EZMedicalGroup.com

Hospital Management System

Anik Muhib
New York City College of Technology

Table of Contents

Summary

In this project I developed a database system for the Hospital Management Systems. I have gone through Planning Phase, Analyses Phase, Design Phase, Development & Implementation and the last Operation & Maintenance. For the Planning Phase I went over the business requirements and make a Project plan and Project Document. Then the Analyses Phase I had to find out all the entities attribute and the super type/subtype entities. Because this hospital has many types of employees also many types of patient for example outpatient and resident patient. This business requirement makes me to create an EER Conceptual Diagram, in a Design Phase I had to implement a database design based on business requirements and also very importantly I had to see the relationships on each entity. Logical Model is another way of looking at the EER Diagram. Then I have created a Normalized Model in Design Phase, which is more break down of a Logical Model. This normalization model reduces redundancy and dependency of data. Development & Implementation phase I had to create a data dictionary and physical model diagram in order to make the blueprint of relational database. Finally, in Operational & Maintenance phase I had to create all the tables and test some queries to see the data flow of the database.

We upgrade this project into version 2.0 where we added new version by creating business reports and improve the performance of full working application. We created business reports Store Procedure for the hospital database. Hospital employees need those reports in order to complete their work. Store Procedure have been created based on business scenario.

We also upgrade this project into version 3.0 and we created Index for the business reports. Index will increase the performance for runtime of business reports. We also created View for the query reports. Some of the Views covered many query reports. In order to increase the performance of Hospital Management Systems Database we had to create the Index and View in this upgraded version.

Objective

The goal will be to develop a database management system based on the business requirement. Therefore, I will design & implement a database application for a Hospital Management System Application named EZMedicalGroup. There are multiple requirements in order to complete the whole process. The application will automate the managing of patients and hospital personnel activities also we can assign patient to the hospital, we can assign volunteers with the physician and employees there are many more things to do. I will design this system to allow hospital employees to conveniently manage the entire hospital management system. I will use **Three-Tiered Web-based Client/Server** – for customers Physicians, nurses and other medical professionals to manage the day activities in the hospital online via a browser. **Two-Tiered Windows-Client Client/Server** – Front line workers such as emergency room administrators, nurses, registration clerks etc.

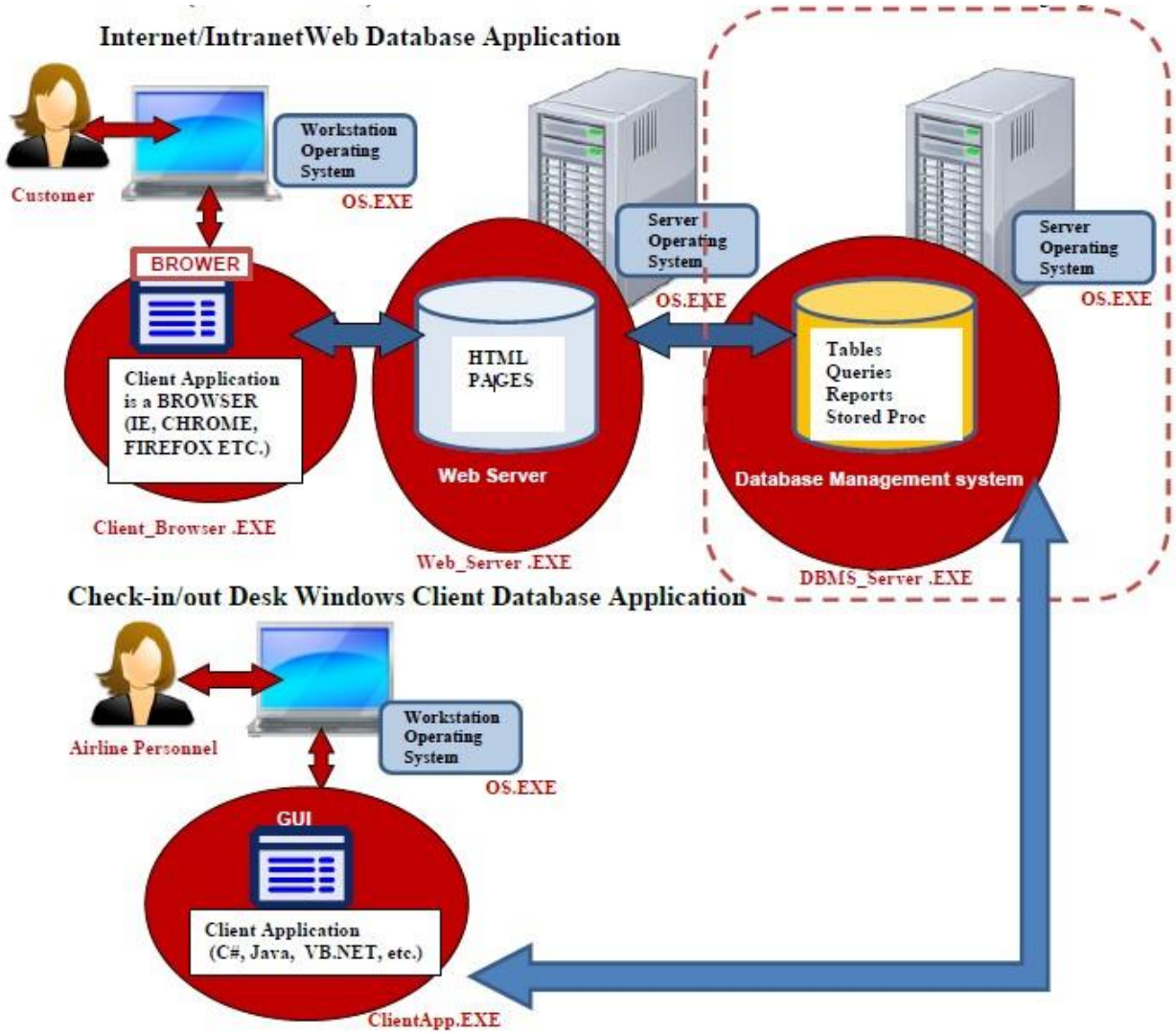
Database Tier – both the Three-tiered Web and Two-tiered Windows client/server will SHARE the same Database tier.

It will also support dozens of major cities around the world. In addition, provide a great user experience in Hospital Reservations.

There are multiple requirements in order to complete the whole process. The Project must need planning analyzing, designing and implementing phases to successfully complete the projects.

Targeted Technical Architecture

The system is a combination of 1) Three-Tiered Web Database Application with a web-based browser front-end for the users that want to make reservations via the internet or travel agency employees that require a browser-based application. In addition, we have an additional architecture: 2) Two-Tiered Windows Database Application, with a custom client front-end to for the application used by check-in desk at the airports and other airline users that require fast performance. Note that both these architecture (Web & Custom client) share the same DBMS back-end data store. Below is the full architecture we are targeting:



This project will focus only on the DBMS or backend design and implementation using a DBMS, NOT on the front-end or client application hosted in the Browser/Web Server or Windows Client (another group will work on this in parallel).

Targeted Methodology & plan

We will use our project management database lifecycle methodology for the project by implementing the Planning Phase, Analysis Phase, and part of the Design Phase for the Hospital Reservations System.

Below are the 5 PHASES & their main deliverables:

- Comments on this Project's requirements based on the above Methodology
- Final design (Physical Model), Development & Implementation are out of the scope of this project. Future projects will address this section.
- In this PROJECT, our goals for this Hospital Management System Reservations System is as follows:
- Model the business requirements using an EER Conceptual Model
- Transform the conceptual model to a logical model and then a normalized logical model that targets a Relational Database Management System.

Below are the detailed requirements and activities of the Planning, Analysis & Design Phases:

Phases	Description
Phase 1: Planning & Discovery 	▪ *Purpose:* - *Develop the plan, understand the business and discover existing information systems.* - *Interview, discover, etc.* ▪ *Database Professional Role:* - *Database/Systems Analyst* - *Business Analyst* ▪ *Deliverables:* 1. *Primary Database Deliverable – Project Document* **Main deliverables in this phase for this PROJECT!**
Phase 2: Analysis 	▪ *Purpose:* 1. *Analyze/derive detailed user requirements for data & develop data model to represent requirements.* 1. *Identify the Business Rules, use correct Naming convention guidelines etc.* 2. *Create a detailed ER Conceptual Data Model (Entity-Relational Model - ER Diagram)* 3. *Create a detailed EER Conceptual Data Model (Enhanced Entity-Relational Model - EER Diagram)* ▪ *Database Professional Role:* - *Database/Systems Analyst*
Phase 3: Design 	▪ *Purpose:* - *Develop a detailed design of database Information System based on all specifications and requirements.* - *Create a Normalized Logical Data Model (Logical Schema)* - *Create a Data Dictionary* - *Create a Physical Data Model Schema Diagram* - *Implement Technical Specification for Performance/Efficiency, Data integrity, Security, Backup/Disaster Recovery, etc.* ▪ *Database Professional Role:* - *Database Analyst – Logical Data Model* - *Database Administrator & Analyst – Physical Data Model* - *All Roles may be involved* ▪ *Activities include:* 1. *Create detailed Logical Data Model (Logical Schema) as follows:* • *TRANSFORM – Convert ER/EER Diagram to Logical Model Diagram* • *NORMALIZATION – Process of breaking down tables with abnormalities to produce smaller, well-structure tables to reduce redundancy and inconsistencies.* 2. *Create detailed Physical Data Model (Physical Schema) as follows:* • *PHYSICAL SCHEMA DIAGRAM – Convert Normalized Logical Model to the Physical Schema Diagram based on DBMS to be used.* • *PHYSICAL TECHNICAL SPECIFICATIONS – Technical specifications for performance, storage & hardware requirements, security, backup & disaster recovery, compliance, etc.* ▪ *Project Management Activities:* • *PM Activities – Create Design Document, Update Plan etc.*

Phases	Description
Phase 4: **Development and Implementation**  	▪ ***Purpose:*** - *Develop & physically implement the design created in design phase.* - *Develop & implement* ***Physical Model*** ***1) Physical Schema Diagram*** *2)* ***Technical Specifications for performance, efficiency, data integrity, security, disaster recover, backup etc. (CST3604 topic)*** ▪ ***Database Professional Role:*** - *Database Developer* - *Database Administrator* ▪ ***Activities include:*** 1. *Install & configure Database Management System (DBMS)* 2. *Write, test & execute all program/scripts that access, created or modify the database (use SQL, Programming Language or special database processing language)* 3. *Write, test & execute all program/scripts for required reports, User-views or displays, graphs etc. (use SQL, Programming Language or special database processing language)* 4. *Unit & Integration Testing* 5. *Load/Import data into DBMS* 6. *Finalize documentation*

Phases	Description
Phase 4: **Operations & Maintenance**  	▪ ***Purpose:*** - *Maintain, operate & backup the Information System* - *Repeat entire Methodology as changes are required* ▪ ***Database Professional Role:*** - *Database Analyst – Re-design for new business requirements & errors in database design* - *Database Developer – Re-design & implement for new business requirements & errors in database design* - *Database Administrator – Maintain, operate, backup & improve performance*

Business Requirements

Hospital Management System Business & Technical Detailed Requirements

- A Business Analyst was hired by Mr. Rodriguez to compile the list or requirements based on the results of interviews and conversations with the various business stakeholders.
- Below are the requirements captured by the Business Analyst:

- This EZMedicalGroup.com Hospital Management System is designed to allow hospital physicians, nurses and employees to manage outpatients and resident patient hospital activities.
- The application is to support our hospitals in major cities in the US and around the world. Both the client application and the website need to capture the following information:

- As a large service organization, EZ Medical Group depends on four major groups of persons for its continued success: employees, physicians, patients, and volunteers. Also note that a small number of persons in the hospital community do not belong to any of these four groups. A particular person may belong to two (or more) of these groups at a given time, for example, a volunteer or employee may also be a patient at the hospital at some point in time or an employee also be volunteer on their spare time.

- For our patients we need to store their person ID or identifier for that person, name composed of first, middle & last name. Also, address which include the following components: address, city, state, zip code and country. In addition, gender, contact phone & email. Additional attributes of patient include first contact date or date when patient first engaged with the hospital, emergency contact name, emergency contact number and the final attribute for a patient is a credit card. We accept major credit cards for customers to pay for their co-pay visit or services rendered. Therefore we need to capture the following credit card information: credit card number, credit card merchant name such as AMX, Visa, Master Card etc., credit card owner name and credit card expiration date. Note that a patient can have more than one credit card, therefore, a patient can use any of the many types of major credit card he or she owns to pay for services or co-pay. In addition, the credit card can be owned by the patient, spouse or corporation who co-owns the credit card.

- For our physicians we need to store their person ID or identifier, name composed of first, middle & last name. Also, address composed of the following components: address, city, state, zip code and country. In addition, gender, contact phone & email.
  Other attributes of physicians are pager number and a DEA number (a physician needs a DEA registration number from the Drug Enforcement Administration to be able to prescribe controlled substances).

- We have four main types of employees we need to track: RN Nurses, LPN Nurses, staff and technicians. In addition, to other types of employees that are of no interest now. For all four types of employees, we required the following information: person ID or identifier, name composed of first, middle & last name. Also, address which include the following components: address, city, state, zip code and country. In addition, gender, contact phone & email, in addition to the date hired.

- Finally, for our volunteer we are interested in capturing their person ID, name composed of first, middle & last name, address which include the following components: address, city, state, zip code and country, gender, contact phone, email, their many skills, interest ID which is an identifier we use to represent different interest , and finally interest description. The table below shows example of the interest id we use and matching description

Interest ID	*Interest Description*
1	Emergency Room
2	Patient Ward
3	Administrative work
4	Radiology work
Etc..	Etc..

- A care center is a treatment location within our hospitals. Example of care centers are maternity, emergency room, multiple sclerosis center, ambulatory, outpatient surgery, diagnostic services etc. Each care center has a care center ID, care center name and care center address.

- We have two types of patients, outpatient and resident patient. for both outpatient & resident patient we need to store their person ID, name composed of first, middle & last name, address which include the following

components: address, city, state, zip code and country. In addition, gender, contact phone & email. Other attributes patient includes first contact date or date when patient first engaged with the hospital, emergency contact name, emergency contact number, credit cards information: credit card number, credit card merchant name, credit card owner name and credit card expiration date.

- What distinguish an outpatient is the attribute checkback date since an outpatient may come in for many reasons, including routine examinations at an outpatient care center. Each outpatient is scheduled for zero or more visits. A visit has several attributes: visit number which is a unique ID, visit date and visit time. Note that an instance of visit cannot exist without an outpatient instance. For resident patients, we need to store the date admitted and the date discharged since resident patients stay in the hospital for a period of time. In our hospital we only have these two types of patients' outpatient and resident patient no other type of patient exists. A patient cannot be an outpatient and resident patient at the same time.

- For resident patients, we need to store the date admitted and the date discharged since resident patients stay in the hospital for a period of time. In our hospital we only have these two types of patients' outpatient and resident patient no other type of patient exists. A patient cannot be an outpatient and resident patient at the same time.

- Each resident patient must be assigned a bed. Because the hospitals don't always fill all its beds, a bed may or may not have a resident patient assigned to it at a given time.

- A care center can contain one or more beds (up tot any number), but there are care center that do not have beds, and a bed must belong to at least one care center. Note that as with visit, without a care center a bed cannot exist. For our beds, we need to store their Bed ID which is compose of two components: Bed number and room number. In addition, we use a code to identify the bed type or bed type ID and a description for each bed type ID. The table below lists some of the bed type IDs and their descriptions:

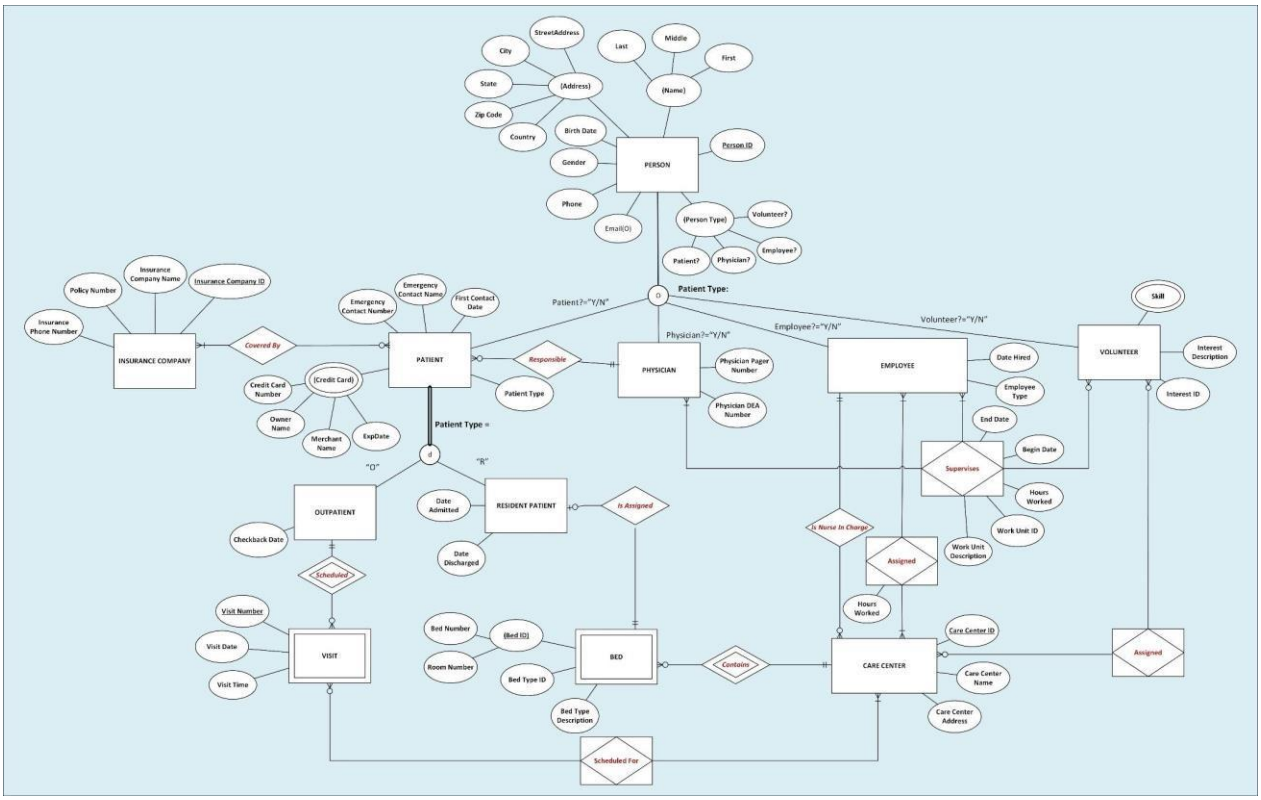
Bed Type ID	Description
1	Standard Manual Bed
2	Electric Automatic Bed
3	Clinitron Air Bed
4	Stretcher
Etc..	Etc..

- Each patient has one and only one physician responsible for that patient. A given physician may not be responsible for a patient at a given time or may be responsible for one or more patients.
- We also need to capture the insurance information for all patients. For insurance company, we need to store the insurance company ID, insurance company name, policy number, insurance company phone number. A patient can be covered by many insurance companies and insurance companies can covered many patients. Nevertheless, a patient must be covered by at least one insurance company.
- Our employees must be assigned to a care center. Employees can be assigned to many care centers and many care centers assigned employees, but a care center must have at least one employee assigned. When an employee is assigned to a care center, we need to capture the hours worked.
- Each care center is supervised by an employee who is the nurse in charge. A nurse in charge can supervise none or more care center, but a care center can only be supervised by one nurse in charge.
- **Each volunteer is supervised by an employee and a physician. Supervision requires participation of all three for a volunteer to be able to work. Note that not all employees and physicians supervised volunteer, and a volunteer can be supervised by one or more physicians and employee, but a volunteer must be supervised. When an employee(s) and physician(s) supervise a volunteer, we need to keep track of the begin date or date volunteer began to work and the end date of the job. In addition, the number of hours worked by the volunteer, the Unit ID or code we use for identifying a care center unit for volunteers to work and the unit description. The table below shows example of the code we use and the description:**

Unit ID	Unit Description
1	Emergency Room
2	Outpatient Care
3	Radiology
4	Pediatrics
Etc..	Etc..

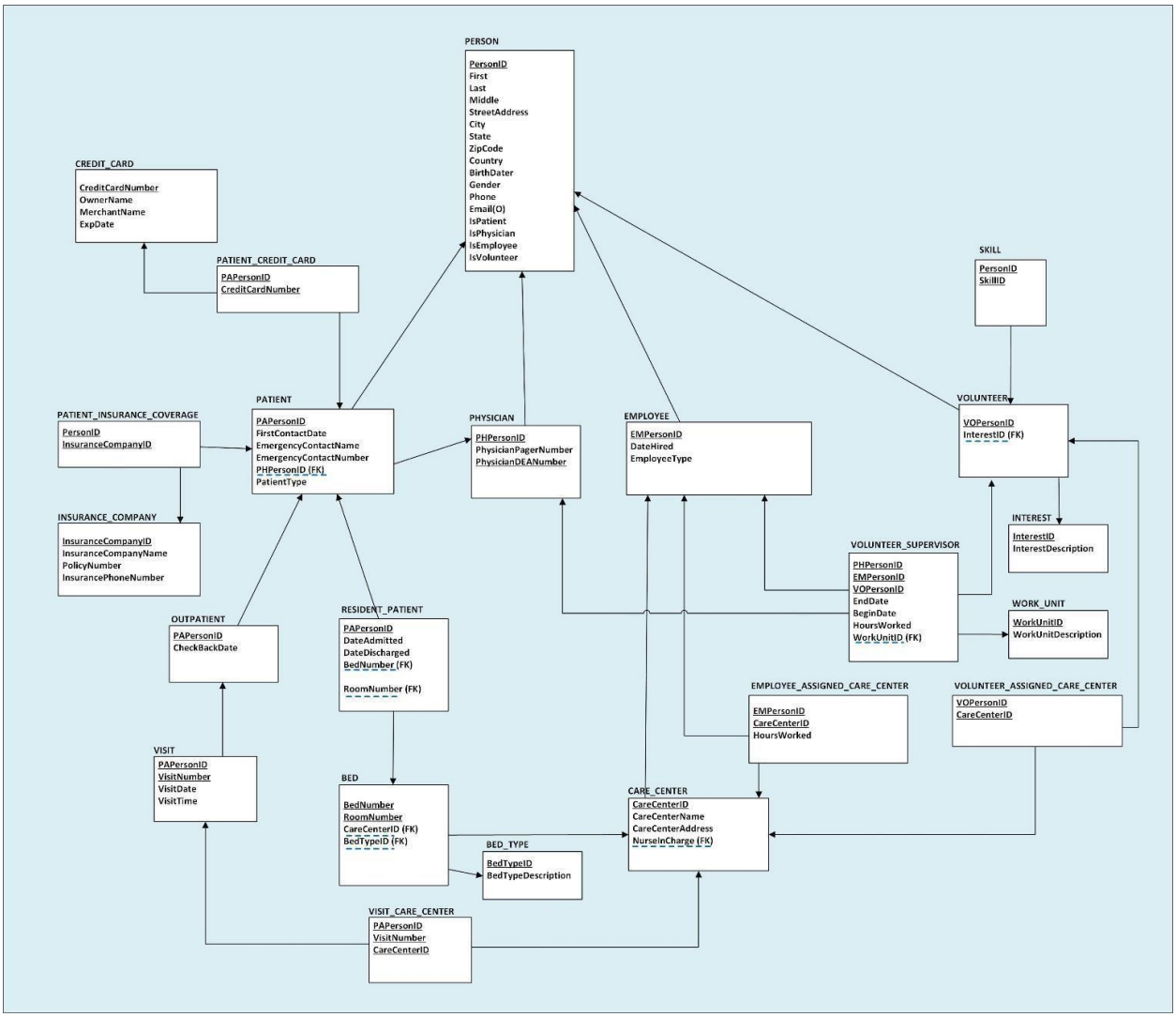
EER Conceptual Model

I have created this EER Conceptual Model based on the business requirements. EER Model is the deliverable for the Analyses Phase. This EER Model clearly explained all the entities and its attributes. Also, shows the relationships between the entities.



Normalized Logical Model

I Normalized this database design technique which organizes tables in a manner that reduces redundancy and dependency of data. I describe the data in as much detail as possible, without regard to how they will be physical implemented in the database. Features of this logical data model includes all entities and relationships among them. All attributes for each entity are specified. It divides larger tables to smaller tables and links them using relationships. I have gone through Database Normal Forms. 1NF, 2NF and 3NF Rules. This is the 3NF shows below.



Physical Model Data Dictionary

I have created a Physical Model Data Dictionary where I have collection of descriptions of the data objects or items in a data model for my benefit in future necessary steps for example to create a Physical Model Schema Diagram also, this data dictionary will help me to create table. First step in analyzing phase I identify each object and its relationship to other objects. This process is called data modeling and results in a picture of object relationships. After each data object or item is given a descriptive name, its relationship is described. In this data dictionary I have 22 tables and their attribute name, data type, oracle data type, if its required, constraints and description.

Person					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Person ID. Unique identifier for a person instance
First	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	First name of a person
Last	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	Last name of a person
Middle	Variable Character	VARCHAR2 (20)			Middle name of a person
StreetAddress	Variable Character	VARCHAR2 (80)	Y	NOT-NULL	Street name
City	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	City/Borough name
State	Variable Character	CHAR (2)	Y	NOT-NULL	State name
ZipCode	Number	NUMBER (5)	Y	NOT-NULL	Zip Code
Country	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	Country name
BirthDates	Date	DATE	Y	NOT-NULL	Date of Birth MM/DD/YY
Gender	Variable Character	CHAR (1)	Y	NOT-NULL	Person's gender (M/F)

Phone	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	Person's phone number
Email(0)	Variable Character	VARCHAR2 (20)			Person's email (Optional)
IsPatient	Number	VARCHAR2 (3)	Y	NOT-NULL	Unique identifier for a Patient
IsEmployee	Number	VARCHAR2 (3)	Y	NOT-NULL	Unique identifier for Employee
IsVolunteer	Number	VARCHAR2 (3)	Y	NOT-NULL	Unique identifier for Volunteer
IsPhysician	Number	VARCHAR2 (3)	Y	NOT-NULL	Unique identifier for Volunteer

Physician					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PHPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Physician's unique identification
PhysicianPageNumber	Number	NUMBER (4)	Y	NOT-NULL	Physician's page number
PhysicianDEANumber	Number	VARCHAR2(20)	Y	NOT-NULL	Physician's DEA number unique Identifier

Employee					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
EMPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Employee's unique identification
DateHired	Date	DATE	Y	NOT-NULL	Hire date MM/DD/YY
EmployeeType	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	Type of employee

Patient					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PAPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Patient's unique identification
FirstContactDate	Date	DATE	Y	NOT-NULL	Patient's contact date MM/DD/YY
EmergencyContactName	Variable Character	VARCHAR2 (50)	Y	NOT-NULL	Emergency contact name
EmergencyContactNumbe	Variable Character	VARCHAR2 (10)	Y	NOT-NULL	Emergency contact number
PHPersonID(FK)	Number	NUMBER (4)	Y	Foreign key NOT-NULL	Physician's Unique Identification
PatientType	Variable Character	VARCHAR2 (2)	Y	NOT-NULL	Resident/Out patient

Interest					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
InterestID	Variable Character	VARCHAR2 (2)	Y	Primary key NOT-NULL	Interest unique identification number
InterestDescription	Character	CHAR (50)	Y	NOT-NULL	Interest description

Volunteer					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
VOPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Volunteer's unique identification
InterestID(FK)	Number	NUMBER (2)	Y	Foreign key NOT-NULL	Foreign key of Interest ID

Credit_Card					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
CreditCardNumber	Number	NUMBER (16)	Y	Primary key NOT-NULL	Patient's credit card number
OwnerName	Variable Character	VARCHAR2 (50)	Y	NOT-NULL	Credit card holder name
MerchanName	Variable Character	VARCHAR2 (50)	Y	NOT-NULL	Product name
ExpDate	Date	DATE	Y	NOT-NULL	Credit card expiration date MM/DD/YY

Patient_Credit_Card					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PAPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Patient's unique identification
CreditCardNumber	Number	NUMBER (16)	Y	Primary key NOT-NULL	Patient's credit card number

Insurance_Company					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
InsuranceCompanyID	Variable Character	VARCHAR2 (20)	Y	Primary key NOT-NULL	Insurance company's unique identification
InsuranceCompanyName	Variable Character	VARCHAR2 (50)	Y	NOT-NULL	Insurance company's name
PolicyNumber	Variable Character	VARCHAR2 (50)	Y	NOT-NULL	Policy number
InsurancePhoneNumber	Variable Character	VARCHAR2 (10)	Y	NOT-NULL	Insurance phone number

Patient_Insurance_Coverage					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PAPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Patient's unique identification
InsuranceCompanyID	Variable Character	VARCHAR2 (20)	Y	Primary key NOT-NULL	Insurance company's unique identification

OutPatient					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PAPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Patient's unique identification
CheckBackDate	Date	DATE	Y	NOT-NULL	Check back date MM/DD/YY

Visit					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PAPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Patient's unique identification
VisitNumber	Number	NUMBER (20)	Y	Primary key NOT-NULL	Visit Number
VisitDate	Date	DATE	Y	NOT-NULL	Visit Date DD/MM/YY
VisitTime	Number	NUMBER (4)	Y	NOT-NULL	Visit time

Care_Center					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
CareCenterID	Variable Character	VARCHAR2 (20)	Y	Primary key NOT-NULL	Care center's unique identification
CareCenterName	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	Care center's name
CareCenterAddress	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	Care center's address
NurseInCharge(FK)	Number	NUMBER (4)	Y	Foreign key NOT-NULL	Employee type nurse in charge

Visit_Care_Center					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
OPAPersonID	Number	NUMBER(4)	Y	Primary key NOT-NULL	Person's unique identification
VisitNumber	Number	NUMBER (20)	Y	Primary key NOT-NULL	Visit number
CareCenterID	Variable Character	VARCHAR2 (20)	Y	Primary key NOT-NULL	Care center's unique identification

Bed_Type					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
BedTypeID	Number	NUMBER (1)	Y	Primary key NOT-NULL	Bed type's unique identification
BedTypeDescription	Variable Character	VARCHAR2 (50)	Y	NOT-NULL	Bed type's description

Bed					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
BedNumber	Number	NUMBER (20)	Y	Primary key NOT-NULL	Bed number
RoomNumber	Variable Character	VARCHAR2 (4)	Y	Primary key NOT-NULL	Room number
CareCenterID	Variable Character	VARCHAR2 (20)	Y	Primary key NOT-NULL	Care center ID
BedTypeID(FK)	Number	NUMBER (1)	Y	Foreign key NOT-NULL	Bed type ID

ResidentPatient					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PAPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Patient's unique identification
DateAdmitted	Date	DATE	Y	NOT-NULL	Admition date
DateDischarged	Date	DATE	Y	NOT-NULL	Discharge date
BedNumber(FK)	Number	NUMBER (4)	Y	Foreign key NOT-NULL	Bed number
RoomNumber(FK)	Variable Character	VARCHAR2 (4)	Y	Foreign key NOT-NULL	Room number

Employee_Assigned_Care_Center					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
EMPersonID	Number	NUMBER (4)	Y	Composite key NOT-NULL	Employee's unique identification
CareCenterID	Variable Character	VARCHAR2 (20)	Y	Composite key NOT-NULL	Care center's ID
HoursWorked	Number	NUMBER (3)	Y	NOT-NULL	Working hours

Work_Unit					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
WorkUnitID	Number	NUMBER (2)	Y	Primary key NOT-NULL	Work unit identification number
WorkUnitDescription	Variable Character	VARCHAR2 (20)	Y	NOT-NULL	Work unit description

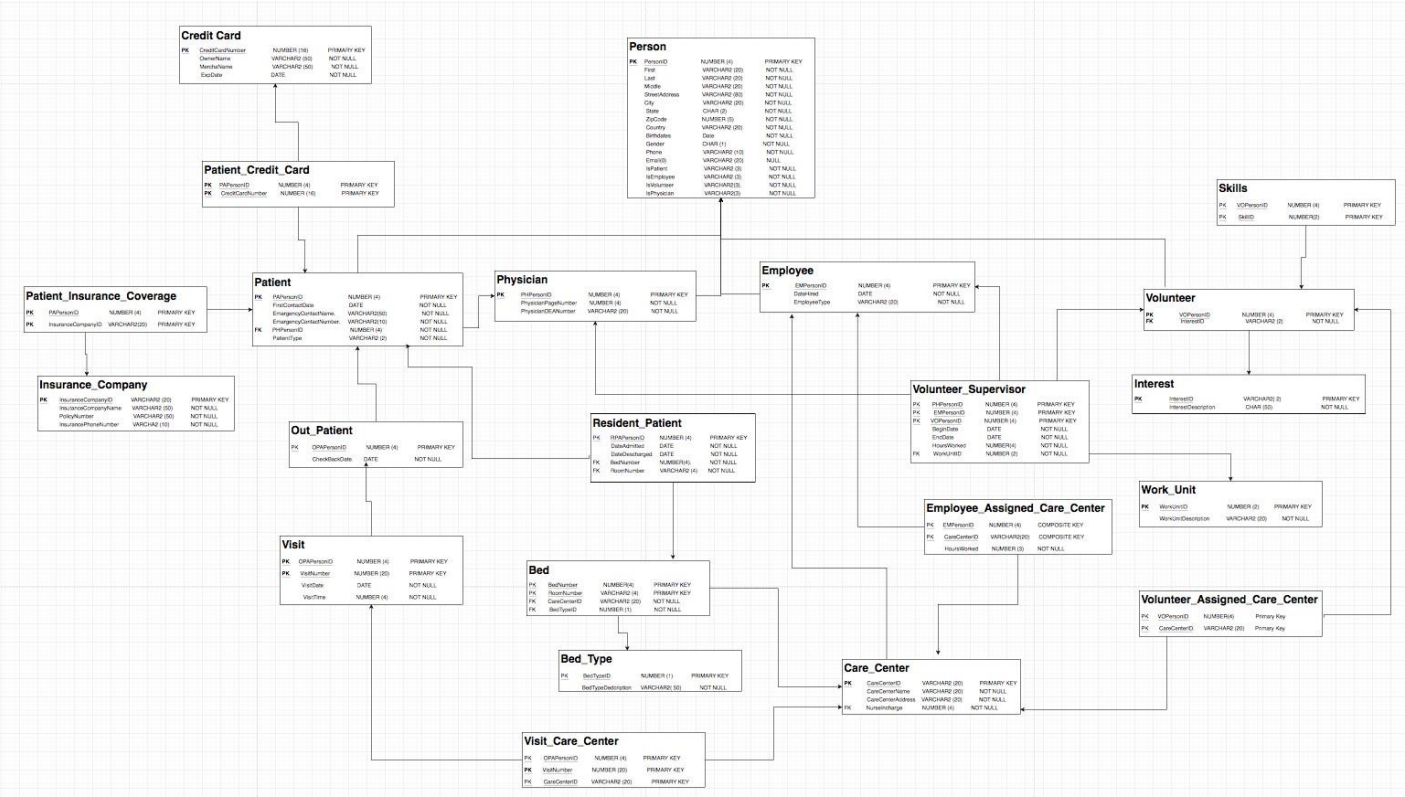
Volunteer_Supervisor					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PHPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Phisician's unique identification
EMPersonI	Number	NUMBER (4)	Y	Primary key NOT-NULL	Employee's unique identification
VOPersonI	Number	NUMBER (4)	Y	Primary key NOT-NULL	Volunteer's unique identification
BeginDate	Date	DATE	Y	NOT-NULL	Begin date MM/DD/YY
EndDate	Date	DATE	Y	NOT-NULL	End date MM/DD/YY
HoursWorked	Number	NUMBER (4)	Y	NOT-NULL	Working hours
WorkUnitID(FK)	Number	NUMBER (2)	Y	Foreign key NOT-NULL	Work unit identification number

Skills					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
PersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Person's identification number
SkillID	Number	NUMBER (2)	Y	Primary key NOT-NULL	Skill's identification number

Volunteer_Assigned_Care_Center					
Attribute Name	Data Type	Oracle Data Type	Required?	Constraints	Description/ Purpose
VOPersonID	Number	NUMBER (4)	Y	Primary key NOT-NULL	Volunteer unique identification number
CareCenterID	Variable Character	VARCHAR2 (20)	Y	Primary key NOT-NULL	Care center unique identification number

Physical Model Schema Diagram

Physical model schema diagram represents the actual design blueprint of a relational database. It represents how data should be structured and related in a specific DBMS so we can see how the model will be built in the database. In this schema I have 22 table in this database model and it shows all table structures, including column name, column data type, column constraints, primary key, foreign key, and relationships between tables. This schema diagram represents a detailed picture of the entities needed for the hospital database, which is going to be helpful to see the entire database and their relationships.



Development & Implementation

Tables are the basic building blocks of any database. They hold the rows and columns of our data. I have created 22 tables with their attributes name, column data type and column constrain. I had to create to patent table first or the table who's does not have any dependency. After treating that I have cheated the associate tables, where the foreign keys are used as a composite key. Not every column has the same data type, it depends on the column. Also, the length of the data is different. Its based on the column type.

```
CREATE TABLE PERSON
(
 PersonID      NUMBER (4)    PRIMARY KEY,
 First         VARCHAR2 (20) NOT NULL,
 Last          VARCHAR2 (20) NOT NULL,
 Middle        VARCHAR2 (20) NOT NULL,
 StreetAddress VARCHAR2 (80) NOT NULL,
 City           VARCHAR2 (20) NOT NULL,
 State          CHAR(2)       NOT NULL,
 ZipCode        NUMBER(5)     NOT NULL,
 Country        VARCHAR2 (20) NOT NULL,
 BirthDates     DATE          NOT NULL,
 Gender         CHAR(1)  NOT NULL,
 Phone          VARCHAR2 (10) NOT NULL,
 Email          VARCHAR2 (20) NULL,
 IsPatient     VARCHAR2 (3)  NOT NULL,
 IsPhysician   VARCHAR2 (3)  NOT NULL,
 IsEmployee    VARCHAR2 (3)  NOT NULL,
 IsVolunteer   VARCHAR2 (3)  NOT NULL
);
```

```
CREATE TABLE PHYSICIAN
(
     PHPersonID                    NUMBER (4)                PRIMARY KEY,
        PhysicianPageNumber        NUMBER (4)                 NOT NULL,
       PhysicianDEANumber          VARCHAR2 (20)
NOT NULL,
    CONSTRAINT PHPersonID_FK
    FOREIGN KEY (PhPersonID)
    REFERENCES PERSON(PersonID)

    );
```

```
 CREATE TABLE EMPLOYEE
(
EMPersonID            NUMBER (4)    NOT NULL,
 DateHired             DATE          NOT NULL,
 EmployeeType          VARCHAR2 (20) NOT NULL,

 CONSTRAINT  EMPerson_ID_PK    PRIMARY KEY (EMPerson_ID),

 CONSTRAINT  EMPerson_ID_FK
 FOREIGN KEY (EMPerson_ID)
 REFERENCES  PERSON(Person_ID)
);



CREATE TABLE PATIENT
(
          PAPersonID                      NUMBER (4)                      NOT NULL,
          FirstContactDate                DATE                             NOT NULL,
    EmargencyContactName            VARCHAR2(50)                    NOT NULL,
EmargencyContactNumber          VARCHAR2(10)                  NOT NULL,
          PHPersonID                      NUMBER (4)                       NOT NULL,
          PatientType                     VARCHAR2 (2)                     NOT NULL,
       CONSTRAINT PAPersonID_PK PRIMARY KEY(PAPersonID),

       CONSTRAINT PAPersonID_FK
       FOREIGN KEY (PAPersonID)
       REFERENCES PERSON(PersonID),

       FOREIGN KEY (PHPersonID)
       REFERENCES PHYSICIAN(PHPersonID)
       );



CREATE TABLE VOLUNTEER
(
    VOPersonID                  NUMBER (4)                   NOT NULL,
    InterestID                  VARCHAR2(2)                 NOT NULL,

    CONSTRAINT VOPersonID_PK PRIMARY KEY(VOPersonID),
    CONSTRAINT VOPersonID_FK
    FOREIGN KEY (VOPersonID)
    REFERENCES PERSON(PersonID),

    FOREIGN KEY(InterestID)
    REFERENCES INTEREST(InterestID)

    );
```

```
 CREATE TABLE SKILL
(
        VOPersonID              NUMBER (4)                NOT NULL,
       SkillID                NUMBER(2)              NOT NULL,


        PRIMARY KEY(VOPersonID, SkillID),


        FOREIGN KEY (VOPersonID)
        REFERENCES VOLUNTEER(VOPersonID)

        );
```

```
CREATE TABLE CREDIT_CARD
(
       CreditCardNumber              NUMBER (16)             PRIMARY KEY, OwnerName
       VARCHAR2 (50)          NOT NULL,
       MerchaName                    VARCHAR2 (50)          NOT NULL,
       ExpDate                       DATE                    NOT NULL

             );
```

```
CREATE TABLE PATIENT_CREDIT_CARD
(
       PAPersonID                NUMBER (4)              NOT NULL,
       CreditCardNumber         NUMBER (16)              NOT NULL,

PRIMARY KEY(PAPersonID, CreditCardNumber),

FOREIGN KEY(PAPersonID)
REFERENCES PATIENT(PAPersonID),

FOREIGN KEY(CreditCardNumber)
REFERENCES CREDIT_CARD(CreditCardNumber)
);
```

```
CREATE TABLE INSURANCE_COMPANY
(
       InsuranceCompanyID            VARCHAR2 (20)            PRIMARY KEY,
       InsutanceCompanyName          VARCHAR2 (50)               NOT NULL,
       PolicyNumber                 VARCHAR2 (50)             NOT NULL,
       InsurancePhoneNumber         VARCHAR2 (10)             NOT NULL

             );

             PATIENT_INSURANCE_COVERAGE

PAPersonID                    NUMBER (4)         NOT NULL,
InsuranceCompanyID           VARCHAR2(20)        NOT NULL,

PRIMARY KEY(PAPersonID, InsuranceCompanyID),

FOREIGN KEY(PAPersonID)
REFERENCES PATIENT(PAPersonID),

FOREIGN KEY (InsuranceCompanyID)
REFERENCES  INSURANCE_COMPANY(InsuranceCompanyID)

);



CREATE TABLE OUT_PATIENT
(
OPAPersonID             NUMBER (4)             NOT NULL,
CheckBackDate           DATE                    NOT NULL,

CONSTRAINT OPAPersonID_PK PRIMARY KEY(OPAPersonID),

CONSTRAINT OPAPersonID_FK
FOREIGN KEY(OPAPersonID)
REFERENCES PATIENT(PAPersonID)
);



CREATE TABLE VISIT
(
OPAPersonID                NUMBER (4)           NOT NULL,
VisitNumber               NUMBER (20)           NOT NULL,
VisitDate                 DATE                   NOT NULL,
VisitTime                 NUMBER (4)            NOT NULL,

PRIMARY KEY(OPAPersonID, VisitNumber),

FOREIGN KEY(OPAPersonID)
REFERENCES OUT_PARIENT(OPAPersonID) );
```

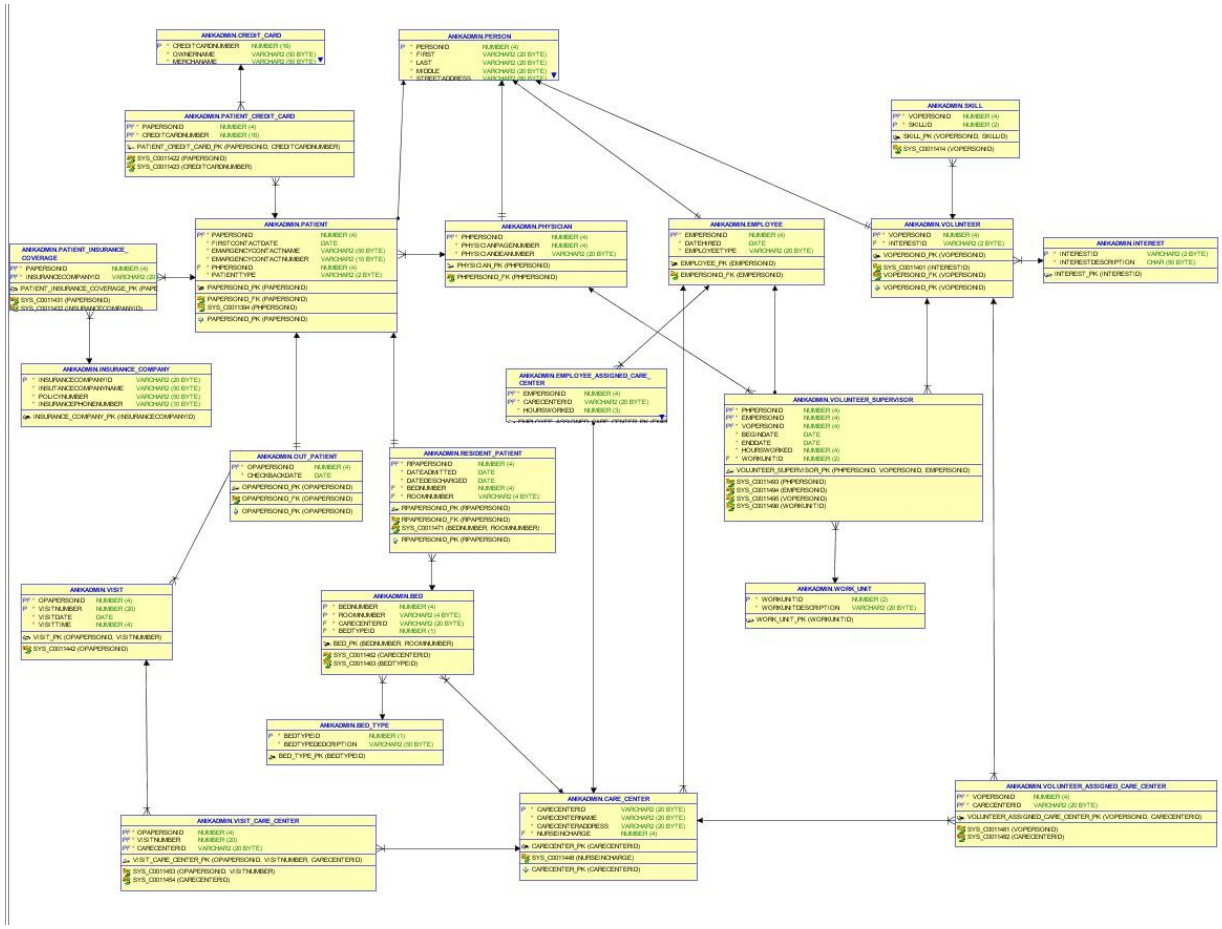
```
CREATE TABLE CARE_CENTER (

CareCenterID              VARCHAR2 (20)          NOT NULL,
CareCenterName            VARCHAR2 (20)           NOT NULL,
CareCenterAddress         VARCHAR2 (20)           NOT NULL,
NurseIncharge             NUMBER (4)              NOT NULL,

CONSTRAINT CareCenter_PK PRIMARY KEY(CareCenterID),
FOREIGN KEY(NurseIncharge)
REFERENCES EMPLOYEE(EMPersonID)
);


CREATE TABLE VISIT_CARE_CENTER
(
OPAPersonID             NUMBER (4)              NOT NULL,
VisitNumber             NUMBER (20)             NOT NULL,
CareCenterID           VARCHAR2 (20)            NOT NULL,


PRIMARY KEY(OPAPersonID, VisitNumber, CareCenterID),
FOREIGN KEY(OPAPersonID,VisitNumber)
REFERENCES VISIT(OPAPersonID, VisitNumber),
FOREIGN KEY (CareCenterID)
REFERENCES CARE_CENTER(CareCenterID)

);


CREATE TABLE BED_TYPE
(
BedTypeID               NUMBER(1)             PRIMARY KEY,
BedTypeDedcription      VARCHAR2(50)             NOT NULL
);


CREATE TABLE BED
(
BedNumber               NUMBER (4)          NOT NULL,
RoomNumber              VARCHAR2 (4)        NOT NULL,
CareCenterID             VARCHAR2 (20)      NOT NULL,
BedTypeID                NUMBER (1)         NOT NULL,

PRIMARY KEY(BedNumber, RoomNUmber),
FOREIGN KEY(CareCenterID)
REFERENCES CARE_CENTER(CareCenterID),
FOREIGN KEY (BedTypeID)
REFERENCES  BED_TYPE(BedTypeID)

);
```

```
CREATE TABLE RESIDENT_PATIENT (

RPAPersonID          NUMBER (4)          NOT NULL,
DateAdmitted           DATE               NOT NULL,
DateDescharged        DATE                NOT NULL,
BedNumber            NUMBER(4)           NOT NULL,
RoomNumber           VARCHAR2 (4)       NOT NULL,

CONSTRAINT RPAPersonID_PK PRIMARY KEY(RPAPersonID),

CONSTRAINT RPAPersonID_FK
FOREIGN KEY(RPAPersonID)
REFERENCES PATIENT(PAPersonID),

FOREIGN KEY(BedNumber, RoomNumber)
REFERENCES BED(BedNumber, RoomNumber)

);


CREATE TABLE EMPLOYEE_ASSIGNED_CARE_CENTER
(
EMPersonID           NUMBER (4)          NOT NULL,
CareCenterID         VARCHAR2(20)        NOT NULL,
HoursWorked          NUMBER (3)         NOT NULL,

PRIMARY KEY(EMPErsonID, CareCenterID),

FOREIGN KEY(EMPersonID)
REFERENCES EMPLOYEE(EMPersonID),
FOREIGN KEY (CareCenterID)
REFERENCES CARE_CENTER( CareCenterID)

);


CREATE TABLE VOLUNTEER_ASSIGNED_CARE_CENTER
(
VOPersonID           NUMBER(4)           NOT NULL,
CareCenterID       VARCHAR2 (20)        NOT NULL,

PRIMARY KEY (VOPersonID, CareCenterID),

FOREIGN KEY(VOPersonID)
REFERENCES VOLUNTEER(VOPersonID),

FOREIGN KEY (CareCenterID)
REFERENCES CARE_CENTER(CareCenterID)

);
```

```
CREATE TABLE WORK_UNIT (

 WorkUnitID           NUMBER (2)    PRIMARY KEY,
 WorkUnitDescription  VARCHAR2 (20) NOT NULL
);


CREATE TABLE VOLUNTEER_SUPERVISOR
(
 PHPersonID            NUMBER (4)    NOT NULL,
 EMPersonID            NUMBER (4)    NOT NULL,
 VOPersonID            NUMBER (4)    NOT NULL,
 BeginDate             DATE          NOT NULL,
 EndDate               DATE          NOT NULL,
 HoursWorked           VARCHAR2 (4)  NOT NULL,
 WorkUnitID            NUMBER (2)    NOT NULL,

 PRIMARY KEY(PHPersonID,VOPersonID,EMPersonID),

 FOREIGN KEY (PHPersonID)
 REFERENCES  PHYSICIAN(PHPersonID),

 FOREIGN KEY (EMPersonID)
 REFERENCES  EMPLOYEE(EMPersonID),

 FOREIGN KEY (VOPersonID)
 REFERENCES  VOLUNTEER(VOPersonID),

 FOREIGN KEY (WorkUnitID)
 REFERENCES  WORK_UNIT(WorkUnitID)
);
```

Development & Implementation Physical Schema Diagram

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data. Notice this schema diagram identical with the Normalize Logical Model Diagram. We found this diagram after creating all the require tables for the hospital database. This schema diagram defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. This schema diagram will help me to program the database and make it useful.



Development & Implementation Testing

INSERT STATEMENTS for person table

I Created 10 INSERT QUERY as follows to the PERSON table.

```
Insert Into PERSON values (1111, 'Joe','Smith','S', '111 Jay Street','Brooklyn','NY',11201,'BK','01-JAN-71','M',3473456653,'JSmith@xyz.com', 'N','N','N','Y');
Insert Into PERSON values (2222, 'Angel','Rodriguez' ,'K', '22 Flatbush Ave','Brooklyn','NY',11202,'BK','01-MAY-13','M',3477474747,'Arod@xyz.com', 'N','N','N','Y');
Insert Into PERSON values (3333, 'SIMON','ANDY' ,'JON', '150 brick st','New York','NY',11012,'USA','01-may-16','F',6462222222,'SABADBr@xyz.com', 'N','N','N','Y');
Insert Into PERSON values (4444, 'RONY','AMY' ,'KELLY', '157 br00K st','New York','NY',11014,'USA','01-may-17','M',3472121212,'SABAGGGr@xyz.com', 'N','N','N','Y');
Insert Into PERSON values (5555, 'AMUN','DICKY' ,'ABU', '158 br00K st','New York','NY',11015,'USA','01-may-15','F',9293445566,'VBVBVBV@xyz.com', 'N','N','N','Y');
Insert Into PERSON values (6666, 'Smith','Arnold' ,'Junior', '250 jay st','New York','NY',11013,'USA','01-APR-21','M',1111111111,'SJumior@etz.com', 'Y','N','N','N');
Insert Into PERSON values (7777, 'JOHN','RON' ,'POL', '350 banj st','New York','NY',11812,'USA','01-jun-12','F',3333333333,'Ssakfaklsr@etz.com', 'Y','N','N','N');
Insert Into PERSON values (8888, 'ANDY','JACK' ,'SAM', '100 brke st','New York','NY',11222,'USA','01-may-13','F',4444444444,'SsakfDSGDGsr@etz.com', 'Y','N','N','N');
Insert Into PERSON values (9999, 'MURRY','HAMR' ,'JEAMS', '225 BLAKE st','New York','NY',11225,'USA','02-APRIL-12','M',5555555555,'SDASGDGsr@etz.com', 'Y','N','N','N');
Insert Into PERSON values (1000, 'BENJY','DONALD' ,'KEN', '33 PITKIN st','New York','NY',11208,'USA','01-MAY-13','M',4747474747,'HHHHHH@etz.com', 'Y','N','N','N');
```

This is the PERSON table with 10 new record that I executed

	PERSONID	FIRST	LAST	MIDDLE	STREETADDRESS	CITY	STATE	ZIPCODE	COUNTRY	BIRTHDATES	GENDER	PHONE	EMAIL	ISPATIENT	ISEMPLOYEE	ISVOLUNTEER	ISPHYSICIAN
1	7777	JOHN	RON	POL	350 banj st	New York	NY	11812	USA	01-JUN-12	F	3333333333	Ssakfaklsr@et...	Y	N	N	N
2	8888	ANDY	JACK	SAM	100 brke st	New York	NY	11222	USA	01-MAY-13	F	4444444444	SsakfDSGDGsr@...	Y	N	N	N
3	1111	Joe	Smith	S	111 Jay Street	Brooklyn	NY	11201	BK	01-JAN-71	M	3473456653	JSmith@xyz.com	N	N	N	Y
4	2222	Angel	Rodriguez	K	22 Flatbush Ave	Brooklyn	NY	11202	BK	01-MAY-13	M	3477474747	Arod@xyz.com	N	N	N	Y
5	3333	SIMON	ANDY	JON	150 brick st	New York	NY	11012	USA	01-MAY-16	F	6462222222	SABADBr@xyz.com	N	N	N	Y
6	4444	RONY	AMY	KELLY	157 br00K st	New York	NY	11014	USA	01-MAY-17	M	3472121212	SABAGGGr@xyz.com	N	N	N	Y
7	5555	AMUN	DICKY	ABU	158 br00K st	New York	NY	11015	USA	01-MAY-15	F	9293445566	VBVBVBV@xyz.com	N	N	N	Y
8	6666	Smith	Arnold	Junior	250 jay st	New York	NY	11013	USA	01-APR-21	M	1111111111	SJumior@etz.com	Y	N	N	N
9	9999	MURRY	HAMR	JEAMS	225 BLAKE st	New York	NY	11225	USA	02-APR-12	M	5555555555	SDASGDGsr@etz...	Y	N	N	N
10	1000	BENJY	DONALD	KEN	33 PITKIN st	New York	NY	11208	USA	01-MAY-13	M	4747474747	HHHHHH@etz.com	Y	N	N	N

INSERT STATEMENTS for Credit Card table

I Created 5 INSERT QUERY as follows to the Credit Card table.

```
Insert Into CREDIT_CARD values (1234567891234567, 'Josep Smith ', 'Visa','01-may-21');
Insert Into CREDIT_CARD values (1122334455667788, 'Noah Emma ', 'Chase Freedom Unlimited','01-JUN-22');
Insert Into CREDIT_CARD values (9988774455661122, 'Liam Olivia', 'Capital One Visa','07-APRIL-22');
Insert Into CREDIT_CARD values (3322116655449988, 'Jacob Charlotte', 'Chase Freedom Visa','08-JUN-23');
Insert Into CREDIT_CARD values (7788445566991122, 'Ethan Harper', 'Chase Diamond','09-MAY-21');
```

This is the Credit Card table with 5 new record that I executed.

	CREDITCARDNUMBER	OWNERNAME	MERCHANAME	EXPDATE
1	1234567891234567	Josep Smith	Visa	01-MAY-21
2	1122334455667788	Noah Emma	Chase Freedom Unlimited	01-JUN-22
3	9988774455661122	Liam Olivia	Capital One Visa	07-APR-22
4	3322116655449988	Jacob Cha...	Chase Freedom Visa	08-JUN-23
5	7788445566991122	Ethan Harper	Chase Diamond	09-MAY-21

INSERT STATEMENTS for Physician Table

I Created 5 INSERT QUERY as follows to the Physician table.

```
Insert Into PHYSICIAN values (1111, '1212', 'DN12');
Insert Into PHYSICIAN values (2222, '1313', 'DN13');
Insert Into PHYSICIAN values (3333, '1414', 'DN14');
Insert Into PHYSICIAN values (4444, '1515', 'DN15');
Insert Into PHYSICIAN values (5555, '1616', 'DN16');
```

This is the Physician table with 5 new record after executed.

	PHPERSONID	PHYSICIANPAGENUMBER	PHYSICIANDEANUMBER
1	1111	1212	DN12
2	2222	1313	DN13
3	3333	1414	DN14
4	4444	1515	DN15
5	5555	1616	DN16

INSERT STATEMENTS for Patient Table

I Created 5 INSERT QUERY as follows to the Patient table.

```
Insert Into PATIENT values (6666, '01-JAN-20', 'John Smith',3475151515,1111,'0');
Insert Into PATIENT values (7777, '05-FEB-20', 'Chloe Emilly',3471515151,2222,'R');
Insert Into PATIENT values (8888, '17-APR-20', 'Poppy Jack',6464252829,3333,'0');
Insert Into PATIENT values (9999, '11-AUG-20', 'Man Amor',7178963214,4444,'R');
Insert Into PATIENT values (1000, '21-JUN-19', 'Camel Jo',9296547893,5555,'0');
```

This is the Patient table with 5 new record after executed.

	PAPERSONID	FIRSTCONTACTDATE	EMARGENCYCONTACTNAME	EMARGENCYCONTACTNUMBER	PHPERSONID	PATIENTTYPE
1	6666	11-MAR-19	David Johnson	9296765444	1111	0
2	7777	05-FEB-20	Chloe Emilly	3471515151	2222	R
3	8888	17-APR-20	Poppy Jack	6464252829	3333	0
4	9999	11-AUG-20	Man Amor	7178963214	4444	R
5	1000	21-JUN-19	Camel Jo	9296547893	5555	0

INSERT STATEMENTS for Patient Credit Card Table

I Created 5 INSERT QUERY as follows to the Patient Credit Card table.

```
Insert Into PATIENT_CREDIT_CARD values (6666,1234567891234567);
Insert Into PATIENT_CREDIT_CARD values (7777,1122334455667788);
Insert Into PATIENT_CREDIT_CARD values (8888,9988774455661122);
Insert Into PATIENT_CREDIT_CARD values (9999,3322116655449988);
Insert Into PATIENT_CREDIT_CARD values (1000,7788445566991122);
```

This is the Patient Credit Card table with 5 new record after executed.

	PAPERSONID	CREDITCARDNUMBER
1	6666	1234567891234567
2	7777	1122334455667788
3	9999	3322116655449988
4	1000	7788445566991122

INSERT STATEMENTS for Insurance Company Table

I Created 5 INSERT QUERY as follows to the Insurance Company table.

```
Insert Into INSURANCE_COMPANY values ('AB111', 'United Health', 'PLC123',3474125632);
Insert Into INSURANCE_COMPANY values ('AB222', 'Cigna Health', 'PLC124',7188521479);
Insert Into INSURANCE_COMPANY values ('AB333', 'Blue Shield', 'PLC125',9295321478);
Insert Into INSURANCE_COMPANY values ('AB444', 'Aetna', 'PLC126',3472365984);
Insert Into INSURANCE_COMPANY values ('AB555', 'Metro Plus', 'PLC127',9295624568);
```

This is the Insurance Company table with 5 new record after executed.

	INSURANCECOMPANYID	INSUTANCECOMPANYNAME	POLICYNUMBER	INSURANCEPHONENUMBER
1	AB222	Cigna Health	PLC124	7188521479
2	AB333	Blue Shield	PLC125	9295321478
3	AB444	Aetna	PLC126	3472365984
4	AB555	Metro Plus	PLC127	9295624568
5	AB111	United Health	PLC123	3474125632

INSERT STATEMENTS for Patient Insurance Coverage Table

I Created 5 INSERT QUERY as follows to the Patient Insurance Coverage table.

```
Insert Into PATIENT_INSURANCE_COVERAGE values (6666, 'AB111');
Insert Into PATIENT_INSURANCE_COVERAGE values (7777, 'AB222');
Insert Into PATIENT_INSURANCE_COVERAGE values (8888, 'AB333');
Insert Into PATIENT_INSURANCE_COVERAGE values (9999, 'AB444');
Insert Into PATIENT_INSURANCE_COVERAGE values (1000, 'AB555');
```

This is the Patient Insurance Coverage table with 5 new record after executed.

	PAPERSONID	INSURANCECOMPANYID
1	7777	AB222
2	8888	AB333
3	9999	AB444
4	1000	AB555
5	6666	AB111

SELECT STATEMENT #1

I already created the PERSON table, now creating the SELECT statement that will return only one record that include all columns based on primary key.

```
SELECT *
FROM PERSON
WHERE PersonID = 1111;
```

I execute the query in SQL Developer and here are the results.

	PERSONID	FIRST	LAST	MIDDLE	STREETADDRESS	CITY	STATE	ZIPCODE	COUNTRY	BIRTHDATES	GENDER	PHONE	EMAIL	ISPATIENT	ISEMPLOYEE	ISVOLUNTEER	ISPHYSICIAN
1	1111	Joe	Smith	S	111 Jay Street	Brooklyn	NY	11201	BK	01-JAN-71	M	3473456653	JSmith@xyz.com	N	N	N	Y

SELECT STATEMENT #2

I have created the SELECT statement that will return multiple records that includes all columns based on some criteria

```
SELECT FIRSTCONTACTDATE,EMARGENCYCONTACTNAME, EMARGENCYCONTACTNUMBER
FROM patient
GROUP BY FIRSTCONTACTDATE,EMARGENCYCONTACTNAME, EMARGENCYCONTACTNUMBER
```

I execute the query in SQL Developer and here are the results.

	FIRSTCONTACTDATE	EMARGENCYCONTACTNAME	EMARGENCYCONTACTNUMBER
1	11-MAR-19	David Johnson	9296765444
2	05-FEB-20	Chloe Emilly	3471515151
3	21-JUN-19	Camel Jo	9296547893
4	17-APR-20	Poppy Jack	6464252829
5	11-AUG-20	Man Amor	7178963214

SELECT STATEMENT #3

I have created the SELECT Statement that will return multiple records from multiple tables including associate table.

```
SELECT PATIENT.PAPersonID, PATIENT_CREDIT_CARD.PAPersonID,patient_credit_card.creditcardnumber, CREDIT_CARD.CreditCardNumber
FROM PATIENT, PATIENT_CREDIT_CARD, CREDIT_CARD
WHERE PATIENT.PAPersonID = PATIENT_CREDIT_CARD.PAPersonID
AND PATIENT_CREDIT_CARD.CreditCardNumber = CREDIT_CARD.CreditCardNumber
AND PATIENT.PAPersonID = 6666;
```

I execute the query in SQL Developer and here are the results.

	PAPERSONID	PAPERSONID_1	CREDITCARDNUMBER	CREDITCARDNUMBER_1
1	6666	6666	1234567891234567	1234567891234567

UPDATE STATEMENT #1

I create the SELECT all statement to see all the records from PATIENT table.

```
Select *
From patient;
```

Here I have 5 records and next UPDATE query I will update some of the columns from first record where PAPersonID = 6666.

	PAPERSONID	FIRSTCONTACTDATE	EMARGENCYCONTACTNAME	EMARGENCYCONTACTNUMBER	PHPERSONID	PATIENTTYPE
1	6666	01-JAN-20	John Smith	3475151515	1111	O
2	7777	05-FEB-20	Chloe Emilly	3471515151	2222	R
3	8888	17-APR-20	Poppy Jack	6464252829	3333	O
4	9999	11-AUG-20	Man Amor	7178963214	4444	R
5	1000	21-JUN-19	Camel Jo	9296547893	5555	O

I have created the UPDATE statement that will update all the record except the Primary Key and Foreign Key of one record only based on the Primary Key.

```
Update Patient
SET FIRSTCONTACTDATE = '11-Mar-19',
    EMARGENCYCONTACTNAME = 'David Johnson',
    EMARGENCYCONTACTNUMBER = 9296765444,
    PATIENTTYPE = 'R'
WHERE papersonid = 6666;
```

To see the different, I have created the SELECT all statement to see the update records in PATIENT table.

```
Select *
From patient;
```

Notice, first record has updated.

	PAPERSONID	FIRSTCONTACTDATE	EMARGENCYCONTACTNAME	EMARGENCYCONTACTNUMBER	PHPERSONID	PATIENTTYPE
1	6666	11-MAR-19	David Johnson	9296765444	1111	R
2	7777	05-FEB-20	Chloe Emilly	3471515151	2222	R
3	8888	17-APR-20	Poppy Jack	6464252829	3333	O
4	9999	11-AUG-20	Man Amor	7178963214	4444	R
5	1000	21-JUN-19	Camel Jo	9296547893	5555	O

Update Statement 2

My table group with an associative entity table that contains the Composite Key only. None of my associate entity table has additional columns beside Primary Keys. Therefore, I cannot modify any of my associate entity table.

Delete Statement 1

In order to delete a record from any on my table groups tables, first I have to delete the record from the child table or the table has foreign key of the table that I want to delete. It this case I want to delete a record from Insurance table, therefore first I have to delete the foreign key record from the Patient_Insurance _Coverage table then I can delete that record from the Insurance table.

I create the SELECT all statement to see all the records from Patient_Insurance_Coverage table.

```
SELECT *
FROM patient_insurance_coverage;
```

This is the Patient_Insurance_Coverage Table before executing the DELETE statement.

	PAPERSONID	INSURANCECOMPANYID
1	6666	AB111
2	7777	AB222
3	8888	AB333
4	9999	AB444
5	1000	AB555

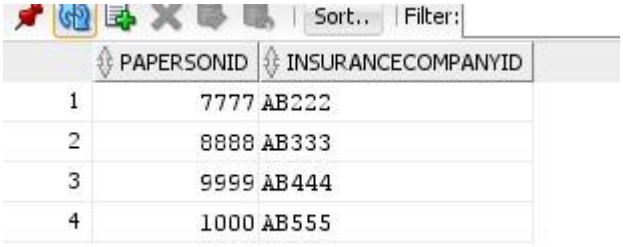
I have created the DELETE query to delete the PAPersonID = 6666 and InsuranceCompanyID = 'AB111'

```
DELETE FROM patient_insurance_coverage
WHERE papersonid = 6666
and InsuranceCompanyID = 'AB111';
```

Now, I create the SELECT all statement to see all the records from Patient_Insurance_Coverage table.

```
SELECT *
FROM patient_insurance_coverage;
```

As you can see, the records whose PAPersonID = 6666 and InsuranceCompanyID = 'AB111' was deleted.



Sort.. Filter:

	PAPERSONID	INSURANCECOMPANYID
1	7777	AB222
2	8888	AB333
3	9999	AB444
4	1000	AB555

InsuranceCompanyID = ‘AB111’ was deleted from the Patient_Insurance_Coverage table. Now I can delete this record from the Insurance_Company table.

I create the SELECT all statement to see all the records from Insurance_Company table.

```
SELECT *
FROM Insurance_Company;
```

This is the Insurance_Company Table before executing the DELETE statement.

	INSURANCECOMPANYID	INSUTANCECOMPANYNAME	POLICYNUMBER	INSURANCEPHONENUMBER
1	AB111	United Health	PLC123	3474125632
2	AB222	Cigna Health	PLC124	7188521479
3	AB333	Blue Shield	PLC125	9295321478
4	AB444	Aetna	PLC126	3472365984
5	AB555	Metro Plus	PLC127	9295624568

We create the DELETE query to delete the InsuranceCompanyID = 'AB111'

```
DELETE FROM Insurance_Company
 WHERE InsuranceCompanyID = 'AB111';
```

As you can see, the record whose InsuranceCompanyID = 'AB111' was deleted.

	INSURANCECOMPANYID	INSUTANCECOMPANYNAME	POLICYNUMBER	INSURANCEPHONENUMBER
1	AB222	Cigna Health	PLC124	7188521479
2	AB333	Blue Shield	PLC125	9295321478
3	AB444	Aetna	PLC126	3472365984
4	AB555	Metro Plus	PLC127	9295624568

Delete Statement 2

This is the Patient Credit Card ASSOCIATE ENTITY TABLE before executing the DELETE statement.

I create the SELECT all statement to see all the records from Patient_Credit_Card table.

```
SELECT*
FROM Patient_Credit_Card;
```

	PAPERSONID	CREDITCARDNUMBER
1	6666	1234567891234567
2	7777	1122334455667788
3	8888	9988774455661122
4	9999	3322116655449988
5	1000	7788445566991122

We create the DELETE query to delete the PApersonID = 8888 AND CreditCardNumber = 9988774455661122;

```
DELETE FROM patient_credit_card
WHERE papersonid = 8888
AND CreditCardNumber = 9988774455661122;
```

I create the SELECT all statement to see all the records from Patient_Credit_Card table.

```
SELECT*
FROM Patient_Credit_Card;
```

As you can see, the record whose PApersonID = 8888 AND CreditCardNumber = 9988774455661122 was deleted.

	PAPERSONID	CREDITCARDNUMBER
1	6666	1234567891234567
2	7777	1122334455667788
3	9999	3322116655449988
4	1000	7788445566991122

Conclusion

EZMedicalGroup is a complete package for a hospital management system where all kind of employees can interact by assigning patients, volunteers, bed for resident patient, patient can pay the fee by credit card or if they have medical insurance, and this database will record all the persons and patient's information etc. In order to complete the whole process, I had to do planning, analyze, design, development & implementation and finally operation & maintenance all steps to successfully complete the project. I had to create EER conceptual model diagram, normalized logical model, data dictionary, schema diagram to see the blueprint of entire database. I have created 22 require table in DB Developer and implement them in a physical schema diagram to see the relationship among the tables. Finally, after creating the table I had to test them to see the data flow. I was successfully able to run SELECT, INSERT, DELETE and UPDATE quires. This Project is fully understandable and designed by Business Requirement to get the job done.

Hospital Management System Database Design & Implementation

Version 2.0

Upgrade Objectives

This section is the upgrade version 2.0 of the Hospital Management System Database. Project management methodology using the database application development lifecycle. We were following all the phase do this upgrade, start with planning, analysis, design, development and maintenance phase to upgrade this version by creating business reports and improve the performance of full working application. We also created business reports Store Procedure for the database and it will be easier for the application developer to get the data from the database based on their hospital management demands. All the store procedure has been created based on the business scenario. Those reports are important for hospital employees, they might need every week or month in order to run the business.

Business Report Queries

1) **Report objectives:** Report objective is to serve the food for resident patient.

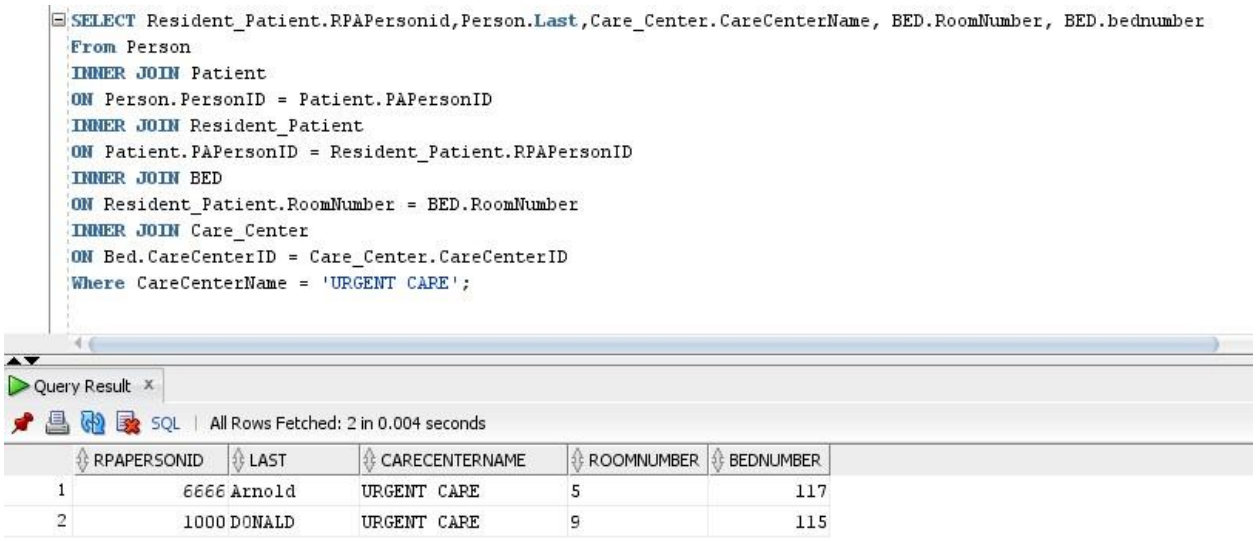
Persona this report is targeting: Supervisor

Decision this report will support: Based on this report room service employee can make decision where and who to serve the food.

Data need to create the report: We need Resident patient ID, Patient last name, Room number, Bed number

Additional details: Based on the information in this report supervisor will know patient's ID and Last name which will help to identify the patient and Care Center name, room number and bed number to identify the location of that patient.

```
SELECT Resident_Patient.RPAPersonid,Person.Last,Care_Center.CareCenterName,
BED.RoomNumber, BED.bednumber
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Resident_Patient
ON Patient.PAPersonID = Resident_Patient.RPAPersonID
INNER JOIN BED
ON Resident_Patient.RoomNumber = BED.RoomNumber
INNER JOIN Care_Center
ON Bed.CareCenterID = Care_Center.CareCenterID
Where CareCenterName = 'URGENT CARE';
```



```
SELECT Resident_Patient.RPAPersonid,Person.Last,Care_Center.CareCenterName, BED.RoomNumber, BED.bednumber
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Resident_Patient
ON Patient.PAPersonID = Resident_Patient.RPAPersonID
INNER JOIN BED
ON Resident_Patient.RoomNumber = BED.RoomNumber
INNER JOIN Care_Center
ON Bed.CareCenterID = Care_Center.CareCenterID
Where CareCenterName = 'URGENT CARE';
```

Query Result

SQL | All Rows Fetched: 2 in 0.004 seconds

	RPAPERSONID	LAST	CARECENTERNAME	ROOMNUMBER	BEDNUMBER
1	6666	Arnold	URGENT CARE	5	117
2	1000	DONALD	URGENT CARE	9	115

2) **Report objectives:** The objective is to see if Hospital management need to hire more physician

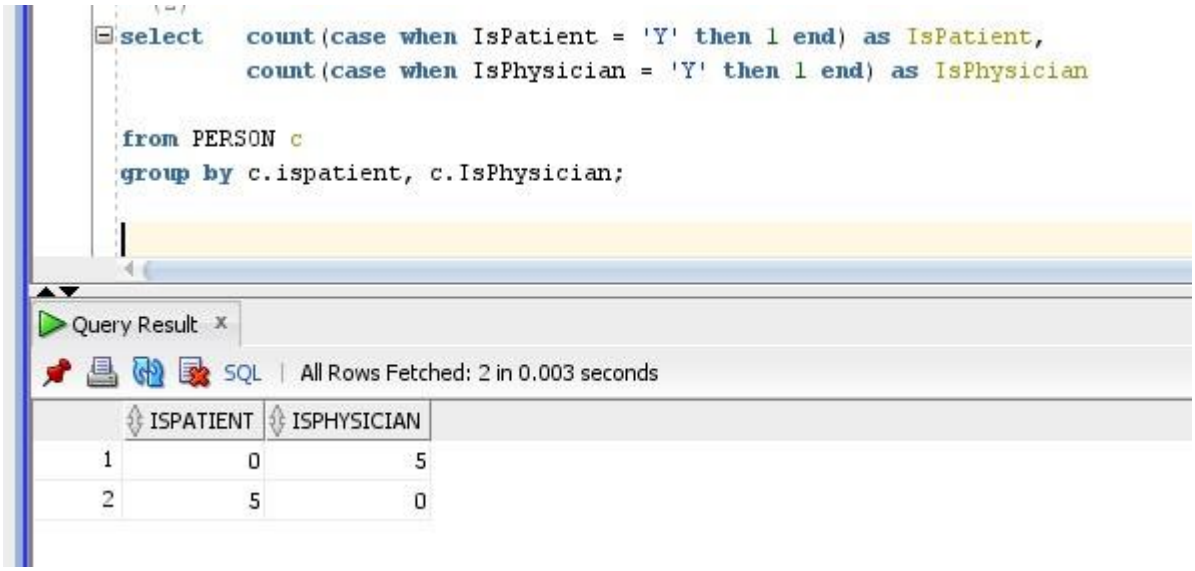
Persona this report is targeting: Hiring Manager

Decision this report will support: Hiring Manager will make decision if they need to hire more physician in this hospital.

Data need to create the report: We need total number of patient and physician

Additional details: This report will provide the total number of patient and physician are exist in this hospital and based on this report hiring manager will decide if they need to hire more physician in this hospital.

```
select   count(case when IsPatient = 'Y' then 1 end) as IsPatient,
count(case when IsPhysician = 'Y' then 1 end) as IsPhysician
        from
PERSON c
group by c.ispatient, c.IsPhysician;
```



3) **Report objectives:** To provide 10% salary increase to employees

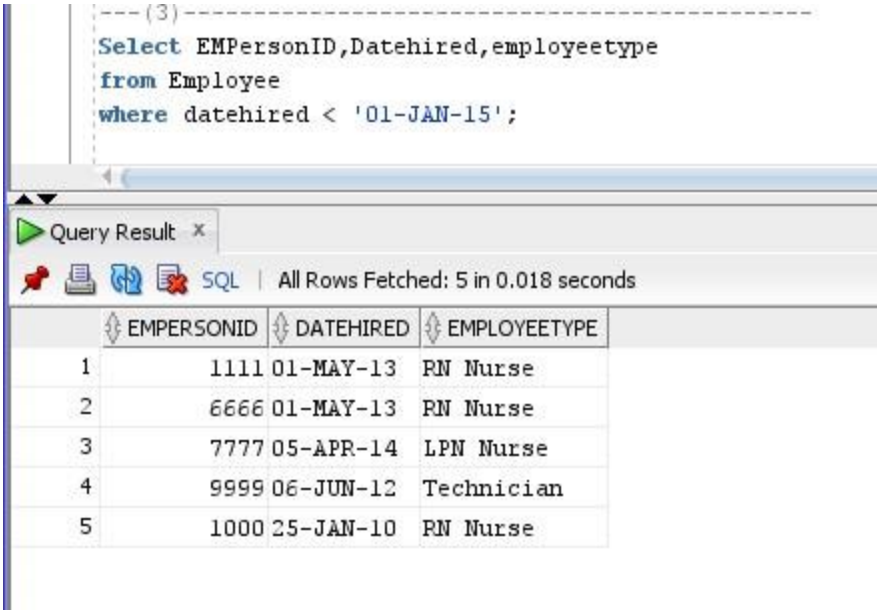
Persona this report is targeting: Human Resource

Decision this report will support: Based on the report HR can make decision to provide 10% salary increase to the employee who got hired before January 1st 2015.

Data need to create the report: We need Employee ID, Date hired and Employee type

Additional details: Query report will display Employee ID, Date hire and Employee type. By looking at the date hired column HR can make decision who to give 10% salary increase

```
Select EMPersonID,Datehired,employeetype from
Employee
where datehired < '01-JAN-15';
```



	EMPERSONID	DATEHIRED	EMPLOYEETYPE
1	1111	01-MAY-13	RN Nurse
2	6666	01-MAY-13	RN Nurse
3	7777	05-APR-14	LPN Nurse
4	9999	06-JUN-12	Technician
5	1000	25-JAN-10	RN Nurse

4) Report objectives: Patient who does not have medical insurance, send them full coverage insurance offer.

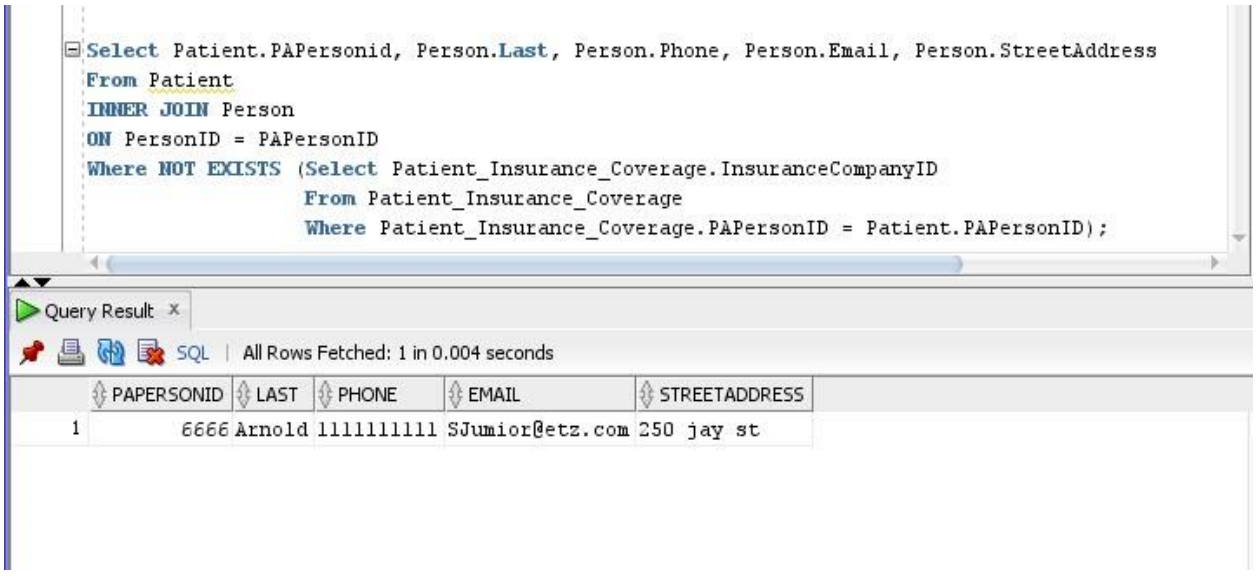
Persona this report is targeting: Marketing

Decision this report will support: Based on this report marketing employee can make decision who to contact and send them medical insurance offer.

Data need to create the report: The Patient ID, Name and Contact Number, Email and Street address.

Additional details: Query report will display the Patient ID, Contact Name and Contact Number of all the patients who does not have an insurance coverage. Business objectives for the Marketing people is to reach out those patients and send them free full coverage insurance offer.

```
Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email, Person.StreetAddress From
Patient
INNER JOIN Person
ON PersonID = PAPersonID
Where NOT EXISTS (Select Patient_Insurance_Coverage.InsuranceCompanyID
                  From Patient_Insurance_Coverage
                  Where Patient_Insurance_Coverage.PAPersonID = Patient.PAPersonID);
```



5) **Report objectives:** To hire more RN Nurse

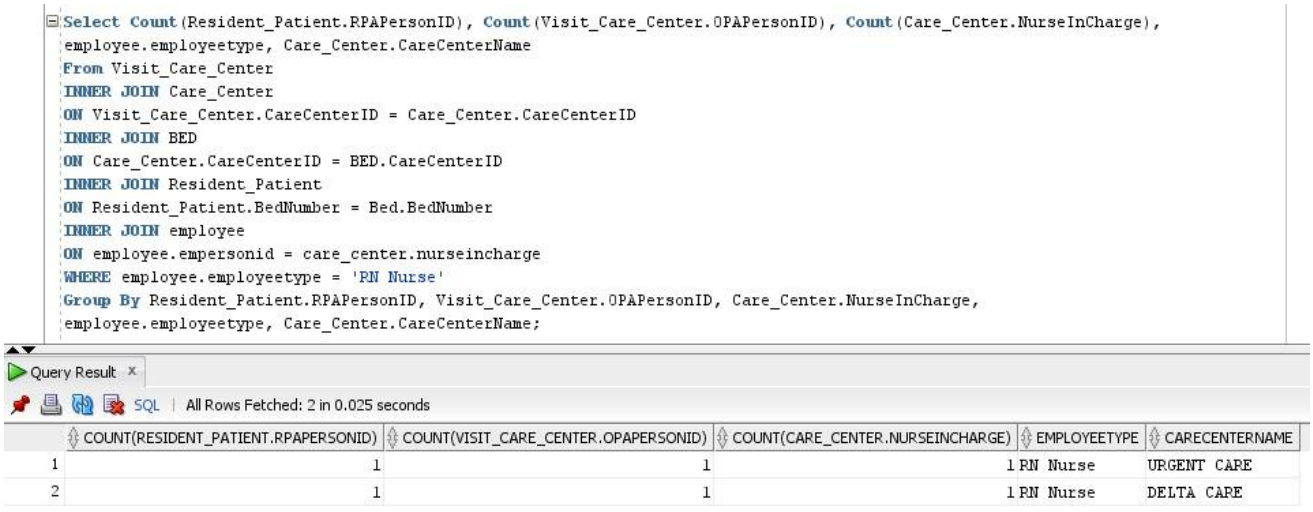
Persona this report is targeting: Hiring Manager

Decision this report will support: This report will help the hiring manager to make decision if they need to hire more RN Nurse for the care center.

Data need to create the report: Total number of Resident Patient ID and Out patient ID also nurse in charge ID, Employee type and Care center name.

Additional details: We know resident patient and outpatient both can admit to the care center so we have both patient's information and also report will show the RN Nurse who is working in the care center.

```
Select Count(Resident_Patient.RPAPersonID), Count(Visit_Care_Center.OPAPersonID),
Count(Care_Center.NurseInCharge),
employee.employeetype, Care_Center.CareCenterName
From Visit_Care_Center
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
INNER JOIN employee
ON employee.empersonid = care_center.nurseincharge
WHERE employee.employeetype = 'RN Nurse'
Group By Resident_Patient.RPAPersonID, Visit_Care_Center.OPAPersonID, Care_Center.NurseInCharge,
employee.employeetype, Care_Center.CareCenterName;
```



```
Select Count(Resident_Patient.RPAPersonID), Count(Visit_Care_Center.OPAPersonID), Count(Care_Center.NurseInCharge),
employee.employeetype, Care_Center.CareCenterName
From Visit_Care_Center
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
INNER JOIN employee
ON employee.empersonid = care_center.nurseincharge
WHERE employee.employeetype = 'RN Nurse'
Group By Resident_Patient.RPAPersonID, Visit_Care_Center.OPAPersonID, Care_Center.NurseInCharge,
employee.employeetype, Care_Center.CareCenterName;
```

Query Result

SQL | All Rows Fetched: 2 in 0.025 seconds

	COUNT(RESIDENT_PATIENT.RPAPERSONID)	COUNT(VISIT_CARE_CENTER.OPAPERSONID)	COUNT(CARE_CENTER.NURSEINCHARGE)	EMPLOYEETYPE	CARECENTERNAME
1	1	1	1	RN Nurse	URGENT CARE
2	1	1	1	RN Nurse	DELTA CARE

6) **Report objectives:** To give the employee monthly Metro Card

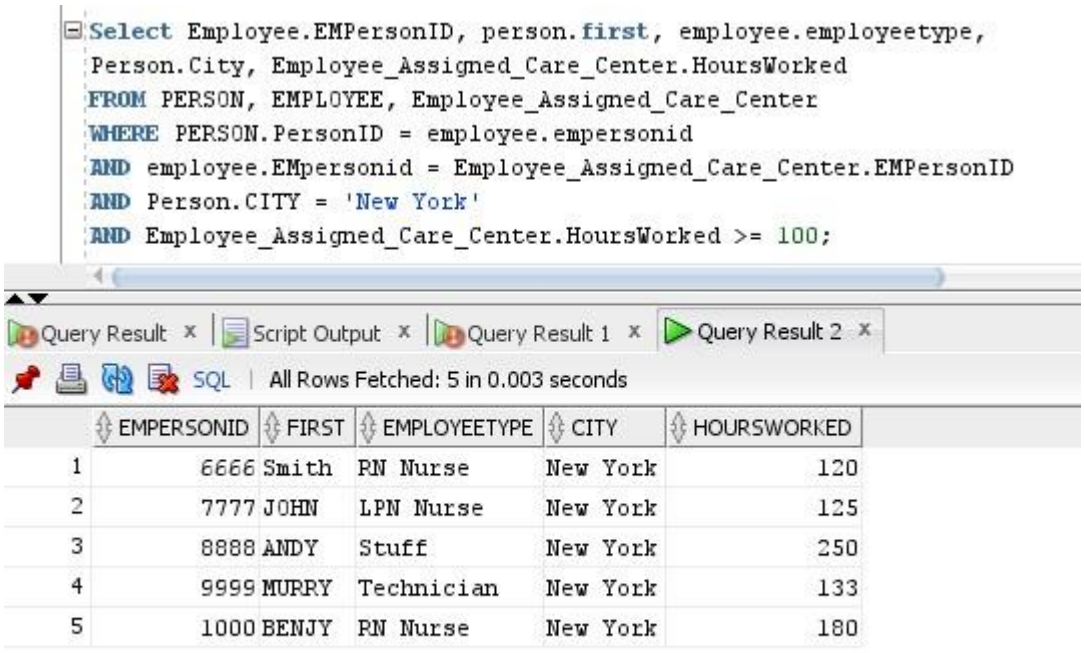
Persona this report is targeting: Employee Supervisor

Decision this report will support: This report will help the supervisor to make the decision to give the monthly Metro Card to the employee who live in New York and worked more the 100 hours.

Data need to create the report: Employee ID, First Name, Employee Type, City and Working Hours

Additional details: Query report that will displays the Employee ID, First Name, Employee Type, City and Working Hours of all the employees who lives in 'New York' and worked more then 100 hours. Business objective for the Supervisor is to give them free monthly Metro Card for public transportation.

```
Select Employee.EMPersonID, person.first, employee.employeetype,
Person.City, Employee_Assigned_Care_Center.HoursWorked
FROM PERSON, EMPLOYEE, Employee_Assigned_Care_Center
WHERE PERSON.PersonID = employee.empersonid
AND employee.EMpersonid = Employee_Assigned_Care_Center.EMPersonID
AND Person.CITY = 'New York'
AND Employee_Assigned_Care_Center.HoursWorked >= 100;
```



```
Select Employee.EMPersonID, person.first, employee.employeetype,
Person.City, Employee_Assigned_Care_Center.HoursWorked
FROM PERSON, EMPLOYEE, Employee_Assigned_Care_Center
WHERE PERSON.PersonID = employee.empersonid
AND employee.EMpersonid = Employee_Assigned_Care_Center.EMPersonID
AND Person.CITY = 'New York'
AND Employee_Assigned_Care_Center.HoursWorked >= 100;
```

Query Result | Script Output | Query Result 1 | Query Result 2

SQL | All Rows Fetched: 5 in 0.003 seconds

	EMPERSONID	FIRST	EMPLOYEETYPE	CITY	HOURSWORKED
1	6666	Smith	RN Nurse	New York	120
2	7777	JOHN	LPN Nurse	New York	125
3	8888	ANDY	Stuff	New York	250
4	9999	MURRY	Technician	New York	133
5	1000	BENJY	RN Nurse	New York	180

7) Report objectives: Make schedule for the technician to inspect all the Electric Automated Bed

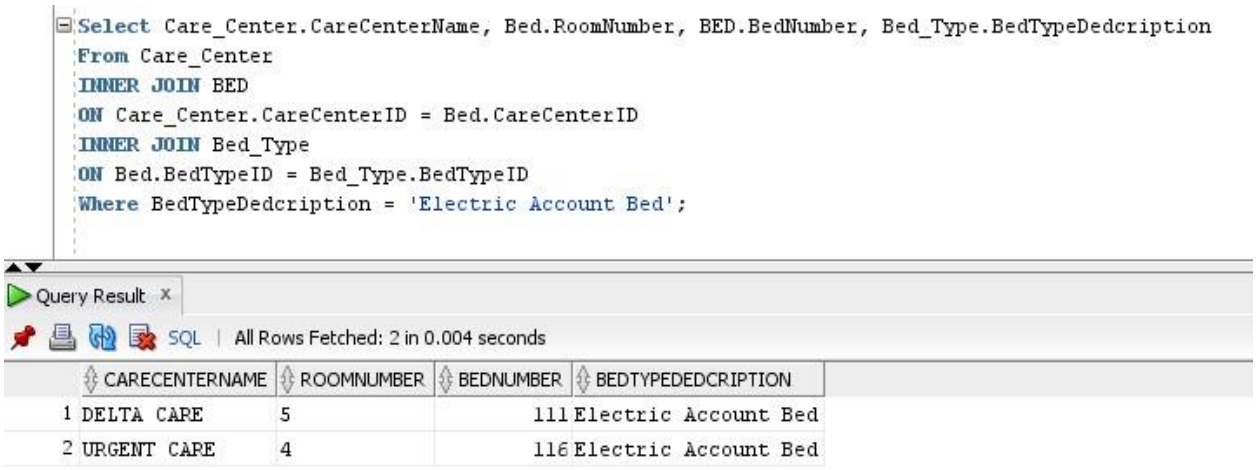
Persona this report is targeting: Schedule Manager

Decision this report will support: This report will help the Schedule Manager to see all the information about the location of electronic bed and send electrician for monthly inspection.

Data need to create the report: Care Center Name, Room Number, Bed Number, Bed type description.

Additional details: Report will display Care center name, room number and bed number will help to navigate where is electric bed are located. This way technician easily can go there and do their work.

```
Select Care_Center.CareCenterName, Bed.RoomNumber, BED.BedNumber,
Bed_Type.BedTypeDedcription
From Care_Center
INNER JOIN BED
ON Care_Center.CareCenterID = Bed.CareCenterID
INNER JOIN Bed_Type
ON Bed.BedTypeID = Bed_Type.BedTypeID
Where BedTypeDedcription = 'Electric Account Bed';
```



	CARECENTERNAME	ROOMNUMBER	BEDNUMBER	BEDTYPEDEDCRIPTION
1	DELTA CARE	5	111	Electric Account Bed
2	URGENT CARE	4	116	Electric Account Bed

8) Report objectives: Follow up call and reschedule out patient

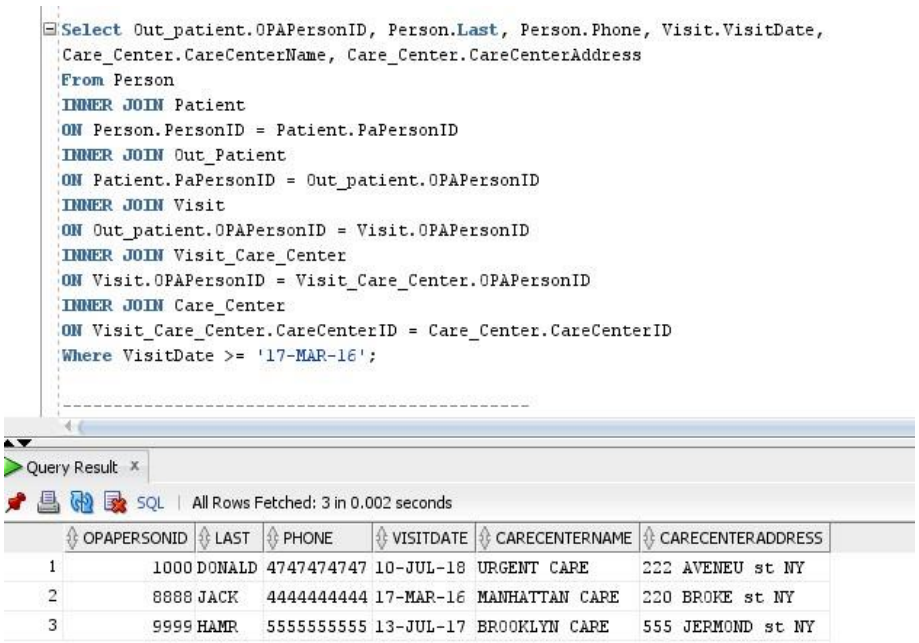
Persona this report is targeting: Front-desk assistant

Decision this report will support: Based on this report front-desk assistant can see who are the patient are visited any given date and able to call them for follow up or reschedule.

Data need to create the report: Our patient ID, Patient Last name, Phone number, Visit Date, Care center name, Care Center address

Additional details: Report will display patient ID, name and Phone number which is going to help the employee to contact with the patient and also report will display which care center they have visited so then Front-desk assistant can put them in the same care center.

```
Select Out_patient.OPAPersonID, Person.Last, Person.Phone, Visit.VisitDate,
Care_Center.CareCenterName, Care_Center.CareCenterAddress
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PaPersonID
INNER JOIN Out_Patient
ON Patient.PaPersonID = Out_patient.OPAPersonID
INNER JOIN Visit
ON Out_patient.OPAPersonID = Visit.OPAPersonID
INNER JOIN Visit_Care_Center
ON Visit.OPAPersonID = Visit_Care_Center.OPAPersonID
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
Where VisitDate >= '17-MAR-16';
```



	OPAPERSONID	LAST	PHONE	VISITDATE	CARECENTERNAME	CARECENTERADDRESS
1	1000	DONALD	4747474747	10-JUL-18	URGENT CARE	222 AVENEU st NY
2	8888	JACK	4444444444	17-MAR-16	MANHATTAN CARE	220 BROKE st NY
3	9999	HAMR	5555555555	13-JUL-17	BROOKLYN CARE	555 JERMOND st NY

9) Report objectives: Collect Credit Card Information

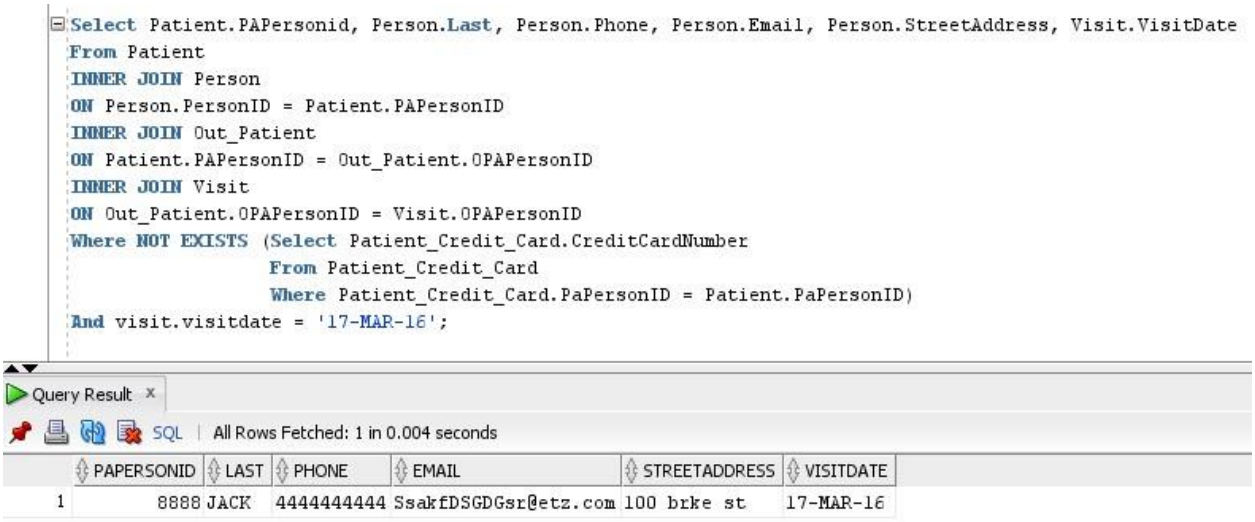
Persona this report is targeting: Accounts

Decision this report will support: This report will target to those patients who visited particular date and did not have a credit card so hospital Account personal can make decision who to contact.

Data need to create the report: Patient ID, Patient Last name, Phone number, Email, Street Address and Visit Date.

Additional details: Query report will display all the contact information so that Account personal can contact them by phone, email and mailing address are available.

```
Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email,
Person.StreetAddress, Visit.VisitDate
From Patient
INNER JOIN Person
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Out_Patient
ON Patient.PAPersonID = Out_Patient.OPAPersonID
INNER JOIN Visit
ON Out_Patient.OPAPersonID = Visit.OPAPersonID
Where NOT EXISTS (Select Patient_Credit_Card.CreditCardNumber
                  From Patient_Credit_Card
                  Where Patient_Credit_Card.PaPersonID = Patient.PaPersonID)
And visit.visitdate = '17-MAR-16';
```



10) Report objectives: Hire new volunteer if needed

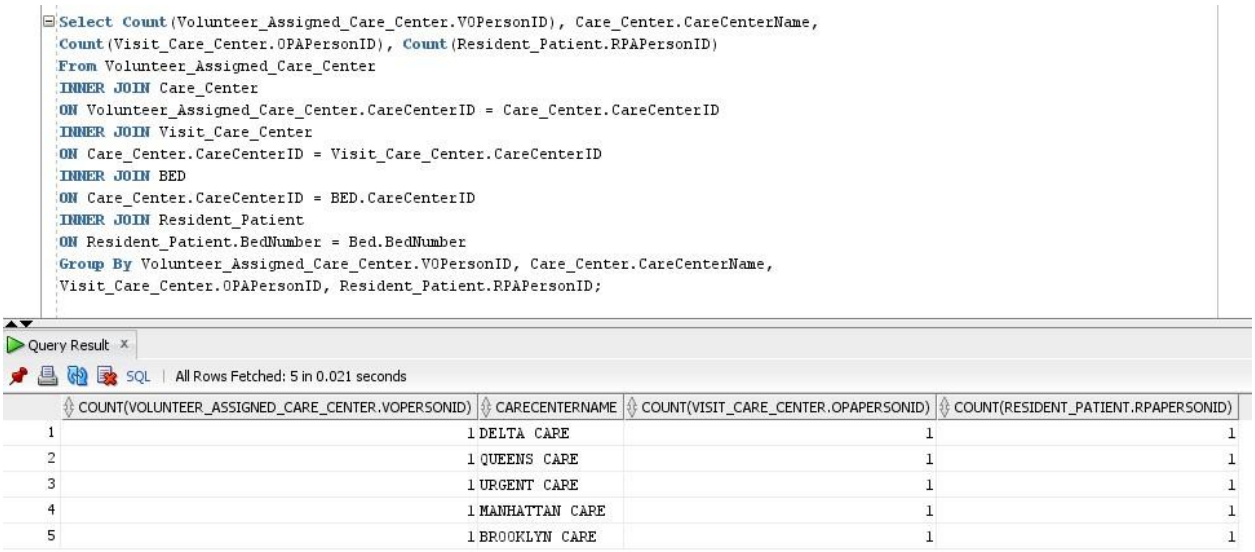
Persona this report is targeting: Volunteer Supervisor

Decision this report will support: Based on total number of outpatient and resident patient Volunteer Supervisor will make decision if hospital need more volunteer or not.

Data need to create the report: Volunteer ID, Care Center name, Out Patient and resident Patient

Additional details: Query report will display total number of Volunteer, Care center name and total number of resident and outpatient who is currently working in a care center. Business objective for the Volunteer Supervisor to make decision if they need to hire more volunteer for those care center. It's all depends on number of patients those care center has.

```
Select Count(Volunteer_Assigned_Care_Center.VOPersonID),
Care_Center.CareCenterName,
Count(Visit_Care_Center.OPAPersonID), Count(Resident_Patient.RPAPersonID)
From Volunteer_Assigned_Care_Center
INNER JOIN Care_Center
ON Volunteer_Assigned_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN Visit_Care_Center
ON Care_Center.CareCenterID = Visit_Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
Group By Volunteer_Assigned_Care_Center.VOPersonID, Care_Center.CareCenterName,
Visit_Care_Center.OPAPersonID, Resident_Patient.RPAPersonID;
```



```
Select Count(Volunteer_Assigned_Care_Center.VOPersonID), Care_Center.CareCenterName,
Count(Visit_Care_Center.OPAPersonID), Count(Resident_Patient.RPAPersonID)
From Volunteer_Assigned_Care_Center
INNER JOIN Care_Center
ON Volunteer_Assigned_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN Visit_Care_Center
ON Care_Center.CareCenterID = Visit_Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
Group By Volunteer_Assigned_Care_Center.VOPersonID, Care_Center.CareCenterName,
Visit_Care_Center.OPAPersonID, Resident_Patient.RPAPersonID;
```

Query Result

SQL | All Rows Fetched: 5 in 0.021 seconds

	COUNT(VOLUNTEER_ASSIGNED_CARE_CENTER.VOPERSONID)	CARECENTERNAME	COUNT(VISIT_CARE_CENTER.OPAPERSONID)	COUNT(RESIDENT_PATIENT.RPAPERSONID)
1	1	DELTA CARE	1	1
2	1	QUEENS CARE	1	1
3	1	URGENT CARE	1	1
4	1	MANHATTAN CARE	1	1
5	1	BROOKLYN CARE	1	1

Business Reports Store Procedures

1) **Store Procedure Objectives:** Store Procedure objective is to serve the food for resident patient.
   Persona this Store Procedure is targeting: Supervisor
   Decision this Store Procedure will support: Based on this report room service employee can make decision where and who to serve the food.
   Data need to create the Store Procedure: We need Resident patient ID, Patient last name, Room number, Bed number
   Additional details: Based on the information in this report supervisor will know patient's ID and Last name which will help to identify the patient and Care Center name, room number and bed number to identify the location of that patient.

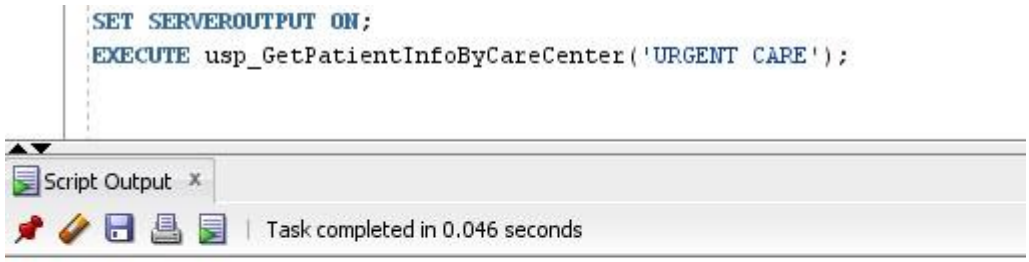
```
Create or Replace Procedure usp_GetPatientInfoByCareCenter(p_CareCenter IN VARCHAR2)

IS

    --declare variable needed for SELECT INTO statement

    v_RPAPersonid       NUMBER(4);
v_LastName          VARCHAR2(20);
v_CareCenterName    VARCHAR2(20);
v_RoomNumber        VARCHAR2(4);
v_BedNumber         NUMBER(20);

    --Declare Cursor
    CURSOR cur_Patient IS

    --Query cursor will point to results
        SELECT Resident_Patient.RPAPersonid,Person.Last,Care_Center.CareCenterName,
BED.RoomNumber, BED.bednumber
        From Person
        INNER JOIN Patient
        ON Person.PersonID = Patient.PAPersonID
        INNER JOIN Resident_Patient
        ON Patient.PAPersonID = Resident_Patient.RPAPersonID
        INNER JOIN BED
        ON Resident_Patient.RoomNumber = BED.RoomNumber
        INNER JOIN Care_Center
        ON Bed.CareCenterID = Care_Center.CareCenterID
        Where CareCenterName = p_CareCenter;
    Begin
        OPEN cur_Patient;

        LOOP
            FETCH cur_Patient INTO  v_RPAPersonid, v_LastName, v_CareCenterName, v_RoomNumber,
v_BedNumber;
            EXIT WHEN cur_Patient%NOTFOUND;
```

```
            DBMS_OUTPUT.PUT_LINE( 'Patient info: ' ||v_RPAPersonid|| ' ' ||v_LastName|| '
' ||v_CareCenterName||
            ' ' ||v_RoomNumber|| ' ' ||v_BedNumber);

            END LOOP;

            CLOSE cur_Patient;
    END usp_GetPatientInfoByCareCenter;
```

```
Create or Replace Procedure usp_GetPatientInfoByCareCenter(p_CareCenter IN VARCHAR2)
IS
    --declare variable needed for SELECT INTO statement

    v_RPAPersonid       NUMBER(4);
    v_LastName          VARCHAR2(20);
    v_CareCenterName    VARCHAR2(20);
    v_RoomNumber        VARCHAR2(4);
    v_BedNumber         NUMBER(20);

    --Declare Cursor
    CURSOR cur_Patient IS
    --Query cursor will point to results
        SELECT Resident_Patient.RPAPersonid,Person.Last,Care_Center.CareCenterName, BED.RoomNumber, BED.bednumber
        From Person
        INNER JOIN Patient
        ON Person.PersonID = Patient.PAPersonID
        INNER JOIN Resident_Patient
        ON Patient.PAPersonID = Resident_Patient.RPAPersonID
        INNER JOIN BED
        ON Resident_Patient.RoomNumber = BED.RoomNumber
        INNER JOIN Care_Center
        ON Bed.CareCenterID = Care_Center.CareCenterID
        Where CareCenterName = p_CareCenter;

    Begin
        OPEN cur_Patient;

        LOOP
            FETCH cur_Patient INTO  v_RPAPersonid, v_LastName, v_CareCenterName, v_RoomNumber, v_BedNumber;
            EXIT WHEN cur_Patient%NOTFOUND;

            DBMS_OUTPUT.PUT_LINE( 'Patient info: ' ||v_RPAPersonid|| ' ' ||v_LastName|| ' ' ||v_CareCenterName||
            ' ' ||v_RoomNumber|| ' ' ||v_BedNumber);

            END LOOP;
            CLOSE cur_Patient;
    END usp_GetPatientInfoByCareCenter;
```

```
Script Output ×
Task completed in 0.148 seconds
Procedure USP_GETPATIENTINFOBYCARECENTER compiled
```

```
SET SERVEROUTPUT ON;
EXECUTE usp_GetPatientInfoByCareCenter('URGENT CARE');
```




```
Patient info: 6666 Arnold URGENT CARE 5 117
Patient info: 1000 DONALD URGENT CARE 9 115


PL/SQL procedure successfully completed.
```

2) Store Procedure objectives: The objective is to see if Hospital management need to hire more physician

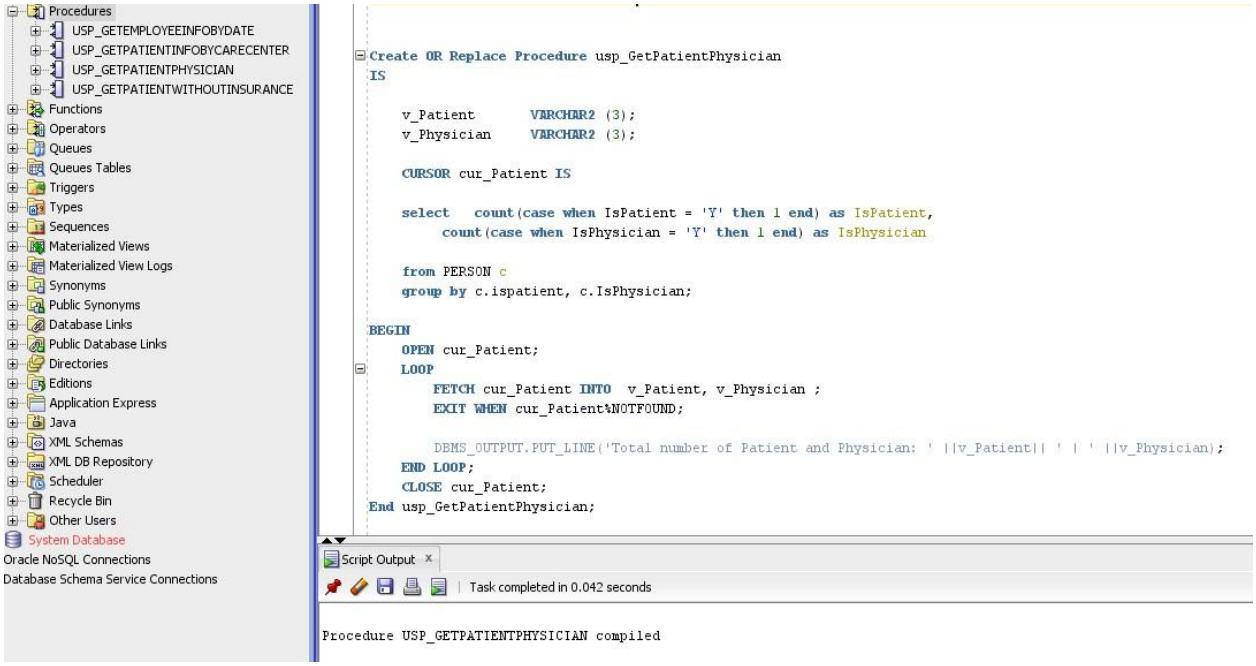
Persona this Store Procedure is targeting: Hiring Manager

Decision this Store Procedure will support: Hiring Manager will make decision if they need to hire more physician in this hospital.

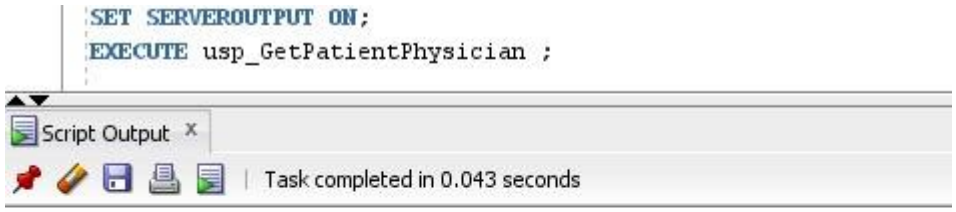
Data need to create the Store Procedure: We need total number of patient and physician

Additional details: This Store Procedure will provide the total number of patient and physician are exist in this hospital and based on this report hiring manager will decide if they need to hire more physician in this hospital.

```
Create OR Replace Procedure usp_GetPatientPhysician
IS
    v_Patient       VARCHAR2 (3);
v_Physician     VARCHAR2 (3);

    CURSOR cur_Patient IS

    select   count(case when IsPatient = 'Y' then 1 end) as IsPatient,
count(case when IsPhysician = 'Y' then 1 end) as IsPhysician
           from
PERSON c
    group by c.ispatient, c.IsPhysician;

BEGIN
    OPEN cur_Patient;
    LOOP
        FETCH cur_Patient INTO  v_Patient, v_Physician ;
        EXIT WHEN cur_Patient%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Total number of Patient and Physician: ' ||v_Patient||
' | ' ||v_Physician);

    END LOOP;
    CLOSE cur_Patient;
End usp_GetPatientPhysician;
```



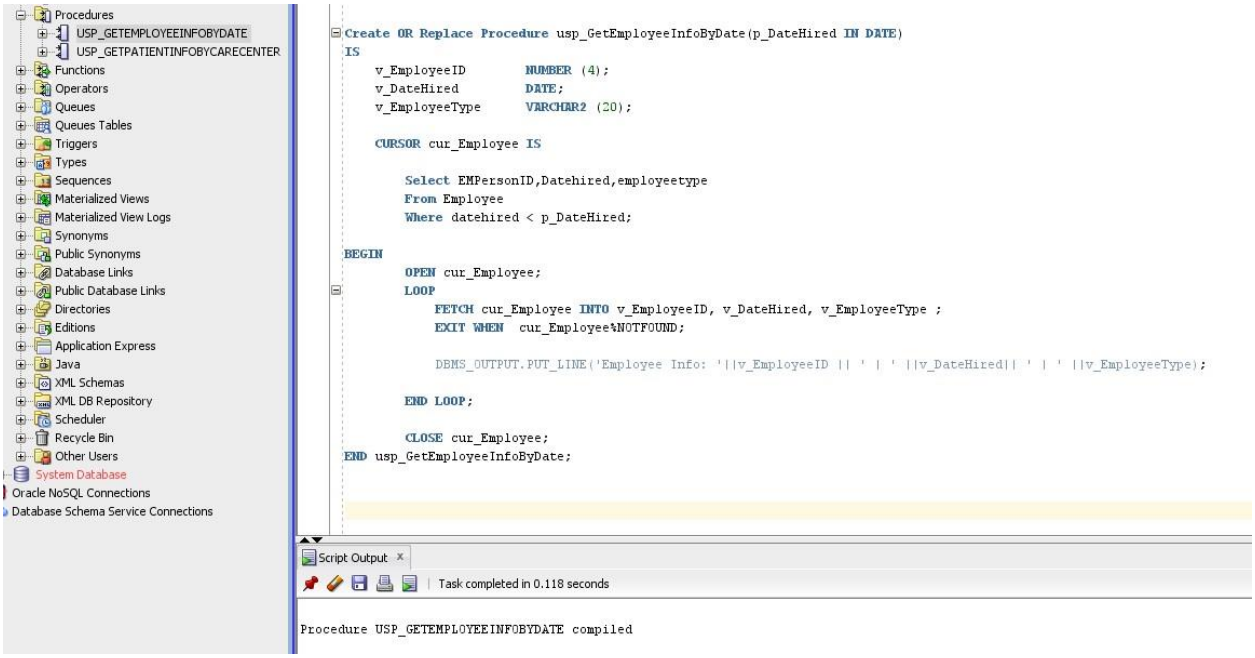

```
SET SERVEROUTPUT ON;
EXECUTE usp_GetPatientPhysician ;
```



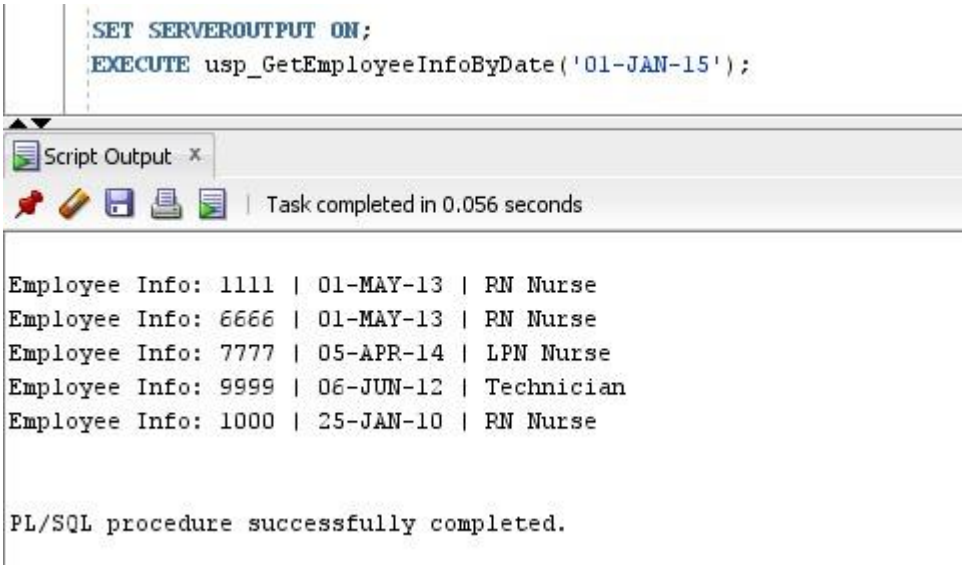

```
Total number of Patient and Physician: 0 | 5
Total number of Patient and Physician: 7 | 0


PL/SQL procedure successfully completed.
```

3) **Store Procedure objectives:** To provide 10% salary increase to employees
Persona this Store Procedure is targeting: Human Resource
Decision this Store Procedure will support: HR can make decision to provide 10% salary increase to the employee who got hired before January 1st 2015.
Data need to create the Store Procedure: We need Employee ID, Date hired and Employee type
Additional details: Store Procedure will display Employee ID, Date hire and Employee type. By looking at the date hired column HR can make decision who to give 10% salary increase

```
Create OR Replace Procedure usp_GetEmployeeInfoByDate(p_DateHired IN DATE) IS
    v_EmployeeID           NUMBER (4);
v_DateHired                     DATE;
v_EmployeeType      VARCHAR2 (20);

    CURSOR cur_Employee IS

        Select EMPersonID,Datehired,employeetype
        From Employee
        Where datehired < p_DateHired;

BEGIN
        OPEN cur_Employee;
        LOOP
            FETCH cur_Employee INTO v_EmployeeID, v_DateHired, v_EmployeeType ;
            EXIT WHEN  cur_Employee%NOTFOUND;

            DBMS_OUTPUT.PUT_LINE('Employee Info: '||v_EmployeeID || ' | ' ||v_DateHired||
' | ' ||v_EmployeeType);

        END LOOP;

        CLOSE cur_Employee;
END usp_GetEmployeeInfoByDate;
```




```
SET SERVEROUTPUT ON;
EXECUTE usp_GetEmployeeInfoByDate('01-JAN-15');
```



4) **Store Procedure objectives:** Patient who does not have medical insurance, send them full coverage insurance offer.

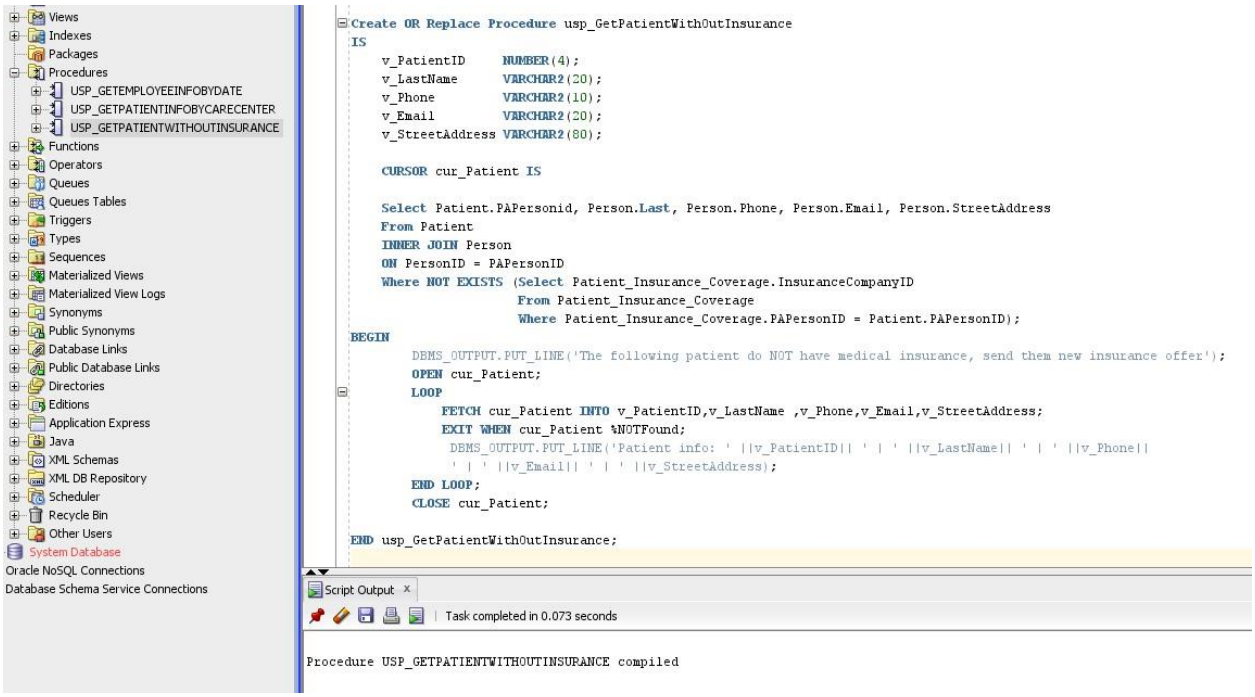
Persona this Store Procedure is targeting: Marketing

Decision this Store Procedure will support: Based on Store Procedure marketing employee can make decision who to contact and send them medical insurance offer.

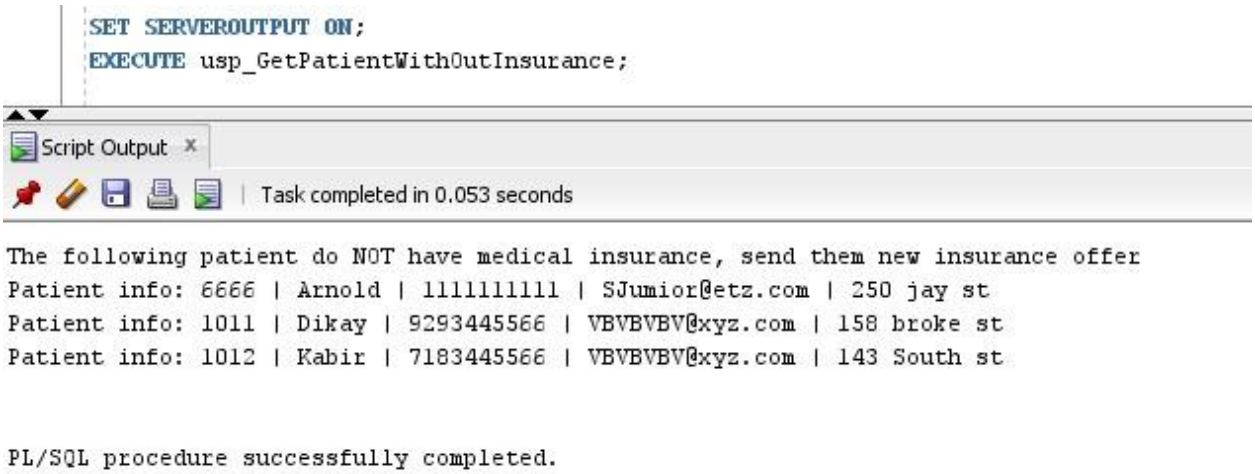
Data need to create the Store Procedure: The Patient ID, Name and Contact Number, Email and Street address.

Additional details: Store Procedure will display the Patient ID, Contact Name and Contact Number of all the patients who does not have an insurance coverage. Business objectives for the Marketing people is to reach out those patients and send them free full coverage insurance offer.

```
Create OR Replace Procedure usp_GetPatientWithOutInsurance IS
    v_PatientID     NUMBER(4);
v_LastName      VARCHAR2(20);     v_Phone
VARCHAR2(10);     v_Email
VARCHAR2(20);     v_StreetAddress
VARCHAR2(80);

    CURSOR cur_Patient IS

    Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email,
Person.StreetAddress
    From Patient
    INNER JOIN Person
    ON PersonID = PAPersonID
    Where NOT EXISTS (Select Patient_Insurance_Coverage.InsuranceCompanyID
                      From Patient_Insurance_Coverage
                      Where Patient_Insurance_Coverage.PAPersonID = Patient.PAPersonID); BEGIN
        DBMS_OUTPUT.PUT_LINE('The following patient do NOT have medical insurance, send them
new insurance offer');
        OPEN cur_Patient;
        LOOP
            FETCH cur_Patient INTO v_PatientID,v_LastName
,v_Phone,v_Email,v_StreetAddress;
            EXIT WHEN cur_Patient %NOTFound;
             DBMS_OUTPUT.PUT_LINE('Patient info: ' ||v_PatientID|| ' | ' ||v_LastName|| ' |
' ||v_Phone||
             ' | ' ||v_Email|| ' | ' ||v_StreetAddress);
        END LOOP;
        CLOSE cur_Patient;

END usp_GetPatientWithOutInsurance;
```

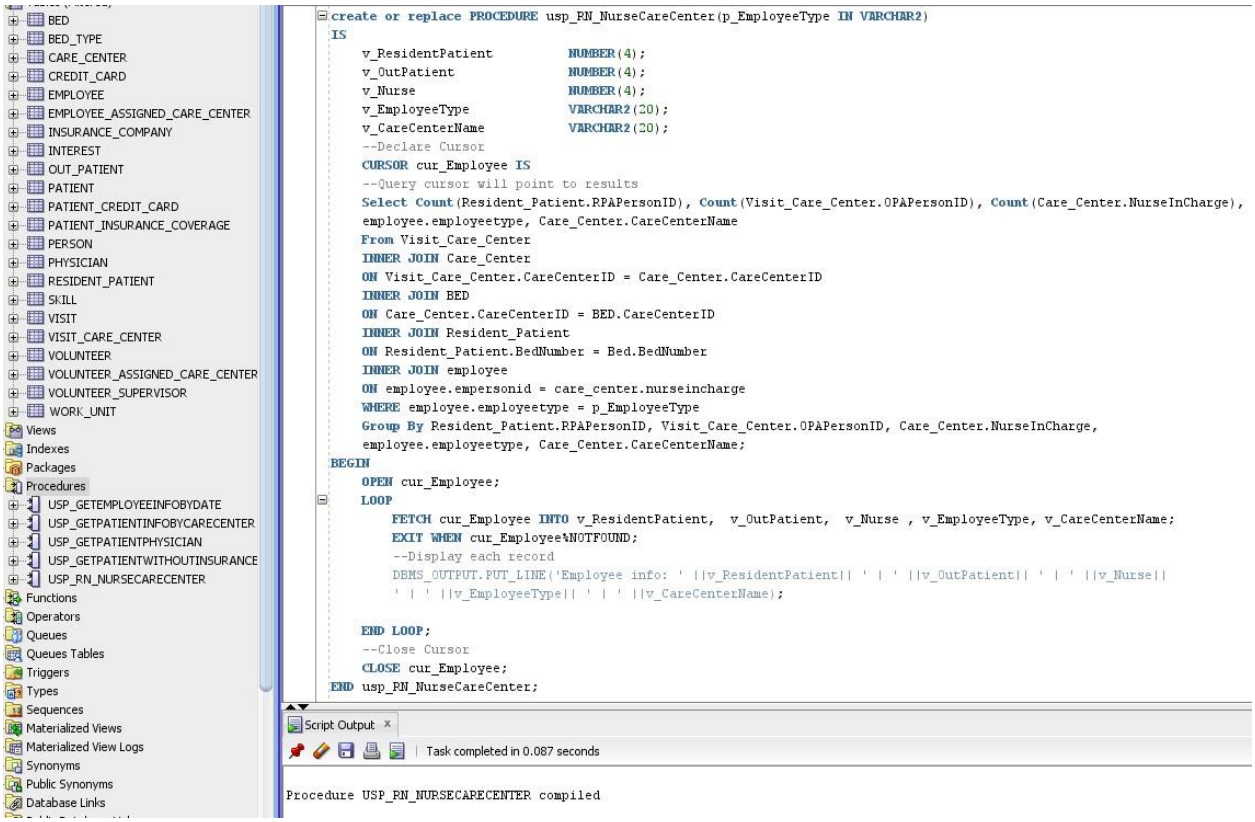



```
SET SERVEROUTPUT ON;
EXECUTE usp_GetPatientWithOutInsurance;
```

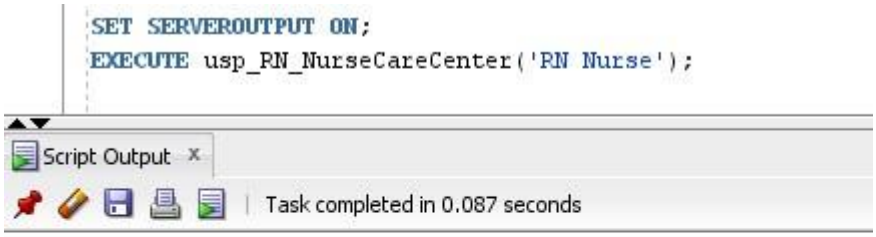


5) **Store Procedure objectives:** To hire more RN
Persona this Store Procedure is targeting: Hiring Manager
Decision this Store Procedure will support: This Store Procedure will help the hiring manager to make decision if they need to hire more RN Nurse for the care center.
Data need to create the Store Procedure: Total number of Resident Patient ID and Outpatient ID also nurse in charge ID, Employee type and Care center name.
Additional details: We know resident patient and outpatient both can admit to the care center so we have both patient's information and also Store Procedure will show the RN Nurse who is working in the care center.

```
create or replace PROCEDURE usp_RN_NurseCareCenter(p_EmployeeType IN VARCHAR2) IS
    v_ResidentPatient          NUMBER(4);
v_OutPatient               NUMBER(4);     v_Nurse
NUMBER(4);     v_EmployeeType
VARCHAR2(20);     v_CareCenterName
VARCHAR2(20);
    --Declare Cursor
    CURSOR cur_Employee IS
    --Query cursor will point to results
    Select Count(Resident_Patient.RPAPersonID), Count(Visit_Care_Center.OPAPersonID),
Count(Care_Center.NurseInCharge),
    employee.employeetype, Care_Center.CareCenterName
    From Visit_Care_Center
    INNER JOIN Care_Center
    ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
    INNER JOIN BED
    ON Care_Center.CareCenterID = BED.CareCenterID
    INNER JOIN Resident_Patient
    ON Resident_Patient.BedNumber = Bed.BedNumber
    INNER JOIN employee
    ON employee.empersonid = care_center.nurseincharge
    WHERE employee.employeetype = p_EmployeeType
    Group By Resident_Patient.RPAPersonID, Visit_Care_Center.OPAPersonID,
Care_Center.NurseInCharge,
    employee.employeetype, Care_Center.CareCenterName;
BEGIN
    OPEN cur_Employee;
    LOOP
        FETCH cur_Employee INTO v_ResidentPatient,  v_OutPatient,  v_Nurse ,
v_EmployeeType, v_CareCenterName;
EXIT WHEN cur_Employee%NOTFOUND;
        --Display each record
        DBMS_OUTPUT.PUT_LINE('Employee info: ' ||v_ResidentPatient|| ' | '
||v_OutPatient|| ' | ' ||v_Nurse||
        ' | ' ||v_EmployeeType|| ' | ' ||v_CareCenterName);
    END LOOP;
    --Close Cursor
    CLOSE cur_Employee;
END usp_RN_NurseCareCenter;
```



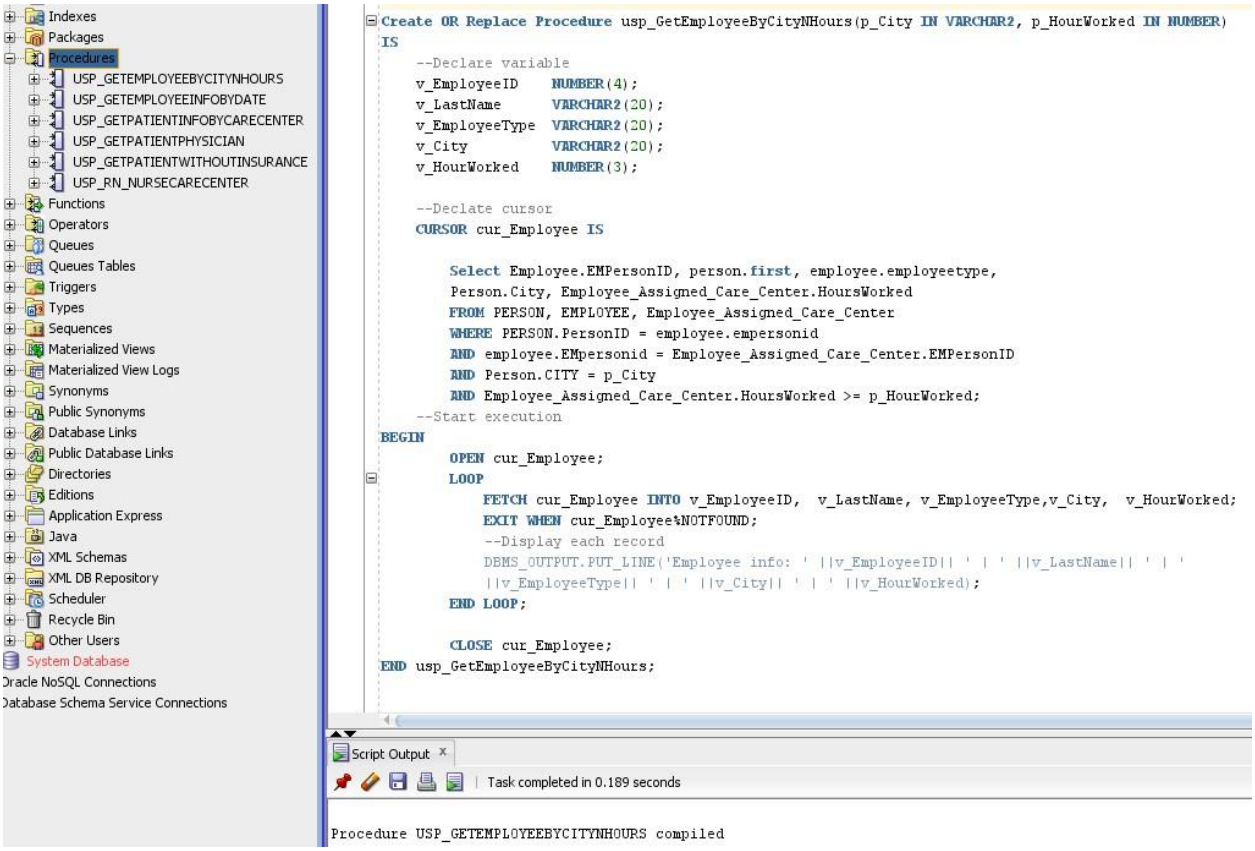

```
SET SERVEROUTPUT ON;
EXECUTE usp_RN_NurseCareCenter('RN Nurse');
```



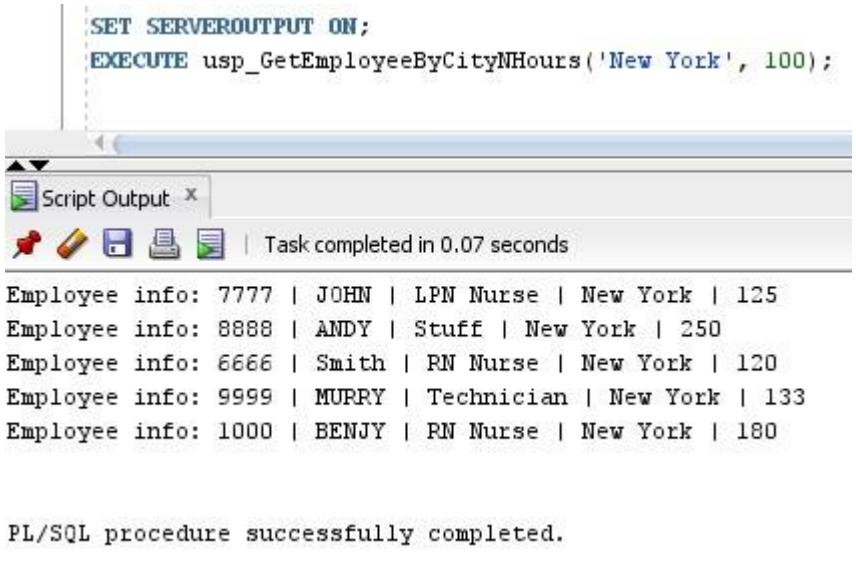

```
Employee info: 1 | 1 | 1 | RN Nurse | URGENT CARE
Employee info: 1 | 1 | 1 | RN Nurse | DELTA CARE


PL/SQL procedure successfully completed.
```

6) **Store Procedure objectives:** To give the employee monthly Metro Card
Persona this Store Procedure is targeting: Employee Supervisor
Decision this Store Procedure will support: This Store Procedure will help the supervisor to make the decision to give the monthly Metro Card to the employee who live in New York and worked more the 100 hours.
Data need to create the Store Procedure: Employee ID, First Name, Employee Type, City and Working Hours
Additional details: Store Procedure that will displays the Employee ID, First Name, Employee Type, City and Working Hours of all the employees who lives in 'New York' and worked more than 100 hours. Business objective for the Supervisor is to give them free monthly Metro Card for public transportation.

```
Create OR Replace Procedure usp_GetEmployeeByCityNHours(p_City IN VARCHAR2, p_HourWorked IN
NUMBER)
IS
    --Declare variable
v_EmployeeID    NUMBER(4);
v_LastName      VARCHAR2(20);
v_EmployeeType  VARCHAR2(20);
v_City          VARCHAR2(20);
v_HourWorked    NUMBER(3);

    --Declate cursor
    CURSOR cur_Employee IS

        Select Employee.EMPersonID, person.first, employee.employeetype,
        Person.City, Employee_Assigned_Care_Center.HoursWorked
        FROM PERSON, EMPLOYEE, Employee_Assigned_Care_Center
        WHERE PERSON.PersonID = employee.empersonid
        AND employee.EMpersonid = Employee_Assigned_Care_Center.EMPersonID
        AND Person.CITY = p_City
        AND Employee_Assigned_Care_Center.HoursWorked >= p_HourWorked;
    --Start execution
BEGIN
        OPEN cur_Employee;
        LOOP
            FETCH cur_Employee INTO v_EmployeeID,  v_LastName, v_EmployeeType,v_City,
v_HourWorked;
            EXIT WHEN cur_Employee%NOTFOUND;
            --Display each record
            DBMS_OUTPUT.PUT_LINE('Employee info: ' ||v_EmployeeID|| ' | ' ||v_LastName|| '
| '
            ||v_EmployeeType|| ' | ' ||v_City|| ' | ' ||v_HourWorked);
END LOOP;

        CLOSE cur_Employee;
END usp_GetEmployeeByCityNHours;
```




```
SET SERVEROUTPUT ON;
EXECUTE usp_GetEmployeeByCityNHours('New York', 100);
```



7) Store Procedure objectives: Store Procedure make schedule for the technician to inspect all the Electric and Automated Bed.

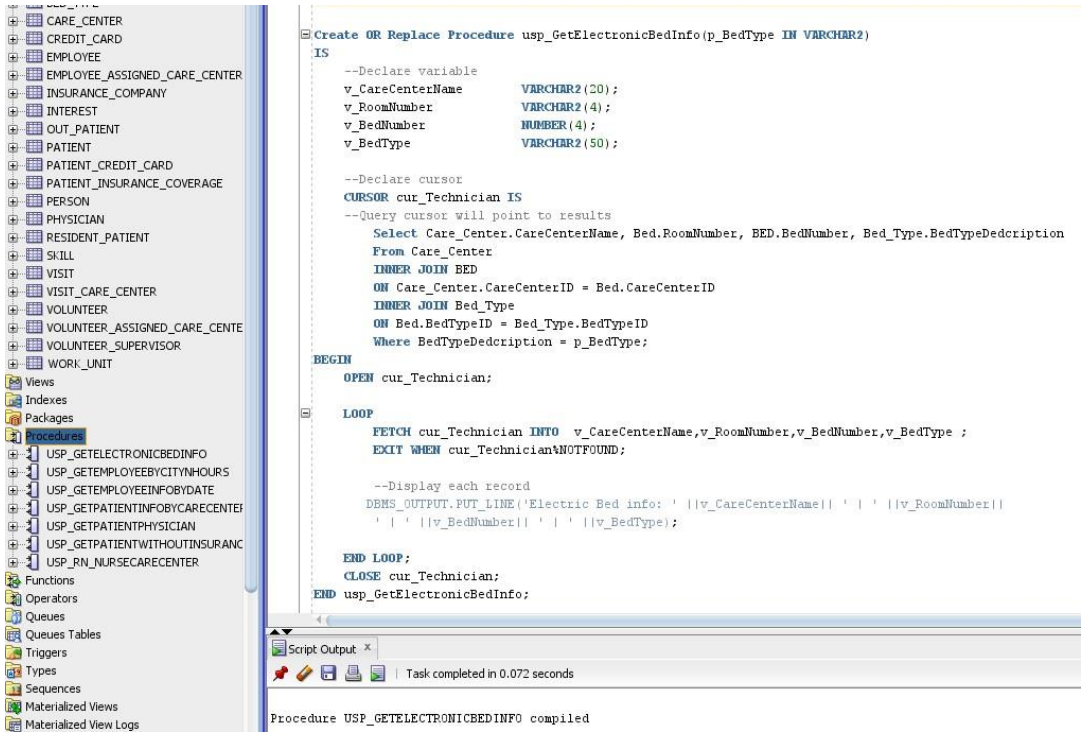
Persona this is targeting: Schedule Manager

Decision this Store Procedure will support: This Store Procedure will help the Schedule Manager to see all the information about the location of electronic bed and send electrician for monthly inspection.

Data need to create the Store Procedure: Care Center Name, Room Number, Bed Number, Bed type description.

Additional details: Store Procedure will display Care center name, room number and bed number will help to navigate where is electric bed are located. This way technician easily can go there and do their work.

```
Create OR Replace Procedure usp_GetElectronicBedInfo(p_BedType IN VARCHAR2)
IS
    --Declare variable
    v_CareCenterName        VARCHAR2(20);
v_RoomNumber            VARCHAR2(4);
v_BedNumber             NUMBER(4);
v_BedType               VARCHAR2(50);

    --Declare cursor
    CURSOR cur_Technician IS
    --Query cursor will point to results
        Select Care_Center.CareCenterName, Bed.RoomNumber, BED.BedNumber,
Bed_Type.BedTypeDedcription
        From Care_Center
        INNER JOIN BED
        ON Care_Center.CareCenterID = Bed.CareCenterID
        INNER JOIN Bed_Type
        ON Bed.BedTypeID = Bed_Type.BedTypeID
        Where BedTypeDedcription = p_BedType;
BEGIN
    OPEN cur_Technician;

    LOOP
        FETCH cur_Technician INTO  v_CareCenterName,v_RoomNumber,v_BedNumber,v_BedType ;
        EXIT WHEN cur_Technician%NOTFOUND;

        --Display each record
       DBMS_OUTPUT.PUT_LINE('Electric Bed info: ' ||v_CareCenterName|| ' | '
||v_RoomNumber||
        ' | ' ||v_BedNumber|| ' | ' ||v_BedType);

    END LOOP;
    CLOSE cur_Technician;
END usp_GetElectronicBedInfo;
```




```
--Turning Server Output On
SET SERVEROUTPUT ON;
--Execute Store Procedure
EXECUTE usp_GetElectronicBedInfo('Electric Account Bed');
```


```
Electric Bed info: DELTA CARE | 5 | 111 | Electric Account Bed
Electric Bed info: URGENT CARE | 4 | 116 | Electric Account Bed


PL/SQL procedure successfully completed.
```

8) Store Procedure objectives: Store Procedure will Follow up call and reschedule out patient

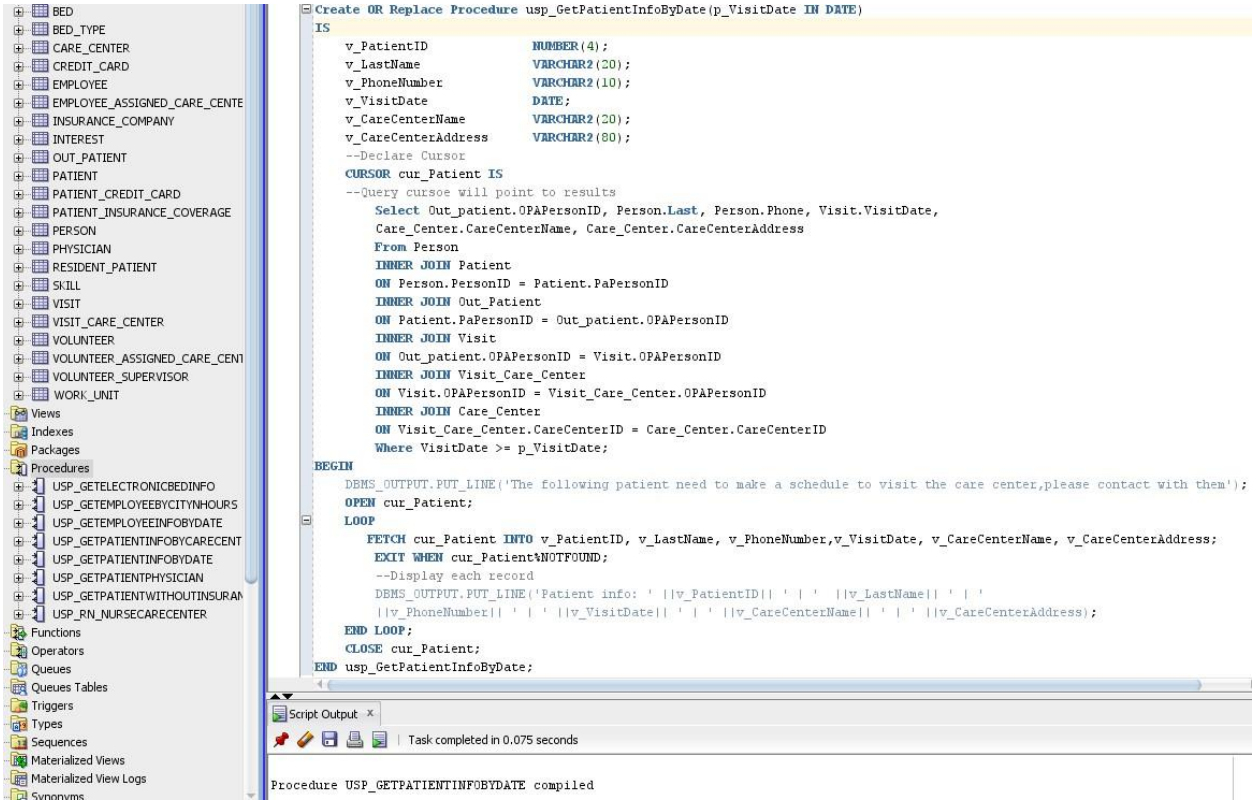
Persona this Store Procedure is targeting: Front-desk assistant

Decision this will support: Based on this Store Procedure front-desk assistant can see who are the patient are visited any given date and able to call them for follow up or reschedule.

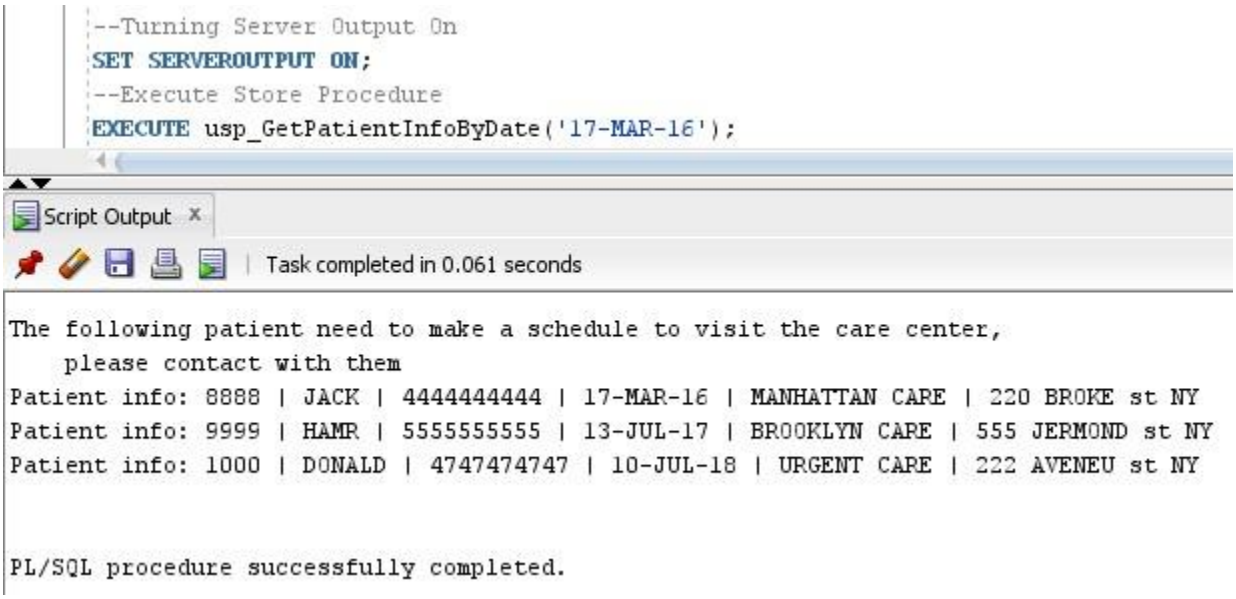
Data need to create the Store Procedure: Our patient ID, Patient Last name, Phone number, Visit Date, Care center name, Care Center address

Additional details: Store Procedure will display patient ID, name and Phone number which is going to help the employee to contact with the patient and also store procedure will display which care center they have visited so then Front-desk assistant can put them in the same care center.

```
Create OR Replace Procedure usp_GetPatientInfoByDate(p_VisitDate IN DATE)
IS
    v_PatientID             NUMBER(4);
v_LastName              VARCHAR2(20);
v_PhoneNumber           VARCHAR2(10);
v_VisitDate             DATE;
v_CareCenterName        VARCHAR2(20);
v_CareCenterAddress     VARCHAR2(80);
    --Declare Cursor
    CURSOR cur_Patient IS
    --Query cursoe will point to results
        Select Out_patient.OPAPersonID, Person.Last, Person.Phone, Visit.VisitDate,
        Care_Center.CareCenterName, Care_Center.CareCenterAddress
        From Person
        INNER JOIN Patient
        ON Person.PersonID = Patient.PaPersonID
        INNER JOIN Out_Patient
        ON Patient.PaPersonID = Out_patient.OPAPersonID
        INNER JOIN Visit
        ON Out_patient.OPAPersonID = Visit.OPAPersonID
        INNER JOIN Visit_Care_Center
        ON Visit.OPAPersonID = Visit_Care_Center.OPAPersonID
        INNER JOIN Care_Center
        ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
        Where VisitDate >= p_VisitDate;
BEGIN
    DBMS_OUTPUT.PUT_LINE('The following patient need to make a schedule to visit the care
center,please contact with them');
    OPEN cur_Patient;
    LOOP
       FETCH cur_Patient INTO v_PatientID, v_LastName,
v_PhoneNumber,v_VisitDate, v_CareCenterName, v_CareCenterAddress;         EXIT
WHEN cur_Patient%NOTFOUND;
        --Display each record
        DBMS_OUTPUT.PUT_LINE('Patient info: ' ||v_PatientID|| ' | '  ||v_LastName|| ' | '
        ||v_PhoneNumber|| ' | ' ||v_VisitDate|| ' | ' ||v_CareCenterName|| ' | '
||v_CareCenterAddress);
    END LOOP;
    CLOSE cur_Patient;
END usp_GetPatientInfoByDate;
```




```
--Turning Server Output On
SET SERVEROUTPUT ON;
--Execute Store Procedure
EXECUTE usp_GetPatientInfoByDate('17-MAR-16');
```



9) Store Procedure objectives: Collect Credit Card Information

Persona this Store Procedure is targeting: Accounts

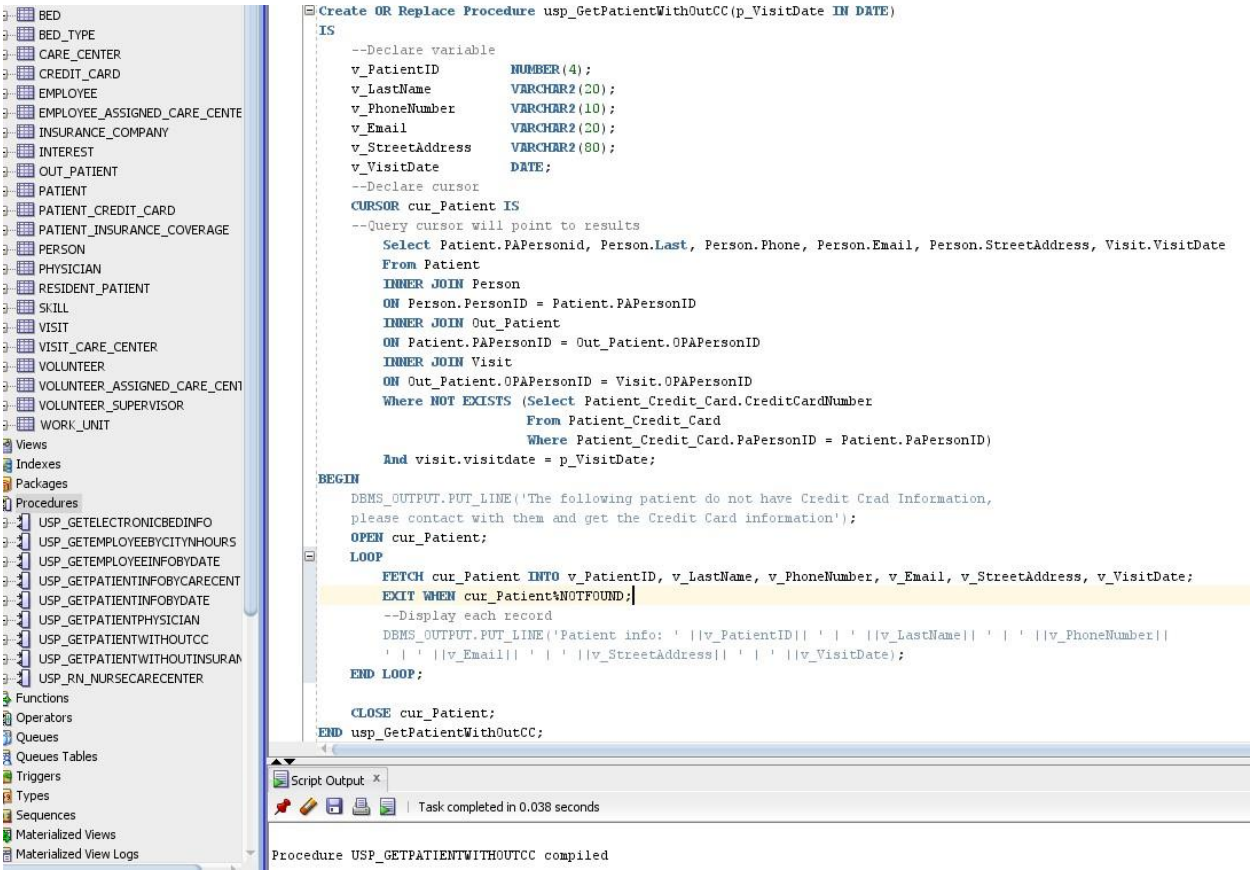
Decision this Store Procedure will support: This Store Procedure will target to those patients who visited particular date and did not have a credit card so hospital Account personal can make decision who to contact.

Data need to create the Store Procedure: Patient ID, Patient Last name, Phone number, Email, Street Address and Visit Date.

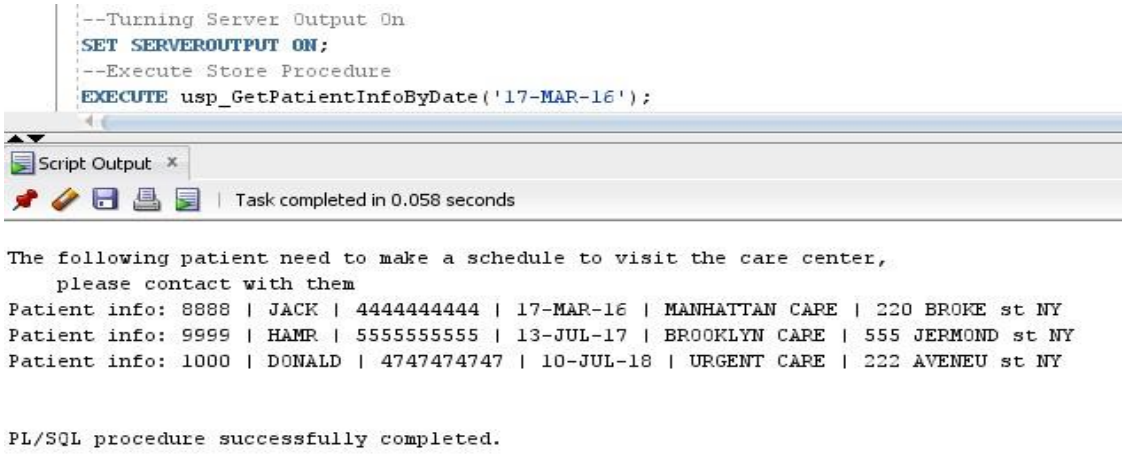
Additional details: Store Procedure will display all the contact information so that Account personal can contact them by phone, email and mailing address are available.

```
Create OR Replace Procedure usp_GetPatientWithOutCC(p_VisitDate IN DATE)
IS
    --Declare variable
v_PatientID         NUMBER(4);
v_LastName          VARCHAR2(20);
v_PhoneNumber       VARCHAR2(10);
v_Email             VARCHAR2(20);
v_StreetAddress     VARCHAR2(80);
v_VisitDate         DATE;     --
Declare cursor
    CURSOR cur_Patient IS
    --Query cursor will point to results
        Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email,
Person.StreetAddress, Visit.VisitDate
        From Patient
        INNER JOIN Person
        ON Person.PersonID = Patient.PAPersonID
        INNER JOIN Out_Patient
        ON Patient.PAPersonID = Out_Patient.OPAPersonID
        INNER JOIN Visit
        ON Out_Patient.OPAPersonID = Visit.OPAPersonID
        Where NOT EXISTS (Select Patient_Credit_Card.CreditCardNumber
                          From Patient_Credit_Card
                          Where Patient_Credit_Card.PaPersonID = Patient.PaPersonID)
And visit.visitdate = p_VisitDate;
BEGIN
    DBMS_OUTPUT.PUT_LINE('The following patient do not have Credit Crad Information,
    please contact with them and get the Credit Card information');
OPEN cur_Patient;
    LOOP
        FETCH cur_Patient INTO v_PatientID, v_LastName, v_PhoneNumber, v_Email,
v_StreetAddress, v_VisitDate;
        EXIT WHEN cur_Patient%NOTFOUND;
        --Display each record
        DBMS_OUTPUT.PUT_LINE('Patient info: ' ||v_PatientID|| ' | ' ||v_LastName|| ' | '
||v_PhoneNumber||
        ' | ' ||v_Email|| ' | ' ||v_StreetAddress|| ' | ' ||v_VisitDate);
    END LOOP;
```

```
    CLOSE cur_Patient;
END usp_GetPatientWithOutCC;
```




```
--Turning Server Output On
SET SERVEROUTPUT ON;
--Execute Store Procedure
EXECUTE usp_GetPatientInfoByDate('17-MAR-16');
```



10) Store Procedure objectives: Hire new volunteer if needed

Persona this Store Procedure is targeting: Volunteer Supervisor

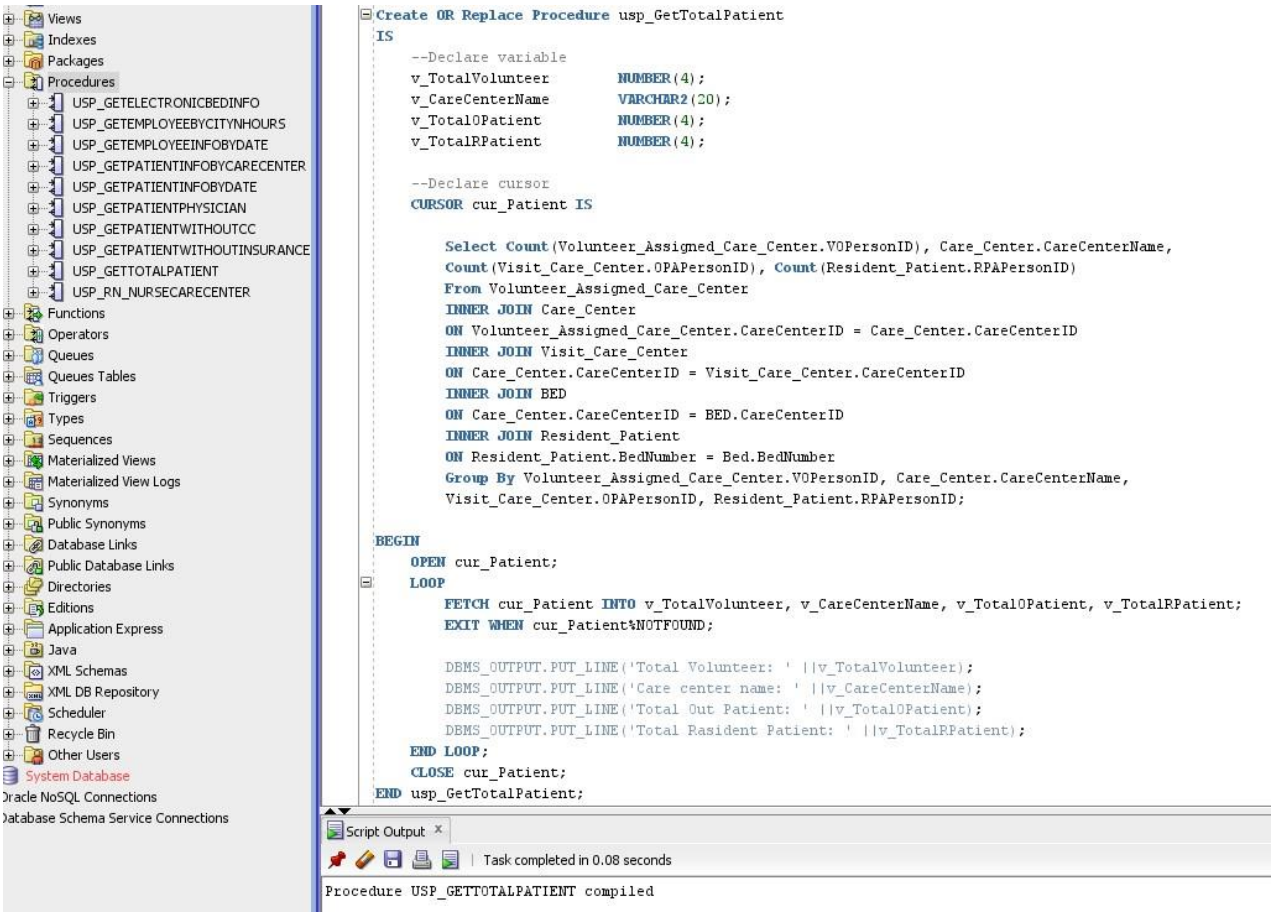
Decision this Store Procedure will support: Based on total number of outpatient and resident patient Volunteer Supervisor will make decision if hospital need more volunteer or not.

Data need to create the Store Procedure: Volunteer ID, Care Center name, Out Patient and resident Patient

Additional details: Store Procedure will display total number of Volunteer, Care center name and total number of resident and outpatient who is currently working in a care center. Business objective for the Volunteer Supervisor to make decision if they need to hire more volunteer for those care center. It's all depends on number of patients those care center has.

```
Create OR Replace Procedure usp_GetTotalPatient
IS
    --Declare variable
    v_TotalVolunteer          NUMBER(4);
v_CareCenterName           VARCHAR2(20);
v_TotalOPatient              NUMBER(4);
v_TotalRPatient         NUMBER(4);

    --Declare cursor
    CURSOR cur_Patient IS

        Select Count(Volunteer_Assigned_Care_Center.VOPersonID),
Care_Center.CareCenterName,
        Count(Visit_Care_Center.OPAPersonID), Count(Resident_Patient.RPAPersonID)
        From Volunteer_Assigned_Care_Center
        INNER JOIN Care_Center
        ON Volunteer_Assigned_Care_Center.CareCenterID = Care_Center.CareCenterID
        INNER JOIN Visit_Care_Center
        ON Care_Center.CareCenterID = Visit_Care_Center.CareCenterID
        INNER JOIN BED
        ON Care_Center.CareCenterID = BED.CareCenterID
        INNER JOIN Resident_Patient
        ON Resident_Patient.BedNumber = Bed.BedNumber
        Group By Volunteer_Assigned_Care_Center.VOPersonID, Care_Center.CareCenterName,
        Visit_Care_Center.OPAPersonID, Resident_Patient.RPAPersonID;
BEGIN
    OPEN cur_Patient;
    LOOP
        FETCH cur_Patient INTO v_TotalVolunteer, v_CareCenterName, v_TotalOPatient,
v_TotalRPatient;
        EXIT WHEN cur_Patient%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Total Volunteer: ' ||v_TotalVolunteer);
        DBMS_OUTPUT.PUT_LINE('Care center name: ' ||v_CareCenterName);
        DBMS_OUTPUT.PUT_LINE('Total Out Patient: ' ||v_TotalOPatient);
```

```
        DBMS_OUTPUT.PUT_LINE('Total Rasident Patient: ' ||v_TotalRPatient);
END LOOP;
    CLOSE cur_Patient;
END usp_GetTotalPatient;
```



```
--Turning Server Output On SET
SERVEROUTPUT ON;
--Execute Store Procedure
EXECUTE usp_GetTotalPatient;
```

```
--Turning Server Output On
SET SERVEROUTPUT ON;
--Execute Store Procedure
EXECUTE usp_GetTotalPatient;
```

Script Output

Task completed in 0.087 seconds

```
Total Volunteer: 1
Care center name: DELTA CARE
Total Out Patient: 1
Total Rasident Patient: 1
Total Volunteer: 1
Care center name: QUEENS CARE
Total Out Patient: 1
Total Rasident Patient: 1
Total Volunteer: 1
Care center name: URGENT CARE
Total Out Patient: 1
Total Rasident Patient: 1
Total Volunteer: 1
Care center name: MANHATTAN CARE
Total Out Patient: 1
Total Rasident Patient: 1
Total Volunteer: 1
Care center name: BROOKLYN CARE
Total Out Patient: 1
Total Rasident Patient: 1


PL/SQL procedure successfully completed.
```

Conclusion

This section was the upgrade version 2.0 of the Hospital Management System Database. Project management methodology used the database application development lifecycle. We were following all the phase do this upgrade, start with planning, analysis, design, development and maintenance phase to upgrade this version by creating business reports and improve the performance of full working application. We also created business reports Store Procedure for the hospital management system. All the store procedure has been created based on the business scenario that hospital might face. Those reports are important for hospital Physician, supervisor, Frond-desk assistant and many other employees. They might need every week or month in order to run the business.

Hospital Management System Database Design & Implementation

Version 3.0

Upgrade Objectives

This section is the upgrade version 3.0 of the Hospital Management System Database. Project management methodology using the database application development lifecycle. We were following all the phase do this upgrade, start with planning, analysis, design, development and maintenance phase to upgrade this version by creating business reports and improve the performance of full working application. We also created Index for the business reports. Index will increase the performance for runtime of business reports. We also created View for the query reports. Some of the Views covered many query reports. In order to increase the performance of Hospital Management Systems Database we had to create the Index and View in this upgraded version.

Create Secondary Index Statements

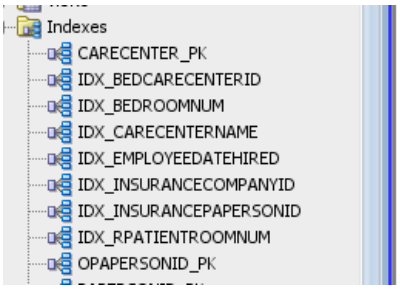
Business Report #1

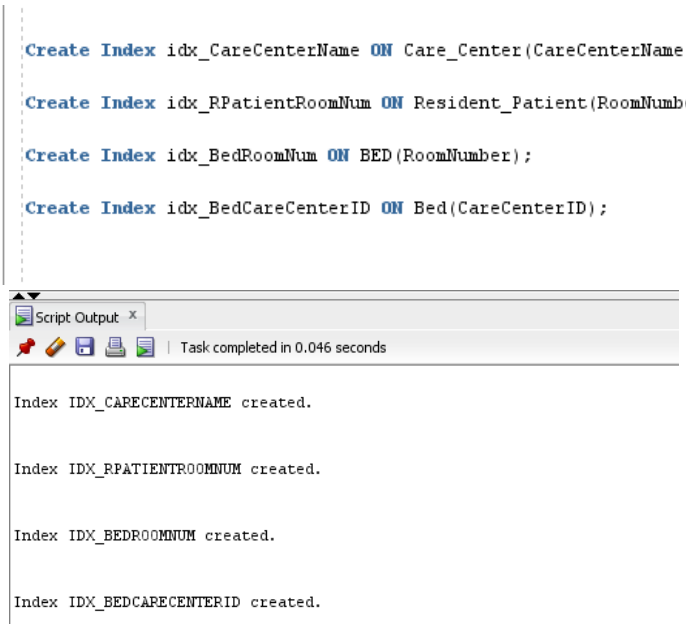
Query description: Based on the information in this report supervisor will know patient's ID and Last name which will help to identify the patient and Care Center name, room number and bed number to identify the location of that patient.

```
SELECT Resident_Patient.RPAPersonid,Person.Last,Care_Center.CareCenterName,
BED.RoomNumber, BED.bednumber
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Resident_Patient
ON Patient.PAPersonID = Resident_Patient.RPAPersonID
INNER JOIN BED
ON Resident_Patient.RoomNumber = BED.RoomNumber
INNER JOIN Care_Center
ON Bed.CareCenterID = Care_Center.CareCenterID
Where CareCenterName = 'URGENT CARE';
```

Create Secondary Index Statement for Business Report #1

```
Create Index idx_CareCenterName ON Care_Center(CareCenterName);
Create Index idx_RPatientRoomNum ON Resident_Patient(RoomNumber);
Create Index idx_BedRoomNum ON BED(RoomNumber);
Create Index idx_BedCareCenterID ON Bed(CareCenterID);
```





Business Report #2

Query description: This report will provide the total number of patient and physician are exist in this hospital and based on this report hiring manager will decide if they need to hire more physician in this hospital

```
select   count(case when IsPatient = 'Y' then 1 end) as IsPatient,
         count(case when IsPhysician = 'Y' then 1 end) as IsPhysician

from PERSON c
group by c.ispatient, c.IsPhysician;
```

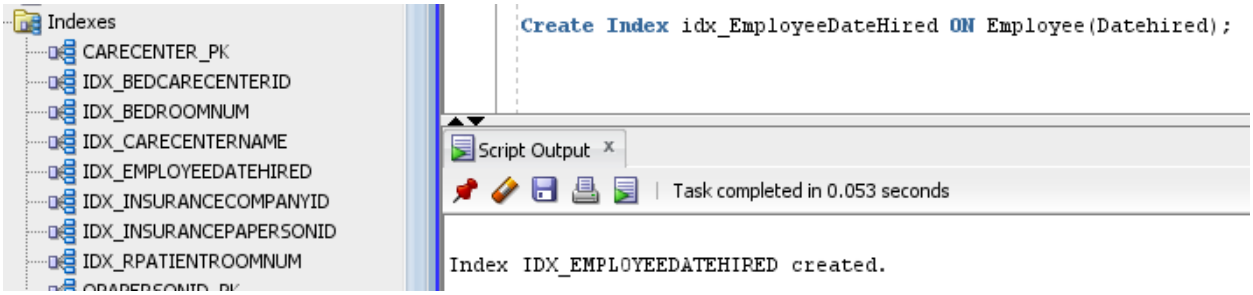
- Primary Key Index for PERSON table on PersonID. Oracle does not support GROUP BY so No secondary index required for IsPatient and IsPhysician.

Business Report #3

```
Select EMPersonID,Datehired,employeetype
from Employee
where datehired < '01-JAN-15';
```

Create Secondary Index Statement for Business Report #3

```
Create Index idx_EmployeeDateHired ON Employee(Datehired);
```



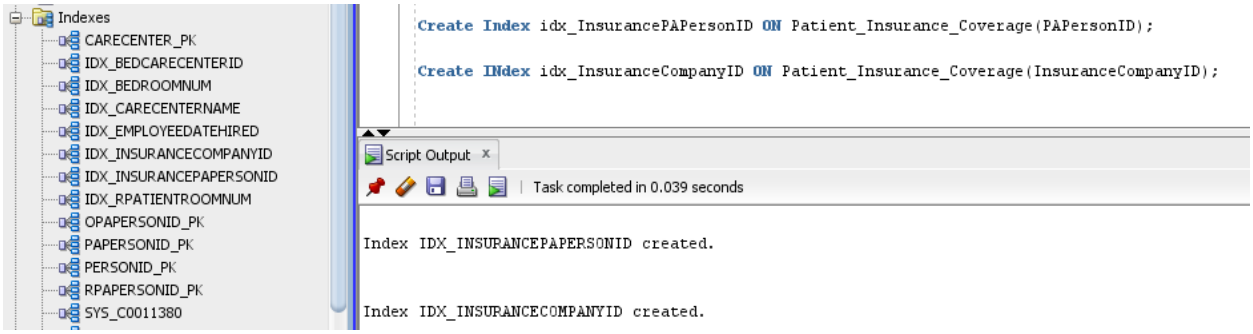
Business Report #4

Query description: Query report will display the Patient ID, Contact Name and Contact Number of all the patients who does not have an insurance coverage. Business objectives for the Marketing people is to reach out those patients and send them free full coverage insurance offer.

```
Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email, Person.StreetAddress
From Patient
INNER JOIN Person
ON PersonID = PAPersonID
Where NOT EXISTS (Select Patient_Insurance_Coverage.InsuranceCompanyID
                  From Patient_Insurance_Coverage
                  Where Patient_Insurance_Coverage.PAPersonID = Patient.PAPersonID);
```

Create Secondary Index Statement for Business Report #4

```
Create Index idx_InsurancePAPersonID ON Patient_Insurance_Coverage(PAPersonID);

Create INdex idx_InsuranceCompanyID ON Patient_Insurance_Coverage(InsuranceCompanyID);
```



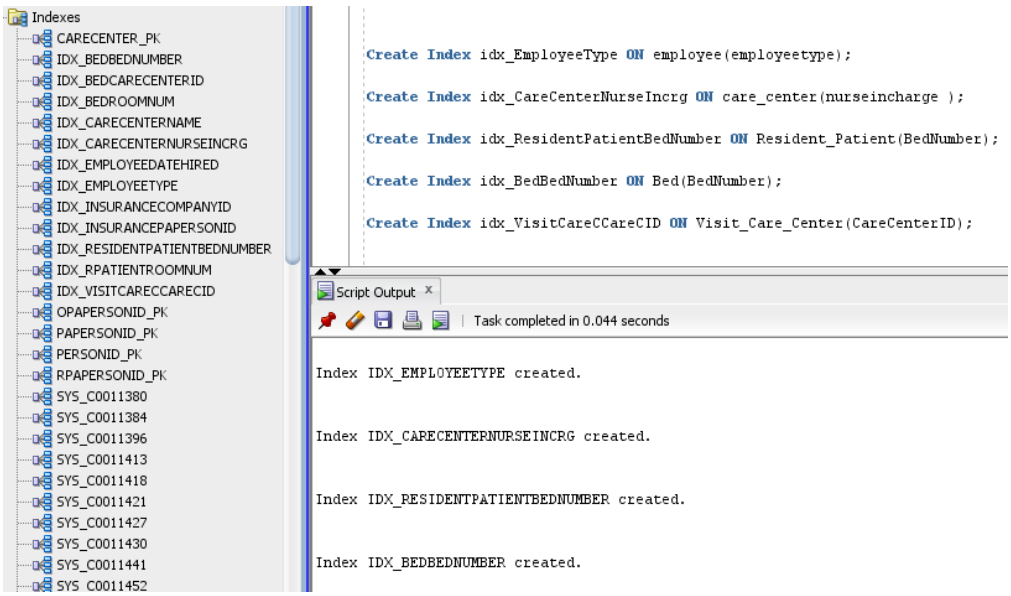
Business Report #5

Query description: Query report will help the hiring manager to make decision if they need to hire more RN Nurse for the care center. We know resident patient and outpatient both can admit to the care center so we have both patient's information and also report will show the RN Nurse who is working in the care center.

```
Select Count(Resident_Patient.RPAPersonID), Count(Visit_Care_Center.OPAPersonID),
Count(Care_Center.NurseInCharge),
employee.employeetype, Care_Center.CareCenterName
From Visit_Care_Center
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
INNER JOIN employee
ON employee.empersonid = care_center.nurseincharge
WHERE employee.employeetype = 'RN Nurse'
Group By Resident_Patient.RPAPersonID, Visit_Care_Center.OPAPersonID,
Care_Center.NurseInCharge,
employee.employeetype, Care_Center.CareCenterName;
```

Create Secondary Index Statement for Business Report #5

```
Create Index idx_EmployeeType ON employee(employeetype);
Create Index idx_CareCenterNurseIncrg ON care_center(nurseincharge );
Create Index idx_ResidentPatientBedNumber ON Resident_Patient(BedNumber);
Create Index idx_BedBedNumber ON Bed(BedNumber);
Create Index idx_VisitCareCCareCID ON Visit_Care_Center(CareCenterID);
```



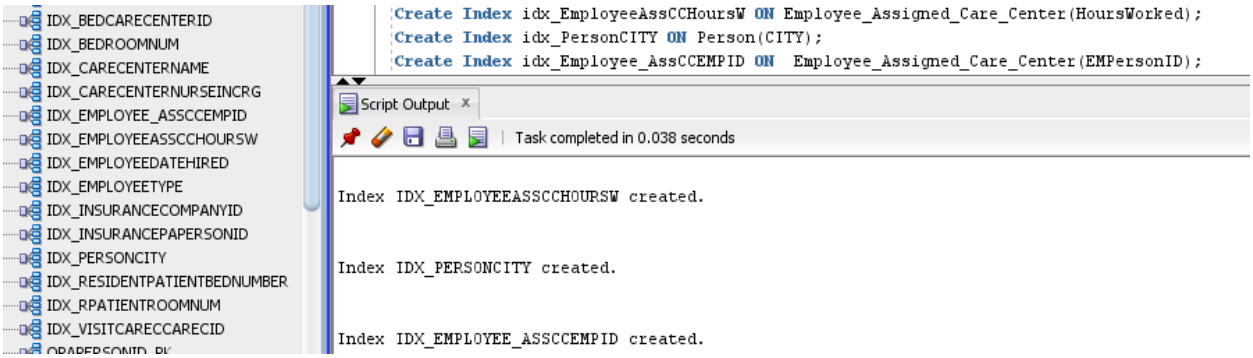
Business Report #6

Query description: Query report that will displays the Employee ID, First Name, Employee Type, City and Working Hours of all the employees who lives in 'New York' and worked more then 100 hours. Business objective for the Supervisor is to give them free monthly Metro Card for public transportation.

```
Select Employee.EMPersonID, person.first, employee.employeetype,
Person.City, Employee_Assigned_Care_Center.HoursWorked
FROM PERSON, EMPLOYEE, Employee_Assigned_Care_Center
WHERE PERSON.PersonID = employee.empersonid
AND employee.EMpersonid = Employee_Assigned_Care_Center.EMPersonID
AND Person.CITY = 'New York'
AND Employee_Assigned_Care_Center.HoursWorked >= 100;
```

Create Secondary Index Statement for Business Report #6

```
Create Index idx_EmployeeAssCCHoursW ON Employee_Assigned_Care_Center(HoursWorked);
Create Index idx_PersonCITY ON Person(CITY);
Create Index idx_Employee_AssCCEMPID ON  Employee_Assigned_Care_Center(EMPersonID);
```



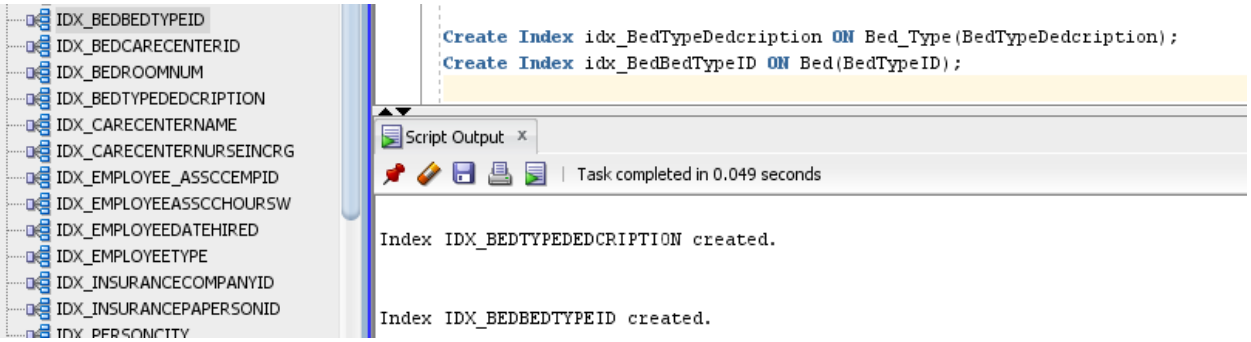
Business Report #7

Query description: Report will display Care center name, room number and bed number will help to navigate where is electric bed are located. This way technician easily can go there and do their work.

```
Select Care_Center.CareCenterName, Bed.RoomNumber, BED.BedNumber,
Bed_Type.BedTypeDedcription

From Care_Center
INNER JOIN BED
ON Care_Center.CareCenterID = Bed.CareCenterID
INNER JOIN Bed_Type
ON Bed.BedTypeID = Bed_Type.BedTypeID
Where BedTypeDedcription = 'Electric Account Bed';
```

Create Secondary Index Statement for Business Report #7

```
Create Index idx_BedTypeDedcription ON Bed_Type(BedTypeDedcription);
Create Index idx_BedBedTypeID ON Bed(BedTypeID);
```



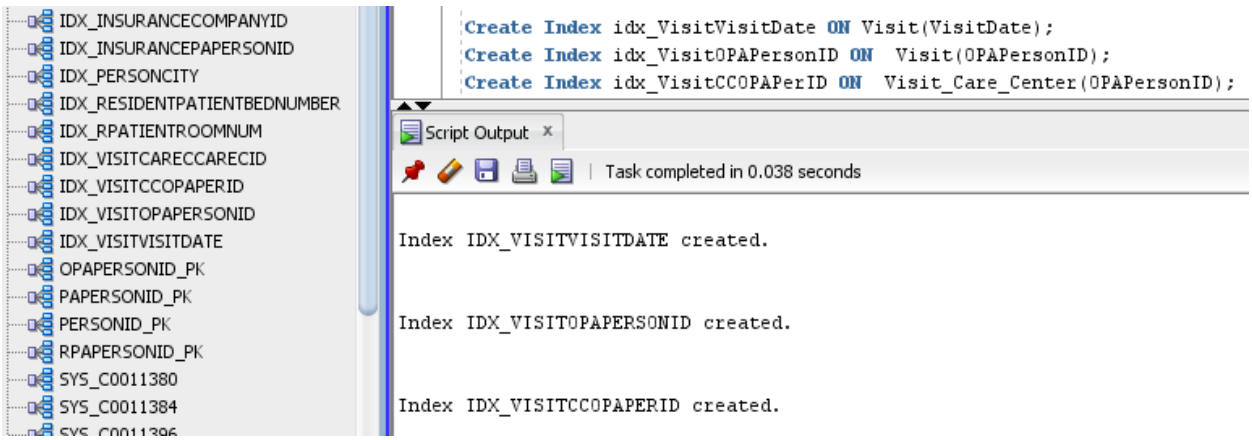
Business Report #8

Query description: Report will display patient ID, name and Phone number which is going to help the employee to contact with the patient and also report will display which care center they have visited so then Front-desk assistant can put them in the same care center.

```
Select Out_patient.OPAPersonID, Person.Last, Person.Phone, Visit.VisitDate,
Care_Center.CareCenterName, Care_Center.CareCenterAddress
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PaPersonID
INNER JOIN Out_Patient
ON Patient.PaPersonID = Out_patient.OPAPersonID
INNER JOIN Visit
ON Out_patient.OPAPersonID = Visit.OPAPersonID
INNER JOIN Visit_Care_Center
ON Visit.OPAPersonID = Visit_Care_Center.OPAPersonID
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
Where VisitDate >= '17-MAR-16';
```

Create Secondary Index Statement for Business Report #8

```
Create Index idx_VisitVisitDate ON Visit(VisitDate);
Create Index idx_VisitOPAPersonID ON  Visit(OPAPersonID);
Create Index idx_VisitCCOPAPerID ON  Visit_Care_Center(OPAPersonID);
```



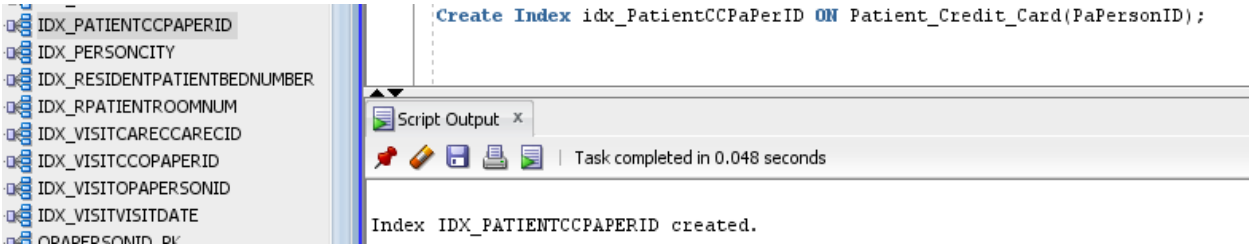
Business Report #9

Query description: Query report will display all the contact information so that Account personal can contact them by phone, email and mailing address are available.

```
Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email, Person.StreetAddress,
Visit.VisitDate
From Patient
INNER JOIN Person
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Out_Patient
ON Patient.PAPersonID = Out_Patient.OPAPersonID
INNER JOIN Visit
ON Out_Patient.OPAPersonID = Visit.OPAPersonID
Where NOT EXISTS (Select Patient_Credit_Card.CreditCardNumber
                  From Patient_Credit_Card
                  Where Patient_Credit_Card.PaPersonID = Patient.PaPersonID)
And visit.visitdate = '17-MAR-16';
```

Create Secondary Index Statement for Business Report #9

```
Create Index idx_PatientCCPaPerID ON Patient_Credit_Card(PaPersonID);
```



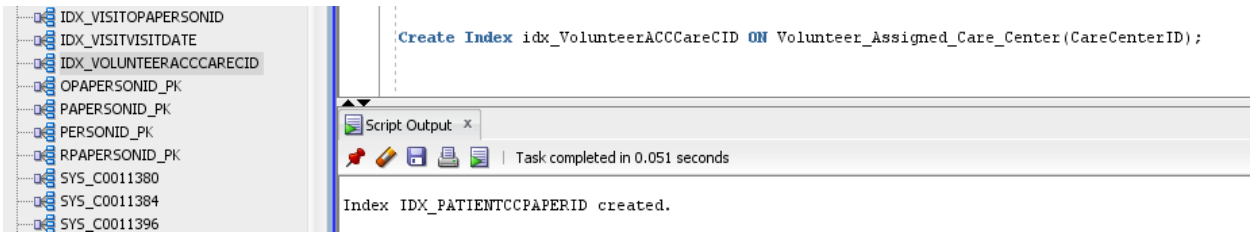
Business Report #10

Query description: Query report will display total number of Volunteer, Care center name and total number of resident and outpatient who is currently working in a care center. Business objective for the Volunteer Supervisor to make decision if they need to hire more volunteer for those care center. It's all depends on number of patients those care center has.

```
Select Count(Volunteer_Assigned_Care_Center.VOPersonID), Care_Center.CareCenterName,
Count(Visit_Care_Center.OPAPersonID), Count(Resident_Patient.RPAPersonID)
From Volunteer_Assigned_Care_Center
INNER JOIN Care_Center
ON Volunteer_Assigned_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN Visit_Care_Center
ON Care_Center.CareCenterID = Visit_Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
Group By Volunteer_Assigned_Care_Center.VOPersonID, Care_Center.CareCenterName,
Visit_Care_Center.OPAPersonID, Resident_Patient.RPAPersonID;
```

Create Secondary Index Statement for Business Report #10

```
Create Index idx_VolunteerACCCareCID ON Volunteer_Assigned_Care_Center(CareCenterID);
```



Create View Statements

View #1

Here is some Business Reports from Hospital Management System. Based on those Business Reports we are going to create a View, name Person_RPatient

Business Report #1

```
SELECT Resident_Patient.RPAPersonid,Person.Last,Care_Center.CareCenterName,
BED.RoomNumber, BED.bednumber
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Resident_Patient
ON Patient.PAPersonID = Resident_Patient.RPAPersonID
INNER JOIN BED
ON Resident_Patient.RoomNumber = BED.RoomNumber
INNER JOIN Care_Center
ON Bed.CareCenterID = Care_Center.CareCenterID
Where CareCenterName = 'URGENT CARE';
```

Business Report #2

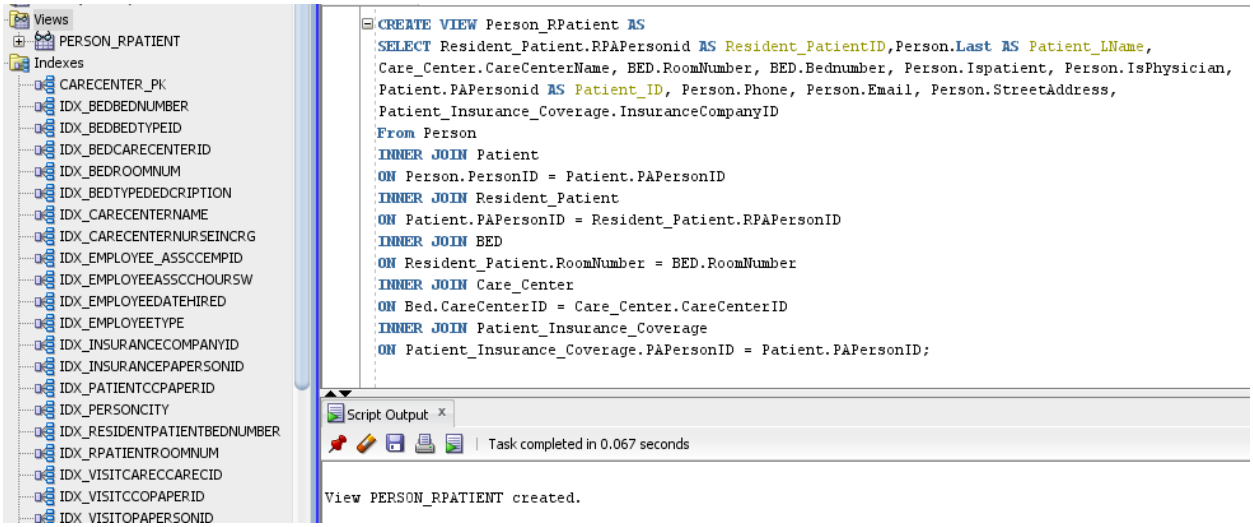
```
select   count(case when IsPatient = 'Y' then 1 end) as IsPatient,
         count(case when IsPhysician = 'Y' then 1 end) as IsPhysician

from PERSON c
group by c.ispatient, c.IsPhysician;
```

Business Report #4

```
Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email,
Person.StreetAddress
From Patient
INNER JOIN Person
ON PersonID = PAPersonID
Where NOT EXISTS (Select Patient_Insurance_Coverage.InsuranceCompanyID
                  From Patient_Insurance_Coverage
                  Where Patient_Insurance_Coverage.PAPersonID = Patient.PAPersonID);
```

Create View for Business Report 1,2 and 4:

```
CREATE VIEW Person_RPatient AS
SELECT Resident_Patient.RPAPersonid AS Resident_PatientID,Person.Last AS
Patient_LName,
Care_Center.CareCenterName, BED.RoomNumber, BED.Bednumber, Person.Ispatient,
Person.IsPhysician,
Patient.PAPersonid AS Patient_ID, Person.Phone, Person.Email,
Person.StreetAddress,
Patient_Insurance_Coverage.InsuranceCompanyID
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Resident_Patient
ON Patient.PAPersonID = Resident_Patient.RPAPersonID
INNER JOIN BED
ON Resident_Patient.RoomNumber = BED.RoomNumber
INNER JOIN Care_Center
ON Bed.CareCenterID = Care_Center.CareCenterID
INNER JOIN Patient_Insurance_Coverage
ON Patient_Insurance_Coverage.PAPersonID = Patient.PAPersonID;
```



View #2

Here is some Business Reports from Hospital Management System. Based on those Business Reports we are going to create a View name Person_Employee.

Business Report #3

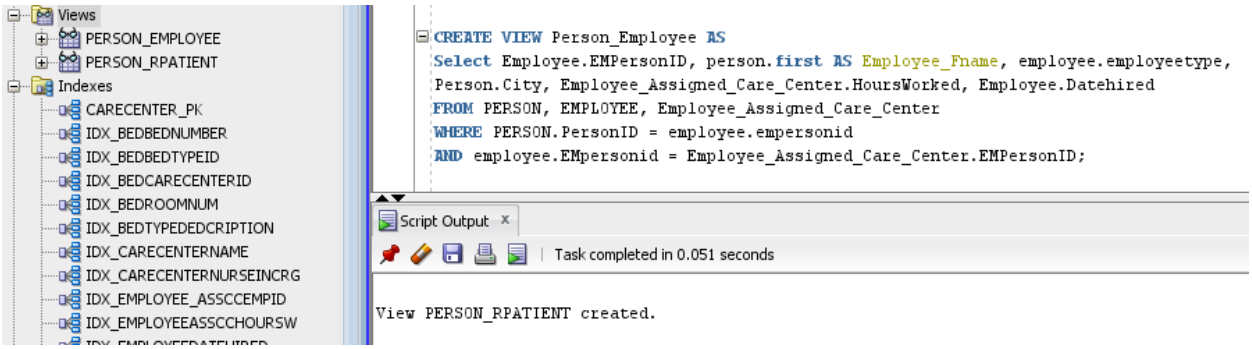
```
Select EMPersonID,Datehired,employeetype
from Employee
where datehired < '01-JAN-15';
```

Business Report #6

```
Select Employee.EMPersonID, person.first, employee.employeetype,
Person.City, Employee_Assigned_Care_Center.HoursWorked
FROM PERSON, EMPLOYEE, Employee_Assigned_Care_Center
WHERE PERSON.PersonID = employee.empersonid
AND employee.EMpersonid = Employee_Assigned_Care_Center.EMPersonID
AND Person.CITY = 'New York'
AND Employee_Assigned_Care_Center.HoursWorked >= 100;
```

Create View for Business Report 3 and 6:

```
CREATE VIEW Person_Employee AS
Select Employee.EMPersonID, person.first AS Employee_Fname,
employee.employeetype,
Person.City, Employee_Assigned_Care_Center.HoursWorked, Employee.Datehired
FROM PERSON, EMPLOYEE, Employee_Assigned_Care_Center
WHERE PERSON.PersonID = employee.empersonid
AND employee.EMpersonid = Employee_Assigned_Care_Center.EMPersonID;
```



View #3

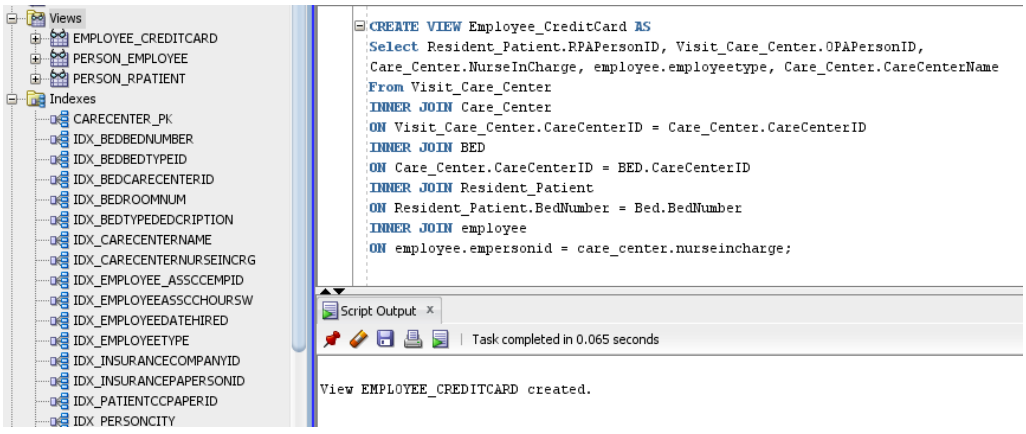
Here is a Business Reports from Hospital Management System. Based on those Business Reports we are going to create a View name Employee_CreditCard.

Business Report #5

```
Select Count(Resident_Patient.RPAPersonID),
Count(Visit_Care_Center.OPAPersonID), Count(Care_Center.NurseInCharge),
employee.employeetype, Care_Center.CareCenterName
From Visit_Care_Center
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
INNER JOIN employee
ON employee.empersonid = care_center.nurseincharge
WHERE employee.employeetype = 'RN Nurse'
Group By Resident_Patient.RPAPersonID, Visit_Care_Center.OPAPersonID,
Care_Center.NurseInCharge,
employee.employeetype, Care_Center.CareCenterName;
```

Create View for Business Report #5

```
 CREATE VIEW Employee_CreditCard AS
Select Resident_Patient.RPAPersonID, Visit_Care_Center.OPAPersonID,
Care_Center.NurseInCharge, employee.employeetype, Care_Center.CareCenterName
From Visit_Care_Center
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
INNER JOIN employee
ON employee.empersonid = care_center.nurseincharge;
```



View #4

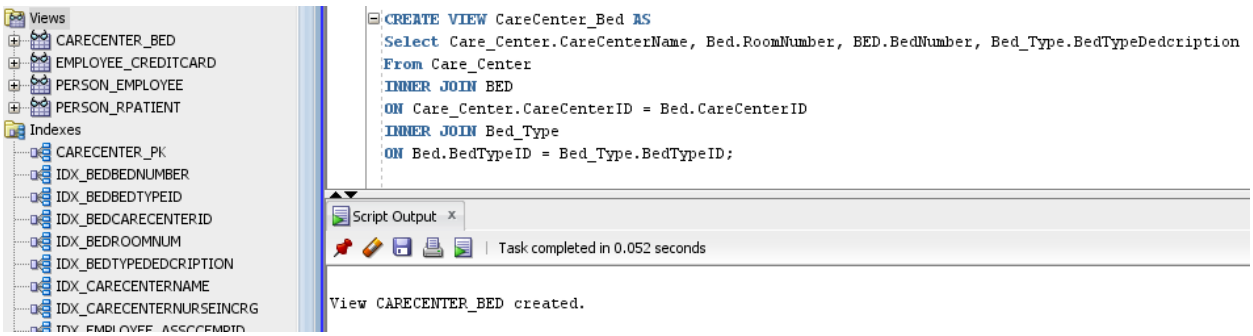
Here is a Business Reports from Hospital Management System. Based on those Business Reports we are going to create a View name CareCenter_Bed.

Business Report #7

```
Select Care_Center.CareCenterName, Bed.RoomNumber, BED.BedNumber,
Bed_Type.BedTypeDedcription
From Care_Center
INNER JOIN BED
ON Care_Center.CareCenterID = Bed.CareCenterID
INNER JOIN Bed_Type
ON Bed.BedTypeID = Bed_Type.BedTypeID
Where BedTypeDedcription = 'Electric Account Bed';
```

Create View for Business Report #7

```
CREATE VIEW CareCenter_Bed AS
Select Care_Center.CareCenterName, Bed.RoomNumber, BED.BedNumber,
Bed_Type.BedTypeDedcription
From Care_Center
INNER JOIN BED
ON Care_Center.CareCenterID = Bed.CareCenterID
INNER JOIN Bed_Type
ON Bed.BedTypeID = Bed_Type.BedTypeID;
```



View #5

Here is some Business Reports from Hospital Management System. Based on those Business Reports we are going to create a View name Person_OutPatient

Business Report #8

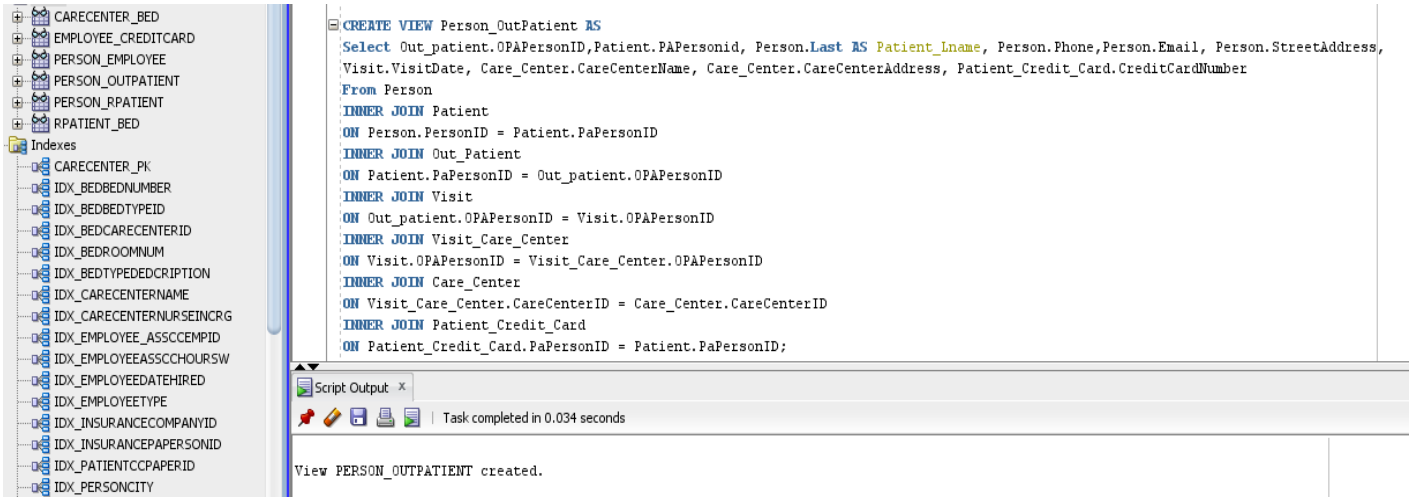
```
Select Out_patient.OPAPersonID, Person.Last, Person.Phone, Visit.VisitDate,
Care_Center.CareCenterName, Care_Center.CareCenterAddress
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PaPersonID
INNER JOIN Out_Patient
ON Patient.PaPersonID = Out_patient.OPAPersonID
INNER JOIN Visit
ON Out_patient.OPAPersonID = Visit.OPAPersonID
INNER JOIN Visit_Care_Center
ON Visit.OPAPersonID = Visit_Care_Center.OPAPersonID
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
Where VisitDate >= '17-MAR-16';
```

Business Report #9

```
Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email, Person.StreetAddress,
Visit.VisitDate
From Patient
INNER JOIN Person
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Out_Patient
ON Patient.PAPersonID = Out_Patient.OPAPersonID
INNER JOIN Visit
ON Out_Patient.OPAPersonID = Visit.OPAPersonID
Where NOT EXISTS (Select Patient_Credit_Card.CreditCardNumber
                  From Patient_Credit_Card
                  Where Patient_Credit_Card.PaPersonID = Patient.PaPersonID)
And visit.visitdate = '17-MAR-16';
```

Create View for Business Report 8 and 9:

```
CREATE VIEW Person_OutPatient AS
Select Out_patient.OPAPersonID,Patient.PAPersonid, Person.Last AS
Patient_Lname, Person.Phone,Person.Email, Person.StreetAddress,
Visit.VisitDate, Care_Center.CareCenterName, Care_Center.CareCenterAddress,
Patient_Credit_Card.CreditCardNumber
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PaPersonID
INNER JOIN Out_Patient
ON Patient.PaPersonID = Out_patient.OPAPersonID
INNER JOIN Visit
ON Out_patient.OPAPersonID = Visit.OPAPersonID
INNER JOIN Visit_Care_Center
ON Visit.OPAPersonID = Visit_Care_Center.OPAPersonID
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN Patient_Credit_Card
ON Patient_Credit_Card.PaPersonID = Patient.PaPersonID;
```



View #6

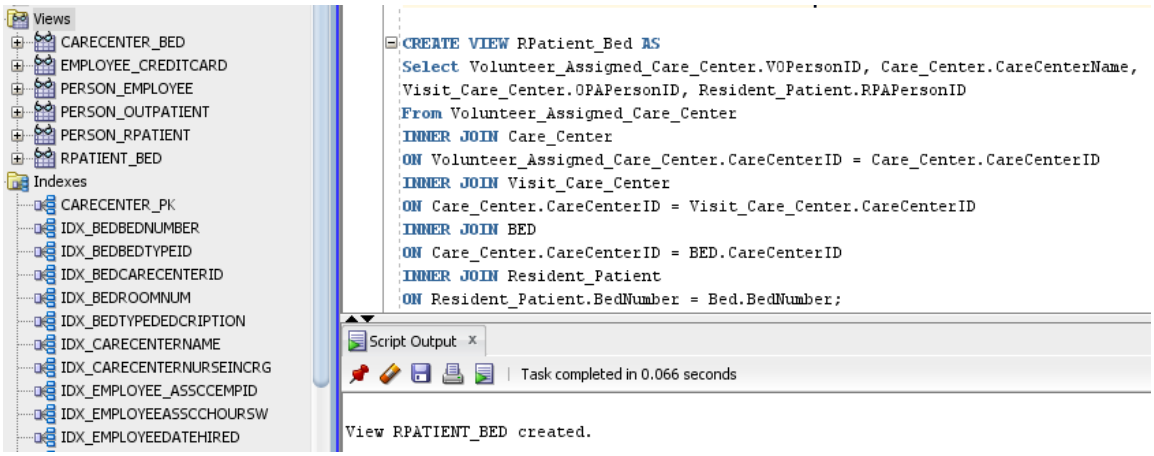
Here is a Business Reports from Hospital Management System. Based on those Business Reports we are going to create a View name RPatient_Bed

Business Report #10

```
Select Count(Volunteer_Assigned_Care_Center.VOPersonID),
Care_Center.CareCenterName,
Count(Visit_Care_Center.OPAPersonID), Count(Resident_Patient.RPAPersonID)
From Volunteer_Assigned_Care_Center
INNER JOIN Care_Center
ON Volunteer_Assigned_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN Visit_Care_Center
ON Care_Center.CareCenterID = Visit_Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
Group By Volunteer_Assigned_Care_Center.VOPersonID,
Care_Center.CareCenterName,
Visit_Care_Center.OPAPersonID, Resident_Patient.RPAPersonID;
```

Create View for Business Report #10

```
CREATE VIEW RPatient_Bed AS
Select Volunteer_Assigned_Care_Center.VOPersonID, Care_Center.CareCenterName,
Visit_Care_Center.OPAPersonID, Resident_Patient.RPAPersonID
From Volunteer_Assigned_Care_Center
INNER JOIN Care_Center
ON Volunteer_Assigned_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN Visit_Care_Center
ON Care_Center.CareCenterID = Visit_Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber;
```



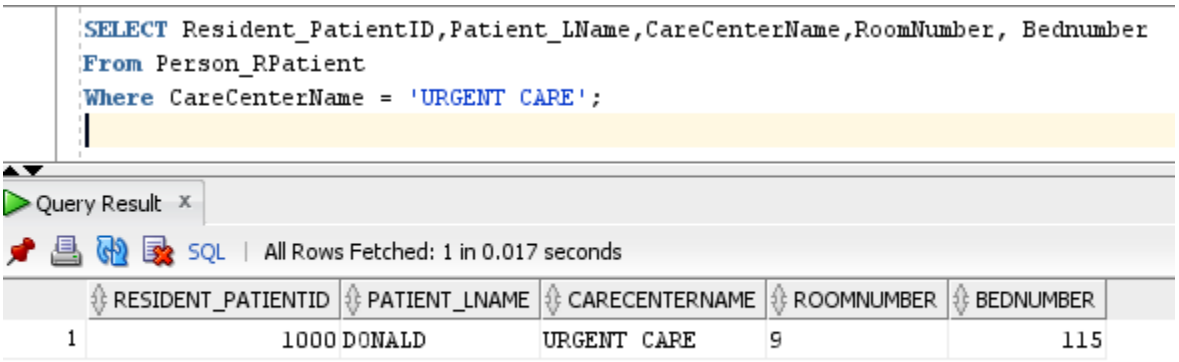
10 Business Reports Using Views

1) Report objectives: Report objective is to serve the food for resident patient.
Persona this report is targeting: Supervisor
Decision this report will support: Based on this report room service employee can make decision where and who to serve the food.
Data need to create the report: We need Resident patient ID, Patient last name, Room number, Bed number

```
SELECT Resident_Patient.RPAPersonid,Person.Last,Care_Center.CareCenterName,
BED.RoomNumber, BED.bednumber
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Resident_Patient
ON Patient.PAPersonID = Resident_Patient.RPAPersonID
INNER JOIN BED
ON Resident_Patient.RoomNumber = BED.RoomNumber
INNER JOIN Care_Center
ON Bed.CareCenterID = Care_Center.CareCenterID
Where CareCenterName = 'URGENT CARE';
```

NEW QUERY REPORT:

```
SELECT Resident_PatientID,Patient_LName,CareCenterName,RoomNumber, Bednumber
From Person_RPatient
Where CareCenterName = 'URGENT CARE';
```



53) Report objectives: The objective is to see if Hospital management need to hire more physician

Persona this report is targeting: Hiring Manager

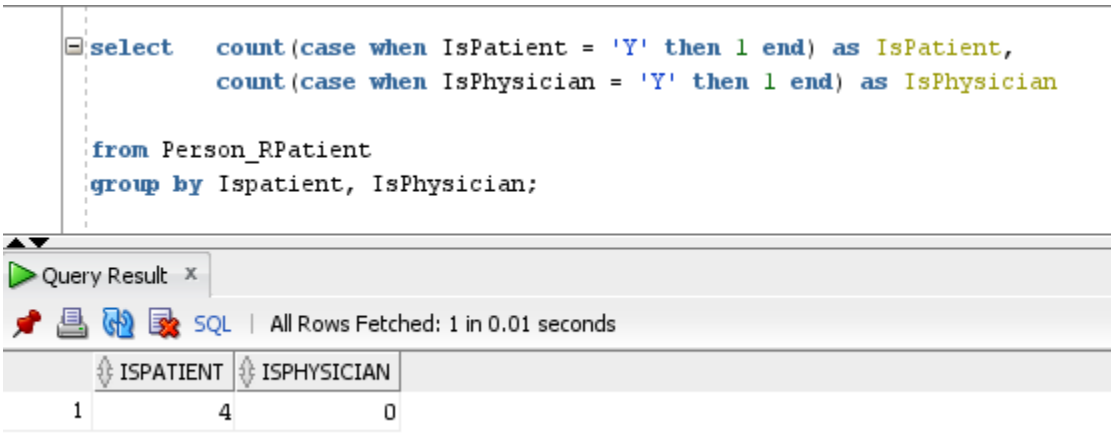
Decision this report will support: Hiring Manager will make decision if they need to hire more physician in this hospital.

Data need to create the report: We need total number of patient and physician

```
select    count(case when IsPatient = 'Y' then 1 end) as IsPatient,
          count(case when IsPhysician = 'Y' then 1 end) as IsPhysician

from PERSON c
group by c.ispatient, c.IsPhysician;
```

NEW QUERY REPORT

```
select    count(case when IsPatient = 'Y' then 1 end) as IsPatient,
          count(case when IsPhysician = 'Y' then 1 end) as IsPhysician

from Person_RPatient
group by Ispatient, IsPhysician;
```

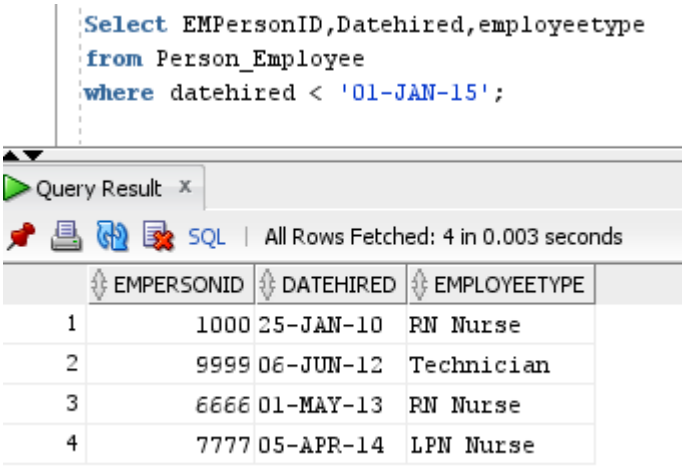


54) Report objectives: To provide 10% salary increase to employees
Persona this report is targeting: Human Resource
Decision this report will support: Based on the report HR can make decision to provide 10% salary increase to the employee who got hired before January 1st 2015.
Data need to create the report: We need Employee ID, Date hired and Employee type

```
Select
EMPersonID,Datehired,employee
type from Employee
where datehired < '01-JAN-15';
```

NEW QUERY REPORT

```
Select EMPersonID,Datehired,employeetype
from Person_Employee
where datehired < '01-JAN-15';
```



```
Select EMPersonID,Datehired,employeetype
from Person_Employee
where datehired < '01-JAN-15';
```

Query Result

SQL | All Rows Fetched: 4 in 0.003 seconds

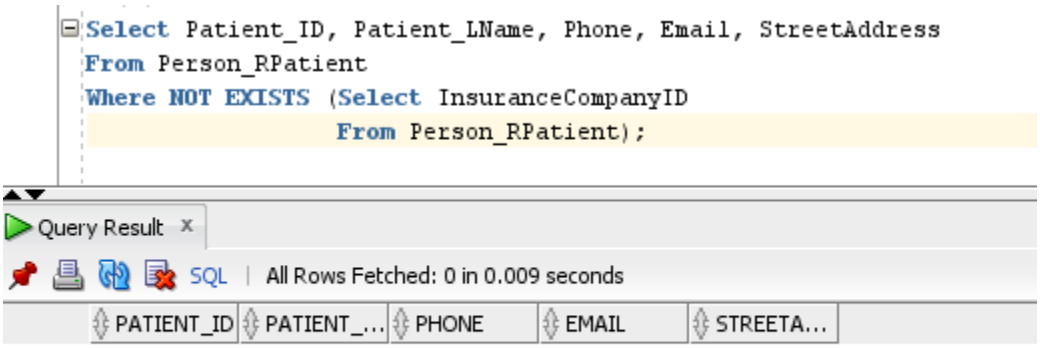
	EMPERSONID	DATEHIRED	EMPLOYEETYPE
1	1000	25-JAN-10	RN Nurse
2	9999	06-JUN-12	Technician
3	6666	01-MAY-13	RN Nurse
4	7777	05-APR-14	LPN Nurse

4) Report objectives: Patient who does not have medical insurance, send them full coverage insurance offer.
Persona this report is targeting: Marketing
Decision this report will support: Based on this report marketing employee can make decision who to contact and send them medical insurance offer.
Data need to create the report: The Patient ID, Name and Contact Number, Email and Street address.

```
Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email,
Person.StreetAddress
From Patient
INNER JOIN Person
ON PersonID = PAPersonID
Where NOT EXISTS (Select Patient_Insurance_Coverage.InsuranceCompanyID
                  From Patient_Insurance_Coverage
                  Where Patient_Insurance_Coverage.PAPersonID = Patient.PAPersonID);
```

NEW QUERY REPORT:

```
Select Patient_ID, Patient_LName, Phone, Email, StreetAddress
From Person_RPatient
Where NOT EXISTS (Select InsuranceCompanyID
                  From Person_RPatient);
```

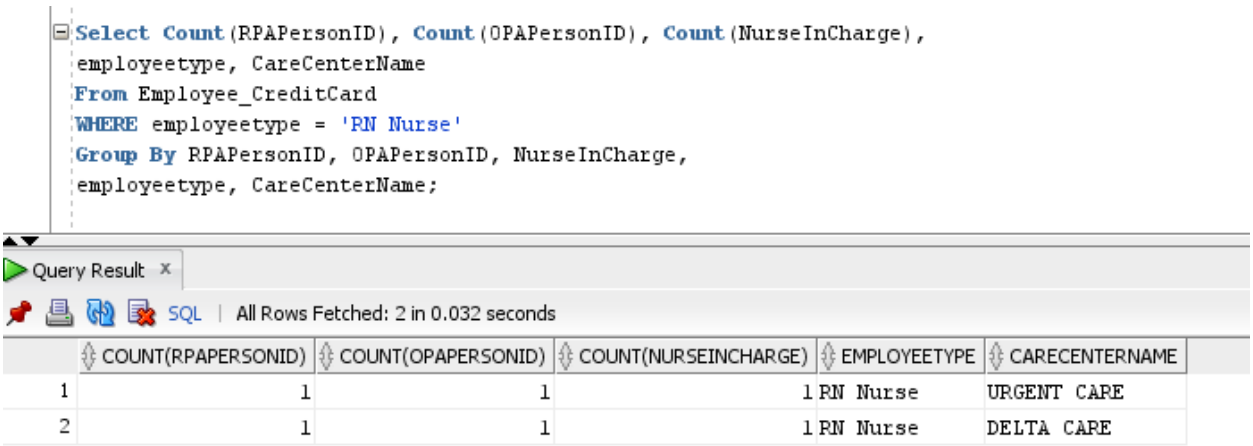


5) Report objectives: To hire more RN Nurse
Persona this report is targeting: Hiring Manager
Decision this report will support: This report will help the hiring manager to make decision if they need to hire more RN Nurse for the care center.
Data need to create the report: Total number of Resident Patient ID and Out patient ID also nurse in charge ID, Employee type and Care center name.

```
Select Count(Resident_Patient.RPAPersonID),
Count(Visit_Care_Center.OPAPersonID), Count(Care_Center.NurseInCharge),
employee.employeetype, Care_Center.CareCenterName
From Visit_Care_Center
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID = Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
INNER JOIN employee
ON employee.empersonid = care_center.nurseincharge
WHERE employee.employeetype = 'RN Nurse'
Group By Resident_Patient.RPAPersonID, Visit_Care_Center.OPAPersonID,
Care_Center.NurseInCharge, employee.employeetype,
Care_Center.CareCenterName;
```

NEW QUERY REPORT:

```
Select Count(RPAPersonID), Count(OPAPersonID), Count(NurseInCharge),
employeetype, CareCenterName
From Employee_CreditCard
WHERE employeetype = 'RN Nurse'
Group By RPAPersonID, OPAPersonID, NurseInCharge,
employeetype, CareCenterName;
```



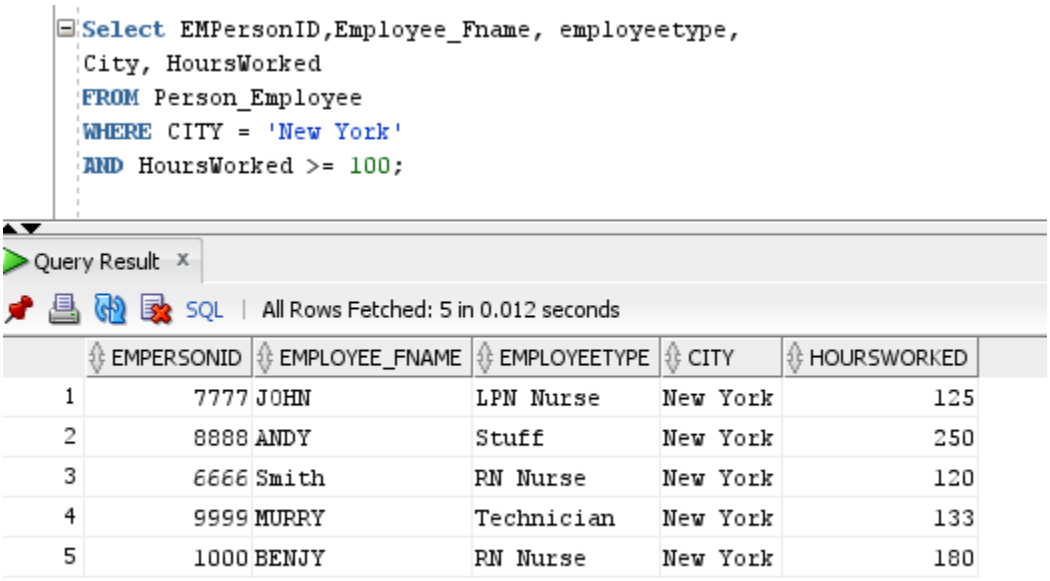
	COUNT(RPAPERSONID)	COUNT(OPAPERSONID)	COUNT(NURSEINCHARGE)	EMPLOYEETYPE	CARECENTERNAME
1	1	1	1	RN Nurse	URGENT CARE
2	1	1	1	RN Nurse	DELTA CARE

6) Report objectives: To give the employee monthly Metro Card
Persona this report is targeting: Employee Supervisor
Decision this report will support: This report will help the supervisor to make the decision to give the monthly Metro Card to the employee who live in New York and worked more the 100 hours.
Data need to create the report: Employee ID, First Name, Employee Type, City and Working Hours

```
Select Employee.EMPersonID, person.first, employee.employeetype,
Person.City, Employee_Assigned_Care_Center.HoursWorked
FROM PERSON, EMPLOYEE, Employee_Assigned_Care_Center
WHERE PERSON.PersonID = employee.empersonid
AND employee.EMpersonid = Employee_Assigned_Care_Center.EMPersonID
AND Person.CITY = 'New York'
AND Employee_Assigned_Care_Center.HoursWorked >= 100;
```

NEW QUERY REPORT:

```
Select EMPersonID,Employee_Fname, employeetype,
City, HoursWorked
FROM Person_Employee
WHERE CITY = 'New York'
AND HoursWorked >= 100;
```



```
Select EMPersonID,Employee_Fname, employeetype,
City, HoursWorked
FROM Person_Employee
WHERE CITY = 'New York'
AND HoursWorked >= 100;
```

Query Result

SQL | All Rows Fetched: 5 in 0.012 seconds

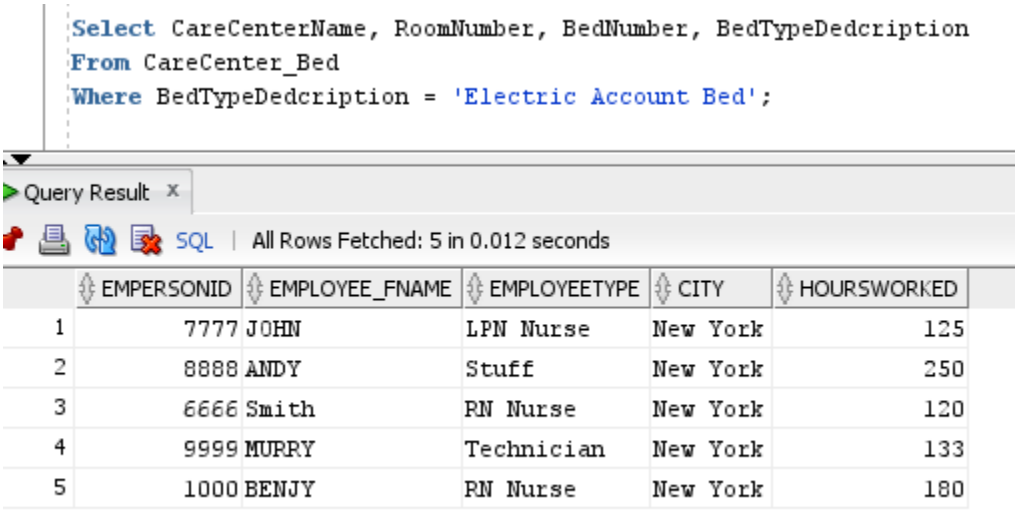
	EMPERSONID	EMPLOYEE_FNAME	EMPLOYEETYPE	CITY	HOURSWORKED
1	7777	JOHN	LPN Nurse	New York	125
2	8888	ANDY	Stuff	New York	250
3	6666	Smith	RN Nurse	New York	120
4	9999	MURRY	Technician	New York	133
5	1000	BENJY	RN Nurse	New York	180

7) Report objectives: Make schedule for the technician to inspect all the Electric Automated Bed
Persona this report is targeting: Schedule Manager
Decision this report will support: This report will help the Schedule Manager to see all the information about the location of electronic bed and send electrician for monthly inspection.
Data need to create the report: Care Center Name, Room Number, Bed Number, Bed type description.

```
Select Care_Center.CareCenterName, Bed.RoomNumber, BED.BedNumber,
Bed_Type.BedTypeDedcription
From Care_Center
INNER JOIN BED
ON Care_Center.CareCenterID = Bed.CareCenterID
INNER JOIN Bed_Type
ON Bed.BedTypeID = Bed_Type.BedTypeID
Where BedTypeDedcription = 'Electric Account Bed';
```

NEW QUERY REPORT:

```
Select CareCenterName, RoomNumber, BedNumber, BedTypeDedcription
From CareCenter_Bed
Where BedTypeDedcription = 'Electric Account Bed';
```



```
Select CareCenterName, RoomNumber, BedNumber, BedTypeDedcription
From CareCenter_Bed
Where BedTypeDedcription = 'Electric Account Bed';
```

Query Result

SQL | All Rows Fetched: 5 in 0.012 seconds

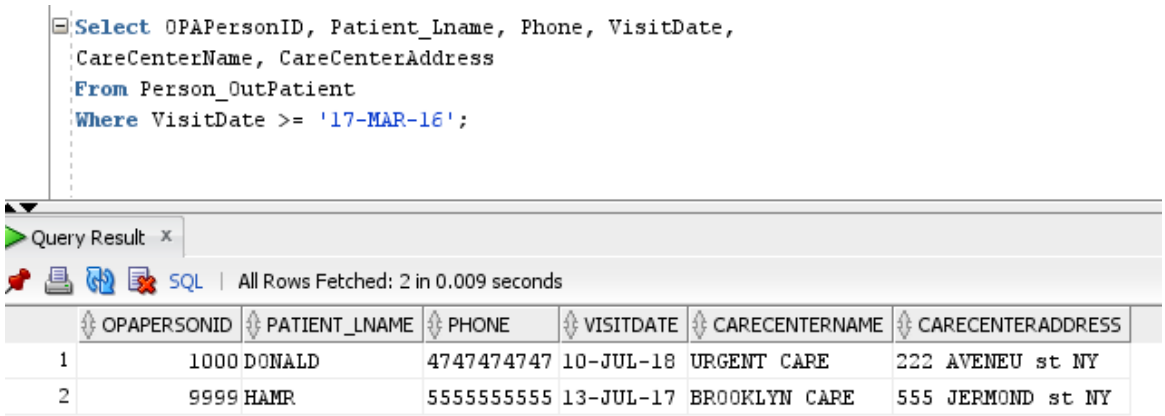
	EMPERSONID	EMPLOYEE_FNAME	EMPLOYEETYPE	CITY	HOURSWORKED
1	7777	JOHN	LPN Nurse	New York	125
2	8888	ANDY	Stuff	New York	250
3	6666	Smith	RN Nurse	New York	120
4	9999	MURRY	Technician	New York	133
5	1000	BENJY	RN Nurse	New York	180

8) Report objectives: Follow up call and reschedule out patient
Persona this report is targeting: Front-desk assistant
Decision this report will support: Based on this report front-desk assistant can see who are the patient are visited any given date and able to call them for follow up or reschedule.
Data need to create the report: Our patient ID, Patient Last name, Phone number, Visit Date, Care center name, Care Center address

```
Select Out_patient.OPAPersonID, Person.Last, Person.Phone,
Visit.VisitDate,
Care_Center.CareCenterName, Care_Center.CareCenterAddress
From Person
INNER JOIN Patient
ON Person.PersonID = Patient.PaPersonID
INNER JOIN Out_Patient
ON Patient.PaPersonID = Out_patient.OPAPersonID
INNER JOIN Visit
ON Out_patient.OPAPersonID = Visit.OPAPersonID
INNER JOIN Visit_Care_Center
ON Visit.OPAPersonID = Visit_Care_Center.OPAPersonID
INNER JOIN Care_Center
ON Visit_Care_Center.CareCenterID =
Care_Center.CareCenterID Where VisitDate >= '17-
MAR-16';
```

NEW QUERY REPORT:

```
Select OPAPersonID, Patient_Lname, Phone, VisitDate,
CareCenterName, CareCenterAddress
From Person_OutPatient
Where VisitDate >= '17-MAR-16';
```



```
Select OPAPersonID, Patient_Lname, Phone, VisitDate,
CareCenterName, CareCenterAddress
From Person_OutPatient
Where VisitDate >= '17-MAR-16';
```

Query Result

SQL | All Rows Fetched: 2 in 0.009 seconds

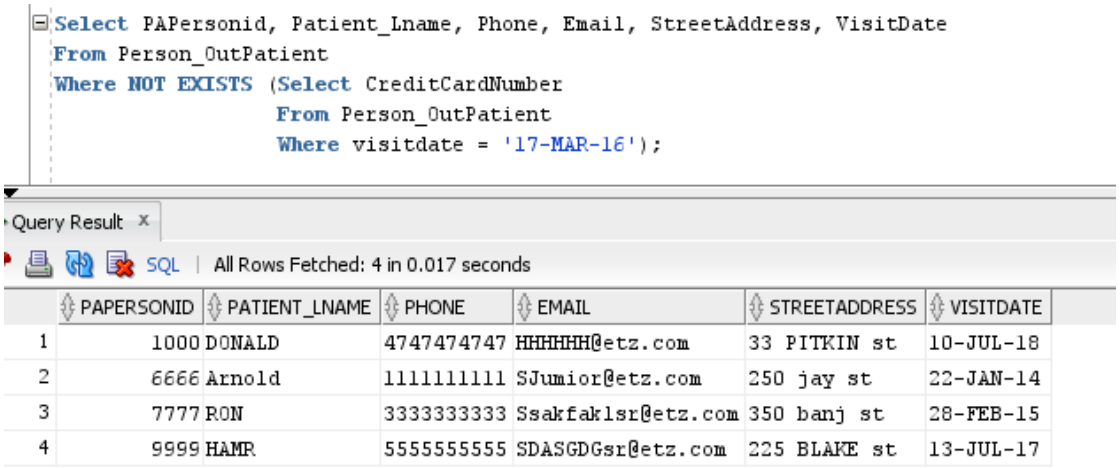
	OPAPERSONID	PATIENT_LNAME	PHONE	VISITDATE	CARECENTERNAME	CARECENTERADDRESS
1	1000	DONALD	4747474747	10-JUL-18	URGENT CARE	222 AVENEU st NY
2	9999	HAMR	5555555555	13-JUL-17	BROOKLYN CARE	555 JERMOND st NY

9) Report objectives: Collect Credit Card Information
Persona this report is targeting: Accounts
Decision this report will support: This report will target to those patients who visited particular date and did not have a credit card so hospital Account personal can make decision who to contact.
Data need to create the report: Patient ID, Patient Last name, Phone number, Email, Street Address and Visit Date.

```
Select Patient.PAPersonid, Person.Last, Person.Phone, Person.Email,
Person.StreetAddress, Visit.VisitDate
From Patient
INNER JOIN Person
ON Person.PersonID = Patient.PAPersonID
INNER JOIN Out_Patient
ON Patient.PAPersonID = Out_Patient.OPAPersonID
INNER JOIN Visit
ON Out_Patient.OPAPersonID = Visit.OPAPersonID
Where NOT EXISTS (Select Patient_Credit_Card.CreditCardNumber
                  From Patient_Credit_Card
                  Where Patient_Credit_Card.PaPersonID = Patient.PaPersonID)
And visit.visitdate = '17-MAR-16';
```

NEW QUERY REPORT:

```
Select PAPersonid, Patient_Lname, Phone, Email, StreetAddress, VisitDate
From Person_OutPatient
Where NOT EXISTS (Select CreditCardNumber
                  From Person_OutPatient
                  Where visitdate = '17-MAR-16');
```



```
Select PAPersonid, Patient_Lname, Phone, Email, StreetAddress, VisitDate
From Person_OutPatient
Where NOT EXISTS (Select CreditCardNumber
                  From Person_OutPatient
                  Where visitdate = '17-MAR-16');
```

Query Result

SQL | All Rows Fetched: 4 in 0.017 seconds

	PAPERSONID	PATIENT_LNAME	PHONE	EMAIL	STREETADDRESS	VISITDATE
1	1000	DONALD	4747474747	HHHHHH@etz.com	33 PITKIN st	10-JUL-18
2	6666	Arnold	1111111111	SJumior@etz.com	250 jay st	22-JAN-14
3	7777	RON	3333333333	Ssakfaklsr@etz.com	350 banj st	28-FEB-15
4	9999	HAMR	5555555555	SDASGDGsr@etz.com	225 BLAKE st	13-JUL-17

10) Report objectives: Hire new volunteer if needed

Persona this report is targeting: Volunteer Supervisor

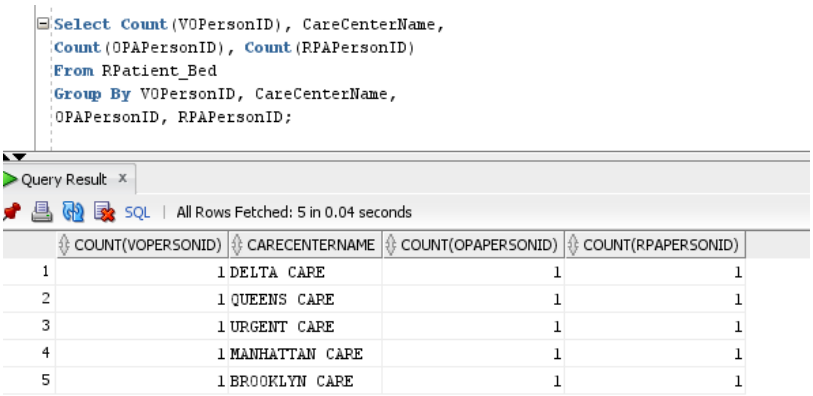
Decision this report will support: Based on total number of outpatient and resident patient Volunteer Supervisor will make decision if hospital need more volunteer or not.

Data need to create the report: Volunteer ID, Care Center name, Out Patient and resident Patient

```
Select Count(Volunteer_Assigned_Care_Center.VOPersonID),
Care_Center.CareCenterName,
Count(Visit_Care_Center.OPAPersonID),
Count(Resident_Patient.RPAPersonID)
From Volunteer_Assigned_Care_Center
INNER JOIN Care_Center
ON Volunteer_Assigned_Care_Center.CareCenterID =
Care_Center.CareCenterID
INNER JOIN Visit_Care_Center
ON Care_Center.CareCenterID = Visit_Care_Center.CareCenterID
INNER JOIN BED
ON Care_Center.CareCenterID = BED.CareCenterID
INNER JOIN Resident_Patient
ON Resident_Patient.BedNumber = Bed.BedNumber
Group By Volunteer_Assigned_Care_Center.VOPersonID,
Care_Center.CareCenterName,  Visit_Care_Center.OPAPersonID,
Resident_Patient.RPAPersonID;
```

NEW QUERY REPORT:

```
 Select Count(VOPersonID), CareCenterName,
Count(OPAPersonID), Count(RPAPersonID)
From RPatient_Bed
Group By VOPersonID, CareCenterName,
OPAPersonID, RPAPersonID;
```



	COUNT(VOPERSONID)	CARECENTERNAME	COUNT(OPAPERSONID)	COUNT(RPAPERSONID)
1	1	DELTA CARE	1	1
2	1	QUEENS CARE	1	1
3	1	URGENT CARE	1	1
4	1	MANHATTAN CARE	1	1
5	1	BROOKLYN CARE	1	1

Conclusion

This section was the upgrade version 3.0 of the Hospital Management System Database. Project management methodology used the database application development lifecycle. We were following all the phase to do this upgrade, start with planning, analysis, design, development and maintenance phase to upgrade this version by creating business reports and improve the performance of full working application. We also created Index for the business reports. Index will increase the performance for runtime of business reports. We also created View for the query reports. Some of the Views covered many query reports. In order to increase the performance of Hospital Management Systems Database we had to create the Index and View in this upgraded version.