL	Wexford	15116	5558423751	Christine	PA
4	10008	Young	1566 Ashwood Drive	NULL	Cranberry	15222	5556422153	Andrew	PA
5	10009	Espinoza	3279 Christie Way	NULL	Westborough	01581	5558431267	George	MA

/* Populating tables that have a Foreign keys.
Tables: INS_BRIDGE, PROC_INFO */

```
INSERT INTO INS_BRIDGE VALUES ('100000', '1000', '3215846921-32', '1007', '74618A-1692', NULL, NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100001', '1012', '5884312-86', '1010', '74618A-1692',
'1006', '321SDG-1432')
```

```
INSERT INTO INS_BRIDGE VALUES ('100002', '1000', '3212185343-77', '1008', '481DFH', NULL,
NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100003', '1007', 'U 1694852', NULL, NULL, NULL, NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100004', '1004', '65482641-D', '1007', 'PA 1623482667',
NULL, NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100005', '1009', 'U 1354268165473', NULL, NULL, NULL,
NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100006', '1001', '245DFV', '1009', 'U 123456789101',
NULL, NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100007', '1002', 'PA 1632 BRGR', '1011', 'H16-B3A6',
NULL, NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100008', '1010', 'EW-13-16-22-GG', NULL, NULL, NULL,
NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100009', '1007', '15248A-3211', NULL, NULL, NULL, NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100010', '1004', '3456-328-1348', '1007', '13548D-8432',
NULL, NULL)
```

```
INSERT INTO INS_BRIDGE VALUES ('100011', '1005', '48265-42887', '1009', 'U 124789512644',
'1006', '384TFH-1844')
```

```
INSERT INTO INS_BRIDGE VALUES ('100012', '1003', '53485 3546', '1006', '482LYG-018',
NULL, NULL)
```

SELECT *							
FROM INS_BRIDGE;							
100 %							
Results Messages							
	Patient_ID	Prim_Code	Prim_Pt_ID	Sec_Code	Sec_Pt_ID	Ter_Code	Ter_PT_ID
1	100000	1000	3215846921-32	1007	74618A-1692	NULL	NULL
2	100001	1012	5884312-86	1010	74618A-1692	1006	321SDG-1432
3	100002	1000	3212185343-77	1008	481DFH	NULL	NULL
4	100003	1007	U 1694852	NULL	NULL	NULL	NULL
5	100004	1004	65482641-D	1007	PA 1623482667	NULL	NULL
6	100005	1009	U 1354268165473	NULL	NULL	NULL	NULL
7	100006	1001	245DFV	1009	U 123456789101	NULL	NULL
8	100007	1002	PA 1632 BRGR	1011	H16-B3A6	NULL	NULL
9	100008	1010	EW-13-16-22-GG	NULL	NULL	NULL	NULL
10	100009	1007	15248A-3211	NULL	NULL	NULL	NULL
11	100010	1004	3456-328-1348	1007	13548D-8432	NULL	NULL
12	100011	1005	48265-42887	1009	U 124789512644	1006	384TFH-1844
13	100012	1003	53485 3546	1006	482LYG-018	NULL	NULL

```
INSERT INTO PROC_INFO VALUES ('100001', 'S06.890A', '2018-05-10', '10005')
```

```

INSERT INTO PROC_INFO VALUES ('100001', 'S14.109A', '2018-05-10', '10005')
INSERT INTO PROC_INFO VALUES ('100002', '4A1204Z', '2019-07-22', '10006')
INSERT INTO PROC_INFO VALUES ('100002', '2W50X0Z', '2019-08-08', '10007')
INSERT INTO PROC_INFO VALUES ('100002', 'HZ30ZZZ', '2020-02-14', '10008')
INSERT INTO PROC_INFO VALUES ('100003', '0JH638Z', '2019-06-26', '10008')
INSERT INTO PROC_INFO VALUES ('100003', 'Z12.11', '2020-01-29', '10009')
INSERT INTO PROC_INFO VALUES ('100003', '2W50X0Z', '2020-03-10', '10005')
INSERT INTO PROC_INFO VALUES ('100004', 'S06.890A', '2020-01-10', '10006')
INSERT INTO PROC_INFO VALUES ('100004', 'HZ30ZZZ', '2020-02-01', '10005')
INSERT INTO PROC_INFO VALUES ('100005', 'S14.109A', '2019-11-26', '10009')
INSERT INTO PROC_INFO VALUES ('100006', '4A1204Z', '2019-11-21', '10007')
INSERT INTO PROC_INFO VALUES ('100006', 'Z12.11', '2018-05-01', '10005')
INSERT INTO PROC_INFO VALUES ('100006', 'S14.109A', '2019-07-06', '10009')
INSERT INTO PROC_INFO VALUES ('100007', '2W50X0Z', '2018-12-02', '10006')
INSERT INTO PROC_INFO VALUES ('100007', 'HZ30ZZZ', '2019-03-15', '10008')
INSERT INTO PROC_INFO VALUES ('100008', 'S06.890A', '2019-07-06', '10005')
INSERT INTO PROC_INFO VALUES ('100009', '0JH638Z', '2019-01-20', '10009')
INSERT INTO PROC_INFO VALUES ('100010', 'S06.890A', '2019-01-20', '10007')
INSERT INTO PROC_INFO VALUES ('100010', '0Z12.11', '2020-02-16', '10005')
INSERT INTO PROC_INFO VALUES ('100010', '2W50X0Z', '2019-10-01', '10006')
INSERT INTO PROC_INFO VALUES ('100011', 'S14.109A', '2019-07-13', '10006')
INSERT INTO PROC_INFO VALUES ('100011', 'S06.890A', '2019-07-13', '10006')
INSERT INTO PROC_INFO VALUES ('100012', '4A1204Z', '2018-07-19', '10009')
INSERT INTO PROC_INFO VALUES ('100012', 'S06.890A', '2018-12-01', '10008')
INSERT INTO PROC_INFO VALUES ('100012', '2W50X0Z', '2019-02-13', '10007')
INSERT INTO PROC_INFO VALUES ('100012', 'Z12.11', '2019-04-08', '10006')
INSERT INTO PROC_INFO VALUES ('100012', 'HZ30ZZZ', '2019-09-04', '10005')
INSERT INTO PROC_INFO VALUES ('100012', '0JH638Z', '2020-04-22', '10007')

```

<div> <div>SELECT *</div> <div>FROM PROC_INFO;</div> </div>				
100 %				
<div> <div>Results</div> <div>Messages</div> </div>				
	Patient_ID	ICD10_Code	DOS	Phys_ID
1	100001	S06.890A	2018-05-10	10005
2	100001	S14.109A	2018-05-10	10005
3	100002	4A1204Z	2019-07-22	10006
4	100002	2w50x0Z	2019-08-08	10007
5	100002	HZ30ZZZ	2020-02-14	10008
6	100003	QJH638Z	2019-06-26	10008
7	100003	Z12.11	2020-01-29	10009
8	100003	2w50x0Z	2020-03-10	10005
9	100004	S06.890A	2020-01-10	10006
10	100004	HZ30ZZZ	2020-02-01	10005
11	100005	S14.109A	2019-11-26	10009
12	100006	4A1204Z	2019-11-21	10007
13	100006	Z12.11	2018-05-01	10005
14	100006	S14.109A	2019-07-06	10009
15	100007	2w50x0Z	2018-12-02	10006
16	100007	HZ30ZZZ	2019-03-15	10008
17	100008	S06.890A	2019-07-06	10005
18	100009	QJH638Z	2019-01-20	10009
19	100010	S06.890A	2019-01-20	10007
20	100010	QZ12.11	2020-02-16	10005
21	100010	2w50x0Z	2019-10-01	10006
22	100011	S14.109A	2019-07-13	10006
23	100011	S06.890A	2019-07-13	10006
24	100012	4A1204Z	2018-07-19	10009
25	100012	S06.890A	2018-12-01	10008
26	100012	2w50x0Z	2019-02-13	10007
27	100012	Z12.11	2019-04-08	10006
28	100012	HZ30ZZZ	2019-09-04	10005
29	100012	QJH638Z	2020-04-22	10007

LISTING OF THREE SQL SELECT STATEMENTS FROM TWO OR MORE OF THE TABLES CONTAINED IN YOUR DATABASE

Prompt: UPMC has denied all claims due to an unrelated credentialing issue, all claims for each procedure going to a UPMC insurance product must be accessed to refile the claims.

UPMC Insurance_Codes range from 1000-1003

```
SELECT PATIENT_INFO.Patient_ID AS "Patient ID",
       trim(Pt_First_Name) + ' ' + trim(Pt_Last_Name) AS "Patient Name",
       Prim_Code AS "Primary",
       ISNULL(Sec_Code, '') AS "Secondary",
       ISNULL(Ter_Code, '') AS "Tertiary",
       DOS AS "Date of Service", ICD10_Code AS "Procedure Code"
FROM PATIENT_INFO
JOIN INS_BRIDGE ON PATIENT_INFO.Patient_ID = INS_BRIDGE.Patient_ID
JOIN PROC_INFO ON PATIENT_INFO.Patient_ID = PROC_INFO.Patient_ID
WHERE Prim_Code BETWEEN 1000 AND 1003 OR
       Sec_Code BETWEEN 1000 AND 1003 OR
       Ter_Code BETWEEN 1000 AND 1003

ORDER BY PATIENT_INFO.Patient_ID
```

/*Prompt: UPMC has denied all claims due to an unrelated credentialing issue, all claims for each procedure going to a UPMC insurance product must be accessed to refile the claims.

UPMC Insurance_Codes range from 1000-1003*/

```
SELECT PATIENT_INFO.Patient_ID AS "Patient ID", |
       trim(Pt_First_Name) + ' ' + trim(Pt_Last_Name) AS "Patient Name",
       Prim_Code AS "Primary",
       ISNULL(Sec_Code, '') AS "Secondary",
       ISNULL(Ter_Code, '') AS "Tertiary",
       DOS AS "Date of Service", ICD10_Code AS "Procedure Code"
FROM PATIENT_INFO
JOIN INS_BRIDGE ON PATIENT_INFO.Patient_ID = INS_BRIDGE.Patient_ID
JOIN PROC_INFO ON PATIENT_INFO.Patient_ID = PROC_INFO.Patient_ID
WHERE Prim_Code BETWEEN 1000 AND 1003 OR
       Sec_Code BETWEEN 1000 AND 1003 OR
       Ter_Code BETWEEN 1000 AND 1003

ORDER BY PATIENT_INFO.Patient_ID
```

100 %

Results Messages

	Patient ID	Patient Name	Primary	Secondary	Tertiary	Date of Service	Procedure Code
1	100002	Michael Miller	1000	1008	0	2019-07-22	4A1204Z
2	100002	Michael Miller	1000	1008	0	2019-08-08	2w50x0Z
3	100002	Michael Miller	1000	1008	0	2020-02-14	HZ30ZZZ
4	100006	Leah Manning	1001	1009	0	2019-11-21	4A1204Z
5	100006	Leah Manning	1001	1009	0	2018-05-01	Z12.11
6	100006	Leah Manning	1001	1009	0	2019-07-06	S14.109A
7	100007	Kelli Hendrickson	1002	1011	0	2018-12-02	2w50x0Z
8	100007	Kelli Hendrickson	1002	1011	0	2019-03-15	HZ30ZZZ
9	100012	Etel Arteaga	1003	1006	0	2018-07-19	4A1204Z
10	100012	Etel Arteaga	1003	1006	0	2018-12-01	S06.890A
11	100012	Etel Arteaga	1003	1006	0	2019-02-13	2w50x0Z
12	100012	Etel Arteaga	1003	1006	0	2019-04-08	Z12.11
13	100012	Etel Arteaga	1003	1006	0	2019-09-04	HZ30ZZZ
14	100012	Etel Arteaga	1003	1006	0	2020-04-22	0WH638Z

Prompt: The billing department is being audited and needs to pull all records for the procedures performed by Dr. Timmons in 2019.

Timmons PHYS_ID = 10006

```
SELECT PHYS_INFO.Phys_ID AS "Provider Number",
       trim(Phys_FirstName) + ' ' + trim(Phys_Name) AS "Provider Name",
       DOS AS "Date of Service",
       ICD10_Code AS "Procedure Code",
       trim(Pt_First_Name) + ' ' + trim(Pt_Last_Name) AS "Patient Name",
       PATIENT_INFO.Patient_ID AS "Patient ID"

FROM   PHYS_INFO
       JOIN PROC_INFO ON PHYS_INFO.Phys_ID = PROC_INFO.Phys_ID
       JOIN PATIENT_INFO ON PROC_INFO.Patient_ID = PATIENT_INFO.Patient_ID

WHERE  PHYS_INFO.Phys_ID = '10006' AND
       DOS BETWEEN '2019-01-01' AND '2019-12-31'

ORDER BY DOS
```

/*Prompt: The billing department is being audited and needs to pull all records for the procedures performed by Dr. Timmons in 2019.

Timmons PHYS_ID = 10006

*/

```
SELECT PHYS_INFO.Phys_ID AS "Provider Number",
       trim(Phys_FirstName) + ' ' + trim(Phys_Name) AS "Provider Name",
       DOS AS "Date of Service",
       ICD10_Code AS "Procedure Code",
       trim(Pt_First_Name) + ' ' + trim(Pt_Last_Name) AS "Patient Name",
       PATIENT_INFO.Patient_ID AS "Patient ID"

FROM   PHYS_INFO
       JOIN PROC_INFO ON PHYS_INFO.Phys_ID = PROC_INFO.Phys_ID
       JOIN PATIENT_INFO ON PROC_INFO.Patient_ID = PATIENT_INFO.Patient_ID

WHERE  PHYS_INFO.Phys_ID = '10006' AND
       DOS BETWEEN '2019-01-01' AND '2019-12-31'

ORDER BY DOS
```

100 %

Results Messages

	Provider Number	Provider Name	Date of Service	Procedure Code	Patient Name	Patient ID
1	10006	John Timmons	2019-04-08	Z12.11	Etel Arteaga	100012
2	10006	John Timmons	2019-07-13	S14.109A	Charles Mendoza	100011
3	10006	John Timmons	2019-07-13	S06.890A	Charles Mendoza	100011
4	10006	John Timmons	2019-07-22	4A1204Z	Michael Miller	100002
5	10006	John Timmons	2019-10-01	2W50X0Z	Edna Rogers	100010

Prompt: Billing standards were recently updated and procedure code HZ30ZZZ which represents 'Individual Counseling for Substance Abuse Treatment, Cognitive' is now being merged with 'Individual Counseling for Substance Abuse Treatment, Behavioral' (ICD HZ31ZZZ) for billing

purposes and all procedures with the code HZ30ZZZ need to be replaced with HZ32ZZZ that represents 'Individual Counseling for Substance Abuse Treatment, Cognitive-Behavioral,' and rebilled to the appropriate insurance company as an updated claim, starting with the primary provider.

```
SELECT ICD10_Code AS "Procedure Code",
       PATIENT_INFO.Patient_ID AS "Patient ID",
       trim(Pt_First_Name) + ' ' + trim(Pt_Last_Name) AS "Patient Name",
       Ins_Name AS "Primary Insurance",
       trim(Ins_add1) + ' ' + ISNULL(trim(Ins_add2), '') + ' ' + trim(Ins_city) +
       ', ' + trim(Ins_state) + ' ' + trim(Ins_zip) AS "Claims Address"

FROM   PROC_INFO
JOIN   PATIENT_INFO ON PROC_INFO.Patient_ID = PATIENT_INFO.Patient_ID
JOIN   INS_BRIDGE ON PATIENT_INFO.Patient_ID = INS_BRIDGE.Patient_ID
JOIN   INS_INFO ON INS_BRIDGE.Prim_Code = INS_INFO.Insurance_Code

WHERE  ICD10_CODE = 'HZ30ZZZ'

ORDER BY [Patient ID]
```

/*Prompt: Billing standards were recently updated and procedure code HZ30ZZZ which represents 'Individual Counseling for Substance Abuse Treatment, Cognitive' is now being merged with 'Individual Counseling for Substance Abuse Treatment, Behavioral' (ICD HZ31ZZZ) for billing purposes and all procedures with the code HZ30ZZZ need to be replaced with HZ32ZZZ that represents 'Individual Counseling for Substance Abuse Treatment, Cognitive-Behavioral,' and rebilled to the appropriate insurance company as an updated claim, starting with the primary provider.*/

```
SELECT ICD10_Code AS "Procedure Code",
       PATIENT_INFO.Patient_ID AS "Patient ID",
       trim(Pt_First_Name) + ' ' + trim(Pt_Last_Name) AS "Patient Name",
       Ins_Name AS "Primary Insurance",
       trim(Ins_add1) + ' ' + ISNULL(trim(Ins_add2), '') + ' ' + trim(Ins_city) + ', ' + trim(Ins_state) + ' ' + trim(Ins_zip) AS "Claims Address"

FROM   PROC_INFO
JOIN   PATIENT_INFO ON PROC_INFO.Patient_ID = PATIENT_INFO.Patient_ID
JOIN   INS_BRIDGE ON PATIENT_INFO.Patient_ID = INS_BRIDGE.Patient_ID
JOIN   INS_INFO ON INS_BRIDGE.Prim_Code = INS_INFO.Insurance_Code

WHERE  ICD10_CODE = 'HZ30ZZZ'

ORDER BY [Patient ID]
```

100 %

Results Messages

	Procedure Code	Patient ID	Patient Name	Primary Insurance	Claims Address
1	HZ30ZZZ	100002	Michael Miller	UPMC Health Plan	PO Box 2999 Pittsburgh, PA 15230-2999
2	HZ30ZZZ	100004	Melanie Preston	Keystone Blue	Claims Processing PO Box 898819 Camp Hill, PA 17...
3	HZ30ZZZ	100007	Kelli Hendrickson	UPMC for You MA	PO Box 2995 Pittsburgh, PA 15230-2999
4	HZ30ZZZ	100012	Etel Arteaga	UPMC Community HealthChoices	PO Box 106042 Pittsburgh, PA 15230-2999

LISTING OF AN SQL INSERT STATEMENT TO ADD AT LEAST ONE RECORD INTO ONE (OR MORE) OF THE TABLE(S) CONTAINED IN THE DATABASE

A new patient sent in registration documents prior to their appointment date before the database was launched. To test the system, the new patient has been added using the following:

```
SET IDENTITY_INSERT dbo.PATIENT_INFO OFF
```

```
INSERT INTO PATIENT_INFO  
    (Pt_Last_Name, Pt_First_Name, Pt_Middle_Name, DOB,  
    Pt_Add1, Pt_Add2, Pt_City, Pt_State, Pt_Zip, Pt_Phone) VALUES  
    ('Malone', 'Hubert', 'P', '2000-03-26', '3802 Carriage Lane', NULL,  
    'Bloomsberg', 'PA', '17815', '555 259 9895');
```

```
INSERT INTO INS_BRIDGE  
    (Patient_ID, Prim_Code, Prim_Pt_ID, Sec_Code, Sec_Pt_ID, Ter_Code,  
    Ter_PT_ID) VALUES  
    ('100013', '1005', '321B-65W3X', NULL, NULL, NULL, NULL);
```

Select *

FROM PATIENT_INFO

WHERE Pt_Last_Name = 'Malone';

100 %

Results Messages

	Patient_ID	Pt_Last_Name	Pt_First_Name	Pt_Middle_Name	DOB	Pt_Add1	Pt_Add2	Pt_City	Pt_State	Pt_Zip	Pt_Phone
1	100013	Malone	Hubert	P	2000-03-26	3802 Carriage Lane	NULL	Bloomsberg	PA	17815	555 259 9895

Select *

FROM INS_BRIDGE

WHERE Patient_ID = '100013';

100 %

Results Messages

	Patient_ID	Prim_Code	Prim_Pt_ID	Sec_Code	Sec_Pt_ID	Ter_Code	Ter_PT_ID
1	100013	1005	321B-65W3X	NULL	NULL	NULL	NULL

LISTING OF AN SQL UPDATE STATEMENT TO UPDATE AT LEAST ONE RECORD IN ONE (OR MORE) OF THE TABLE(S) CONTAINED IN THE DATABASE

Expanding on the previous problem where all procedure codes 'HZ30ZZZ' will be merged with 'HZ31ZZZ' and replaced with 'HZ32ZZZ,' rather than pull each individual account to update the procedure code, we can efficiently accomplish our goal by using an UPDATE statement as follows:

```
UPDATE PROC_INFO
SET ICD10_Code = 'HZ32ZZZ'
WHERE ICD10_Code = 'HZ30ZZZ' OR ICD10_Code = 'HZ31ZZZ';
```

Results of previously referenced SELECT statement:

- Note – There were no procedures with the code HZ31ZZZ in the system, but it was added for future reference if the update needs to be performed again.

	Procedure Code	Patient ID	Patient Name	Primary Insurance	Claims Address
1	HZ30ZZZ	100002	Michael Miller	UPMC Health Plan	PO Box 2999 Pittsburgh, PA 15230-2999
2	HZ30ZZZ	100004	Melanie Preston	Keystone Blue	Claims Processing PO Box 898819 Camp Hill, PA 17...
3	HZ30ZZZ	100007	Kelli Hendrickson	UPMC for You MA	PO Box 2995 Pittsburgh, PA 15230-2999
4	HZ30ZZZ	100012	Etel Arteaga	UPMC Community HealthChoices	PO Box 106042 Pittsburgh, PA 15230-2999

After UPDATE statement is executed:

```
SELECT ICD10_Code AS "Procedure Code",
PATIENT_INFO.Patient_ID AS "Patient ID",
trim(Pt_First_Name) + ' ' + trim(Pt_Last_Name) AS "Patient Name",
Ins_Name AS "Primary Insurance",
trim(Ins_add1) + ' ' + ISNULL(trim(Ins_add2), '') + ' ' + trim(Ins_city) + ', ' + trim(Ins_state) + ' ' + trim(Ins_zip) AS "Claims Address"
FROM PROC_INFO
JOIN PATIENT_INFO ON PROC_INFO.Patient_ID = PATIENT_INFO.Patient_ID
JOIN INS_BRIDGE ON PATIENT_INFO.Patient_ID = INS_BRIDGE.Patient_ID
JOIN INS_INFO ON INS_BRIDGE.Prim_Code = INS_INFO.Insurance_Code
WHERE ICD10_CODE = 'HZ32ZZZ'
ORDER BY [Patient ID]
```

	Procedure Code	Patient ID	Patient Name	Primary Insurance	Claims Address
1	HZ32ZZZ	100002	Michael Miller	UPMC Health Plan	PO Box 2999 Pittsburgh, PA 15230-2999
2	HZ32ZZZ	100004	Melanie Preston	Keystone Blue	Claims Processing PO Box 898819 Camp Hill, PA 17...
3	HZ32ZZZ	100007	Kelli Hendrickson	UPMC for You MA	PO Box 2995 Pittsburgh, PA 15230-2999
4	HZ32ZZZ	100012	Etel Arteaga	UPMC Community HealthChoices	PO Box 106042 Pittsburgh, PA 15230-2999

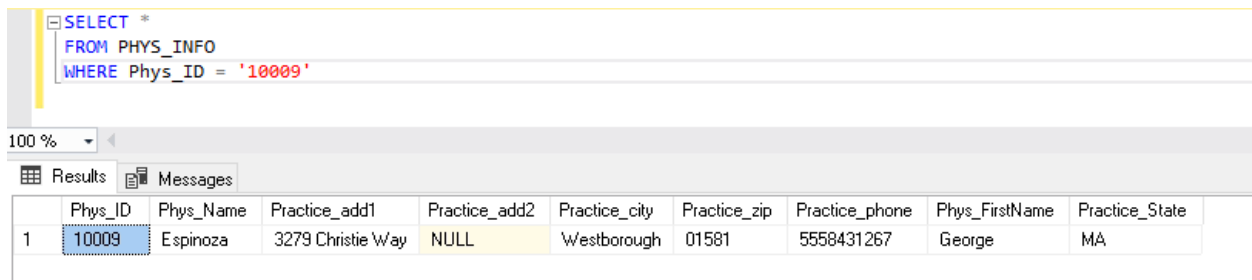
LISTING OF AN SQL DELETE STATEMENT TO DELETE AT LEAST ONE RECORD IN ONE (OR MORE) TABLE(S) CONTAINED IN THE DATABASE

Dr. Espinoza's group contract has ended and he has decided to begin in-house billing. So far, all of his procedure records have been sent to his practice location and have been backed up into a cold-storage server in our medical billing facility. We now need to remove him from our active system as well as the procedures he has performed to avoid billing conflicts using the following:

```
DELETE FROM PROC_INFO  
WHERE Phys_ID = '10009';
```

```
DELETE FROM PHYS_INFO  
WHERE Phys_ID = '10009';
```

Before deleting Dr. Espinoza from the system:

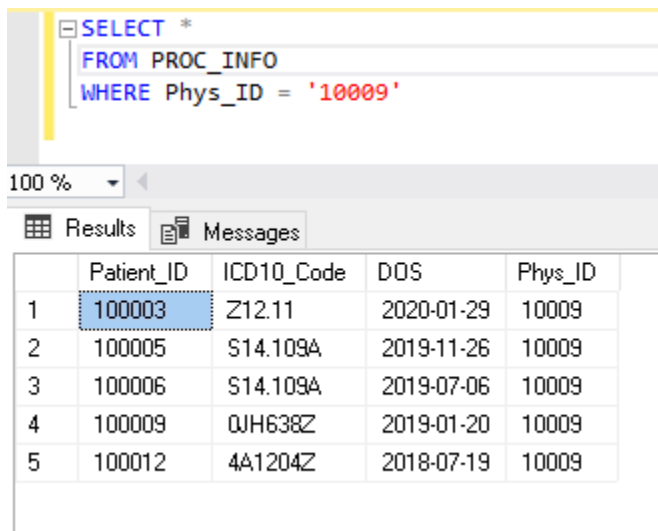


The screenshot shows a SQL query window with the following query:

```
SELECT *  
FROM PHYS_INFO  
WHERE Phys_ID = '10009'
```

The results pane shows a single record for Dr. Espinoza:

	Phys_ID	Phys_Name	Practice_add1	Practice_add2	Practice_city	Practice_zip	Practice_phone	Phys_FirstName	Practice_State
1	10009	Espinoza	3279 Christie Way	NULL	Westborough	01581	5558431267	George	MA



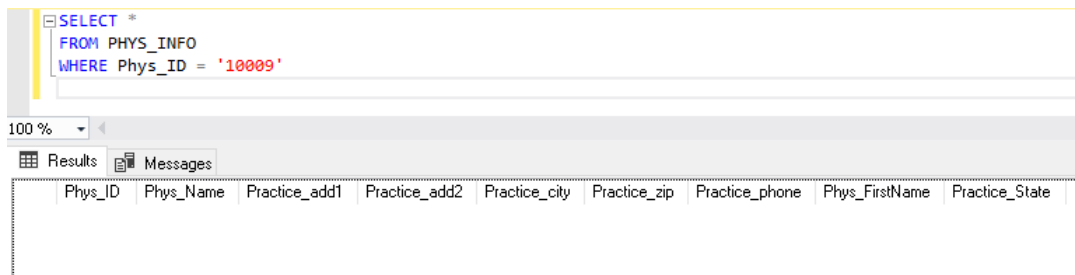
The screenshot shows a SQL query window with the following query:

```
SELECT *  
FROM PROC_INFO  
WHERE Phys_ID = '10009'
```

The results pane shows five procedure records for Dr. Espinoza:

	Patient_ID	ICD10_Code	DOS	Phys_ID
1	100003	Z12.11	2020-01-29	10009
2	100005	S14.109A	2019-11-26	10009
3	100006	S14.109A	2019-07-06	10009
4	100009	0JH638Z	2019-01-20	10009
5	100012	4A1204Z	2018-07-19	10009

After Dr. Espinoza has been deleted from the system:



The screenshot shows the same SQL query window as before, but the results pane is empty, indicating that Dr. Espinoza's record has been successfully deleted from the PHYS_INFO table.

Phys_ID	Phys_Name	Practice_add1	Practice_add2	Practice_city	Practice_zip	Practice_phone	Phys_FirstName	Practice_State
---------	-----------	---------------	---------------	---------------	--------------	----------------	----------------	----------------

SELECT *

FROM PROC_INFO

WHERE Phys_ID = '10009'

100 %

Results

Messages

Patient_ID	ICD10_Code	DOS	Phys_ID
------------	------------	-----	---------

OPTIONAL – PICTURE OF ONE OR MORE CREATED FORMS

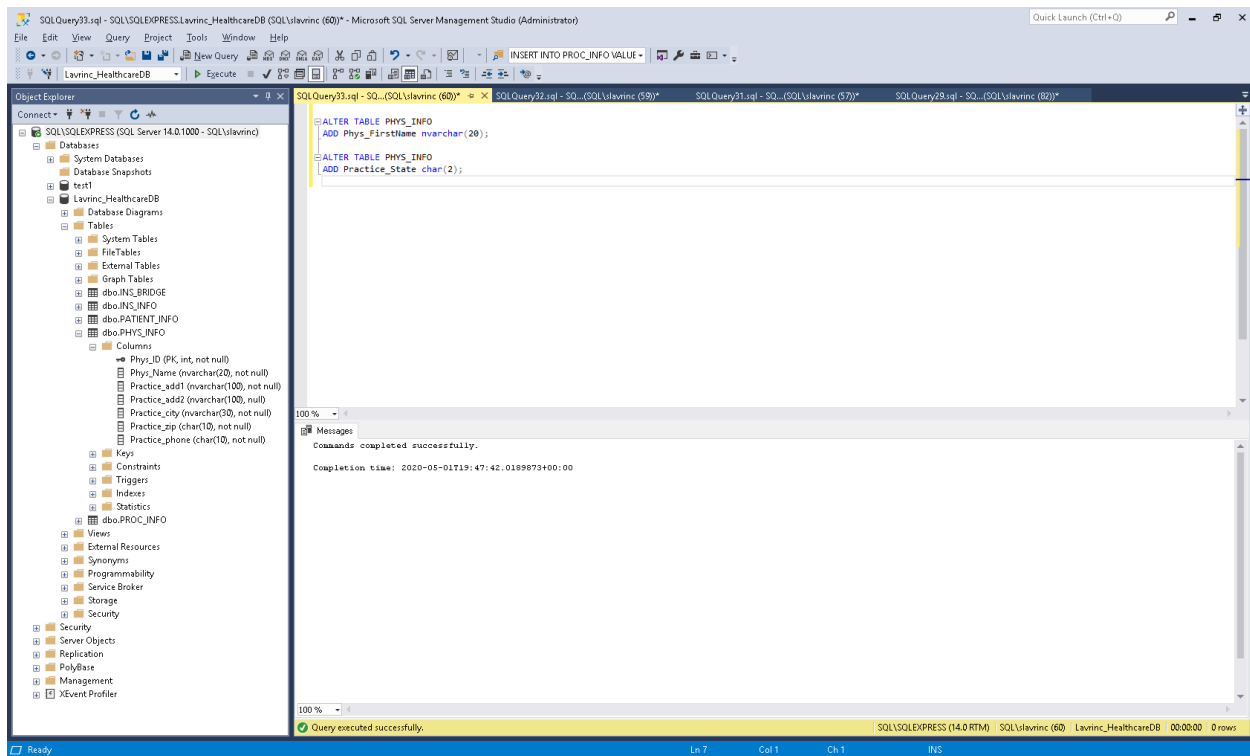
**OPTIONAL – PICTURE OF ONE OR MORE CREATED REPORT(S) USING
MICROSOFT SQL SERVER REPORT BUILDER**

OPTIONAL - OTHER INFORMATION YOU WANT TO INCLUDE REGARDING YOUR DATABASE PROJECT

Shortly after filling in the content for the tables, I realized I forgot some key column components in the PHYS_INFO table. Lacking a column to enter both the practice state and the physician's first name, I altered the tables to add these additional fields using the following:

```
ALTER TABLE PHYS_INFO  
ADD Phys_FirstName nvarchar(20);
```

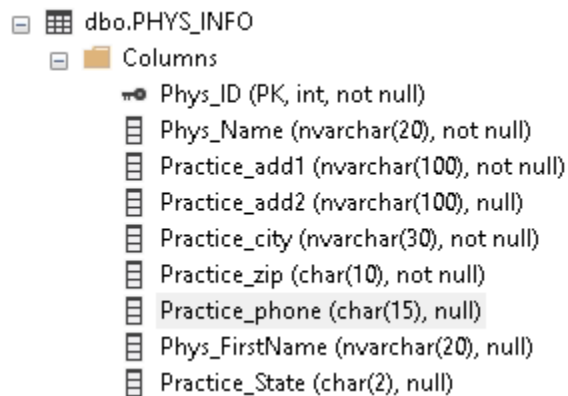
```
ALTER TABLE PHYS_INFO  
ADD Practice_State char(2);
```



dbo.PHYS_INFO
Columns
Phys_ID (PK, int, not null)
Phys_Name (nvarchar(20), not null)
Practice_add1 (nvarchar(100), not null)
Practice_add2 (nvarchar(100), null)
Practice_city (nvarchar(30), not null)
Practice_zip (char(10), not null)
Practice_phone (char(15), null)
Phys_FirstName (nvarchar(20), null)
Practice_State (char(2), null)

I've altered the column Practice_phone to be 15 characters long rather than 10 as I predict the phone numbers will occasionally have extensions using the following:

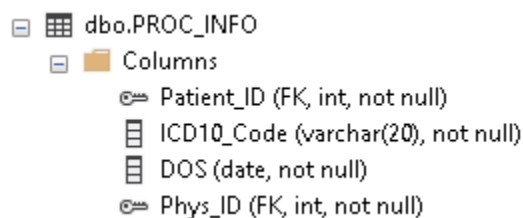
```
ALTER TABLE PHYS_INFO  
ALTER COLUMN Practice_phone char(15);
```



dbo.PHYS_INFO
Columns
Phys_ID (PK, int, not null)
Phys_Name (nvarchar(20), not null)
Practice_add1 (nvarchar(100), not null)
Practice_add2 (nvarchar(100), null)
Practice_city (nvarchar(30), not null)
Practice_zip (char(10), not null)
Practice_phone (char(15), null)
Phys_FirstName (nvarchar(20), null)
Practice_State (char(2), null)

Assigning a procedure ID was unnecessary in the context of this project. So, I removed the Proc_ID column from the PROC_INFO table using the following:

```
ALTER TABLE PROC_INFO  
DROP COLUMN Proc_ID;
```



dbo.PROC_INFO
Columns
Patient_ID (FK, int, not null)
ICD10_Code (varchar(20), not null)
DOS (date, not null)
Phys_ID (FK, int, not null)

Additional Thoughts:

I previously worked as an A/R specialist for a medical billing facility. When sending claims on the USER end of a database, I came across multiple issues with data that had certain CONSTRAINTS added which prevented the system itself from being able to perform properly. Specifically, I recall one instance where we had a patient and insurance provider from a different country, and no matter how I attempted to enter the correct data for the patient and insurance company address, the system would not accept it based on parameters created by the database programmers. Because the system was almost entirely automated, there was no way to address my claims by hand, and with the basic USER access I had, I could not adjust CONSTRAINTS required for the state, zip code or country columns that were causing our system to produce an error message.

With that in mind, while I understand the importance of practicing these CONSTRAINTS in this project, I wouldn't recommend adding too many as they can interfere with the purpose of the

database if they are over-utilized. Especially in those outlier scenarios that may not be predicted by individuals less familiar with the impact.

Finally, because there was a separate payments department in the company I was working for, I'm not familiar with what kind of tables and data are necessary for them to complete their roles. Additional input and/or experience is required to continue to develop this database to meet all of the needs of a hospital or medical billing facility.

SUMMARY OF YOUR CREATED DATABASE MANAGEMENT SYSTEM

This database was created as an exploratory project to enhance my understanding of the way databases work within a medical billing context. It includes Patient Information, Procedure Information, Physician Information, and Insurance Information to allow quick access and functionality to the users which inherently discourages profit loss from timely filing and disorganization. Utilizing a database for a medical billing system allows the ability to change large amounts of information all at once which is beneficial in an evolving environment where data may change rapidly depending on annual updates. The data in this system will grow exponentially as new patients, procedures, doctors and insurance companies are added, and being able to pull accounts that meet certain parameters increases efficiency and allows more opportunity for practice growth.

My biggest realization while working on this project is the scope of responsibilities that must be covered in order for the database to be useful for each user role. While I'm less familiar with the requirements a database must adhere to in order for some departments to be fully functional, this project can be utilized as a base model with intentions to expand the data included based on additional roles and responsibilities of the users.