**Module Code & Module Title**



**CC6001NA Advanced Database Systems Development**

**Assessment Weightage & Type**

**40% Individual coursework**

**Year and Semester**

**2018-19 Autumn**

**Student Name: Rajat Shrestha**

**London Met ID:** **17030954**

**College ID: np01cp4a170021**

**Assignment Due Date: 30th January 2020**

**Assignment Submission Date: 30th January 2020**

*I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.*

Table of Contents

[1. Introduction 1](#_Toc35007528)

[2. Normalization 2](#_Toc35007529)

[2.1. Figure 1 3](#_Toc35007530)

[2.1.1. Un-normalized Form (UNF): 4](#_Toc35007531)

[2.1.2. First Normal Form (1NF): 5](#_Toc35007532)

[2.1.3. Second Normal Form (2NF): 6](#_Toc35007533)

[2.1.4. Third Normal Form (3NF): 7](#_Toc35007534)

[2.2. Figure 2 8](#_Toc35007535)

[2.2.1. Assumptions For Figure 2: 8](#_Toc35007536)

[2.2.2. Un-normalized Form (UNF): 10](#_Toc35007537)

[2.2.3. First Normal Form (1NF): 10](#_Toc35007538)

[2.2.4. Second Normal Form (2NF): 11](#_Toc35007539)

[2.2.5. Third Normal Form (3NF): 12](#_Toc35007540)

[2.3. Integration 13](#_Toc35007541)

[3. ER-Diagram 14](#_Toc35007542)

[3.1. Assumptions: 14](#_Toc35007543)

[3.2. Final ER-diagram: 15](#_Toc35007544)

[4. Data Dictionary 16](#_Toc35007545)

[5. Generation of Database 19](#_Toc35007546)

[5.1. Create Statements 19](#_Toc35007547)

[5.1.1. Generating DDL Script and Creating Tables: 19](#_Toc35007548)

[5.2. Insert Statements 24](#_Toc35007549)

[5.3. Select Statements 30](#_Toc35007550)

[5.3.1. Day Activities 30](#_Toc35007551)

[5.3.2. Activity 31](#_Toc35007552)

[5.3.3. Itinerary 32](#_Toc35007553)

[5.3.4. Destination 33](#_Toc35007554)

[5.3.5. Packages 34](#_Toc35007555)

[5.3.6. Customer Packages 35](#_Toc35007556)

[5.3.7. Package Guide 36](#_Toc35007557)

[5.3.8. Customer 37](#_Toc35007558)

[5.3.9. Staff 38](#_Toc35007559)

[5.3.10. Role 39](#_Toc35007560)

[6. Implementation of Web-Based Database Application 40](#_Toc35007561)

[6.1 Basic Webforms 40](#_Toc35007562)

[6.1.1. Staff Details 40](#_Toc35007563)

[6.1.2. Designation Details 41](#_Toc35007564)

[6.1.3. Customer Details 42](#_Toc35007565)

[6.1.4. Package Details 43](#_Toc35007566)

[6.1.5. Tour Guide Details 44](#_Toc35007567)

[6.2. Complex Webforms 45](#_Toc35007568)

[6.2.1. Customer-Package Schedule Form 45](#_Toc35007569)

[6.2.2. Staff-Role Mapping Form 46](#_Toc35007570)

[6.2.3. Package-Activity Schedule Form 47](#_Toc35007571)

[6.3. Dashboard 48](#_Toc35007572)

[7. Testing: 49](#_Toc35007573)

[7.1. Simple Forms: 49](#_Toc35007574)

[7.1.1. Adding Data 49](#_Toc35007575)

[7.1.2. Editing Data 50](#_Toc35007576)

[7.1.3. Deleting Data 51](#_Toc35007577)

[7.1.4. Adding Duplicate Data 52](#_Toc35007578)

[7.1.5. Deleting referenced entry 53](#_Toc35007579)

[7.2. Complex Forms: 54](#_Toc35007580)

[7.2.1. Filtering the Package-Customer Form 54](#_Toc35007581)

[7.2.2. Filtering Staff-Role mapping Form 55](#_Toc35007582)

[7.2.3. Filtering Package-Activity Schedule Form 56](#_Toc35007583)

[7.3. Dashboard 57](#_Toc35007584)

[7.3.1. Testing All of the Links 57](#_Toc35007585)

[7.3.2. Testing the charts in the Dashboard 58](#_Toc35007586)

[8. User Manual 59](#_Toc35007587)

[8.1. CRUD Operations 59](#_Toc35007588)

[8.1.1. Addition 59](#_Toc35007589)

[8.1.2. Editing 60](#_Toc35007590)

[8.1.3. Deleting 61](#_Toc35007591)

[9. Further Discussion 62](#_Toc35007592)

[References 63](#_Toc35007593)

[Appendix 64](#_Toc35007594)

**Table of Figures:**

[Figure 1: Basic Entities observed via the initial observation 2](#_Toc35007595)

[Figure 2: Figure 1 as provided by the question 3](#_Toc35007596)

[Figure 3: Figure 2 as provided by the question for tracking information 8](#_Toc35007597)

[Figure 4: ER-Diagram created using the given entities 15](file:///C:\Users\Rajat\Desktop\AD%20Documentation.docx#_Toc35007598)

[Figure 5: Process of generating the DDL Script via SQL developer Datamodeler 19](#_Toc35007599)

[Figure 6: Running the DDL Script in SQL Developer 23](#_Toc35007600)

[Figure 7: Running the Insert Statements to populate the database in SQL Developer 29](#_Toc35007601)

[Figure 8: Running select Statement on Day Activities Table 30](#_Toc35007602)

[Figure 9: Running select Statement on Activity Table 31](#_Toc35007603)

[Figure 10: Running select Statement on Itinerary Table 32](#_Toc35007604)

[Figure 11: Running select Statement on Destination Table 33](#_Toc35007605)

[Figure 12: Running select Statement on Packages Table 34](#_Toc35007606)

[Figure 13: Running select Statement on Customer packages Table 35](#_Toc35007607)

[Figure 14: Running select Statement on Package Guide Table 36](#_Toc35007608)

[Figure 15: Running select Statement on Customer Table 37](#_Toc35007609)

[Figure 16: Running select Statement on Staff Table 38](#_Toc35007610)

[Figure 17: Running select Statement on Role Table 39](#_Toc35007611)

[Figure 18: Simple Web-forms for Staff Details 40](#_Toc35007612)

[Figure 19: Simple Web-forms for Designation Details 41](#_Toc35007613)

[Figure 20: Simple Web-forms for Customer Details 42](#_Toc35007614)

[Figure 21: Simple Web-forms for Package Details 43](#_Toc35007615)

[Figure 22: Simple Web-forms for Tour Guide Details 44](#_Toc35007616)

[Figure 23: Complex Web-forms for Customer-Package Schedule Form 45](#_Toc35007617)

[Figure 24: Complex Web-forms for Staff-Role Mapping Form 46](#_Toc35007618)

[Figure 25: Complex Web-forms for Package Activity Schedule Form 47](#_Toc35007619)

[Figure 26: The Dashboard with Links to webforms and Charts 48](#_Toc35007620)

[Figure 27: Dashboard with details on the Number of data entered in the database 48](#_Toc35007621)

[Figure 28: Entering Data in the Customer Form 49](#_Toc35007622)

[Figure 29: New Customer inserted in the customer table 49](#_Toc35007623)

[Figure 30: Staff data before editing 50](#_Toc35007624)

[Figure 31: Editing Staff Data 50](#_Toc35007625)

[Figure 32: Staff data after editing 50](#_Toc35007626)

[Figure 33: Designations before deleting 51](#_Toc35007627)

[Figure 34: Intern designation removed after Deleting 51](#_Toc35007628)

[Figure 35: Editing the Tour-guide to be duplicate 52](#_Toc35007629)

[Figure 36: Exception occurred due to duplicate data 52](#_Toc35007630)

[Figure 37: All the Packages are currently referenced 53](#_Toc35007631)

[Figure 38: Deleting one referenced data (package) 53](#_Toc35007632)

[Figure 39: Exception occurred while deleting referenced item 53](#_Toc35007633)

[Figure 40: Selecting Package Id to show customer booking the selected package 54](#_Toc35007634)

[Figure 41: Table showing customers who is booking the selected Package 54](#_Toc35007635)

[Figure 42: Selecting the Designation to count the no of staffs in it and staff details 55](#_Toc35007636)

[Figure 43: Staff details and count of the selected designation 55](#_Toc35007637)

[Figure 44: Selecting Package Id to view the enlisted activities 56](#_Toc35007638)

[Figure 45: Enlisted Activities of the selected package 56](#_Toc35007639)

[Figure 46: Clicking the staff role link 57](#_Toc35007640)

[Figure 47: Getting redirected to the staff-role page 57](#_Toc35007641)

[Figure 48: Charts and data before updating data in the database 58](#_Toc35007642)

[Figure 49: Charts and data after updating the database 58](#_Toc35007643)

**Table of Tables:**

[Table 1: Storing Value from the given figure in Atomic cells 3](#_Toc35007644)

[Table 2: Storing the given data in Atomic cells 9](#_Toc35007645)

[Table 3: Data Dictionary for the Activity table 16](#_Toc35007646)

[Table 4: Data Dictionary for Day Activities table 16](#_Toc35007647)

[Table 5: Data Dictionary for Itinerary Table 16](#_Toc35007648)

[Table 6: Data Dictionary for Destination Table 16](#_Toc35007649)

[Table 7: Data Dictionary for Package Table 17](#_Toc35007650)

[Table 8: Data Dictionary for Staff Table 17](#_Toc35007651)

[Table 9: Data Dictionary for Customer Table 17](#_Toc35007652)

[Table 10: Data Dictionary for Customer-Package Table 17](#_Toc35007653)

[Table 11: Data Dictionary for Package Guide Table 18](#_Toc35007654)

[Table 12: Data Dictionary for role Table 18](#_Toc35007655)

[Table 13: Test 1 Simple Form Adding Data 49](#_Toc35007656)

[Table 14: Test 2 Simple Form Editing Data 50](#_Toc35007657)

[Table 15: Test 3 Simple Form Removing data 51](#_Toc35007658)

[Table 16: Test 4 Simple Form Adding duplicate data 52](#_Toc35007659)

[Table 17: Test 5 Simple Form deleting referenced data 53](#_Toc35007660)

[Table 18: Test 6 Complex Form Customer-Package Filtering 54](#_Toc35007661)

[Table 19: Test 7 Complex Form Staff role mapping 55](#_Toc35007662)

[Table 20: Test 8 Complex Form Package activities 56](#_Toc35007663)

[Table 21: test 9 Checking Links in dashboard 57](#_Toc35007664)

[Table 22: Test 10: Checking Graphs in Dashboard 58](#_Toc35007665)

# Introduction

This is the first coursework of advanced database systems development year-long module and details in the development of a database system to manage daily operations in a travel company. This project includes the development of a Dashboard to view the overall picture of the data entered in the database, view data in the database, add data in the database and edit the data in the database. At first, this report discusses the overall process of the development of how the entities were managed and normalized into various tables to make the database more effective by reducing data redundancy. The report for database design includes the steps for Normalization, Final Third Normal Form of the database, Integrating the Normalization of two different viewpoints of daily transaction, the addition of New entities, assumptions, and finally, the combined ER-Diagram created according to the provided instruction. Then the tables are created, populated with data and then thoroughly checked. After the insertion of data simple forms are created to manipulate the data, complex forms for viewing data from multiple tables. The forms are created by the help of .Net Framework of C# Programming Language by providing an interface to do various operations on the connected Oracle Database.

The Final Application created is quite useful to view insights on the data inserted to the database and provides basic CRUD functions for essential entities such as Designation, Staffs, Customers, Assigned Tour-Guide, and Packages. Adding Packages is currently only possible via manual insertion by the administrator so it is not included. On top of the basic forms, the complex forms give the option to view various option such as Packages booked by the customer, Staff Count and Details, and Package-Activity details which provide essential operations to the operator. The dashboard provided also includes various tables and charts to give insight on the data stored. The instructions to use all of these forms in the application are described in the user manual and are extensively tested. All of the essential screenshots of the development phase is provided in chronological order.

# Normalization

It is the process of correcting table structure to reduce redundancy and data anomalies, which minimizes storage space. It applies a series of rules called normal forms (Coronel & Morris, 2018). The database in the coursework is required to be normalized till 3NF.

According to the scenario the tour company have the following relation between the entities:

* Staffs have different roles assigned one for each
* Multiple Staffs (specifically tour guides) could be assigned for a tour
* Customer can only take one package from multiple available packages.
* The tour and package are taken as the same entity
* A registered customer might or might not have a tour assigned
* The role is taken as an attribute of staff

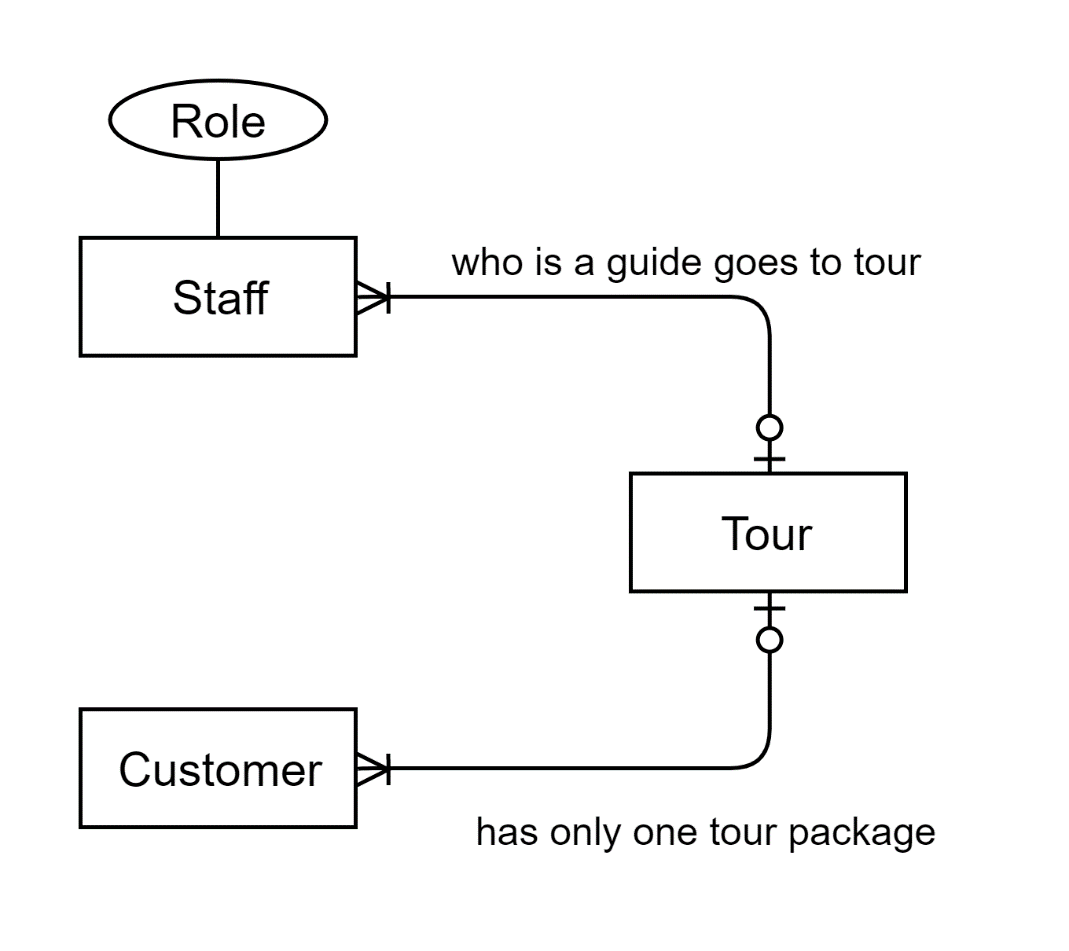


Figure : Basic Entities observed via the initial observation

## Figure 1

The given figure gives us detail about the package record.

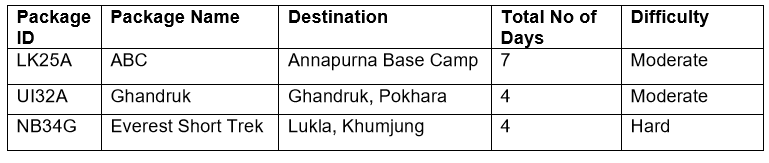


Figure : Figure 1 as provided by the question

As all the information is enough for a package record, storing the given records as atomic value(i.e. storing each value in unique cells) we get:

Table : Storing Value from the given figure in Atomic cells

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Package ID** | **Package Name** | **Destination** | **Days** | **Difficulty** |
| LK25A | ABC | Annapurna Base Camp | 7 | Moderate |
| UI32A | Ghandruk | Ghandruk | 4 | Moderate |
| UI32A | Ghandruk | Pokhara | 4 | Moderate |
| NB34G | Everest Short Trek | Lukla | 4 | Hard |
| NB34G | Everest Short Trek | Khumjung | 4 | Hard |

### Un-normalized Form (UNF):

A database is said to be in UNF when it has not been normalized at all. The rules for creating a un-normalized form are:

* Entity and its attributes should be identified
* A Primary key need to be stated
* The repeating group should be acknowledged.

In relation, a distinctive describing a group of multiple entries for a single key attribute occurrence can be known as a repeating group. Example: Multiple items purchased by a customer in a bill (Coronel & Morris, 2018). From the above table we can observe that the destination column has repeating groups in it, representing the given data in UNF:

UNF: (**Package\_ID**, Pakage\_Name, {Destination}, Total\_Days, Difficulty)

The destination is the repeating group because a package might have multiple destinations, for example, Pokhara and Ghandruk. Package\_ID is taken as the primary key as it can uniquely identify all columns.

### First Normal Form (1NF):

In First Normal Form, only atomic values are allowed at each cell and discourage repeating groups. For the database to be in 1NF it must be in UNF. The other rules for 1NF are:

* Primary Keys should be identified.
* Repeating groups from UNF must be separated.
* New table should have Composite Primary key including the Primary key of the original table.

As Package\_Name, Total\_Days, and Difficulty depend upon the Package\_ID, it is assigned as a primary key. Package\_Name also gives Destination

Package\_ID 🡺 Pakage\_Name, Total\_Days, Difficulty

Package\_ID 🡺 Destination

Removing the repeating group and creating the Destination Package and assigning Package\_ID and Destination\_ID as composite key and creating the first normal form.

Package (**Package\_ID**, Pakage\_Name, Total\_Days, Difficulty)

Destination\_Package (**Package\_ID\*, Destination**)

### Second Normal Form (2NF):

For the relation to be in Second Normal Form it must already be in First Normal Form and should not include any partial dependencies. Partial dependency is the kind of functional dependencies in which a non-key is dependent on part of a composite key (Coronel & Morris, 2018). The rules for 2NF are:

* All the functional dependencies between no key and composite key and parts of the composite key should be shown for the entities with composite Primary Key.
* Non-keys which are dependent on a part of composite key should be moved to a new table and identify its keys.

In the following situation, there are no Partial dependencies as all the non-key elements are directly dependent on their respective primary key.

Package\_ID 🡺 Pakage\_Name, Total\_Days, Difficulty

Package\_ID and Destination gives the unique value for every package-destination pair:

Package (**Package\_ID**, Pakage\_Name, Total\_Days, Difficulty)

Destination\_Package (**Package\_ID\***, **Destination**)

### Third Normal Form (3NF):

For the database to be in Third Normal Form it must already be in Second Normal Form and should not include any transitive dependencies. Transitive dependency is a type of functional dependency in which a non-key is dependent on another non-key element (Coronel & Morris, 2018). The rules for 3NF are:

* Functional dependencies between no key and non-key should be separated into a new table in case of an entity with multiple Non-key.
* Primary Keys of the new table should be identified.

There are no transitive dependencies as every none key entries are dependent on the primary key assigned for each table.

Package (**Package\_ID**, Pakage\_Name, Total\_Days, Difficulty)

Destination\_Package (**Package\_ID\***, **Destination**)

Since there are no transitive dependencies this is the final normalized form

## Figure 2

The second figure gives us detail about the tracking information.

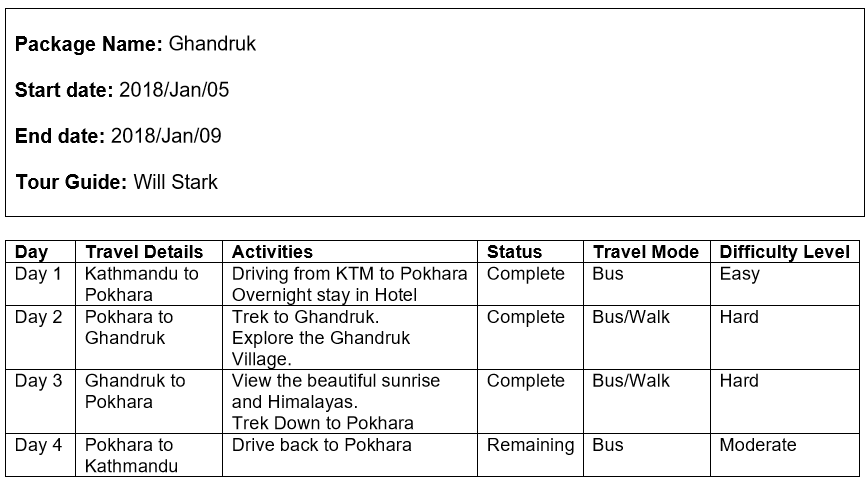


Figure : Figure 2 as provided by the question for tracking information

### Assumptions For Figure 2:

* Activity, Travel Mode and Status depends on the Activity ID assigned for dividing activities
* A day in a package will determine travel details, difficulty and unique sets of activities consisting of activity, travel mode and the status of the day.
* Package ID details the tracking information of the tour.
* Status describes the activity progress

Storing the given records in unique cells we get the following table:

Table : Storing the given data in Atomic cells

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Package ID** | **Package Name** | **Start Date** | **End Date** | **Guide** | **Day** | **Travel Details** | **Difficulty** | **Activity ID** | **Activities** | **Travel Mode** | **Status** |
| UI32A | Ghandruk | 1st Jan 2019 | 7th Jan 2019 | Will Stark | Day 1 | Kathmandu to Pokhara | Easy | A1 | Driving from KTM to Pokhara. | Bus | Complete |
| UI32A | Ghandruk | 1st Jan 2019 | 7th Jan 2019 | Will Stark | Day 1 | Kathmandu to Pokhara | Easy | A2 | Overnight stay in the Hotel. | Walk | Complete |
| UI32A | Ghandruk | 1st Jan 2019 | 7th Jan 2019 | Will Stark | Day 2 | Pokhara to Ghandruk | Hard | A3 | Trek to Ghandruk. | Bus | Complete |
| UI32A | Ghandruk | 1st Jan 2019 | 7th Jan 2019 | Will Stark | Day 2 | Pokhara to Ghandruk | Hard | A4 | Explore the Ghandruk Village. | Walk | Complete |
| UI32A | Ghandruk | 1st Jan 2019 | 7th Jan 2019 | Will Stark | Day 3 | Ghandruk to Pokhara | Easy | A5 | View the sunrise and Himalayas. | Walk | Complete |
| UI32A | Ghandruk | 1st Jan 2019 | 7th Jan 2019 | Will Stark | Day 3 | Ghandruk to Pokhara | Easy | A6 | Trek Down to Pokhara | Bus | Complete |
| UI32A | Ghandruk | 1st Jan 2019 | 7th Jan 2019 | Will Stark | Day 4 | Pokhara to Kathmandu | Moderate | A7 | Drive back to Kathmandu | Bus | Remaining |

### Un-normalized Form (UNF):

From the above table we can observe that the destination column has repeating groups in it, representing the given data in UNF:

UNF: (**Package\_ID**, Package\_Name, Start\_Date, End\_Date, Tour\_Guide, {Day, Travel\_Details, Difficulty\_Level, {Activity\_ID, Activity, Travel\_Mode, Status}})

### First Normal Form (1NF):

Package ID is taken as the Primary key as it gives

Package\_ID 🡺 Package\_Name, Start\_Date, End\_Date, Tour\_Guide,

Package\_ID, Day 🡺 Travel\_Details, Difficulty\_Level

Package\_ID, Day, Activity\_ID 🡺 Activity, Travel\_Mode, Status

Removing the repeating group and creating the Itenary\_Tour. Assigning Package\_ID as Primary key and creating the first normal form.

Tour (**Package\_ID**, Package\_Name, Start\_Date, End\_Date, Tour\_Guide)

Itenary\_Tour (**Package\_ID**\*, **Day**, Travel\_Details, Difficulty\_Level)

Day\_Activities (**Package\_ID**\*, **Day\***, **Activity\_ID**, Activity, Travel\_Mode, Status)

### Second Normal Form (2NF):

Here only Activity and Travel mode are in part dependent because it is not related to the Package ID and Package ID but directly dependent on the Activity ID.

Package\_ID 🡺 Package\_Name, Start\_Date, End\_Date, Tour\_Guide

Package\_ID, Day 🡺 Travel\_Details, Difficulty\_Level

Package\_ID 🡺 **X**

Day 🡺 **X**

Package\_ID, Day, Activity\_ID 🡺 Status

Activity\_ID 🡺 Travel\_Mode, Activities

Removing the partial dependency, we get:

Tour (**Package\_ID**, Package\_Name, Start\_Date, End\_Date, Tour\_Guide)

Itenary\_Tour (**Package\_ID**\*, **Day**, Travel\_Details, Difficulty\_Level, Status)

Day\_Activities (**Package\_ID**\*, **Day\***, **Activity\_ID**, Status)

Activities(**Activity\_ID**, Activity, Travel\_Mode)

### Third Normal Form (3NF):

There are no transitive dependencies as every none key entries are dependent on the primary key assigned for each table.

Tour (**Package\_ID**, Package\_Name, Start\_Date, End\_Date, Tour\_Guide)

Itenary\_Tour (**Package\_ID**\*, **Day**, Travel\_Details, Difficulty\_Level)

Day\_Activities (**Package\_ID**\*, **Day\***, **Activity\_ID\***, Status)

Activities(**Activity\_ID**, Activities, Travel\_Mode)

Since there are no transitive dependencies this is the final normalized form

## Integration

Combining the tables, we get the following entities. Also adding entities for staffs and customers.

Day\_activities (**Package\_Id\*, Day\*, Activity\_Id**\*, Status)

Activity (**Activity\_Id**, Activity\_Name, Travel\_Mode)

Itinerary (**Package\_Id\*, Day**, Difficulty\_Level, Travel\_Details)

Destination (**Package\_Id\***, **Destination**)

Packages (**Package\_Id\*,** Package\_Name, Total\_Days, Start\_Date, End\_Date, Difficulty)

Customer\_Package (**Customer\_Id\*, Package\_Id\***)

Package\_Guide (**Package\_Id\*, Staff\_Id\***)

Customer (**Customer\_Id**, First\_Name, Last\_Name Address, Phone\_No)

Staff (**Staff\_Id**, First\_Name, Last\_Name, Address, Phone\_No, Role\_Id)

Role (**Role\_Id**, Description)

# ER-Diagram

An Entity-relationship diagram represents the relationship between the entities in the database. ERD is one of the most common data but effective models where objects are divided into entities and their characteristics into attributes and entities are connected via elaborate relationships. (Nishadha, 2017). SQL Developer Data Modeler is a free graphical tool that improves productivity and simplifies data modelling tasks where users can create, browse and edit, logical, relational, physical, multi-dimensional, and data type models supporting collaborative development through integrated source code control (Oracle, 2020). SQL Developer Data Modeler was used to Create the final ER-Diagram.

Now, we have all the tables from figure 1 and 2 we can observe that the package table from figure 1 and tour table from figure 2 gives the same information. Now for integration the created tables into one and adding new tables for clarity we need to lay down some assumptions.

## Assumptions:

* Package from the figure(1) and tour from the figure(2) represent the same table.
* Difficulty in package table gives the difficulty in the overall package
* The difficulty in Itinerary gives the difficulty level in each day of the package.
* Staffs have Name (first name and last name), address and phone number.
* Staffs are identified by Staff ID.
* Guides are also staff and have guide ID which is referenced to the staff ID in a tour.
* Each staff have only one role assigned to them by the company
* The guide from the package is removed as a package might have multiple guides.
* While adding guides in the tour guide table, the user must only enter the staff who is a guide. (ie. The database will accept a receptionist as a tour guide but that is not ideal)
* Customers have Name (first name and last name), address and phone number.
* Staffs are identified by Staff ID
* Customers can only book one package only once at a time
* A Package can have no guide assigned; the user must avoid while entering data

## Final ER-diagram:

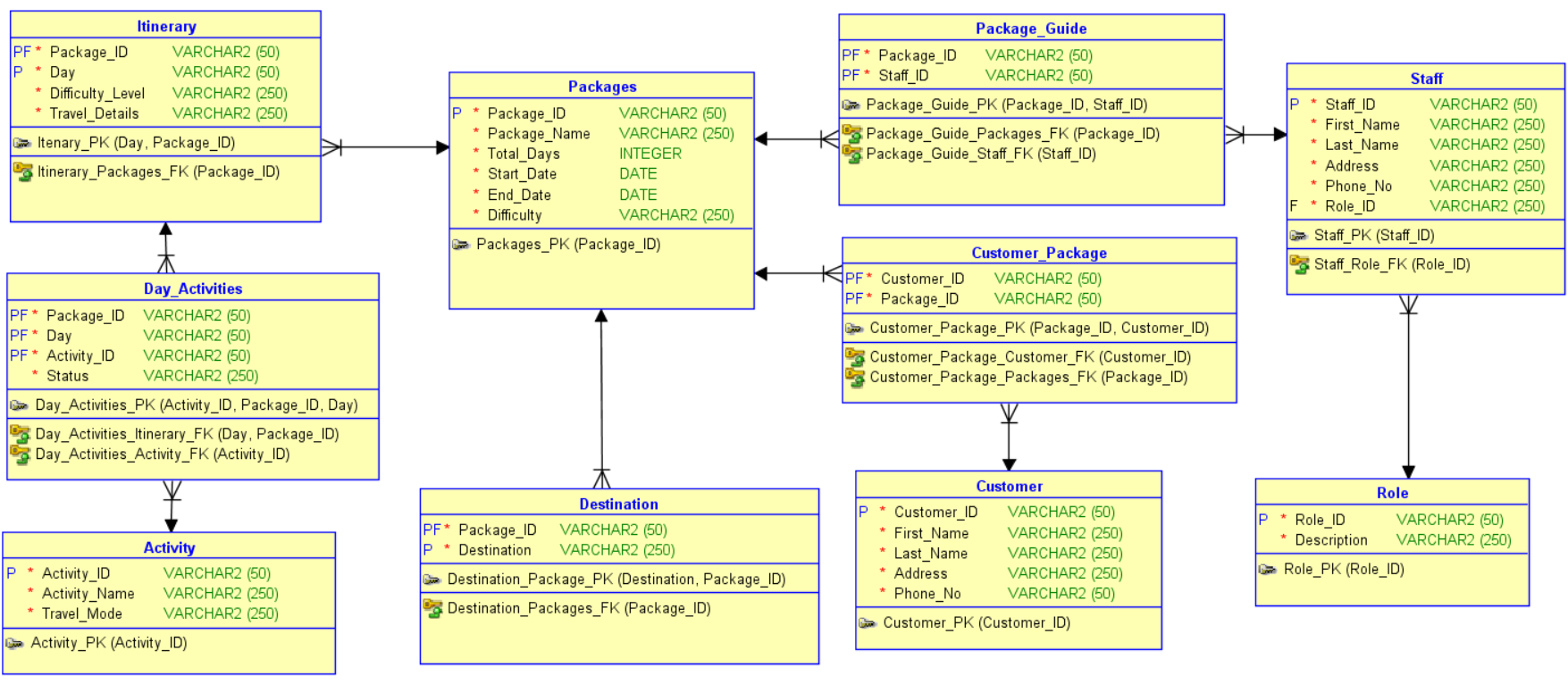


Figure : ER-Diagram created using the given entities

# Data Dictionary

Table : Data Dictionary for the Activity table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Activity\_ID | Y | Logical Type | VARCHAR | P |  | VARCHAR2 | 50 | “A1” |
| Activity\_Name | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Doing This” |
| Travel\_Mode | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Bus” |

Table : Data Dictionary for Day Activities table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Activity\_ID | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “A1” |
| Day | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “Day 1” |
| Package\_ID | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “GAND8” |
| Status | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Incomplete” |

Table : Data Dictionary for Itinerary Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Day | Y | Logical Type | VARCHAR | P |  | VARCHAR2 | 50 | “Day 1” |
| Difficulty\_Level | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Hard” |
| Package\_ID | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “GAND8” |
| Travel\_Details | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “KTM - PKR” |

Table : Data Dictionary for Destination Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Destination | Y | Logical Type | VARCHAR | P |  | VARCHAR2 | 250 | “Kathmandu” |
| Package\_ID | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “GAND8” |

Table : Data Dictionary for Package Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Difficulty | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Hard” |
| End\_Date | Y | Logical Type | Date |  |  | DATE |  | “1 JAN 2020” |
| Package\_ID | Y | Logical Type | VARCHAR | P |  | VARCHAR2 | 50 | “GAND8” |
| Package\_Name | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “ABC Trek” |
| Start\_Date | Y | Logical Type | Date |  |  | DATE |  | “1 JAN 2020” |
| Total\_Days | Y | Logical Type | Integer |  |  | INTEGER |  | 4 |

Table : Data Dictionary for Staff Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Address | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Jamal” |
| First\_Name | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Ram” |
| Last\_Name | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Karki” |
| Phone\_No | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | 213122323 |
| Role\_ID | Y | Logical Type | VARCHAR |  | F | VARCHAR2 | 250 | “G” |
| Staff\_ID | Y | Logical Type | VARCHAR | P |  | VARCHAR2 | 50 | “S110” |

Table : Data Dictionary for Customer Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Address | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Jamal” |
| Customer\_ID | Y | Logical Type | VARCHAR | P |  | VARCHAR2 | 50 | “C119” |
| First\_Name | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Ram” |
| Last\_Name | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Karki” |
| Phone\_No | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 50 | 213122323 |

Table : Data Dictionary for Customer-Package Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Customer\_ID | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “C119” |
| Package\_ID | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “GAND8” |

Table : Data Dictionary for Package Guide Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Package\_ID | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “GAND8” |
| Staff\_ID | Y | Logical Type | VARCHAR | P | F | VARCHAR2 | 50 | “S110” |

Table : Data Dictionary for role Table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column\_Name** | **Mandatory** | **DataType Kind** | **Logical Type Name** | **PK** | **FK** | **Native Type** | **T Size** | **Example** |
| Description | Y | Logical Type | VARCHAR |  |  | VARCHAR2 | 250 | “Does This” |
| Role\_ID | Y | Logical Type | VARCHAR | P |  | VARCHAR2 | 50 | “DT” |

# Generation of Database

## Create Statements

### Generating DDL Script and Creating Tables:

The Datamodeler allows the generation of DDL Script according to the designed ER-Model. The following figure shows how the script was created and the script is also included. Then the script is pasted into the SQL Developer and ran to generate the tables.

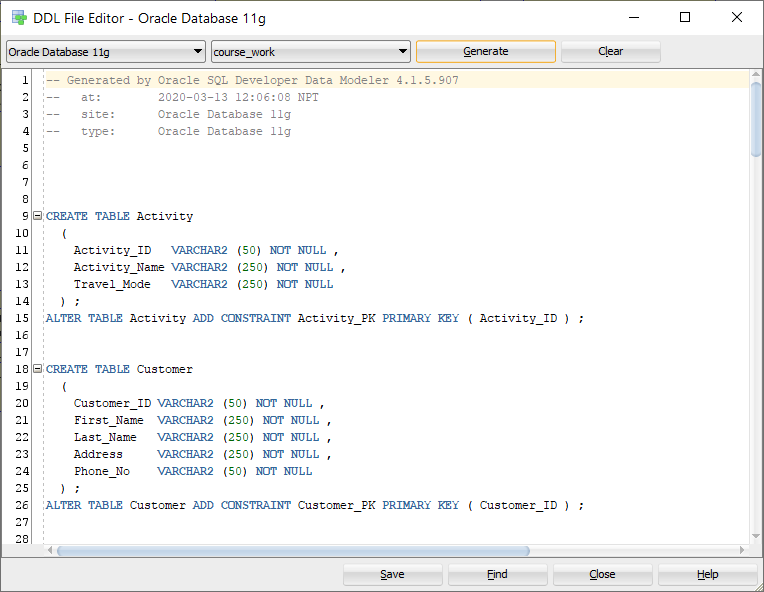


Figure : Process of generating the DDL Script via SQL developer Datamodeler

**DDL Script:**

CREATE TABLE Activity

  (

    Activity\_ID   VARCHAR2 (50) NOT NULL ,

    Activity\_Name VARCHAR2 (250) NOT NULL ,

    Travel\_Mode   VARCHAR2 (250) NOT NULL

  ) ;

ALTER TABLE Activity ADD CONSTRAINT Activity\_PK PRIMARY KEY ( Activity\_ID ) ;

CREATE TABLE Customer

  (

    Customer\_ID VARCHAR2 (50) NOT NULL ,

    First\_Name  VARCHAR2 (250) NOT NULL ,

    Last\_Name   VARCHAR2 (250) NOT NULL ,

    Address     VARCHAR2 (250) NOT NULL ,

    Phone\_No    VARCHAR2 (50) NOT NULL

  ) ;

ALTER TABLE Customer ADD CONSTRAINT Customer\_PK PRIMARY KEY ( Customer\_ID ) ;

CREATE TABLE Customer\_Package

  (

    Customer\_ID VARCHAR2 (50) NOT NULL ,

    Package\_ID  VARCHAR2 (50) NOT NULL

  ) ;

ALTER TABLE Customer\_Package ADD CONSTRAINT Customer\_Package\_PK PRIMARY KEY ( Package\_ID, Customer\_ID ) ;

CREATE TABLE Day\_Activities

  (

    Package\_ID  VARCHAR2 (50) NOT NULL ,

    DAY         VARCHAR2 (50) NOT NULL ,

    Activity\_ID VARCHAR2 (50) NOT NULL ,

    Status      VARCHAR2 (250) NOT NULL

  ) ;

ALTER TABLE Day\_Activities ADD CONSTRAINT Day\_Activities\_PK PRIMARY KEY ( Activity\_ID, Package\_ID, DAY ) ;

CREATE TABLE Destination

  (

    Package\_ID  VARCHAR2 (50) NOT NULL ,

    Destination VARCHAR2 (250) NOT NULL

  ) ;

ALTER TABLE Destination ADD CONSTRAINT Destination\_Package\_PK PRIMARY KEY ( Destination, Package\_ID ) ;

CREATE TABLE Itinerary

  (

    Package\_ID       VARCHAR2 (50) NOT NULL ,

    DAY              VARCHAR2 (50) NOT NULL ,

    Difficulty\_Level VARCHAR2 (250) NOT NULL ,

    Travel\_Details   VARCHAR2 (250) NOT NULL

  ) ;

ALTER TABLE Itinerary ADD CONSTRAINT Itenary\_PK PRIMARY KEY ( DAY, Package\_ID ) ;

CREATE TABLE Package\_Guide

  (

    Package\_ID VARCHAR2 (50) NOT NULL ,

    Staff\_ID   VARCHAR2 (50) NOT NULL

  ) ;

ALTER TABLE Package\_Guide ADD CONSTRAINT Package\_Guide\_PK PRIMARY KEY ( Package\_ID, Staff\_ID ) ;

CREATE TABLE Packages

  (

    Package\_ID   VARCHAR2 (50) NOT NULL ,

    Package\_Name VARCHAR2 (250) NOT NULL ,

    Total\_Days   INTEGER NOT NULL ,

    Start\_Date   DATE NOT NULL ,

    End\_Date     DATE NOT NULL ,

    Difficulty   VARCHAR2 (250) NOT NULL

  ) ;

ALTER TABLE Packages ADD CONSTRAINT Packages\_PK PRIMARY KEY ( Package\_ID ) ;

CREATE TABLE Role

  (

    Role\_ID     VARCHAR2 (50) NOT NULL ,

    Description VARCHAR2 (250) NOT NULL

  ) ;

ALTER TABLE Role ADD CONSTRAINT Role\_PK PRIMARY KEY ( Role\_ID ) ;

CREATE TABLE Staff

  (

    Staff\_ID   VARCHAR2 (50) NOT NULL ,

    First\_Name VARCHAR2 (250) NOT NULL ,

    Last\_Name  VARCHAR2 (250) NOT NULL ,

    Address    VARCHAR2 (250) NOT NULL ,

    Phone\_No   VARCHAR2 (250) NOT NULL ,

    Role\_ID    VARCHAR2 (250) NOT NULL

  ) ;

ALTER TABLE Staff ADD CONSTRAINT Staff\_PK PRIMARY KEY ( Staff\_ID ) ;

ALTER TABLE Customer\_Package ADD CONSTRAINT Customer\_Package\_Customer\_FK FOREIGN KEY ( Customer\_ID ) REFERENCES Customer ( Customer\_ID ) ;

ALTER TABLE Customer\_Package ADD CONSTRAINT Customer\_Package\_Packages\_FK FOREIGN KEY ( Package\_ID ) REFERENCES Packages ( Package\_ID ) ;

ALTER TABLE Day\_Activities ADD CONSTRAINT Day\_Activities\_Activity\_FK FOREIGN KEY ( Activity\_ID ) REFERENCES Activity ( Activity\_ID ) ;

ALTER TABLE Day\_Activities ADD CONSTRAINT Day\_Activities\_Itinerary\_FK FOREIGN KEY ( DAY, Package\_ID ) REFERENCES Itinerary ( DAY, Package\_ID ) ;

ALTER TABLE Destination ADD CONSTRAINT Destination\_Packages\_FK FOREIGN KEY ( Package\_ID ) REFERENCES Packages ( Package\_ID ) ;

ALTER TABLE Itinerary ADD CONSTRAINT Itinerary\_Packages\_FK FOREIGN KEY ( Package\_ID ) REFERENCES Packages ( Package\_ID ) ;

ALTER TABLE Package\_Guide ADD CONSTRAINT Package\_Guide\_Packages\_FK FOREIGN KEY ( Package\_ID ) REFERENCES Packages ( Package\_ID ) ;

ALTER TABLE Package\_Guide ADD CONSTRAINT Package\_Guide\_Staff\_FK FOREIGN KEY ( Staff\_ID ) REFERENCES Staff ( Staff\_ID ) ;

ALTER TABLE Staff ADD CONSTRAINT Staff\_Role\_FK FOREIGN KEY ( Role\_ID ) REFERENCES Role ( Role\_ID ) ;

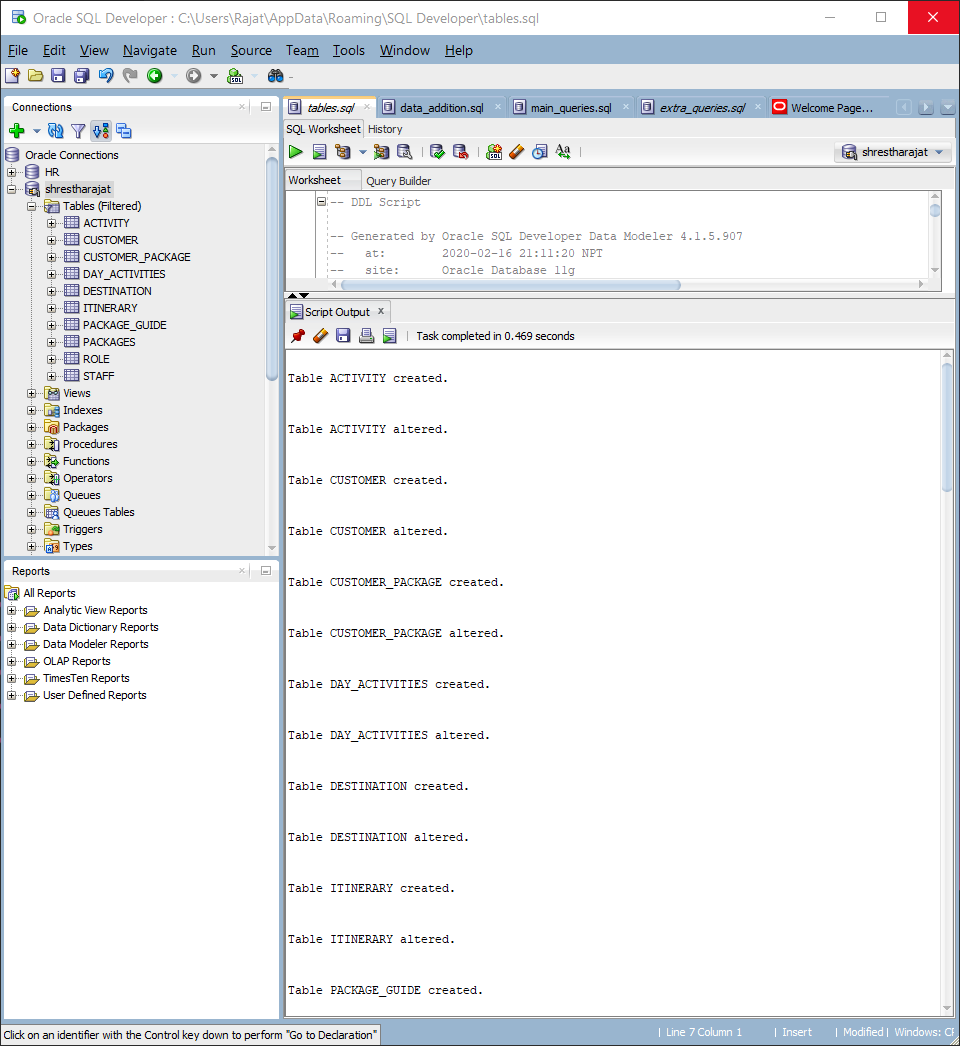


Figure : Running the DDL Script in SQL Developer

## Insert Statements

After creating the tables, they need to be populated. The following data were inserted into the database for testing the database via sql developer

-- Populating the database

INSERT INTO packages (PACKAGE\_ID, PACKAGE\_NAME, TOTAL\_DAYS, START\_DATE, END\_DATE, DIFFICULTY)

    WITH names AS (

        SELECT 'GAND8', 'Ghandruk', 4, '01-JAN-2020',

'04-JAN-2020', 'Moderate' FROM dual UNION ALL

        SELECT 'ABC21', 'Annapurna BC', 6, '12-FEB-2020',

'17-FEB-2020', 'Hard' FROM dual UNION ALL

        SELECT 'PUNH1', 'Poon-Hill', 5, '01-JAN-2020',

'05-JAN-2020', 'Hard' FROM dual UNION ALL

        SELECT 'EBC77', 'Everest BC', 8, '01-JAN-2020',

'08-JAN-2020', 'Hard' FROM dual UNION ALL

        SELECT 'LUKL1', 'Lukla', 3, '01-MAR-2020',

'03-MAR-2020', 'Moderate' FROM dual

    )

    SELECT \* FROM names;

INSERT INTO destination (PACKAGE\_ID, DESTINATION)

    WITH names AS (

        SELECT 'GAND8', 'Pokhara' FROM dual UNION ALL

        SELECT 'GAND8', 'Ghandruk' FROM dual UNION ALL

        SELECT 'ABC21', 'Pokhara' FROM dual UNION ALL

        SELECT 'ABC21', 'Ghandruk' FROM dual UNION ALL

        SELECT 'ABC21', 'ABC' FROM dual UNION ALL

        SELECT 'PUNH1', 'Pokhara' FROM dual UNION ALL

        SELECT 'PUNH1', 'Ghorepani' FROM dual UNION ALL

        SELECT 'PUNH1', 'PoonHill' FROM dual UNION ALL

        SELECT 'PUNH1', 'Ghandruk' FROM dual UNION ALL

        SELECT 'LUKL1', 'Lukla' FROM dual UNION ALL

        SELECT 'EBC77', 'Lukla' FROM dual UNION ALL

        SELECT 'EBC77', 'Namche-Bazzar' FROM dual UNION ALL

        SELECT 'EBC77', 'Dingboche' FROM dual UNION ALL

        SELECT 'EBC77', 'EBC' FROM dual

        )

    SELECT \* FROM names;

INSERT INTO itinerary (PACKAGE\_ID, DAY, TRAVEL\_DETAILS, DIFFICULTY\_LEVEL)

    WITH names AS (

        SELECT 'GAND8', 'day-1', 'Kathmandu-Pokhara', 'Easy' FROM dual UNION ALL

        SELECT 'GAND8', 'day-2', 'Pokhara-Ghandruk', 'Hard' FROM dual UNION ALL

        SELECT 'GAND8', 'day-3', 'Ghandruk-Pokhara', 'Hard' FROM dual UNION ALL

        SELECT 'GAND8', 'day-4', 'Pokhara-Kathmandu', 'Easy' FROM dual UNION ALL

        SELECT 'ABC21', 'day-1', 'Kathmandu-Pokhara', 'Easy' FROM dual UNION ALL

        SELECT 'ABC21', 'day-2', 'Pokhara-Ghandruk', 'Hard' FROM dual UNION ALL

        SELECT 'ABC21', 'day-3', 'Ghandruk-ABC', 'Hard' FROM dual UNION ALL

        SELECT 'ABC21', 'day-4', 'ABC-Ghandruk', 'Hard' FROM dual UNION ALL

        SELECT 'ABC21', 'day-5', 'Ghandruk-Pokhara', 'Hard' FROM dual UNION ALL

        SELECT 'ABC21', 'day-6', 'Pokhara-Kathmandu', 'Easy' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-1', 'Kathmandu-Pokhara', 'Easy' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-2', 'Pokhara-Ghorepani', 'Hard' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-3', 'Gorepani-Ghandruk', 'Hard' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-4', 'Ghandruk-Pokhara', 'Hard' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-5', 'Pokhara-Kathmandu', 'Easy' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-1', 'Kathmandu-Lukla', 'Easy' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-2', 'Lukla', 'Easy' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-3', 'Lukla-Kathmandu', 'Easy' FROM dual UNION ALL

        SELECT 'EBC77', 'day-1', 'Kathmandu-Lukla', 'Easy' FROM dual UNION ALL

        SELECT 'EBC77', 'day-2', 'Lukla-Namche Bazzar', 'Hard' FROM dual UNION ALL

        SELECT 'EBC77', 'day-3',

'Namche Bazzar-Dingboche', 'Hard' FROM dual UNION ALL

        SELECT 'EBC77', 'day-4', 'Dingboche-EBC', 'Hard' FROM dual UNION ALL

        SELECT 'EBC77', 'day-5', 'EBC-Dingboche', 'Hard' FROM dual UNION ALL

        SELECT 'EBC77', 'day-6',

'Dingboche-Namche Bazzar', 'Hard' FROM dual UNION ALL

        SELECT 'EBC77', 'day-7', 'Namche Bazzar-Lukla', 'Easy' FROM dual UNION ALL

        SELECT 'EBC77', 'day-8', 'Lukla-Kathmandu', 'Easy' FROM dual

    )

    SELECT \* FROM names;

INSERT INTO activity (ACTIVITY\_ID, ACTIVITY\_NAME, TRAVEL\_MODE)

    WITH names AS (

        SELECT 'A1', 'Driving from KTM to Pokhara', 'bus' FROM dual UNION ALL

        SELECT 'A2', 'Overnight stay in Hotel', 'stay' FROM dual UNION ALL

        SELECT 'A3', 'Trek to Ghandruk', 'walk' FROM dual UNION ALL

        SELECT 'A4', 'Explore the Ghandruk Village', 'walk' FROM dual UNION ALL

        SELECT 'A5', 'View the sunrise and Himalayas', 'walk' FROM dual UNION ALL

        SELECT 'A6', 'Trek Down to Pokhara', 'walk' FROM dual UNION ALL

        SELECT 'A7', 'Drive back to Kathmandu', 'walk' FROM dual UNION ALL

        SELECT 'A8', 'Trek to ABC from Ghandruk', 'walk' FROM dual UNION ALL

        SELECT 'A9', 'Explore ABC', 'walk' FROM dual UNION ALL

        SELECT 'A10', 'Trek back to Ghandruk from ABC', 'walk' FROM dual UNION ALL

        SELECT 'A11', 'Bus to halfway for ghorepani', 'walk' FROM dual UNION ALL

        SELECT 'A12', 'Trek to Ghorepani', 'walk' FROM dual UNION ALL

        SELECT 'A13', 'Hike to Poonhill', 'walk' FROM dual UNION ALL

        SELECT 'A14', 'Viewing Sunrise', 'walk' FROM dual UNION ALL

        SELECT 'A15', 'Flight to Lukla', 'plane' FROM dual UNION ALL

        SELECT 'A16', 'Explore Lukla', 'walk' FROM dual UNION ALL

        SELECT 'A17', 'Overnight stay in Lukla', 'stay' FROM dual UNION ALL

        SELECT 'A18', 'Flight from Lukla to kathmandu', 'plane' FROM dual UNION ALL

        SELECT 'A19', 'Hike from Lukla to Namche bazzar', 'walk' FROM dual UNION ALL

        SELECT 'A20', 'Trek from Namche to Dingboche', 'walk' FROM dual UNION ALL

        SELECT 'A21', 'Trek from Dingboche to EBC', 'walk' FROM dual UNION ALL

        SELECT 'A22', 'Explore EBC', 'walk' FROM dual UNION ALL

        SELECT 'A23', 'Trek from EBC to Dingboche', 'walk' FROM dual UNION ALL

        SELECT 'A24', 'Trek from Dingboche to Namche', 'walk' FROM dual UNION ALL

        SELECT 'A25', 'Trek from Namche bazzar to Lukla', 'walk' FROM dual

    )

    SELECT \* FROM names;

INSERT INTO day\_activities (PACKAGE\_ID, DAY, ACTIVITY\_ID, STATUS)

    WITH names AS (

        SELECT 'GAND8', 'day-1', 'A1', 'Incomplete' FROM dual UNION ALL

        SELECT 'GAND8', 'day-1', 'A2', 'Incomplete' FROM dual UNION ALL

        SELECT 'GAND8', 'day-2', 'A3', 'Incomplete' FROM dual UNION ALL

        SELECT 'GAND8', 'day-2', 'A4', 'Incomplete' FROM dual UNION ALL

        SELECT 'GAND8', 'day-3', 'A5', 'Incomplete' FROM dual UNION ALL

        SELECT 'GAND8', 'day-4', 'A6', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-1', 'A1', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-1', 'A2', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-2', 'A3', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-2', 'A4', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-3', 'A8', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-3', 'A9', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-4', 'A10', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-5', 'A5', 'Incomplete' FROM dual UNION ALL

        SELECT 'ABC21', 'day-6', 'A6', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-1', 'A1', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-1', 'A2', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-2', 'A11', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-2', 'A12', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-3', 'A13', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-3', 'A14', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-3', 'A3', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-4', 'A5', 'Incomplete' FROM dual UNION ALL

        SELECT 'PUNH1', 'day-5', 'A6', 'Incomplete' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-1' , 'A15', 'Incomplete' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-1' , 'A16', 'Incomplete' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-1' , 'A17', 'Incomplete' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-2' , 'A16', 'Incomplete' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-2' , 'A17', 'Incomplete' FROM dual UNION ALL

        SELECT 'LUKL1', 'day-3' , 'A18', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-1' , 'A15', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-1' , 'A16', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-1' , 'A17', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-2' , 'A19', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-3' , 'A20', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-4' , 'A21', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-4' , 'A22', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-5' , 'A23', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-6' , 'A24', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-7' , 'A25', 'Incomplete' FROM dual UNION ALL

        SELECT 'EBC77', 'day-8' , 'A18', 'Incomplete' FROM dual

    )

    SELECT \* FROM names;

INSERT INTO role (ROLE\_ID, DESCRIPTION)

    WITH names AS (

        SELECT 'TA', 'Tour Agent, helps customer' FROM dual UNION ALL

        SELECT 'G', 'Guide, goes to tours' FROM dual UNION ALL

        SELECT 'MA', 'Manager, the main manager of the office' FROM dual UNION ALL

        SELECT 'DV', 'Driver, driver for the company' FROM dual UNION ALL

        SELECT 'IN', 'Intern, agent intern in the company' FROM dual

    )

    SELECT \* FROM names;

INSERT INTO staff (STAFF\_ID, FIRST\_NAME, LAST\_NAME, ADDRESS, PHONE\_NO, ROLE\_ID)

    WITH names AS (

        SELECT 'S001', 'Rajat', 'Shrestha', 'Samakhushi, TownPlanning',

'9182934234', 'G' FROM dual UNION ALL

        SELECT 'S002', 'Ashley', 'Green', 'Baluwatar',

'9182343123', 'TA' FROM dual UNION ALL

        SELECT 'S003', 'Shia', 'Miton', 'patan',

'9182323423', 'TA' FROM dual UNION ALL

        SELECT 'S004', 'Ramendra', 'Sharma', 'Gongabu',

'9184322123', 'G' FROM dual UNION ALL

        SELECT 'S005', 'Silas', 'BK', 'Butwal',

'9182931112', 'G' FROM dual UNION ALL

        SELECT 'S006', 'Rabin', 'Tamang', 'Sankhamul',

'91323934234', 'G' FROM dual UNION ALL

        SELECT 'S007', 'Prabin', 'Maskey', 'Baluwatar',

'9182343123', 'G' FROM dual UNION ALL

        SELECT 'S008', 'Pranish', 'Chettri', 'baneshwor',

'9132323423', 'IN' FROM dual UNION ALL

        SELECT 'S009', 'Ramu', 'Sharma', 'Gongabu',

'9184322123', 'G' FROM dual UNION ALL

        SELECT 'S010', 'Ganesh', 'KC', 'Birendranagar',

'9182931112', 'MA' FROM dual

    )

    SELECT \* FROM names;

INSERT INTO customer (CUSTOMER\_ID, FIRST\_NAME, LAST\_NAME, ADDRESS, PHONE\_NO)

    WITH names AS (

        SELECT 'C001', 'Ramesh', 'Nakarmi', 'Samakhushi, TownPlanning', '9182934234' FROM dual UNION ALL

    SELECT 'C002', 'John', 'LeBron', 'Baluwatar', '9182343423'  FROM dual UNION ALL

    SELECT 'C003', 'Saran', 'Manandar', 'patan', '9182323433'  FROM dual UNION ALL

    SELECT 'C004', 'Susma', 'Shakya', 'Gongabu', '9184422123' FROM dual UNION ALL

    SELECT 'C005', 'Siliya', 'Malla', 'Butwal', '9182931111'  FROM dual UNION AL       SELECT 'C006', 'Shreyash', 'Subedi', 'Sankhamul', '91224234' FROM dual UNION ALL

    SELECT 'C007', 'Bhuwan', 'Khanal', 'Baluwatar', '9382343123' FROM dual UNION ALL

    SELECT 'C008', 'Prashant', 'Budha', 'baneshwor', '933223423' FROM dual UNION ALL

    SELECT 'C009', 'Gopal', 'KC', 'Gongabu', '9184323223' FROM dual UNION ALL

    SELECT 'C010', 'Naren', 'Bajracharya', 'Birendranagar', '9123931112'  FROM dual

    )

    SELECT \* FROM names;

INSERT INTO customer\_package (CUSTOMER\_ID, PACKAGE\_ID)

    WITH names AS (

        SELECT 'C001', 'GAND8' FROM dual UNION ALL

        SELECT 'C002', 'EBC77' FROM dual UNION ALL

        SELECT 'C003', 'EBC77' FROM dual UNION ALL

        SELECT 'C004', 'EBC77' FROM dual UNION ALL

        SELECT 'C005', 'PUNH1' FROM dual UNION ALL

        SELECT 'C006', 'ABC21' FROM dual UNION ALL

        SELECT 'C007', 'PUNH1' FROM dual UNION ALL

        SELECT 'C008', 'ABC21' FROM dual UNION ALL

        SELECT 'C009', 'LUKL1' FROM dual UNION ALL

        SELECT 'C010', 'GAND8' FROM dual

    )

    SELECT \* FROM names;

INSERT INTO package\_guide (PACKAGE\_ID, STAFF\_ID)

    WITH names AS (

        SELECT 'GAND8', 'S001' FROM dual UNION ALL

        SELECT 'ABC21', 'S004' FROM dual UNION ALL

        SELECT 'ABC21', 'S005' FROM dual UNION ALL

        SELECT 'PUNH1', 'S006' FROM dual UNION ALL

        SELECT 'EBC77', 'S007' FROM dual UNION ALL

        SELECT 'EBC77', 'S009' FROM dual

    )

    SELECT \* FROM names;

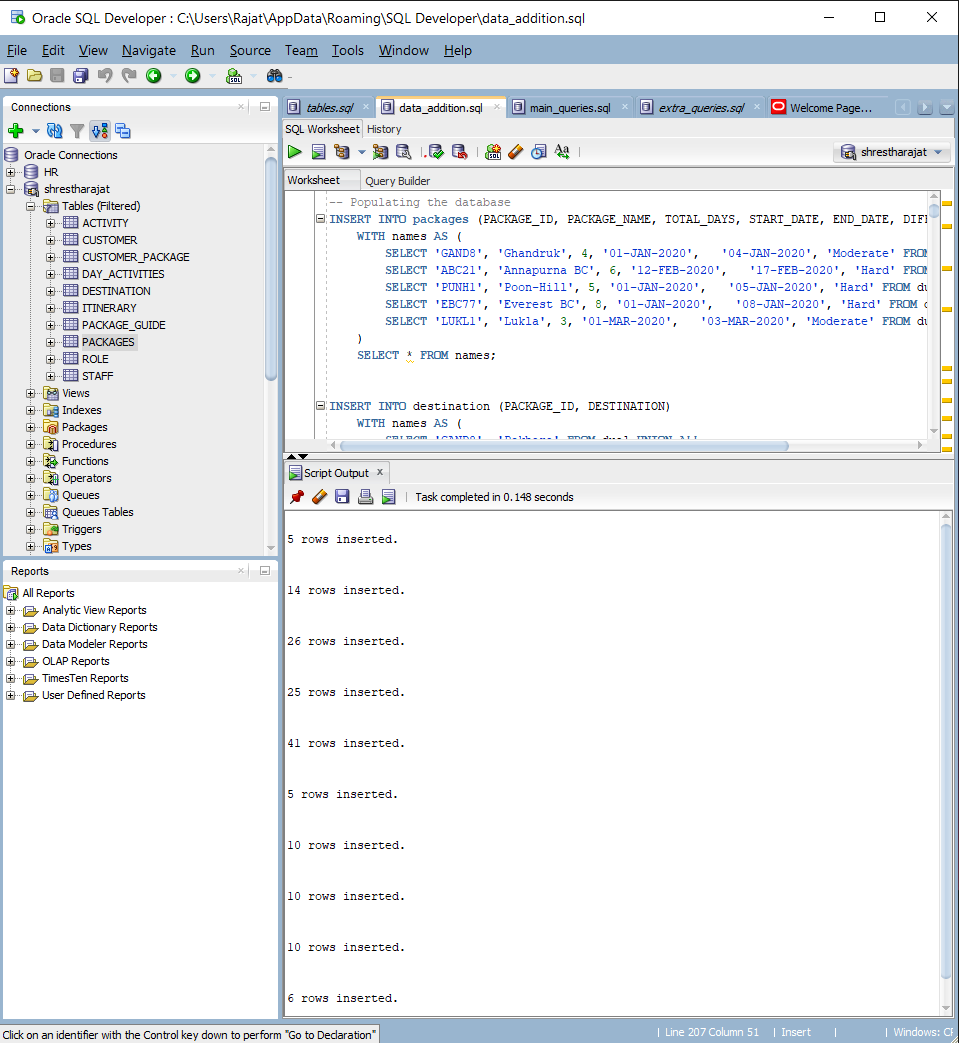


Figure : Running the Insert Statements to populate the database in SQL Developer

## Select Statements

The data inserted are checked using the select statement for each table.

### Day Activities

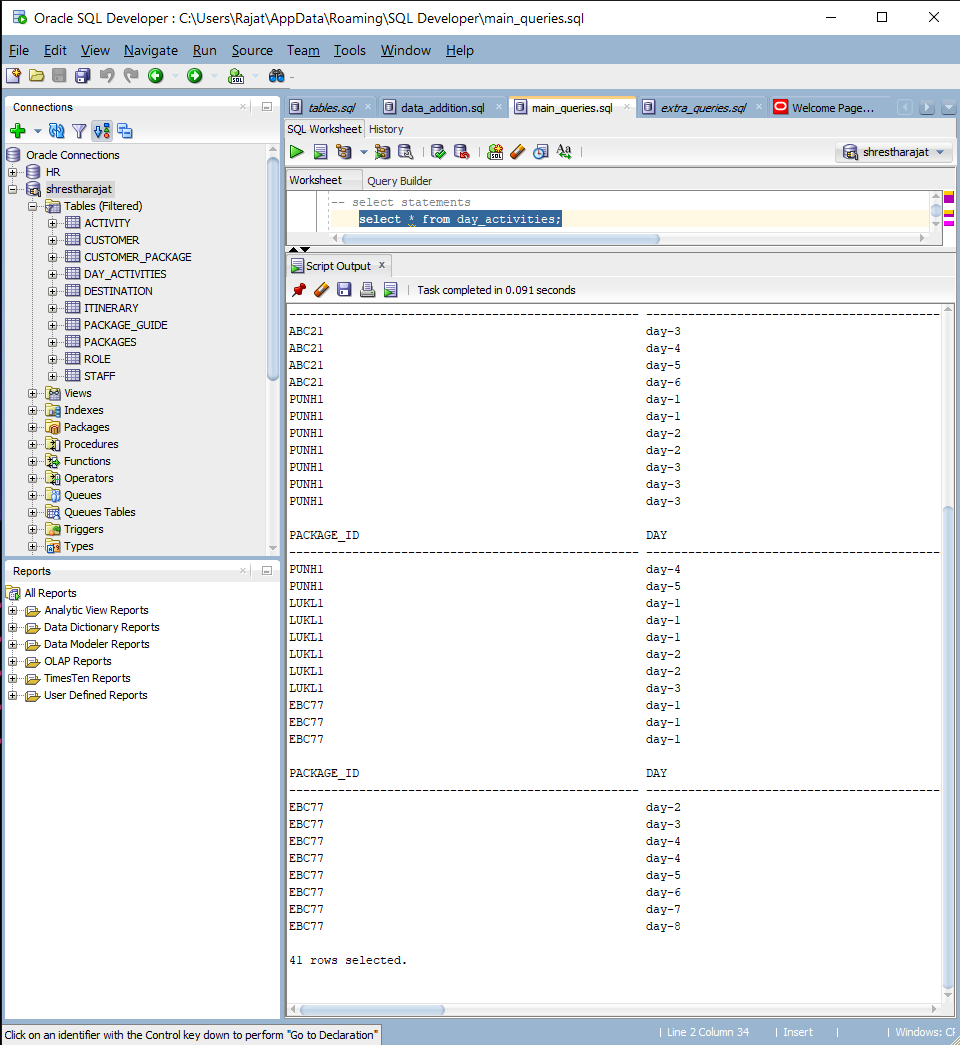


Figure : Running select Statement on Day Activities Table

### Activity

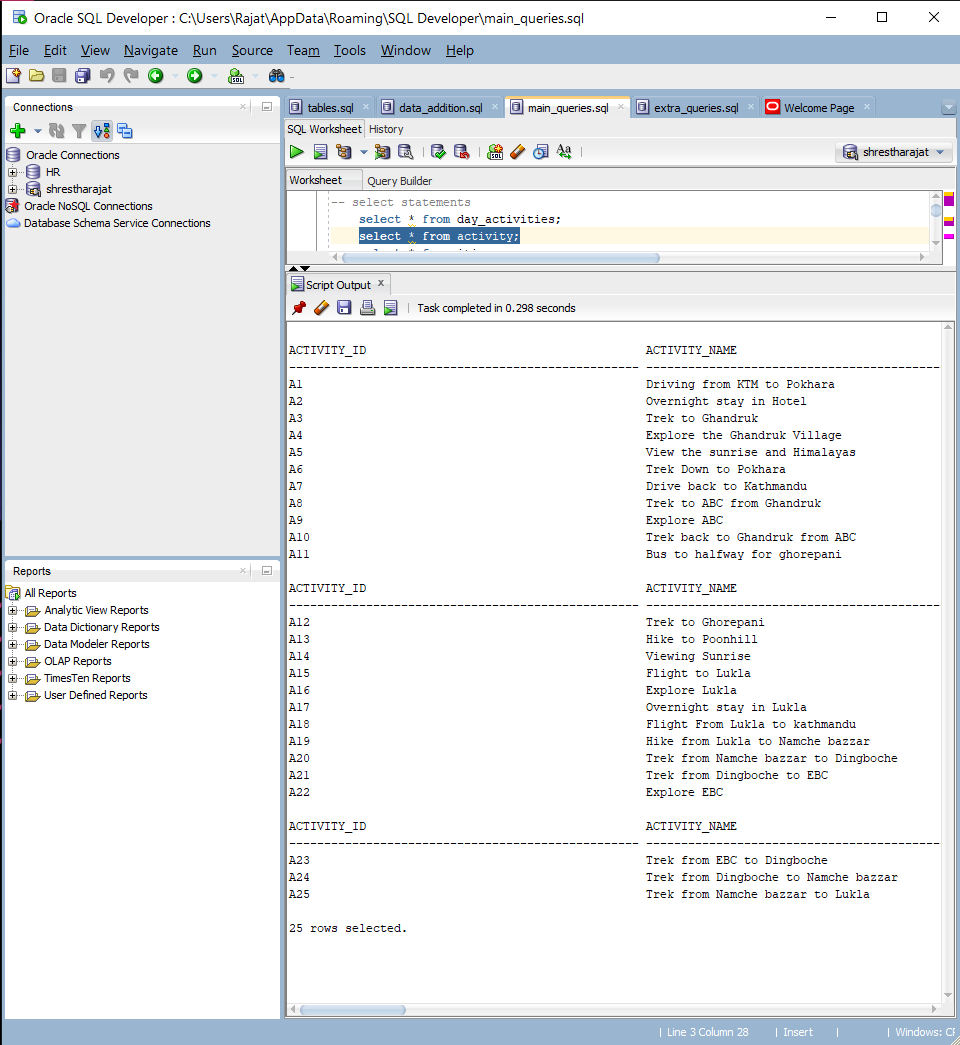


Figure : Running select Statement on Activity Table

### Itinerary

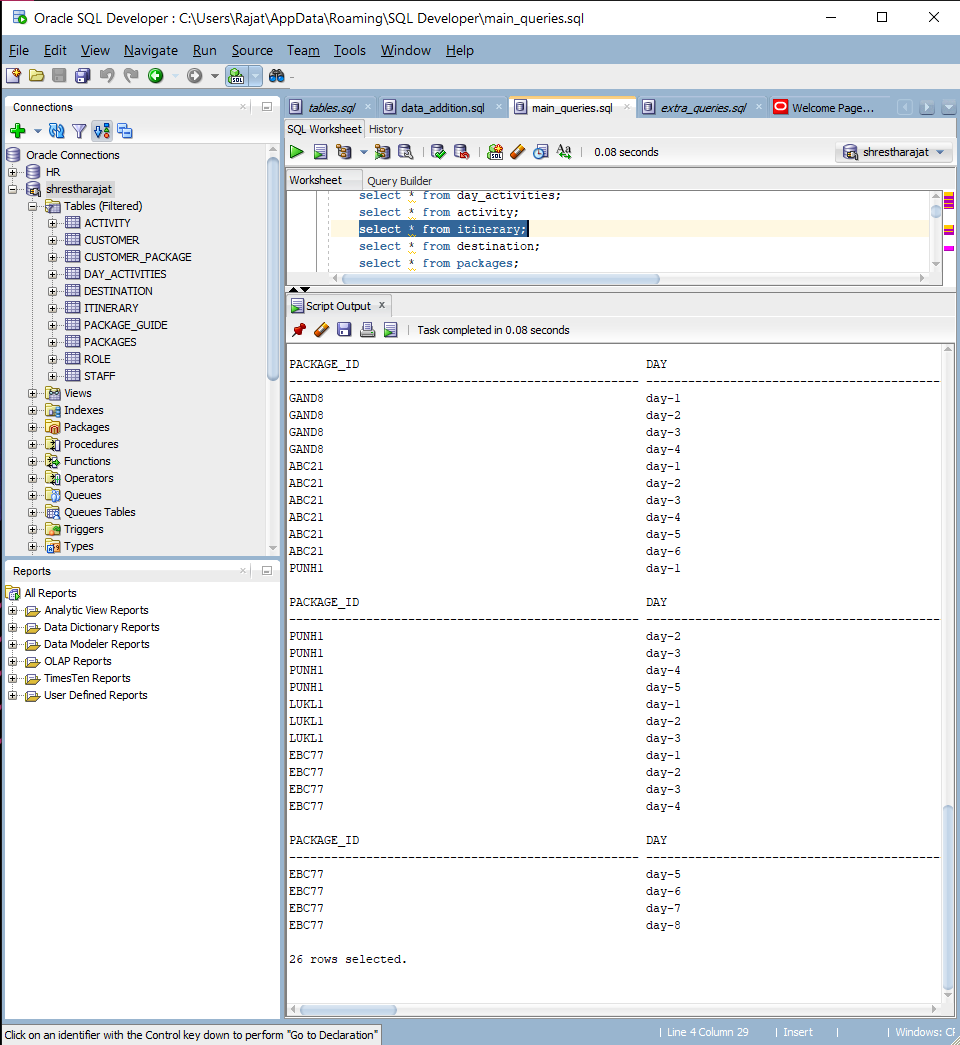


Figure : Running select Statement on Itinerary Table

### Destination

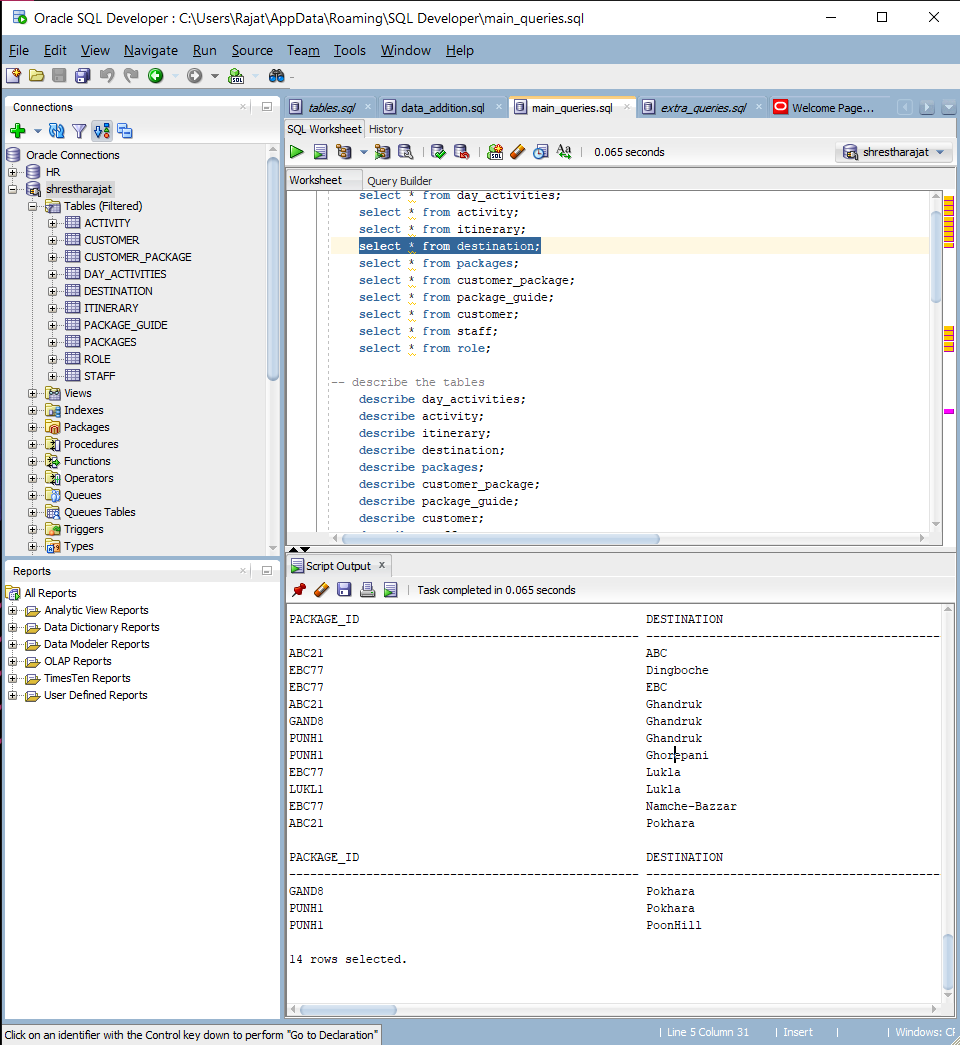
****

Figure : Running select Statement on Destination Table

### Packages

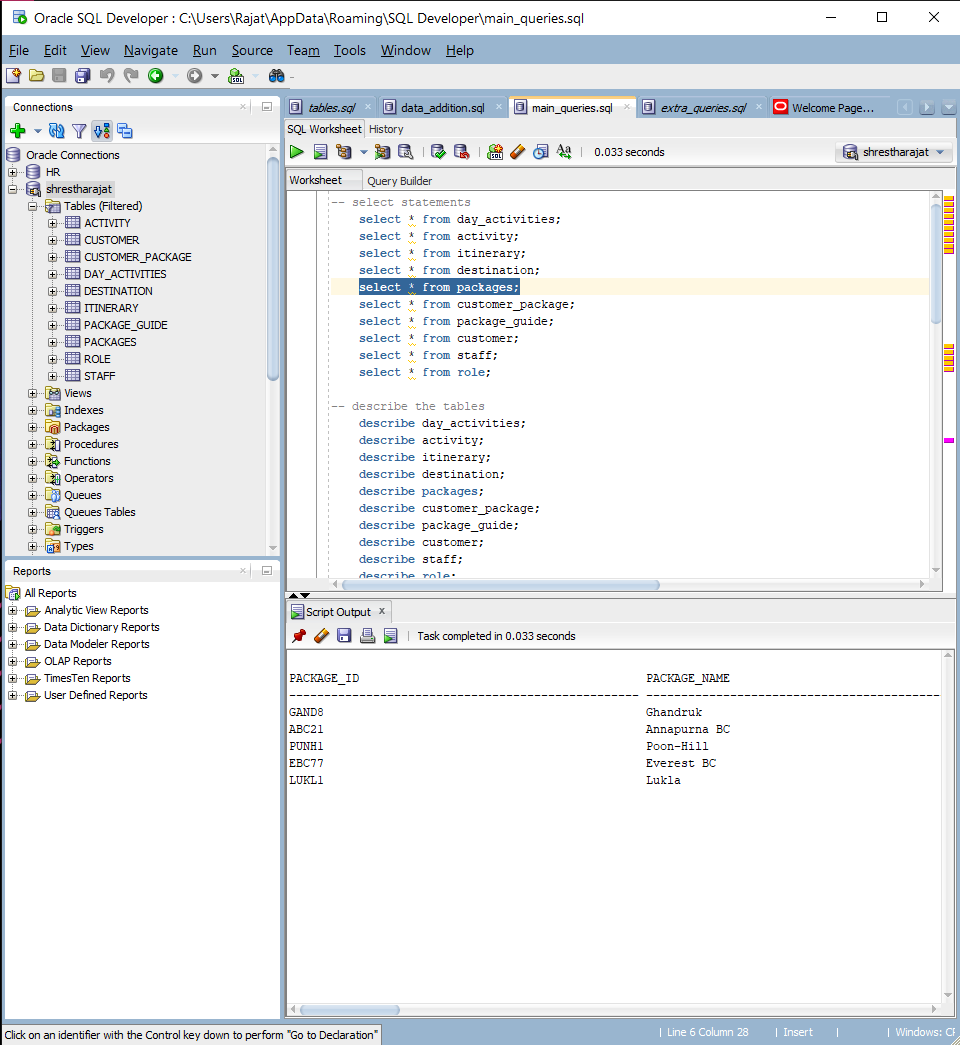


Figure : Running select Statement on Packages Table

### Customer Packages

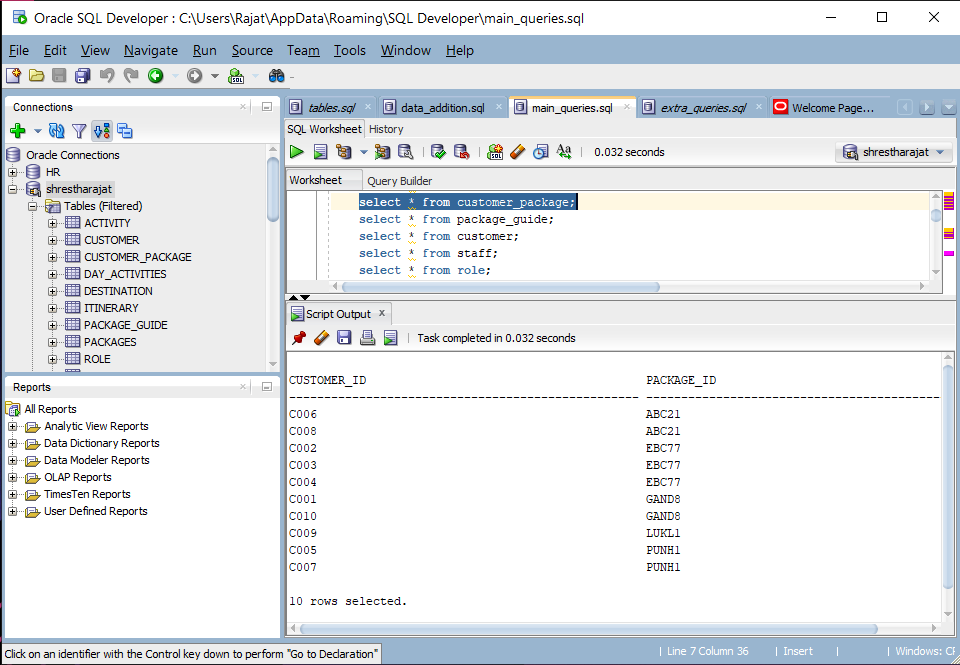


Figure : Running select Statement on Customer packages Table

### Package Guide

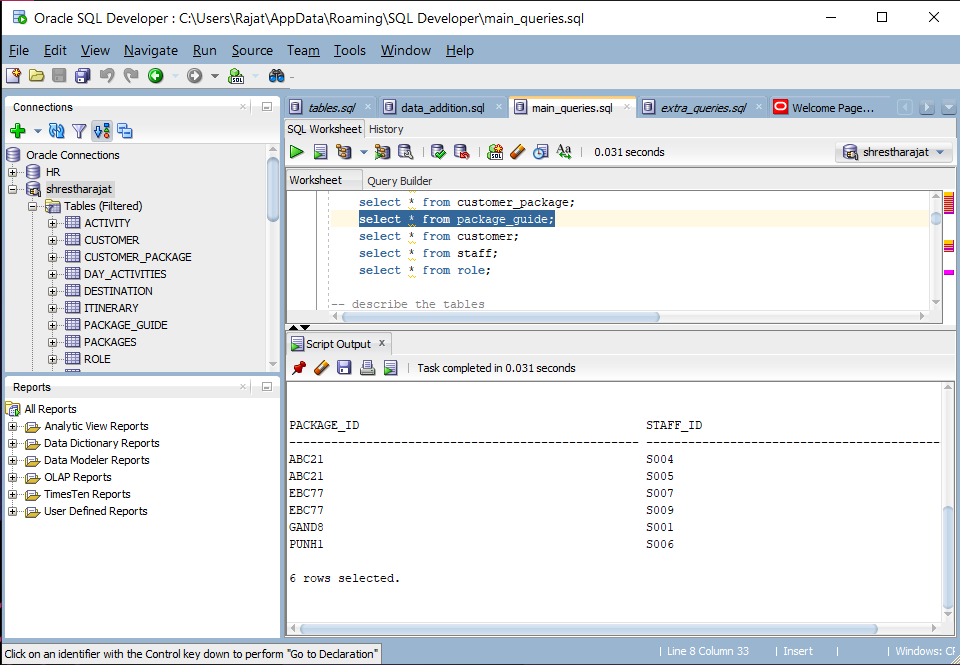


Figure : Running select Statement on Package Guide Table

### Customer

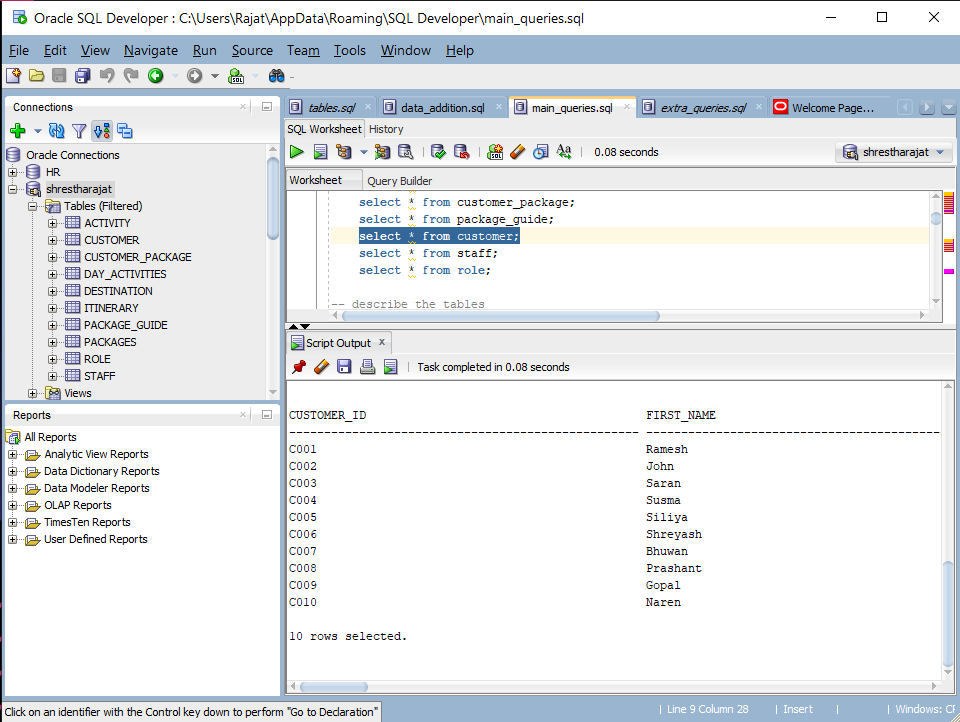


Figure : Running select Statement on Customer Table

### Staff

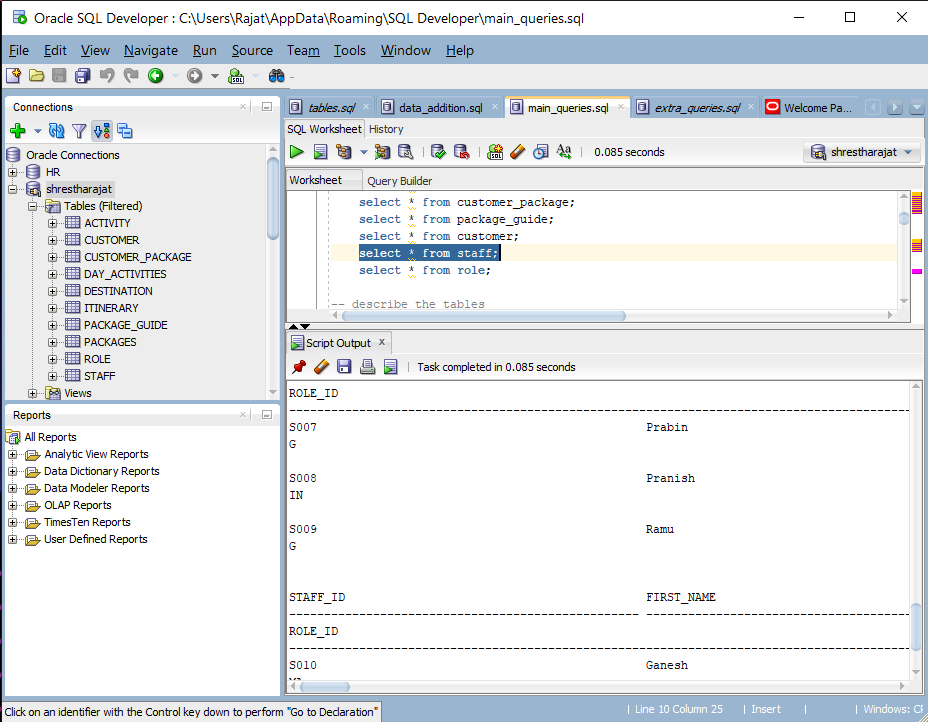


Figure : Running select Statement on Staff Table

### Role

****

Figure : Running select Statement on Role Table

# Implementation of Web-Based Database Application

## 6.1 Basic Webforms

These forms provide basic functionality for adding, updating or deleting any entries. The Add Delete and Edit Buttons on each Form Provide CRUD functionality for each basic form and the steps for each of the operation is detailed in the user manual.

### Staff Details



Figure : Simple Web-forms for Staff Details

### Designation Details

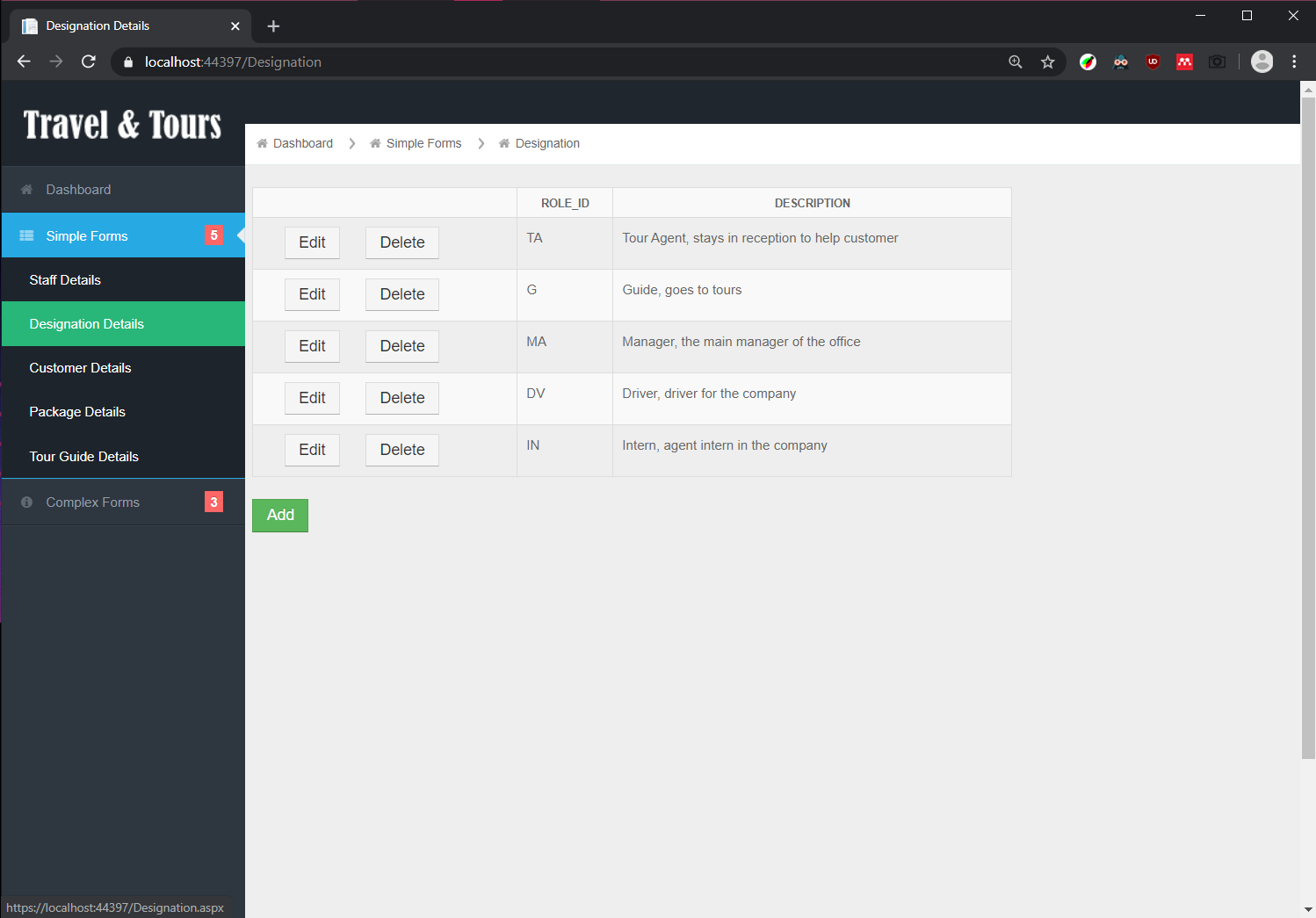


Figure : Simple Web-forms for Designation Details

### Customer Details

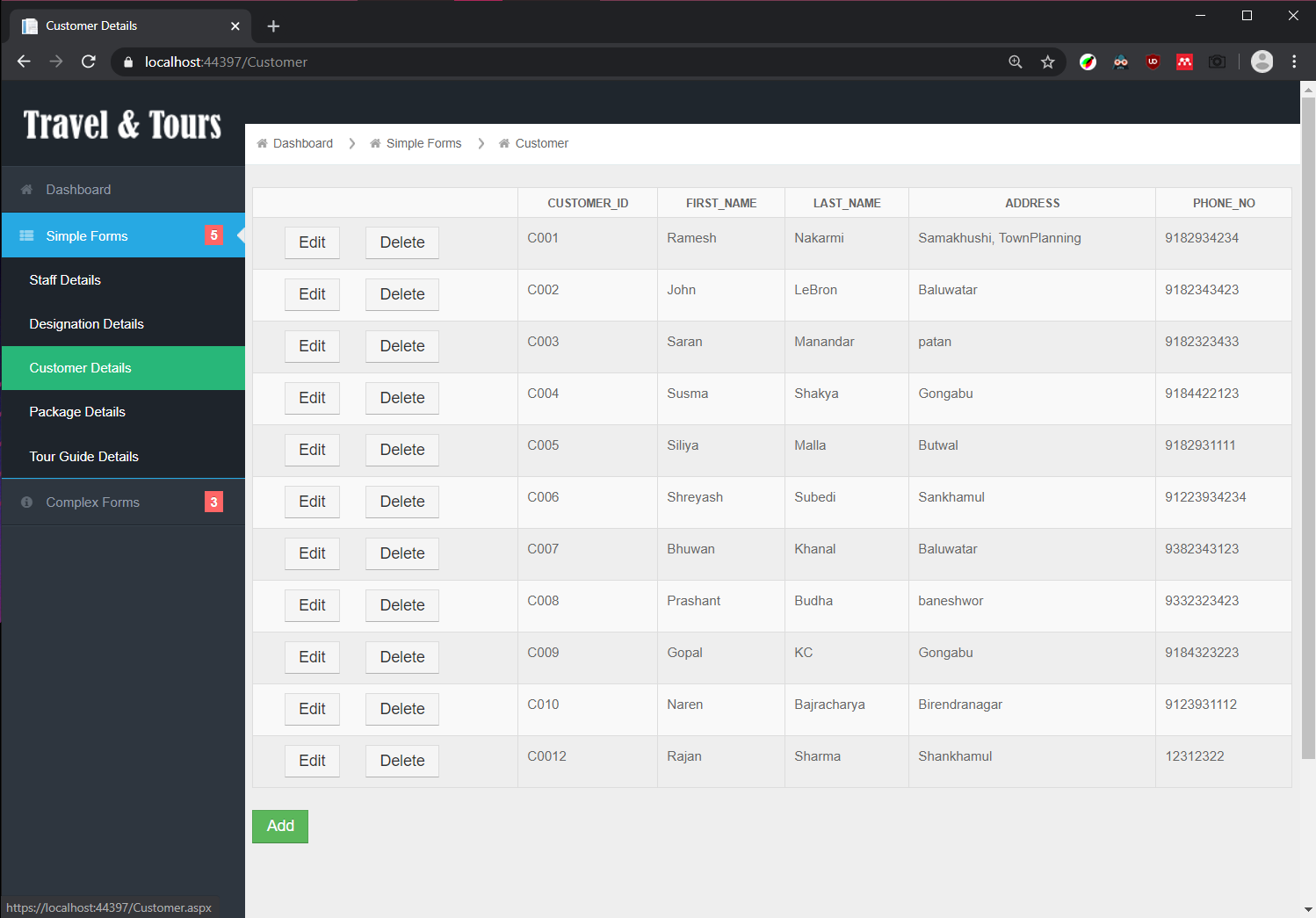


Figure : Simple Web-forms for Customer Details

### Package Details

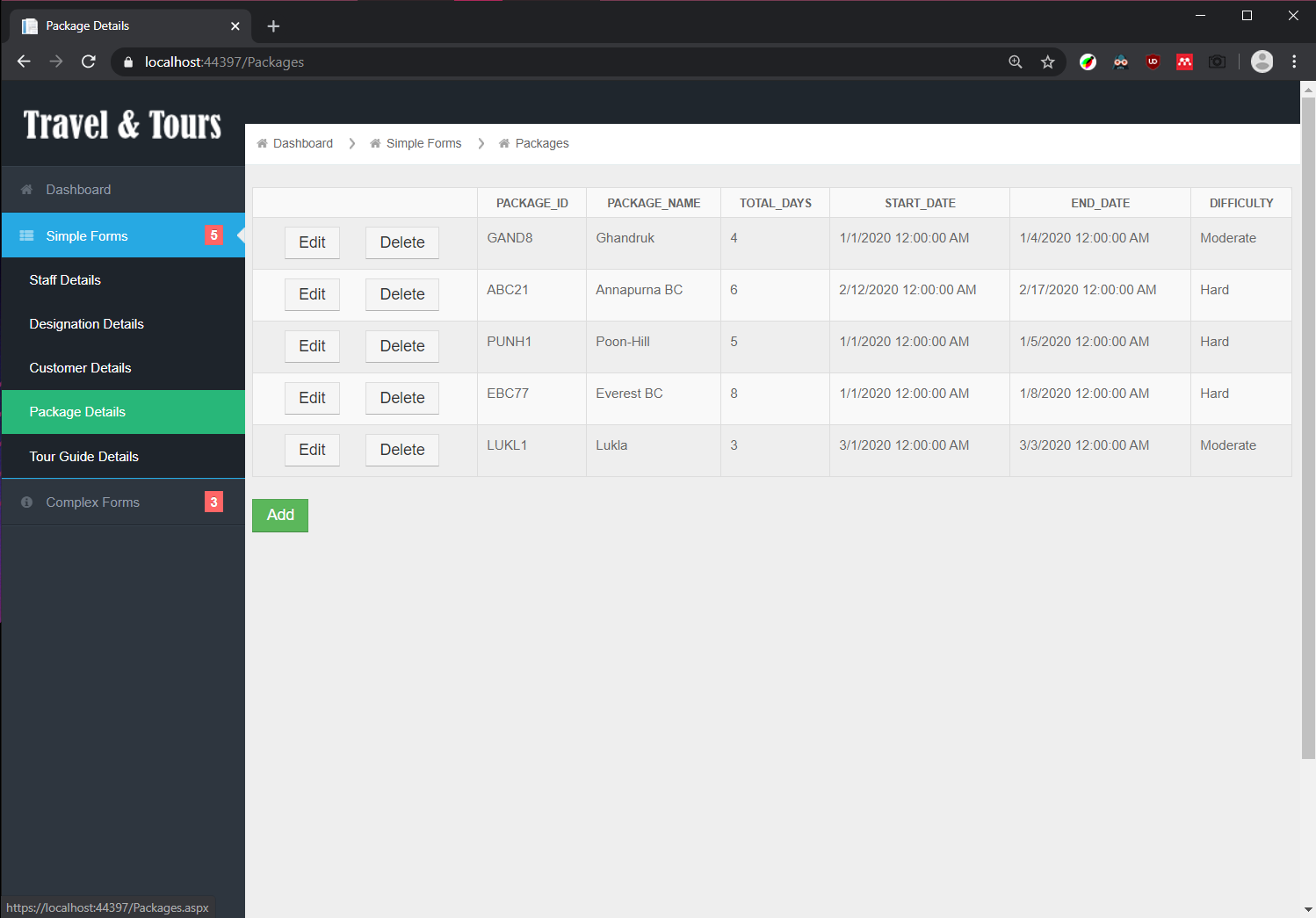


Figure : Simple Web-forms for Package Details

### Tour Guide Details

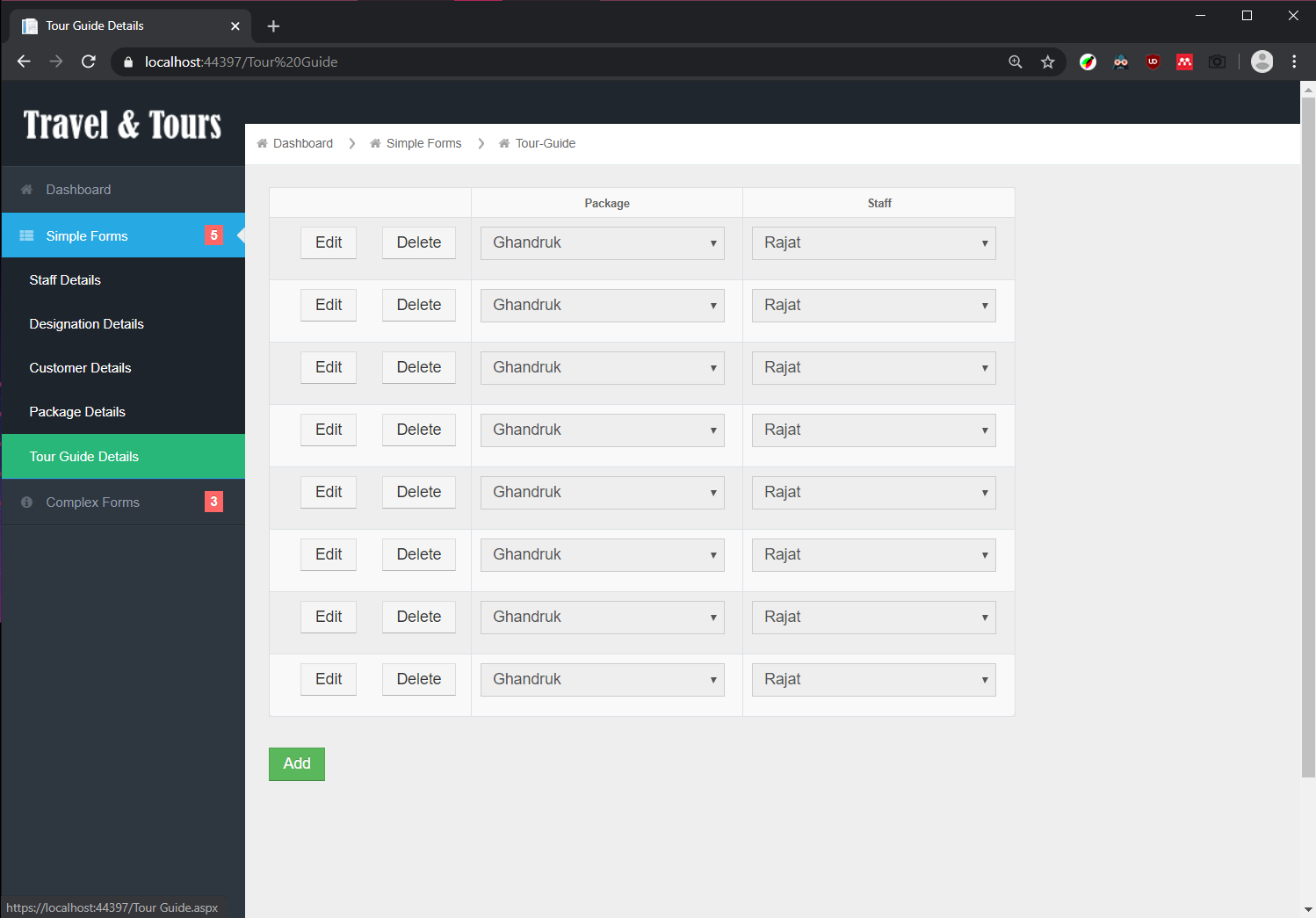


Figure : Simple Web-forms for Tour Guide Details

## 6.2. Complex Webforms

These web forms perform certain complex queries to extract information via joining various tables.

### 6.2.1. Customer-Package Schedule Form

The Customer-Package Schedule Form for any package shows the details of the package and the details of all customer who choose it.

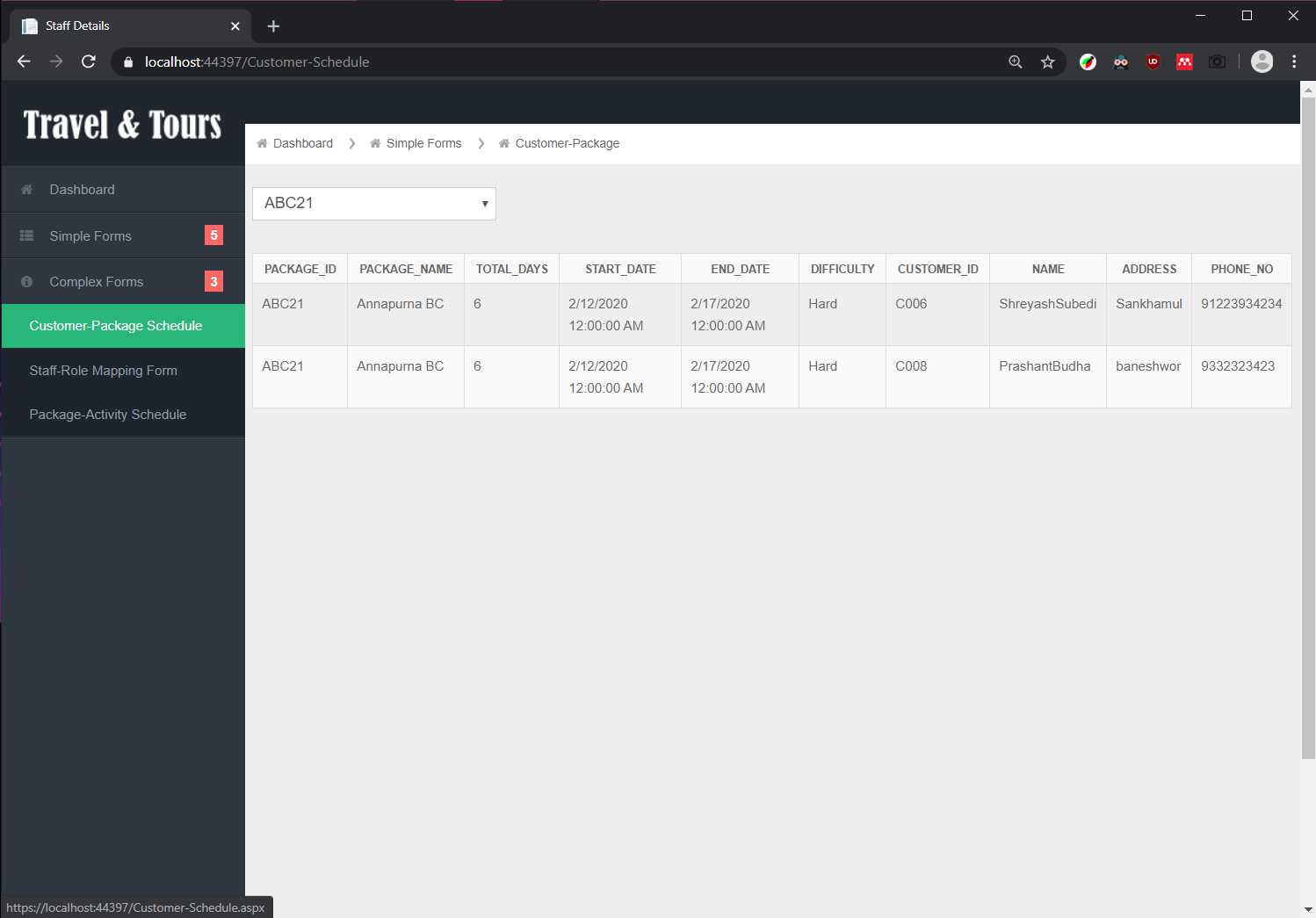


Figure : Complex Web-forms for Customer-Package Schedule Form

### 6.2.2. Staff-Role Mapping Form

Staff Role Mapping Form shows the count of staff of selected role in the company. This Form also shows additionally the details on staff members of the selected role types.

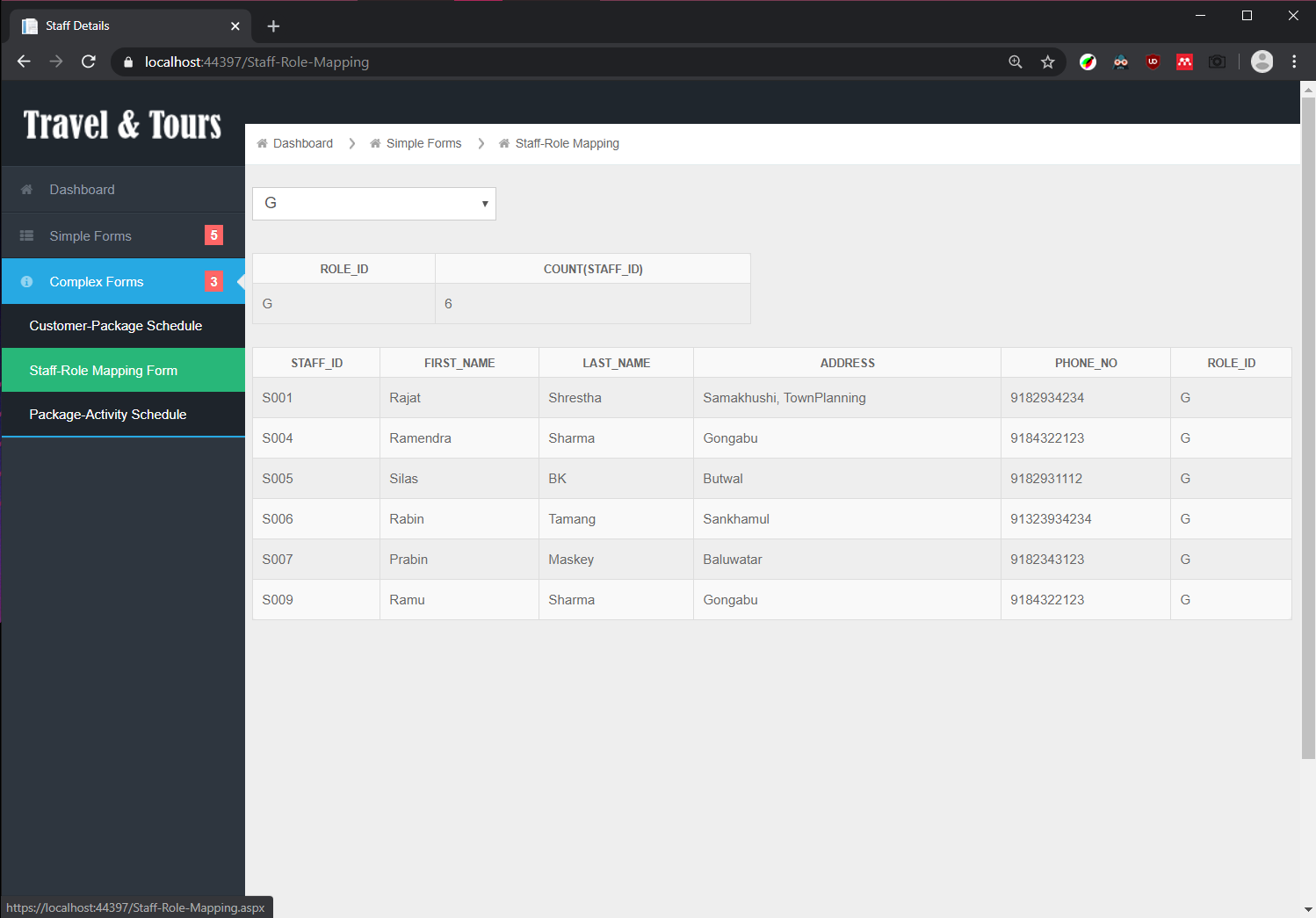


Figure : Complex Web-forms for Staff-Role Mapping Form

### 6.2.3. Package-Activity Schedule Form

Package-Activity Schedule Form shows the details of the activities, travel details, mode for any selected package

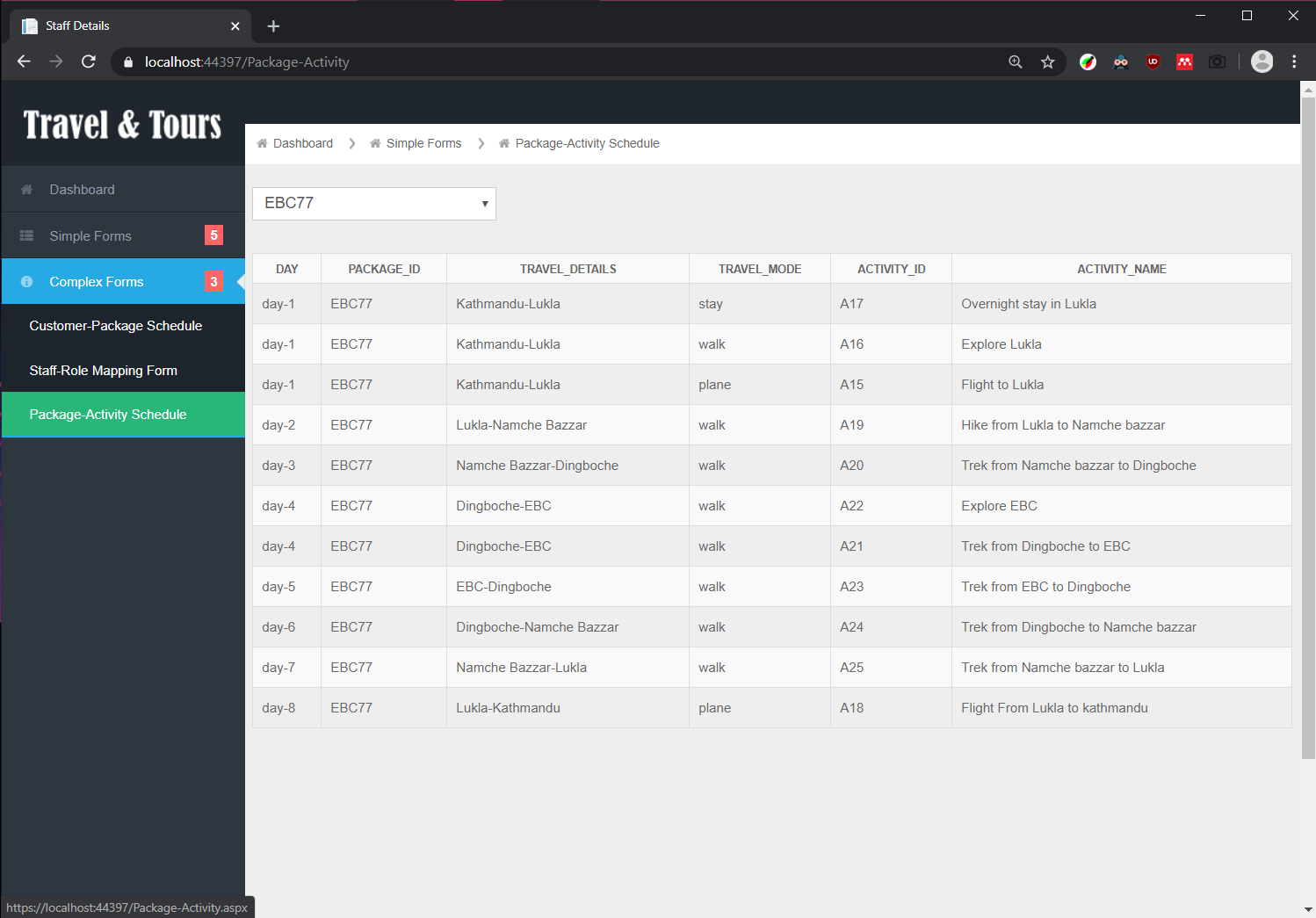


Figure : Complex Web-forms for Package Activity Schedule Form

## 6.3. Dashboard

The simple dashboard acts as the home screen of the application and shows various insights on the contents of the database. This includes links to the various simple and complex web forms. The charts show the insights on the staff count by designation and the customer count on each package. This page also contains the indicator for data inserted in each table.

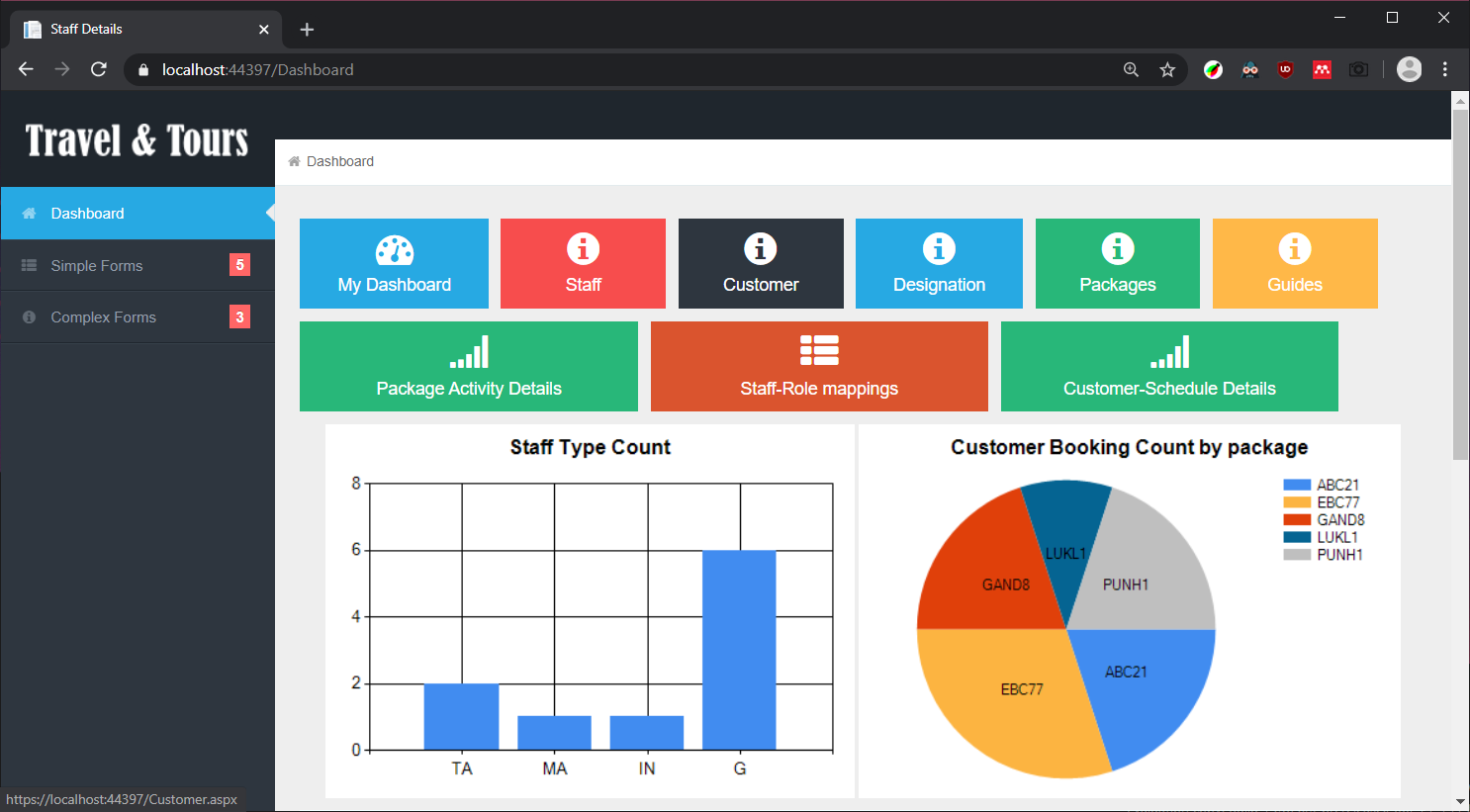


Figure : The Dashboard with Links to webforms and Charts

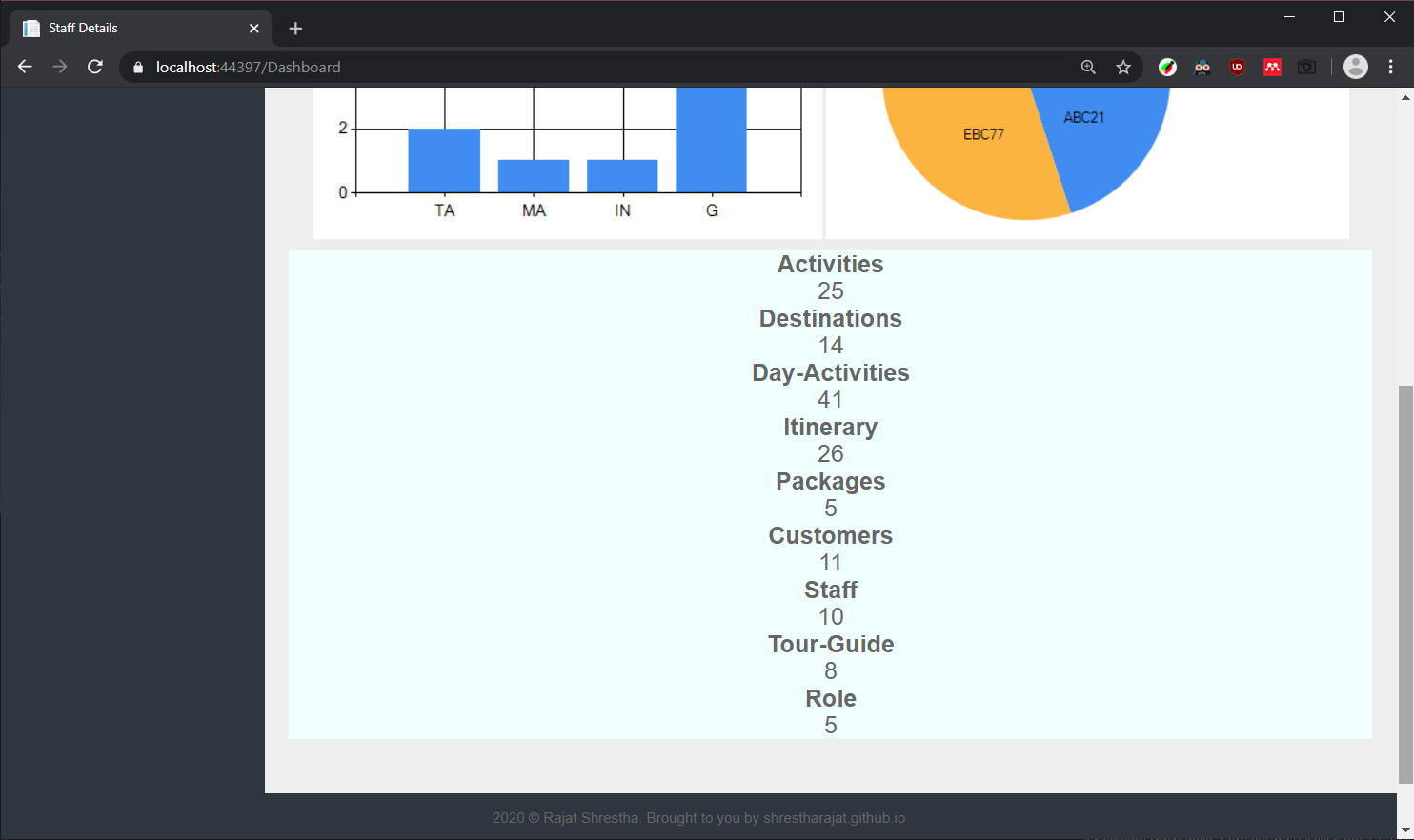


Figure : Dashboard with details on the Number of data entered in the database

# Testing:

## Simple Forms:

Since the Crud operations for the Simple forms are the same each form is selected to carry out unique tasks in the tests.

### Adding Data

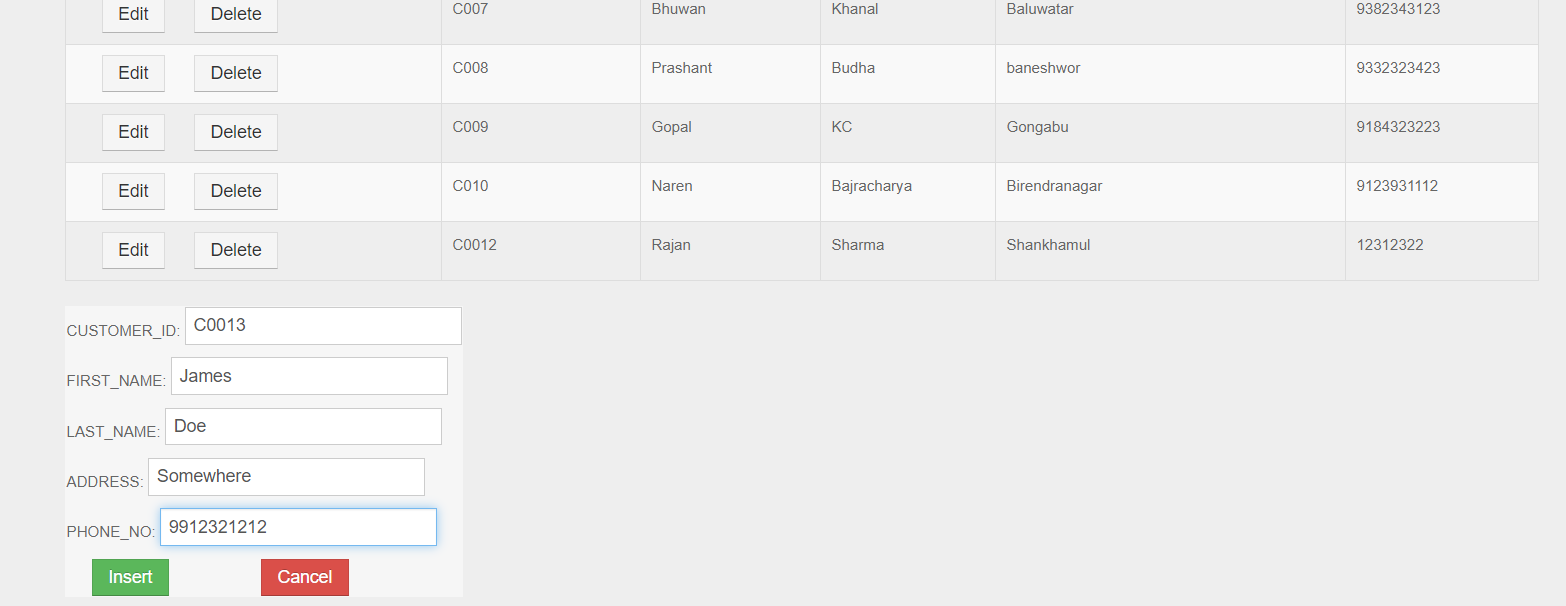


Figure : Entering Data in the Customer Form

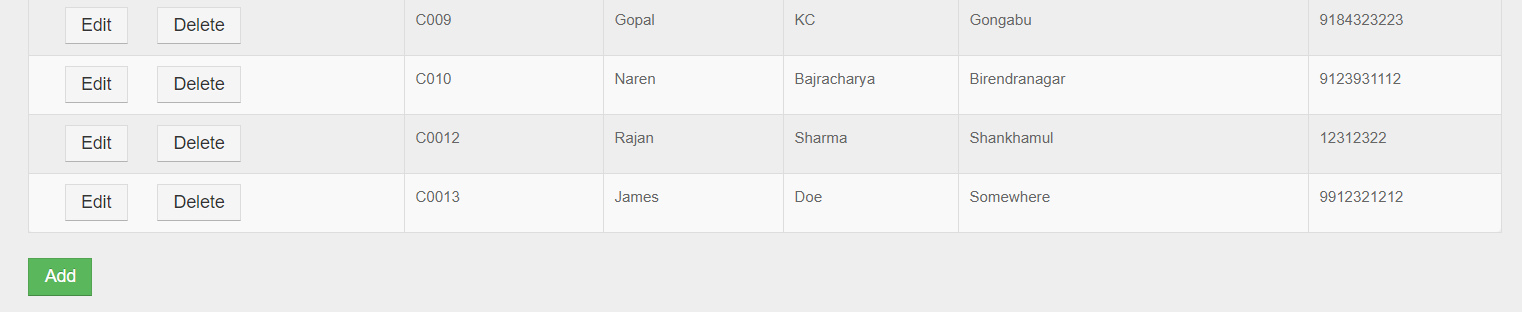


Figure : New Customer inserted in the customer table

|  |  |
| --- | --- |
| Test Summary | Adding data in Customer form |
| Action | Adding customer details |
| Expected Output | Details to be entered in table and shows up in form |
| Actual Output | Details are entered in table and are shown in the form |
| Result | Success |

Table : Test 1 Simple Form Adding Data

### Editing Data



Figure : Staff data before editing

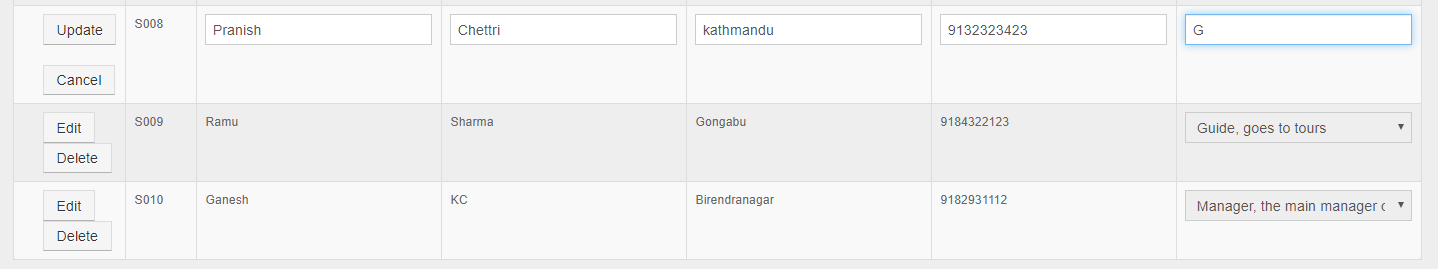
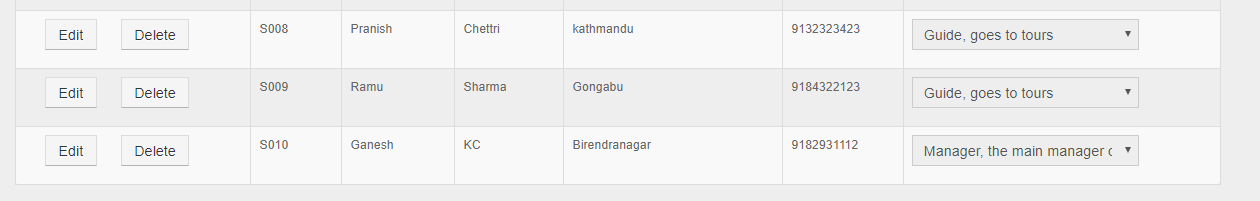


Figure : Editing Staff Data



|  |  |
| --- | --- |
| Test Summary | Edit Staff Details |
| Action | Edit a staff detail in staff form |
| Expected Output | Details of a staff to be updated and Primary key should not be updated |
| Actual Output | Details of the staff is updated, and the PK was made inactive so that it could not be edited |
| Result | Success |

Figure : Staff data after editing

Table : Test 2 Simple Form Editing Data

### Deleting Data

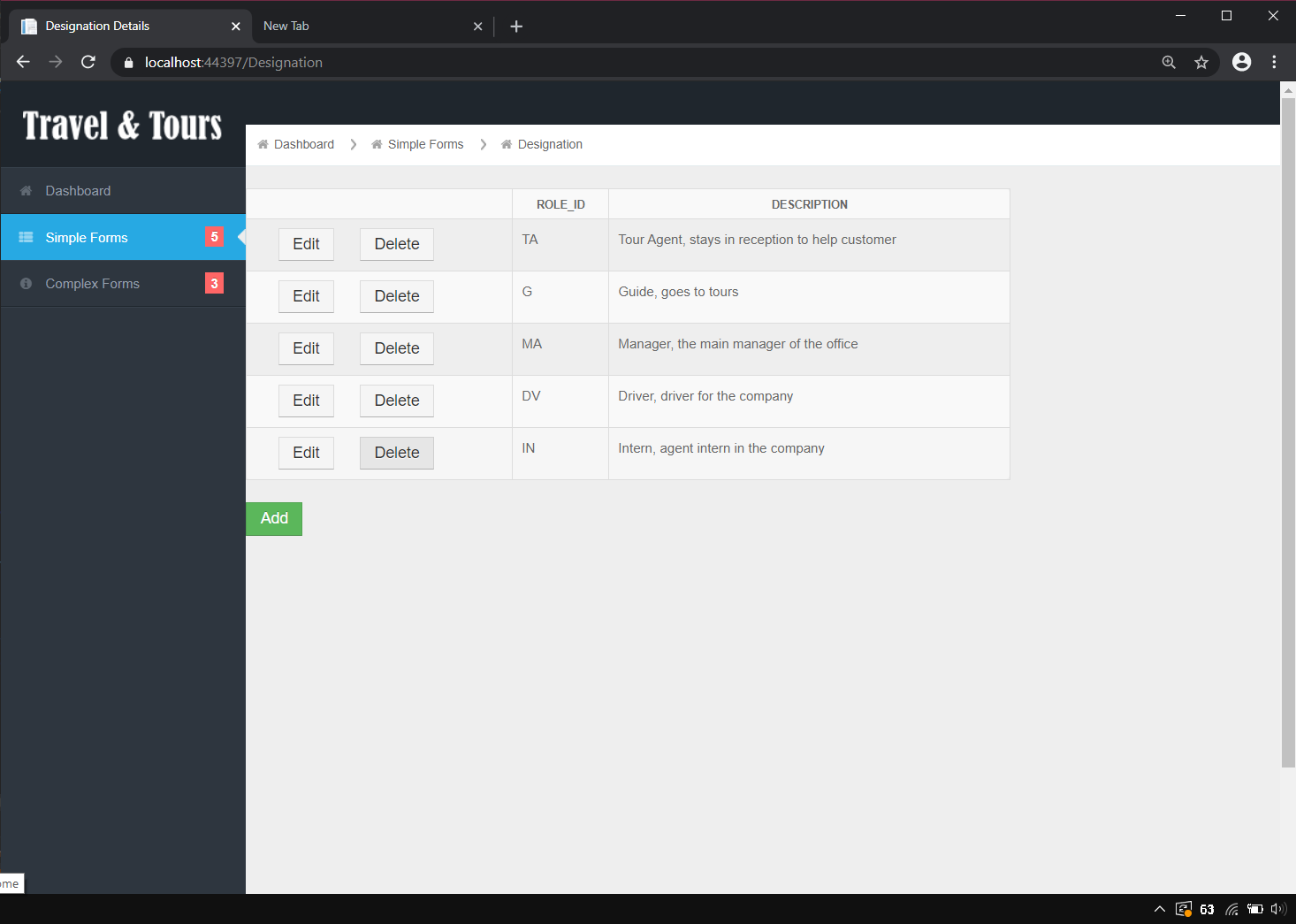


Figure : Designations before deleting

|  |  |
| --- | --- |
| Test Summary | Deleting an entry in Designation form |
| Action | Delete Intern from Designation as there are no interns |
| Expected Output | Intern to be removed |
| Actual Output | The Intern Entry was removed |
| Result | Success |

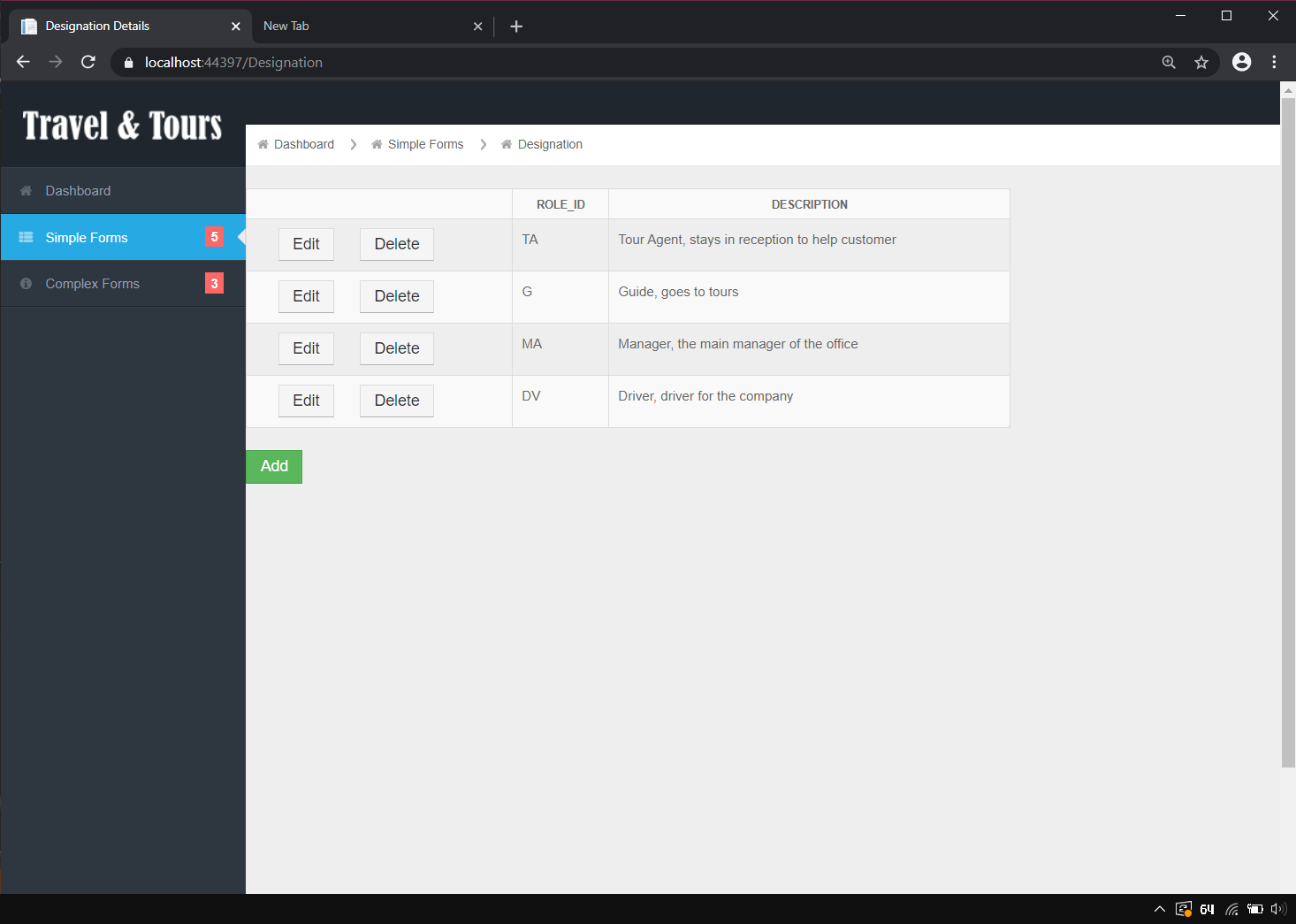


Figure : Intern designation removed after Deleting

Table : Test 3 Simple Form Removing data

### Adding Duplicate Data

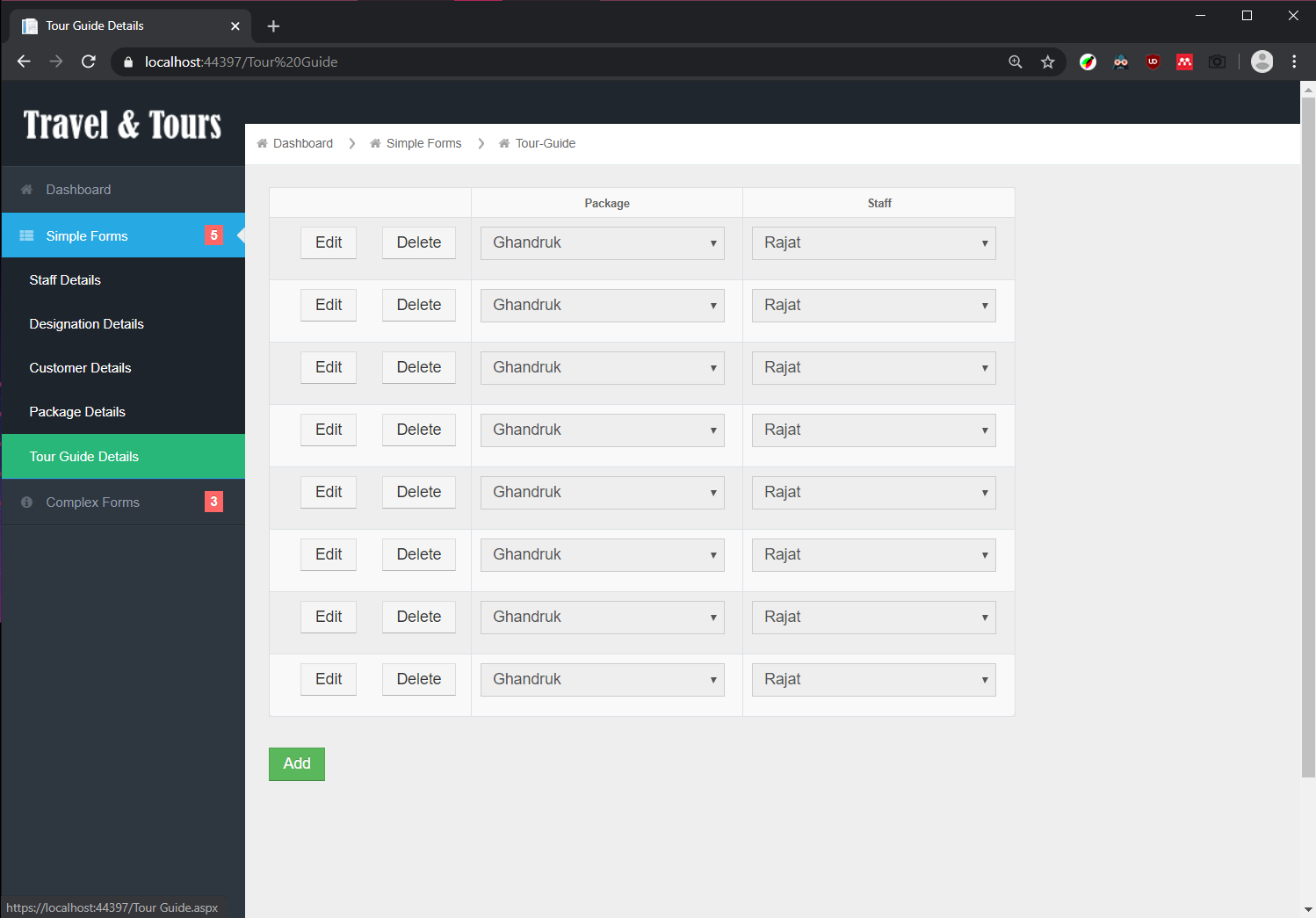


Figure : Editing the Tour-guide to be duplicate

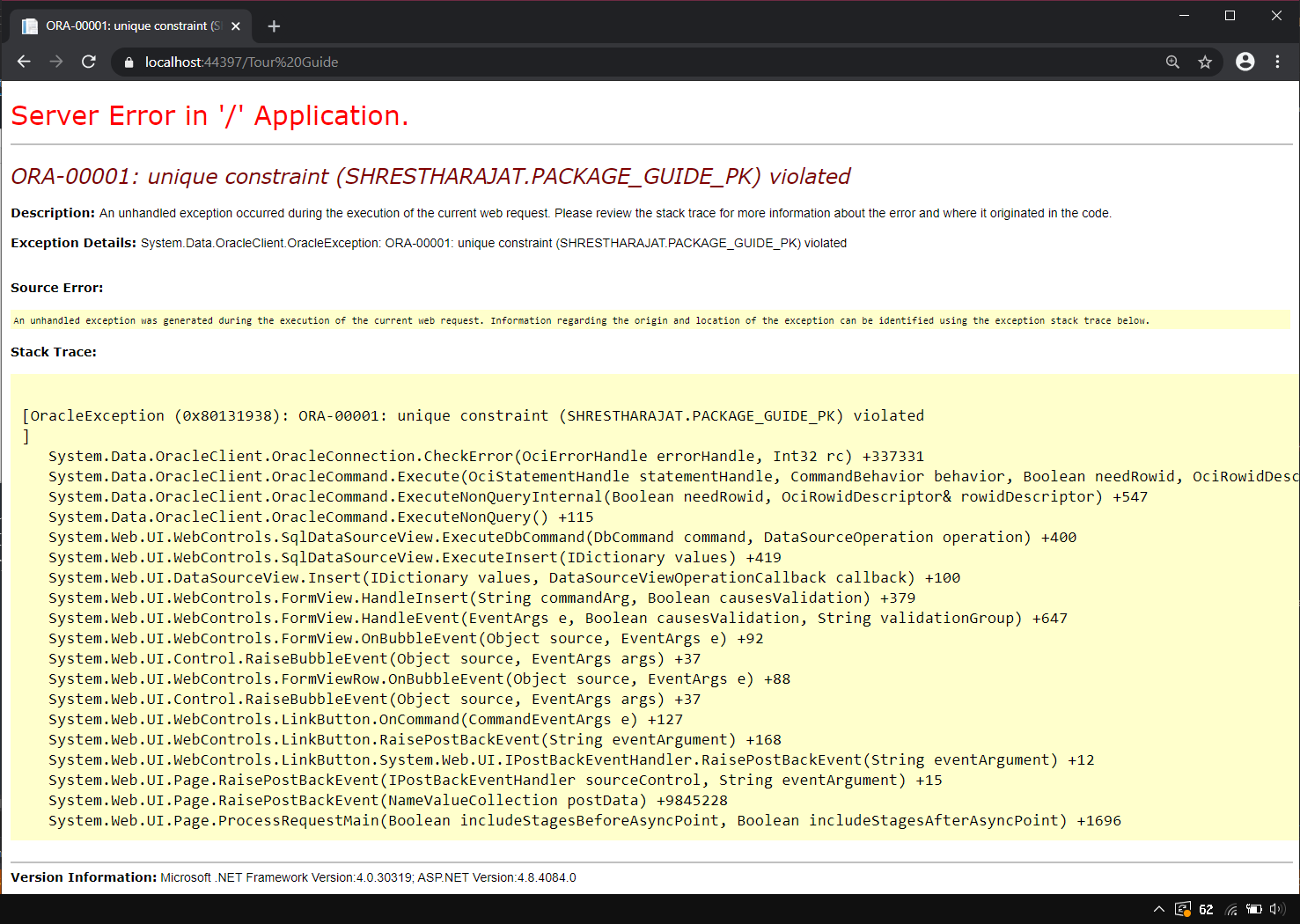


Figure : Exception occurred due to duplicate data

Table : Test 4 Simple Form Adding duplicate data

|  |  |
| --- | --- |
| Test Summary | Try to add Duplicate data in Simple Form |
| Action | Add duplicate data in tour guide form |
| Expected Output | Error occurs and halts the operation |
| Actual Output | An Exception occurred not allowing data to be entered |
| Result | Success |

### Deleting referenced entry

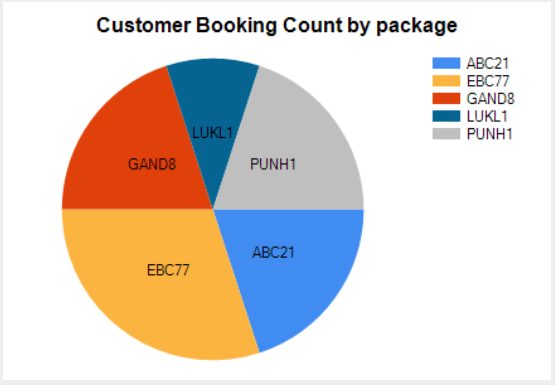


Figure : All the Packages are currently referenced

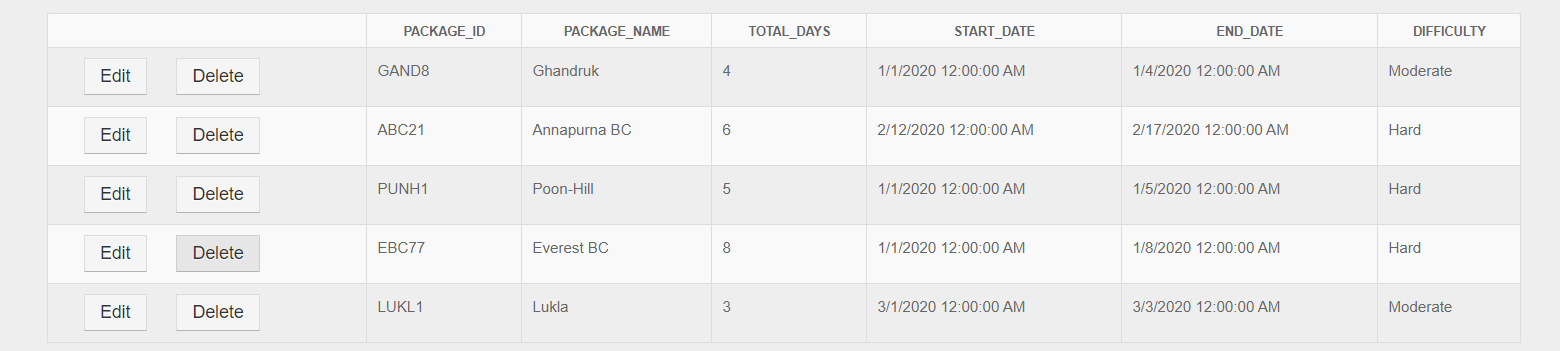


Figure : Deleting one referenced data (package)

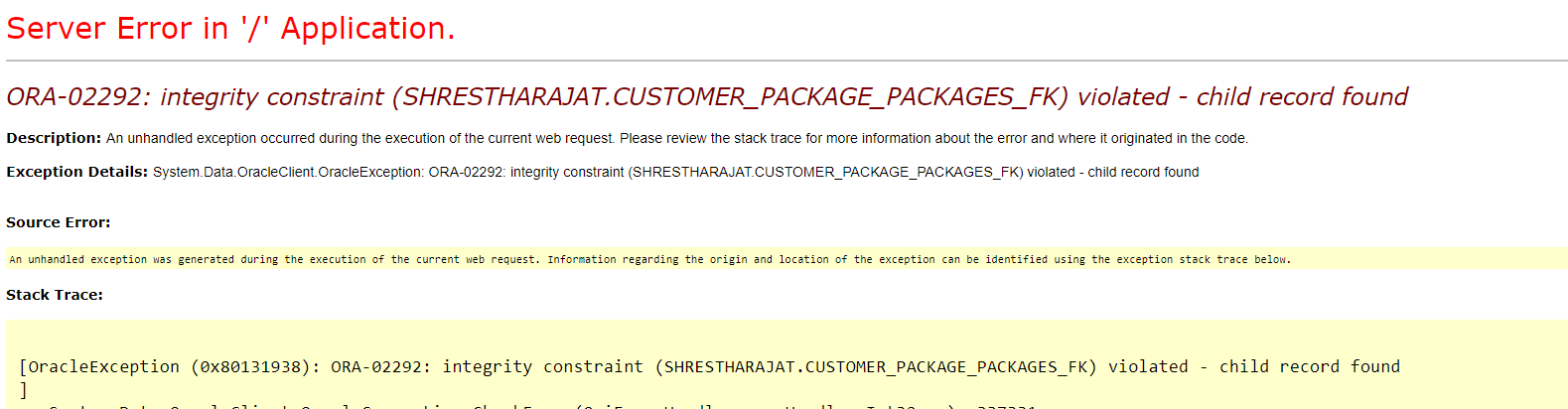


Figure : Exception occurred while deleting referenced item

|  |  |
| --- | --- |
| Test Summary | Try to delete referenced data in Simple Form |
| Action | Delete any package from package form |
| Expected Output | Error occurs and halts the operation |
| Actual Output | An Exception occurred not allowing data to be deleted |
| Result | Success |

Table : Test 5 Simple Form deleting referenced data

## Complex Forms:

### Filtering the Package-Customer Form

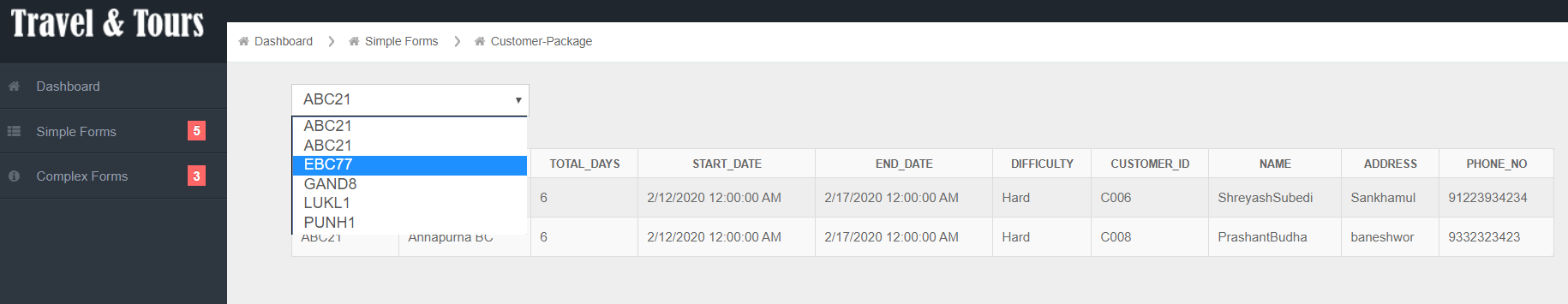


Figure : Selecting Package Id to show customer booking the selected package

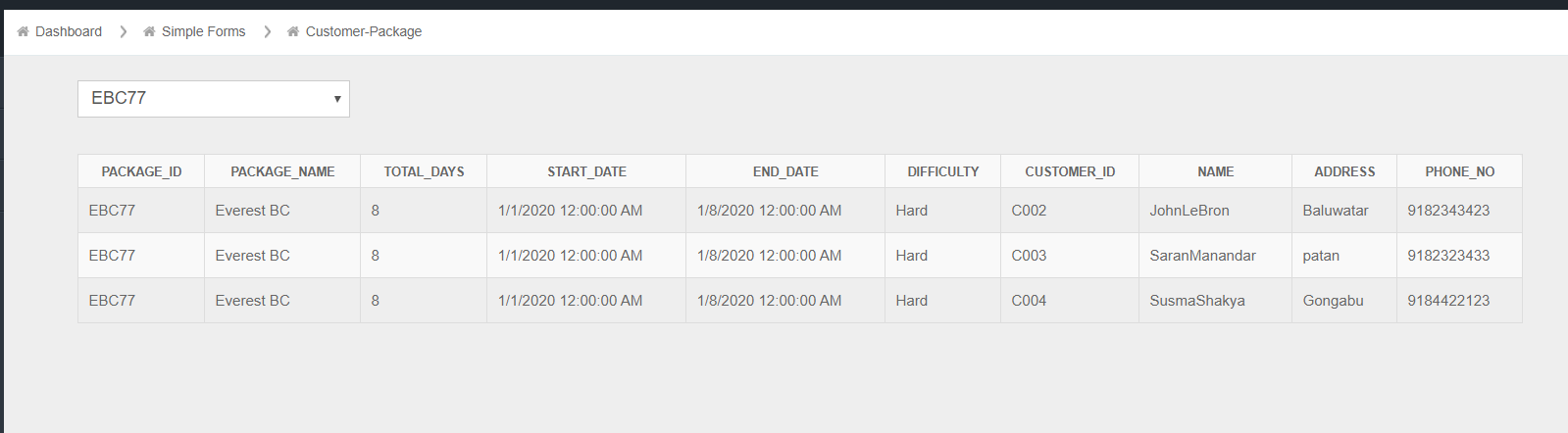


Figure : Table showing customers who is booking the selected Package

Table : Test 6 Complex Form Customer-Package Filtering

|  |  |
| --- | --- |
| Test Summary | Check the Package-Customer Filtering |
| Action | Select any Package |
| Expected Output | The table will show customer data for selected package |
| Actual Output | The table showed customer data for selected package |
| Result | Success |

### Filtering Staff-Role mapping Form

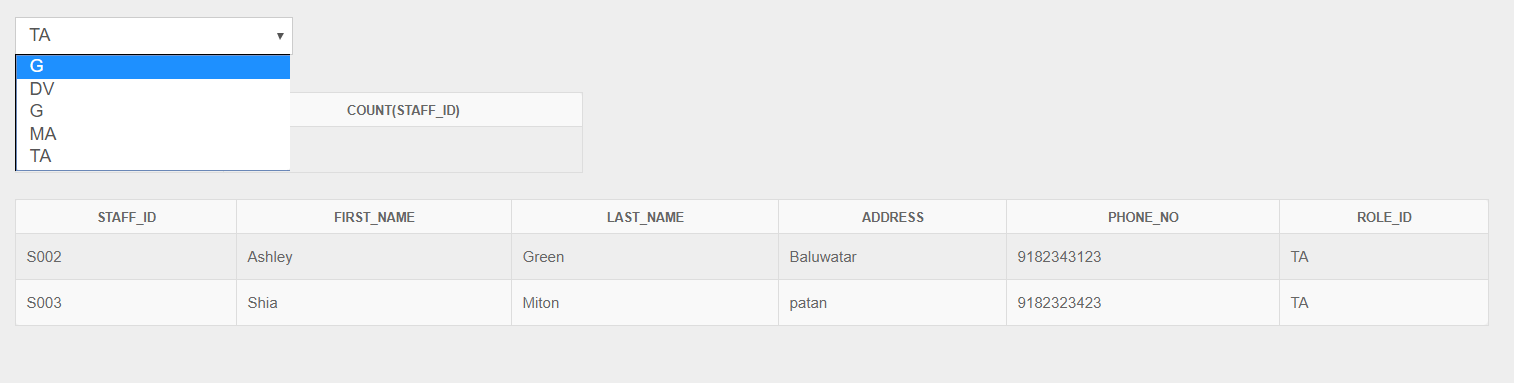


Figure : Selecting the Designation to count the no of staffs in it and staff details

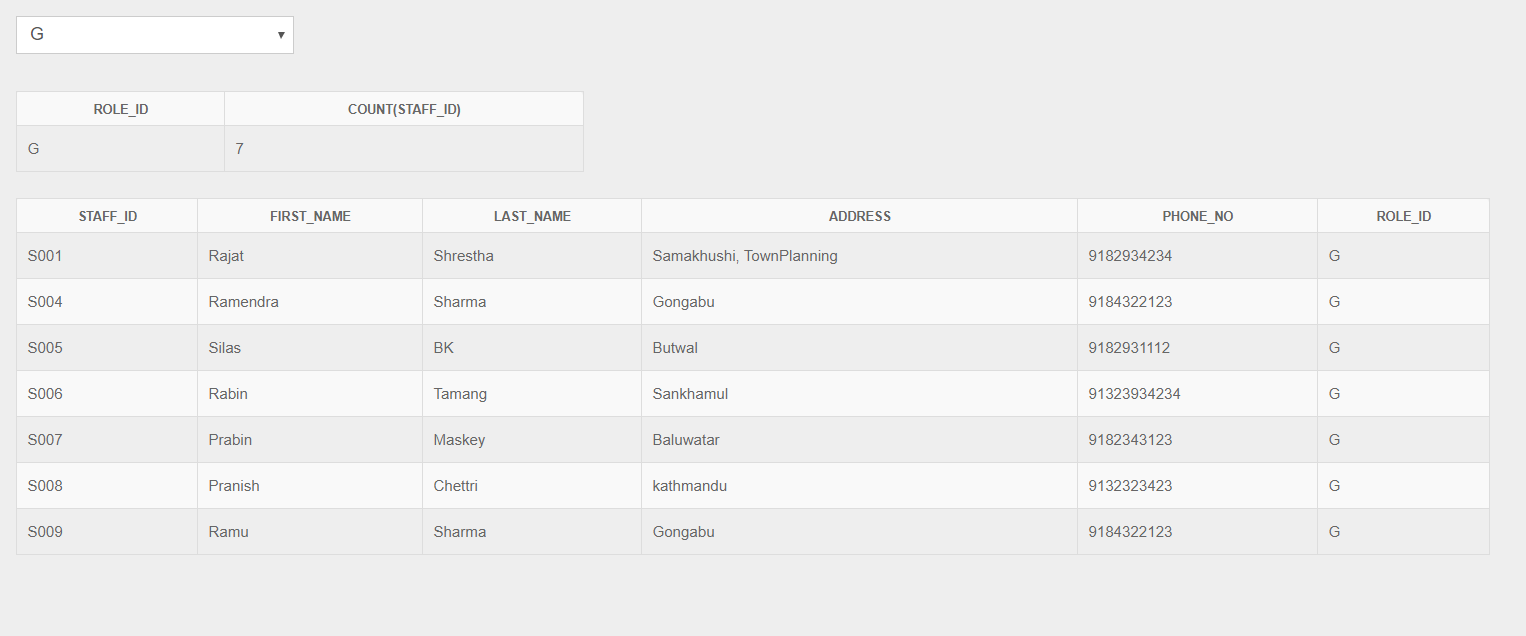


Figure : Staff details and count of the selected designation

Table : Test 7 Complex Form Staff role mapping

|  |  |
| --- | --- |
| Test Summary | Check the Staff Role mapping Filtering |
| Action | Select any Role to Filter |
| Expected Output | The table will show staff data for selected role |
| Actual Output | The table showed staff data for selected role |
| Result | Success |

### Filtering Package-Activity Schedule Form

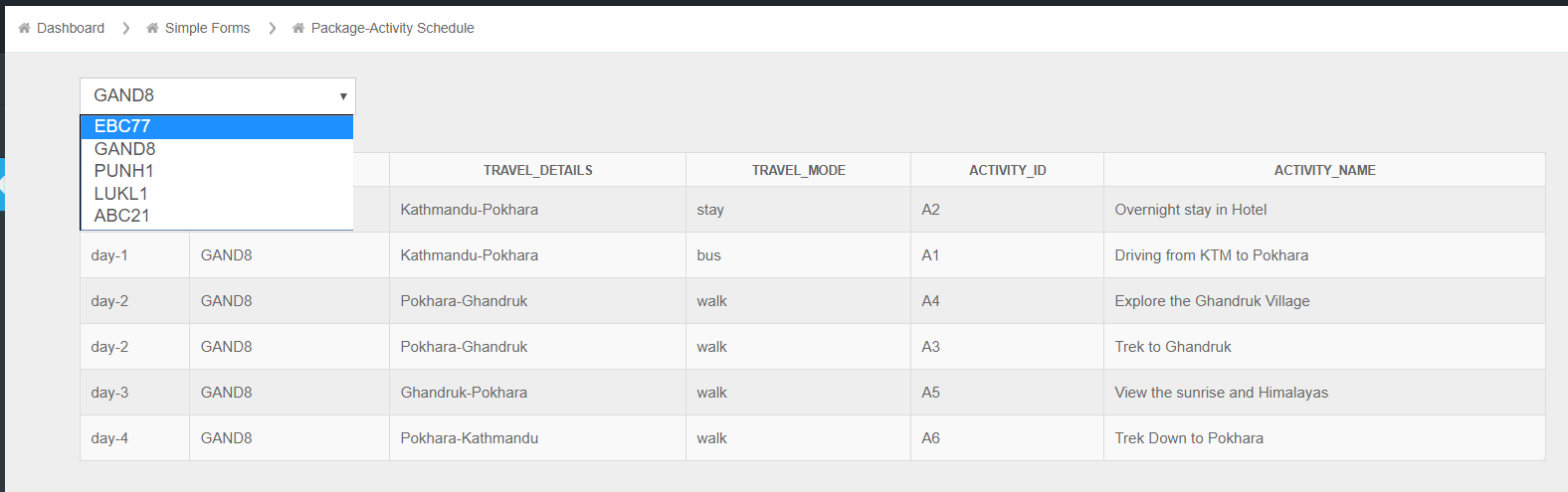


Figure : Selecting Package Id to view the enlisted activities

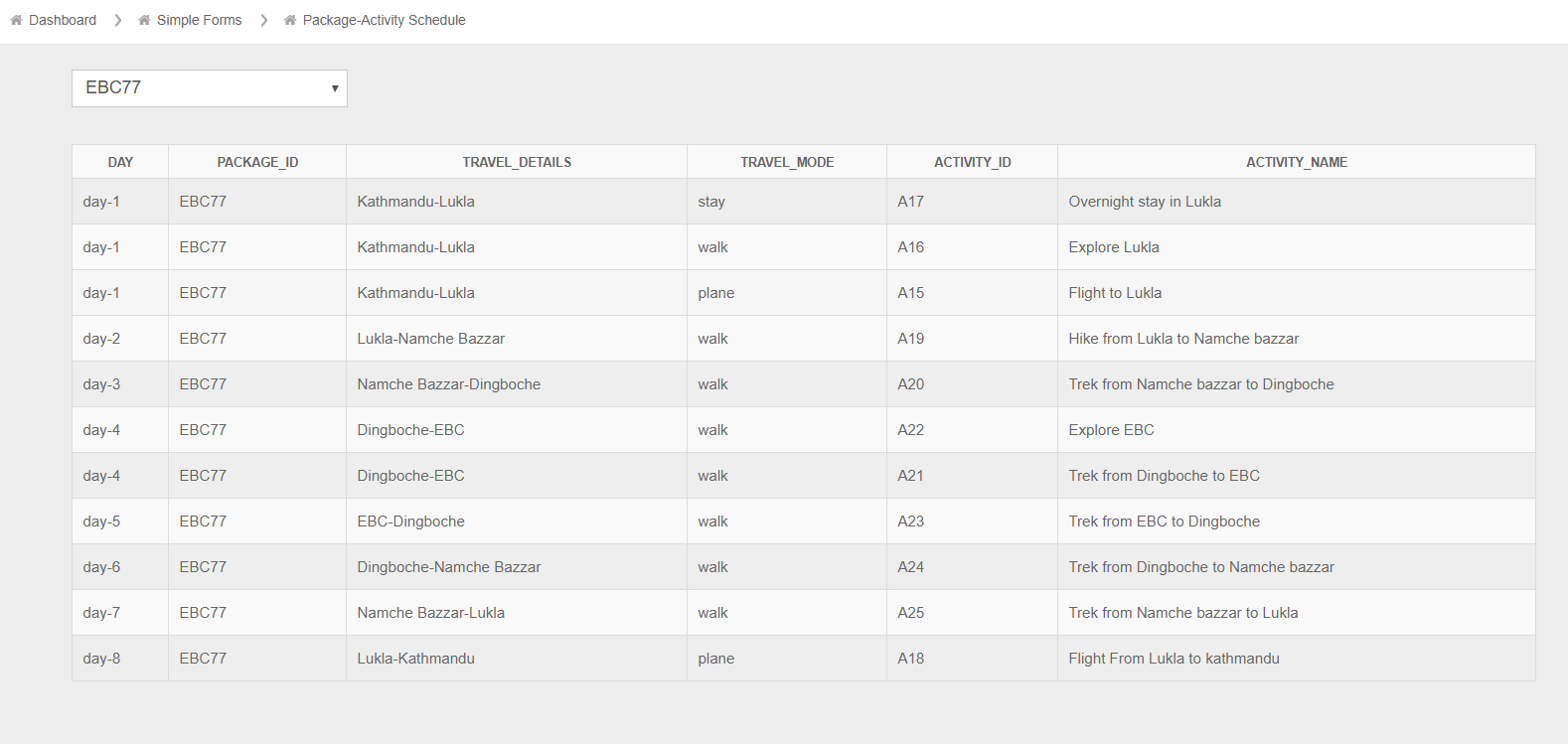


Figure : Enlisted Activities of the selected package

Table : Test 8 Complex Form Package activities

|  |  |
| --- | --- |
| Test Summary | Check the package Activity filtering |
| Action | Select any Package to view Activities |
| Expected Output | The table will Activities of the selected package |
| Actual Output | The table showed Activities of the selected package |
| Result | Success |

## Dashboard

### Testing All of the Links

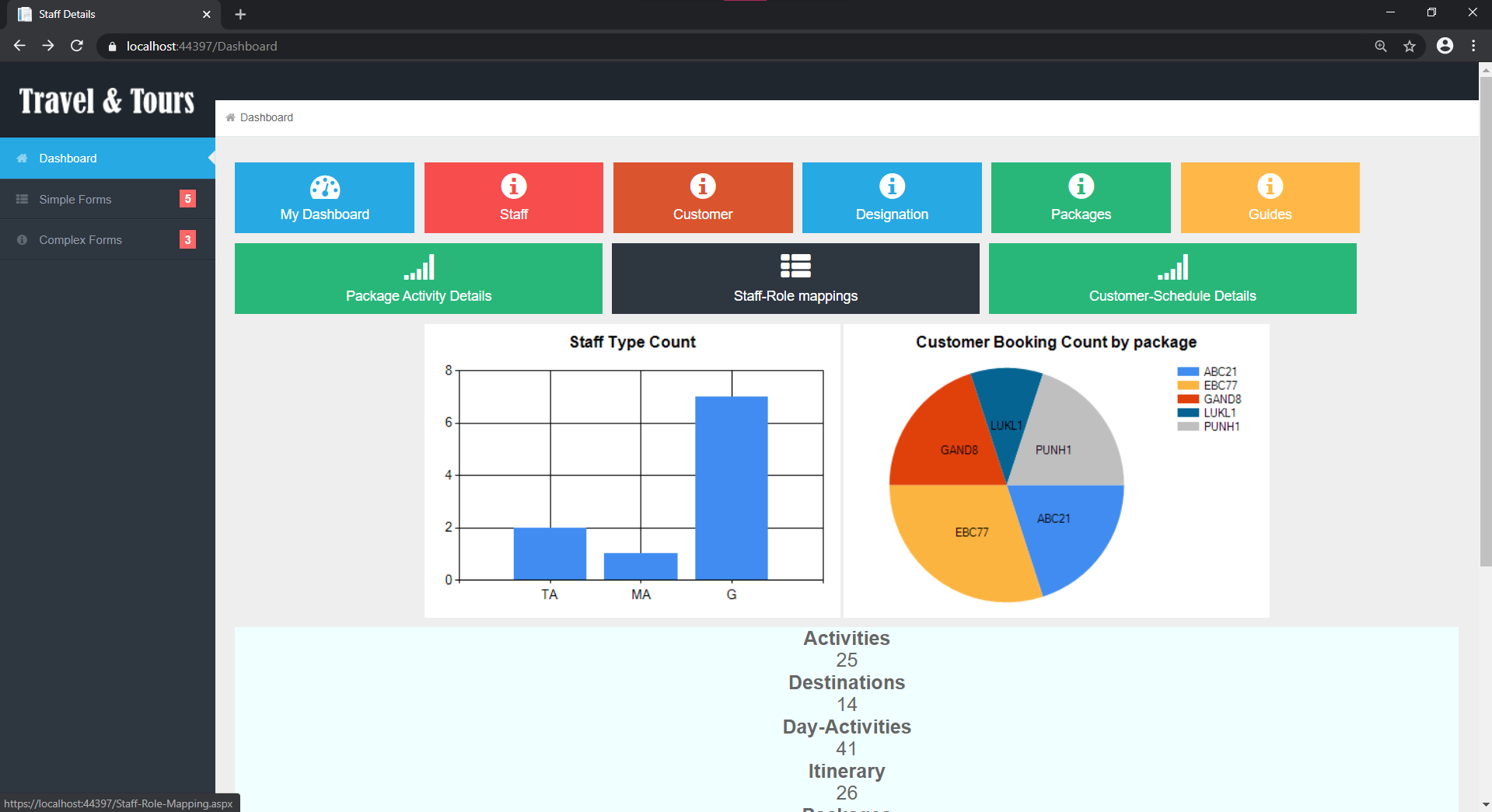


Figure : Clicking the staff role link

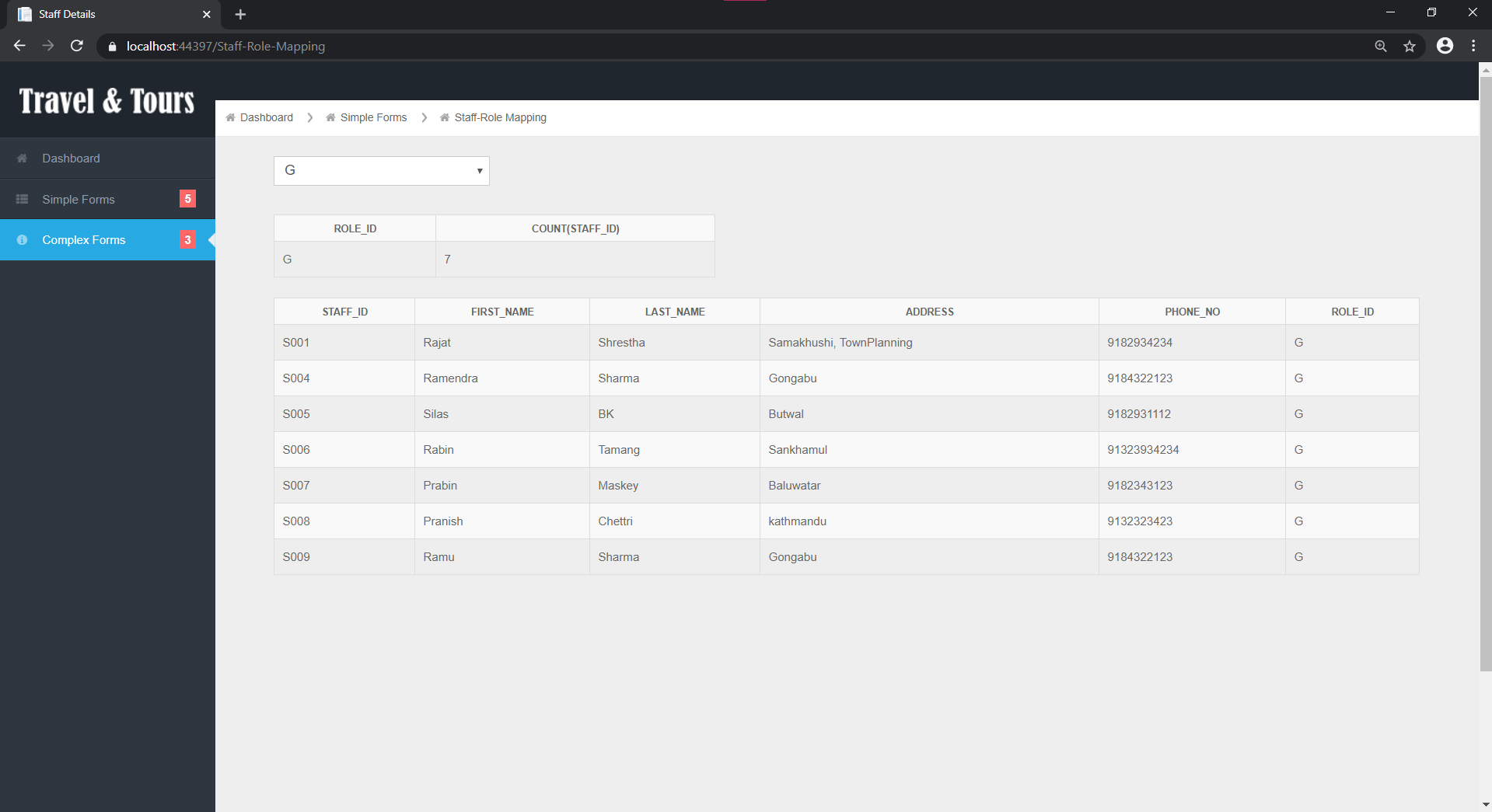


Figure : Getting redirected to the staff-role page

Table : test 9 Checking Links in dashboard

|  |  |
| --- | --- |
| Test Summary | Check the Dashboard links |
| Action | Select Staff-Role mapping link in the dashboard |
| Expected Output | The page will be redirected to the staff-role form page |
| Actual Output | The page was redirected to the staff-role form page |
| Result | Success |

### Testing the charts in the Dashboard

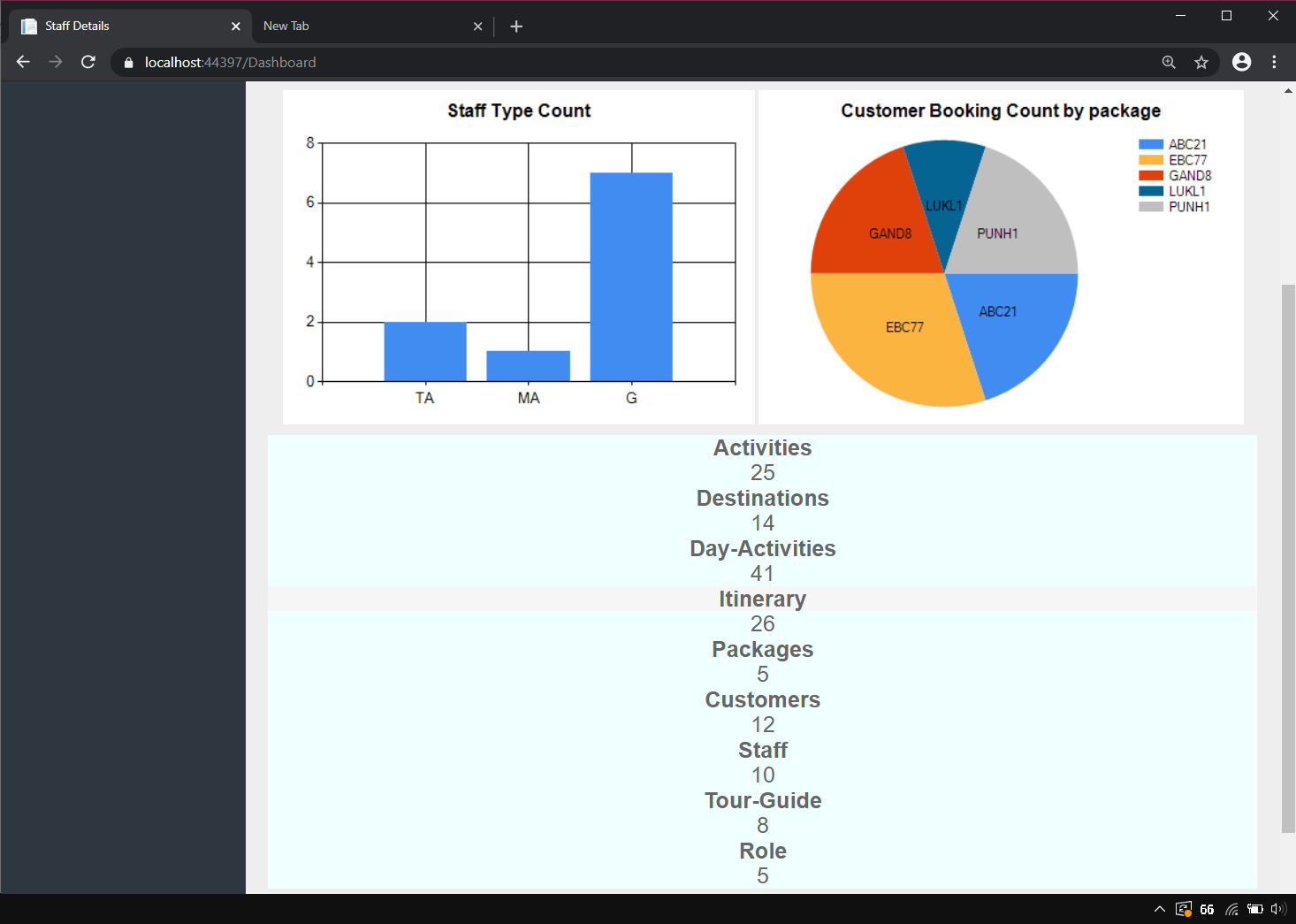


Figure : Charts and data before updating data in the database

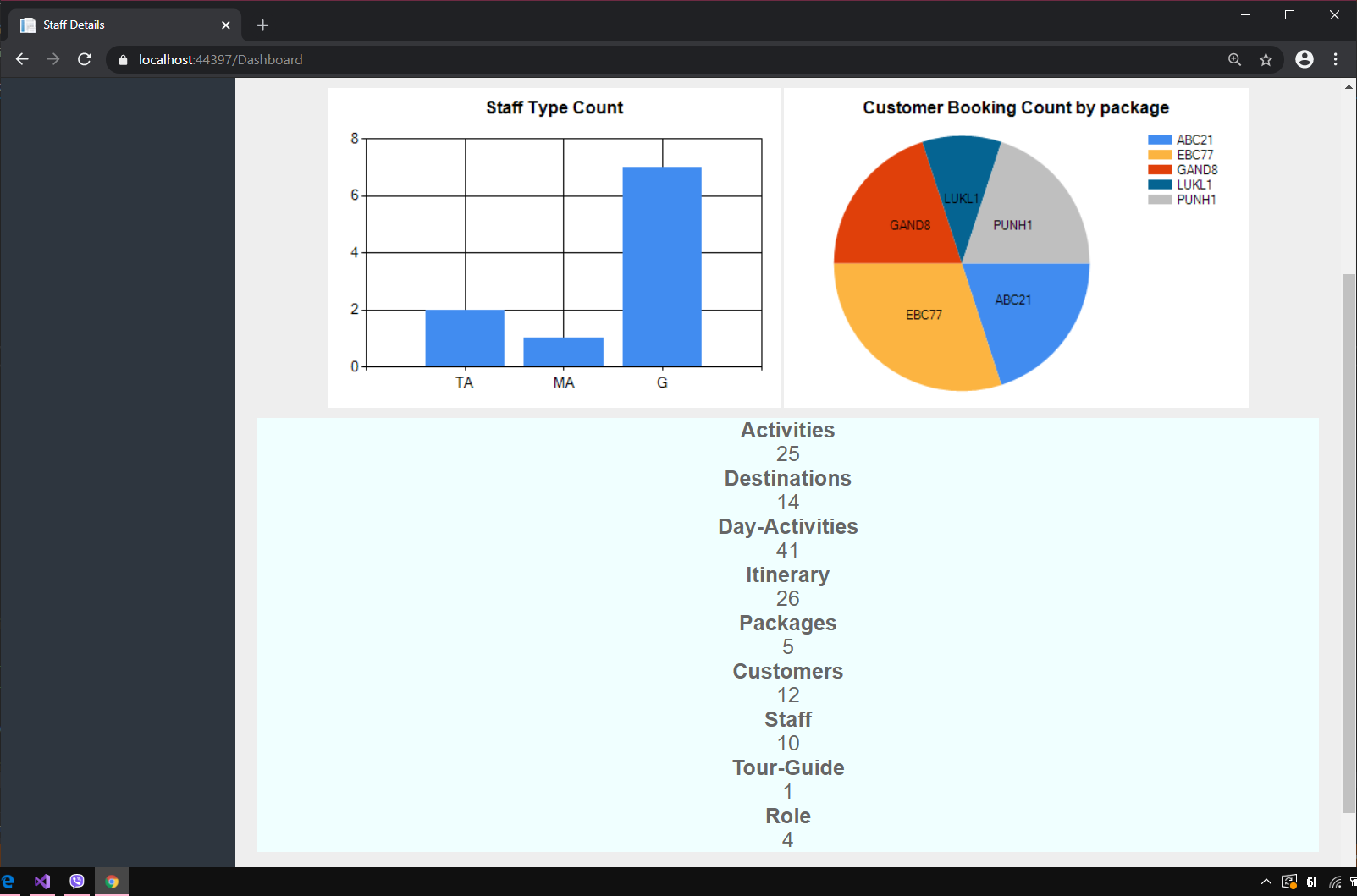


Figure : Charts and data after updating the database

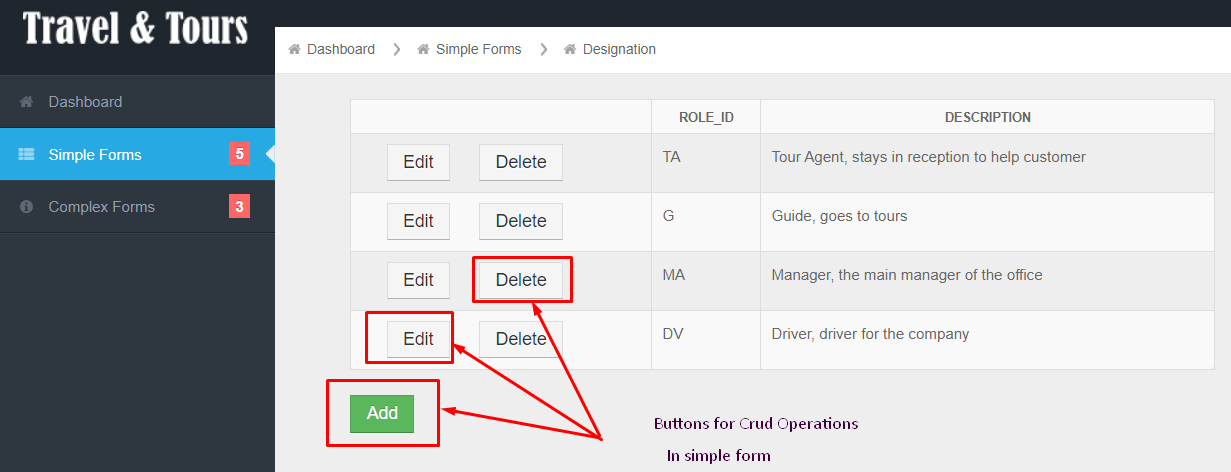
|  |  |
| --- | --- |
| Test Summary | Check the Dashboard charts |
| Action | Modify Data in the Database |
| Expected Output | The Graphs will readjust according to the data in the database. |
| Actual Output | The Graphs was readjusted according to the data in the database. |
| Result | Success |

Table : Test 10: Checking Graphs in Dashboard

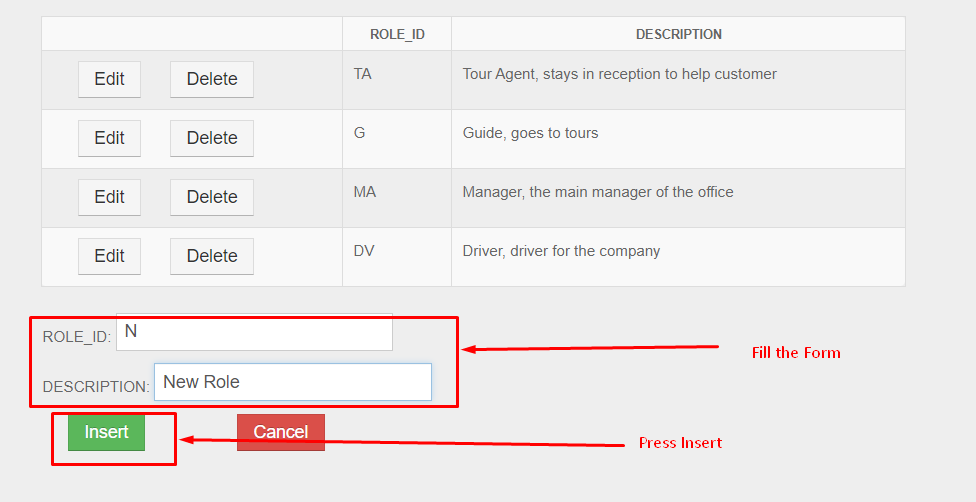
# User Manual

Since the Crud operations for all the form is the same steps

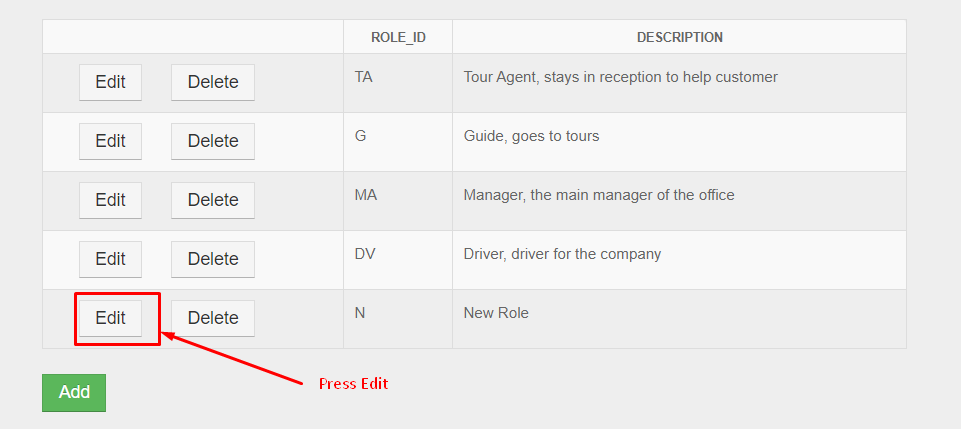
## CRUD Operations



### Addition

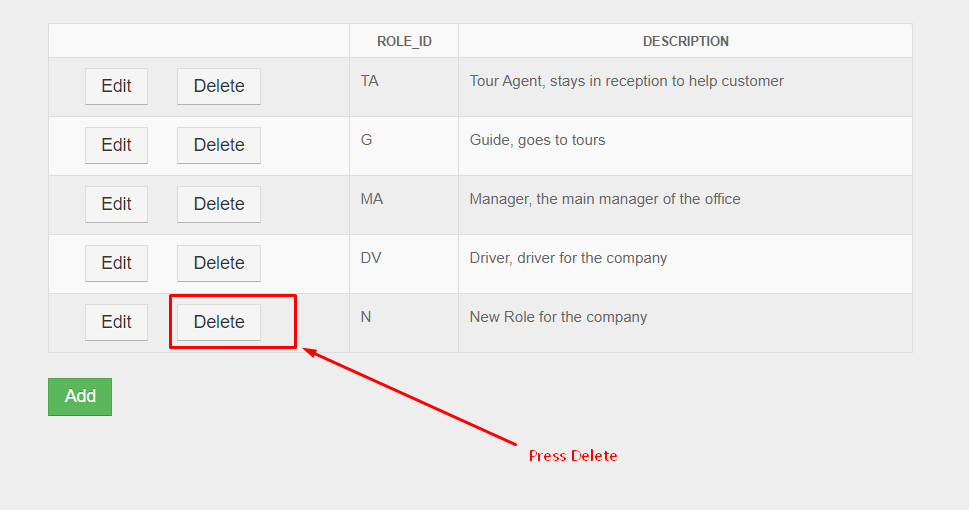


### Editing





### Deleting



# Further Discussion

This coursework was quite fun to do and had a lot of new challenges which required various tricks and tips to continue this coursework has improved the database management and problem skills due the to the complex scenarios involving various entities and relations. Mainly this coursework focuses on using the Given tools for solving the problem:

1. Oracle database
2. Visual Studio
3. Oracle SQL Developer
4. Oracle SQL Developer Datamodeler
5. Visual Studio Code

The Following tools were essential in their own way for hosting a local database server, an Integrated Development Environment to work with ASP .NET framework, an SQL Query Developing platform, ER- Model generator and a simple text editor. After doing the required tasks using core technologies the project was further enhanced by various other technologies which improves various performance and development of the application such. The core and optional technologies used are:

1. HTML
2. CSS
3. Java Script
4. C#
5. Bootstrap
6. jQuery
7. ASP .NET
8. SQL
9. PL/SQL

tools and technologies required for the development of the given project greatly helped on understanding how different technologies work together to create a functional project and exposed various new tricks on how a developer can save time by using such technologies. This project also strongly demonstrated how coding has become easier due to various proprietary software designed to automate and make complex problems simple in the modern day.

# References

Coronel, C. & Morris, S., 2018. *Database Systems: Design, Implementation, & Management.* s.l.:Cengage Learning.

Kreines, D., 2009. *Oracle Data Dictionary Pocket Reference.* 1st ed. Chicago: O'Reilly Media.

Nishadha, 2017. *Ultimate Guide to ER Diagrams ( Entity Relationship Diagrams ).* [Online]   
Available at: https://creately.com/blog/diagrams/er-diagrams-tutorial/  
[Accessed 28 January 2020].

Oracle, 2020. *Data Modeling with Oracle SQL Developer.* [Online]   
Available at: https://www.oracle.com/in/database/technologies/appdev/datamodeler.html  
[Accessed 28 January 2020].

Scottish Qualifications Authority, 2010. *Outcome 1:Fundamentals of Database Design.* [Online]   
Available at: https://www.sqa.org.uk/e-learning/MDBS01CD/page\_06.htm  
[Accessed 12 1 2019].

Watt, A., 2012. *Database Design.* 2nd ed. Vancouver: BCcampus Open Textbook.

### 

# Appendix