## **Project Report on**

## **Database of Electronic Medical Record**

**Submitted By**

Nidhi Vijayvargiya

Student Id 10725904

Contents

[Introduction 3](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444419)

[Design 3](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444420)

[Table Design 3](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444421)

[Assumptions 4](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444422)

[Entity Relationship Modeling 5](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444423)

[Implementation 6](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444425)

[Creating The tables 6](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444426)

[Inserting Data 9](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444427)

[Queries 12](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444428)

[Views and Reports 13](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444429)

[Sub-Queries 14](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444430)

Exists ………………………………………………………………………………………………………………………………………………..15

Index………………………………………………………………………………………………………………………………………………….15

[Built-In-Function 16](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444431)

Join Query…………………………………………………………………………………………………………………………………………17

PL/SQL [18](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444432)

[Conclusion 21](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444433)

[Achieved 21](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444434)

[Shortcomings and Room for Improvement 21](file:///C:\Users\nidhi\Desktop\dbms%20project\report%20(1).docx#_Toc491444435)

# 

# Introduction

In the current scenario, small hospital and healthcare centers have multiple challenges, and their medical records are the most difficult to handle. As we know in the healthcare sector, it is very important to have a prior record (name, condition, doctor's name) of patients for better treatment. For years, our facility has been sticking to the old system of documenting and monitoring. Until now, though, it has been acceptable with newer technology and changing health policies and regulations, we are compelled to make a decision. Do we change our way of doing things to satisfy the expectations and needs of our patients and lawmakers, or do we crack under pressure and push our patients and staff elsewhere?

I am presenting a fully implemented open electronic medical record, which will improve the ability of patients and staff to report and trend in ways that I would not be able to do with paper maps. Information concerning the patient, the orders, the findings of their lab, as well as the doctors and staff. Working with the database class will provide a company with a large amount of insight. I hope that by helping you to see the capabilities and functions of a database, it can help you do what I think is best decision.

# Design

## Table Design

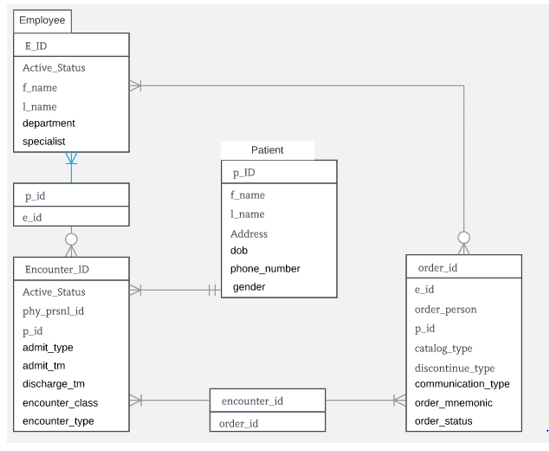
To show you the design, functionality, and importance of an open EMR database, I have collected various data attributes that I wanted to use in the database. This database consists of data used to document simple patient stay detail. However, I would like to mention that this does not contain any piece of data that can be made available, but rather those that I chose to illustrate and include. There are four tables in this open EMR database: Customer, Employee, Encounter, and Order Details. . In each table, there is an underlined attribute. This attribute is primary key, which will uniquely identify each row in the respective table. In this situation, all four tables have a primary key that also serves as a surrogate key. This surrogate key will increase automatically with each new record in our database and has been selected for this reason. This means that the attribute is a foreign key which serves as the primary key for another table. Using foreign keys, I can connect tables together through the shared attribute.

## Assumptions

## When creating a database, assumptions need to be made on how the tables can connect to each other. The PATIENT table includes demographic statistics on patients attending our care facilities. The EMPLOYEE table is like the EMPLOYEE table, but it includes information about all the employees in our organization. This would include doctors, nurses, residents, surgeons, clerks, IT, etc. Also the ENCOUNTER table records details related to the patient's stay, the date of diagnosis, the date of departure, the sort of encounter, the explanation for the visit, and several more will be found in this table. Finally, our ORDER DETAILS table includes information about all orders placed during a patient ENCOUNTER.

## Entity Relationship Modeling

I started collecting information and generated our database design via an entity-relationship diagram once the table information and assumptions were complete. In the design of the database, the E-R diagram plays a vital role since it relates information between tables. Finally, I may point out that it was appropriate to construct two intersection tables, which store the two tables' primary keys. The Many: Many (M:N) relationships between EMPLOYEES to ENCOUNTER and ENCOUNTER to ORDER DETAILS are the product of these tables. Once the E-R Diagram was finished, before constructing the database itself, I started debating what the table construction should be. This involves the correct form of data, main data, null status, and remarks.

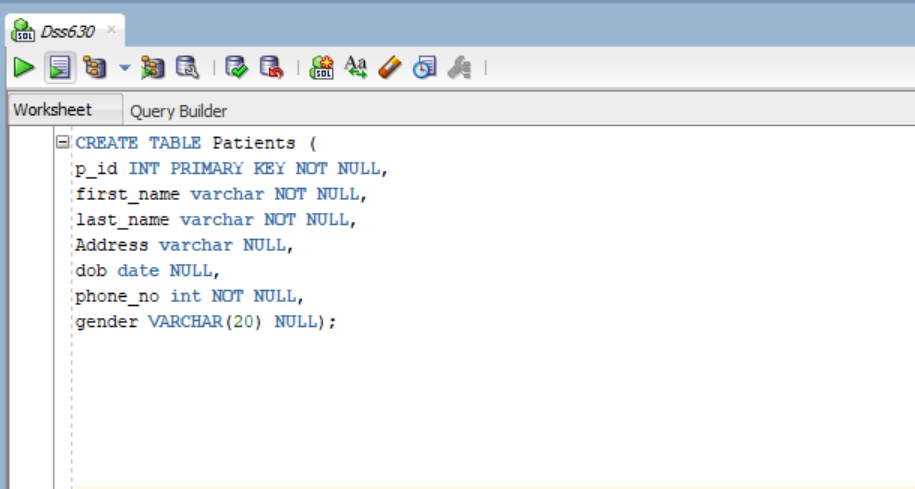


# Implementation

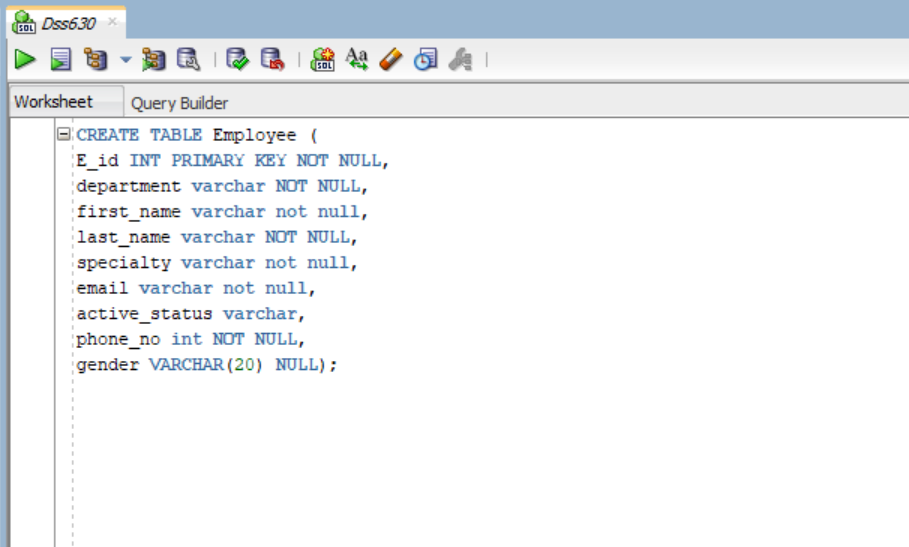
## Creating the tables:

As you know, four tables have been created: PATIENTS, EMPLOYEE, ENCOUNTER and ORDER DETAILS. For two reasons, PATIENTS and EMPLOYEE tables have had to be created first: they have no foreign keys and in other tables their primary key is referenced. ENCOUNTER was the next table created, followed by ORDER DETAILS. There is a foreign key restriction in the ENCOUNTER table that uses the primary key from the PERSON and PERSONNEL tables, so the third table had to be created. If someone has tried to insert a record into ENCOUNTER and there is no primary key in PATIENTS or EMPLOYEE, the new record will not be inserted to retain the integrity of the records. Lastly, the ORDER DETAILS table was introduced as it has a foreign key constraint using the primary key in all three previous tables.

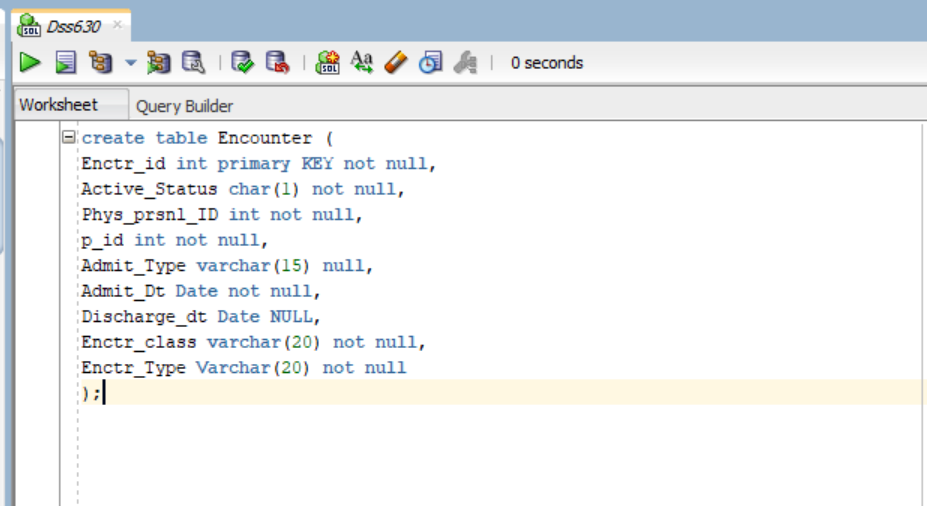
**PATIENTS**



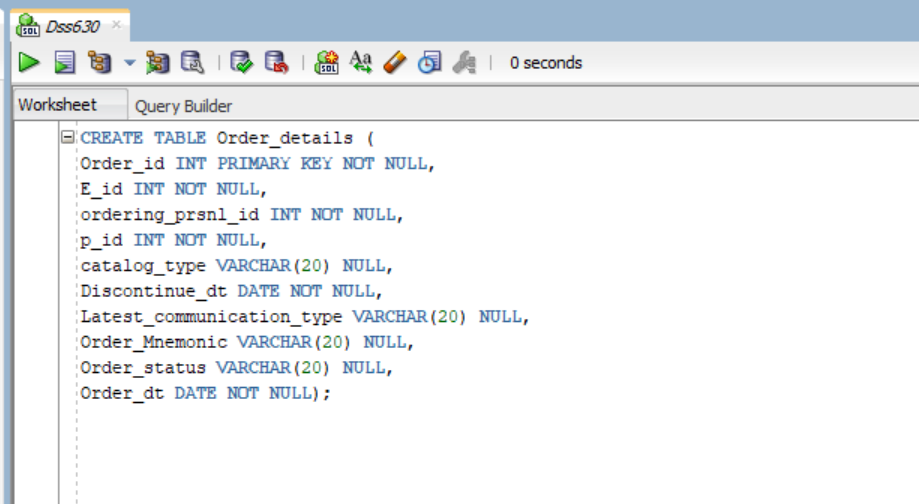
**EMPLOYEE**



**ENCOUNTER**



**ORDER\_DETAILS**

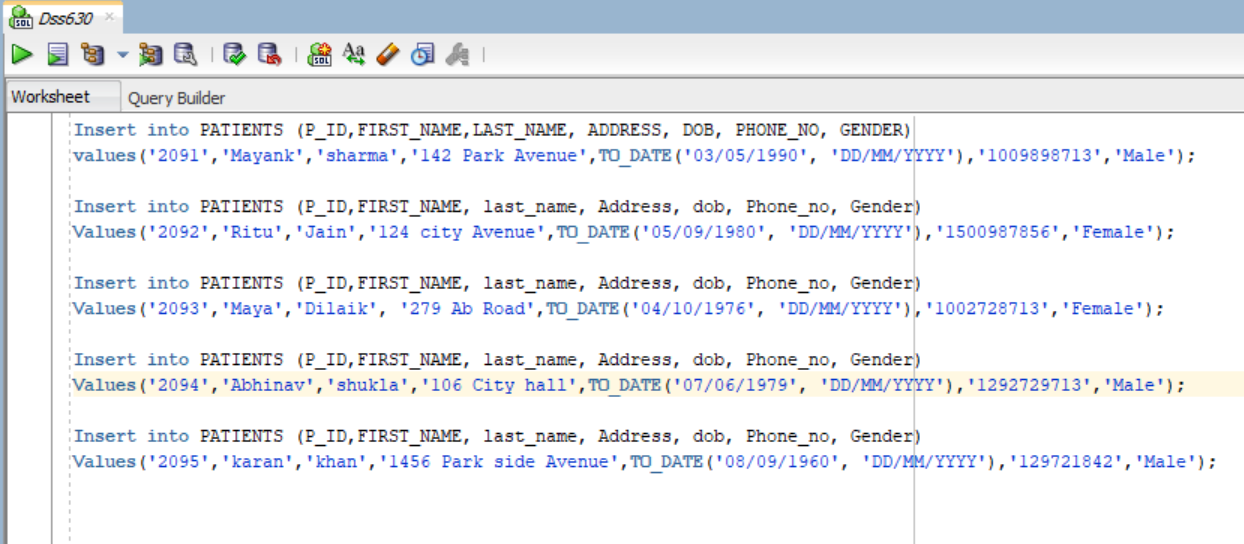


## 

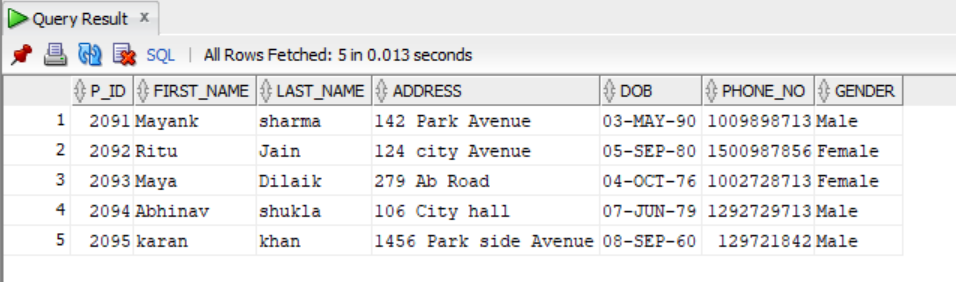
## Inserting Data

I could finally insert the data after all the tables were established, but with the less constraints. Because of the data integrity restrictions, I had to follow the same order in which the tables were originally generated to insert the data. First, test patients were developed, then fake employees, experiences, and details of their orders. At our facility, the values inserted in each of these tables might exist.

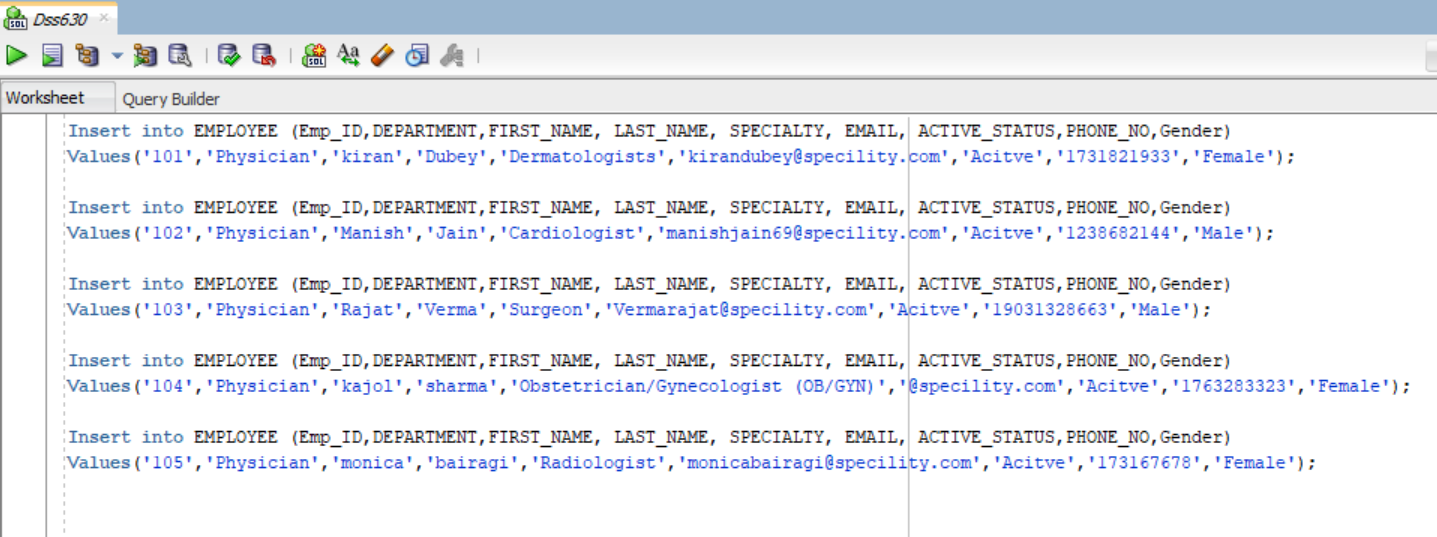
**Table1: PATIENTS**

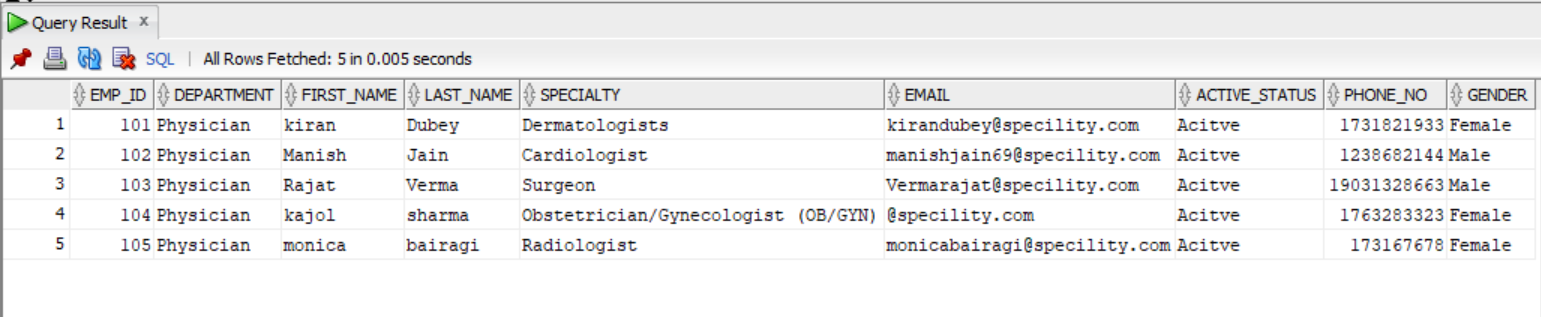




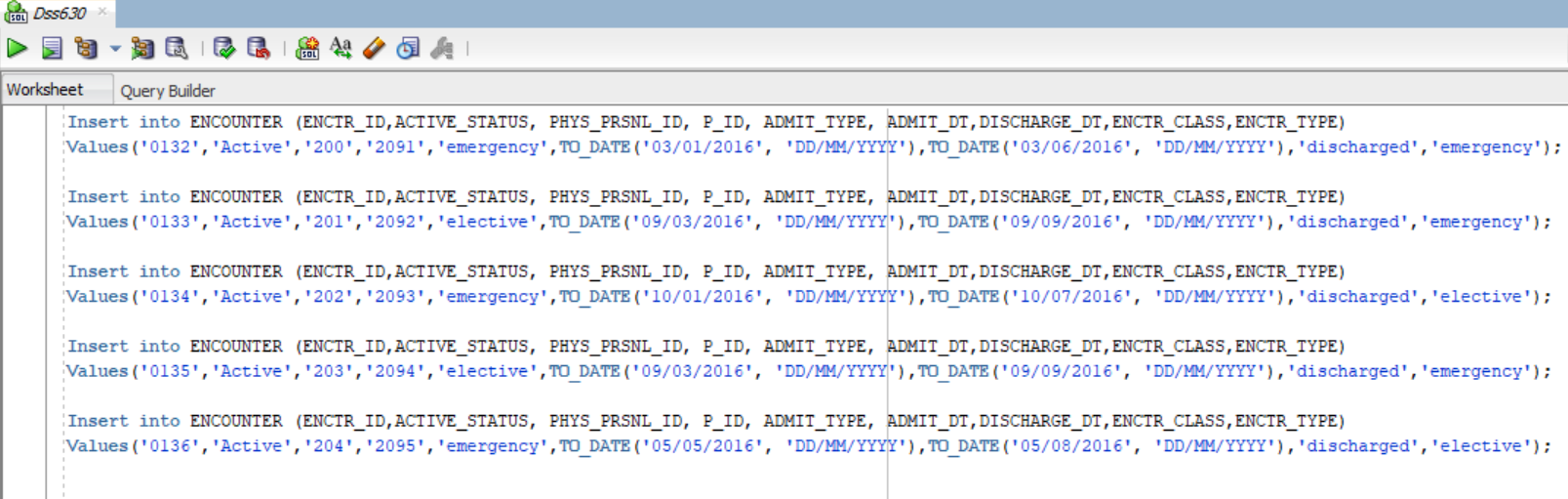


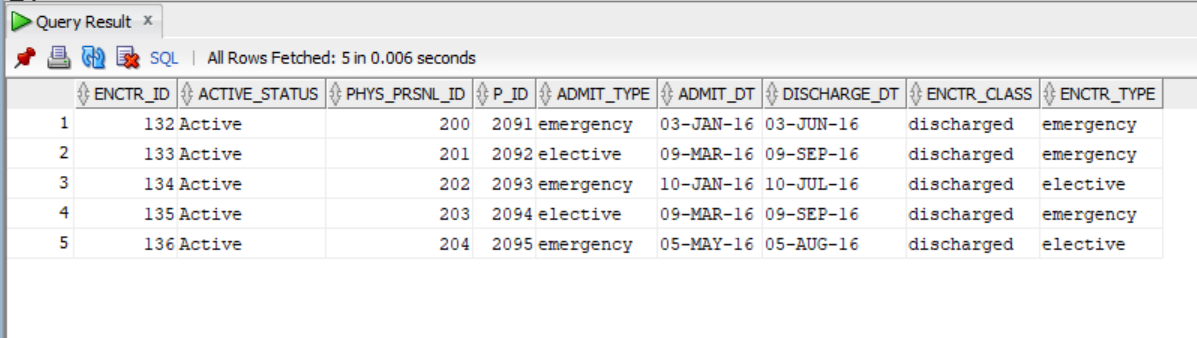
**Table 2: EMPLOYEE**



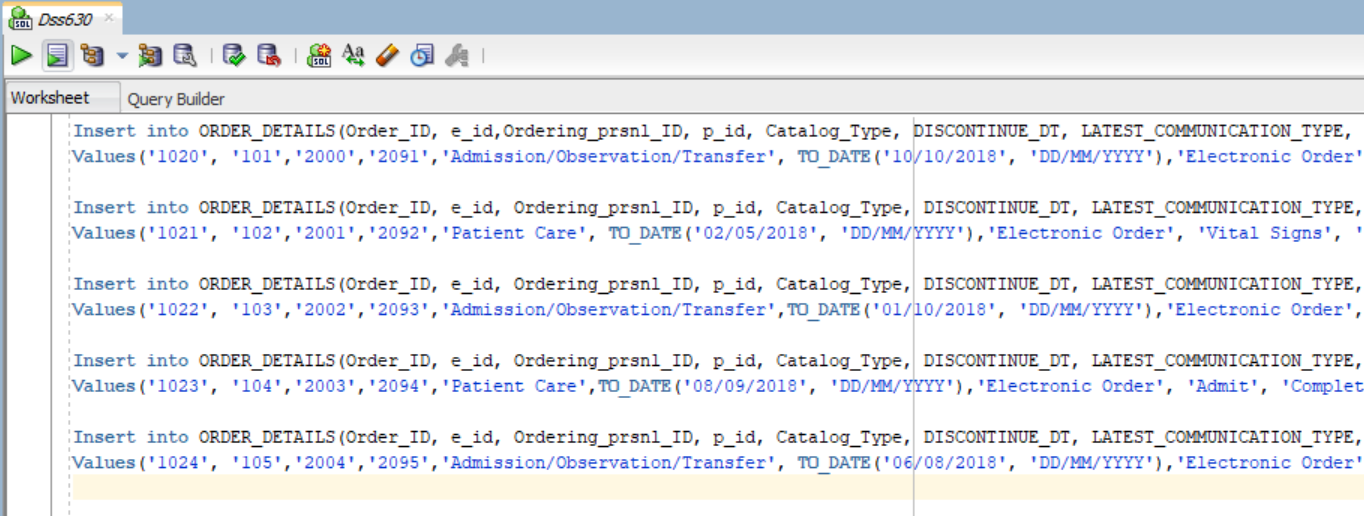


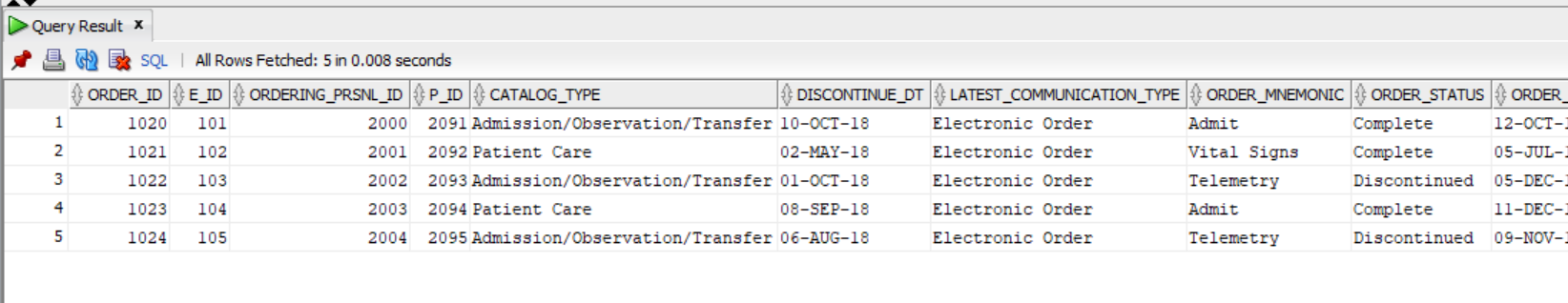
**Table 3: ENCOUNTER**





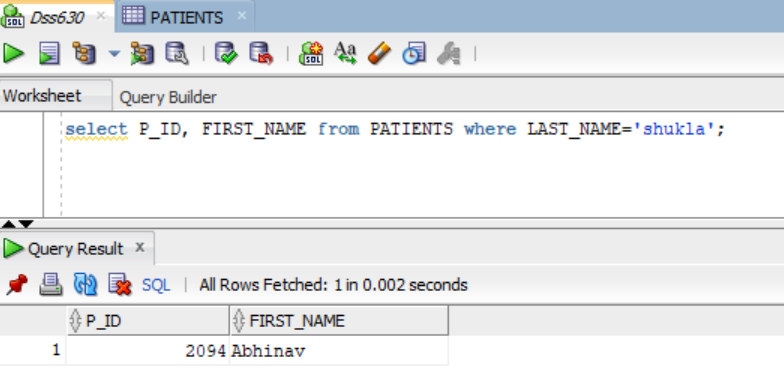
**Table 4: ORDER DETAILS**



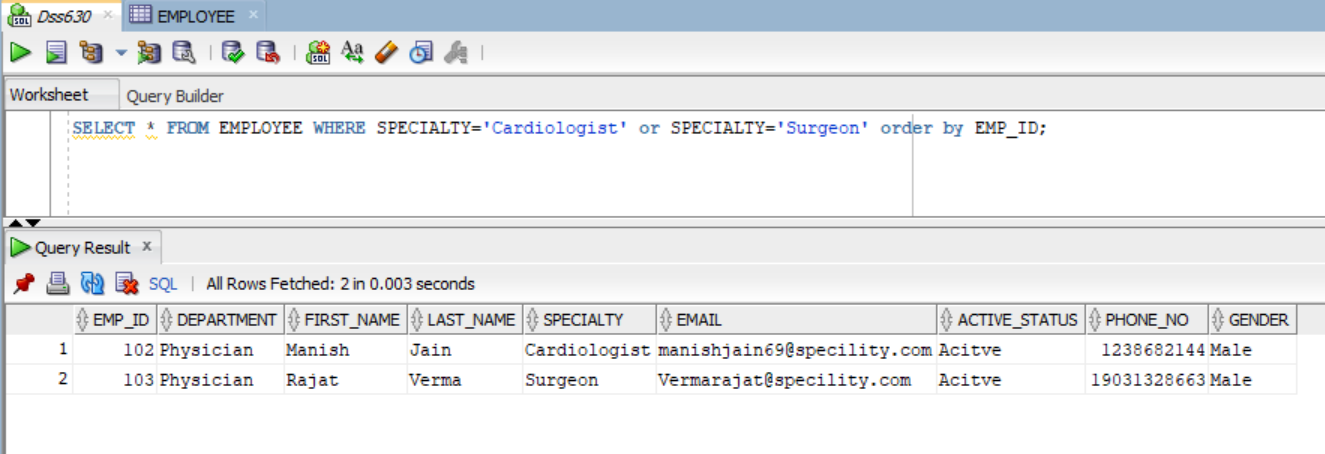


## Queries

I finally had the opportunity to explore the database which i spent a lot of time developing after "dummy data" was applied to the tables. To get the information of the patient, i.e. the first name and patient ID by knowing his last name, I began with a very simple question. In other words, how many patients with that last name do I have? The importance of this question is to track our facility's patient census. I might also take the question a step further with more data and get the data of individuals who are surgeons. In second issue, I want to find out how I have a surgeon in each department.



To find how many surgeons I have in the data I used order by

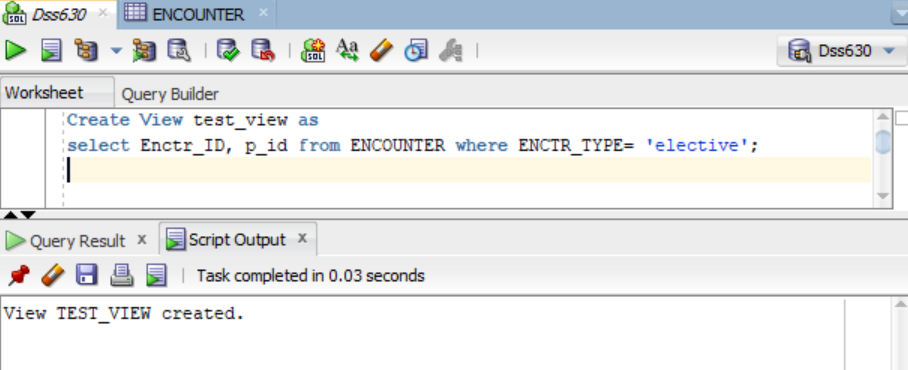


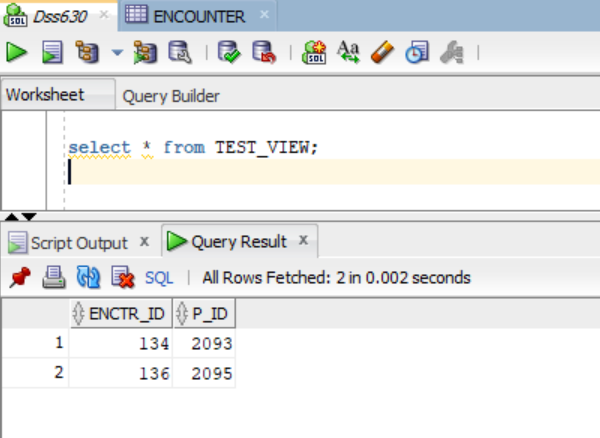
**Views and Reports**

For the end-users of a database, views may become extremely helpful. In order to eliminate the need to type long queries every time it is used, views appear to store frequently used queries. They can also limit the user-seen data and automatically update when tables are updated. Views tend to be large and generic queries that are processed, whereas reports are specific.

**VIEW: 1**

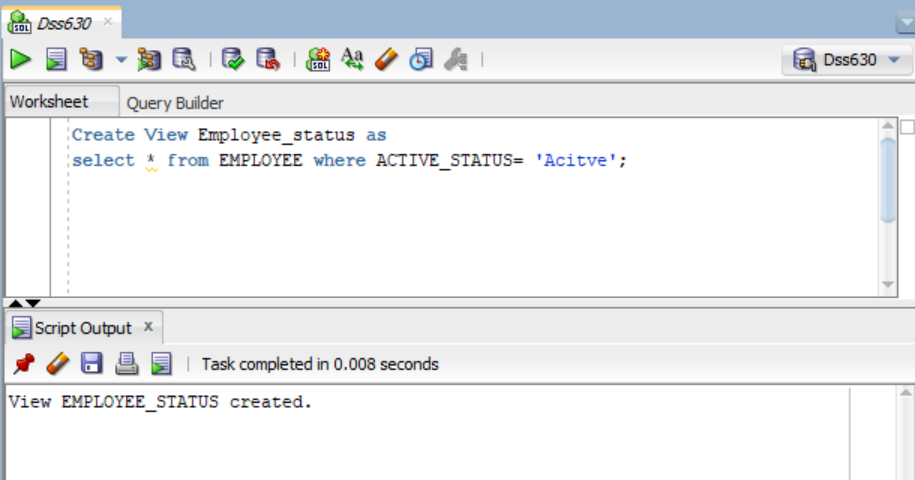
This view created shows Encounter ID and patient ID.

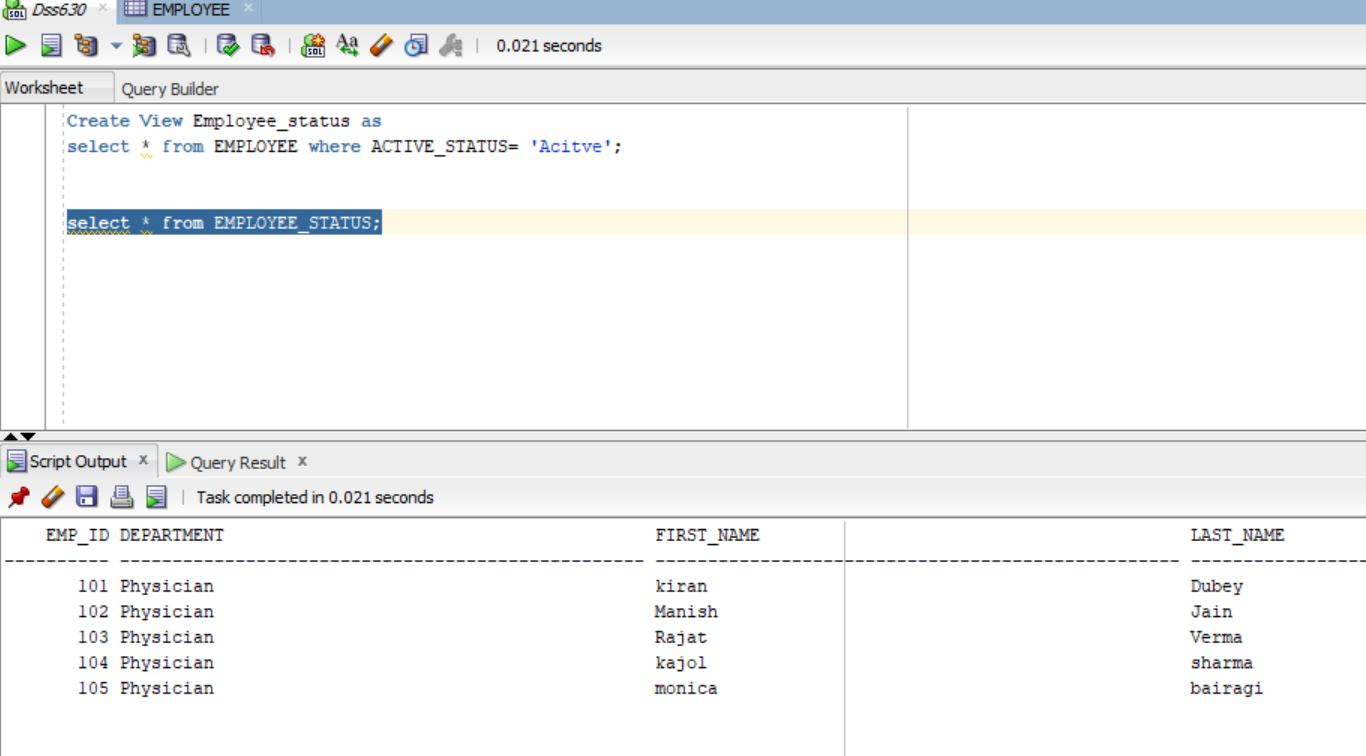




**View 2**

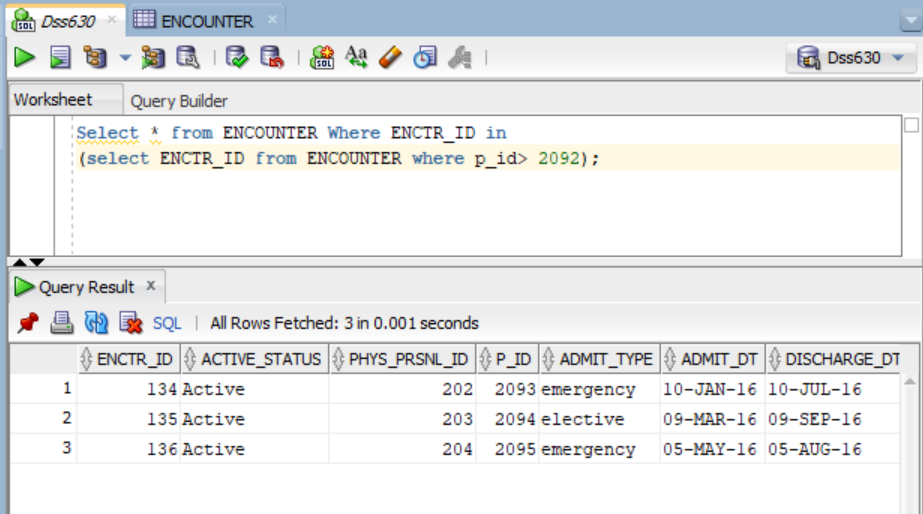
It shows all the data from the Employee Table where the Active\_status is ‘ACTIVE’



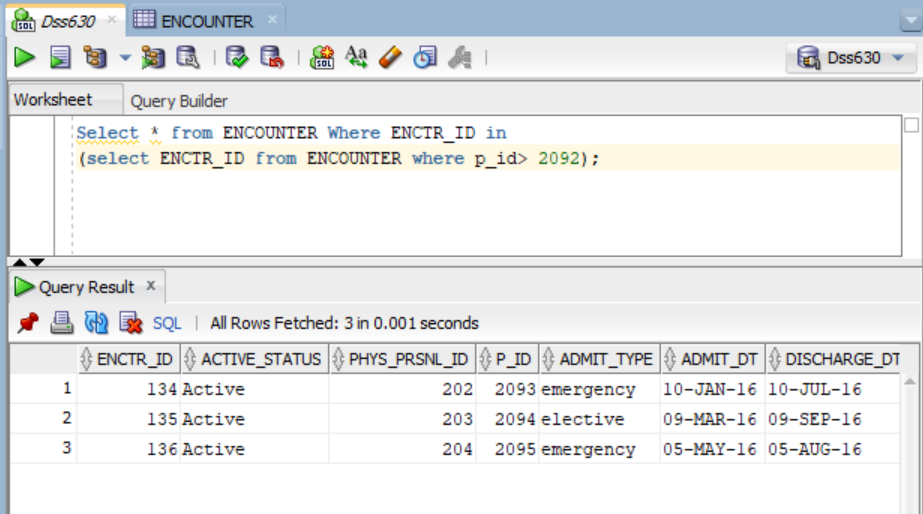


**SUB- QUERY**

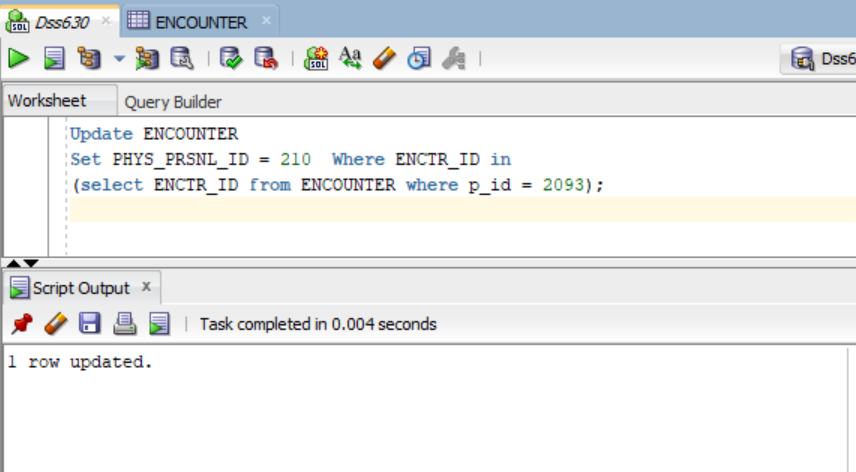
Select statement

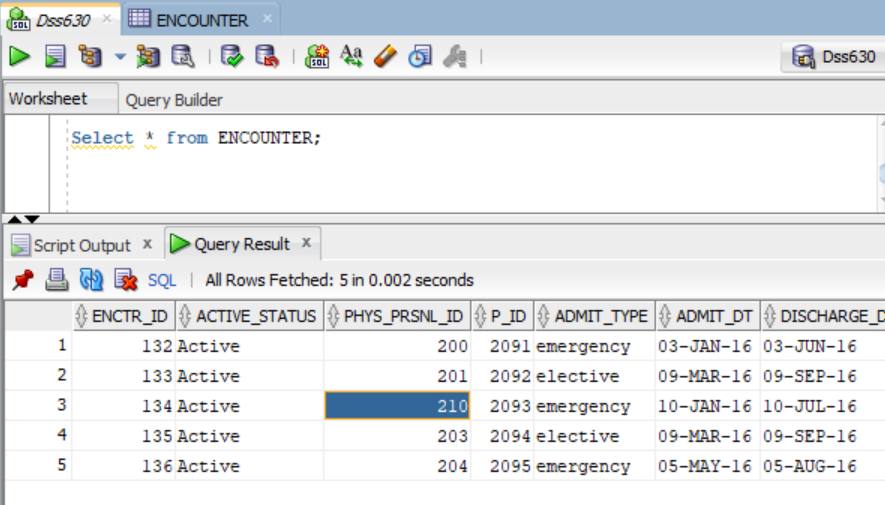


**Update query** This is the data before I ran the update query



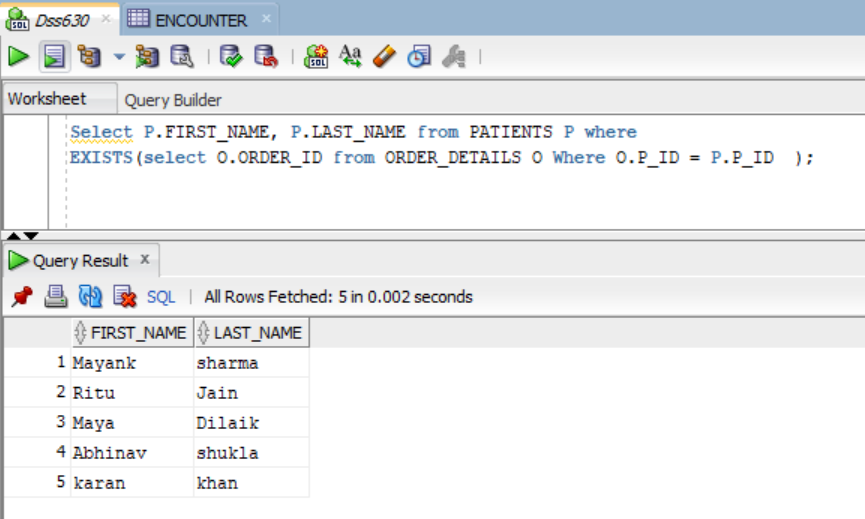
**After Update Query run:**



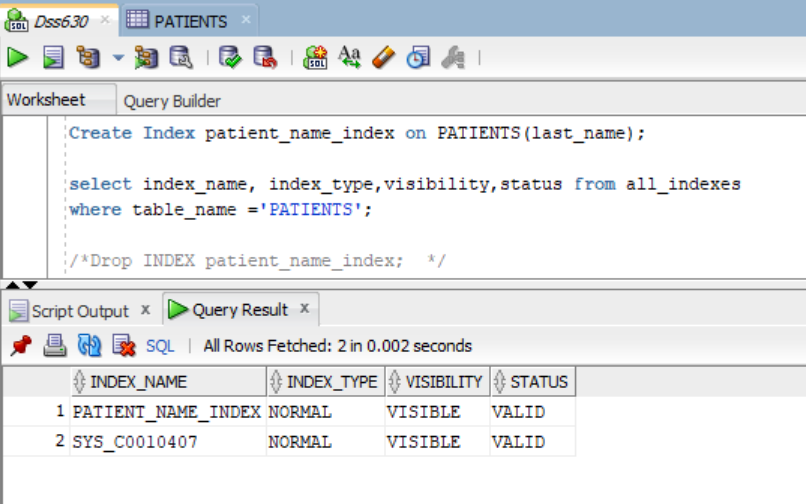


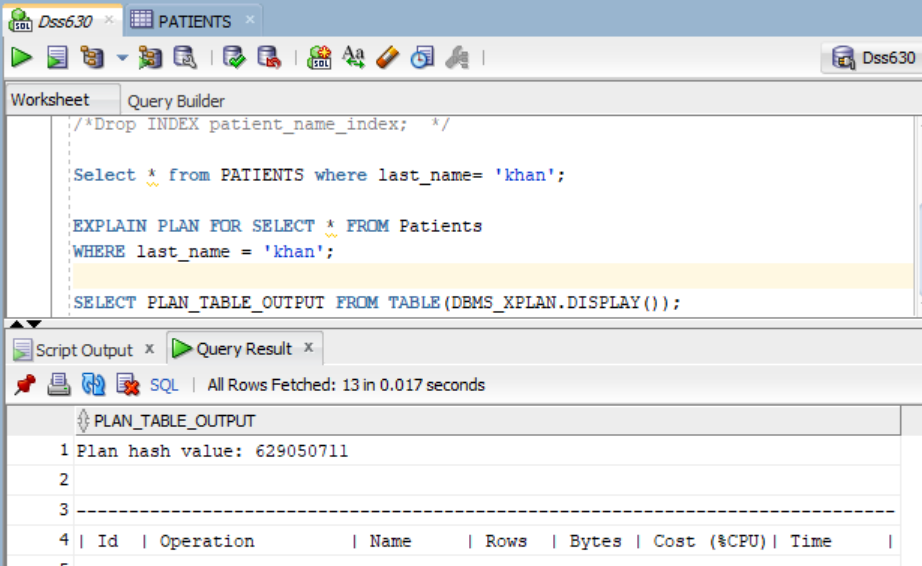
**EXISTS Query:**

Below I am using the EXISTS operator to find name of all the patient who have the order in ORDER\_Details table here I have inserted 5 sample records in each table and all 5 patients have their order present in Order table.



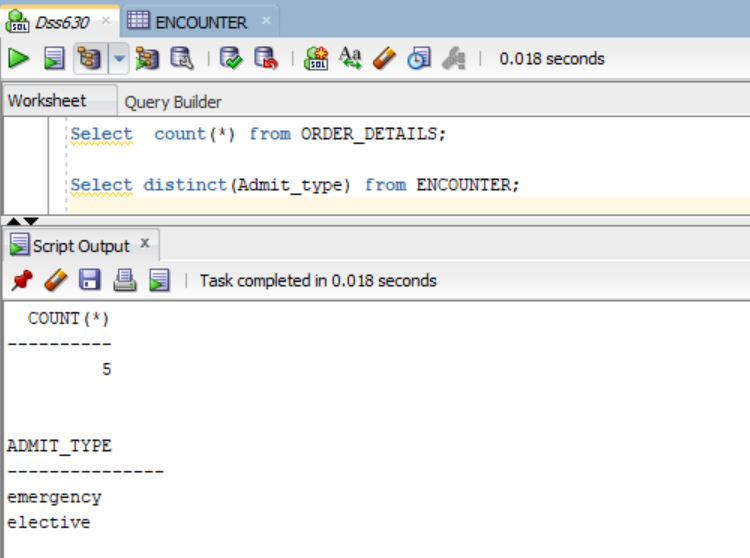
**Indexes** Creation of Index

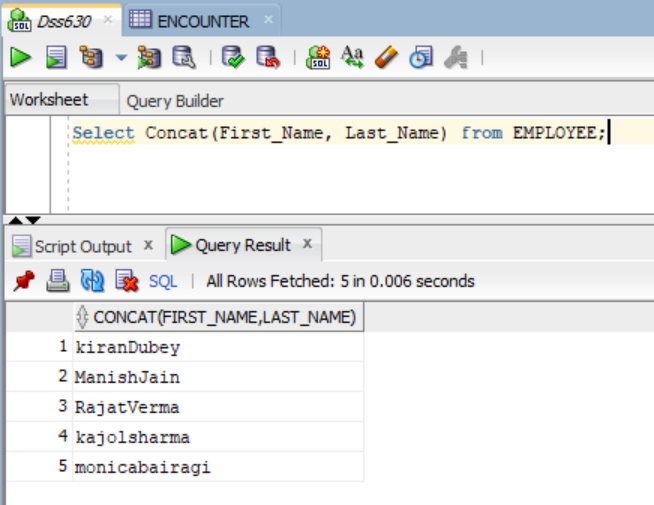


Below is the sql statement for checking Index working

**Built-In Functions**

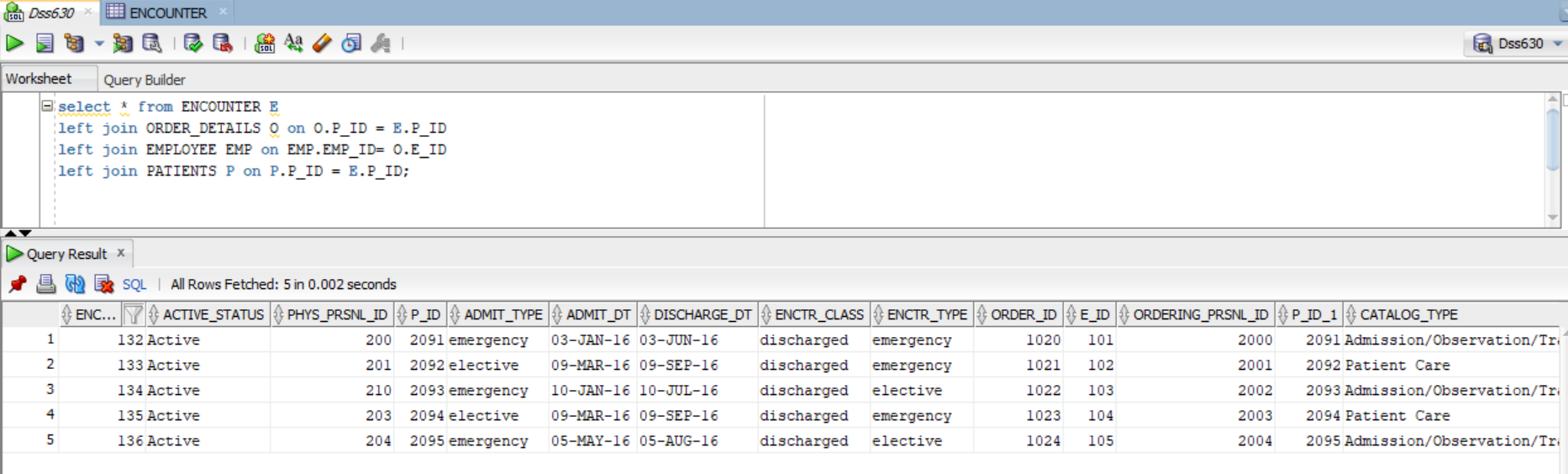
1. **Count and Distinct–**It gives us the number of rows in the table.



**Concat -** This concatenates the two columns i.e. merge two columns data(first name And Last name).

**JOIN Query**

Below is the join Query where I am joining Encounter table with Order details, Employee and Patients using Left Join on Encounter Table.

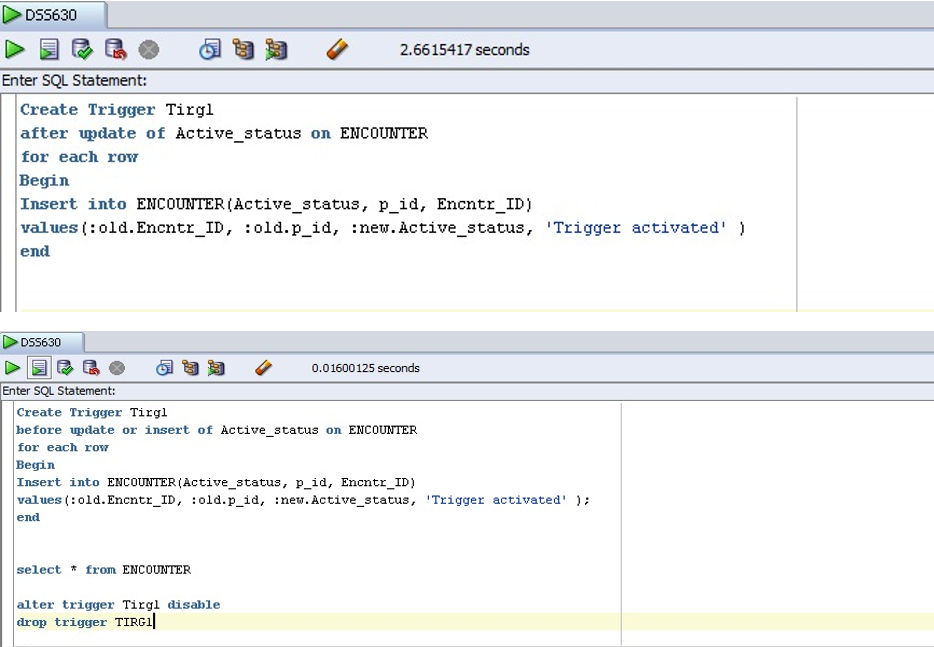


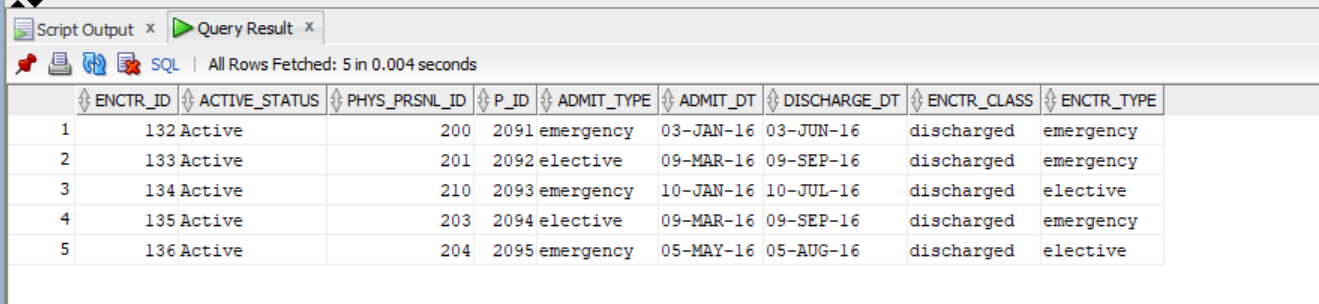
**PL/SQL:**

It is a combination of SQL, along with programming languages' procedural features. It was developed in the early 90's by Oracle Corporation to boost SQL capabilities. PL/SQL is, along with SQL itself and Java, one of three main programming languages embedded in the Oracle Database.

**Trigger:**

A trigger is a specific form of store procedure that runs, such as adding, deleting or updating data in response to certain behavior on the table. It is a database object that is bound to a table and is immediately executed. You can't invoke triggers directly. The only way to do this is to carry out the necessary action without the table to which they are allocated**.**





**Conclusion**

**Achieved**

I have learned a lot in this project and finishing assignments on schedule. I need to develop my database skills by building myself from the ground up, including features, queries,stored processes and views. Even with my small data set, I was still able to achieve all the goals I set when the project started.

**Shortcomings and Room for Improvement**

Ideally, the mock database should provide more information than is retained during a patient experience. Things such as accidents, radiology, anatomy, pharmacy, insurance, etc. would have been best tailored and more accurate to the information management system than our simplified mock database. Improving this database will require a tremendous amount of work to do in its present form. Many tables will have to be added and loaded with data, resulting in various relationships and restrictions that need to be considered between new and current tables. Another smart idea would have been to set default values or checks while constructing a table. This will further minimize the likelihood of typing errors by choosing only specified alternatives, such as the state or gender field in the PATIENTS chart. These weaknesses should though, be taken with a constructive note. By understanding the errors that could occur in the database, it allows us a chance to be more informed and less likely to commit the same errors in subsequent builds.

**References:**

Database Processing: Fundamentals, Design, and Implementation (14th Edition).

<https://www.oracletutorial.com/>

<https://www.w3schools.com/>