Final Report

DBST 663 Group 6

University of Maryland University College

Table of Contents

[Statement of Work 3](#_Toc498790495)

[ERD and Relational Schema 6](#_Toc498790496)

[ERD 6](#_Toc498790497)

[Relational Schema 7](#_Toc498790498)

[Data Distribution Plan 9](#_Toc498790499)

[Implement Distribution Plan, Create DDLs, and Populate Tables 12](#_Toc498790500)

[Hospdba SQL 12](#_Toc498790501)

[Hospdbb SQL 24](#_Toc498790502)

[5 Views and Queries 25](#_Toc498790503)

[Hospdba SQL 25](#_Toc498790504)

[Collect and Report Query Metrics 29](#_Toc498790505)

[Hospdba SQL 29](#_Toc498790506)

[Illustrate 5 Views and Queries 31](#_Toc498790507)

# **Statement of Work**

**Statement of Work**

**Introduction/Project Definition**

Get Well Health System has requested assistance from G Six Consulting to assist in implementing a new database system for their hospital. They currently have employee, hospital, and patient data splintered throughout the health system. Get Well Health System currently has five acute care hospitals, five divisions for home health/hospice agencies, 100 medical practices with varying specialties, and ten long term care facilities. The goal is to implement one database that will hold the data for employees, facilities, and patients.

**Scope of Work**

This is a massive project, therefore it will need to be addressed in phases. The first phase is to create a database that will be used for the five acute care facilities. The database will contain data for employees, providers, hospitals, medications, prescription, and separate visits. It will include the development of structure of the ERD, creation of specific tables, and proof of usability and reliability.

We will not be addressing any hardware issues, or making any changes. Our team will also not be responsible for training all personnel; we will train a set number of individuals, by reviewing the database, and they will be the Get Well Health System’s point of contact. We will also not be uploading large sets of data from all antiquated database systems, Get Well Health System will need to designate scribes to input previous data from the various databases to ensure the database is up to date and ready to run upon implementation.

**Constraints and Assumptions**

There are some constraints when working with a patient’s private identifying information. There are HIPPA guidelines we must follow. We must also follow labor guidelines when handling employee’s information as well. What will be needed in the database will need to be restraints on who is able to access what pieces of data; for example, medications will not need to be seen by all database users, only those providing direct care to the patient. An employee’s salary should not be shown to all users, only employees in compensation.

The assumptions being made is that all facilities will utilize this new database, it is not optional. All facilities will also have super users of the new database system. Facilities will also go live at one time, this will not be staggered implementation. The last assumption made is that all historical data from antiquated database systems are valid and reliable. There will be no changes made when transferring data, and if changes need to be made to instances, it will be noted and updated at a later date.

**Period of Performance**

This first phase is expected to be completed within nine weeks. We, as the consulting company, will present a draft entity relationship diagram, then upon completion demonstrate the performance of the database. The project is expected to begin September 18, 2017 and will conclude

**Work Requirements**

**Kickoff:**

The kickoff for this project will take place the week of September 18, 2017. This will include a brainstorming phase of the team members and decision makers from Get Well Health System. Through the kickoff phase, we will be working to put together the design phase.

**Design Phase:**

The design phase includes the creation of the entity relationship diagram, along with the relationship schema. During this phase, we will be identifying the necessary attributes needed for each table. Legally, as a health system there is data that has to be recorded, and some pieces of data that is supportive of the regulated data.

With regard to this database, it will operate with synchronous control, which means when there are updates or transactions made against the database, the entire database will update with that new data. Data will be kept locally at the five acute care hospital, along with an additional copy in the information systems building.

**Build Phase:**

The build phase consists of the G Six Consulting firm creating the DDL for the skeleton of the database. Tables, constraints, and other pertinent table details will be created. The table will be normalized, and tested to ensure it meets the basic requirements. Servers within each facility, including IS will need to be partitioned to hold this new set of data. The data from the previous database systems will need to be archived for historical purposes.

**Implementation Phase:**

Once the database is built, we will begin the implementation phase. This phase will include the creation of instances within the table. Scribers will need to meet in massive numbers to transfer data from one database to the other. The legacy database systems do not have a file type that is compatible to the new database, therefore it will need to be individually typed or scanned into the system.

Within this phase, users will be selected to work in the system. They will ensure it meets the needs of the health system. If there are changes that need to be made, it will be completed during this stage.

**Training Phase:**

The super users will be trained by G Six Consulting Firm on how to use the database. The super users will train team members throughout the health system. During the initial phases of implementation, our team will remain on site to ensure the database is working without failures.

**Project Handoff/Closure**

This project will be handed off to Get Well Health System on December 3, 2017. Once completed, the database will have various sets of data available; employee information, hospital information, patient information, previous visit data, and data referencing medications/prescriptions. With this being the conclusion of the first phase, the acute care hospitals will continue to have the team on site for support, as other phases of the project is being implemented.

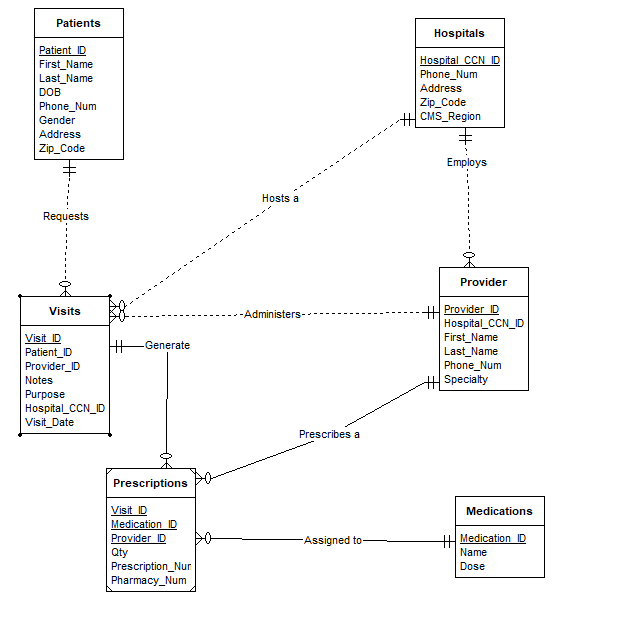
As this project is being closed, the database will be assessed to determine if there are other attributes that need to be added to the database. There will be time, at least 30 days, before the next phase to create updates to the newly created database.

References:

Brujhito. (2016). Docuri. Project assumptions and risks.pdf. Retrieved from <https://docuri.com/download/project-assumptions-and-riskspdf_59ae4825f581710a62011cd1_pdf>

# **ERD and Relational Schema**

## **ERD**



## **Relational Schema**

**Hospital Schema Design**

**Tables**

**Patients**: Patients details.

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Datatype | Constraint | Comment |
| Patient\_ID | Integer | Primary key |  |
| First\_Name | Varchar | Not null |  |
| Last\_Name | Varchar | Not null |  |
| DOB | Date | Not null |  |
| Phone\_Num | Number | Not null |  |
| Gender | Boolean | Not null | F = Female/M = Male |
| Address | Varchar | Not null |  |
| Zip\_Code | Varchar | Not null |  |

**Visits**: Visit request details regarding the Patients.

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Datatype | Constraint | Comment |
| Visit\_ID | Integer | Primary key |  |
| Patient\_ID | Integer | Primary key/Foreign key | Composite PK |
| Provider\_ID | Integer | Primary key/Foreign key | Composite PK |
| Notes | Varchar | Not null |  |
| Purpose | Varchar | Not null |  |
| Hospital\_CCN\_ID | Integer | Primary key/Foreign key | Composite PK |
| Visit\_Date | Date | Not null |  |

**Hospitals**: Hospitals host visits from patients.

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Datatype | Constraint | Comment |
| Hospital\_CCN\_ID | Integer | Primary key |  |
| Phone\_Num | Number | Not null |  |
| Address | Varchar | Not null |  |
| Zip\_Code | Integer | Not null |  |
| CMS\_Region | Varchar | Not null |  |

**Provider**: Patients are administered care by Provider who provides prescriptions.

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Datatype | Constraint | Comment |
| Provider\_ID | Integer | Primary key |  |
| Hospital\_CCN\_ID | Integer | Foreign key |  |
| First\_Name | Varchar | Not null |  |
| Last\_Name | Varchar | Not null |  |
| Phone\_Num | Number | Not null |  |
| Specialty | Varchar | Not null |  |

**Prescriptions**: Are generated for a visit, prescribed by providers, and assigned a medication.

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Datatype | Constraint | Comment |
| Visit\_ID | Integer | Primary key |  |
| Medication\_ID | Integer | Primary key/Foreign key | Composite PK |
| Provider\_ID | Integer | Primary key/Foreign key | Composite PK |
| Qty | Number | Not null |  |
| Prescription\_Nur | Integer | Not null |  |
| Pharmary\_Num | Number | Not null |  |

**Medicatios**: Medication given is determined by the prescription that is submitted.

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Datatype | Constraint | Comment |
| Medication\_ID | Integer | Primary Key |  |
| Name | Varchar |  |  |
| Dose | Varchar |  |  |

# **Data Distribution Plan**

Data Distribution Plan

DBST 663 Group 6

University of Maryland University College

Data Distribution Plan

**Describe the Database Distribution**

In our hospital database, there are 6 tables. We will create two different databases called “Hospdba” and “Hospdbb”, and we will create a link between these two. We will be utilizing the horizontal fragmentation strategy for one table, the “Visits” table. We will create 6 tables in the first database, and then 1 table, the fragmented one, in the second database.

**Identify Tables for Fragmentation**

We have chosen to split our “Visits” table with horizontal fragmentation. The next section will provide the rationale behind this decision.

**Provide Rationale for the Fragmentation**

The rationale behind horizontally fragmenting the “Visits” table is due to the nature of this table having significantly more entries than any of the other tables. Distributing the load of this table across two databases will provide enhanced query speeds and reduce network latency.

**Describe 5 Distributed Views and Queries**

Here are 5 distributed views and queries based upon real-world requirements needed from the hospital.

1. The view will be a union between the “Patients” tables on “Hospdba” and “Hospdbb”.
2. The first query will select all data from the “Patients” table, joined with the “Visits” and “Provider” tables where the “Provider” specialty equals “Radiology”.
3. The second query will select all data from the “Patients” table, joined with the “Provider”, “Visits”, “Prescriptions”, and “Medications” tables where the “Medications” name equals “Penicillin” and the “Visits” date field is equal to or greater than “Jan 1, 2017”.
4. The third query joins together the “Hospital” table, “Visits” table, and “Patients” table to show how many visits a CMS Region has had with the last patient seen, sorted by the number of visits.
5. The fourth query joins together the “Patients” table with the “Visits” table, and a subquery will be performed to determine how many visits a patient has had, in order by visit count.

**Describe the Updates Plan**

In our database, the most likely time needed to update data would be when a doctor needs to go back in for adding additional notes to a visit. We will provide several updates to the “Visits” table, specifically the notes column. The updates will be based on the Patients\_ID, Providers\_ID, and Hospital\_CCN\_ID.

**Describe the Retrieval Strategy**

For our project, we will be creating a few views to pull together the fragmented data. These views will be the primary source for our distributed queries detailed in the second above. Outside of the fragmented “Visits” table, retrieval of all remaining data will be normal.

# **Implement Distribution Plan, Create DDLs, and Populate Tables**

## **Hospdba SQL**

/\*Set timing to on to see query times\*/

SET TIMING ON

/\*Drop all tables\*/

DROP TABLE Prescriptions;

DROP TABLE Visits;

DROP TABLE Provider;

DROP TABLE Patients;

DROP TABLE Hospitals;

DROP TABLE Medications;

/\*Drop Link\*/

DROP DATABASE LINK Linker\_1\_2;

/\*Drop Sequence\*/

DROP SEQUENCE Hospitals\_Seq;

DROP SEQUENCE Visits\_Seq;

DROP SEQUENCE Provider\_Seq;

DROP SEQUENCE Medications\_Seq;

/\*Create all tables\*/

CREATE TABLE Patients (

Patient\_ID NUMBER NOT NULL,

First\_Name VARCHAR(50) NOT NULL,

Last\_Name VARCHAR(50) NOT NULL,

DOB DATE NOT NULL,

Phone\_Num VARCHAR(50) NOT NULL,

Gender VARCHAR(2) NOT NULL,

Address VARCHAR(50) NOT NULL,

Zip\_Code VARCHAR(15) NOT NULL,

Database\_Name VARCHAR(50) NOT NULL,

CONSTRAINT pk\_patients PRIMARY KEY (Patient\_ID)

);

CREATE TABLE Hospitals (

Hospital\_CCN\_ID NUMBER NOT NULL,

Phone\_Num VARCHAR(50) NOT NULL,

Address VARCHAR(50) NOT NULL,

Zip\_Code VARCHAR(15) NOT NULL,

CMS\_Region VARCHAR(50) NOT NULL,

CONSTRAINT pk\_hospitals PRIMARY KEY (Hospital\_CCN\_ID)

);

CREATE TABLE Provider (

Provider\_ID NUMBER NOT NULL,

Hospital\_CCN\_ID NUMBER NOT NULL,

First\_Name VARCHAR(50) NOT NULL,

Last\_Name VARCHAR(50) NOT NULL,

Phone\_Num VARCHAR(50) NOT NULL,

Specialty VARCHAR(50) NOT NULL,

CONSTRAINT pk\_provider PRIMARY KEY (Provider\_ID),

CONSTRAINT fk\_hospital\_2 FOREIGN KEY (Hospital\_CCN\_ID)

REFERENCES Hospitals

);

/\*Patient\_ID foreign key enforced with trigger below\*/

CREATE TABLE Visits (

Visit\_ID NUMBER NOT NULL,

Patient\_ID NUMBER NOT NULL,

Provider\_ID NUMBER NOT NULL,

Notes VARCHAR(50) NOT NULL,

Purpose VARCHAR(50) NOT NULL,

Hospital\_CCN\_ID NUMBER NOT NULL,

Visit\_Date DATE NOT NULL,

CONSTRAINT pk\_visits PRIMARY KEY (Visit\_ID),

CONSTRAINT fk\_provider FOREIGN KEY (Provider\_ID)

REFERENCES Provider,

CONSTRAINT fk\_hospital FOREIGN KEY (Hospital\_CCN\_ID)

REFERENCES Hospitals

);

CREATE TABLE Medications (

Medication\_ID NUMBER NOT NULL,

Name varchar(50) NOT NULL,

Dose varchar(50) NOT NULL,

CONSTRAINT pk\_medications PRIMARY KEY (Medication\_ID)

);

CREATE TABLE Prescriptions (

Visit\_ID NUMBER NOT NULL,

Medication\_ID NUMBER NOT NULL,

Provider\_ID NUMBER NOT NULL,

Qty INT NOT NULL,

Prescription\_Num INT NOT NULL,

Pharmacy\_Num INT NOT NULL,

CONSTRAINT fk\_medication FOREIGN KEY (Medication\_ID)

REFERENCES Medications,

CONSTRAINT fk\_visit FOREIGN KEY (Visit\_ID)

REFERENCES Visits,

CONSTRAINT fk\_provider\_2 FOREIGN KEY (Provider\_ID)

REFERENCES Provider,

CONSTRAINT pk\_prescriptions PRIMARY KEY (Visit\_ID, Medication\_ID, Provider\_ID)

);

/\*Create Database Link\*/

CREATE DATABASE LINK Linker\_1\_2 CONNECT TO system IDENTIFIED BY "0racl3Adm1n" USING 'dbst663b';

/\*Test Database link\*/

SELECT \* FROM DUAL@Linker\_1\_2;

/\* Create indexes on foreign keys\*/

CREATE INDEX fk\_provider on Visits(Provider\_ID);

CREATE INDEX fk\_hospital on Visits(Hospital\_CCN\_ID);

CREATE INDEX fk\_hospital\_2 on Provider(Hospital\_CCN\_ID);

CREATE INDEX fk\_medication on Prescriptions(Medication\_ID);

CREATE INDEX fk\_visit ON Prescriptions (Visit\_ID);

CREATE INDEX fk\_provider\_2 ON Prescriptions (Provider\_ID);

/\* Create sequence\*/

/\*There is no sequence for the Patients table, you cannot use a local object for

a remote database.\*/

CREATE SEQUENCE Hospitals\_Seq

START WITH 1

INCREMENT BY 1;

CREATE SEQUENCE Visits\_Seq

START WITH 1

INCREMENT BY 1;

CREATE SEQUENCE Provider\_Seq

START WITH 1

INCREMENT BY 1;

CREATE SEQUENCE Medications\_Seq

START WITH 1

INCREMENT BY 1;

/\*Insert Test Data\*/

/\*Patients Table\*/

INSERT INTO Patients (Patient\_ID,First\_Name,Last\_Name,DOB,Phone\_Num,Gender,Address,Zip\_Code,Database\_Name)

VALUES ( 1,'John','Fedric','12-FEB-1980','111-143-1242','M','23 block','14500','hospdba');

INSERT INTO Patients (Patient\_ID,First\_Name,Last\_Name,DOB,Phone\_Num,Gender,Address,Zip\_Code,Database\_Name)

VALUES ( 2, 'Aaron', 'Hank', '04-APR-1994', '111-144-0423', 'M', '99 city', '78000', 'hospdba');

INSERT INTO Patients (Patient\_ID,First\_Name,Last\_Name,DOB,Phone\_Num,Gender,Address,Zip\_Code,Database\_Name)

VALUES ( 3, 'Abbey', 'Edward', '27-MAR-1998', '111-924-0598', 'F', 'Main 2A', '34200', 'hospdba');

INSERT INTO Patients (Patient\_ID,First\_Name,Last\_Name,DOB,Phone\_Num,Gender,Address,Zip\_Code,Database\_Name)

VALUES ( 4, 'Abelson', 'Hall', '28-JUL-1955', '111-178-6290', 'F', '13-B sid' ,'36600', 'hospdba');

/\*Insert Data into linked database\*/

INSERT INTO Patients@Linker\_1\_2 (Patient\_ID,First\_Name,Last\_Name,DOB,Phone\_Num,Gender,Address,Zip\_Code,Database\_Name)

VALUES ( 5, 'Bob', 'Lee', '12-JUN-2001', '111-169-8399', 'M', 'Street 14', '42000','hospdbb');

INSERT INTO Patients@Linker\_1\_2 (Patient\_ID,First\_Name,Last\_Name,DOB,Phone\_Num,Gender,Address,Zip\_Code,Database\_Name)

VALUES ( 6, 'Dawane', 'Bravo', '19-SEP-1990', '111-359-0086', 'M', '19 A city', '65000','hospdbb');

INSERT INTO Patients@Linker\_1\_2 (Patient\_ID,First\_Name,Last\_Name,DOB,Phone\_Num,Gender,Address,Zip\_Code,Database\_Name)

VALUES ( 7, 'Andrew', 'Stall', '05-DEC-1998', '342-508-2222', 'M', '52 belon', '77600','hospdbb');

INSERT INTO Patients@Linker\_1\_2 (Patient\_ID,First\_Name,Last\_Name,DOB,Phone\_Num,Gender,Address,Zip\_Code,Database\_Name)

VALUES ( 8, 'Stella', 'Queen', '14-JUL-1992', '566-376-1111', 'F', 'Wagha 4', '64500','hospdbb');

/\*Hospitals Table\*/

INSERT INTO Hospitals (Hospital\_CCN\_ID,Phone\_Num,Address,Zip\_Code,CMS\_Region)

VALUES (Hospitals\_Seq.NEXTVAL, '222-333-4444', '123 abc st.', '12345', '1');

INSERT INTO Hospitals (Hospital\_CCN\_ID,Phone\_Num,Address,Zip\_Code,CMS\_Region)

VALUES (Hospitals\_Seq.NEXTVAL, '111-555-7777', '254 green st.', '29456', '2');

INSERT INTO Hospitals (Hospital\_CCN\_ID,Phone\_Num,Address,Zip\_Code,CMS\_Region)

VALUES (Hospitals\_Seq.NEXTVAL, '345-678-1212', '789 blue st.', '31088', '3');

INSERT INTO Hospitals (Hospital\_CCN\_ID,Phone\_Num,Address,Zip\_Code,CMS\_Region)

VALUES (Hospitals\_Seq.NEXTVAL, '909-808-3459', '1002 red st.', '31771', '4');

/\*Provider Table\*/

INSERT INTO Provider (Provider\_ID, Hospital\_CCN\_ID, First\_Name, Last\_Name, Phone\_Num, Specialty)

VALUES (Provider\_Seq.NEXTVAL, 1, 'Brandon', 'Russell', '678-435-1858', 'Lungs');

INSERT INTO Provider (Provider\_ID, Hospital\_CCN\_ID, First\_Name, Last\_Name, Phone\_Num, Specialty)

VALUES (Provider\_Seq.NEXTVAL, 2, 'Daniela', 'Montero', '404-123-7890', 'Heart');

INSERT INTO Provider (Provider\_ID, Hospital\_CCN\_ID, First\_Name, Last\_Name, Phone\_Num, Specialty)

VALUES (Provider\_Seq.NEXTVAL, 3, 'Patricia', 'Bargueno', '999-888-1698', 'Radiology');

INSERT INTO Provider (Provider\_ID, Hospital\_CCN\_ID, First\_Name, Last\_Name, Phone\_Num, Specialty)

VALUES (Provider\_Seq.NEXTVAL, 4, 'Mickey', 'Mouse', '000-345-7654', 'Family Medicine');

/\*Visits Table\*/

INSERT INTO Visits (Visit\_ID, Patient\_ID, Provider\_ID, Notes, Purpose, Hospital\_CCN\_ID, Visit\_Date)

VALUES (Visits\_Seq.NEXTVAL, 1,1,'Patient needs new air filter','cough',1, '28-JUL-2016');

INSERT INTO Visits (Visit\_ID, Patient\_ID, Provider\_ID, Notes, Purpose, Hospital\_CCN\_ID, Visit\_Date)

VALUES (Visits\_Seq.NEXTVAL, 1,1,'air filter did not work...give advil','cough',1, '28-JUL-2017');

INSERT INTO Visits (Visit\_ID, Patient\_ID, Provider\_ID, Notes, Purpose, Hospital\_CCN\_ID, Visit\_Date)

VALUES (Visits\_Seq.NEXTVAL, 2,2,'Need to order xrays','chest pain',2, '12-OCT-2016');

INSERT INTO Visits (Visit\_ID, Patient\_ID, Provider\_ID, Notes, Purpose, Hospital\_CCN\_ID, Visit\_Date)

VALUES (Visits\_Seq.NEXTVAL, 3,3,'MRI results are negative','left radius broke ',3, '27-OCT-2017');

INSERT INTO Visits (Visit\_ID, Patient\_ID, Provider\_ID, Notes, Purpose, Hospital\_CCN\_ID, Visit\_Date)

VALUES (Visits\_Seq.NEXTVAL, 5,2,'Serious heart problems!!!','Heart Attack',3, '02-Aug-2017');

/\*Commented out insert statements used to test trigger constraints\*/

/\*

INSERT INTO Visits (Visit\_ID, Patient\_ID, Provider\_ID, Notes, Purpose, Hospital\_CCN\_ID, Visit\_Date)

VALUES (Visits\_Seq.NEXTVAL, 125,2,'Serious heart problems!!!','Heart Attack',3, '02-Aug-2017');

\*/

/\*Medications Table\*/

INSERT INTO Medications(Medication\_ID, Name, Dose)

VALUES (Medications\_Seq.NEXTVAL, 'Penicillin', '1mg');

INSERT INTO Medications(Medication\_ID, Name, Dose)

VALUES (Medications\_Seq.NEXTVAL, 'Advil', '2mg');

INSERT INTO Medications(Medication\_ID, Name, Dose)

VALUES (Medications\_Seq.NEXTVAL, 'Aleve', '3mg');

INSERT INTO Medications(Medication\_ID, Name, Dose)

VALUES (Medications\_Seq.NEXTVAL, 'Benadryl', '6ml');

/\*Prescriptions Table\*/

INSERT INTO Prescriptions (Visit\_ID, Medication\_ID, Provider\_ID, Qty, Prescription\_Num, Pharmacy\_Num)

VALUES (1,1,1,1,342, 125);

INSERT INTO Prescriptions (Visit\_ID, Medication\_ID, Provider\_ID, Qty, Prescription\_Num, Pharmacy\_Num)

VALUES (2,2,1,1,789, 273);

INSERT INTO Prescriptions (Visit\_ID, Medication\_ID, Provider\_ID, Qty, Prescription\_Num, Pharmacy\_Num)

VALUES (3,3,2,1,1001, 344);

INSERT INTO Prescriptions (Visit\_ID, Medication\_ID, Provider\_ID, Qty, Prescription\_Num, Pharmacy\_Num)

VALUES (4,1,3,1,052, 888);

INSERT INTO Prescriptions (Visit\_ID, Medication\_ID, Provider\_ID, Qty, Prescription\_Num, Pharmacy\_Num)

VALUES (4,2,3,1,053, 888);

INSERT INTO Prescriptions (Visit\_ID, Medication\_ID, Provider\_ID, Qty, Prescription\_Num, Pharmacy\_Num)

VALUES (5,2,2,1,1002, 767);

/\* Create trigger \*/

/\*This trigger will enforce patient\_id foreign key constraint across the distributed patients table\*/

CREATE OR REPLACE TRIGGER Visit\_Patient\_Insert\_Constraint BEFORE INSERT OR UPDATE ON Visits

FOR EACH ROW

DECLARE

X NUMBER;

BEGIN

SELECT COUNT(\*) INTO X

FROM all\_patients\_view

WHERE Patient\_ID = :New.Patient\_ID;

if X = 0

THEN

raise\_application\_error (-20100, 'Patient\_ID invalid');

END IF;

END;

/

/\*This trigger will enforce patient\_id foreign key constraint across the distributed patients table\*/

CREATE OR REPLACE TRIGGER Visit\_Patient\_Delete\_Constraint BEFORE DELETE OR UPDATE ON Patients

FOR EACH ROW

DECLARE

X NUMBER;

BEGIN

SELECT COUNT(\*) INTO X

FROM Visits

WHERE Patient\_ID = :OLD.Patient\_ID;

if X > 0

THEN

raise\_application\_error (-20100, 'Delete or update children rows from Visits table first');

END IF;

END;

/

/\*These update and delete commands test the delete trigger above\*/

/\*DELETE FROM Patients WHERE Patient\_ID = 1;

DELETE FROM Patients WHERE Patient\_ID = 4;

UPDATE Patients SET Patient\_ID = 9 WHERE Patient\_ID = 2;

DELETE FROM Patients@Linker\_1\_2 WHERE Patient\_ID = 5;

UPDATE Patients@Linker\_1\_2 SET Patient\_ID = 9 WHERE Patient\_ID = 5;\*/

/\*Run Select all for all tables, to ensure insert statements worked.\*/

SELECT \* FROM all\_patients\_view;

SELECT \* FROM Hospitals;

SELECT \* FROM Visits;

SELECT \* FROM Provider;

SELECT \* FROM Prescriptions;

SELECT \* FROM Medications;

## **Hospdbb SQL**

/\*Drop all tables\*/

DROP TABLE Patients;

/\*Drop Link\*/

DROP DATABASE LINK Linker\_3\_4;

/\*Create all tables\*/

CREATE TABLE Patients (

Patient\_ID NUMBER NOT NULL,

First\_Name VARCHAR(50) NOT NULL,

Last\_Name VARCHAR(50) NOT NULL,

DOB DATE NOT NULL,

Phone\_Num VARCHAR(50) NOT NULL,

Gender VARCHAR(2) NOT NULL,

Address VARCHAR(50) NOT NULL,

Zip\_Code Varchar(15) NOT NULL,

Database\_Name VARCHAR(50) NOT NULL,

CONSTRAINT pk\_patients PRIMARY KEY (Patient\_ID)

);

CREATE DATABASE LINK Linker\_3\_4 CONNECT TO system IDENTIFIED BY "0racl3Adm1n" USING 'dbst663a';

/\*This trigger will enforce patient\_id foreign key constraint across the distributed patients table\*/

CREATE OR REPLACE TRIGGER Visit\_Patient\_Delete\_Constraint BEFORE DELETE OR UPDATE ON Patients

FOR EACH ROW

DECLARE

X NUMBER;

BEGIN

SELECT COUNT(\*) INTO X

FROM Visits@Linker\_3\_4

WHERE Patient\_ID = :OLD.Patient\_ID;

if X > 0

THEN

raise\_application\_error (-20100, 'Delete or update children rows from Visits table first');

END IF;

END;

/

/\*These update and delete commands test the delete trigger above\*/

/\*DELETE FROM Patients WHERE Patient\_ID = 5;

DELETE FROM Patients WHERE Patient\_ID = 8;

UPDATE Patients SET Patient\_ID = 9 WHERE Patient\_ID = 5;\*/

commit;

# **5 Views and Queries**

## **Hospdba SQL**

/\*Views\*/

/\*View joins together the horizontally fragmented Patients tables on Hospdba and Hospdbb\*/

CREATE OR REPLACE VIEW all\_patients\_view AS

SELECT \* FROM Patients

UNION

SELECT \* FROM Patients@Linker\_1\_2;

/\*Test the view\*/

SELECT \* FROM all\_patients\_view;

/\*4 Queries\*/

/\*This query joins the "Patients", "Visits", and "Provider" tables where the specialty

of the provider is "Radiology\*/

SELECT TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name) AS Patient, vi.purpose AS Purpose,

TO\_CHAR(pr.first\_name) || ' ' || TO\_CHAR(pr.last\_name) AS Provider FROM all\_patients\_view Pa

INNER JOIN Visits Vi USING (Patient\_ID)

INNER JOIN Provider Pr USING (Provider\_ID)

WHERE Pr.specialty = 'Radiology' ;

/\*This query will show the "Patients" table joined with the "Provider" , "Visits",

"Prescriptions", and "Medication" tables and where the medication is penicillin and the

visit occured on or after 1/1/2017\*/

SELECT TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name) AS Patient, vi.purpose AS Purpose,

TO\_CHAR(pr.first\_name) || ' ' || TO\_CHAR(pr.last\_name) AS Provider, Vi.Visit\_Date AS Visit\_Date,

Med.Name AS Medication, Med.Dose AS Dose, Pres.qty as Quantity FROM all\_patients\_view Pa

INNER JOIN Visits Vi USING (Patient\_ID)

INNER JOIN Provider Pr USING (Provider\_ID)

INNER JOIN Prescriptions Pres USING (Visit\_ID)

INNER JOIN Medications Med USING (Medication\_ID)

WHERE Med.Name = 'Penicillin' AND Vi.Visit\_Date > = '01-Jan-2017';

/\*This query will show the number of visits a CMS region has had along with the

most recent visit date and last patient seen, sorted by the number of visits.\*/

SELECT Hosp.CMS\_Region as CMS\_Region, COUNT(Vi.Visit\_ID) AS Visit\_Count,

MAX(Vi.Visit\_Date) AS Most\_Recent\_Visit,

TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name) AS Last\_Patient\_Seen

FROM Hospitals Hosp

LEFT JOIN Visits Vi On Vi.Hospital\_CCN\_ID = Hosp.Hospital\_CCN\_ID

LEFT JOIN all\_patients\_view Pa ON Vi.Patient\_ID = Pa.Patient\_ID

GROUP BY Hosp.CMS\_Region, TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name)

ORDER BY Visit\_Count;

/\*This query will join together the patient and visits table, and then using a subquery

determine how many visits a patient has had, in order by visit count\*/

SELECT DISTINCT Pa.Patient\_ID, TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name) AS Patient,

(SELECT COUNT(Patient\_ID) FROM Visits iVi WHERE iVi.Patient\_ID = Vi.Patient\_ID) as Visit\_Count

FROM all\_patients\_view Pa

LEFT JOIN Visits Vi ON Vi.Patient\_ID = Pa.Patient\_ID

ORDER BY Visit\_Count DESC;

/\*commit changes to database\*/

commit;

# **Collect and Report Query Metrics**

## **Hospdba SQL**

/\*Repeat the 5 views and queries above, this time using EXPLAIN PLAN to show the execution plan\*/

/\*The view\*/

EXPLAIN PLAN FOR

SELECT \* FROM all\_patients\_view;

SELECT \* FROM table(DBMS\_XPLAN.DISPLAY);

/\*The 4 Queries\*/

/\*This query joins the "Patients", "Visits", and "Provider" tables where the specialty

of the provider is "Radiology\*/

EXPLAIN PLAN FOR

SELECT TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name) AS Patient, vi.purpose AS Purpose,

TO\_CHAR(pr.first\_name) || ' ' || TO\_CHAR(pr.last\_name) AS Provider FROM all\_patients\_view Pa

INNER JOIN Visits Vi USING (Patient\_ID)

INNER JOIN Provider Pr USING (Provider\_ID)

WHERE Pr.specialty = 'Radiology' ;

SELECT \* FROM table(DBMS\_XPLAN.DISPLAY);

/\*This query will show the "Patients" table joined with the "Provider" , "Visits",

"Prescriptions", and "Medication" tables and where the medication is penicillin and the

visit occured on or after 1/1/2017\*/

EXPLAIN PLAN FOR

SELECT TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name) AS Patient, vi.purpose AS Purpose,

TO\_CHAR(pr.first\_name) || ' ' || TO\_CHAR(pr.last\_name) AS Provider, Vi.Visit\_Date AS Visit\_Date,

Med.Name AS Medication, Med.Dose AS Dose, Pres.qty as Quantity FROM all\_patients\_view Pa

INNER JOIN Visits Vi USING (Patient\_ID)

INNER JOIN Provider Pr USING (Provider\_ID)

INNER JOIN Prescriptions Pres USING (Visit\_ID)

INNER JOIN Medications Med USING (Medication\_ID)

WHERE Med.Name = 'Penicillin' AND Vi.Visit\_Date > = '01-Jan-2017';

SELECT \* FROM table(DBMS\_XPLAN.DISPLAY);

/\*This query will show the number of visits a CMS region has had along with the

most recent visit date and last patient seen, sorted by the number of visits.\*/

EXPLAIN PLAN FOR

SELECT Hosp.CMS\_Region as CMS\_Region, COUNT(Vi.Visit\_ID) AS Visit\_Count,

MAX(Vi.Visit\_Date) AS Most\_Recent\_Visit,

TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name) AS Last\_Patient\_Seen

FROM Hospitals Hosp

LEFT JOIN Visits Vi On Vi.Hospital\_CCN\_ID = Hosp.Hospital\_CCN\_ID

LEFT JOIN all\_patients\_view Pa ON Vi.Patient\_ID = Pa.Patient\_ID

GROUP BY Hosp.CMS\_Region, TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name)

ORDER BY Visit\_Count;

SELECT \* FROM table(DBMS\_XPLAN.DISPLAY);

/\*This query will join together the patient and visits table, and then using a subquery

determine how many visits a patient has had, in order by visit count\*/

EXPLAIN PLAN FOR

SELECT DISTINCT Pa.Patient\_ID, TO\_CHAR(pa.first\_name) || ' ' || TO\_CHAR(pa.last\_name) AS Patient,

(SELECT COUNT(Patient\_ID) FROM Visits iVi WHERE iVi.Patient\_ID = Vi.Patient\_ID) as Visit\_Count

FROM all\_patients\_view Pa

LEFT JOIN Visits Vi ON Vi.Patient\_ID = Pa.Patient\_ID

ORDER BY Visit\_Count DESC;

SELECT \* FROM table(DBMS\_XPLAN.DISPLAY);

# **Illustrate 5 Views and Queries**

