**AMERICAN INTERNATIONAL UNIVERSITY BANGLADESH (AIUB)**

***FACULTY OF SCIENCE & TECHNOLOGY***

A picture containing calendar

Description automatically generated

Course Title

**INTRODUCTION TO DATABASE**

**Fall 2023-2024**

**Section: G**

**TITLE**

**TRIPIFY – A TRAVEL & TOURISAM MANAGEMENT SYSTEM**

**Supervised By**

MD Sajid Bin Faisal

**Submitted By: Group no: 03**

|  |  |
| --- | --- |
| **Name** | **ID** |
| SHAILY SAHA | 22-48530-3 |
| PRETOM CHANDRA ROY | 22-48556-3 |
| TURJO DAS DIP | 22-48558-3 |

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **TOPICS** | | | **Page no.** |
| **Title Page** | | | **1** |
| **Table of Content** | | | **2** |
| **1.** | **Introduction** | | **3** |
| **2.** | **Case Study** | | **3** |
| **3.** | **ER Diagram** | | **4** |
| **4.** | **Normalization** | | **5-6** |
| **5.** | **Finalization** | | **7** |
| **6.** | **Table Creation** | | **8-13** |
| **7.** | **Data Insertion** | | **14-16** |
| **8.** | **Query Test** | | **17-25** |
| **9.** | **Database Connection** | | **26-29** |
| **10.** | **Conclusion** | | **30** |
|  | |
|  |
|  |  | |  |

1.Introduction:

The title of the project is “**Tripify – A Travel and Tourism Management System**”. By using the features of Oracle 10g, this system offers seamless booking, transport management, transport, payment processing to make travel planning convenient for users. With a user-friendly interface, it aims to provide a hassle-free experience for travelers. We had to go through case study, ER diagram, the normalization process for each relation, finalization and final tables and value insertion. By integrating key features, the project embraces the future of travel management with our innovative solution.

2.Case Study:

In Tripify - A Travel and Tourism management system, A customer is identified by customer id. The system contains name, address, email, mobile number, and national id. A customer address is composed of house number, street name and city. A customer name is also composed of first name and last name. Firstly, A customer reserves a hotel. One hotel must be reserved by one customer and one customer cannot reserve more than one hotel. The hotel is identified by hotel Id. It also contains contact number, hotel name, hotel license number, hotel location and hotel cost. The amount of reservation is stored into the system. Customer books package. A customer must book a package and a customer cannot book more than one package. Package is identified by Package Id. It also contains Cost, Package name and Destinations. The amount of booking a package is stored into the system. Customer also schedules transport. Customer must schedule a transport and Customer cannot schedule more than once. Transport is identified by transport no. The system also stores ticket cost, arrival time, departure time, departure location and arrival location. Transport is generalized by train, bus, and flight. The amount of scheduling a transport is also stored into the system. After finishing the booking process, the customer makes payment. Payment is generalized by card payment and Mobile banking. Mobile banking is generalized by Nogod and Bkash. Card payments stores card number, card name, expiration date and cvc no. Payment system stores Payment id and amount. Lastly, Customer gives review. which is identified by review id, rating, comments, date.

A screenshot of a computer screen

Description automatically generated

3.Er Diagram:

Figure: Er Diagram of Tripify

4.Normalization:

Reserve Relation:

**UNF:** cus\_id,name,fname,lname,nid,address,hno,city,street,mob\_no,email,h\_id,h\_name,h\_loc,  
h\_contact, h\_license,h\_cost

**1NF:** cus\_id,name,fname,lname,nid,address,hno,city,street,mob\_no,email,h\_id,h\_name,h\_loc,  
h\_contact, h\_license,h\_cost

**2NF:**

1. cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email
2. h\_id (PK), h\_name, h\_loc, h\_contact, h\_license,h\_cost
3. cus\_id (PK), h\_id (FK),r\_amm

**3NF:** Same as 2NF

Schedules Relation:

**UNF:** cus\_id, name, fname, lname, nid, address, hno, city, street, mob\_no, email, t\_no, dept\_loc, arr\_loc, dept\_time, arr\_time, t\_cost

**1NF:** cus\_id, name, fname, lname, nid, address, hno, city, street, mob\_no, email, t\_no, dept\_loc, arr\_loc, dept\_time, arr\_time, t\_cost

**2NF:**

1. t\_no(PK), dept\_loc, arr\_loc, dept\_time, arr\_time, t\_cost
2. cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email
3. cus\_id (PK), t\_no(FK),s\_amm

**3NF:** Same as 2NF

Books Relation:

**UNF:** cus\_id, name, fname, lname, nid, address, hno, city, street, mob\_no, email, p\_id, p\_name, destination, p\_cost

**1NF:** cus\_id, name, fname, lname, nid, address, hno, city, street, mob\_no, email, p\_id, p\_name, destination, p\_cost

**2NF:**

1. p\_id(PK), p\_name, destination, p\_cost
2. cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email
3. cus\_id (PK), p\_id(FK),b\_amm

**3NF:** Same as 2NF

Makes Relation:

**UNF:** cus\_id, name, fname, lname, nid, address, hno, city, street, mob\_no, email, pay\_id, amount, co\_name, c\_no, cvc, expire.

**1NF:** cus\_id, name, fname, lname, nid, address, hno, city, street, mob\_no, email, pay\_id, amount, co\_name, c\_no, cvc, expire.

**2NF:**

1. cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email
2. pay\_id(PK), amount, co\_name, c\_no, cvc, expire
3. cus\_id (PK), pay\_id (FK)

**3NF:** Same as 2NF

Gives Relation:

**UNF:** cus\_id, name, fname, lname, nid, address, hno, city, street, mob\_no, email, rev\_id, rating, r\_date, comment

**1NF:** cus\_id, name, fname, lname, nid, address, hno, city, street, mob\_no, email, rev\_id, rating, r\_date, comment

**2NF:**

1. cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email
2. rev\_id(PK), rating, r\_date, comment
3. cus\_id (PK), rev\_id (FK)

**3NF:** Same as 2NF

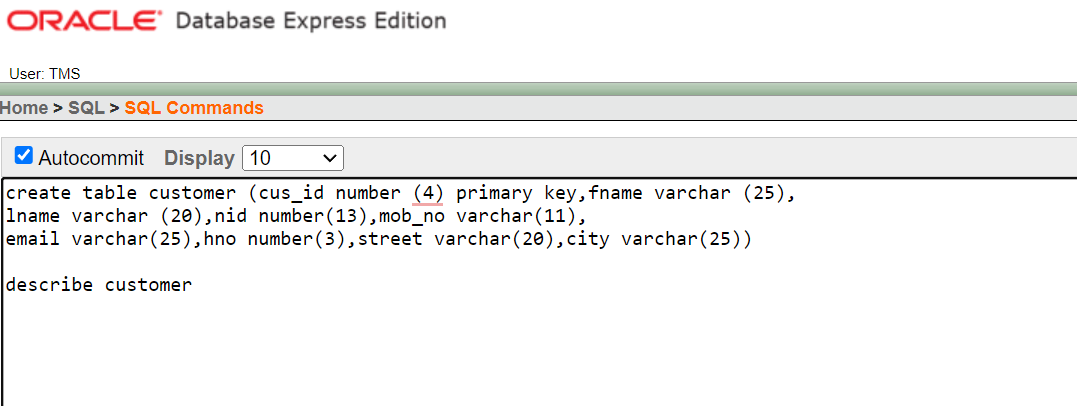
5. Finalization:

1. cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email
2. h\_id (PK), h\_name, h\_loc, h\_contact, h\_license,h\_cost
3. cus\_id (PK), h\_id (FK),r\_amm
4. t\_no(PK), dept\_loc, arr\_loc, dept\_time, arr\_time, t\_cost
5. ~~cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email~~ **×**
6. cus\_id (PK), t\_no(FK),s\_amm
7. p\_id(PK), p\_name, destination, p\_cost
8. cus\_id (PK), p\_id(FK),b\_amm
9. ~~cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email~~ **×**
10. ~~cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email~~ **×**
11. pay\_id(PK), amount, co\_name, c\_no, cvc, expire
12. cus\_id (PK), pay\_id (FK)
13. ~~cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email~~ **×**
14. rev\_id(PK), rating, r\_date, comment
15. cus\_id (PK), rev\_id (FK)

Final Tables:

1. cus\_id (PK), fname, lname, nid, hno, city, street, mob\_no, email (Customer)
2. h\_id (PK), h\_name, h\_loc, h\_contact, h\_license, h\_cost (Hotel)
3. cus\_id (PK), h\_id (FK),r\_amm (Reserves)
4. t\_no(PK), dept\_loc, arr\_loc, dept\_time, arr\_time, t\_cost (Transport)
5. cus\_id (PK), t\_no(FK),s\_amm (Schedules)
6. p\_id(PK), p\_name, destination, p\_cost (Package)
7. cus\_id (PK), p\_id(FK),b\_amm (Books)
8. pay\_id(PK), amount, co\_name, c\_no, cvc, expire (Payment)
9. cus\_id (PK), pay\_id (FK) (Makes)
10. rev\_id(PK), rating, r\_date, comment (Review)
11. cus\_id (PK), rev\_id (FK) (Gives)

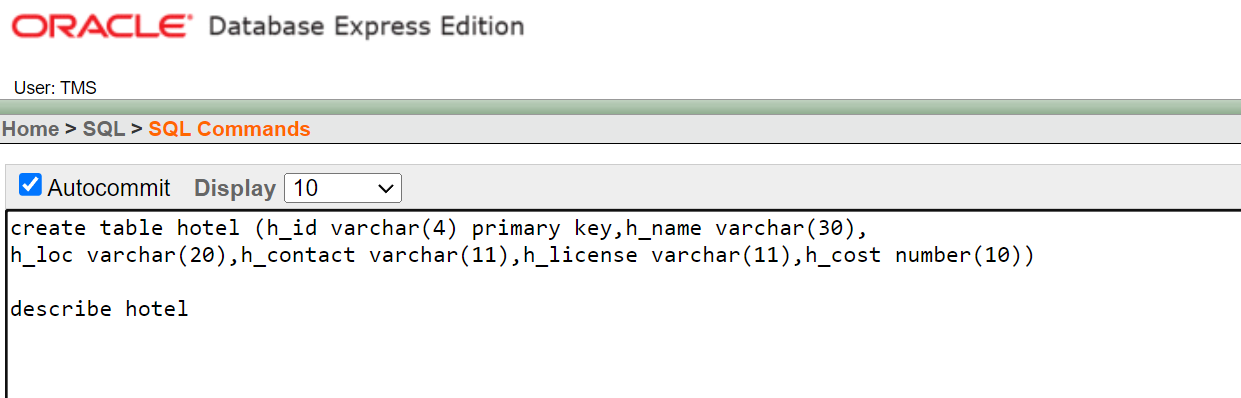
6.Table Creation:

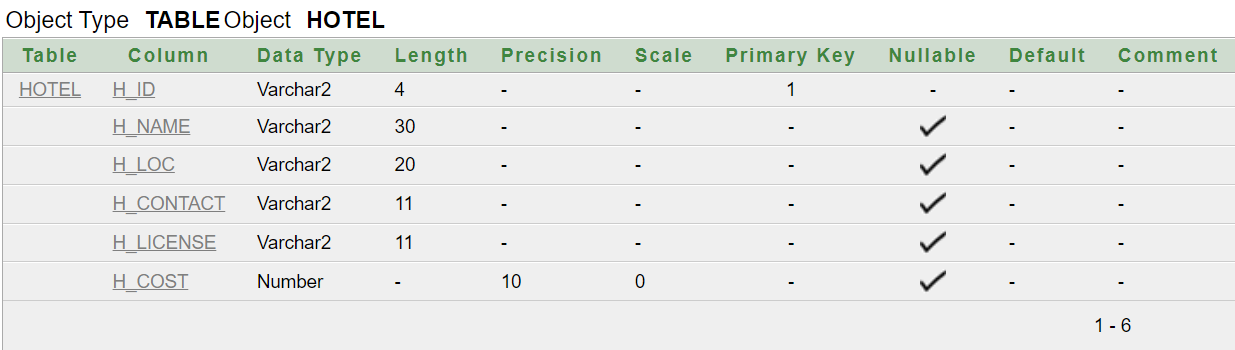


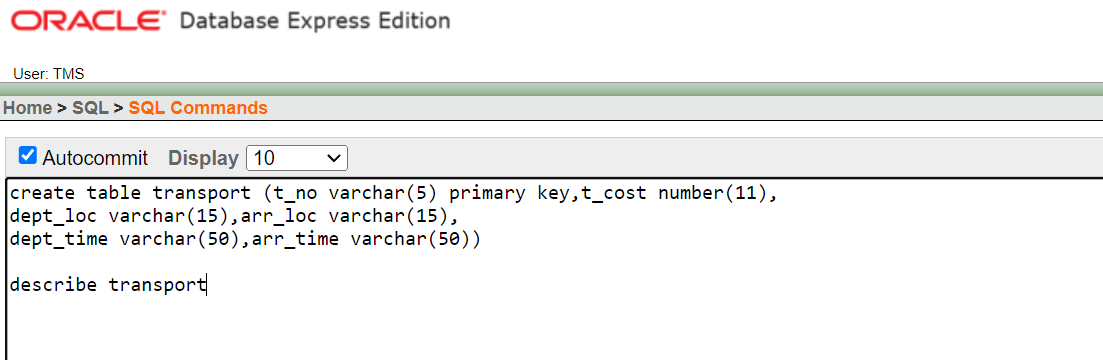
A screenshot of a computer

Description automatically generated

Figure 6.1: Command & Table of Customer



Figure 6.2: Command & Table of Hotel



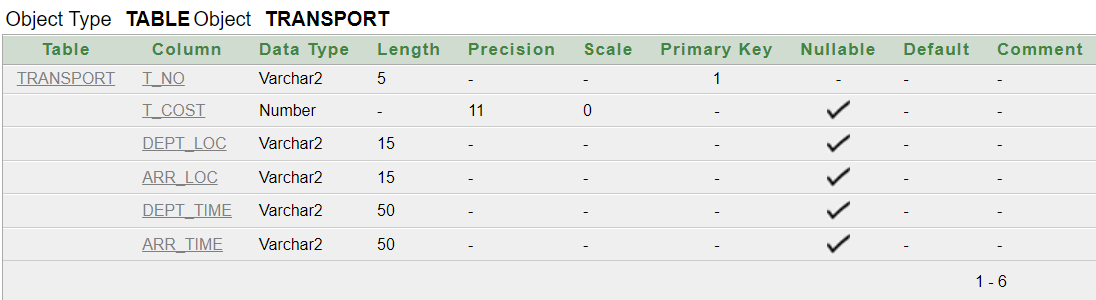
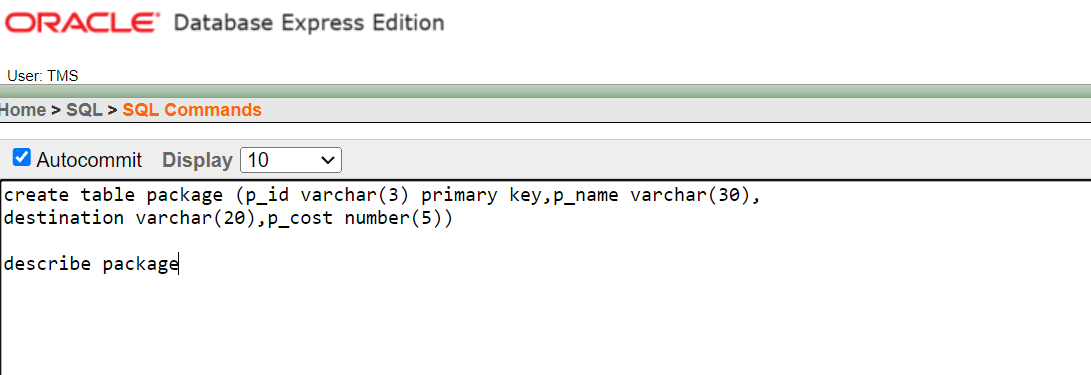


Figure 6.3: Command & Table of Transport

A screenshot of a computer

Description automatically generated

Figure 6.4: Command & Table of Package

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure 6.5: Command & Table of Payment

A screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure 6.6: Command & Table of Review

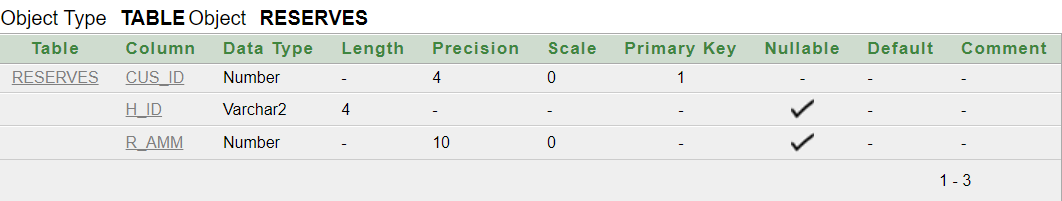
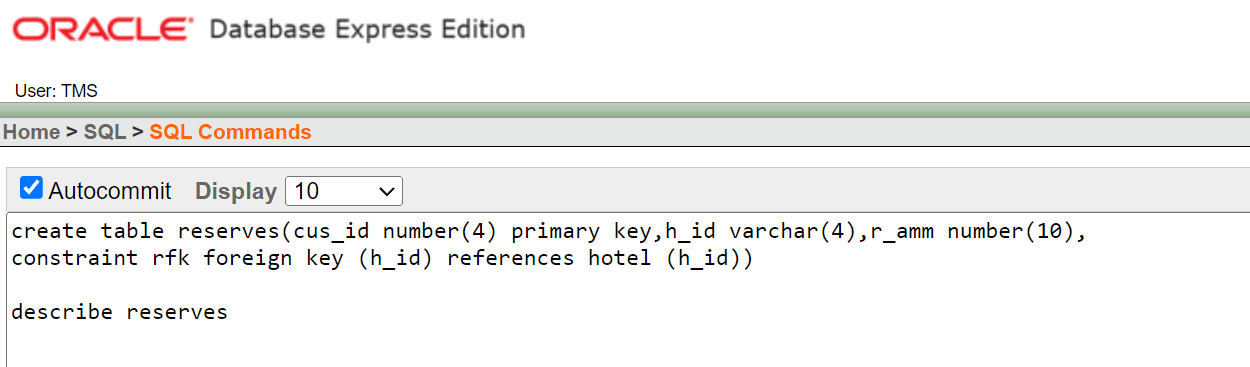


Figure 6.7: Command & Table of Reserves

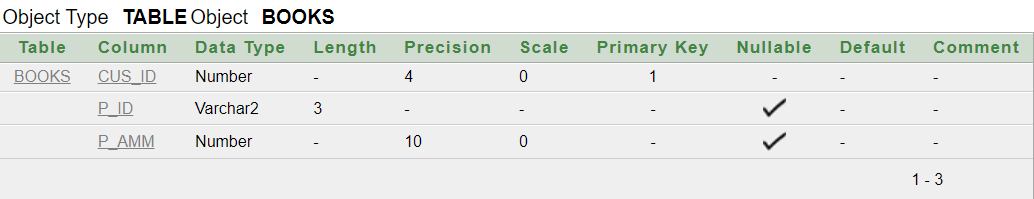
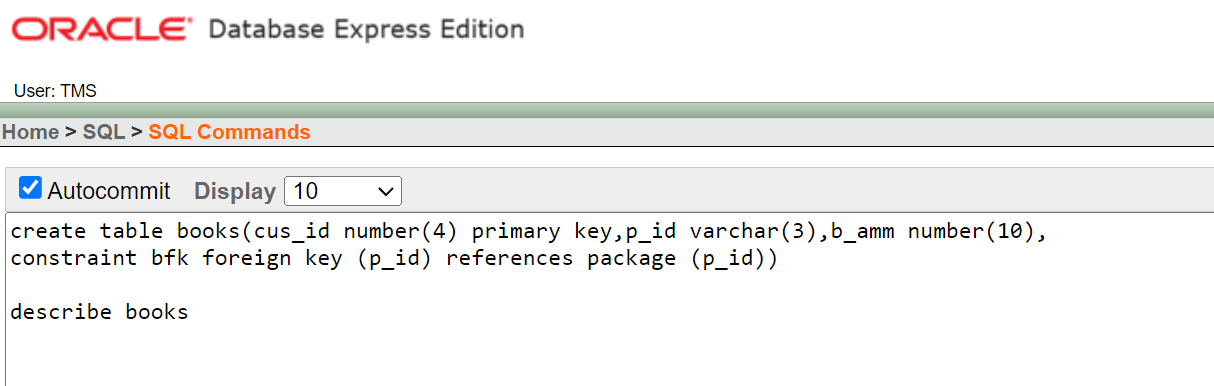


Figure 6.8: Command & Table of Books

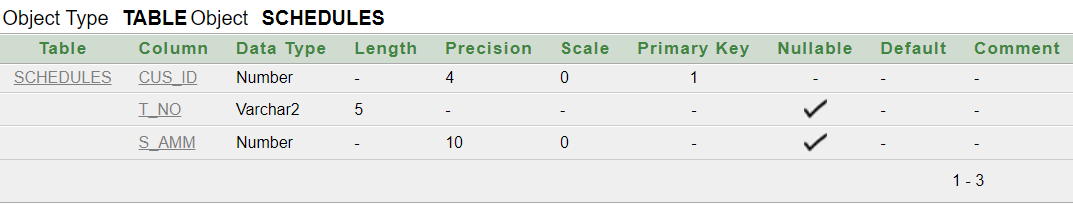
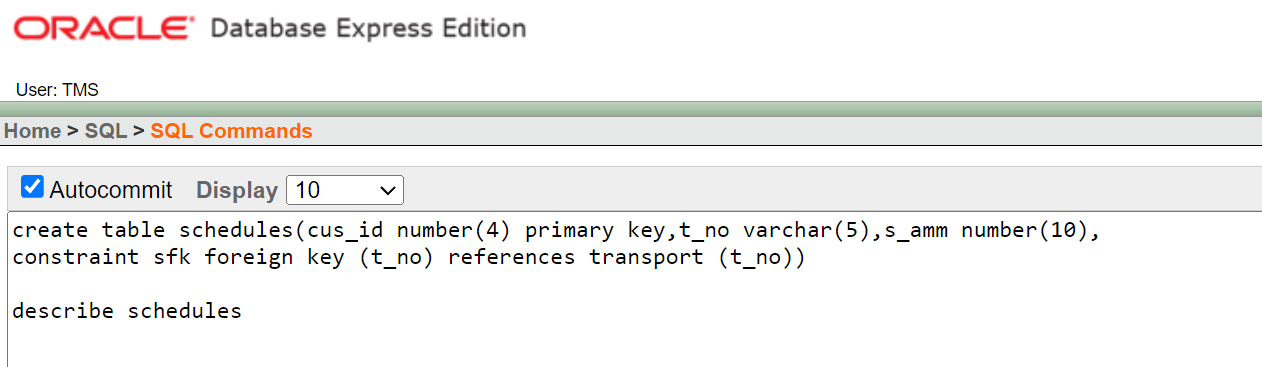


Figure 6.9: Command & Table of Schedules

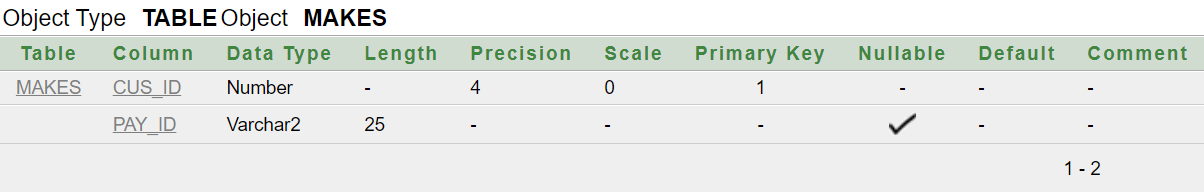
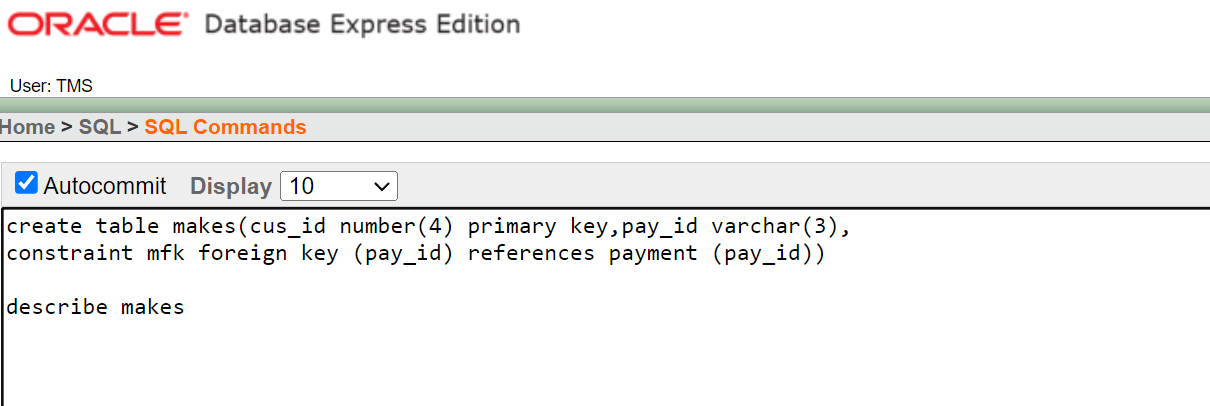


Figure 6.10: Command & Table of Makes

A screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure 6.11: Command & Table of Gives

7.DATA INSERTION: A screenshot of a phone

Description automatically generated

Figure 7.1 : Customer Table

A screenshot of a computer

Description automatically generated

Figure 7.2 : Hotel Table

A screenshot of a computer

Description automatically generated

Figure 7.3 : Transport Table

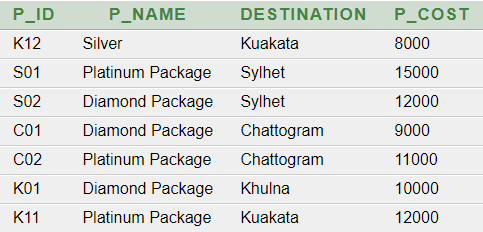


Figure 7.4 : Package Table

A screenshot of a computer

Description automatically generated

Figure 7.5 : Payment Table

A screenshot of a date

Description automatically generated

Figure 7.6 : Review Table

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure 7.7 : Reserves Table Figure 7.8 : Books Table

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure 7.9 : Schedules Table Figure 7.10 : Makes Table

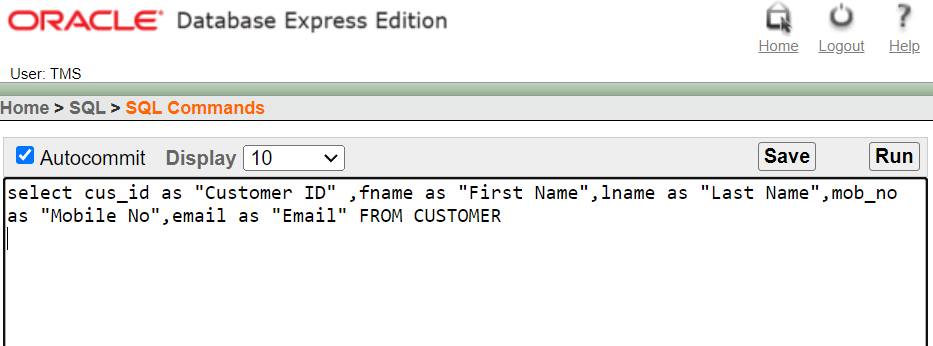
A screenshot of a computer

Description automatically generated

Figure 7.11 : Gives Table

8.Query Test:

A. Simple Query:  
Show the Customer ID, First Name, Last Name , Mobile Number , Email from customer table.



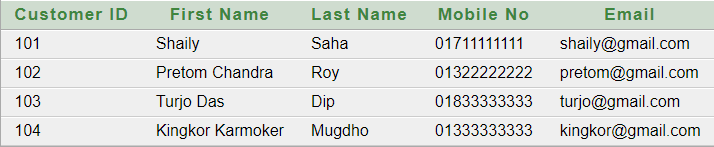
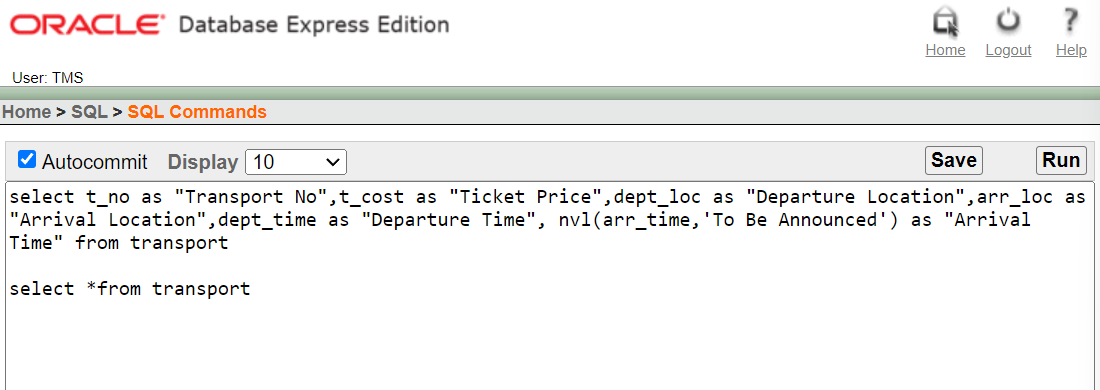


Fig: Simple Query with Result

B. Query with A Single Row Function:

Show all data from transport and replace the null value of the arrival time with the string “To Be Announced”.

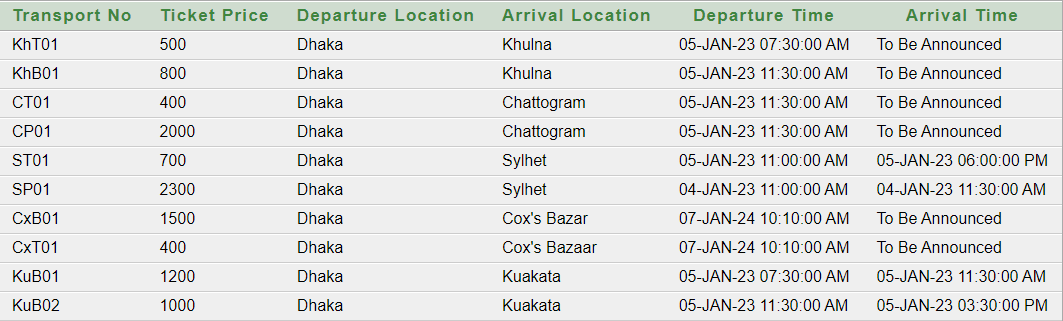


Fig: Query with A Single Row Function

C. Query with A Multiple Row Function / Aggregate Function:

Show the average cost and total count of packages for each destination in the package table. A screenshot of a computer

Description automatically generated

A screenshot of a graph

Description automatically generated

Fig: Query with a Multiple row function/ Aggregate function with result

D. Single Row Subquery and Multiple Row Subquery:

Single Row Subquery:

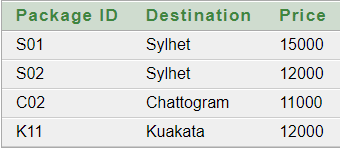
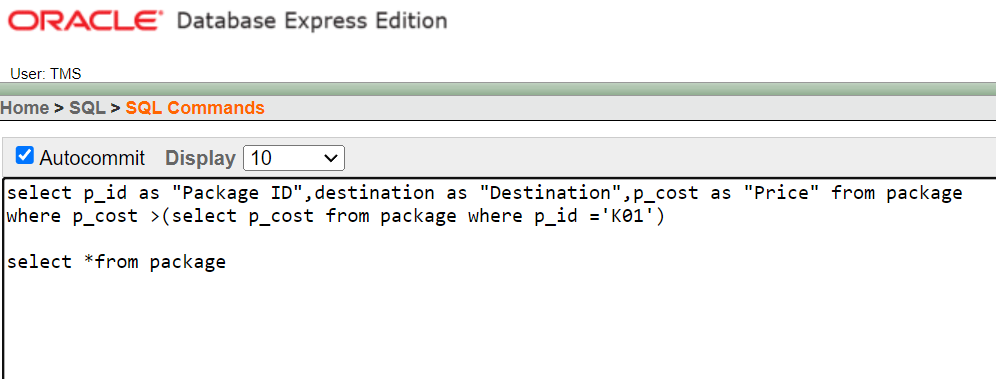
Show the package IDs, destinations, and costs of packages that have a cost higher than the package with ID 'K01'  


Fig: Single Row Sub Query with Result

Multiple Row Subquery:

Show transport numbers, costs, departure locations (DEPT\_LOC), and arrival locations (ARR\_LOC) for transports with costs greater than all the transports arriving at 'Kuakata'.  
A screenshot of a computer

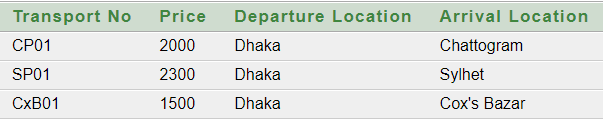
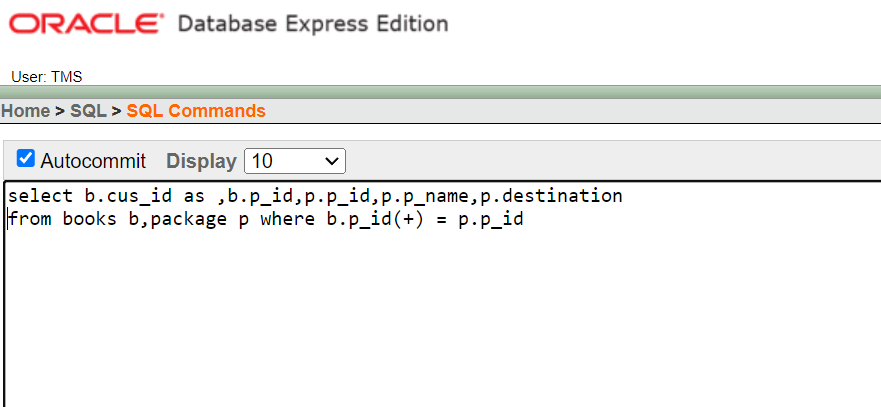
Description automatically generated

Fig: Multiple Row Sub Query with Result

E. Any 2 Kinds of Joining:

Outer Joining:

Perform an outer join with the Books and Package table, showing the customer IDs, package IDs, package names, and destinations.

Fig: Outer Join with Result

Equi Joining:

Perform an equi join with the hotel and reserve table, showing the customer IDs, Hotel IDs, Hotel names, and Hotel location.

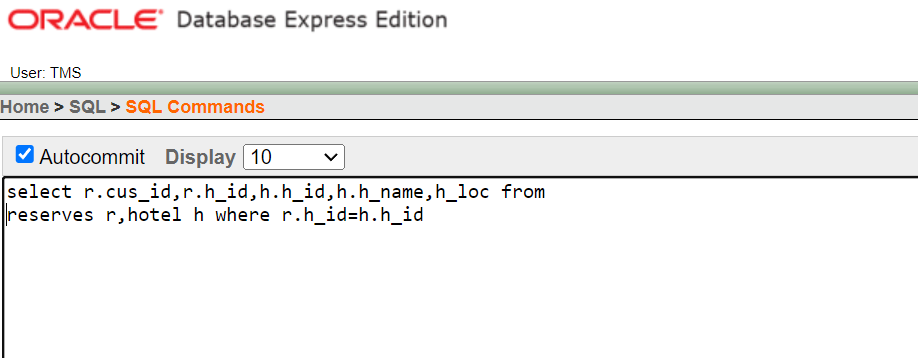


Fig: Equi Join with Result

F. View:

Simple View:

Create a simple view showing the Customer ID, First Name, Last Name, Mobile Number, Email from customer table.

A screenshot of a computer

Description automatically generated

Fig F.1: Command Of Simple View

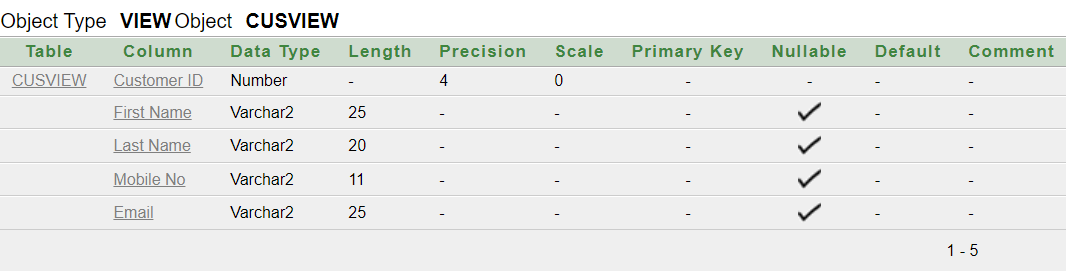


Fig F.2: Description Of Simple View

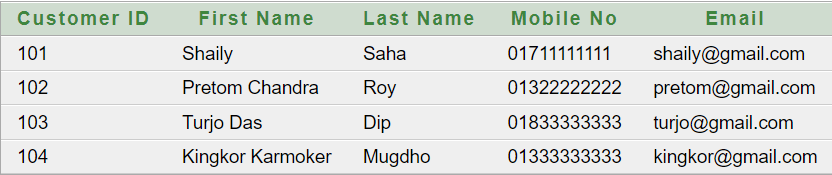


Fig F.3: Result Of Simple View

Complex View:

Create a complex view with euqi-join between book and package showing the customer id, package id, package name, destination with the results sorted by customer ID

A screenshot of a computer

Description automatically generated

Fig F.4: Command Of Simple View

A screenshot of a computer

Description automatically generated

Fig F.5: Description Of Simple View

A close-up of a computer screen

Description automatically generated

Fig F.6: Result Of Simple View

9.Database Connection:

PRETOM CHANDRA ROY 22-48556-3

Connection No 1:

This report details the implementation of Java Database Connectivity (JDBC) in the context of the Travel and Tourism Management System. The objective is to connect to the MySQL database and retrieve information from the "Package" table. The code is executed within the IntelliJ IDEA integrated development environment.

The Java program utilizes JDBC to establish a connection to the MySQL database named "Travel and Tourism Management System." It specifically interacts with the "Package" table, extracting data for display. The code follows the standard JDBC workflow by loading the MySQL JDBC driver, creating a connection, and executing a SELECT query on the "Package" table. The retrieved information, including package details such as name, description, destination, and price, is then printed to the console.

The development environment for this project is IntelliJ IDEA. The code is structured with a main class named `Conn`, which encapsulates the database connectivity logic. While the code successfully achieves its goal of displaying package information, it employs deprecated practices, such as not using try-with-resources for resource management. Additionally, the usage of an empty password for the database connection raises security concerns. It is recommended to adopt more modern and secure coding practices, including explicit resource closure and avoiding deprecated methods.

In summary, the Java program successfully connects to the "Travel and Tourism Management System" database, retrieves data from the "Package" table, and displays the details of various travel packages.

A screenshot of a computer

Description automatically generated

Figure 9.1.1: Description of Package Table from MySQL

A screenshot of a computer

Description automatically generatedFigure 9.1.2: Package Table from MySQL

A screenshot of a computer

Description automatically generatedFigure 9.1.3: Code for the connection between Java Program and Package Table

A screen shot of a computer

Description automatically generated

Figure 9.1.4: Output of the Java Code

Connection No 2:

TURJO DAS DIP 22-48558-3

The provided Java program serves as a part of a **‘Travel & Tourism Management System’**, specifically focusing on retrieving and displaying hotel details from a MySQL database. I use **‘XAMPP Control Panel’** for retrieving data from MySQL database and **‘IntelliJ IDEA’** to write the java code. The program establishes a database connection, executes a query to fetch hotel information, and then formats and prints the results in a tabular format.

A screenshot of a computer

Description automatically generatedFirstly, I use an import statement that brings in all the classes and interfaces from the java.sql package into the current Java source file. Then I write the basic syntax of a java program. Between a ‘try-catch’ block the **Connection** object (**con**) is created using the **DriverManager.getConnection** method, establishing a connection to the MySQL database named "**tms1**." The connection is made to the local server at port **3306**, with the username "**root**" and an **empty password**. Then A **Statement** object (**st**) is created from the connection to execute SQL queries. The **executeQuery** method is used to execute the SQL query (select \* from hotel) to retrieve all records from the "**hotel**" table. The retrieved data is stored in a **ResultSet** object (**rs**), and a formatted table header is printed to the console. A **while loop** iterates through the **ResultSet** to fetch each row of hotel details. The **printf** method is utilized to format and print the hotel details in a tabular format, including columns for HID, Name, Location, Price, Contact, and License No. The program ensures proper resource management by closing the **ResultSet**, **Statement**, and **Connection** objects within a **try-catch** block to handle any potential exceptions. The **try-catch** block is used to catch and handle any exceptions that may occur during the execution of the code, ensuring graceful error handling.

Figure 9.2.1: Description of Hotel Table from MySQL

A screenshot of a computer

Description automatically generated

Figure 9.2.2: Hotel Table from MySQL

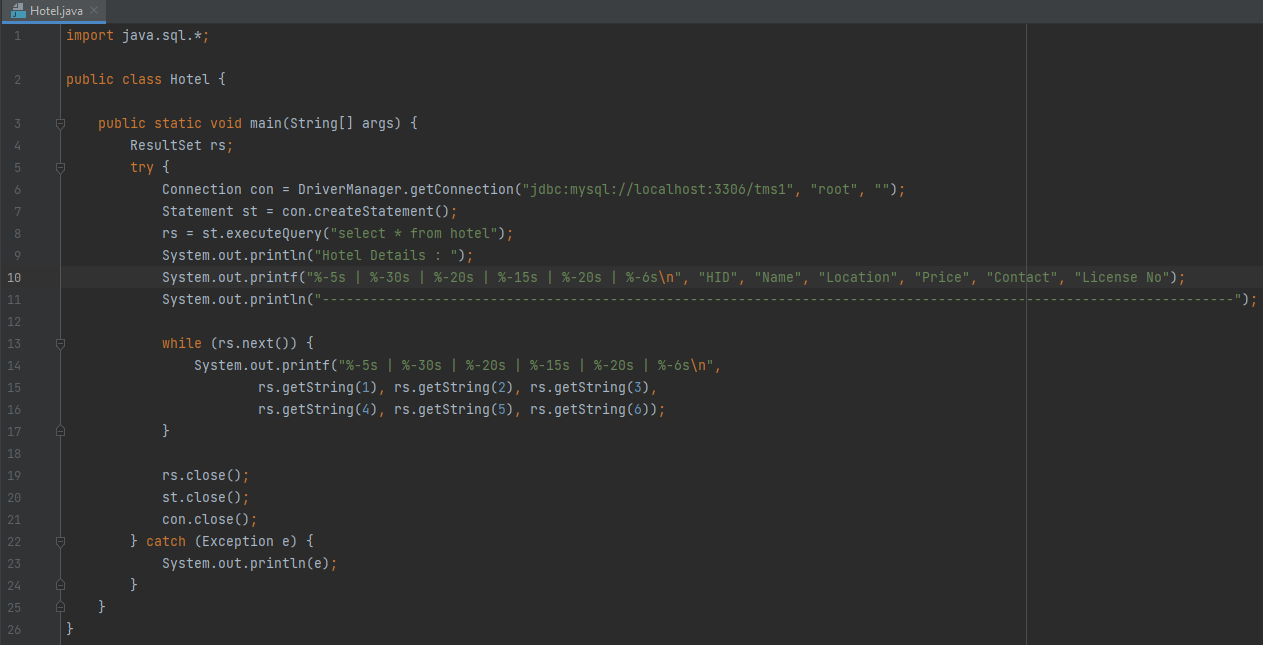


Figure 9.2.3: Code for the connection between Java Program and Hotel Table

A screen shot of a computer program

Description automatically generated

Figure 9.2.4: Output of the Java Code

10.Conclusion:

**“TRIPIFY – A TRAVEL & TOURISAM MANAGEMENT SYSTEM”** using oracle 10g has proven to be an efficient solution that is designed with the primary objective of enhancing the overall travel experience for all users. By efficiently managing bookings, providing real-time information, and offering personalized recommendations, the system aims to streamline the travel process, ensuring convenience and satisfaction for every traveler. Through its user-friendly interface and comprehensive features, the database serves as a valuable tool in promoting seamless journeys and creating a positive impact on the travel and tourism industry.