DBMS PROJECT

**TOURIST MANAGEMENT SYSTEM**

**DONE BY**

**K.KAUSHIK DATTA**

* DBMS PROJECT

Contents

[1. ABSTRACT : 4](#_Toc21985480)

[2. MAIN MODULES : 5](#_Toc21985481)

[3. OBJECTIVES OF THE PROJECT : 5](#_Toc21985482)

[4. REPRESENTATION : 5](#_Toc21985483)

[I. DESIGN MODELS : 5](#_Toc21985484)

[II. ENTITY RELATIONSHIP MODEL [ ER MODEL ] : 5](#_Toc21985485)

[III. UML DIAGRAMS : 6](#_Toc21985486)

[IV. CLASS DIAGRAM - 6](#_Toc21985487)

[USE CASE DIAGRAM - 7](#_Toc21985488)

[ACTIVITY DIAGRAM - 8](#_Toc21985489)

[V. SEQUENCE DIAGRAM - 8](#_Toc21985490)

[DATA FLOW DIAGRAMS : 9](#_Toc21985491)

[CONTEXT LEVEL 11](#_Toc21985492)

[LEVEL 1 DFD 11](#_Toc21985493)

[LEVEL 2 DFD 11](#_Toc21985494)

[TABLES : 11](#_Toc21985495)

[ADMIN TABLE - 11](#_Toc21985496)

[USERS TABLE - 12](#_Toc21985497)

[TRANSPORT TABLE - 12](#_Toc21985498)

[HOTEL TABLE - 13](#_Toc21985499)

[PLACES TABLE - 13](#_Toc21985500)

[ADVERTISEMENT TABLE - 13](#_Toc21985501)

[EVENT TABLE - 14](#_Toc21985502)

[MANAGER TABLE - 15](#_Toc21985503)

[USERS MULTI VALUE TABLE - 15](#_Toc21985504)

[TRANSPORT MULTI VALUE TABLE - 15](#_Toc21985505)

[TRIGGERS : 16](#_Toc21985506)

a. TRIGGERS ON INSERTION

[b. TRIGGERS ON DELETION : 17](#_Toc21985507)

[PROCEDURE : 18](#_Toc21985508)

FUNCTION

[CURSOR : 20](#_Toc21985509)

[20](#_Toc21985510)

a. SQL % FOUND

[b. SQL % NOT FOUND 21](#_Toc21985511)

c. SQL % ROWCOUNT

NORMALIZATION :

[1NF (First Normal Form) Rules 26](#_Toc21985512)

[2NF (Second Normal Form) Rules 27](#_Toc21985513)

[3NF (Third Normal Form) Rules 27](#_Toc21985514)

CONCLUSION :

**TRAVEL MANAGEMENT SYSTEM**

# ABSTRACT :

Tour Management system is a dynamic website for tourism business. It is a dynamic and responsive web design. It is also called “travel technology solution” for agencies & tour operation. Nearly everyone goes on a vacation. For this, ‘A Tourism management system’ would play a vital role in planning the perfect trip. The tourism management system allows the user of the system access all the details, such as location, events, etc.

The main purpose is to help tourism companies to manage customer , hotels etc. The system can also be used for both professional and business trips. The proposed system maintains a centralized repository to make necessary travel arrangements and to retrieve information easily.

***This project is designed for tourists who would like to plan their trip with the help of a travel agency.***

# 2. MAIN MODULES :

In this project, we have two modules :

1. Admin
2. User
3. ADMIN

* Log In with user name and password
* Add / Delete / Modify Routes
* Assign route - ID’s
* Fix prices
* Plan a perfect tour

1. USER

* Register and log in
* Sets a destination
* Find the appropriate routes
* Give details for the journey
* Confirms booking

# OBJECTIVES OF THE PROJECT :

* To provide information about all the routes available through the map.
* To provide information about total time and total distance.
* To save time for the tourist and provide quick and fast results.

# REPRESENTATION :

## DESIGN MODELS :

A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed.

Individual database models are designed based on the rules and concepts of whichever broader data model the designers adopt. Most data models can be represented by an accompanying database diagram.

## ENTITY RELATIONSHIP MODEL [ ER MODEL ] :

The ER or (Entity Relational Model) is a high-level conceptual data model diagram. Entity-Relation model is based on the notion of real-world entities and the relationship between them. ER modeling helps you to analyze data requirements systematically to produce a well-designed database.

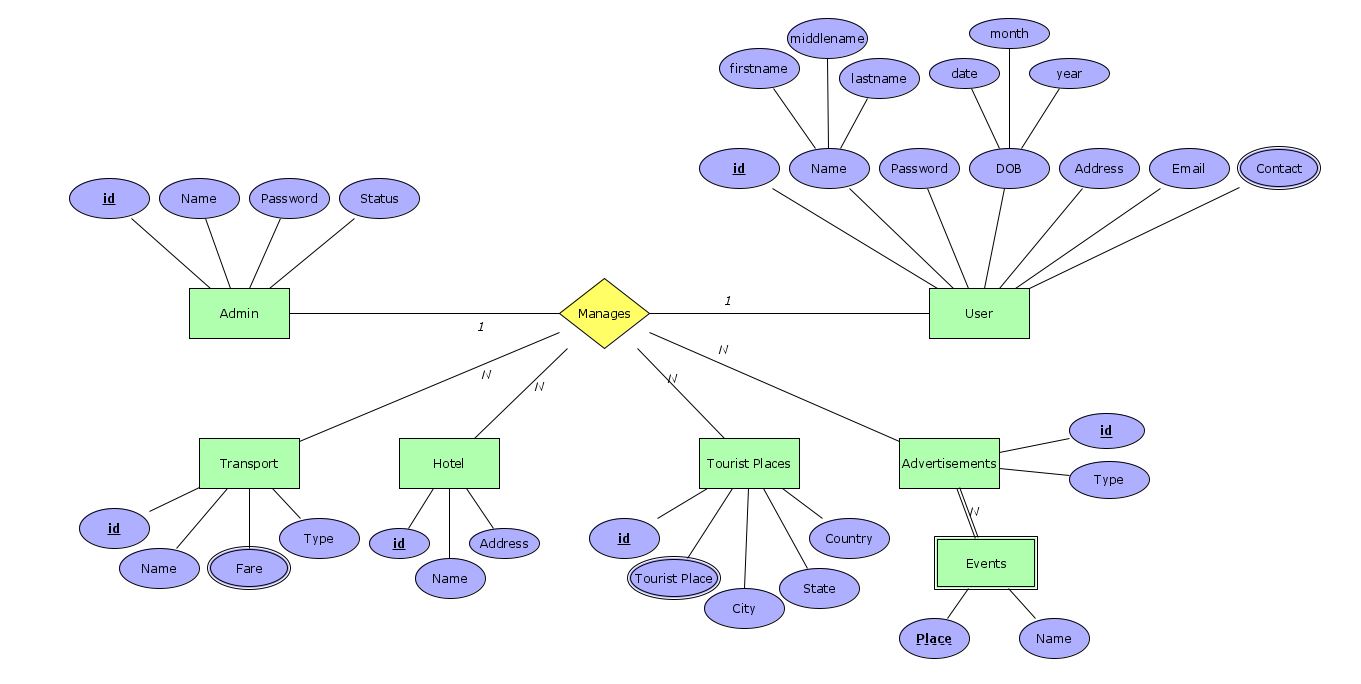


Fig : ER model for travel management system

In the above figure

* ENTITIES - Admin , User , Transport Type , Hotel , Tourist Places , Advertisements
* WEAK ENTITY - Events
* ATTRIBUTES - ID , Name , type , city , state , country , address , fare , password , DOB , email , contact , type , name
* RELATIONSHIPS - Manage
* PRIMARY KEYS - ID (of every entity)
* PARTIAL KEYS - Place

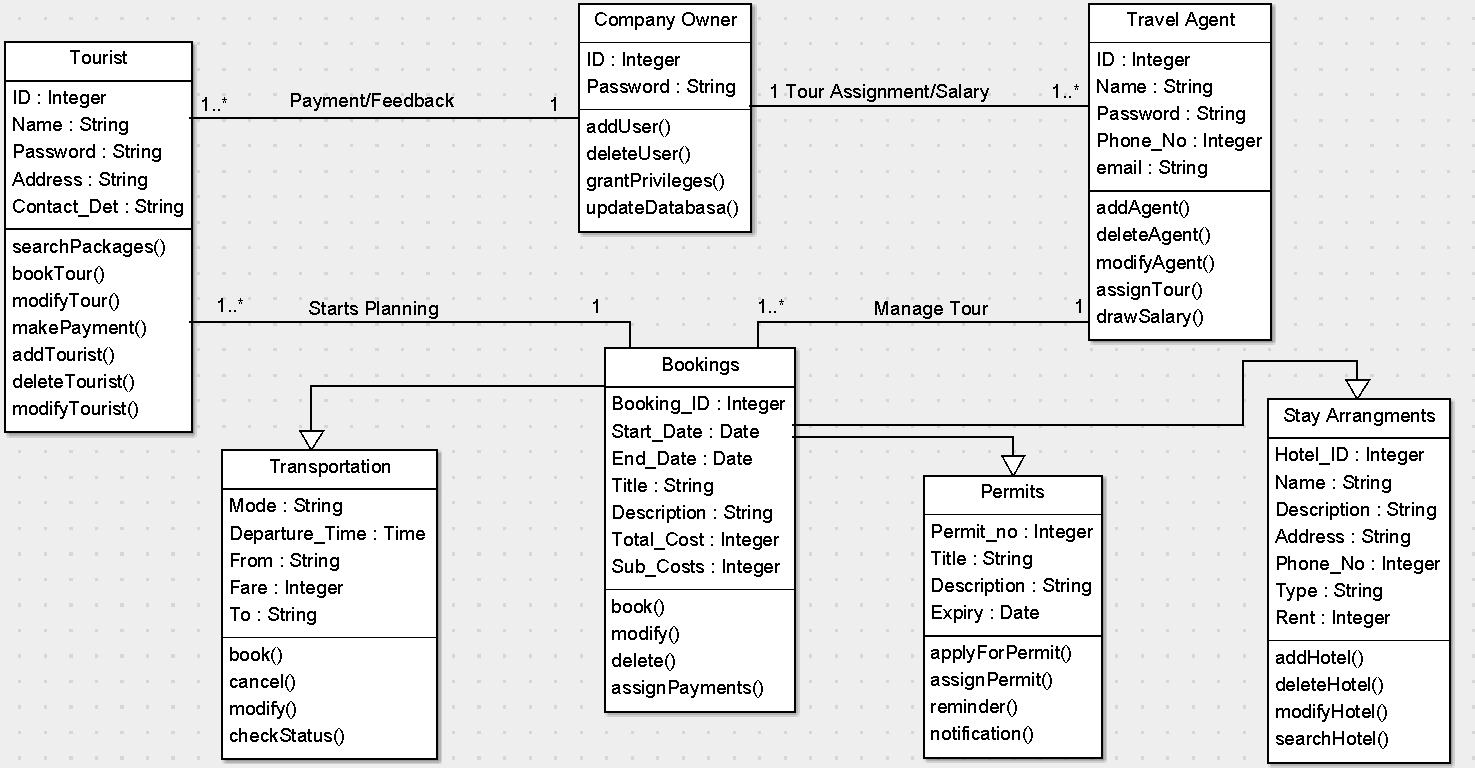
## UML DIAGRAMS :

UML is linked with **object oriented** design and analysis. UML makes the use of elements and forms associations between them to form diagrams.

We can represent it using the following UML diagrams

## CLASS DIAGRAM -

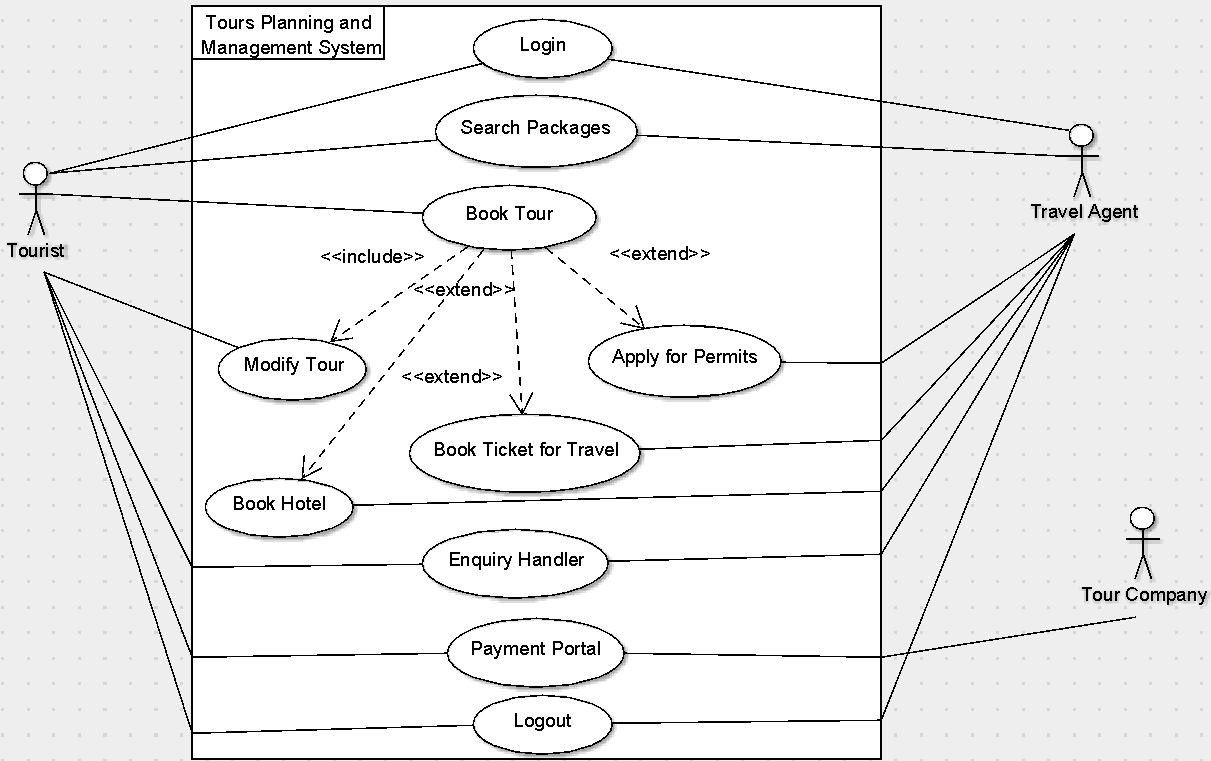
A class diagram is a static structure diagram that describes the structure of a system by showing the system’s classes, their attributes, their methods and the relationships involved among the objects.



## USE CASE DIAGRAM -

A use-case diagram is a representation of a user’s interaction with the system.

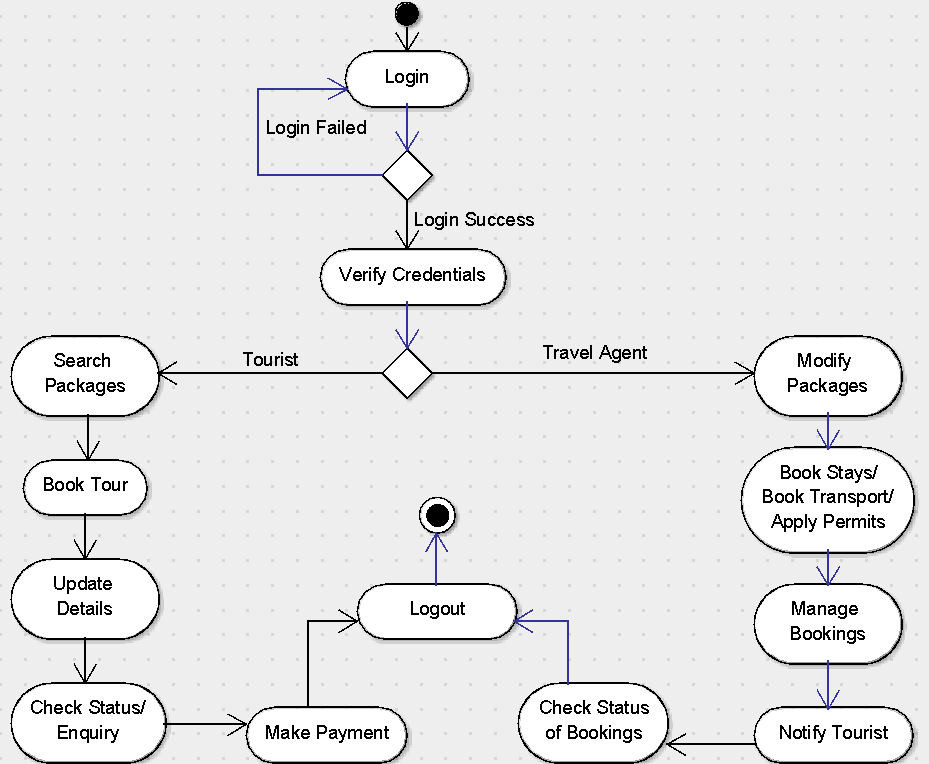
This diagram models the functionality of a system by using actors and use-cases.



## ACTIVITY DIAGRAM -

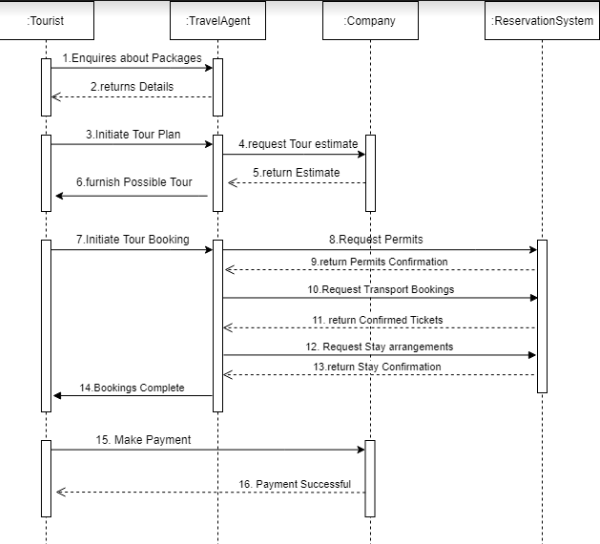
Activity diagrams are used to illustrate the flow of control in a system.

The diagram refers to the steps involved in the execution of a use-case.



## SEQUENCE DIAGRAM -

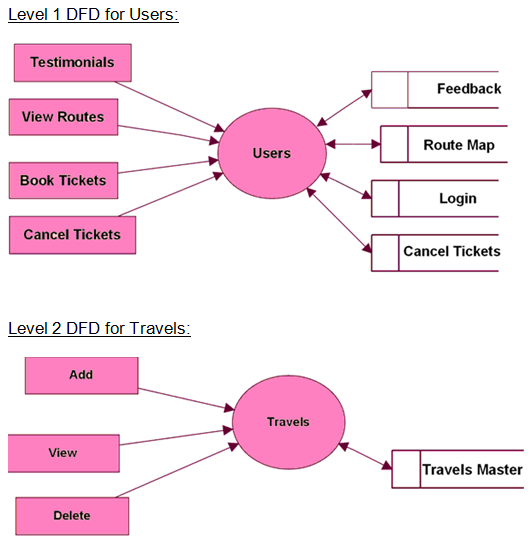
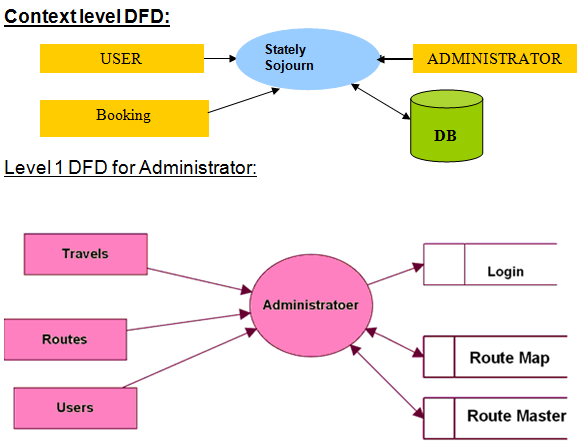
A sequence diagram depicts the interaction between objects in a sequential order, i.e. the order in which these interactions take place.



## DATA FLOW DIAGRAMS :

A data-flow diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself.

DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.



CONTEXT LEVEL - A context level data flow diagram (DFD) provides an at-a-glance look at an information system and the ways it exchanges data with outside entities.

LEVEL 1 DFD - It breaks down the main processes into subprocesses that can then be analyzed and improved on a more intimate level.

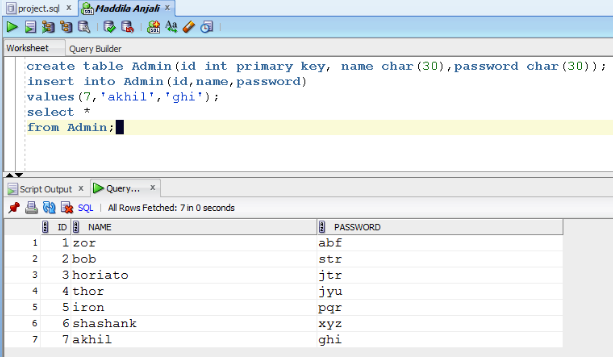
LEVEL 2 DFD - A level 2 data flow diagram (DFD) offers a more detailed look at the processes that make up an information system than a level 1 DFD does. It can be used to plan or record the specific makeup of a system.

# TABLES :

A table is a collection of related data held in a table format within a database. It consists of columns, and rows. In relational databases , a table is a set of data elements using a model of vertical columns and horizontal rows, the cell being the unit where a row and column intersect.

Using the ER diagram we create the following tables -

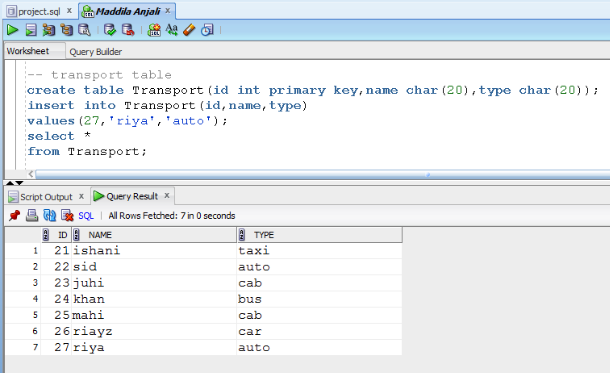
## ADMIN TABLE -



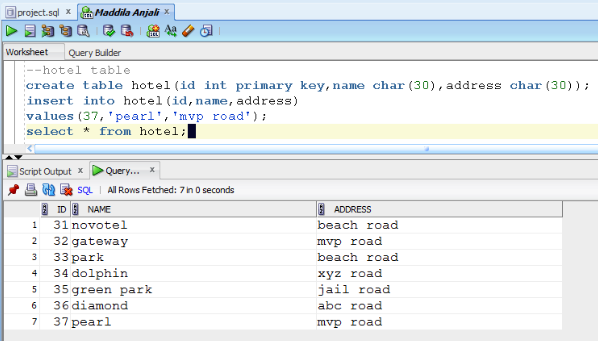
## USERS TABLE -

## 

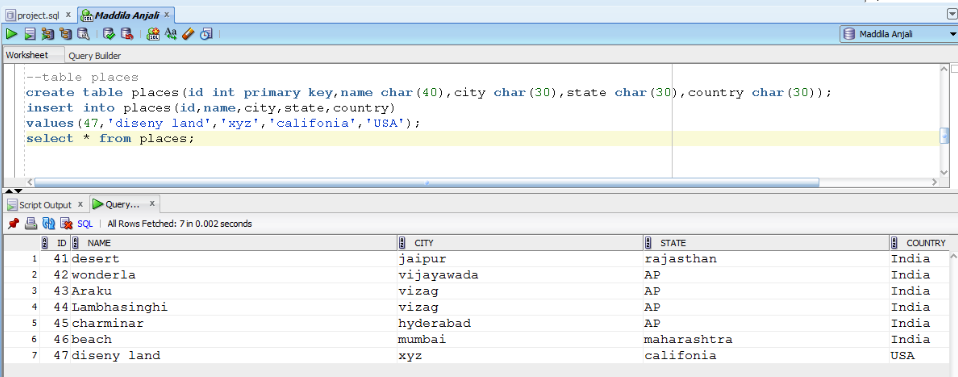
## TRANSPORT TABLE -



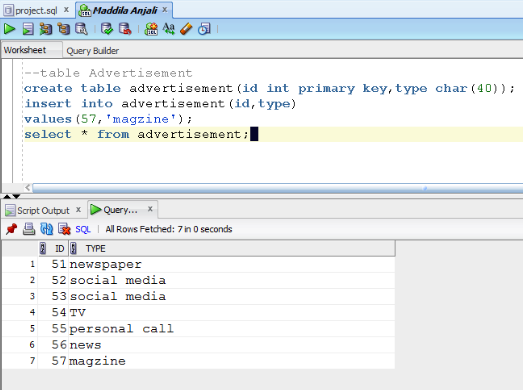
## HOTEL TABLE -



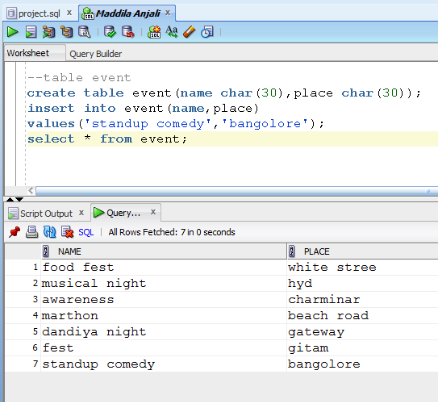
## PLACES TABLE -



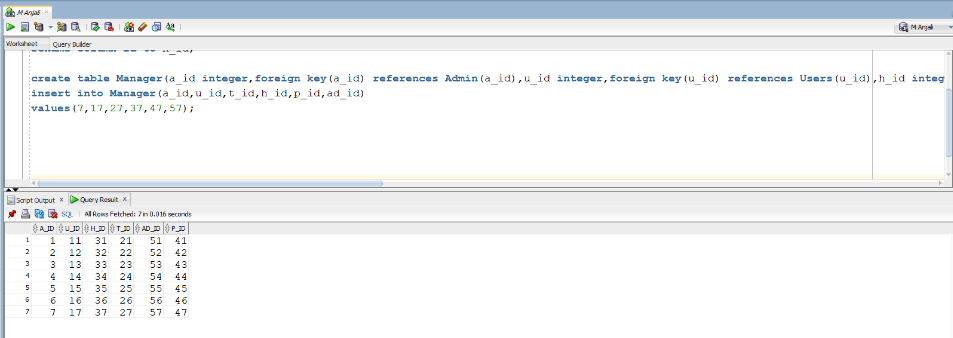
## ADVERTISEMENT TABLE -



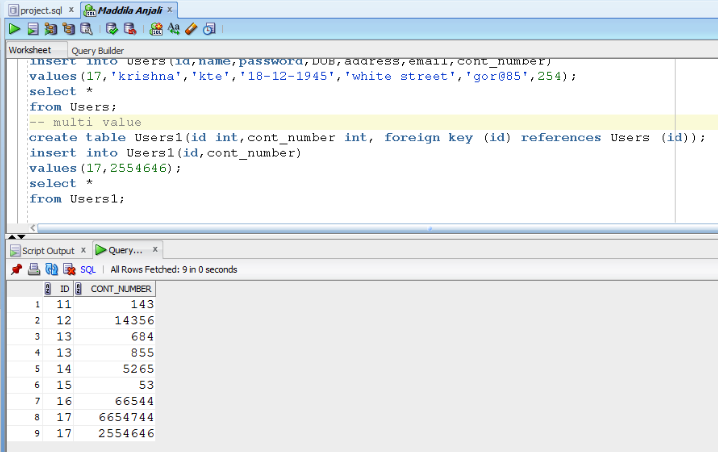
## EVENT TABLE -



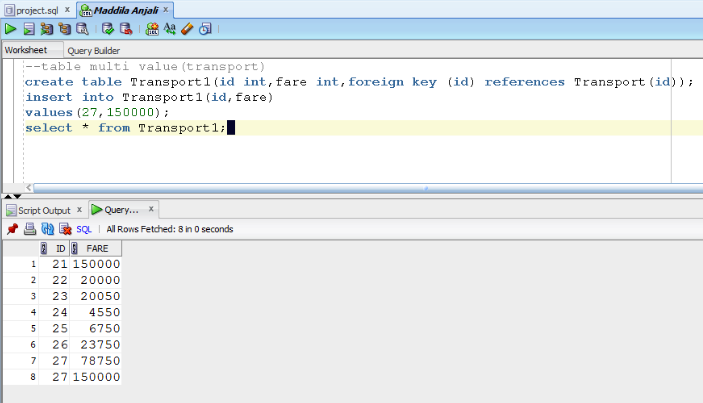
## MANAGER TABLE -



## USERS MULTI VALUE TABLE -



## TRANSPORT MULTI VALUE TABLE -



## TRIGGERS :

A database trigger is special stored procedure that is run when specific actions occur within a database.

1. **TRIGGERS ON INSERTION**

set serveroutput on;

create or replace trigger Trig\_Insert

after insert on Admin

for each row

BEGIN

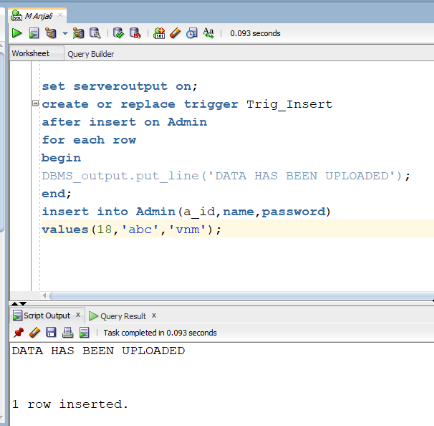
DBMS\_output.put\_line('DATA HAS BEEN UPLOADED');

END ;

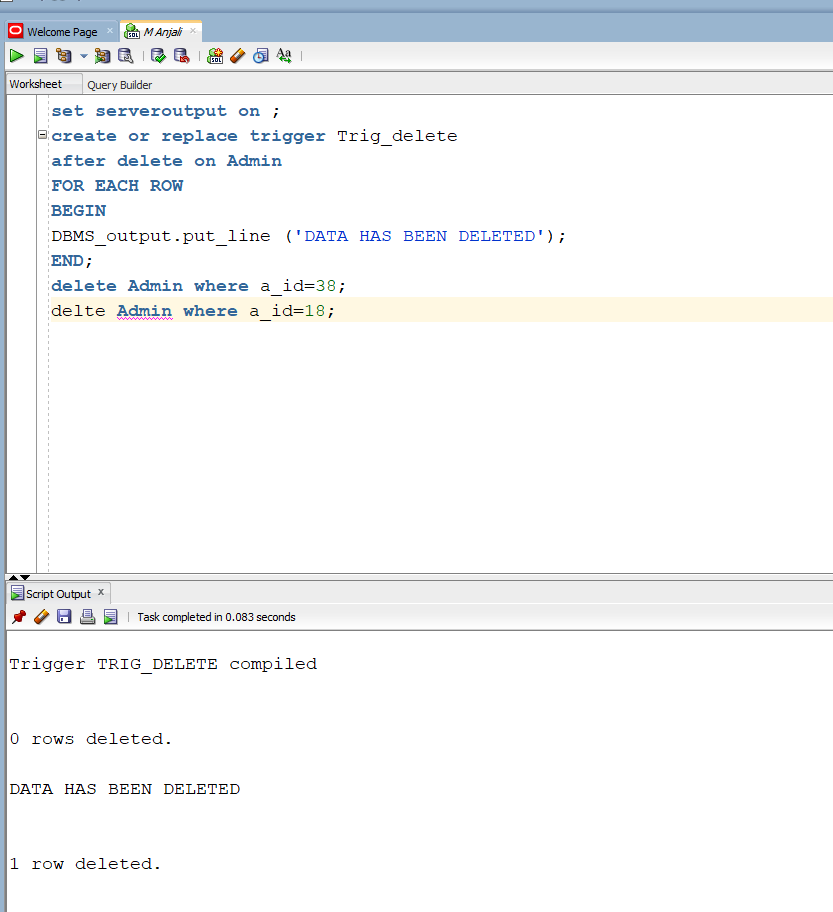
insert into Admin(a\_id,name,password)

values(18,'abc','vnm') ;

OUTPUT :-



## TRIGGERS ON DELETION :



## PROCEDURE :

A PL/SQL procedure is a reusable unit that encapsulates specific business logic of the application. Technically speaking, a PL/SQL procedure is a named block stored as a schema object in the Oracle Database.

CODE -

create or replace procedure P1( A in number , B in char , C in char ) AS

BEGIN

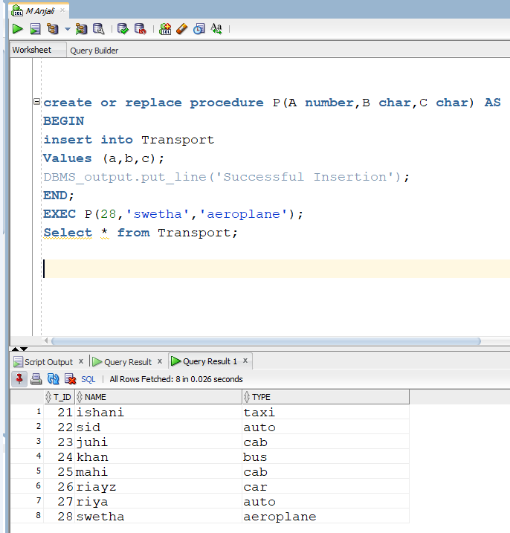
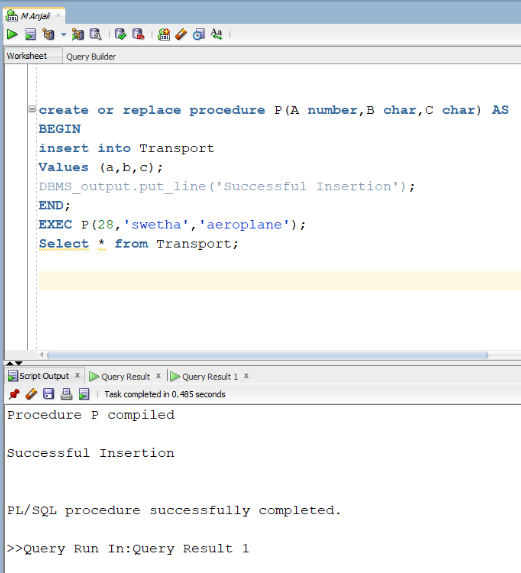
insert into Transport

Values (a , b , c );

DBMS\_output.put\_line( ‘ Successful Insertion’) ;

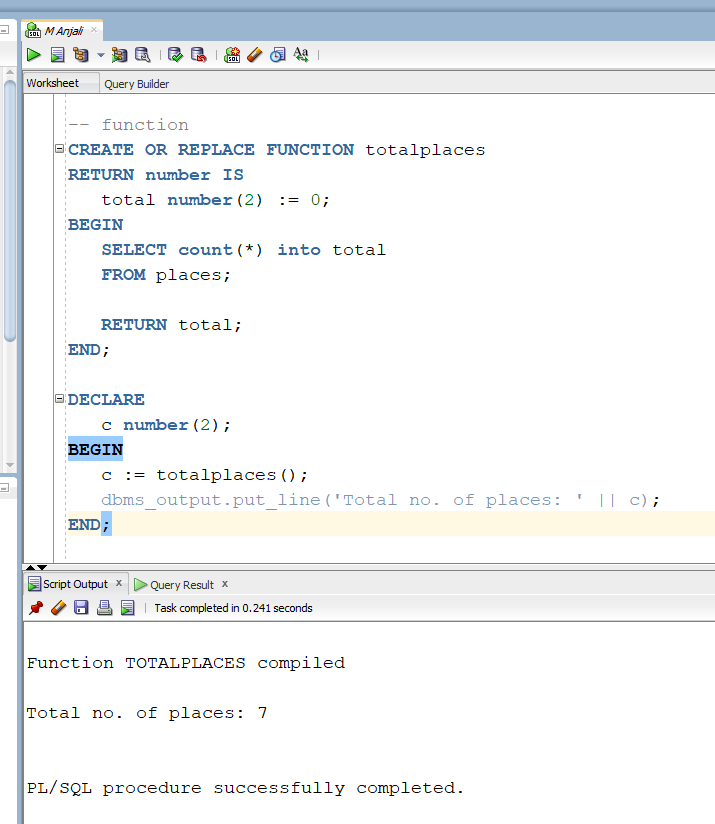
END ;

EXEC P1(28 , ‘ swetha ’ , ‘ aeroplane ‘) ;

Select \* from Transport ; 

FUNCTIONS :

Functions in PL/SQL. A function can be used as a part of SQL expression i.e. we can use them with select/update/merge commands. One most important characteristic of a function is that unlike procedures, it must return a value. function which will compute and return the reverse of a number.



# CURSOR :

# A cursor is a temporary work area created in the system memory when a SQL statement is executed. A cursor contains information on a select statement and the rows of data accessed by it. This temporary work area is used to store the data retrieved from the database, and manipulate this data. A cursor can hold more than one row, but can process only one row at a time. The set of rows the cursor holds is called the active set.

1. SQL % FOUND :

The return value is TRUE, if the DML statements like INSERT, DELETE and UPDATE affect at least one row and if SELECT ….INTO statement return at least one row.

CODE -

Set serveroutput on

BEGIN

delete hotel

Where id = & id ;

If SQL%FOUND then

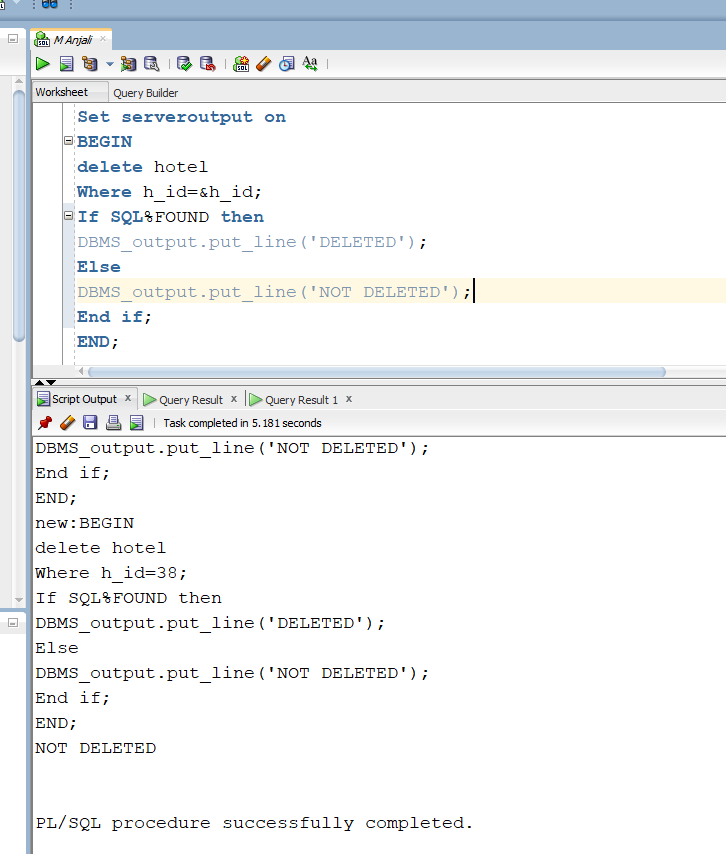
DBMS\_output.put\_line( ‘ DELETED ‘ ) ;

Else

DBMS\_output.put\_line( ‘ NOT DELETED ‘ ) ;

End if ;

END ;



## SQL % NOT FOUND

The return value is FALSE, if DML statements like INSERT, DELETE and UPDATE at least one row and if SELECT ….INTO statement return at least one row.

CODE -

Set serveroutput on

BEGIN

delete hotel

Where id = & id ;

If SQL%NOTFOUND then

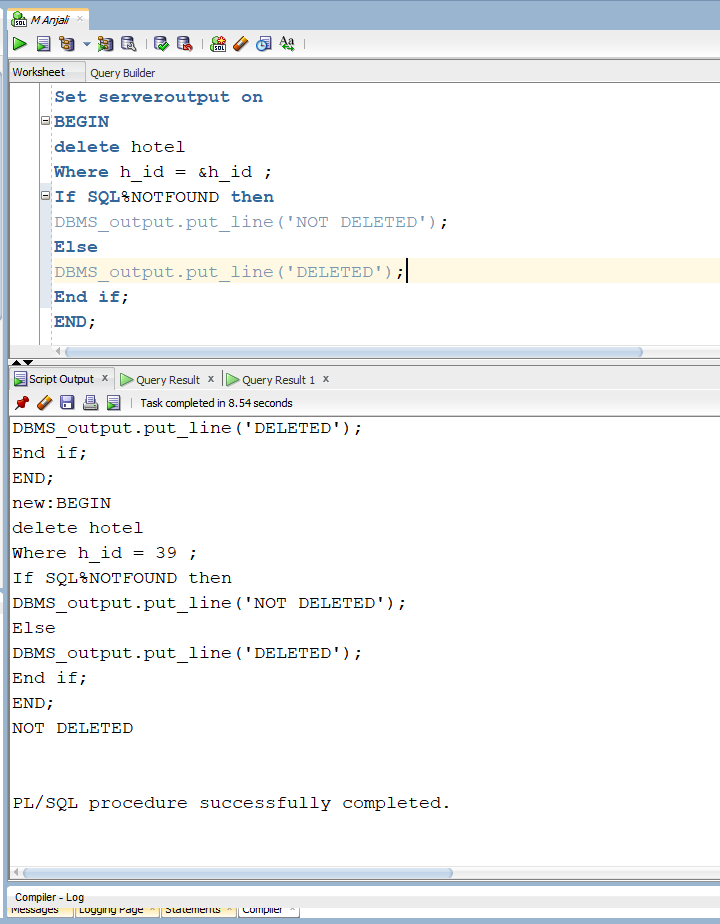
DBMS\_output.put\_line( ‘ NOT DELETED ‘ ) ;

Else

DBMS\_output.put\_line( ‘ DELETED ‘ ) ;

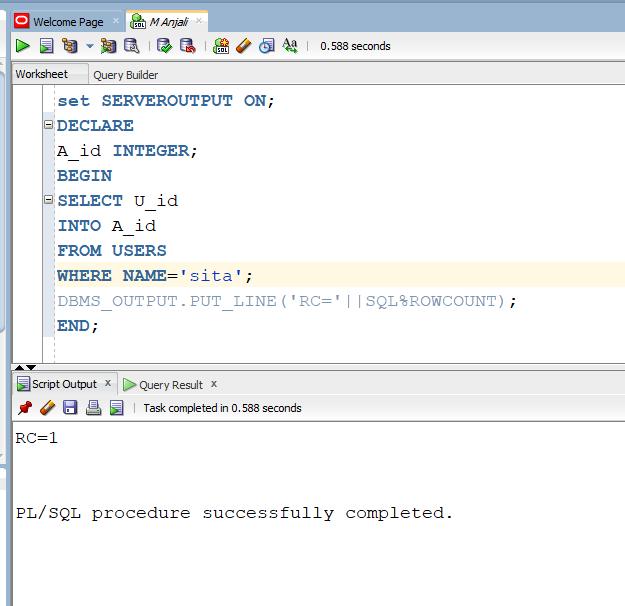
End if ;

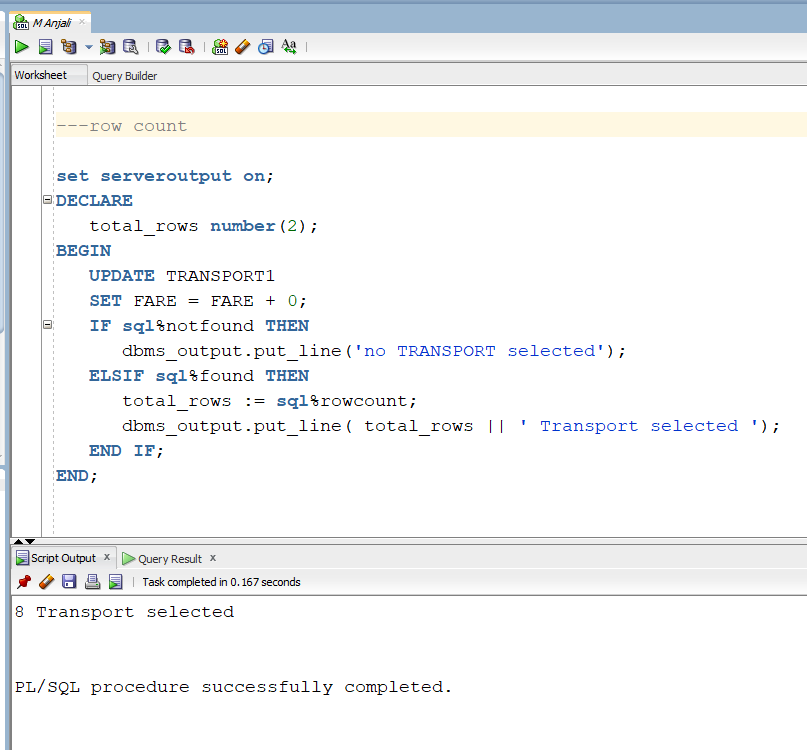
END ;



1. **SQL %ROWCOUNT**

Return the number of rows affected by the DML operations INSERT, DELETE, UPDATE, SELECT





**JDBC CONNECTIVITY -**

JDBC is an acronym for Java Database Connectivity. It’s an advancement for ODBC ( Open Database Connectivity ). JDBC is an standard API specification developed in order to move data from frontend to backend. This API consists of classes and interfaces written in Java. It basically acts as an interface (not the one we use in Java) or channel between your Java program and databases i.e it establishes a link between the two so that a programmer could send data from Java code and store it in the database for future use.

**1. Loading the Driver**  
To begin with, you first need load the driver or register it before using it in the program . Registration is to be done once in your program. You can register a driver in one of two ways mentioned below :

* Class.forName() :  Here we load the driver’s class file into memory at the runtime. No need of using new or creation of object .
* **DriverManager.registerDriver() :** DriverManager is a Java inbuilt class with a static member register. Here we call the constructor of the driver class at compile time .

1. **Create the connection**

**con:**  is a reference to Connection interface.  
**url** : Uniform Resource Locator.

1. **Create a statement**

Once a connection is established you can interact with the database. The JDBCStatement, CallableStatement, and Prepared Statement interfaces define the methods that enable you to send SQL commands and receive data from your database.

1. **Execute the query**

The executeQuery() method of Statement interface is used to execute queries of retrieving values from the database. This method returns the object of ResultSet that can be used to get all the records of a table.  
The executeUpdate(sql query) method ofStatement interface is used to execute queries of updating/inserting .

**CODE :**

import java.sql.\*;

public class test {

public static void main(String[] args)

{

try

{

String Query = "Select \* from Admin" ;

Class.forName("oracle.jdbc.driver.OracleDriver");

Connectioncon= DriverManager.getConnection("jdbc:oracle:thin:@192.168.63.144:1521:ORCL","B100030","B100030");

Statement st = con.createStatement();

ResultSet rs = st.executeQuery(Query);

while(rs.next())

{

int a\_id=rs.getInt(1);

String name=rs.getString(2);

String password=rs.getString(3);

// float age=rs.getFloat(4);

System.out.println(a\_id+" "+name+" "+password+" ");

}

st.close();

con.close();

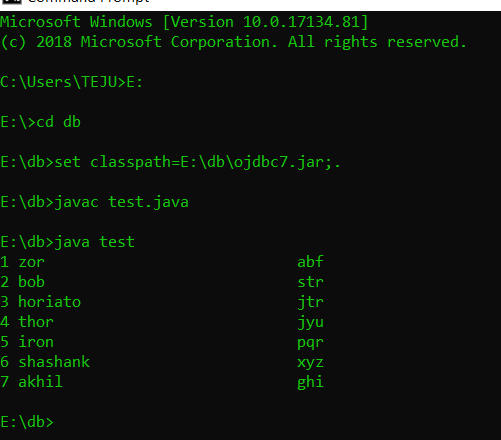
}

catch(Exception e)

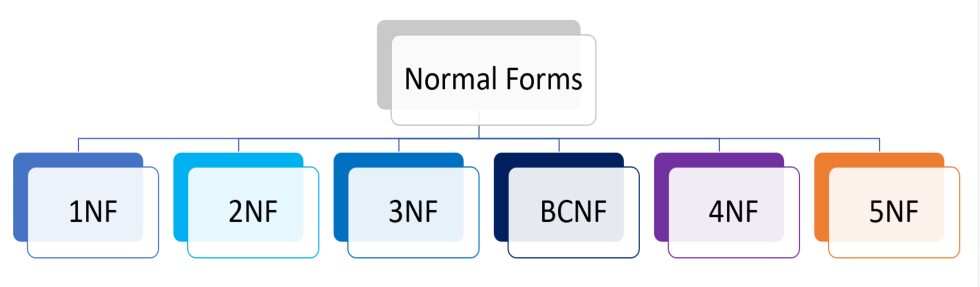
{}

}

}

OUTPUT 

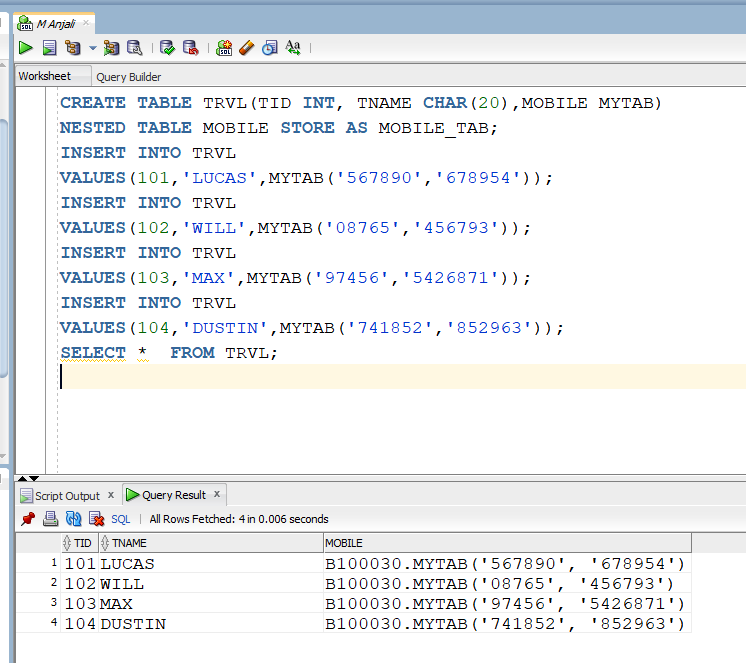
**NORMALISATION -**

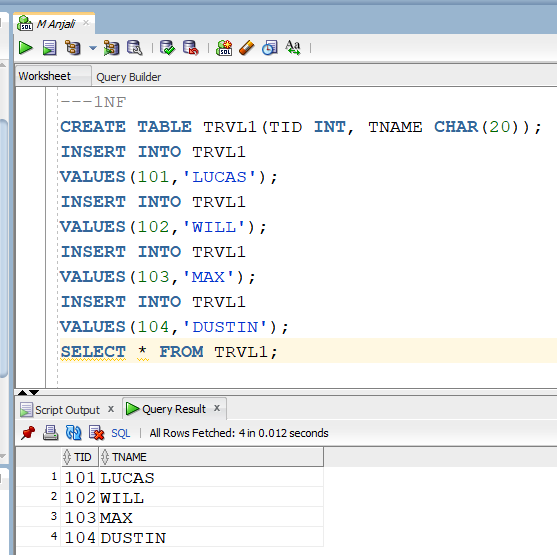


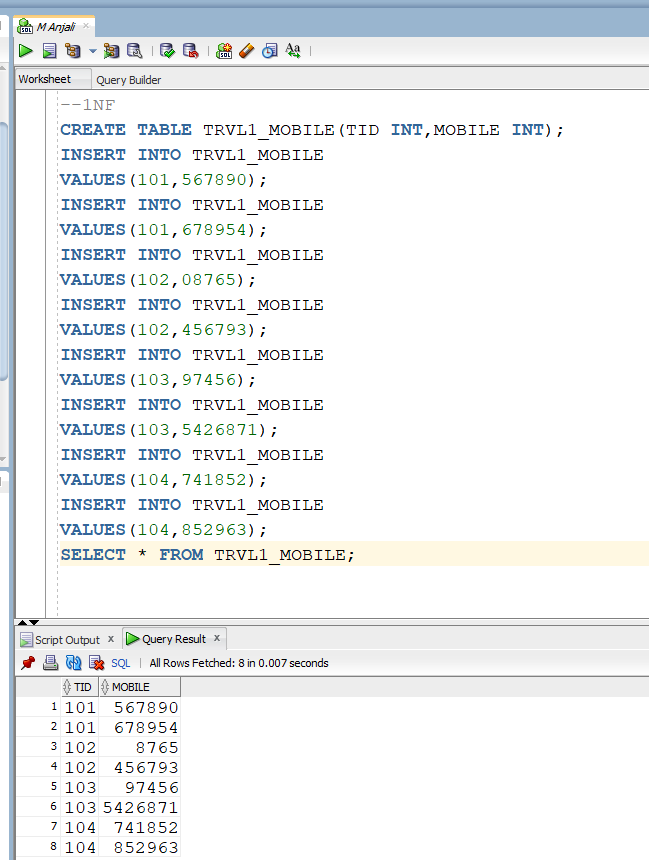
* Normalization is the process of organizing the data in the database.
* Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.
* Normalization divides the larger table into the smaller table and links them using relationship.

## ****1NF (First Normal Form) Rules****

* Each table cell should contain a single value.
* Each record needs to be unique.

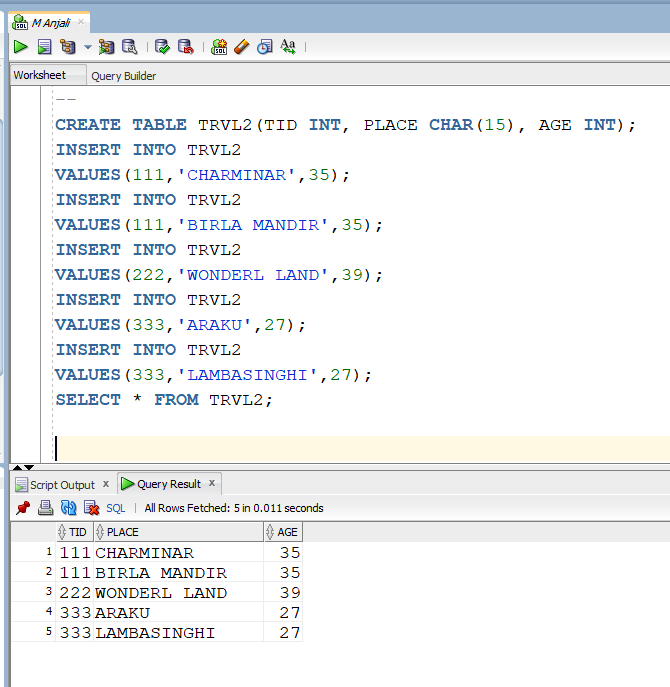


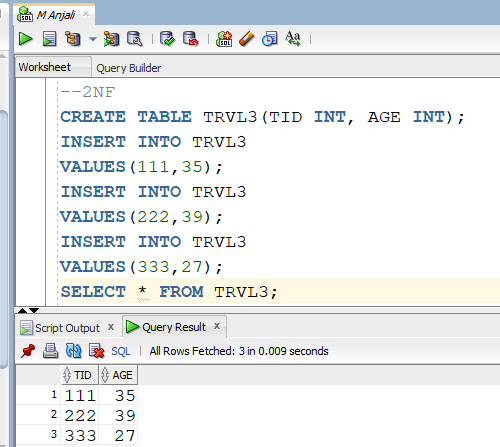


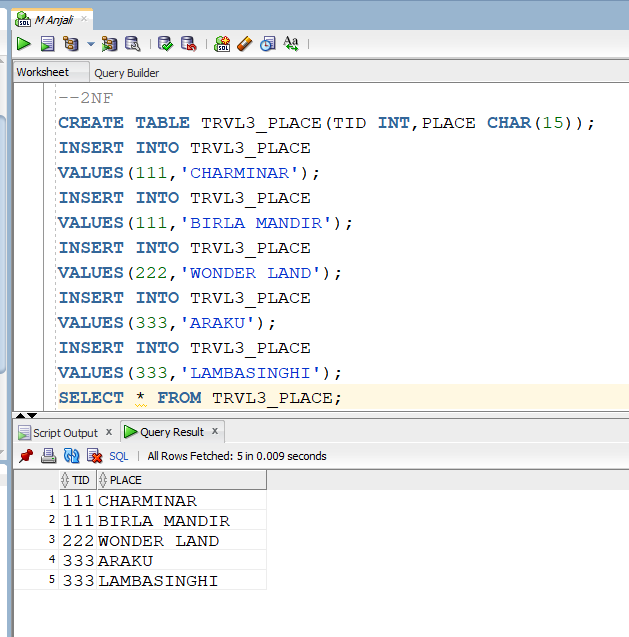


## 2NF (Second Normal Form) Rules

* Rule 1- Be in 1NF
* Rule 2- Single Column Primary Key



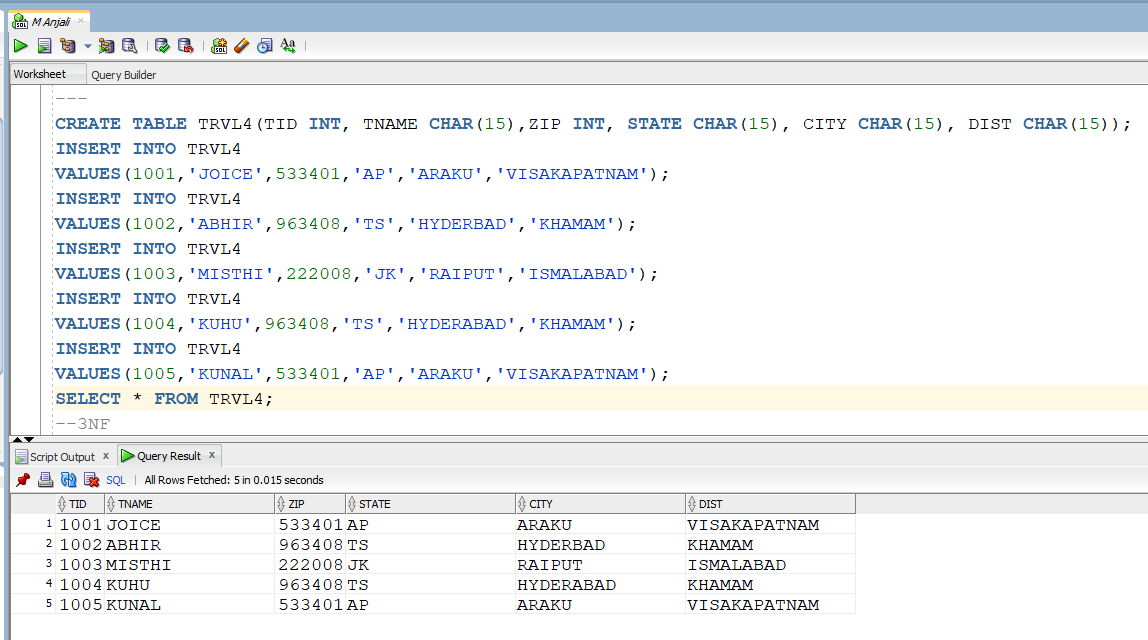


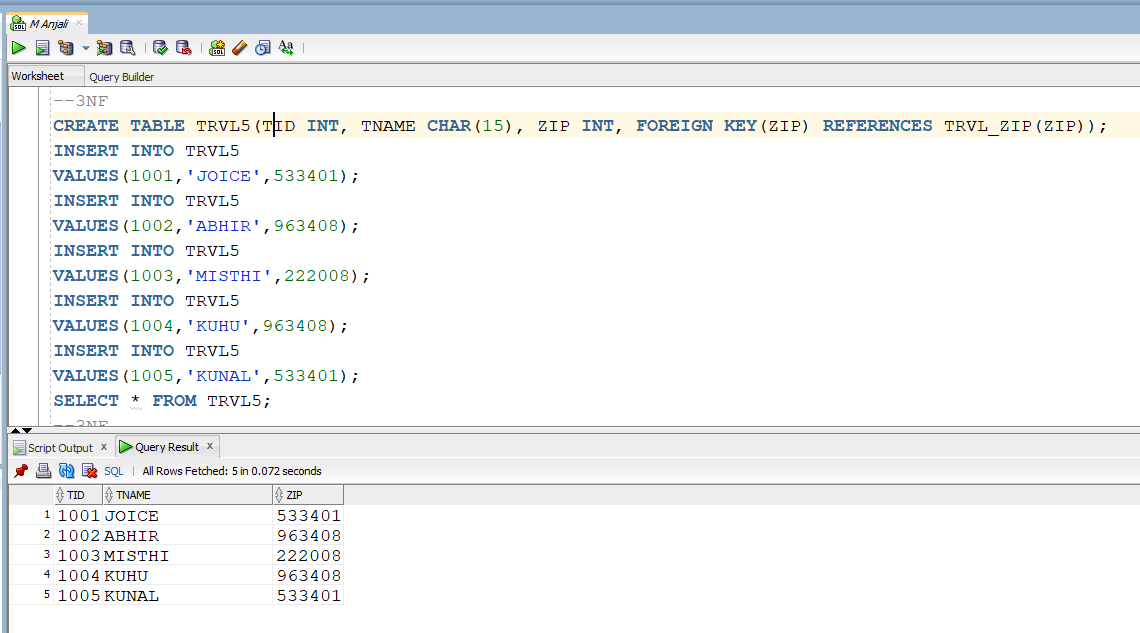


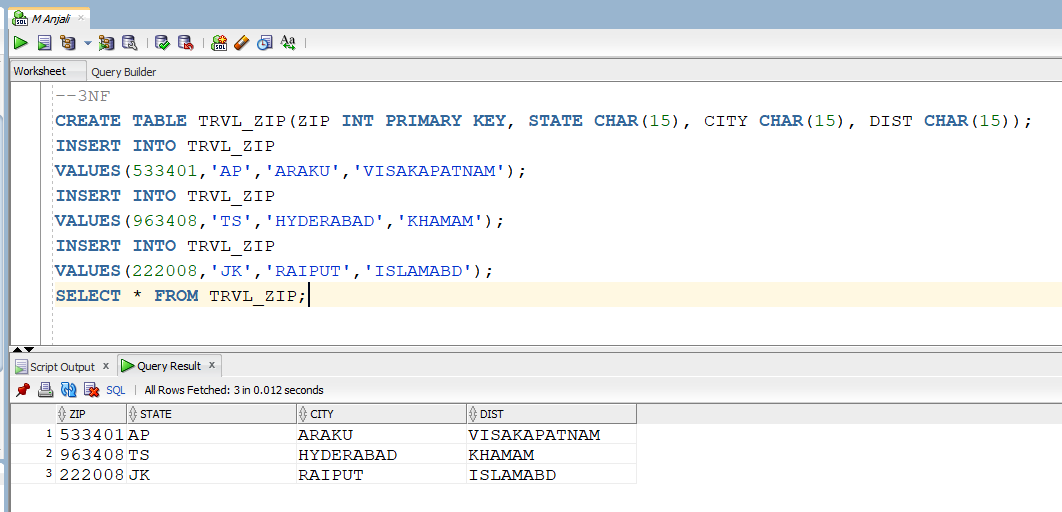
## 3NF (Third Normal Form) Rules

* Rule 1- Be in 2NF
* Rule 2- Has no transitive functional dependencies

To move our 2NF table into 3NF, we again need to again divide our table.







CONCLUSION :

In an increasingly competitive tourism industry, destinations must continuously adapt, develop, and manage their offer to ensure a quality experience for their visitors. On the other hand, technology is changing the tourism experience substantially. The idea of using technology to enhance the experience is not new.

1. Can benefit to promote tourism in our country.
2. Marketing and development of the country improves.