Dhirubhai Ambani
Institute of Information and Communication Technology

Tourist Agencies Management Database

Team ID: S4_T4

IT214: Database Management System
Assign By Prof. Minal Bhise

Mentor TA: Vaishnavi

Prepared By:
201901075 - Shreyas Sakariya
201901102 - Ronak Parmar
201901108 - Ankit Rathod
201901116 - Kishan Gajnotar

Table of contents

Section1: The final version of SRS

1.1 Introduction

Purpose

In today's world, every person must need a break from their boring lives to spend some time traveling with family and friends. In the present system, a customer has to approach various agencies to find details of places and to book tickets which requires a lot of time and effort. But sometimes they didn't get their requirements trip, satisfactory hotels, transport facilities, and destination brochure. Thus we create a project on the Tourist agencies Management Database which helps agencies and provide a better experience to travelers and make their journey satisfactory.

The purpose of the project is to design a system that automates the processes and activities of a travel and tourism agency and also to design a system in which travelers can get all information about traveler agencies and all packages on his requirement. Travelers no need to visit every agency they will get all information from their homes. This can be utilized for both business trips and leisure trips.

Intended Audience and Reading Suggestions

This computer program is additionally valuable for the clients specifically getting to the website.

The intended audience is,

- Travel agencies and travelers
- Project managers
- Users
- Testers
- Interested developers

Product Scope

The Tourism Management System project can prove to be immensely useful for travelers and travel agents with management facilities. According to the statistics provided by the ministry of tourism, there are lots of domestic travelers agencies in our country but they have no software that can provide all information regarding their package to travelers. So by using this software program many travelers who need a comfortable and affordable trip can find their preferred option according to their budget and interest. Also, they can provide relevant information about places situations like they are safe to visit in the covid situation, how is the weather of that place, how many people are going to visit their, etc.

Description

We are designing software that keeps track of various travel places, service availability, and traveling expenses and devising a comprehensive travel strategy.so, it will help give travelers the best package according to their interests and budget.

This covid pandemic has impacted lots in our life since last 2 years. Most people can't go outside from their homes due to this pandemic. But now the covid cases are decreasing and new cases are not coming. so the people started to travel outside. In fact, they started traveling all over India. So Tourist agencies Management Database comes into the

picture where people can find their travel package according to their requirement and make a trip with their family or friends.

The Tourist agencies Management Database will help all travelers to find different packages and book for the same if they're interested. and it also helps the agencies to find all data about travelers. According to the current situation, our management system includes a covid situation at that place so that people can be aware of the covid conditions before visiting that place.

Workflow

In the old manual system, all the records are not kept perfectly because all the work is done on paper, so keeping up-to-date details of the trip, timings of bookings, seat availability for reservation, vehicles, or rooms/hotels availability is not done. Which is the time taking system and needs a lot of manual calculation to keep it update which also can lead to a mistake in calculation.

In this project we create a system that provides information regarding tourist agencies and their package.first, we store all data of our agencies and all about packages in our database then we create a one google form and then give it to the user. He will fill that form and from that, according to his interest and preference we will give him all data, packages and all other information of our agencies. That is the way our system works.

Design and Implementation Constraints

The application should be portable, safe, and can not be accessible by unauthorized users. The application should be well functionality and the application interface should not be complex and it's easy to understand for every people. Time is the very essential point of designing the application. Response time of application should be very less and it should not give any error during service. It has a large amount capacity to store users' data. Also, the application should work on every operating system. The application should be well maintained or up to date.

1.2 Document the Requirements Collection/ Fact-Finding Phase

Background Readings

So we went through the internet finding articles related to the Tourist management and found some issues in which people are suffering in finding a safe place, good hotels in the current covid situation and management also need to lot's of work to maintain all this stuff.

Manual Paper Work
In the old system, all the records are not kept perfectly because all the work is done manually, so keeping up-to-date details of the trip, timings of bookings, seat availability for reservation, vehicles, or rooms/hotels availability is not done. Which is a time taking system and needs a lot of manual calculation to keep it updated which also can lead to a mistake in calculation.

Software implemented system
The solution to the manual paperwork problem is to use a software-implemented system that is more efficient than the manual system. This system is highly automated and makes the traveling agencies' work much easier and flexible. The system doesn't need to do all base tasks manually, such as the forms transactions, and reports which are added advantages. This system is a completely computer-based application that uses concepts of DBMS. Thousands of records can be searched and displayed without taking any significant time and users can use it very easily which leads to the increased trust of the customer.

References

- https://projectsgeek.com/2016/01/tourism-management-system.html
- https://www.gujarattourism.com/
- https://ijcsmc.com/docs/papers/October2019/V8I10201903.pdf
- https://www.touristlink.com/
- https://sites.google.com/site/ignoubcafinalyearprojects/project-report/tours-and-travel-management-system-project-report

Requirements

FUNCTIONAL REQUIREMENTS:

1. Administrator Module: This module provides administrator-related functionality. The administrator manages all information about travelers and has all rights to add, delete, edit and view the data related to travelers, places, routes, restaurants etc.

2. Travels Module: This module provides the details of various packages based on information provided by travelers. A user can select the appropriate packages depending on convenience and accessibility.

3. Routes Module: This module provides information related to various routes connecting sources and destinations. For each route, information such as source, destination, fare, reservation details, pick-up points, etc is provided. Only administrators can add, delete, edit and manage the data. Users can only view the information.

4. Reservation Module: This module provides functionalities that allow a user to book tickets for train, plane, and bus. This module also provides cancellation of previously booked tickets. The module maintains the details of all reservations made so far and allows administrators to either confirm or reject the bookings.

5. Testimonials Module: Users of this application can post their opinions, complaints, and suggestions regarding this portal and services to the administrator. Accordingly, the administrator can take various steps to act on the complaints and suggestions.

1.3 Interview/s –

Interview 1
System: Citizen

Project Reference: SF/SJ/2003/12

Interviewee: 1)Rajesh Gajnotar Designation: Govt.Teacher

Interviewer: 1) Ankit Rathod Designation: Student AT DAIICT
2) Ronak Parmar Designation: Student AT DAIICT
3) Shreyas Sakariya Designation: Student AT DAIICT
4) Kishan Gajnotar Designation: Student AT DAIICT

Date: 8/10/2021 Time: 21:00

Duration: 1 hour 25 minutes Place: Virtual

Purpose of Interview:

- Gathering information about his travel experience and his interest. Also, we have discussed his future plan and places that he will be going to travel in the covid 19 situation.
- Get some idea about how we can arrange some trip(school trip) that is very beneficial to our system and users.
- How has he experienced other traveling websites and what is the suggestion for our current system?

Agenda:

- List all the requirements regarding safety concerns in this covid19 situation
- Also, list all requirements about his trip that he wants.
- Classify them according to the requirements

Interview 2

System: Citizen

Project Reference: SF/SJ/2003/12

Interviewee: 1) Jenish Rathod Designation: IT engineer

Interviewer:
1) Shreyas Sakariya Designation: Student AT DAIICT
2) Kishan Gajnotar Designation: Student AT DAIICT

Date: 5/10/2021 Time: 17:30

Duration: 1 hour 35 minutes Place: Virtual

Purpose of Interview:

- Gathering information about his travel experience and his interest. Also, we have discussed his future plan and places that he will be going to travel.

Agenda:

- List all the information
- Make a list of places that people generally prefer to visit.
- Classify them according to the requirements

Documents to be brought to the interview:

- A website that indicates current information of different Places.

Interview 3

System: Citizen

Project Reference: SF/SJ/2003/12

Interviewee: 1)Hiten Padaliya Designation: Student

Interviewer: 1) Ankit Rathod Designation: Student AT DAIICT
2) Ronak Parmar Designation: Student AT DAIICT

Date: 6/10/2021 Time: 17:30

Duration: 1 hour 30 minutes Place: Virtual

Purpose of Interview:

- Gathering information about his travel experience and his interest. Also, we have discussed his future plan and places that he will be going to travel in the covid 19 situation.

Agenda:

- List all the requirements regarding safety concerns in this covid19 situation
- List all the information
- Classify them according to the requirements

Documents to be brought to the interview:

- A website that indicates current information of different Places.

Combined Requirements gathered from Response/s.

We have gathered all the information from interviews we noted the following Requirements

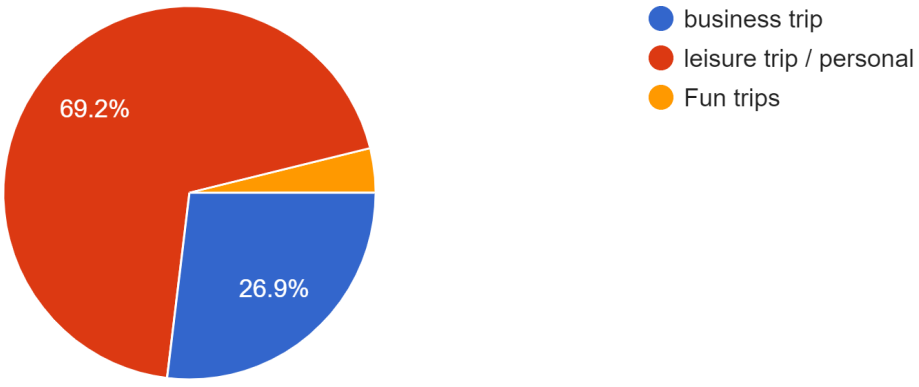
- Transportation services
- Medical emergency
- Reliable package
- Covid precaution
- Guide services
- Photographer
- Good restaurants

Also, we need to provide a good security policy so that the user data will be safe and users can easily trust our system.

1.4 Questionnaire/s

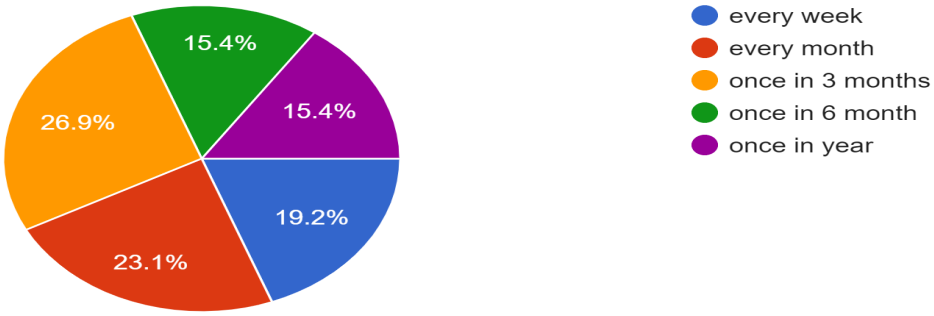
1) Which kind of trip do you have the most?

A. business trip
B. leisure trip / personal



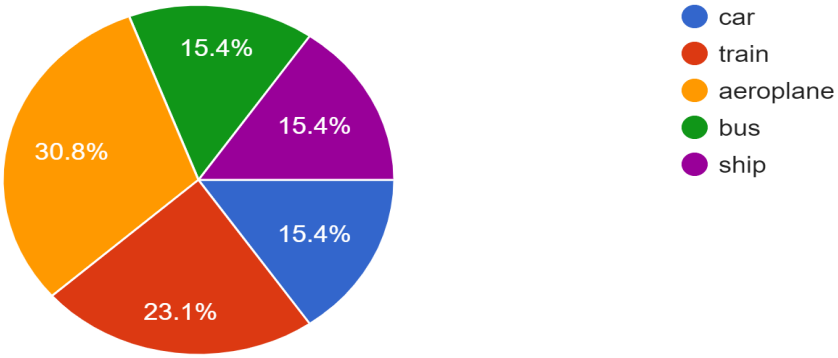
2)How Often do you travel to the outstation?

A. every week
B. every month
C. once in 3 months
D. once in 6 month
E. once in year



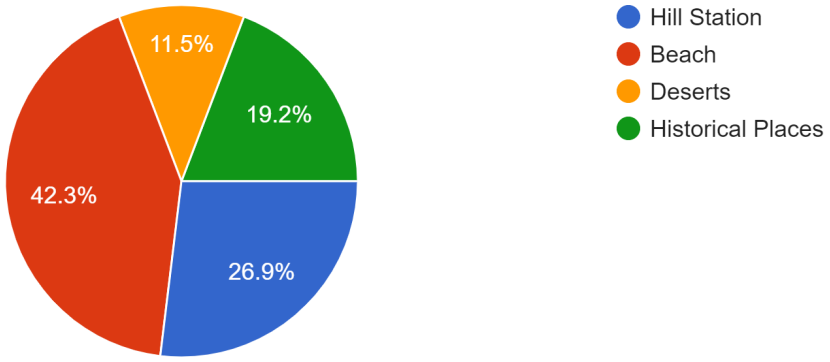
3)What mode of transportation do you prefer most of the time?

A. Car
B. Train
C. Airplane
D. Bus
E. Ship



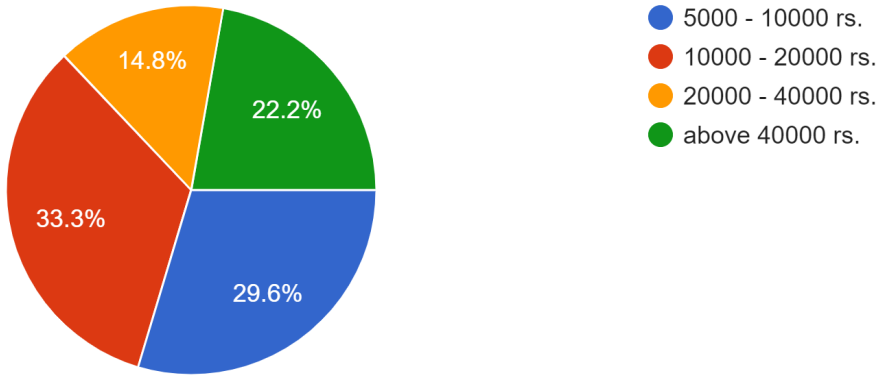
4) What kind of place do you prefer most?

A. Hill Station
B. Beach
C. Deserts
D. Historical Places



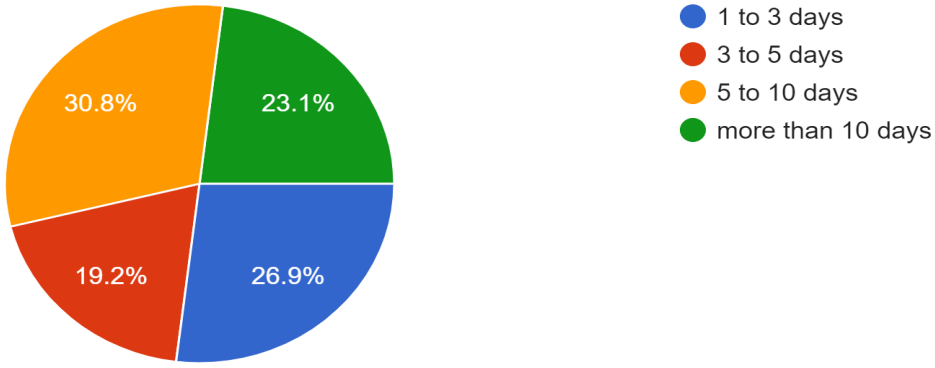
5)What is the package of your trip around?

A. 5000 - 10000 rs.
B. 10000 - 20000 rs.
C. 20000 - 40000 rs.
D. Above 40000 rs.



6)most common length of your trip

A. 1 to 3 days
B. 3 to 5 days
C. 5 to 10 days
D. more than 10 days



Requirements from the questionnaire

- In the covid-19 situation, people like to travel to a safe place which is given by the minister of government. So we have to focus on providing safe places.
- At this time people prefer airplanes to travel most so we have to ensure that we provide good service and best offer to them.
- Most people like to travel with family and friends so we provide the best offer on a family package which reduces travel costs.
- Most people like to travel at Hill station and Beach for peaceful traveling so we research more on this type of place and provide more places like this.
- Most people don't afford expensive traveling packages so we find more offers on train/bus traveling and include more low expenses packages with good services.
- Budget, destination weather, and covid-19 cases in that location are all significant factors in ensuring a safe and comfortable journey. As a result, our algorithm should filter these items and recommend the ideal trip.

1.5 Observation

- Observers can see the actual workflow during visits or personal experiences to add/suggest more requirements that might have been missed by other techniques like background reading, interviews, and questionnaires.

- We will give a good system which provides all functionalities of checking hotel details ,place description,transportation details and detailed information about package description with user understandable interface.

.

- We should ensure that the system gives the highest possible level of accuracy about hotel vacancy, place safety, and transportation details.

- In case of any emergency such as health deterioration, loss of any personal things, forgetting things to any place, etc. The emergency entity provides details such as Type, Contact of agency member, Address, etc.

- The system covers the details about the registration of users which can be registered by themselves by adding data like name, password, email id, and further details. After registration, they can sign in by their username and password.

- At some places the guidelines for the covid19 are not followed strictly, so we include all this type of detailed information in safety descriptions so at the tourist places no our customer suffers of any type of covid infection.

- Tickets can be issued for the user on the issue ticket page on the homepage of the user the certain booked packages only can be issued.

- The system is mainly based on admin. The system will check the admin username and password for authentication. After the verification for authorization, the admin can be able to precede the process. All works are done under his control.

- We will give a review section on our website where you share your experience with our agencies and list your suggestions and complain about our system so we look into our suggestion and make necessary steps on your complaint and provide you with the best experience .

1.6 Fact-Finding Chart

Objective	**Technique**	**Subject(s)**	**Time Commitment**
To get background on Tourist Agencies Management System	Background Reading	Articles, books, publications	2 day
Collect all ideas of their work and try to implement them in our system.	Interview	With Gov.Teacher / Social Worker	1 hour 25 min
Gathering information about his travel experience and his interest and Classify them according to the requirements	Interview	1 Citizen/IT engineer	1 hour 35 min
To establish additional requirements to our system	Interview	1 Citizen	1 hour 30 min
To know the opinion and interest of the people about traveling	Questionnaires/ Google forms	Shared with family and friends and others	2 days

1.7 List Requirements

- The administrator manages all information about travelers and has all
  rights to add, delete, edit and view the data related to travelers, places, routes, restaurants, etc.. the system will check the administrator's username and password for authentication so only the administrator has all rights.

- We will give a good system which provides all functionalities of checking hotel details ,place description,transportation details and detailed information about package description with user understandable interface.

- Systems need to provide a data security policy so that the user data will be safe and don't accessed by anyone. so users can easily trust our system.

- In case of any emergency such as health deterioration, loss of any personal things, forgetting things to any place, etc. Emergency entities provide details such as Type, Contact of agency member, Address, etc.

- In the covid-19 situation, people like to travel to a safe place which is given by the minister of government. So we have to focus on providing safe places.

- We should ensure that the system gives the highest possible level of accuracy about hotel vacancy, place safety, and transportation details.

- We have gathered all the information from interviews we noted the following Requirements
  - Transportation services
  - Covid precaution
  - Guide services
  - Photographer
  - Good restaurants

- The system provides the details of various packages based on information provided by travelers. A user can select the appropriate packages depending on convenience and accessibility.

- The system provides functionalities that allow a user to book tickets for trains, planes, and buses. This module also provides cancellation of previously booked tickets. The system maintains the details of all reservations made so far and allows administrators to either confirm or reject the bookings.

- The system provides information related to various routes connecting sources and destinations. For each route, information such as source, destination, fare, reservation details, pick-up points, etc is provided.

- Users of this application can post their opinions, complaints, and suggestions regarding this portal and services to the administrator. Accordingly, the administrator can take various steps to act on the complaints and suggestions.

- Performance Requirements
  - The database shall be able to accommodate a minimum of 10,000 records of customers.
  - The software shall support the use of multiple users at a time.

- Security Requirements
    - Some of the factors that are identified are to protect the software from accidental or malicious access.
    - Keep a specific log or history data sets
    - Check data integrity for critical variables
    - when the application crashed along with suggestions on how to prevent the error from occurring again.

1.8 User Classes and Characteristics

The system provides access to Visitors, Customers, Managers, and Administrators.

- **Visitor:** Users who are currently not logged in and have not registered to the website are visitors. Visitors can check the availability of flight reservations. In order to book the ticket, the visitor will need to log in with a valid username and password else register with the app and create an account.

- **Customer:** Customers can check the availability of flight reservations and also book tickets by making payments online. They can also view their booking details, payment details, etc.

- **Manager:** The person who manages the system, modifies/add, or delete the customer

- **Administrator**: The administrator can manage customer details, hotels, rooms, fight details, tour package details, etc. Also, the admin can view and print transaction reports, booking reports, cancellation reports, etc.

1.9 Operating Environment

Software Interfaces

- Operating System: Windows XP or Higher versions
- Front End :HTML, CSS and BOOTSTRAP
- Back End : Python Django with PostgreSQL

Hardware Interfaces

- Processor : Pentium IV
- Speed : 2.0 GHz above
- Hard Disk : 40 GB
- RAM : 512 MB
- CD Drive : 48x
- Input devices : Keyboard and mouse
- Monitor : Compatible monitor with 600 x 800 resolutions
- Internet : 100kbps above

External Interface Requirements

- The following sections will introduce the numerous requirements of the system from the point of view of different users and will introduce a number of decisions that have been made regarding implementation.
- It allows users to view quick reports.
- The user interface must be customizable by the administrator.
- All the modules provided with the software must fit into this graphical user interface and accomplish the standard defined.
- The user interface for this system will have to be simple and clear. Most importantly, the ages must be easy to read, easy to understand, and accessible.

1.10 Product Functions

- **Sightseeing Tours:** Tourists can enjoy sightseeing tours to any of the places listed in the Agencies Data File. Before that Customer has to make the booking by registering his/her name in the Data-File and informing the date of journey and No. of operations.

- **Flight ticket booking:** In this module, the customer can search source-destination, check flight timings, and book a ticket online.

- **Bus ticket reservation module:** In the bus ticket reservation module there has been a collection of buses, Customers can book the ticket by selecting the agency, date, and departure time of the bus.

- **Train ticket reservation module:** In the train reservation module customers can check train scheduled timings and they can book tickets online by entering Place, date, and time.

- **Hotel reservation:** After booking a train ticket, bus ticket, or flight ticket the system will provide the option to search hotels. Customers can reserve the hotels by selecting check-in date, check-out date, and hotel name.

- **Holiday package:** The customer can search tour packages by entering a destination, duration of travel, the month of travel, and the number of people.

- **Payment module:** This module allows customers to make payments after the reservation. The customer will receive a ticket receipt by mail after the payment.

- **Report Generation:** Details about the locations, hotels in that location, and final report on the journey fare.
- **My Account:** This module is for customers where customers can access the account module after login id and password. This module will display booked tickets, Billing receipts, ticket cancellation pages, logout, Etc.

- **Cancel booking:** Customer can cancel flight tickets, bus tickets, train tickets, or hotel bookings. The system refunds the paid amount after deducting cancellation charges.

- **Dashboard module:** This module allows the administrator of this site to manage the different aspects of ticket reservation-related activities. The administrator of the site can add or modify the records of flights, buses, trains, holiday packages, etc.

1.11 Privileges

Facilities accessible to Administrator

- Dashboard Module
- Insert, delete and update the record of any module
- Has access to the key database

Facilities accessible to the user

- Create an Account on site
- Flight ticket booking and cancellation
- Train ticket booking and cancellation
- Bus ticket booking and cancellation
- Holiday package selection
- Hotel reservation
- View generated report
- Payment method

1.12 Assumptions

- The user should operate on a computer or laptop along with an internet facility.
- Roles and responsibilities are already established.
- The administrator is already created.
- The coding should be error-free.
- The system should be user-friendly so it is easy to use for all users.
- The user must need to register himself first to use the system.
- The information of all users must be stored in a database that is accessible by the website.

1.13 Business Constraints

A business constraint is anything that interferes with the profitability of a company or business endeavor. Improving profitability requires the removal or reduction of business constraints. Common business constraints include time, financial concerns, and management.

Time Constraints
Time constraints include not only the amount of time required to complete a task but also the amount of time needed to obtain details about places, hotels, and transportation services (manage all traveling schedules). Once identified as a primary constraint, management can take steps to address time factors and improve business performance.

Financial Constraints
Financial factors are often limiting constraints for businesses. They can range from inadequate budget allocations to excessive salaries or overhead expenditures. for example: if more employees are needed, but the budget cannot accommodate additional salaries, growth is limited.

Management and Staffing
As businesses grow and change, their staffing and management need change, as well. This can constrain business growth and productivity when employees cannot adapt to new demands or when additional employees are needed but the capital to pay them is not yet available.

Section 2 : Noun Analysis

2.1 Final Problem Description

In this covid situation, most of the people are bored from staying home. But now the covid cases are decreasing and new cases are not coming. so the people started to travel outside. In fact, they started traveling all over India. So Tourist agencies Management Database comes into the picture where people can find their travel package according to their requirement and make a trip with their family or friends.

In the present system, a customer has to approach various agencies to find details of places and to book tickets which requires a lot of time and effort. But sometimes they didn't get their requirements for the trip like satisfactory hotels, transport facilities, and destination brochure. Thus we create a project on the Tourist agencies Management Database which helps agencies and provide a better experience to travelers and make their journey satisfactory.

In the old manual system, all the records are not kept perfectly because all the work is done on paper, so keeping up-to-date details of the trip, timings of bookings, seat availability for reservation, vehicles, or rooms/hotels availability is not done. Which is a time taking system and needs a lot of manual calculation to keep it updated which also can lead to a mistake in calculation. According to the statistics provided by the ministry of tourism, there are lots of travel agencies in our country but they don't have software that can provide us with all relevant information about tour packages to travelers.

That's why we created a Tourism Management System that can prove to be immense for travelers and travel agents with management facilities. So by using this software program many travelers who need a comfortable and affordable trip can find their preferred option according to their budget and interest. Also, they can provide relevant information about places that are safe to visit in the covid situation, how is the weather of that place, how many people are going to visit there, etc.

In this project, we create a system that provides information regarding tourist agencies and their packages. For that first, we store all data of our agencies and all about packages in our database then we create a one google form and then give it to the user. They will fill that form and from that, according to his interest and preference, we will give him all data, packages, and all other information about our agencies. That is the way our system works. This system should be portable, safe, and can not be accessible by unauthorized users. The system should be well functional and the system interface should not be complex and it's easy to understand for everyone. Response time of the system should be very less and it should not give any error during service. It has a large capacity to store users' data. Also, the system should work on every operating system. The system should be well maintained or up to date.

Our system provides access to Visitors, Customers, Managers, and Administrators. Users who are currently not logged in and have not registered to the website are visitors. Visitors can check the availability of flight reservations. In order to book the ticket, the visitor will need to log in with a valid username and password, else register with the app and create an account. Customers can check the availability of flight reservations and also book tickets by making payments online. They can also view their booking details, payment details, etc. The manager is the person who manages the system, modifies/adds, or deletes the customer. The administrator has access to the key database of our system and manages customer details, hotels, rooms, fight details, tour package details, etc. Also, the admin can view and print transaction reports, booking reports, cancellation reports, etc.

To make the system more efficient we created a google form to take surveys and also did some interviews. From all that we listed some requirements that can improve our system,

- The system provides the details of various packages based on information provided by travelers. A user can select the appropriate packages depending on convenience and accessibility. We should ensure that the system gives the highest possible level of accuracy about hotel vacancy, place safety, and transportation details.
- In the covid-19 situation, people like to travel to a safe place which is given by the minister of government. So we have to focus on providing safe places.

- In case of any emergency such as health deterioration, loss of any personal things, forgetting things to any place, etc. Emergency entities provide details such as Type, Contact of agency member, Address, etc.
- The system provides functionalities that allow a user to book tickets for trains, planes, and buses. This module also provides cancellation of previously booked tickets. The system maintains the details of all reservations made so far and allows administrators to either confirm or reject the bookings.
- Systems need to provide a data security policy so that the user data will be safe and don't access by anyone. so users can easily trust our system.
- We have gathered all the information from the interviews. We noted some requirements like Transportation services, Covid precaution, Guide services, Photographer, Good restaurants, Vehicle facility.
- The administrator manages all information about travelers and has all rights to add, delete, edit and view the data related to travelers, places, routes, restaurants, etc.. the system will check the administrator's username and password for authentication so only the administrator has all rights.
- The system provides information related to various routes connecting sources and destinations. For each route, information such as source, destination, fare, reservation details, pick-up points, etc is provided.
- Users of this application can post their opinions, complaints, and suggestions regarding this portal and services to the administrator. Accordingly, the administrator can take various steps to act on the complaints and suggestions.
- Performance Requirements are like the database shall be able to accommodate a minimum of 10,000 records of customers, the software shall support the use of multiple users at a time.
- Security Requirements: Some of the factors that are identified are to protect the software from accidental or malicious access. Utilize certain cryptographic techniques Keep a specific log or history data set. Assign certain functions to different modules. Check data integrity for critical variables. The software will include an error tracking log that will help the user understand what error occurred. When the application crashed along with suggestions on how to prevent the error from occurring again.

According to users requirements and by the observer our system has the following functionalities

- Sightseeing Tours: Tourists can enjoy sightseeing tours to any of the places listed in the Agencies Data File. Before that Customer has to make the booking by registering his/her name in the Data-File and informing the date of journey and No. of operations.
- Flight ticket booking: In this module, the customer can search source-destination, check flight timings, and book a ticket online.
- Bus ticket reservation module: In the bus ticket reservation module there has been a collection of buses, Customers can book the ticket by selecting the agency, date, and departure time of the bus.
- Train ticket reservation module: In the train reservation module customers can check train scheduled timings and they can book tickets online by entering Place, date, and time.
- Hotel reservation: After booking a train ticket, bus ticket, or flight ticket the system will provide the option to search hotels. Customers can reserve the hotels by selecting check-in date, check-out date, and hotel name.
- Holiday package: The customer can search tour packages by entering a destination, duration of travel, the month of travel, and the number of people.

- ➢ Payment module: This module allows customers to make payments after the reservation. The customer will receive a ticket receipt by mail after the payment.
- ➢ Report Generation: Details about the locations, hotels in that location, and final report on the journey fare.
- ➢ My Account: This module is for customers where customers can access the account module after login id and password. This module will display booked tickets, Billing receipts, ticket cancellation pages, logout, Etc.
- ➢ Cancel booking: Customers can cancel flight tickets, bus tickets, train tickets, or hotel bookings. The system refunds the paid amount after deducting cancellation charges.
- ➢ Dashboard module: This module allows the administrator of this site to manage the different aspects of ticket reservation-related activities. The administrator of the site can add or modify the records of flights, buses, trains, holiday packages, etc.

2.2 Noun (& Verb) Analysis

Table.1. Nouns & Verbs

Noun	Verb
tourist agencies	travel
covid cases	can
tourist	find
management database	make
travel package	present
present system	approach
various agencies	get
book tickets	like
satisfactory hotels	transport
transport facilities	create
check-in date	provide
old manual system	kept
maintenance	experience
seat availability	seat
hotel reservation	keep

manual calculation	lead
travel agencies	have
relevant information	prove
tour packages	allows
book tickets online	update
tourism management	need
travel agents	modify
management facilities	visit
software program	store
ticket reservation module	fight
relevant information	protect
covid situation	guide
visitors	going
google form	operating
system interface	regarding
response time	using
large capacity	sightseeing
covid situation	occurring
managers	depending
visitors	tracking
administrators	reached
customers	pick
flight reservations	registering
users	add

valid username	act
user data	utilize
administrator’s username	delete
reservation details	edit
pick-up points	reject
payments online	report
payment details	fight
various packages	receive
appropriate packages	enjoy
hotel vacancy	prevent
place safety	help
transportation details	assign
covid-19 situation	include
safe places	give
personal things	fill
health deterioration	access
emergency	check
cancellation charges	print
contact	select
agency member	appropriate
address	take
data security policy	charges

transportation services	order
guide	focus
good restaurants	view
vehicle	understand
users requirements	service
sightseeing tours	well
tourists	gives
flight ticket	confirm
customer	surveys
check flight timings	booked
ticket online	bored
ticket reservation module	informing
various steps	selecting
multiple users	Billing
critical variables	maintains
data integrity	allow
certain functions	deducting
malicious access	given
certain cryptographic techniques	
utilize	
bus	
hotel name	

departure time	
bus ticket reservation module	
ticket receipt	
customers	
generation	
account	
billing	
login id	
train	
ticket cancellation	
hotel bookings	
system refunds	

2.3 Accepted Noun & Verbs list

Entity Set with Attributes

- **Admin**(admin_id, admin_name, ad_login_id, password)
- **Customer**(c_id, customername, customername, c_password, gender , contact_no, address, state, country, pincode)
- **Booking**(booking_id, c_id,contact_no, booking_type, booking_date, package_id)
- **Bus**(bus_id, bus_type, ticket_id, arrival time, departure time, bording_point)
- **Train**(train_id, train_type, ticket_id, arrival time, departure time, bording_point, destination_location)
- **Flight**(flight_id, flight_type, ticket_id, arrival time, departure time, boarding_point, destination_location)
- **Cancellation**(cancellation_id, booking_id, cancel_status, payment_id)
- **Payment**(payment_id, booking_id, payment_method, amount)
- **Emergency**(emergecy_id, type, contact)
- **Tour_Package**(package_id, package_name, package_type, no_of_days, package_amount, package_description)
- **Tourist_Places**(place_id, place_name, place_type, place_description, place_images, safety_decription, weather)
- **Hotel**(hotel_id, hotel_address, hotel_name, hotel_type, hotel_image, room_type, checking_date, checkout_date)

Candidate relationship set

- **Customer** pay **Payment**
- **Customer** selecting **Tourist_Place**
- **Admin** update **Tour_Package**
- **Admin** monitor **Customer**
- **Cancellation** access by **Admin**
- **Tour_Package** and **Tour_Places** and **hotel** gives details of tour
- **Emergency** service available at all **Tourist_Places**
- **Payment** confirm the **Bus's** ticket **Booking**
- **Payment** confirm the **Flight's** ticket **Booking**
- **Payment** confirm the **Train's** ticket **Booking**

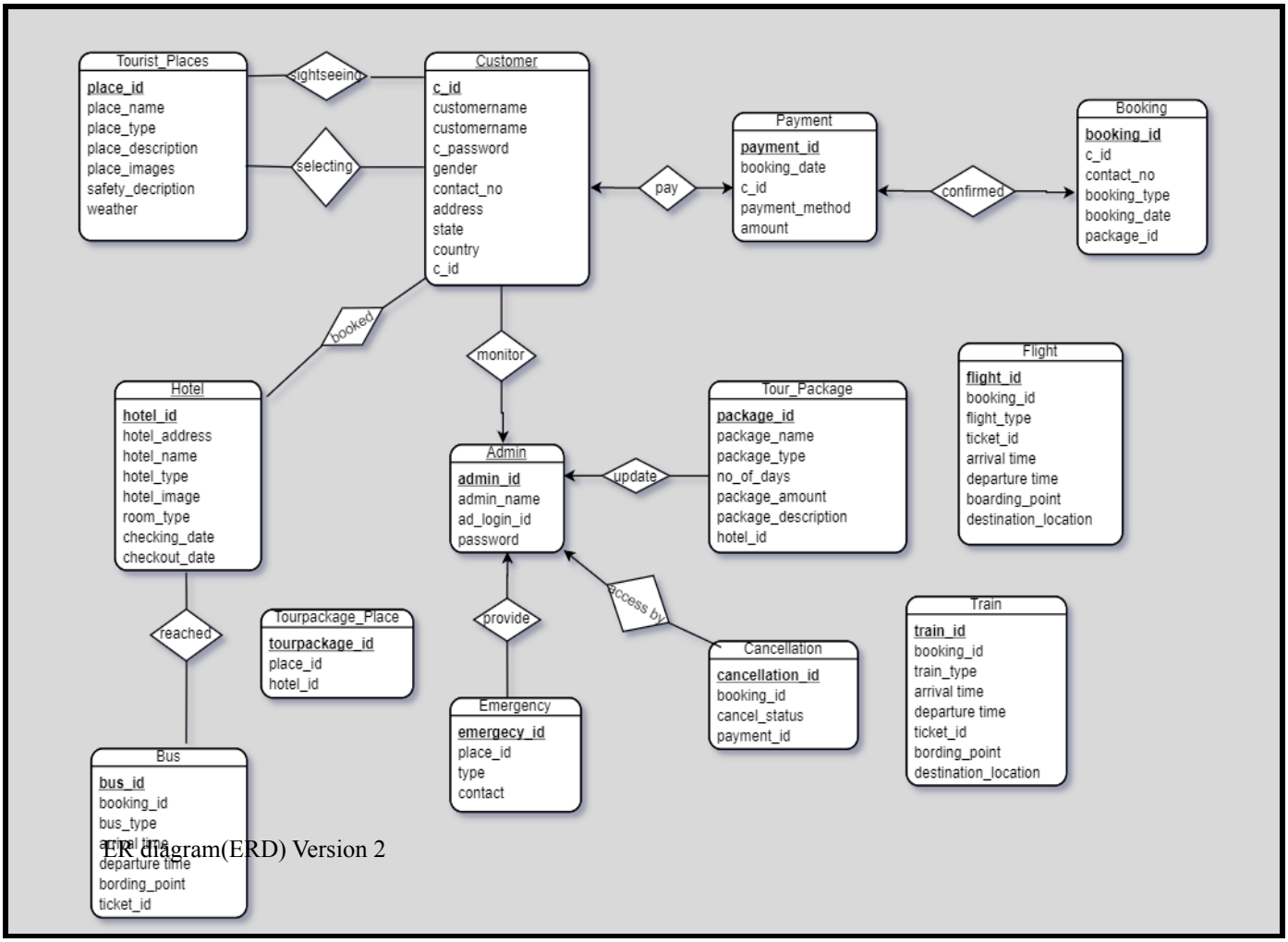
2.4 Rejected Noun & Verbs list

Noun	Reject Reason
tourist	Vague
travel package	Attribute
present system	Irrelevant
old manual system	Vague
seat availability	Attribute
manual calculation	Irrelevant
tour packages	Duplicates
that’s	General
travel agents	Attributes
relevant information	Duplicates
google form	General
system interface	Vague
visitors	Duplicates
users	Duplicates
administrator’s username	Attributes
pick-up points	Attributes

hotel vacancy	Attributes
covid-19 situation	Duplicates
agency member	Attributes
various routes	Irrelevant
users requirements	Vague
data integrity	Vague
malicious access	Vague
certain cryptographic techniques	Irrelevant
customers	General
login id	Attributes
cancellation charges	Vague
hotel name	Attributes
departure time	Attributes
check-in date	Irrelevant

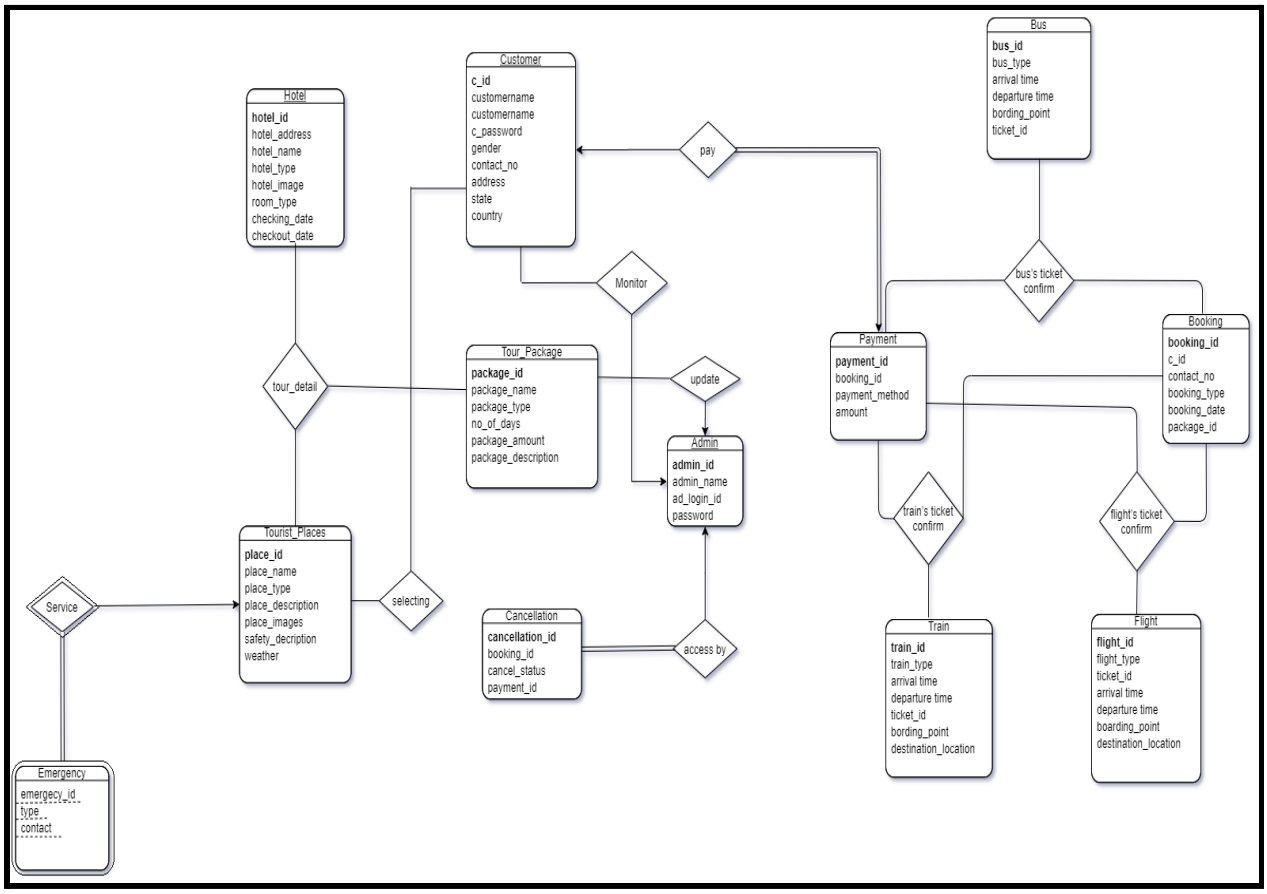
Section3 : ER-Diagrams all versions

3.1 ER Diagram(ERD) Version 1

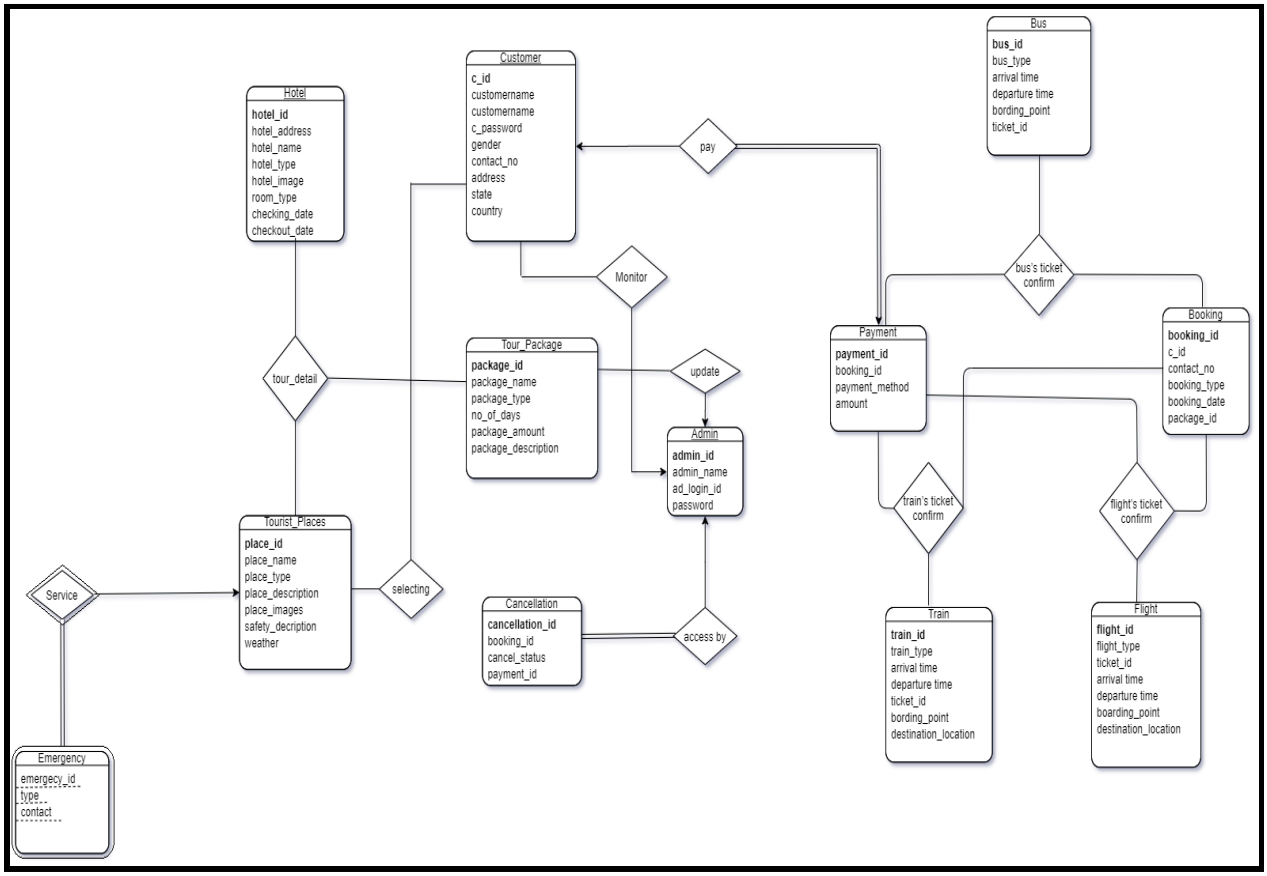


ER diagram(ERD) Version 2

3.2 ER Diagram(ERD) Version 2



3.3 Final ER diagram

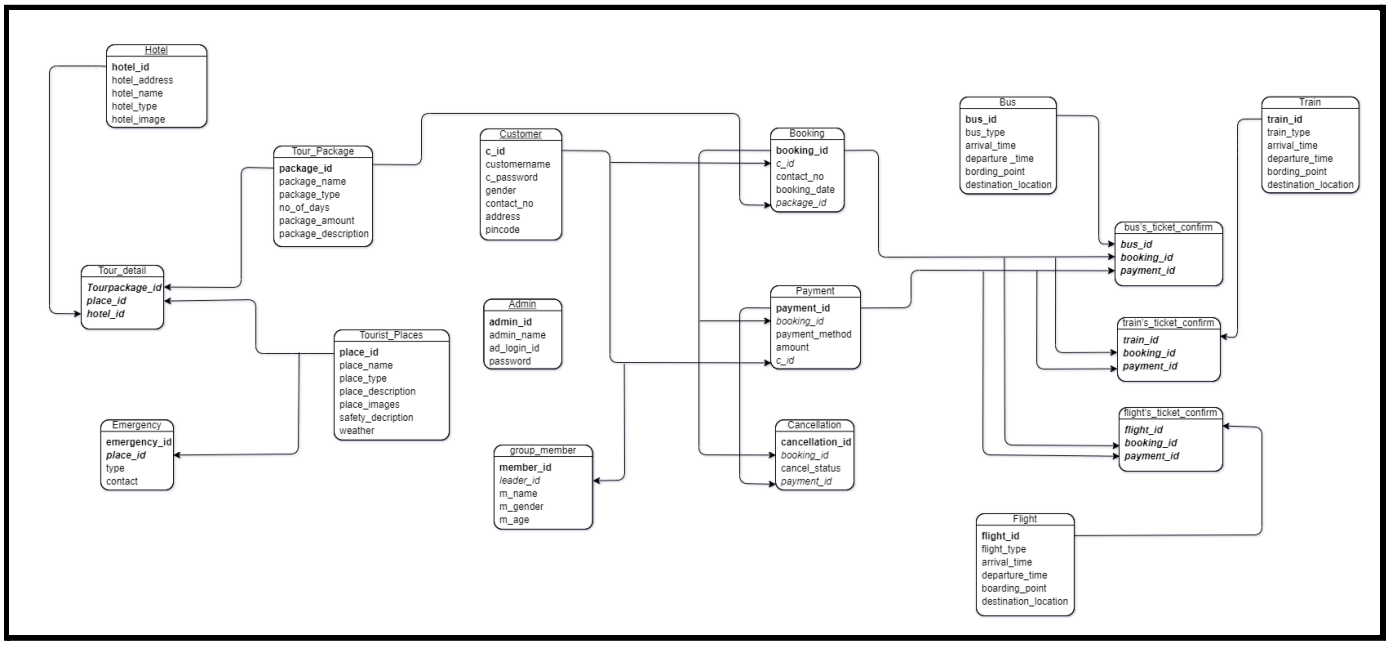


Section4 : Conversion of Final ER-Diagram to Relational Model

4.1 Mapping E-R Model to Relational Model

- Admin(**admin_id**, admin_name, ad_login_id, password)
- Customer(**c_id**, customername, c_password, gender , contact_no, address, pincode)
- group_member(**member_id**,leader_id, m_name, m_age, m_gender)
- Booking(**booking_id**, c_id, contact_no, booking_date, package_id)
- Bus(**bus_id**, bus_type, arrival_time, departure_time, bording_point,destination_location)
- Train(**train_id**, train_type, arrival_time, departure_time, bording_point, destination_location)
- Flight(**flight_id**, flight_type,arrival_time, departure_time, boarding_point, destination_location)
- Cancellation(**cancellation_id**, booking_id, cancel_status, payment_id)
- Payment(**payment_id**, booking_id, c_id, payment_method, amount)
- Emergency(**emergecy_id**, type, place_id, contact)
- Tour_Package(**package_id**,package_name, package_type, no_of_days, package_amount, package_description)
- Tour_detail(**tourpackage_id**, **place_id, hotel_id**)
- Tourist_Places(**place_id**, place_name, place_type, place_description, place_images, safety_decription, weather)
- Hotel(**hotel_id**,hotel_name, hotel_address, hotel_type, hotel_image)
- bus_ticket_confirm(**bus_id, payment_id, booking_id**)
- flight_ticket_confirm(**flight_id, payment_id, booking_id**)
- train_ticket_confirm(**train_id, payment_id, booking_id**)

4.2 Relational Schema



Section5 : Normalization and Schema Refinement

5.1 Original Database

- Admin(**admin_id**, admin_name, ad_login_id, password)
- Customer(**c_id**, customername, c_password, gender , contact_no, address, pincode)
- group_member(**member_id**,leader_id, m_name, m_age, m_gender)
- Booking(**booking_id**, c_id, contact_no, booking_date, package_id)
- Bus(**bus_id**, bus_type, arrival_time, departure_time, bording_point,destination_location)
- Train(**train_id**, train_type, arrival_time, departure_time, bording_point, destination_location)
- Flight(**flight_id**, flight_type,arrival_time, departure_time, boarding_point, destination_location)
- Cancellation(**cancellation_id**, booking_id, cancel_status, payment_id)
- Payment(**payment_id**, booking_id, c_id, payment_method, amount)
- Emergency(**emergecy_id**, type, place_id, contact)
- Tour_Package(**package_id**,package_name, package_type, no_of_days, package_amount, package_description)
- Tour_detail(**tourpackage_id**, **place_id, hotel_id**)
- Tourist_Places(**place_id**, place_name, place_type, place_description, place_images, safety_decription, weather)
- Hotel(**hotel_id**,hotel_name, hotel_address, hotel_type, hotel_image)
- bus_ticket_confirm(**bus_id, payment_id, booking_id**)
- flight_ticket_confirm(**flight_id, payment_id, booking_id**)
- train_ticket_confirm(**train_id, payment_id, booking_id**)

5.2 Functional Dependencies:

- Admin(**admin_id**, admin_name, ad_login_id, password)
  Primary Key : admin_id
  Functional dependencies :
  admin-id → admin_name
  admin-id → ad_login_id
  admin-id → password

- Customer(**c_id**, customername, c_password, gender , contact_no, address, pincode)
  Primary Key : c_id
  Functional dependencies :
  c_id → customername
  c_id → c_password
  c_id → gender
  c_id → contact_no
  c_id → address
  c_id → pincode

- group_member(**member_id** , leader_id , m_name, m_age, m_gender)
  Primary Key : member_id
  Foreign Key : leader_id
  Functional dependencies :
  member_id → m_name
  member_id → m_age
  member_id → m_gender

- Booking(**booking_id**, c_id, contact_no, booking_date, package_id)
  Primary Key : booking_id
  Foreign Key : c_id,package_id
  Functional dependencies :
  booking_id → contact_no
  booking_id → booking_date

- Bus(**bus_id**, bus_type, arrival_time, departure_time, bording_point,destination_location)
  Primary Key : bus_id
  Functional dependencies :
  bus_id → bus_type
  bus_id → arrival_time
  bus_id → departure_time
  bus_id → bording_point
  bus_id → destination_location

- Train(**train_id**, train_type, arrival_time, departure_time, bording_point, destination_location)
  Primary Key : train_id
  Functional dependencies :
  train_id → train_type
  train_id → arrival_time
  train_id → departure_time
  train_id → bording_point
  train_id → destination_location

- Flight(**flight_id**, flight_type, arrival_time, departure_time, boarding_point, destination_location)
  Primary Key: flight_id
  Functional dependencies :
  flight_id → flight_type
  flight_id → arrival_time
  flight_id → departure_time
  flight_id → bording_point
  flight_id → destination_location

- Cancellation(**cancellation_id**, booking_id, cancel_status, payment_id)
  Primary Key : cancellation_id
  Foreign Key : booking_id,payment_id
  Functional dependencies :
  cancellation_id → cancel_status
- Payment(**payment_id**, booking_id, c_id, payment_method, amount)
  Primary Key : payment_id
  Foreign Key : booking_id,c_id
  Functional dependencies :
  payment_id → payment_method
  payment_id → amount

- Emergency(**emergecy_id**, type, place_id, contact)
  Primary Key : emergency_id, place_id
  Foreign Key : place_id
  Functional dependencies :
  emergency_id → type
  emergency_id → contact

- Tour_Package(**package_id**,package_name, package_type, no_of_days, package_amount, package_description)
  Primary Key : package_id
  Functional dependencies :
  package_id → package_name
  package_id → package_type
  package_id → no_of_days
  package_id → package_amount
  package_id → package_description

- Tour_detail(**tourpackage_id**, **place_id, hotel_id**)
  Primary Key : tourpackage_id, place_id,hotel_id
  Foreign Key : tourpackage_id,place_id,hotel_id

- Tourist_Places(**place_id**, place_name, place_type, place_description, place_images, safety_description, weather)
  Primary Key : place_id
  Functional dependencies :
  place_id → place_name
  place_id → place_type
  place_id → place_description
  place_id → place_images
  place_id → safety_description
  place_id → weather

- Hotel(**hotel_id**,hotel_name, hotel_address, hotel_type, hotel_image)
  Primary Key : hotel_id
  Functional dependencies :
  hotel_id → hotel_address
  hotel_id → hotel_name
  hotel_id → hotel_type
  hotel_id → hotel_image

- bus_ticket_confirm(**bus_id, payment_id, booking_id**)
  Primary Key : bus_id,payment_id,booking_id
  Foreign Key : bus_id,payment_id,booking_id

- flight_ticket_confirm(**flight_id, payment_id, booking_id**)
  Primary Key : flight_id ,payment_id ,booking_id
  Foreign Key : flight_id ,payment_id ,booking_id

- train_ticket_confirm(**train_id, payment_id, booking_id**)
  Primary Key : train_id ,payment_id ,booking_id
  Foreign Key : train_id ,payment_id ,booking_id

5.3 Redundancies:

Redundancies occur when we have multiple copies of the same data in our database. Basically, this problem arises when data is not normalised. In our database, we have stored all our data in different types of tables, so we don't have any redundancies in our database.

5.4 Update, insert and delete anomalies:

Insertion deletion and update anomalies mean If the user performing the update does not realize the data is stored redundantly the update will not be done properly. A deletion anomaly is the unintended loss of data due to deletion of other data. ... An insertion anomaly is the inability to add data to the database due to the absence of other data.

But in our database, we have stored all necessary information in different tables and we don't have any non-prime attributes functional dependencies, so we don't have any type of anomalies in our database.

5.5 Normalization to 1NF:

In 1NF form, there should be no repeated columns inside a row or multivalued columns in any database relation, and all attributes should be reliant on the primary key.

In our database design, all the listed attributes in the relations are atomic. We're also just taking one entry for each of the attributes. As a result, the preceding 1NF normalized form is used in database design.

5.6 Normalization to 2NF:

Each and every relationship in the 2NF normalized form should be in the 1NF form, and there should be no partial dependencies in the relations. No nonprime attributes are dependent on any proper subset of the relation's candidate key, according to partial dependency. Any database design relations that have only one Primary key (not composite) are already in the 2NF form.

In our database design there are no partial dependencies, as can be seen from the functional dependencies listed above. As a result, this design has already been normalized to 2NF.

5.7 Normalization to 3NF:

To be in the 3NF, all of the relations must first be in the 2NF form, and there should be no transitive functional dependencies in the relations for non-prime attributes in the 3NF. That mean's, functional dependencies such as non-prime $\rightarrow$ non-prime does not exist.

In our database design, there are no transitive functional dependencies because all dependencies are defined by the primary key (candidate key) only. As a result, this database design is already 3NF normalized.

5.8 Normalization to BCNF :

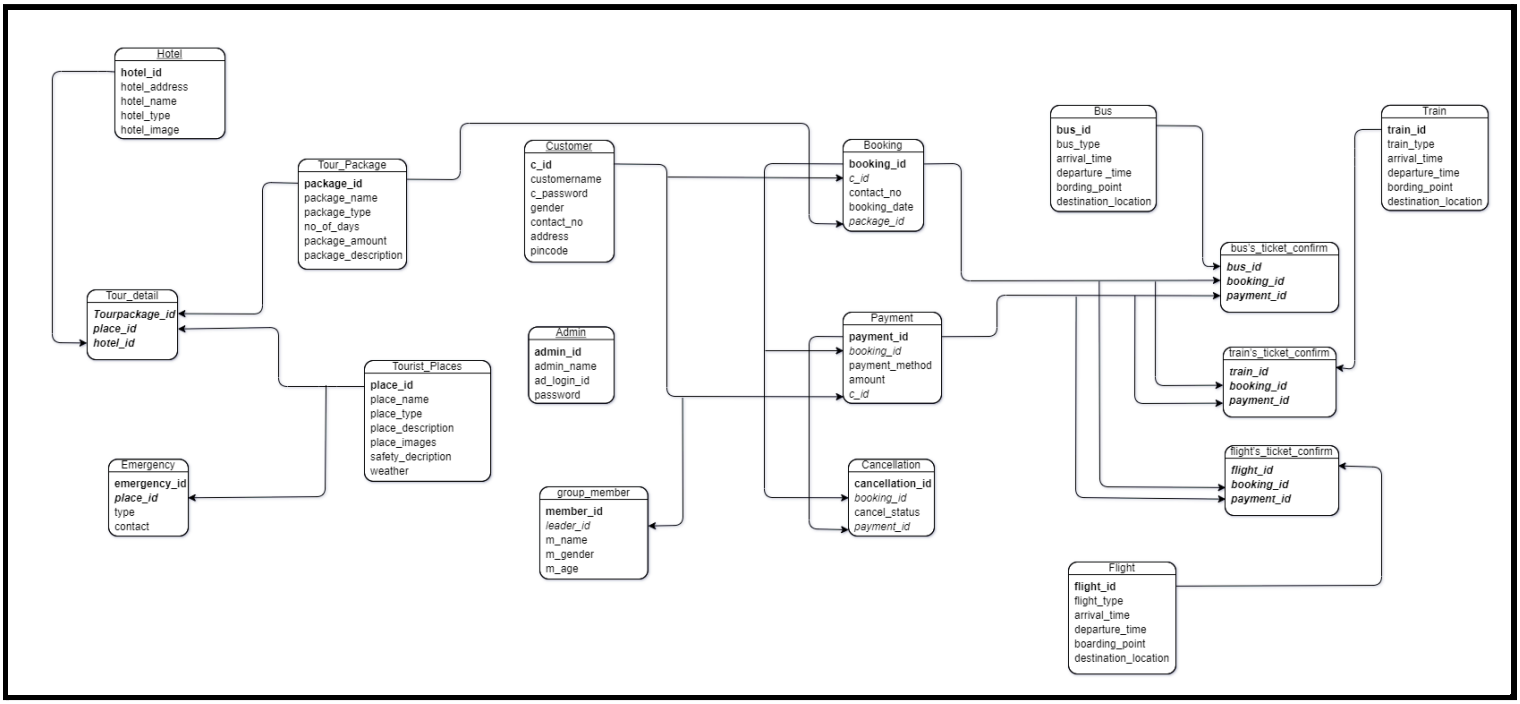
For the relation to be in BCNF form, if all the functional dependencies have a super key or candidate key on their left-hand side, no non-prime attribute can be on the left-hand side.

In our database, the only candidate key, which is also a primary key and it can define all the other attributes present in the relation. Hence all the functional dependencies of type A->B are such that A is a name that is a super key hence it is already normalized to BCNF.

NOTE :

The Bus's_ticket_confirm, train's_ticket_confirm and flight's_ticket_confirm tables have only primary keys so there are no functional dependencies as such in any of the attributes so this relation is already normalized. So, the table is in 1NF,2NF,3NF, and BCNF.

5.9 Final relations with the schema



- Admin(**admin_id**, admin_name, ad_login_id, password)
- Customer(**c_id**, customername, c_password, gender , contact_no, address, pincode)
- group_member(**member_id**,leader_id, m_name, m_age, m_gender)
- Booking(**booking_id**, c_id, contact_no, booking_date, package_id)
- Bus(**bus_id**,bus_type,arrival_time,departure_time,bording_point, destination_location)
- Train(**train_id**, train_type, arrival_time, departure_time, bording_point, destination_location)
- Flight(**flight_id**, flight_type,arrival_time, departure_time, boarding_point, destination_location)
- Cancellation(**cancellation_id**, booking_id, cancel_status, payment_id)
- Payment(**payment_id**, booking_id, c_id, payment_method, amount)
- Emergency(**emergecy_id**, type, place_id, contact)
- Tour_Package(**package_id**,package_name, package_type, no_of_days, package_amount, package_description)
- Tour_detail(**tourpackage_id**, **place_id, hotel_id**)
- Tourist_Places(**place_id**, place_name, place_type, place_description, place_images, safety_decription, weather)
- Hotel(**hotel_id**,hotel_name, hotel_address, hotel_type, hotel_image)
- bus_ticket_confirm(**bus_id, payment_id, booking_id**)
- flight_ticket_confirm(**flight_id, payment_id, booking_id**)
- train_ticket_confirm(**train_id, payment_id, booking_id**)

Section6 : SQL: Final DDL Scripts, Insert statements, 40 SQL Queries with Snapshots of the output of each query

6.1 Final DDL Scripts

```
CREATE TABLE Admin
(
        admin_id integer,
        admin_name varchar(30) not null,
        ad_login_id varchar(30),
        password varchar(30),
        primary key (admin_id)
);
CREATE TABLE Customer
(
        c_id integer,
        customername varchar(30) not null,
        c_password varchar(30),
        gender varchar(6),
        contact_no varchar(30),
        address varchar(300),
        pincode integer,
        primary key (c_id),
        check(gender in ('Female','Male'))
);

CREATE TABLE group_member
(
        member_id integer,
        leader_id integer not null,
        m_name varchar(30) not null,
        m_age integer,
        m_gender varchar(6),
        primary key (member_id),
        check(m_gender in ('Female','Male')),
        foreign key (leader_id) references Customer(c_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);
```

```
CREATE TABLE Booking
(
        booking_id varchar(20),
        c_id integer,
        contact_no varchar(30),
        booking_date varchar(50),
        package_id integer,
        primary key (booking_id),
        foreign key (c_id) references Customer
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key (package_id) references Tour_Package
        ON DELETE SET NULL
        ON UPDATE CASCADE
);

CREATE TABLE Bus
(
        bus_id varchar(20) ,
        bus_type varchar(30),
        arrival_time varchar(50),
        departure_time varchar(50),
        bording_point varchar(30),
        destination_location varchar(30),
        primary key (bus_id)
);

CREATE TABLE Train
(
        train_id varchar(20),
        train_type varchar(30),
        arrival_time varchar(50),
        departure_time varchar(50),
        bording_point varchar(30),
        destination_location varchar(30),
        primary key (train_id)
);
```

```
CREATE TABLE Flight
(
        flight_id varchar(20),
        flight_type varchar(30),
        arrival_time varchar(50),
        departure_time varchar(50),
        boarding_point varchar(30),
        destination_location varchar(30),
        primary key (flight_id)
);

CREATE TABLE Cancellation
(
        cancellation_id varchar(20),
        booking_id varchar(20) not null,
        cancel_status varchar(20),
        payment_id varchar(20) not null,
        primary key (cancellation_id),
        foreign key (booking_id) references Booking
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key (payment_id) references Payment
        ON DELETE CASCADE
        ON UPDATE CASCADE
);
CREATE TABLE Payment
(
        payment_id varchar(20),
        booking_id varchar(20) not null,
        c_id integer not null,
        payment_method varchar(20),
        amount  integer check(amount > 0),
        primary key (payment_id),
        foreign key (booking_id) references Booking
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key (c_id) references Customer
```

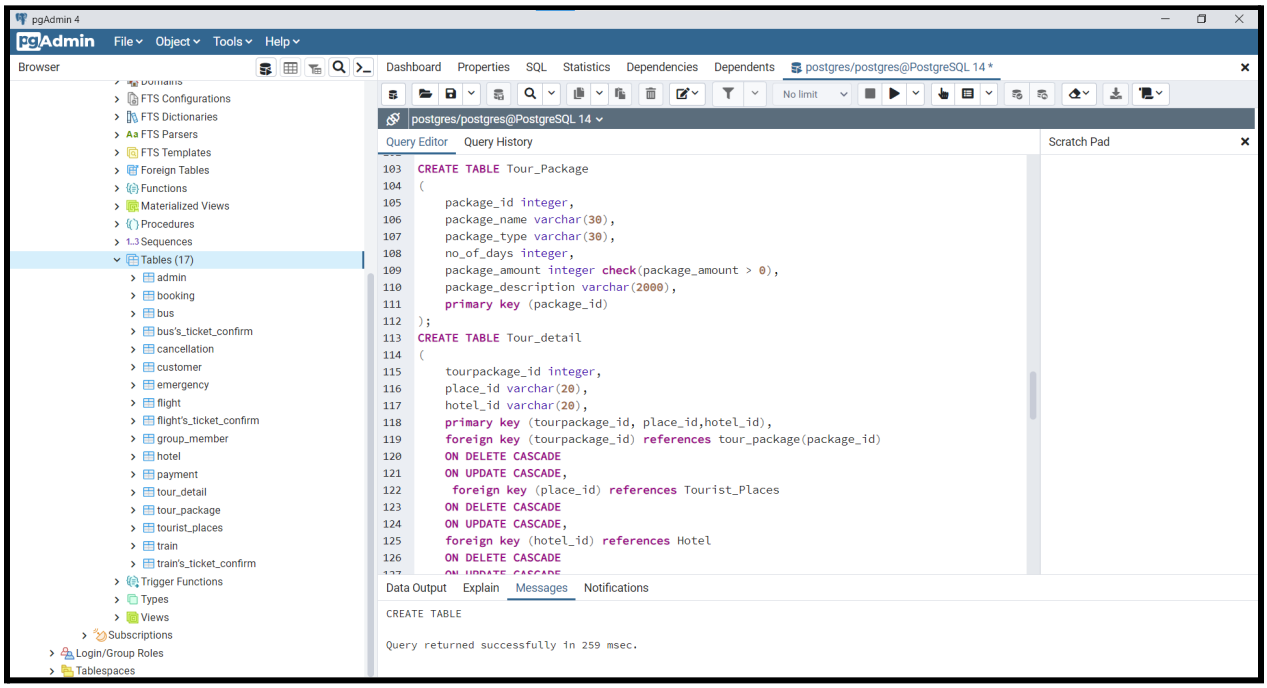
```
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE Emergency
(
        emergency_id varchar(20),
        type varchar(50) ,
        place_id varchar(20),
        contact varchar(30),
        primary key (emergency_id, place_id),
        foreign key (place_id) references Tourist_places
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE Tour_Package
(
        package_id integer,
        package_name varchar(30),
        package_type varchar(30),
        no_of_days integer,
        package_amount integer check(package_amount > 0),
        package_description varchar(2000),
        primary key (package_id)
);

CREATE TABLE Tour_detail
(
        tourpackage_id integer,
        place_id varchar(20),
        hotel_id varchar(20),
        primary key (tourpackage_id, place_id,hotel_id),
        foreign key (tourpackage_id) references tour_package(package_id)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key (place_id) references Tourist_Places
```

```
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key (hotel_id) references Hotel
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE Tourist_Places
(
        place_id varchar(20),
        place_name varchar(30) not null,
        place_type varchar(30),
        place_description varchar(3000),
        place_images varchar(200),
        safety_decription varchar(3000),
        weather varchar(30),
        primary key (place_id)
);

CREATE TABLE Hotel
(
        hotel_id varchar(20),
        hotel_name varchar(30),
        hotel_address varchar(100),
        hotel_type varchar(30),
        hotel_image varchar(200),
        primary key(hotel_id)
);

CREATE TABLE bus_ticket_confirm
(
        bus_id varchar(20),
        payment_id varchar(20),
        booking_id varchar(20),
        primary key(bus_id,payment_id,booking_id),
        foreign key (bus_id) references Bus
        ON DELETE CASCADE
```

```
        ON UPDATE CASCADE,
        foreign key (payment_id) references Payment
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key (booking_id) references Booking
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE train_ticket_confirm
(
        train_id varchar(20),
        payment_id varchar(20),
        booking_id varchar(20),
        primary key(train_id,payment_id,booking_id),
        foreign key (train_id) references Train
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key (payment_id) references Payment
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key (booking_id) references Booking
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE flight_ticket_confirm
(
        flight_id varchar(20),
        payment_id varchar(20),
        booking_id varchar(20),
        primary key(flight_id ,payment_id ,booking_id),
        foreign key(flight_id ) references Flight
        ON DELETE CASCADE
        ON UPDATE CASCADE,
        foreign key(payment_id) references Payment
        ON DELETE CASCADE
```

```
        ON UPDATE CASCADE,
        foreign key (booking_id) references Booking
        ON DELETE CASCADE
        ON UPDATE CASCADE
);
```

6.2 Creation of Tables

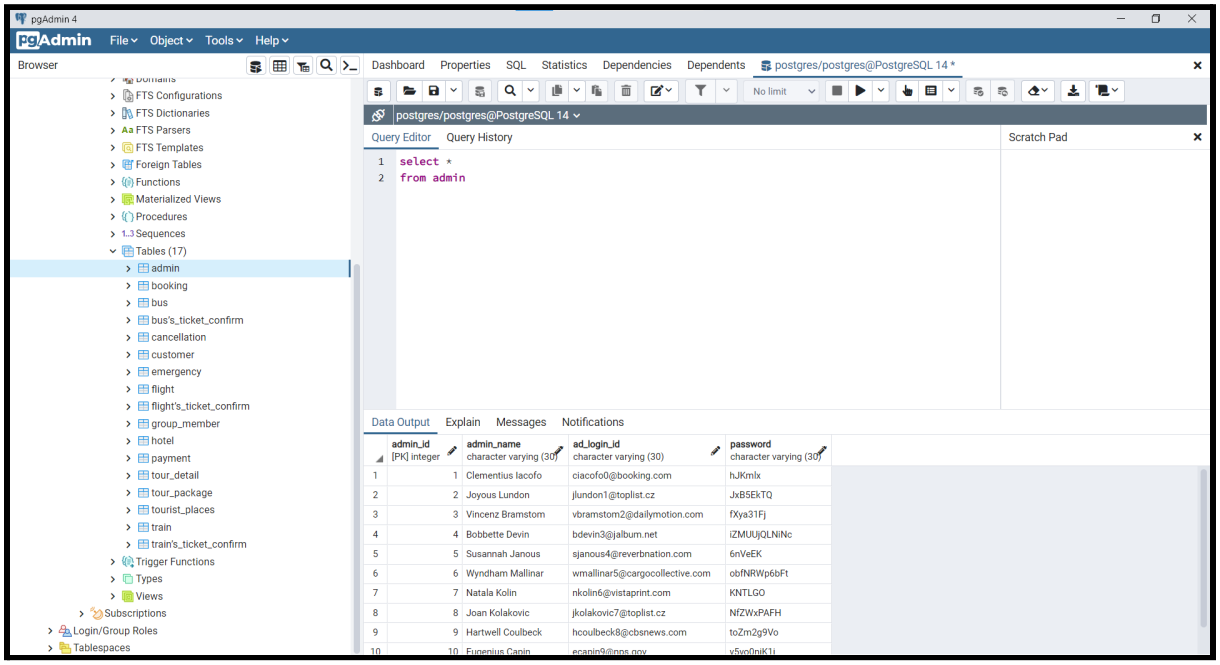


6.3 Insert statements

We have created all tables and inserted all tuples using the following SQL query

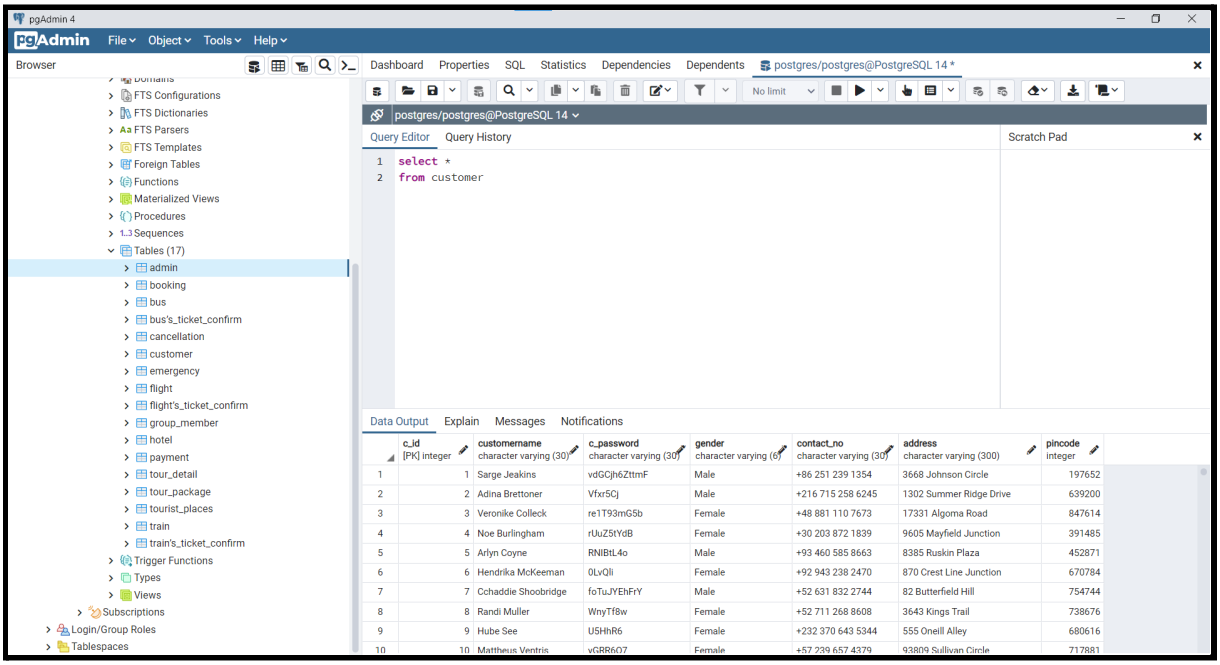
```
COPY t4.Admin
FROM 'E:\SEM5\DBMS\database\admin.csv' DELIMITER ',' CSV HEADER;
```

Admin



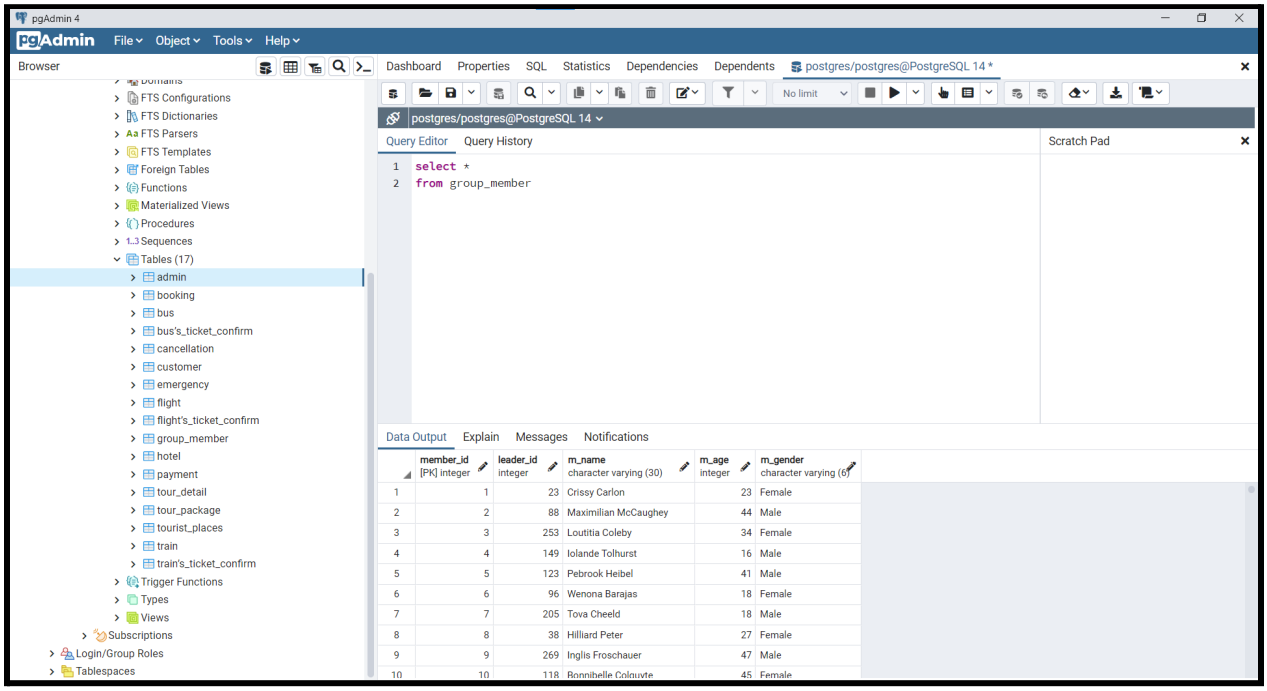
Number of tuples: 10

Customer



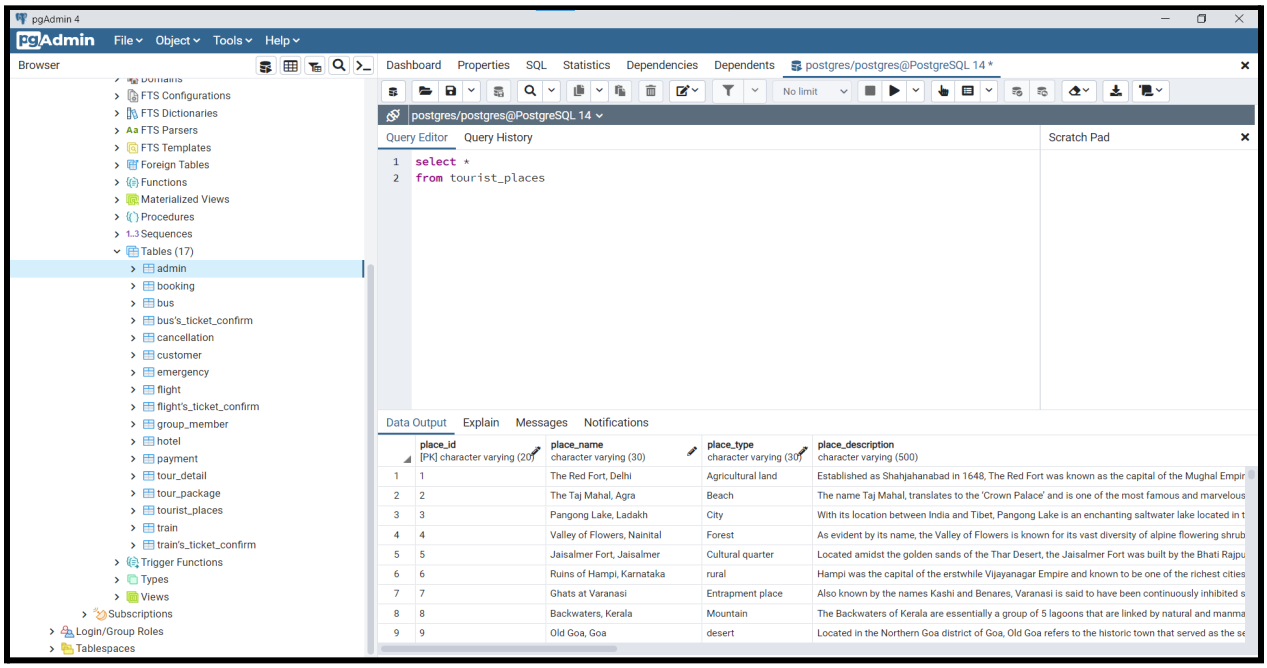
Number of tuples: 300

Group Member



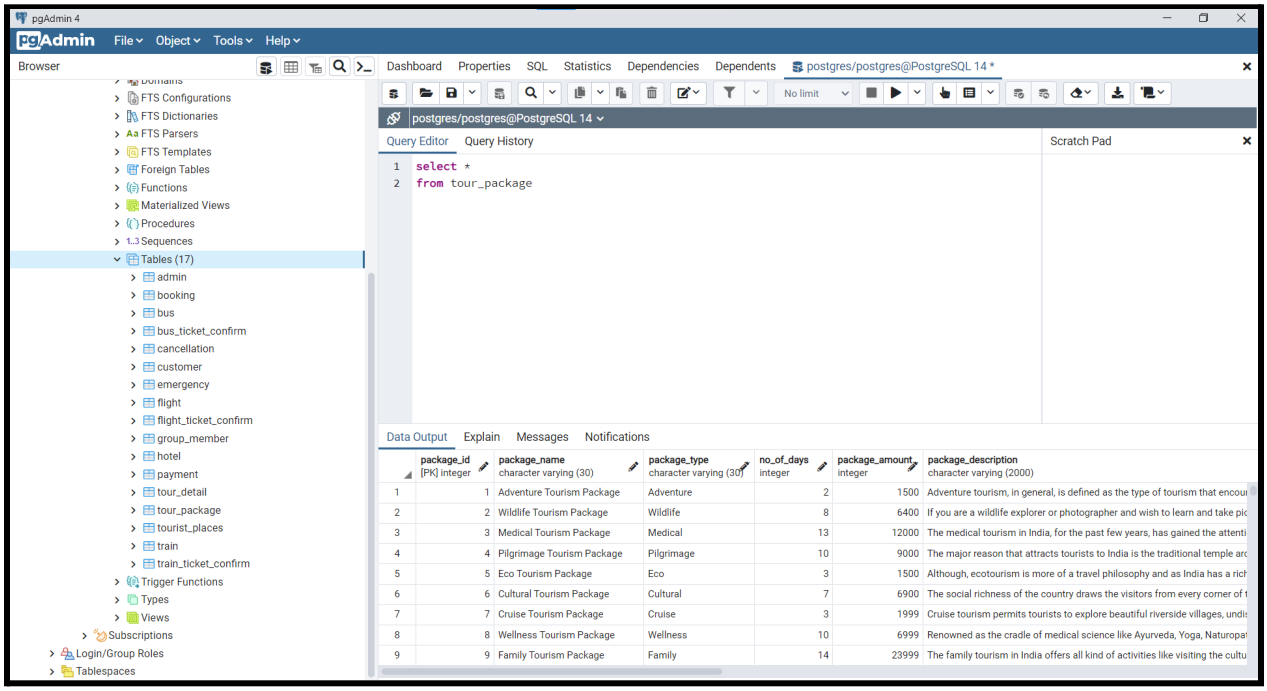
Number of tuples: 500

Tourist Place



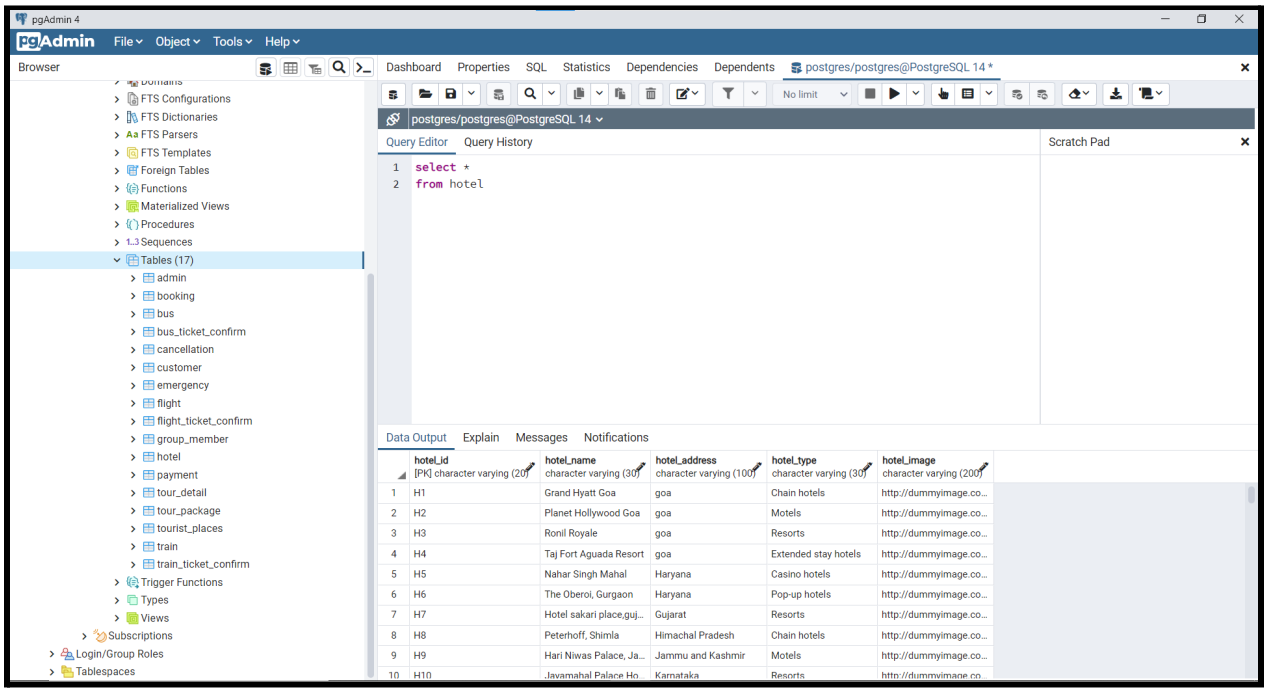
Number of tuples: 100

Tour Package



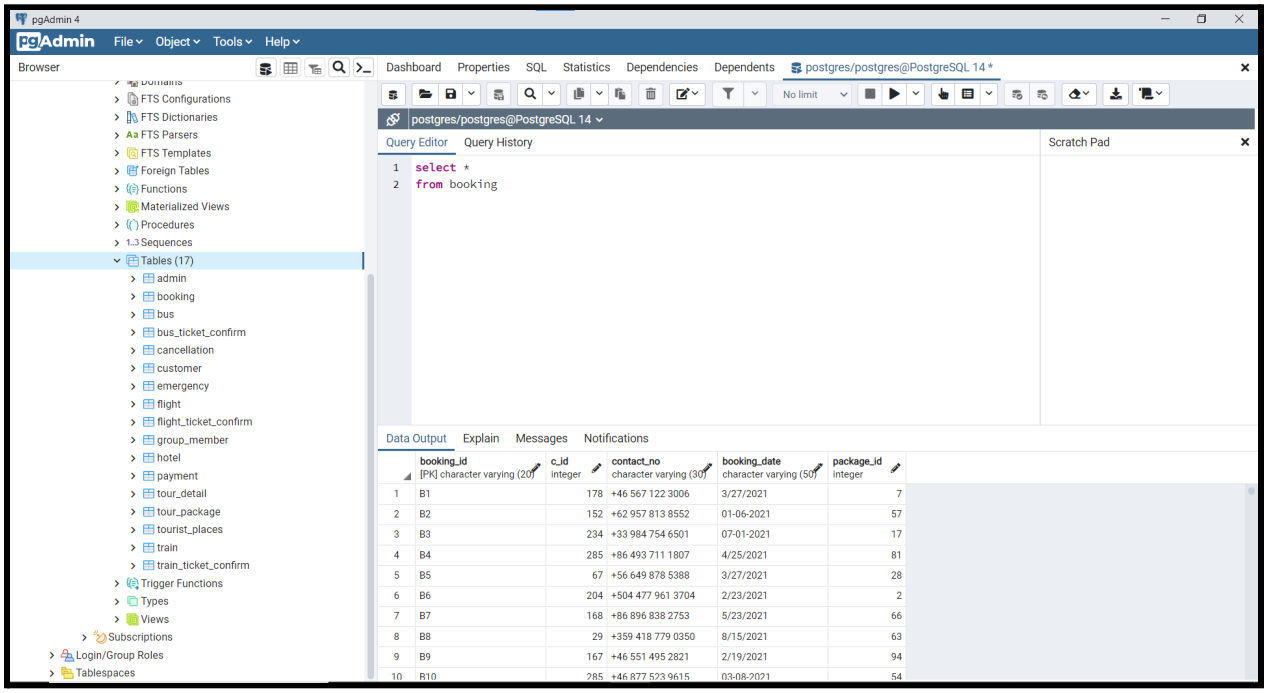
Number of tuples: 100

Hotel



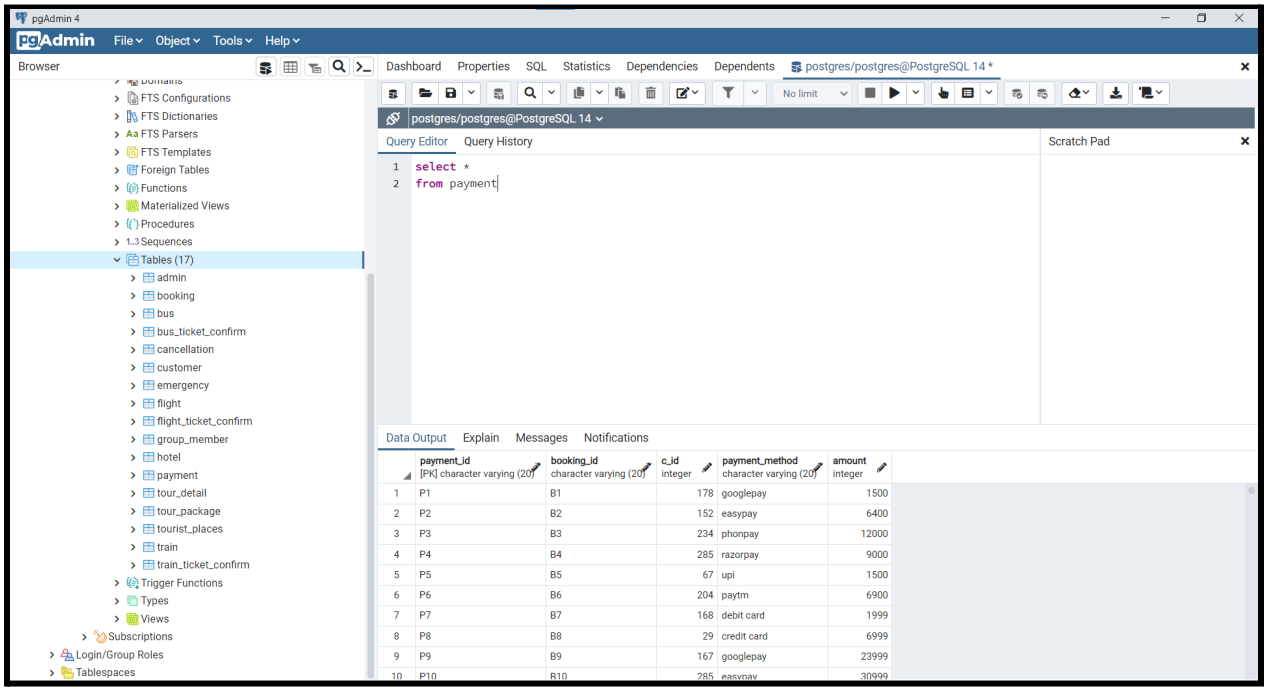
Number of tuples: 80

Booking



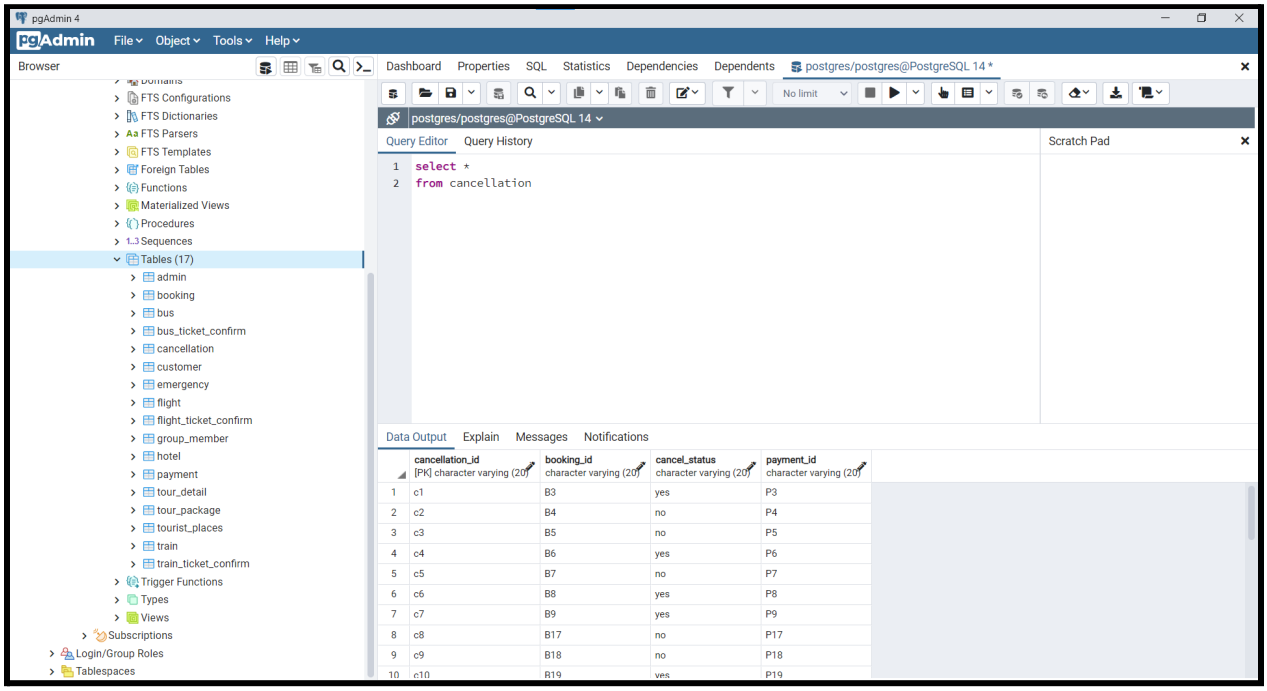
Number of tuples: 200

Payment



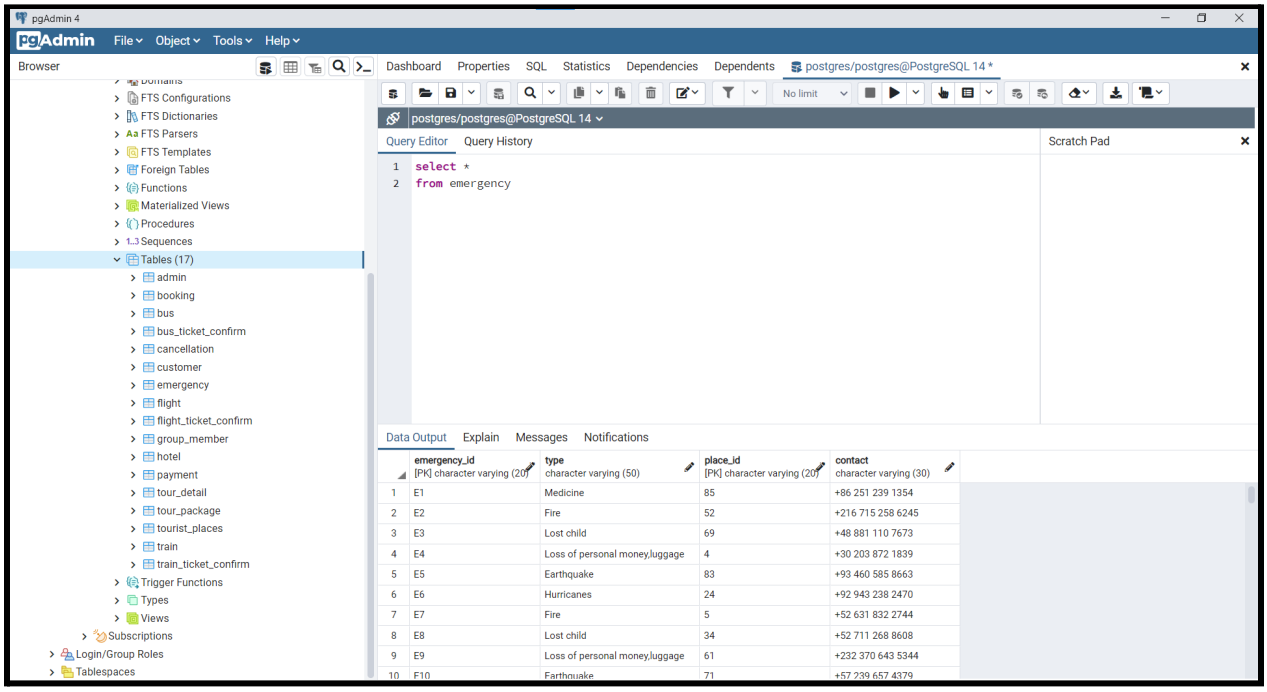
Number of tuples: 200

Cancellation



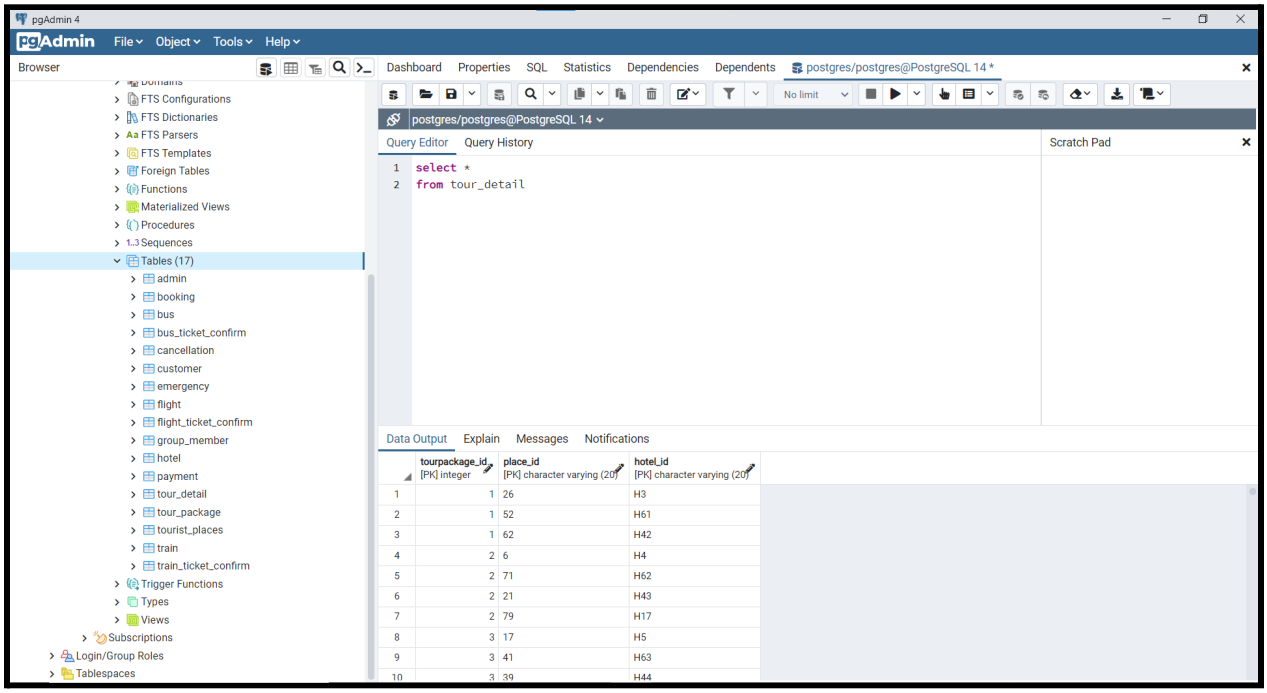
Number of tuples: 30

Emergency



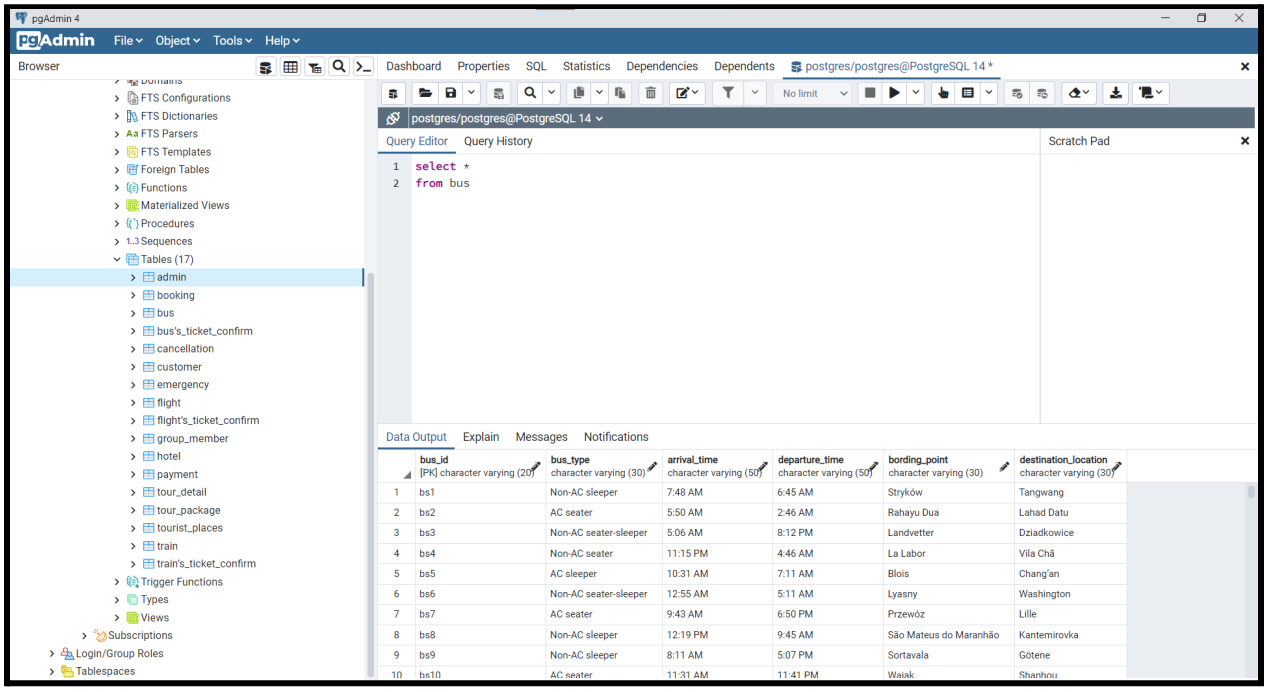
Number of tuples: 80

Tour_detail



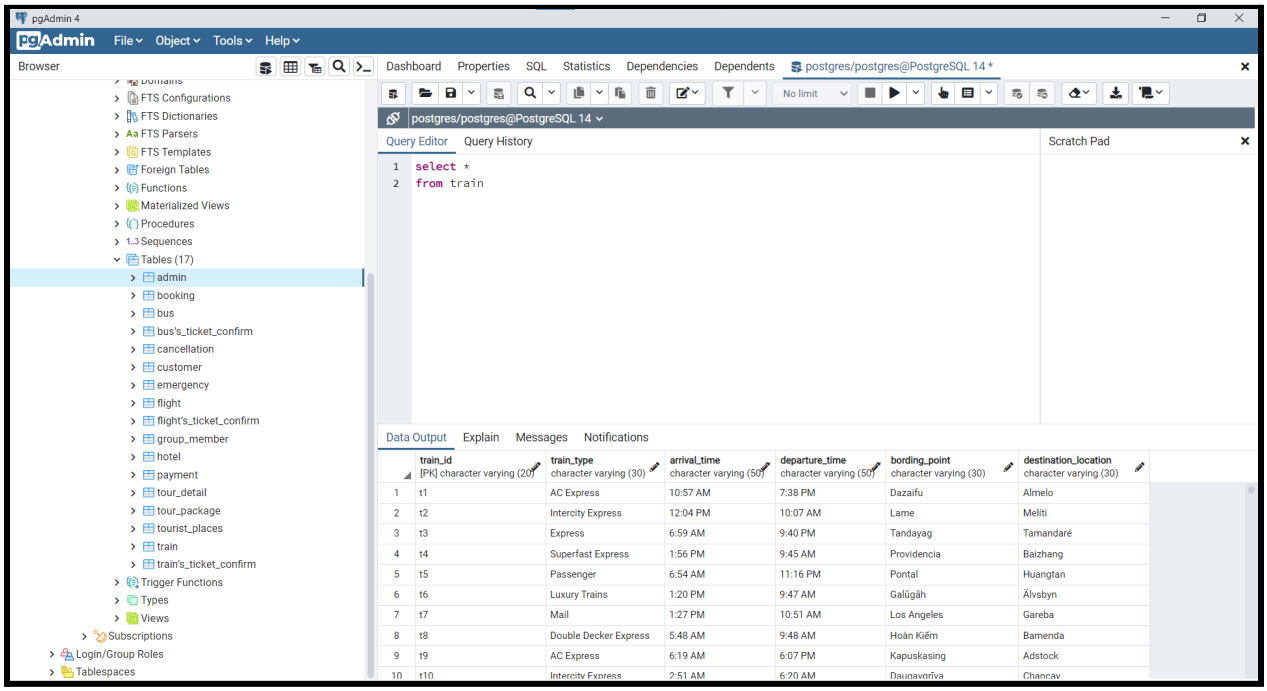
Number of tuples: 375

Bus



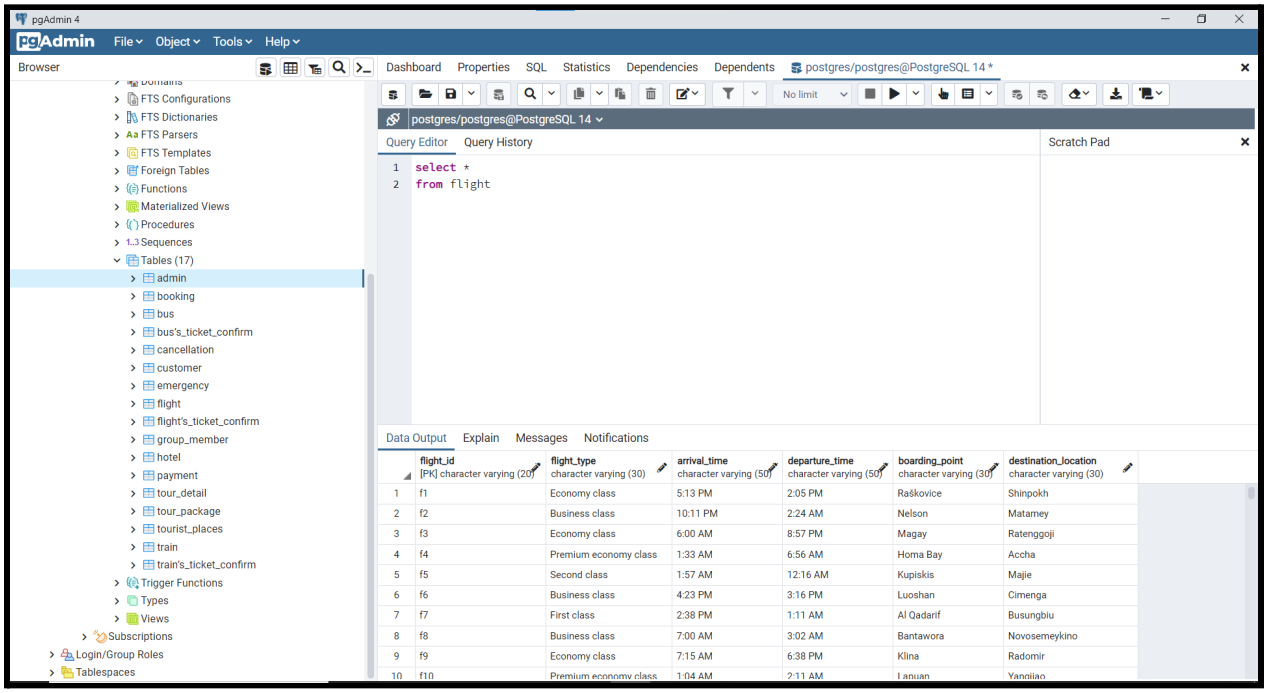
Number of tuples: 100

Train



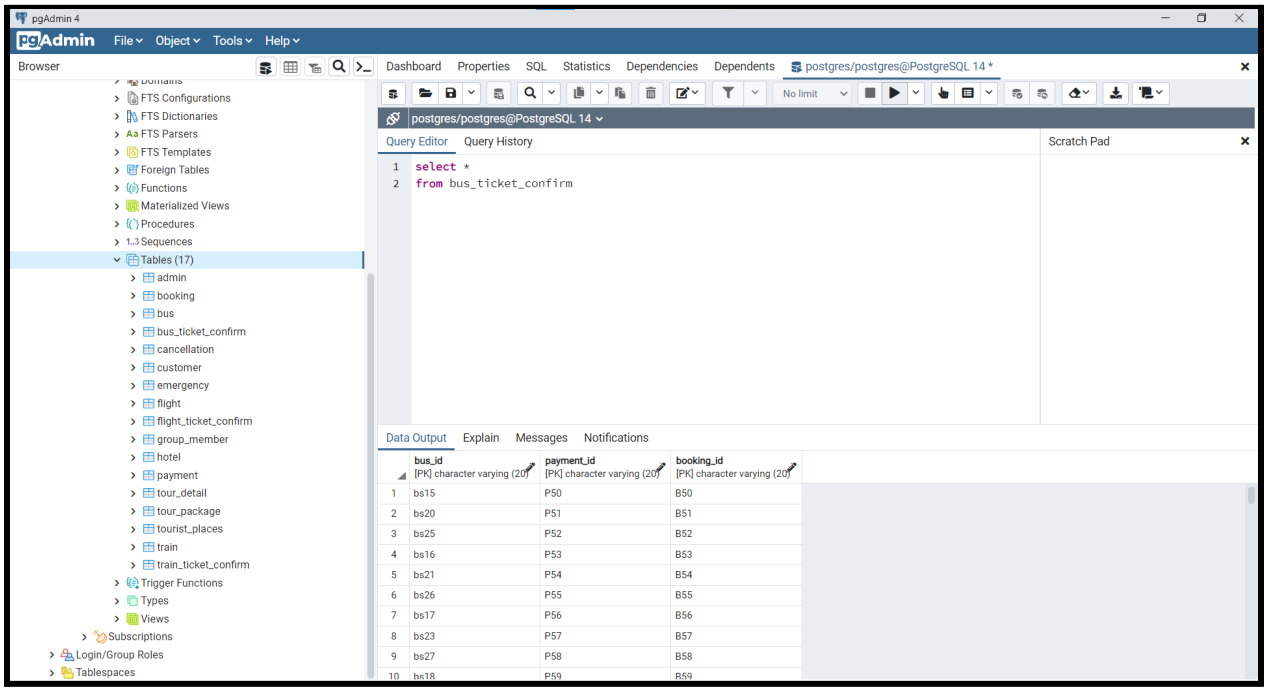
Number of tuples: 150

Flight



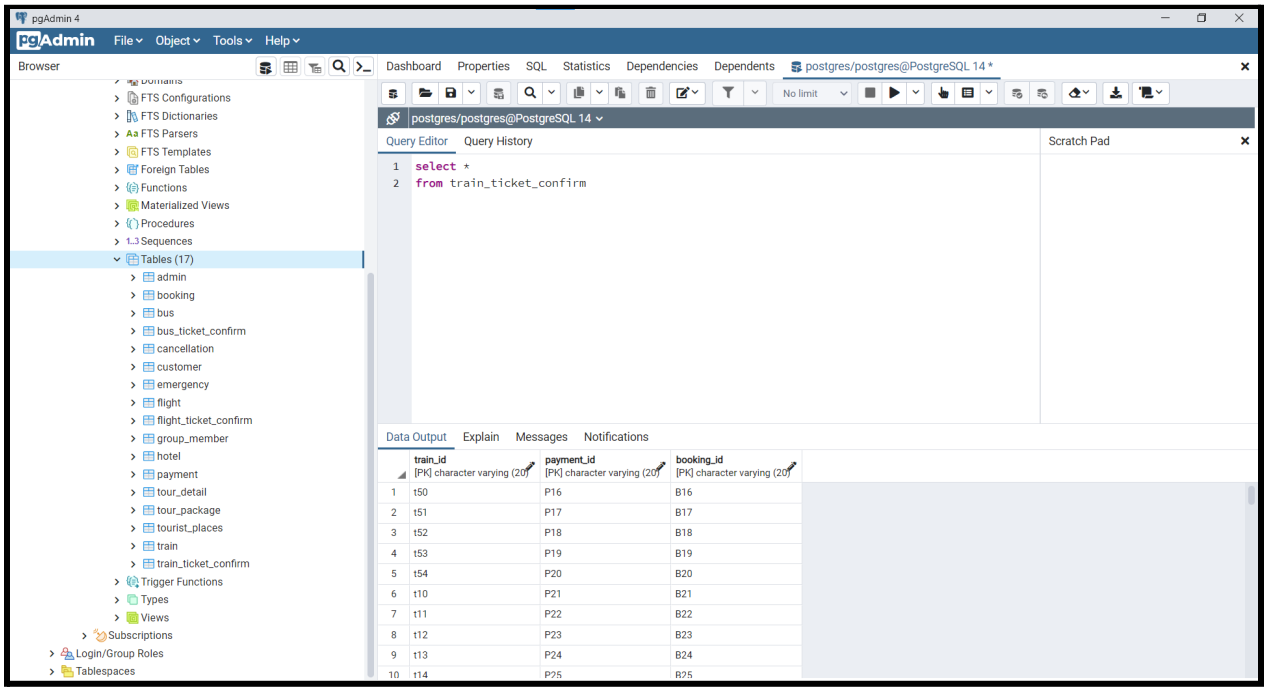
Number of tuples: 100

bus_ticket_confirm



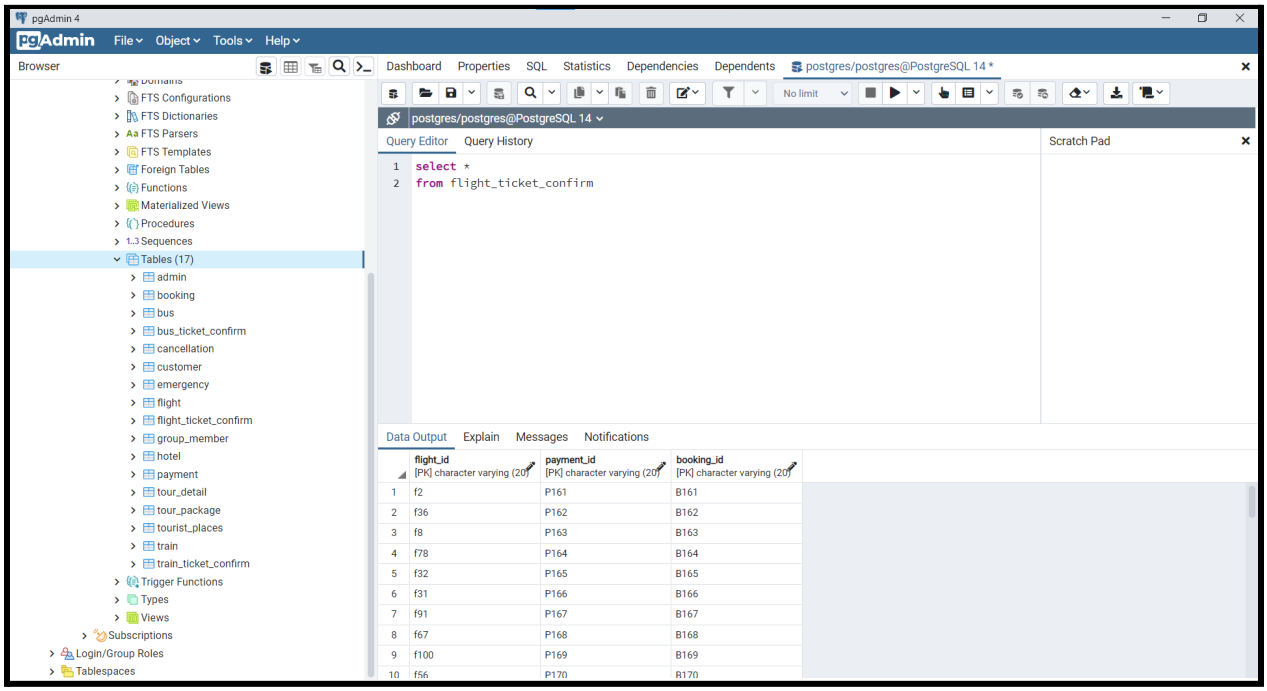
Number of tuples: 79

train_ticket_confirm



Number of tuples: 73

flight_ticket_confirm

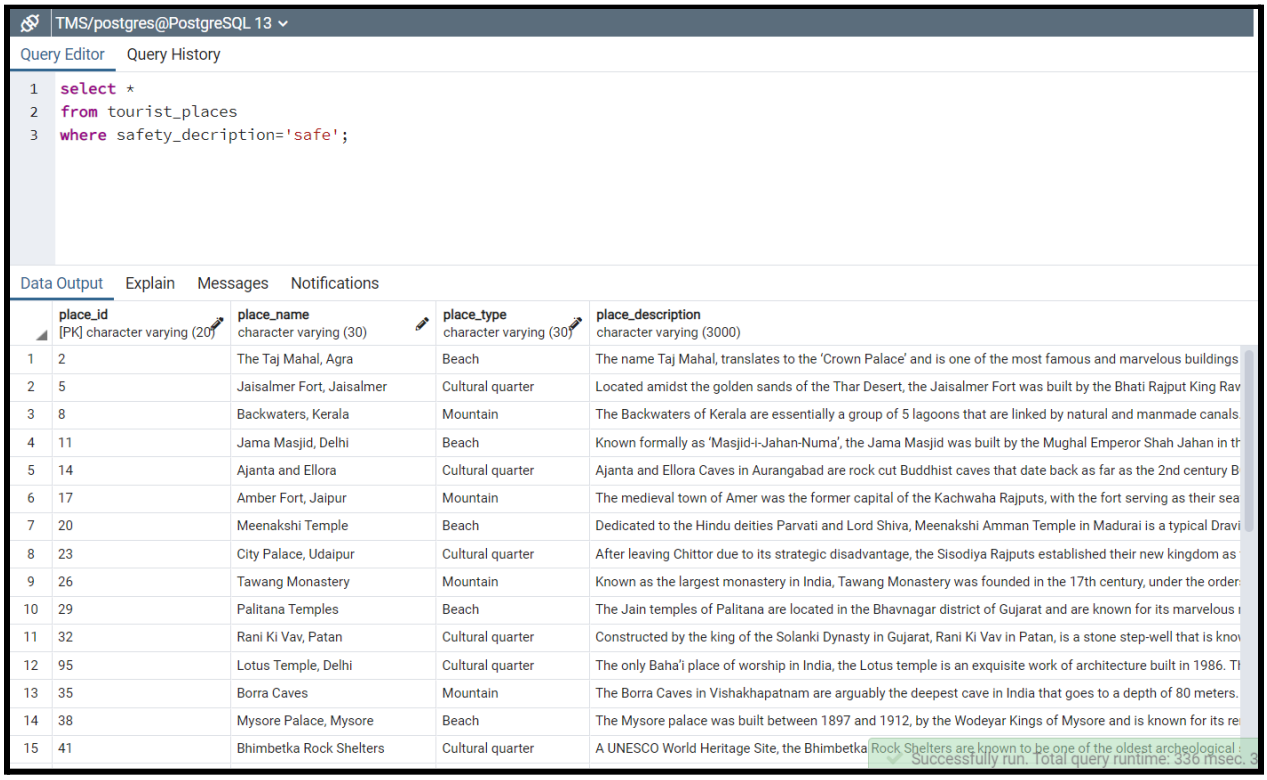


Number of tuples: 48

6.4 40 SQL Queries with Snapshots of output of each query

Q1. List all the tourist place's details that are safe to visit.

```
select *
from tourist_places
where safety_decription='safe';
```

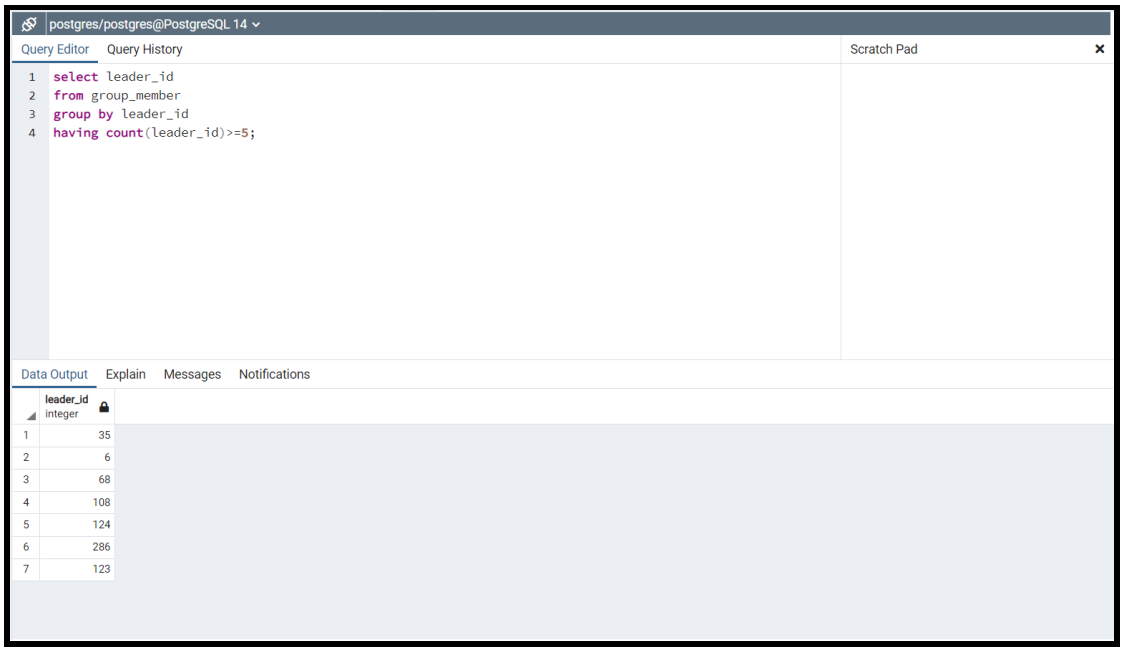


	place_id [PK] character varying (20)	place_name character varying (30)	place_type character varying (30)	place_description character varying (3000)
1	2	The Taj Mahal, Agra	Beach	The name Taj Mahal, translates to the 'Crown Palace' and is one of the most famous and marvelous buildings
2	5	Jaisalmer Fort, Jaisalmer	Cultural quarter	Located amidst the golden sands of the Thar Desert, the Jaisalmer Fort was built by the Bhati Rajput King Rav
3	8	Backwaters, Kerala	Mountain	The Backwaters of Kerala are essentially a group of 5 lagoons that are linked by natural and manmade canals
4	11	Jama Masjid, Delhi	Beach	Known formally as 'Masjid-i-Jahan-Numa', the Jama Masjid was built by the Mughal Emperor Shah Jahan in th
5	14	Ajanta and Ellora	Cultural quarter	Ajanta and Ellora Caves in Aurangabad are rock cut Buddhist caves that date back as far as the 2nd century B
6	17	Amber Fort, Jaipur	Mountain	The medieval town of Amer was the former capital of the Kachwaha Rajputs, with the fort serving as their sea
7	20	Meenakshi Temple	Beach	Dedicated to the Hindu deities Parvati and Lord Shiva, Meenakshi Amman Temple in Madurai is a typical Dravi
8	23	City Palace, Udaipur	Cultural quarter	After leaving Chittor due to its strategic disadvantage, the Sisodiya Rajputs established their new kingdom as
9	26	Tawang Monastery	Mountain	Known as the largest monastery in India, Tawang Monastery was founded in the 17th century, under the order
10	29	Palitana Temples	Beach	The Jain temples of Palitana are located in the Bhavnagar district of Gujarat and are known for its marvelous
11	32	Rani Ki Vav, Patan	Cultural quarter	Constructed by the king of the Solanki Dynasty in Gujarat, Rani Ki Vav in Patan, is a stone step-well that is kno
12	95	Lotus Temple, Delhi	Cultural quarter	The only Baha'i place of worship in India, the Lotus temple is an exquisite work of architecture built in 1986. T
13	35	Borra Caves	Mountain	The Borra Caves in Vishakhapatnam are arguably the deepest cave in India that goes to a depth of 80 meters.
14	38	Mysore Palace, Mysore	Beach	The Mysore palace was built between 1897 and 1912, by the Wodeyar Kings of Mysore and is known for its re
15	41	Bhimbetka Rock Shelters	Cultural quarter	A UNESCO World Heritage Site, the Bhimbetka Rock Shelters are known to be one of the oldest archeological

Successfully run. Total query runtime: 336 msec.

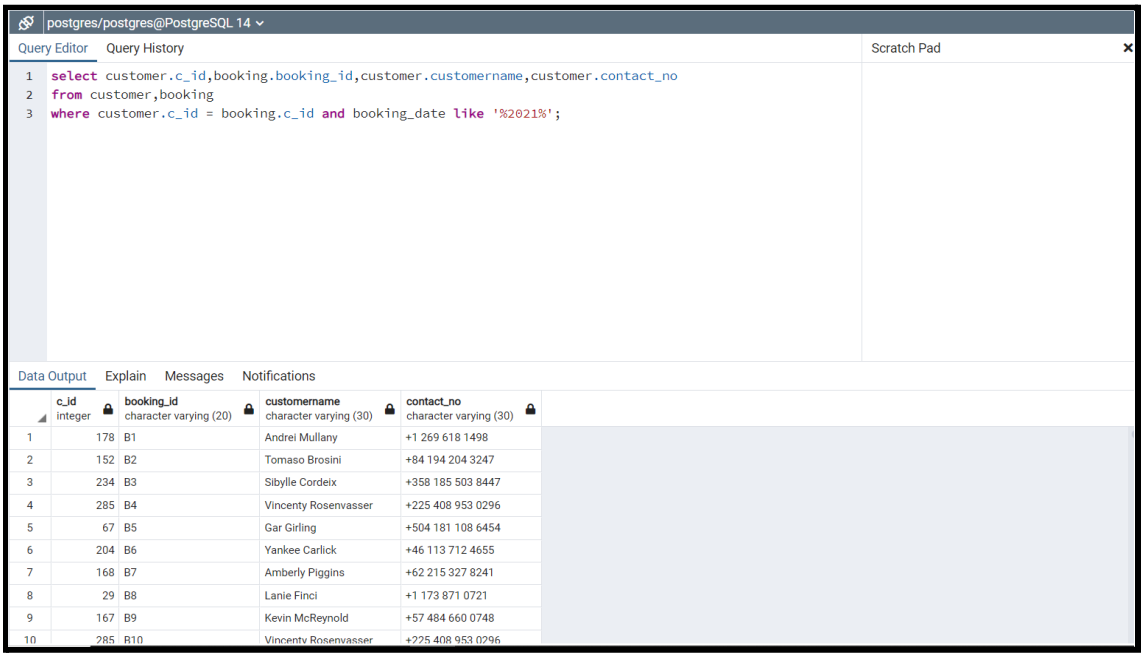
Q2. Display customer_id(leader_id) who have a group of more than 5 members.

```
select leader_id
from group_member
group by leader_id
having count(leader_id)>=5;
```



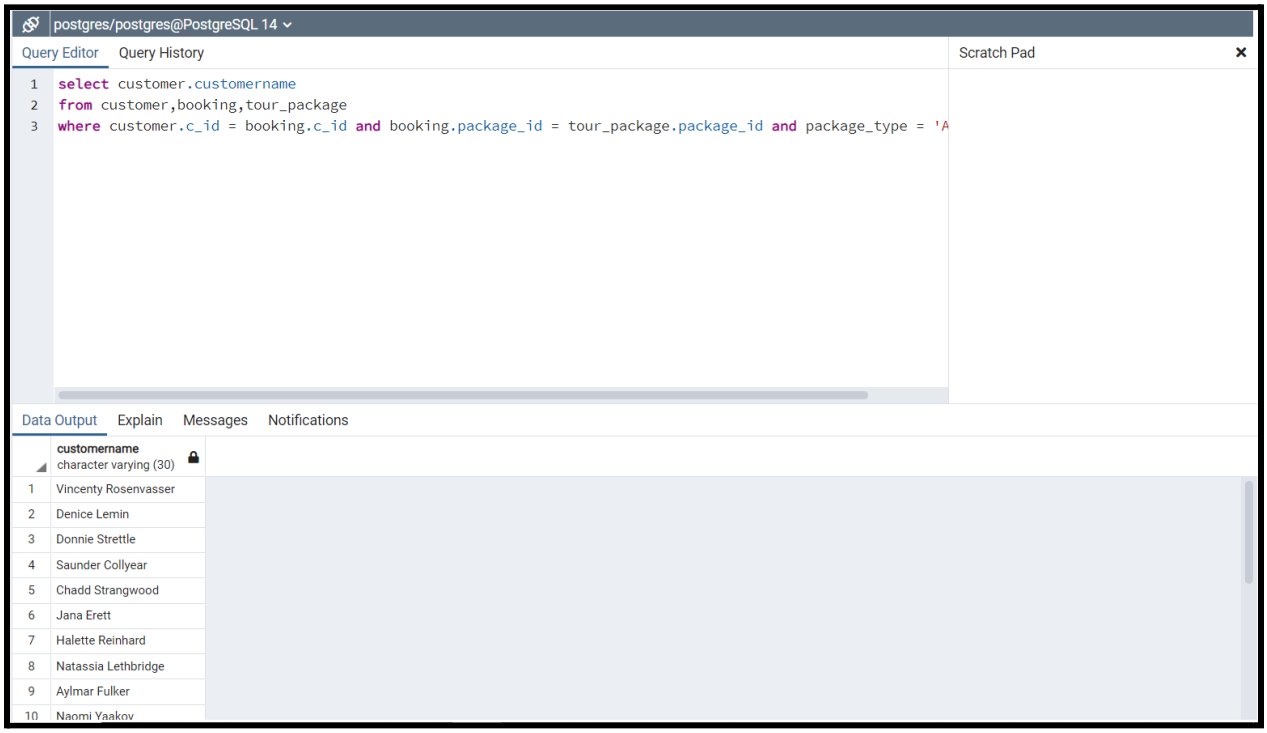
Q3. Print the customer's name, id, contact no. and booking id who have booked their package in the year 2021.

```
select customer.c_id,booking.booking_id, customer.customername,customer.contact_no
from customer,booking
where customer.c_id = booking.c_id and booking_date like '%2021%';
```



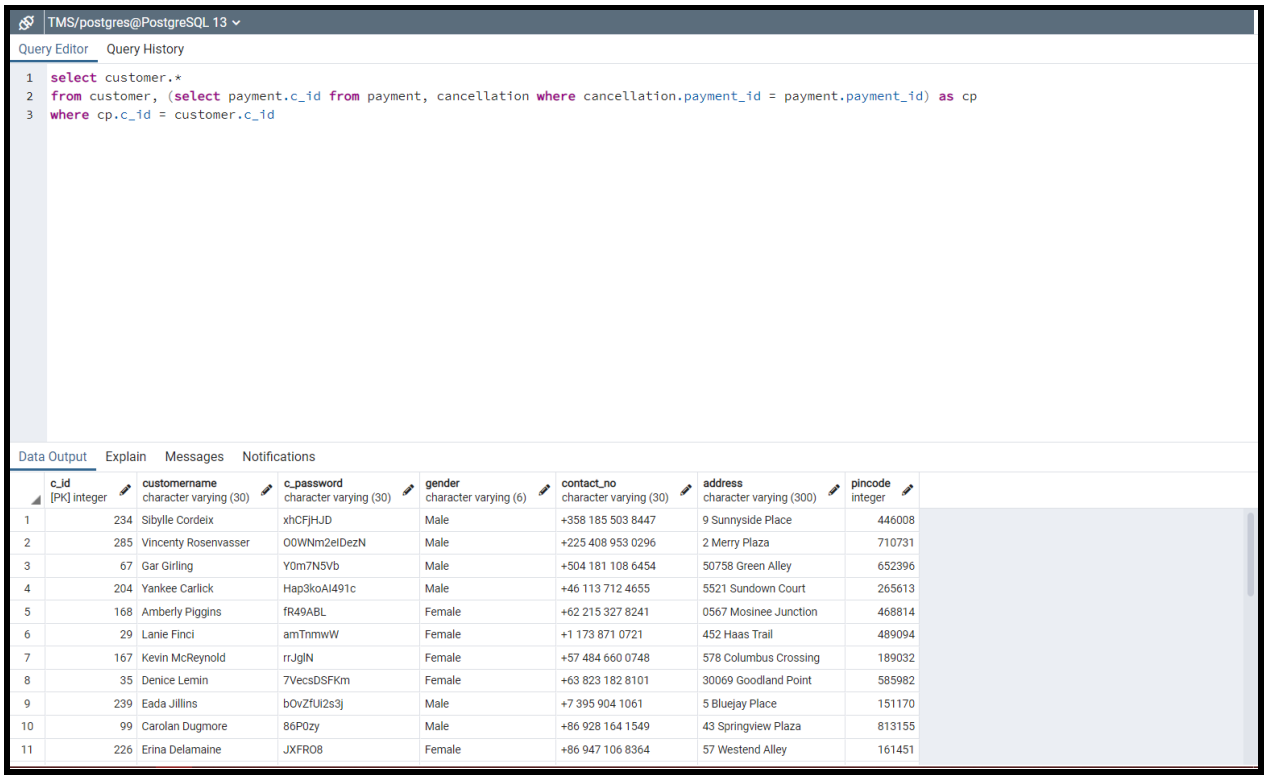
Q4. Show all the customer lists who booked the package of type Adventure.

```
select customer.customername
from customer,booking,tour_package
where customer.c_id = booking.c_id and booking.package_id = tour_package.package_id and package_type =
'Adventure';
```



Q5. List all customer's details who had to cancel their payment.

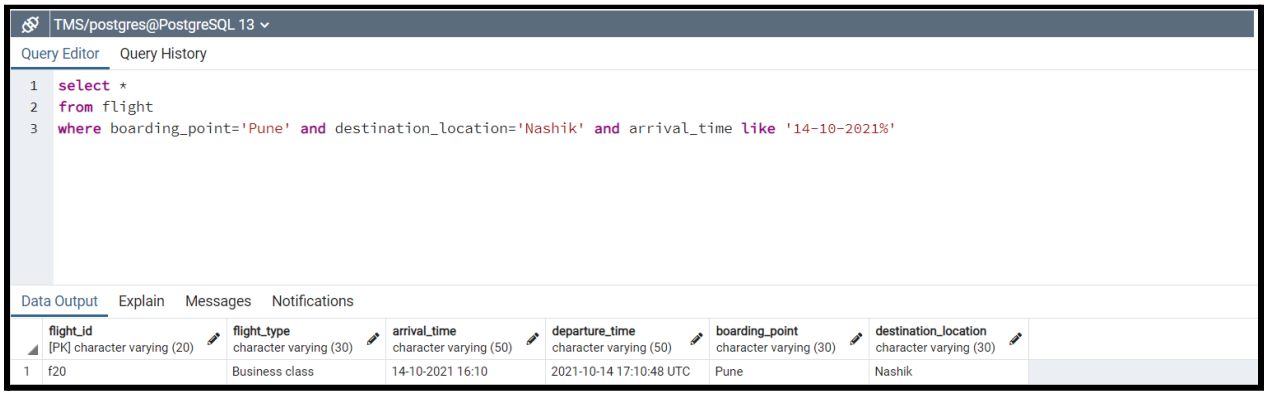
```
select customer.*
from customer, (select payment.c_id from payment, cancellation where cancellation.payment_id =
payment.payment_id) as cp
where cp.c_id = customer.c_id
```



	c_id [PK] integer	customername character varying (30)	c_password character varying (30)	gender character varying (6)	contact_no character varying (30)	address character varying (300)	pincode integer
1	234	Sibylle Cordeix	xhCFjHJD	Male	+358 185 503 8447	9 Sunnyside Place	446008
2	285	Vincenty Rosenvasser	00WNm2eIDezN	Male	+225 408 953 0296	2 Merry Plaza	710731
3	67	Gar Girling	Y0m7N5Vb	Male	+504 181 108 6454	50758 Green Alley	652396
4	204	Yankee Carlick	Hap3koAI491c	Male	+46 113 712 4655	5521 Sundown Court	265613
5	168	Amberly Piggins	fR49ABL	Female	+62 215 327 8241	0567 Mosinee Junction	468814
6	29	Lanie Finci	amTnmwW	Female	+1 173 871 0721	452 Haas Trail	489094
7	167	Kevin McReynold	rrJglN	Female	+57 484 660 0748	578 Columbus Crossing	189032
8	35	Denice Lemin	7VecsDSFKm	Female	+63 823 182 8101	30069 Goodland Point	585982
9	239	Eada Jillins	bOvZfUi2s3j	Male	+7 395 904 1061	5 Bluejay Place	151170
10	99	Carolan Dugmore	86P0zy	Male	+86 928 164 1549	43 Springview Plaza	813155
11	226	Erina Delamaine	JXFRO8	Female	+86 947 106 8364	57 Westend Alley	161451

Q6. List all available flights details from Pune to Nashik on 14-10-2021.

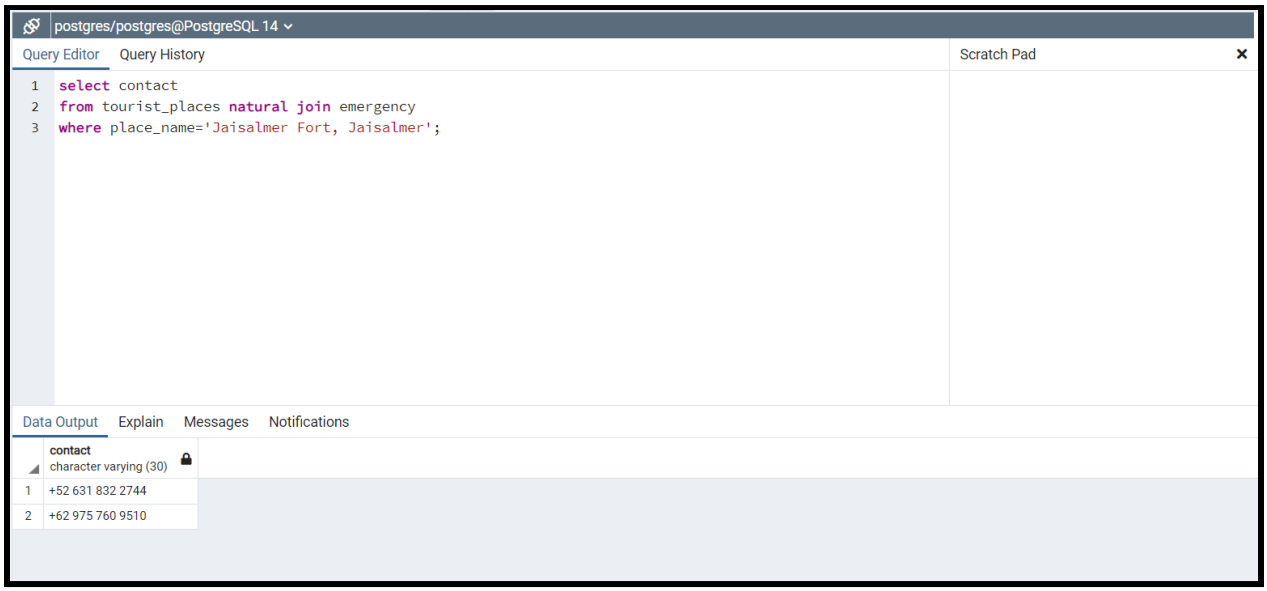
```
select *
from flight
where boarding_point='Pune' and destination_location='Nashik' and arrival_time like '14-10-2021%'
```



	flight_id [PK] character varying (20)	flight_type character varying (30)	arrival_time character varying (50)	departure_time character varying (50)	boarding_point character varying (30)	destination_location character varying (30)
1	f20	Business class	14-10-2021 16:10	2021-10-14 17:10:48 UTC	Pune	Nashik

Q7. Show all emergency contact numbers at Jaisalmer Fort.

```
select contact
from tourist_places natural join emergency
where place_name='Jaisalmer Fort, Jaisalmer';
```

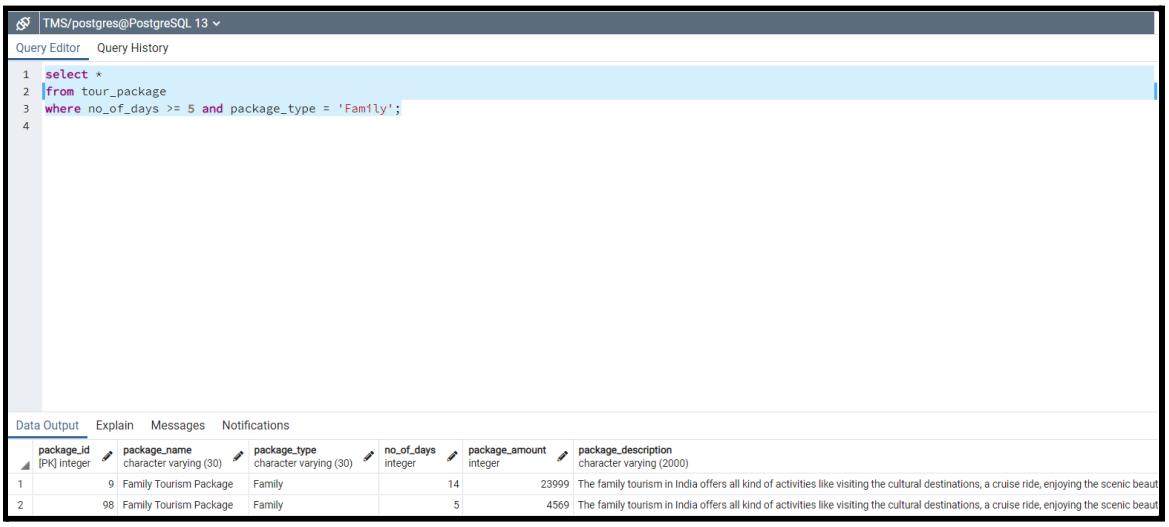


Q8. Find the total amount of profit. (total_profit = sum_of_payment * 20%).

```
select 0.2*sum(amount)
from payment
```

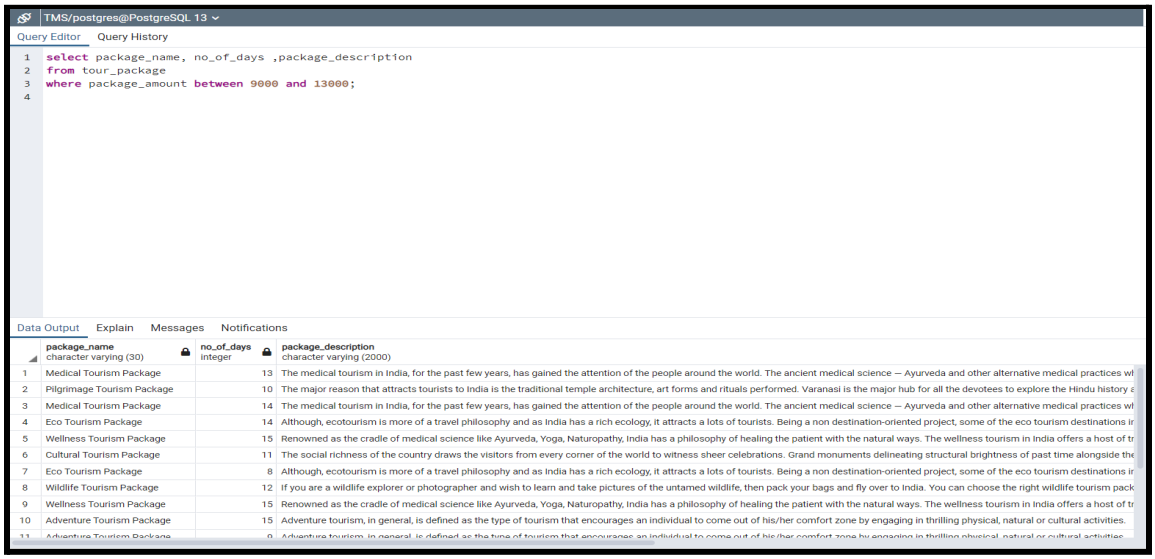
Q9. List the details of the tour package which is suitable for 'Family' and is more than 5 days long.

```
select *
from tour_package
where no_of_days >= 5 and package_type = 'Family';
```



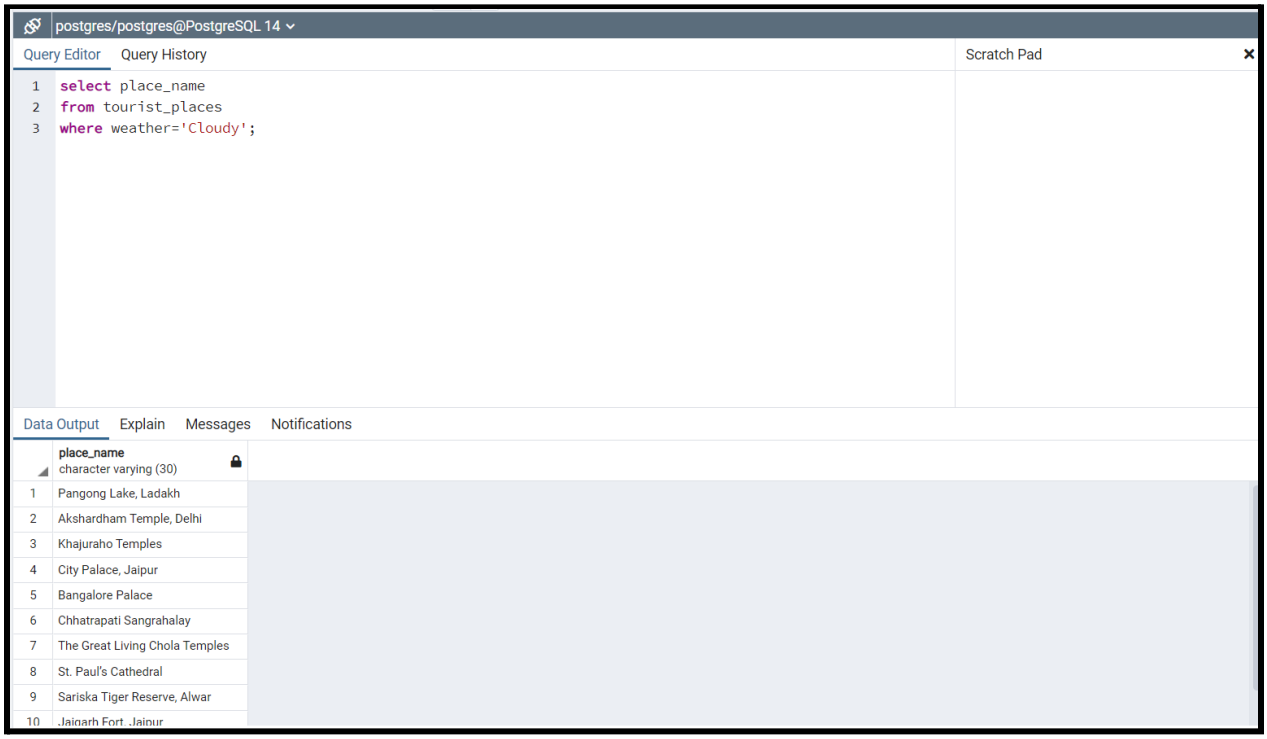
Q10. List the name, description, and no of days of the tour package which amount is between 9000 to 13000.

```
select package_name, no_of_days ,package_description
from tour_package
where package_amount between 9000 and 13000;
```



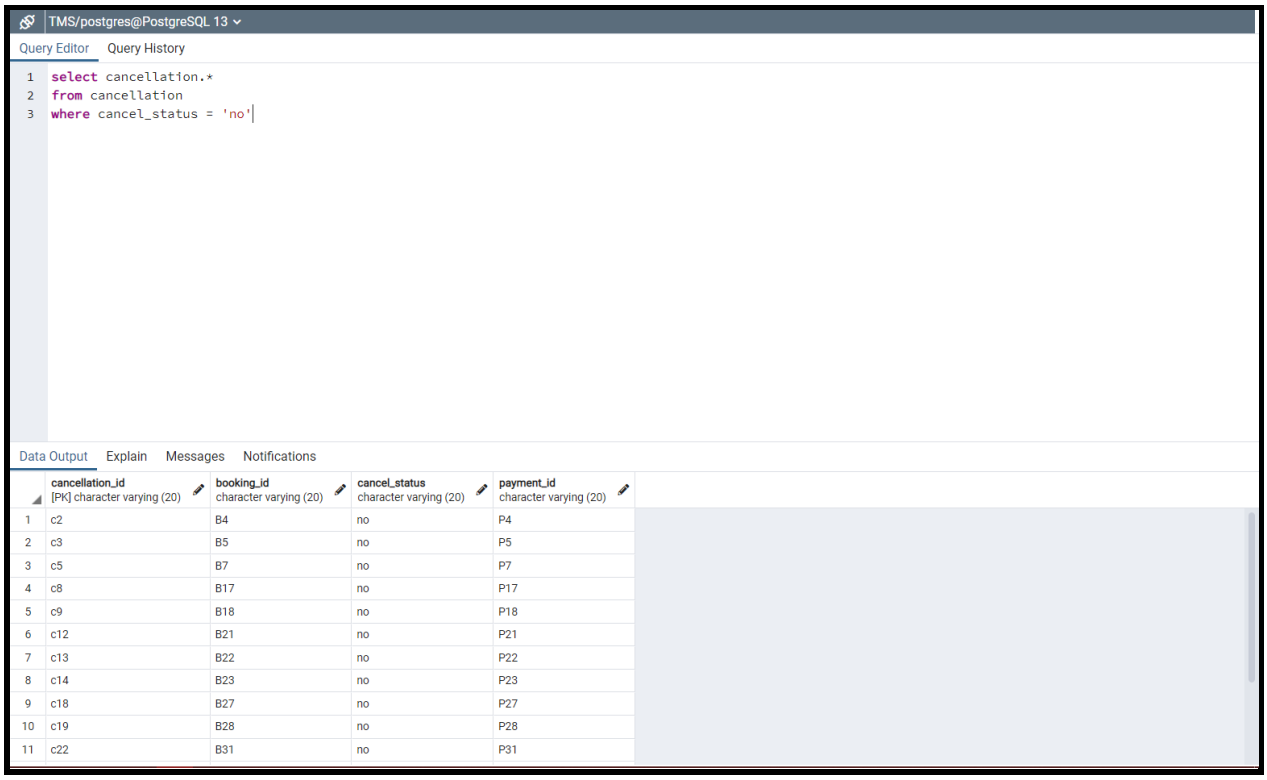
Q11. List all places name where the weather is Partially cloudy.

```
select place_name
from tourist_places
where weather='Cloudy';
```



Q12. List all customer's cancellation details for which cancellation is being processeing.

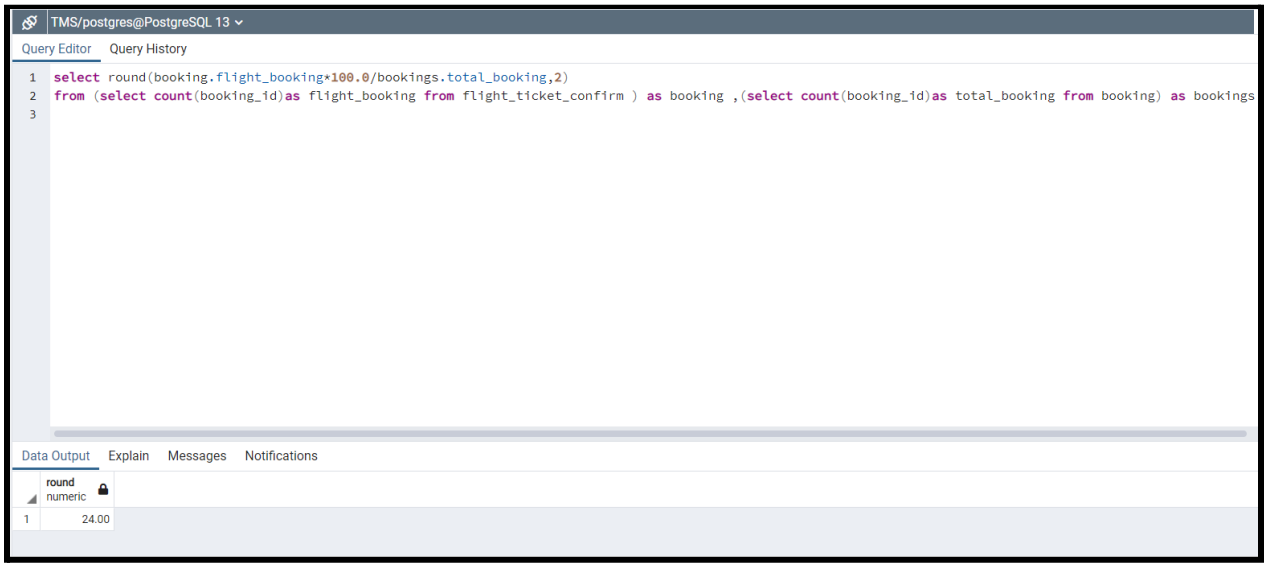
```
select cancellation
from cancellation
where cancel_status = 'no'
```



	cancellation_id [PK] character varying (20)	booking_id character varying (20)	cancel_status character varying (20)	payment_id character varying (20)
1	c2	B4	no	P4
2	c3	B5	no	P5
3	c5	B7	no	P7
4	c8	B17	no	P17
5	c9	B18	no	P18
6	c12	B21	no	P21
7	c13	B22	no	P22
8	c14	B23	no	P23
9	c18	B27	no	P27
10	c19	B28	no	P28
11	c22	B31	no	P31

Q13. Show how many percent of customers traveled by flight.

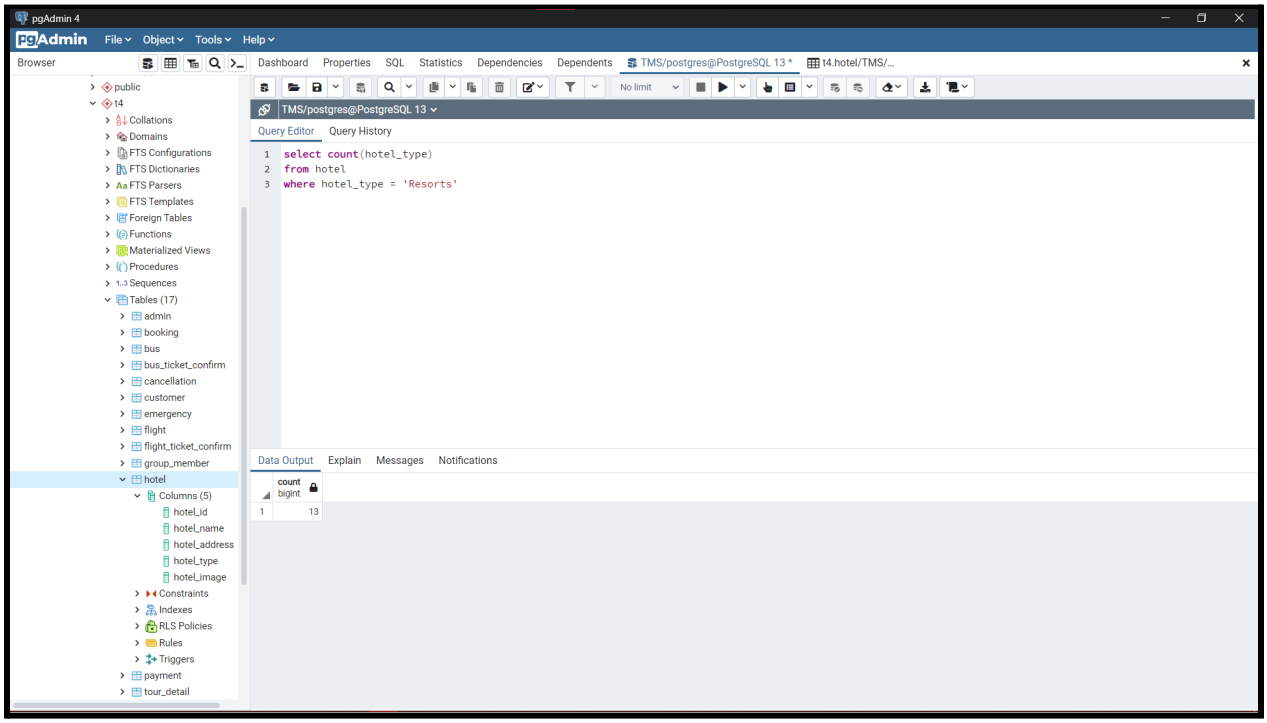
```
select round(booking.flight_booking*100.0/bookings.total_booking,2)
from (select count(booking_id)as flight_booking from flight_ticket_confirm ) as booking ,(select
count(booking_id)as total_booking from booking) as bookings
```



	round numeric
1	24.00

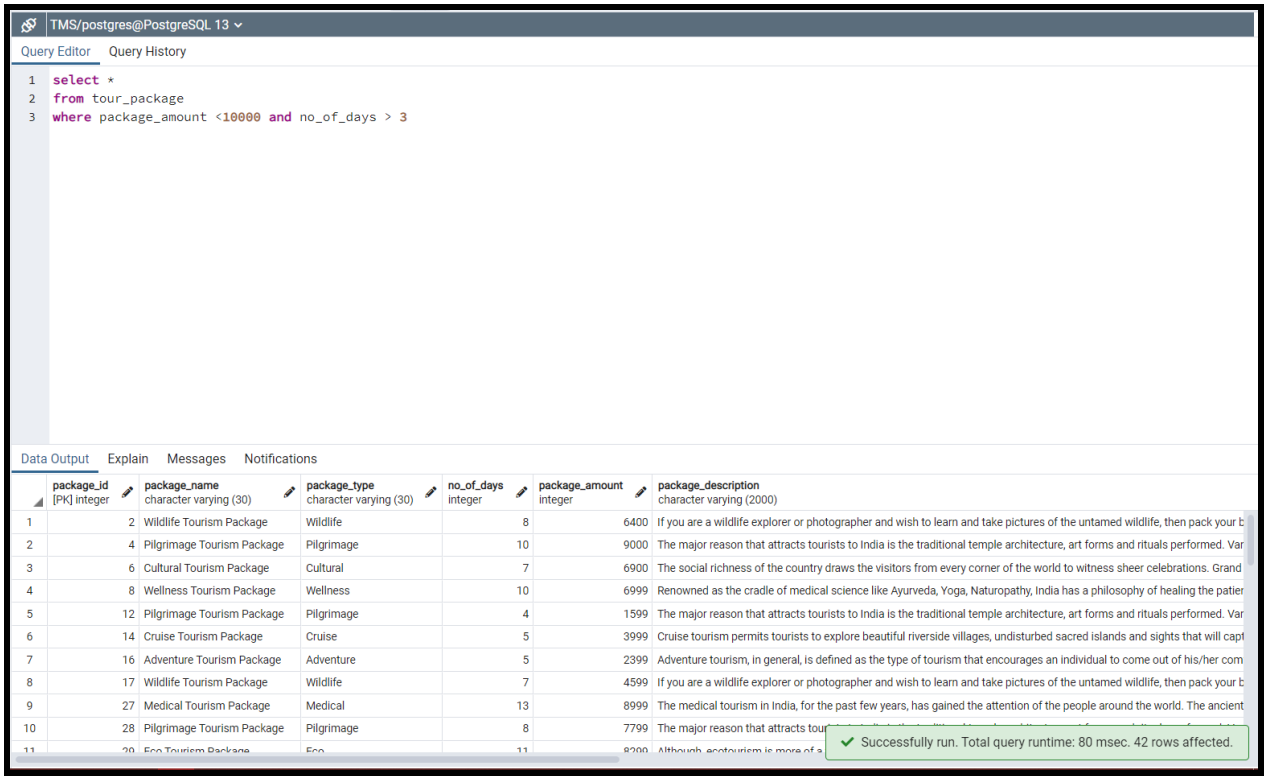
Q14. Show how many hotels are in the resorts.

```
select count(hotel_type)
from hotel
where hotel_type = 'Resorts'
```



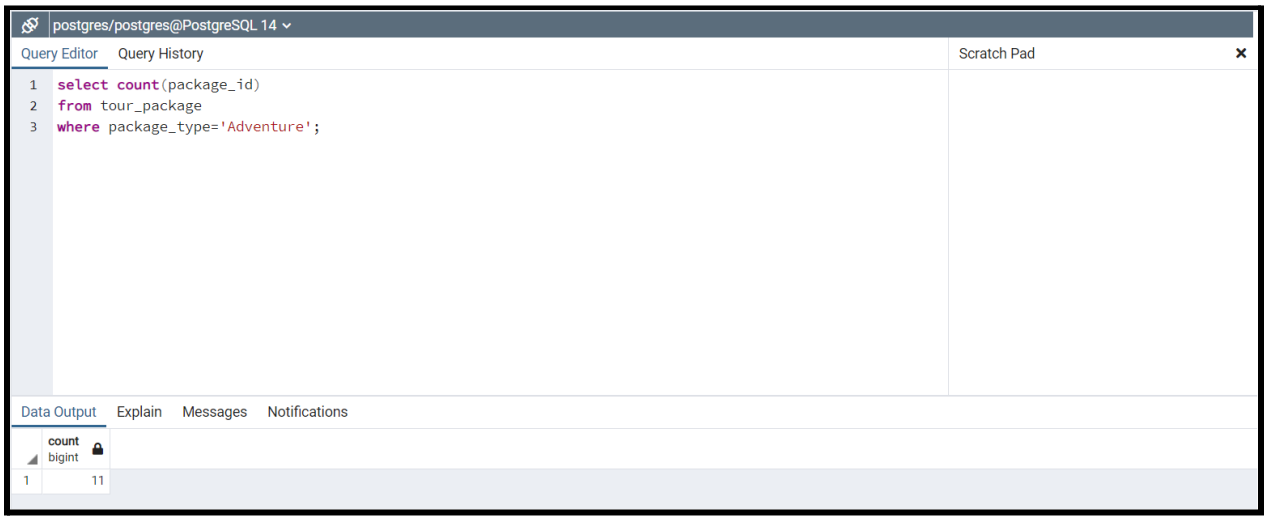
Q15. Give all tour package details which are more than 3 days long and whose package amount is less than 10000.

```
select *
from tour_package
where package_amount <10000 and no_of_days > 3
```



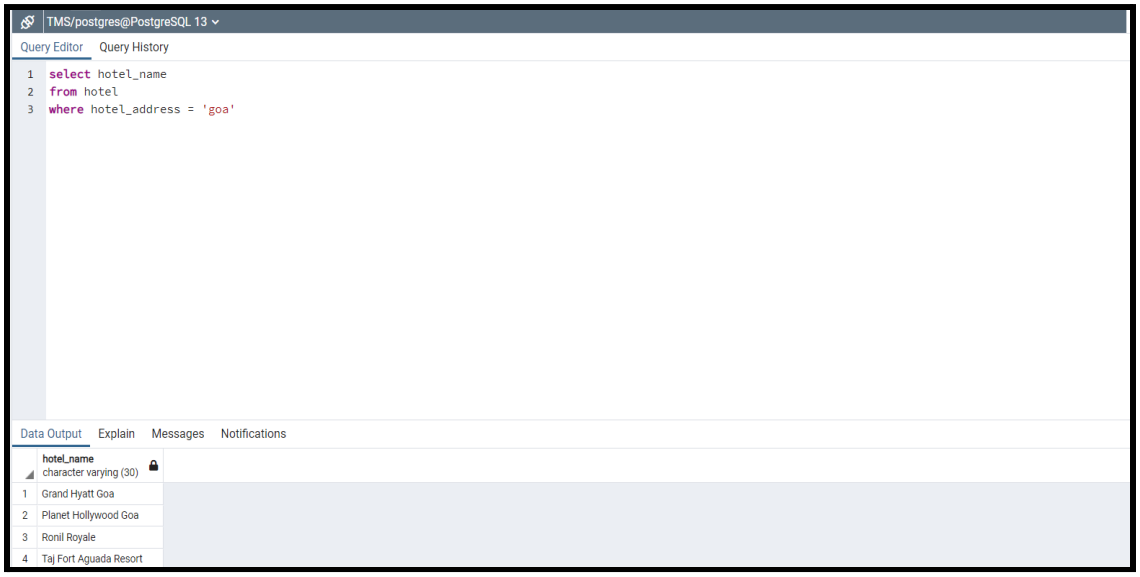
Q16. Show how many packages are available of type adventure.

```
select count(package_id)
from tour_package
where package_type='Adventure';
```



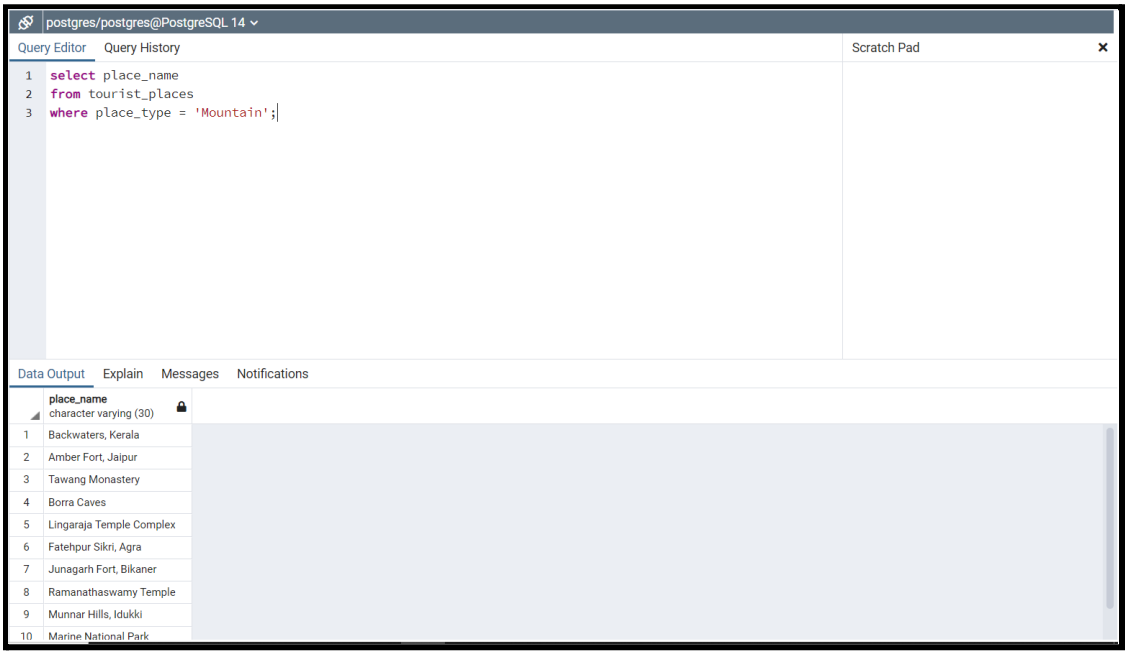
Q17. Show list of all hotels name that is from Goa.

```
select hotel_name
from hotel
where hotel_address = 'goa'
```



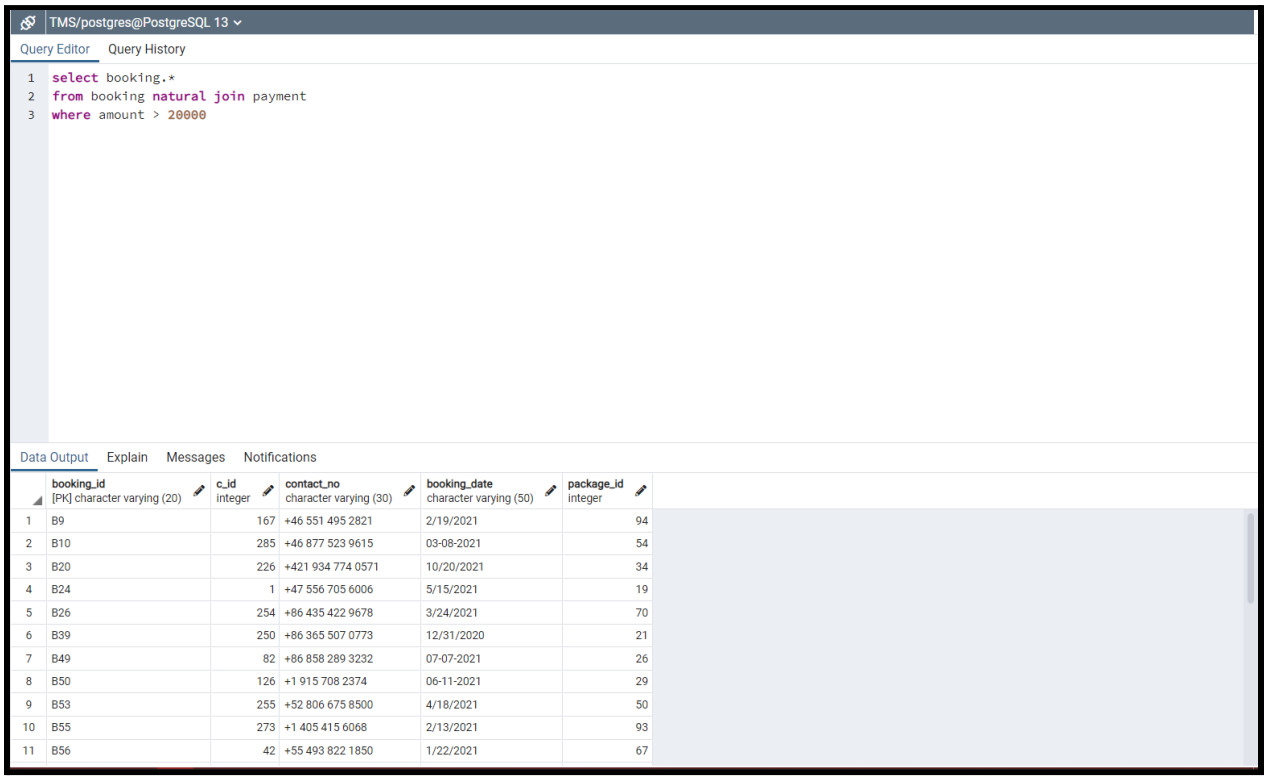
Q18. List the places name which place type is Mountain.

```
select place_name
from tourist_places
where place_type = 'Mountain';
```



Q19. Show all booking information of payments which amount is greater than 20000.

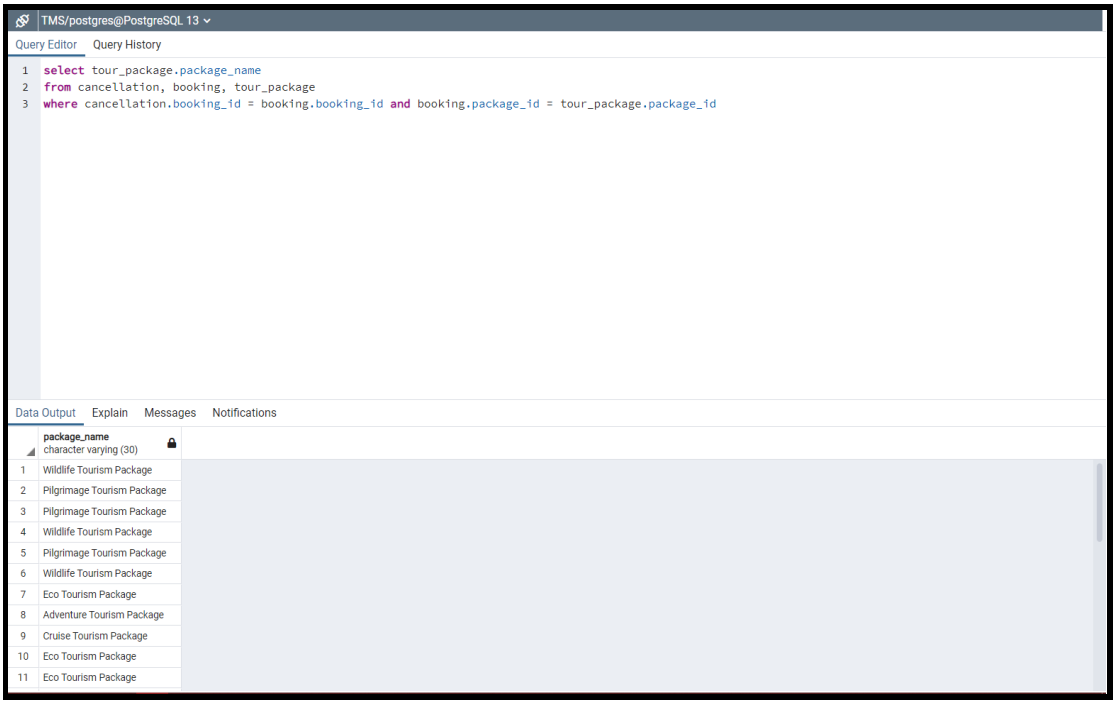
```
select booking.*
from booking natural join payment
where amount > 20000
```



	booking_id [PK] character varying (20)	c_id integer	contact_no character varying (30)	booking_date character varying (50)	package_id integer
1	B9	167	+46 551 495 2821	2/19/2021	94
2	B10	285	+46 877 523 9615	03-08-2021	54
3	B20	226	+421 934 774 0571	10/20/2021	34
4	B24	1	+47 556 705 6006	5/15/2021	19
5	B26	254	+86 435 422 9678	3/24/2021	70
6	B39	250	+86 365 507 0773	12/31/2020	21
7	B49	82	+86 858 289 3232	07-07-2021	26
8	B50	126	+1 915 708 2374	06-11-2021	29
9	B53	255	+52 806 675 8500	4/18/2021	50
10	B55	273	+1 405 415 6068	2/13/2021	93
11	B56	42	+55 493 822 1850	1/22/2021	67

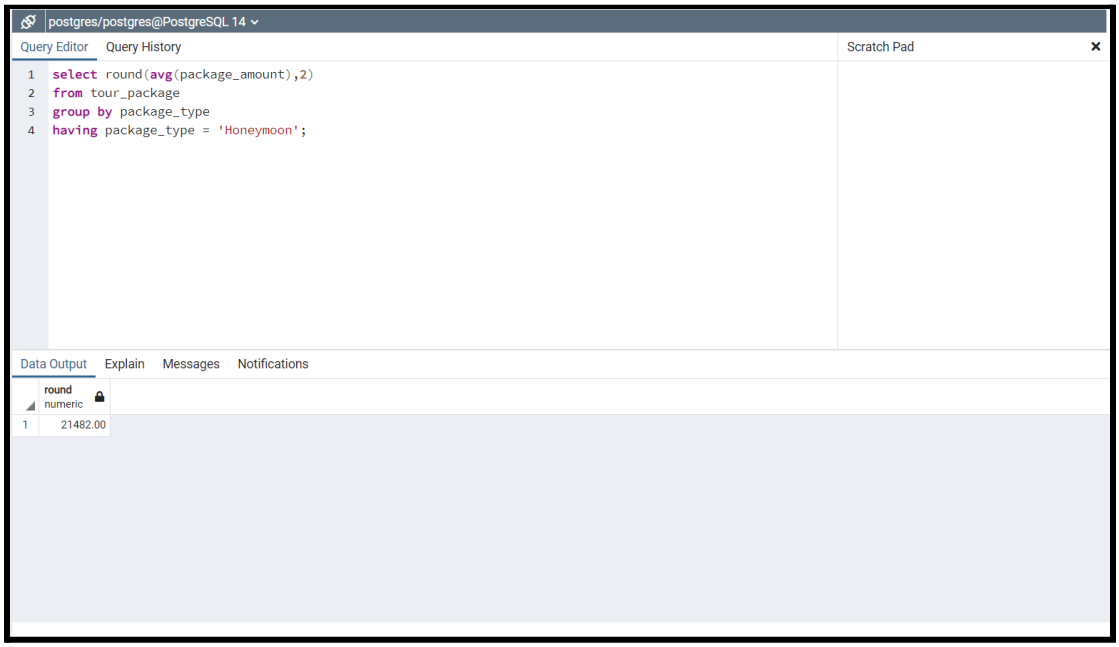
Q20. Show the list of all canceled package_name.

```
select tour_package.package_name
from cancellation, booking, tour_package
where cancellation.booking_id = booking.booking_id and booking.package_id = tour_package.package_id
```



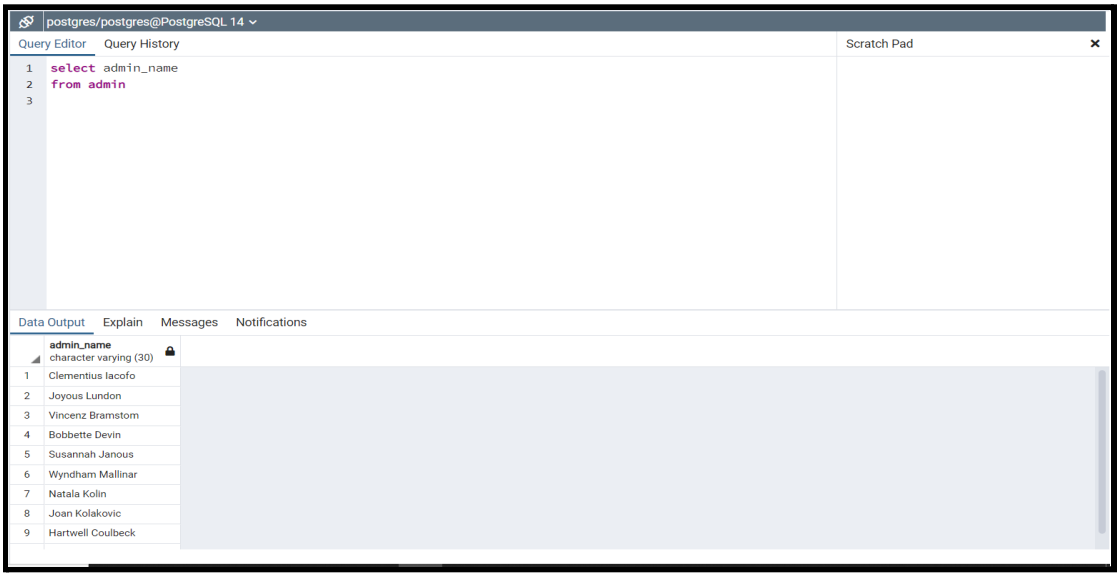
Q21. Show the average amount of Honeymoon packages?

```
select round(avg(package_amount),2)
from tour_package
group by package_type
having package_type = 'Honeymoon';
```



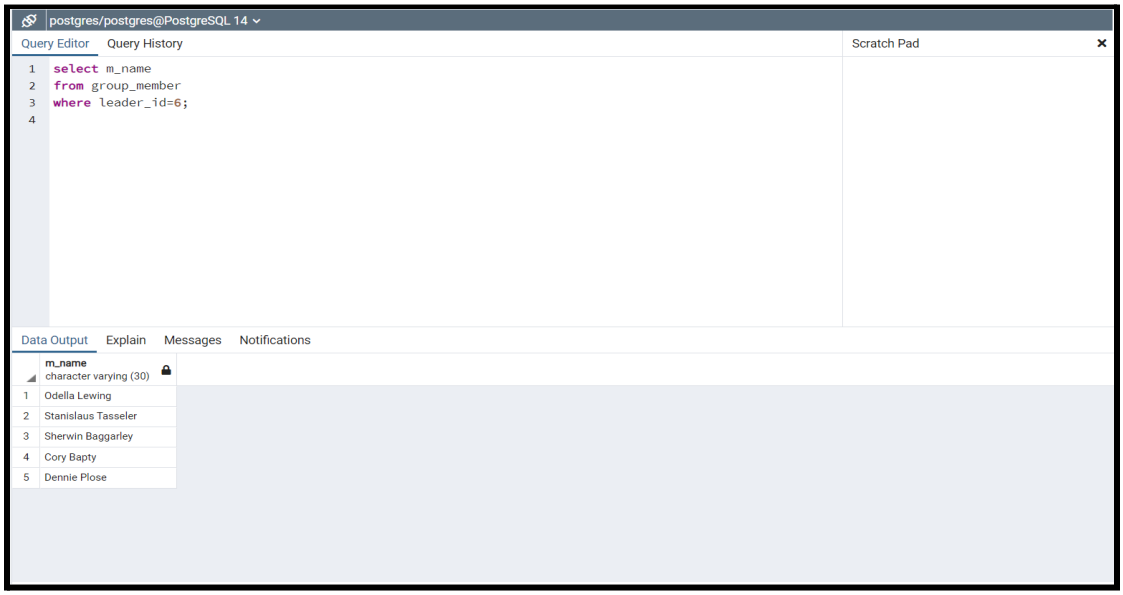
Q22. Show all admin names.

```
select admin_name
from admin
```



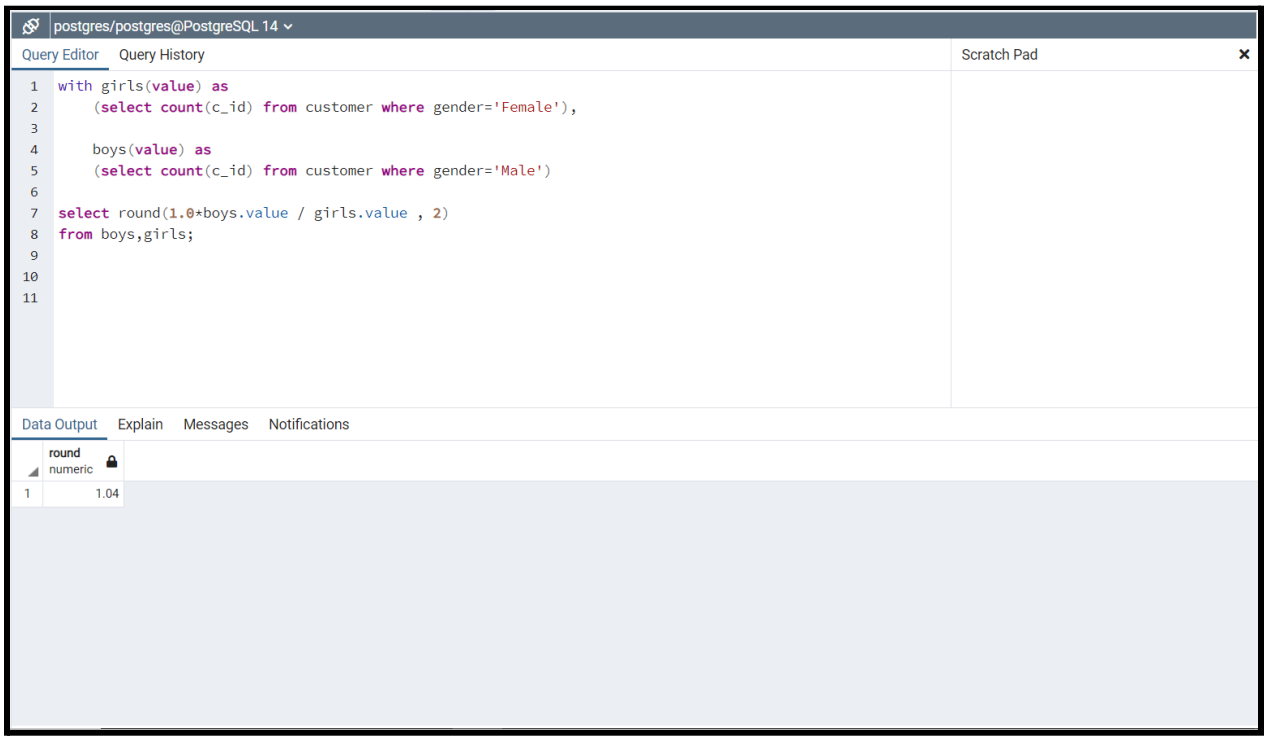
Q23. Display all member names of customer id 10.

```
select m_name
from group_member
where leader_id=6;
```



Q24. Show the ratio of male and female customers.

```
with girls(value) as
        (select count(c_id) from customer where gender='Female'),

        boys(value) as
        (select count(c_id) from customer where gender='Male')

select round(1.0*boys.value / girls.value , 2)
from boys, girls;
```



Q25. Display the highest paid amount till now.

```
select max(amount)
from payment
```

Q26. Show how many customers have canceled their booking.

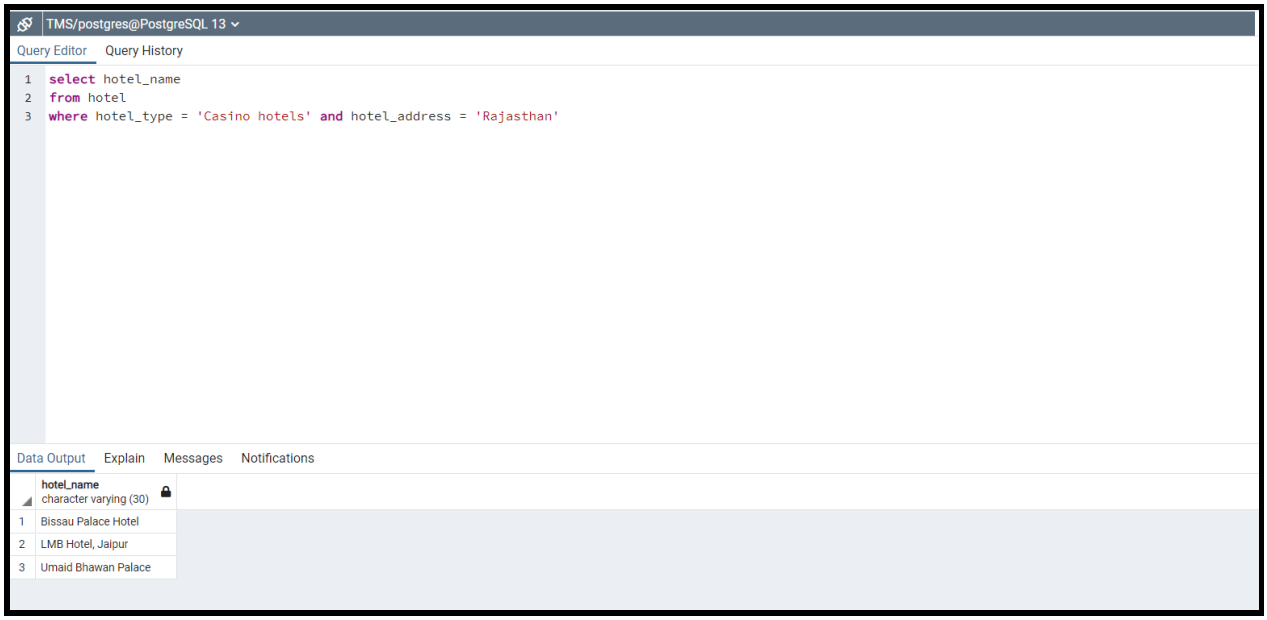
```
select count(cancellation_id)
from cancellation
where cancel_status = 'yes';
```

Q27. Show the arrival time and departure time of the train whose train id is t46.

```
select arrival_time,departure_time
from train
where train_id='t46';
```

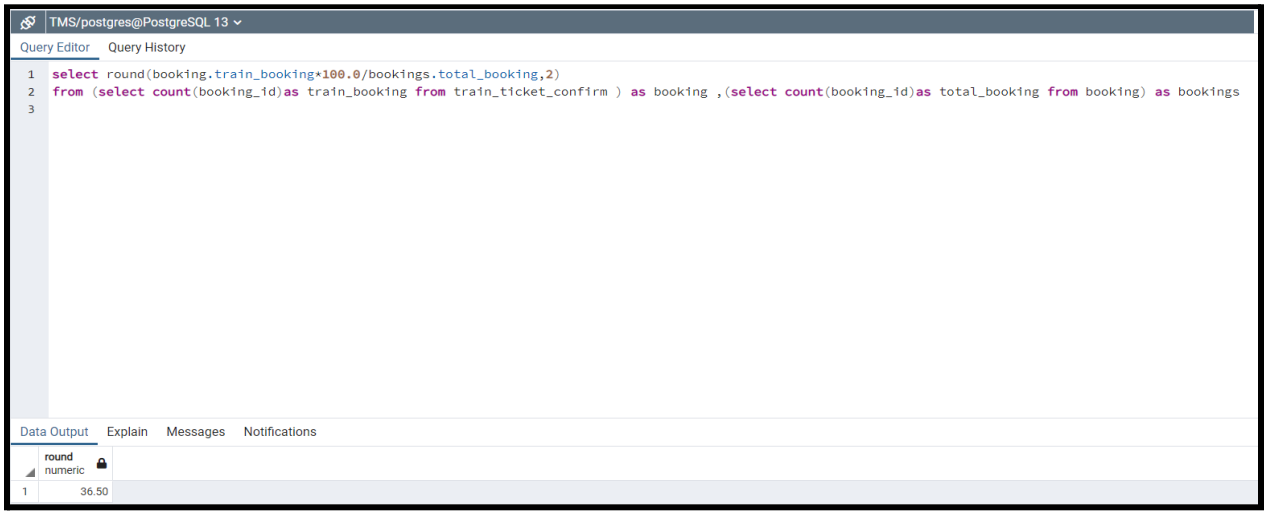
Q28. List all casino hotels' names that are situated in Rajasthan.

```
select hotel_name
from hotel
where hotel_type = 'Casino hotels' and hotel_address = 'Rajasthan'
```



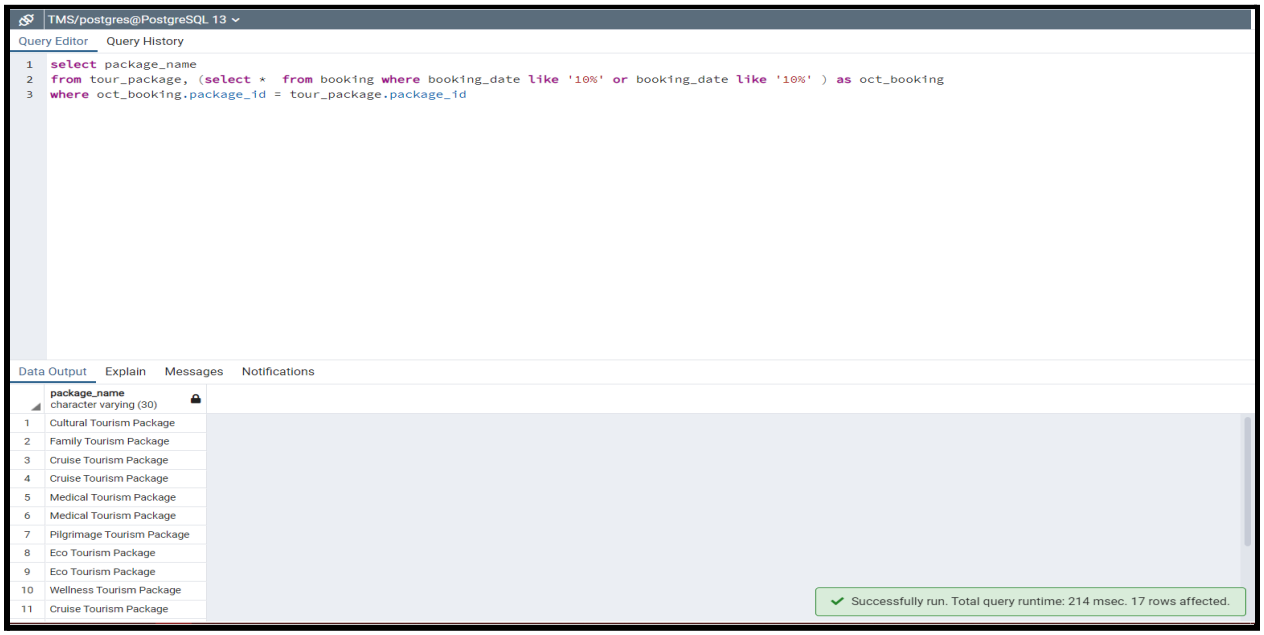
Q29. Show how many percent of customers traveled by train.

```
select round(booking.train_booking*100.0/bookings.total_booking,2)
from (select count(booking_id)as train_booking from train_ticket_confirm ) as booking ,(select
count(booking_id)as total_booking from booking) as bookings
```



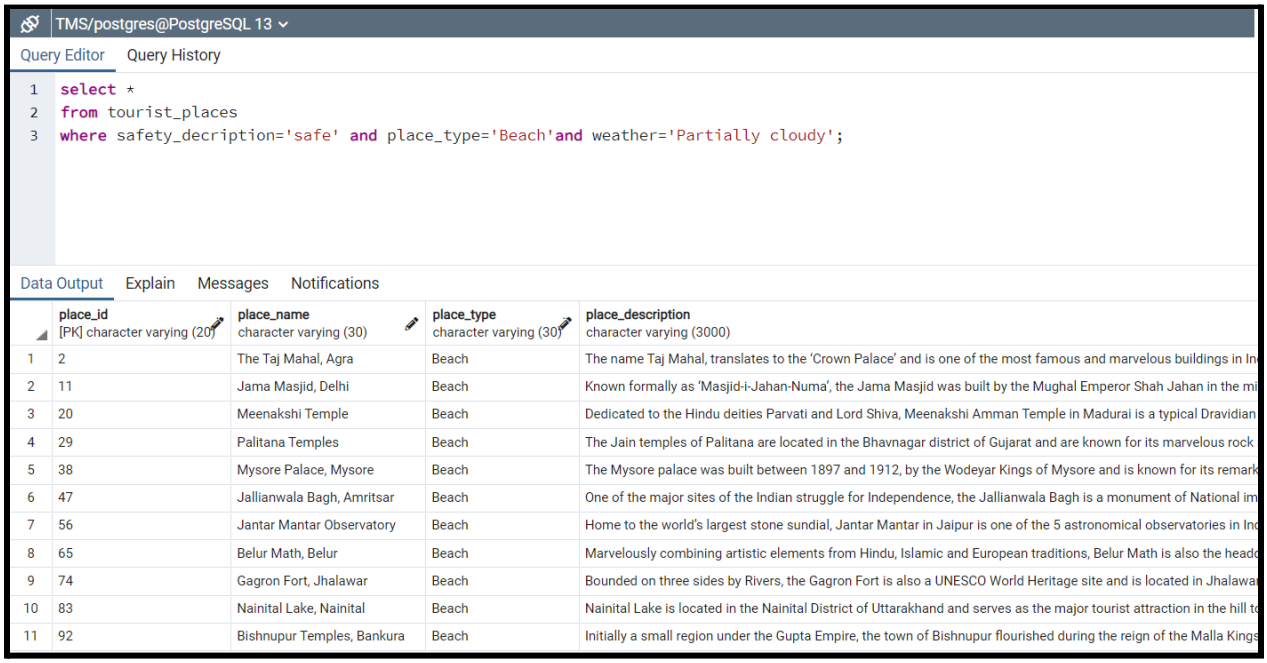
Q30. List all the package names which are booked in october month.

```
select package_name
from tour_package, (select *  from booking where booking_date like '10%' or booking_date like '10%' ) as
oct_booking
where oct_booking.package_id = tour_package.package_id
```



Q31. List all the tourist places details which are beach and safe to visit and their weather is Partially cloudy.

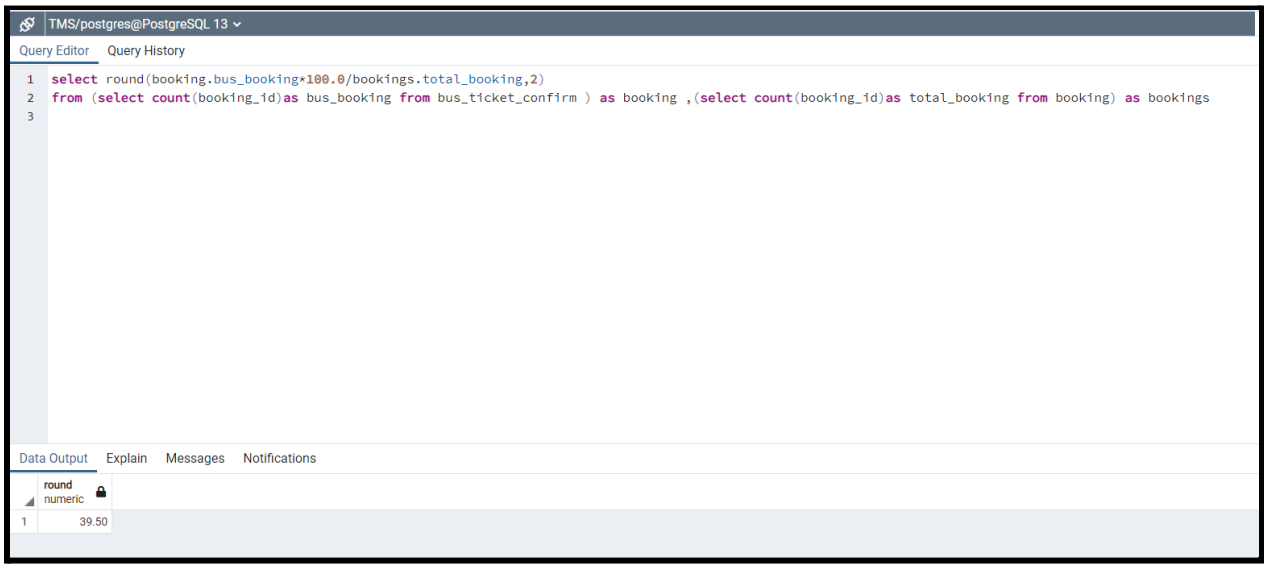
```
select *
from tourist_places
where safety_decription='safe' and place_type='Beach' and weather='Partially cloudy';
```



	place_id [PK] character varying (20)	place_name character varying (30)	place_type character varying (30)	place_description character varying (3000)
1	2	The Taj Mahal, Agra	Beach	The name Taj Mahal, translates to the 'Crown Palace' and is one of the most famous and marvelous buildings in In
2	11	Jama Masjid, Delhi	Beach	Known formally as 'Masjid-i-Jahan-Numa', the Jama Masjid was built by the Mughal Emperor Shah Jahan in the mi
3	20	Meenakshi Temple	Beach	Dedicated to the Hindu deities Parvati and Lord Shiva, Meenakshi Amman Temple in Madurai is a typical Dravidian
4	29	Palitana Temples	Beach	The Jain temples of Palitana are located in the Bhavnagar district of Gujarat and are known for its marvelous rock
5	38	Mysore Palace, Mysore	Beach	The Mysore palace was built between 1897 and 1912, by the Wodeyar Kings of Mysore and is known for its remark
6	47	Jallianwala Bagh, Amritsar	Beach	One of the major sites of the Indian struggle for Independence, the Jallianwala Bagh is a monument of National im
7	56	Jantar Mantar Observatory	Beach	Home to the world's largest stone sundial, Jantar Mantar in Jaipur is one of the 5 astronomical observatories in Ind
8	65	Belur Math, Belur	Beach	Marvelously combining artistic elements from Hindu, Islamic and European traditions, Belur Math is also the headc
9	74	Gagron Fort, Jhalawar	Beach	Bounded on three sides by Rivers, the Gagron Fort is also a UNESCO World Heritage site and is located in Jhalawar
10	83	Nainital Lake, Nainital	Beach	Nainital Lake is located in the Nainital District of Uttarakhand and serves as the major tourist attraction in the hill to
11	92	Bishnupur Temples, Bankura	Beach	Initially a small region under the Gupta Empire, the town of Bishnupur flourished during the reign of the Malla Kings

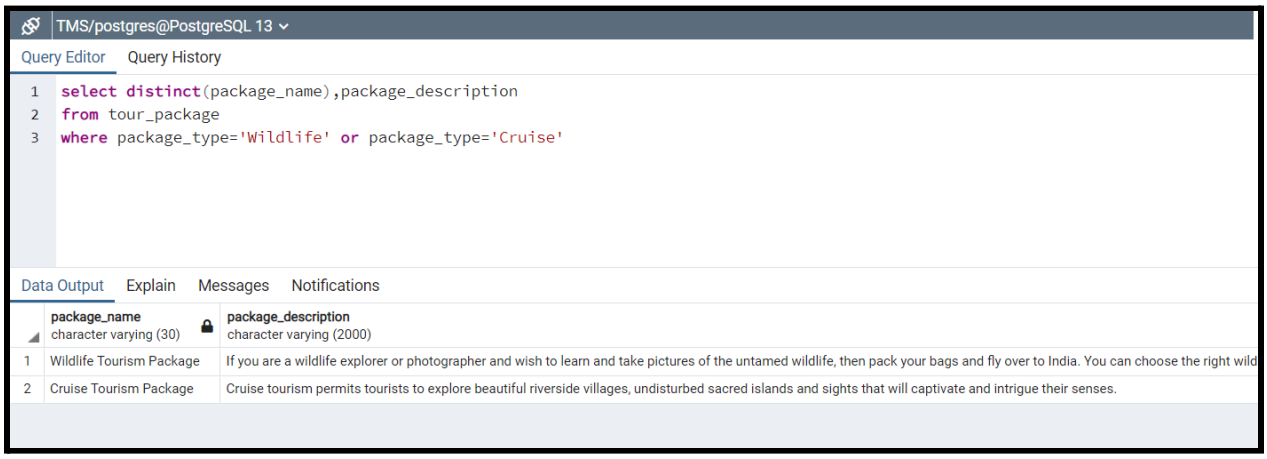
Q32. Show how many percent of customers traveled by bus.

```
select round(booking.bus_booking*100.0/bookings.total_booking,2)
from (select count(booking_id) as bus_booking from bus_ticket_confirm ) as booking , (select
count(booking_id)as total_booking from booking) as bookings
```



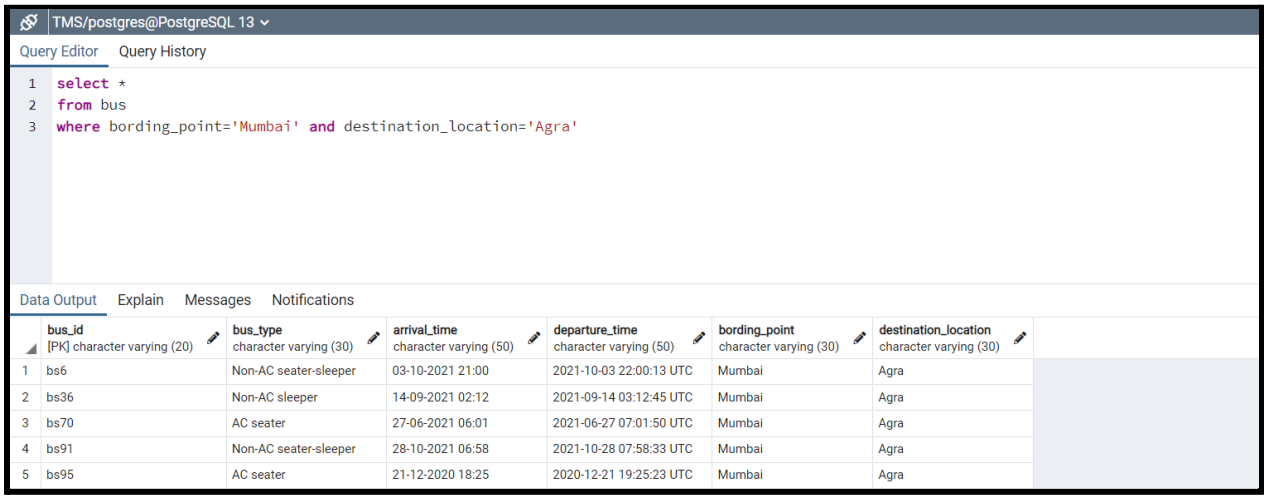
Q33. Give the package description of Wildlife and Cruise package type.

```
select distinct(package_name),package_description
from tour_package
where package_type='Wildlife' or package_type='Cruise'
```



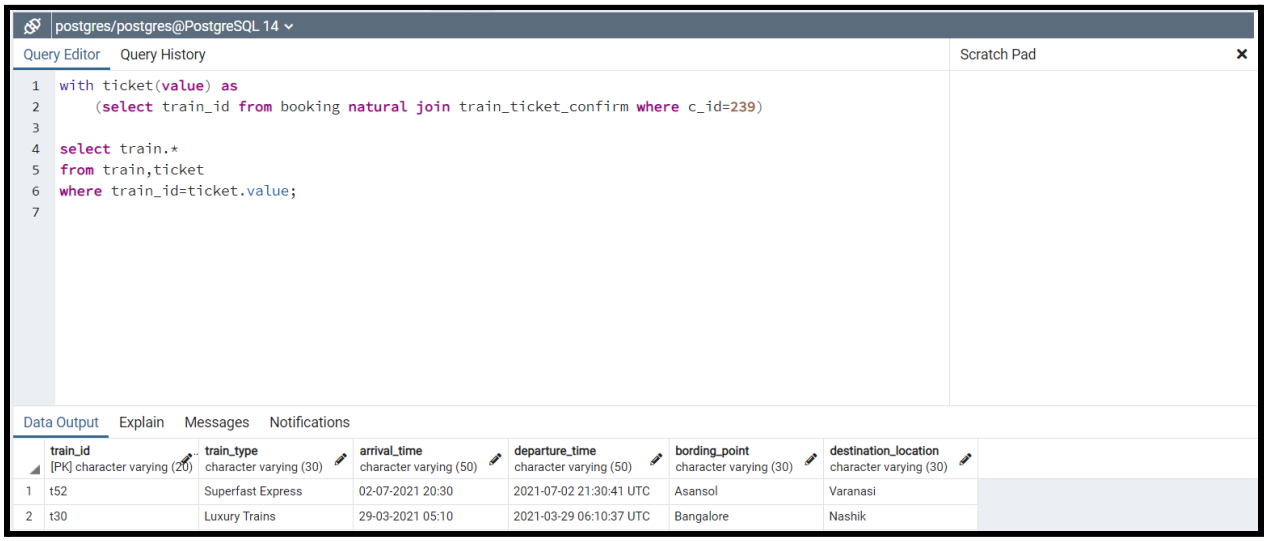
Q34. List all buses details from Mumbai to Agra.

```
select *
from bus
where bording_point='Mumbai' and destination_location='Agra'
```



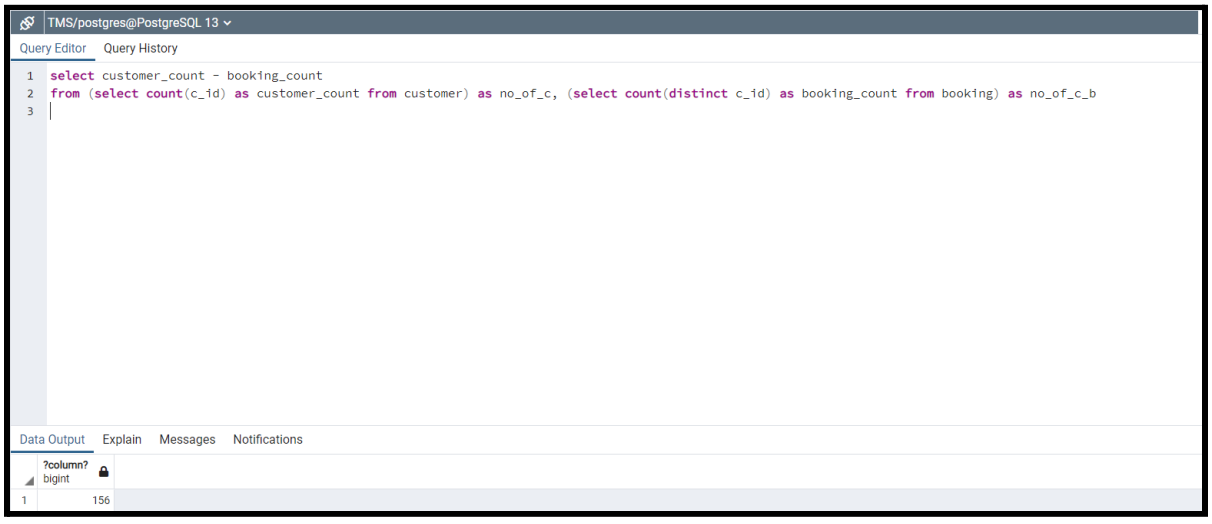
Q35. Display information about the train of customer id 239.

```
with ticket(value) as
        (select train_id from booking natural join train_ticket_confirm where c_id=239)
select train.*
from train, ticket
where train_id=ticket.value;
```



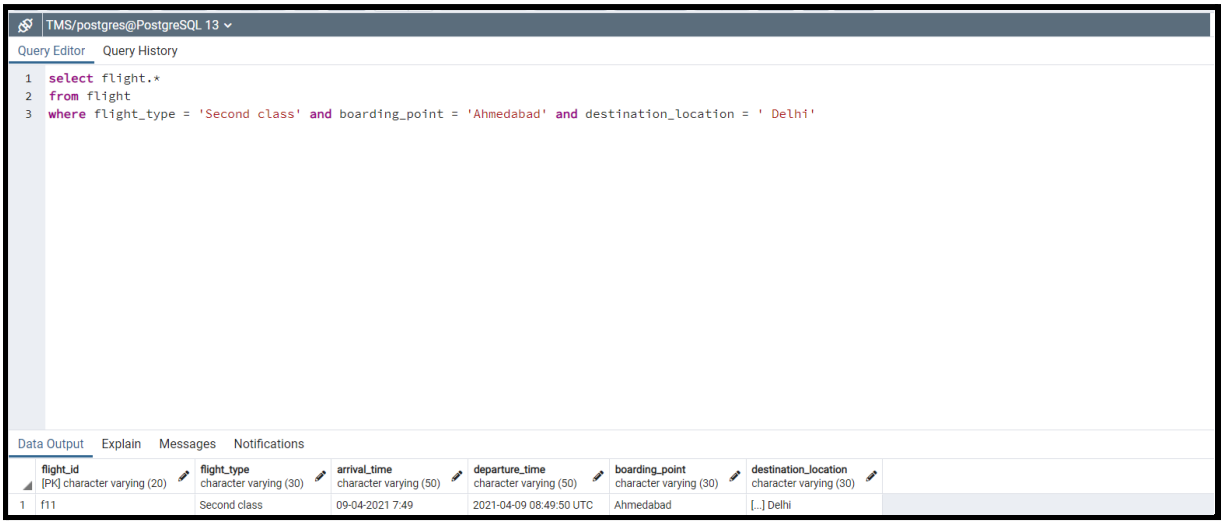
Q36. How many customers have not booked any package yet?

```
select customer_count - booking_count
from (select count(c_id) as customer_count from customer) as no_of_c, (select count(distinct c_id) as
booking_count from booking) as no_of_c_b
```



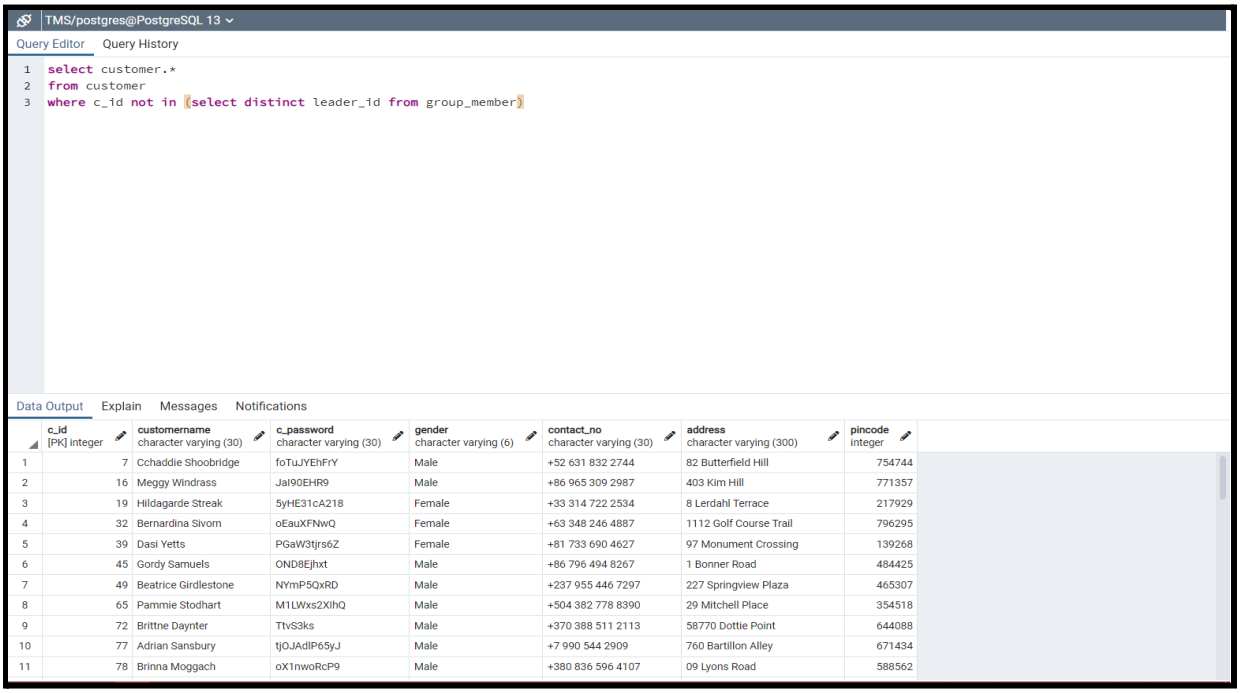
Q37. Show Second class Flight from Ahmedabad to Delhi.

```
select flight.*
from flight
where flight_type = 'Second class' and boarding_point = 'Ahmedabad' and destination_location = ' Delhi'
```



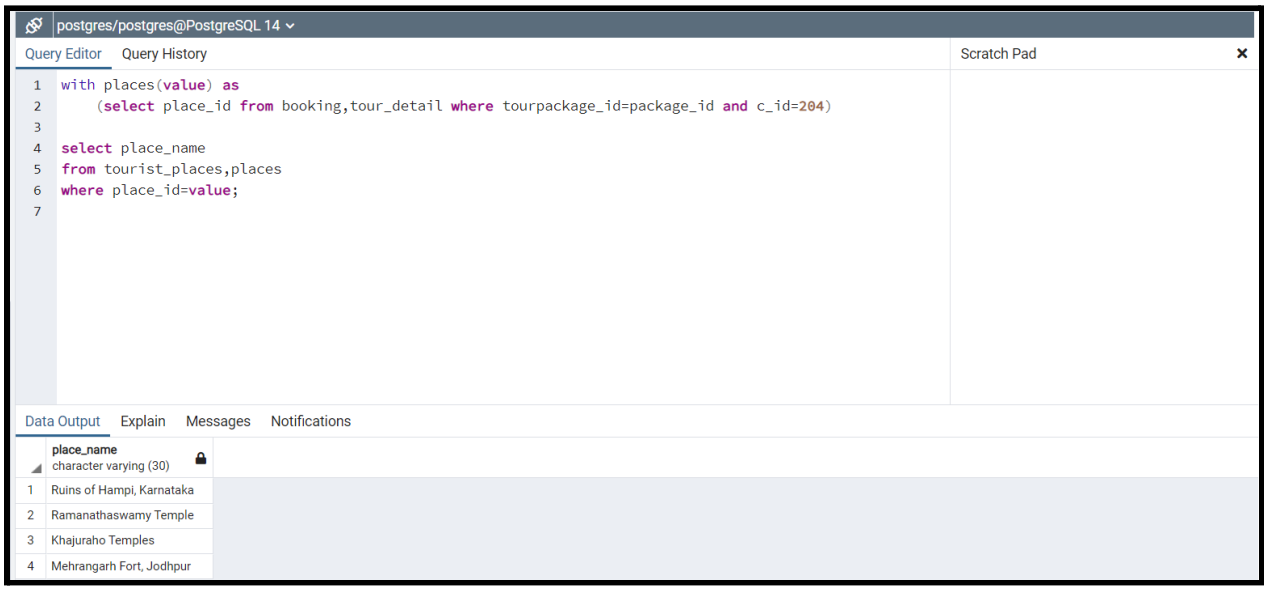
Q38. List details of the customer which doesn't have any members with it.

```
select customer.*
from customer
where c_id not in (select distinct leader_id from group_member)
```



Q39. Show all places name visited by customer id 204.

```
with places(value) as
        (select place_id from booking,tour_detail where tourpackage_id=package_id and c_id=204)

select place_name
from tourist_places, places
where place_id=value;
```



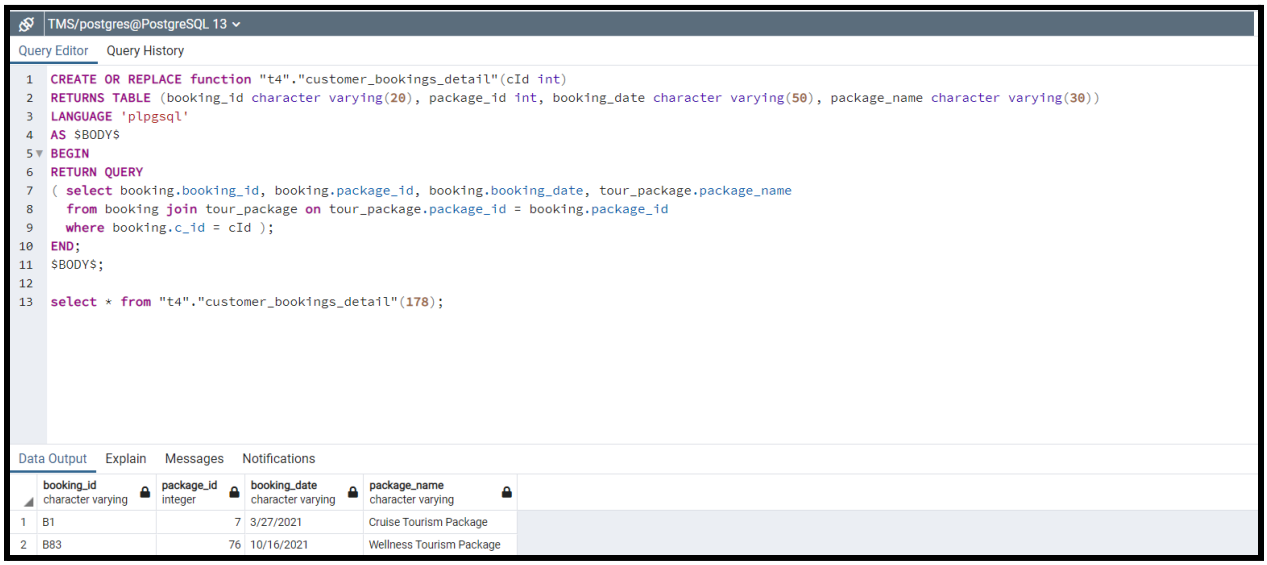
Q40. List the name of the place which is most visited.

```
create view packageplaces(tourpackagecount,pid) as
select count(tourpackage_id), place_id
        from (select tour_detail.tourpackage_id, tour_detail.place_id
      from tour_detail, booking
      where booking.package_id = tour_detail.tourpackage_id) as spp
        group by place_id;

select tourist_places.*
from packageplaces,tourist_places
where packageplaces.pid = tourist_places.place_id and tourpackagecount = (select max(tourpackagecount) from
packageplaces);
```

Function

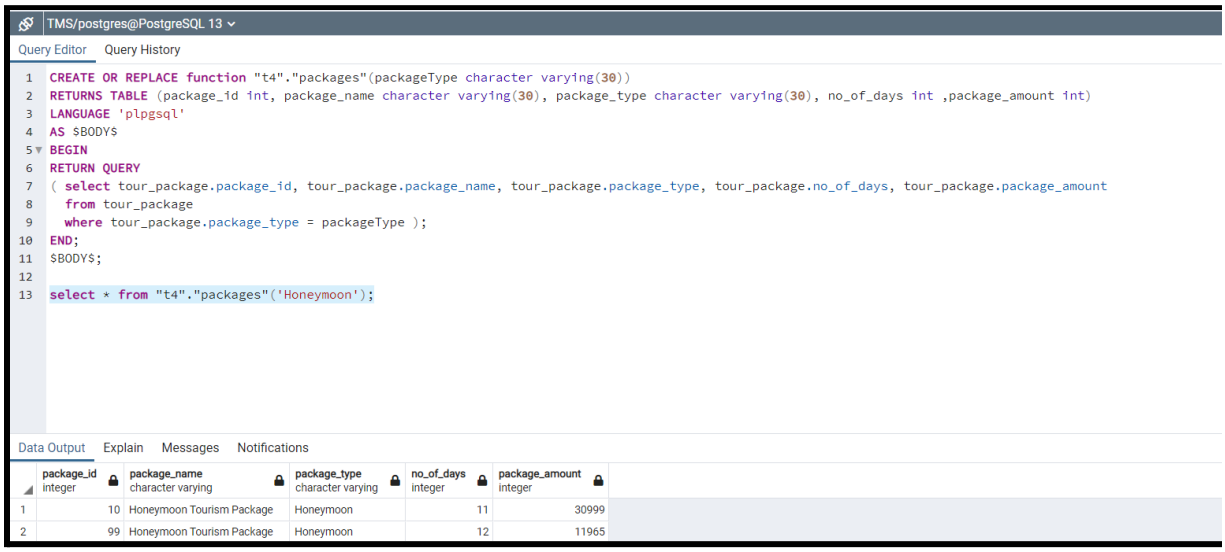
1. Function which shows booking details of customers.

```
CREATE OR REPLACE function "t4"."customer_bookings_detail"(cId int)
RETURNS TABLE (booking_id character varying(20), package_id int, booking_date character varying(50),
package_name character varying(30))
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
RETURN QUERY
( select booking.booking_id, booking.package_id, booking.booking_date, tour_package.package_name
  from booking join tour_package on tour_package.package_id = booking.package_id
  where booking.c_id = cId );
END;
$BODY$;

select * from "t4"."customer_bookings_detail"(178);
```



2. Function which show package type's information

```
CREATE OR REPLACE function "t4"."packages"(packageType character varying(30))
RETURNS TABLE (package_id int, package_name character varying(30), package_type character varying(30),
no_of_days int ,package_amount int)
LANGUAGE 'plpgsql'
AS $BODY$
BEGIN
RETURN QUERY
( select tour_package.package_id, tour_package.package_name, tour_package.package_type,
tour_package.no_of_days, tour_package.package_amount
  from tour_package
  where tour_package.package_type = packageType );
END;
$BODY$;
select * from "t4"."packages"('Honeymoon');
```



Triggers function

1. Trigger function and trigger call for inserting new tuples in customer.

```
CREATE OR REPLACE FUNCTION "t4"."Customer_Update"()
RETURNS trigger
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
c_ID numeric;
BEGIN
SELECT customer.c_id INTO c_ID FROM customer
WHERE customer.c_id=NEW.c_id;
IF(c_ID=NEW.c_id) THEN
RAISE NOTICE 'C_ID already exists. Violates the primary key constraints.';
RETURN null;
ELSE
RAISE NOTICE 'Data inserted successfully.';
RETURN new;
END IF;
END
$BODY$


CREATE TRIGGER "Customer_Trigger"
BEFORE INSERT
ON "t4".customer
FOR EACH ROW
EXECUTE PROCEDURE "t4"."Customer_Update"();
```

SQL Query

Violation Case

TMS/postgres@PostgreSQL 13

Query Editor | Query History

```
CREATE OR REPLACE FUNCTION "t4"."Customer_Update"()
RETURNS trigger
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
c_ID numeric;
BEGIN
SELECT customer.c_id INTO c_ID FROM customer
WHERE customer.c_id=NEW.c_id;
IF(c_ID=NEW.c_id) THEN
RAISE NOTICE 'C_ID already exists. Violates the primary key constraints.';
RETURN null;
ELSE
RAISE NOTICE 'Data inserted successfully.';
RETURN new;
END IF;
END
$BODY$

CREATE TRIGGER "Customer_Trigger"
BEFORE INSERT
ON "t4".customer
FOR EACH ROW
EXECUTE PROCEDURE "t4"."Customer_Update"();\


INSERT INTO t4.customer(
    c_id, customername, c_password, gender, contact_no, address, pincode)
    VALUES (150, 'shreyas', 'shdvjdhc', 'Male', '54443776', 'hbvfjhvhfbv', '360001');
```

Data Output | Explain | Messages | Notifications

```
NOTICE:  C_ID already exists. Violates the primary key constraints.
INSERT 0 0

Query returned successfully in 225 msec.
```

Successful Case

TMS/postgres@PostgreSQL 13

Query Editor | Query History

```
CREATE OR REPLACE FUNCTION "t4"."Customer_Update"()
RETURNS trigger
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
c_ID numeric;
BEGIN
SELECT customer.c_id INTO c_ID FROM customer
WHERE customer.c_id=NEW.c_id;
IF(c_ID=NEW.c_id) THEN
RAISE NOTICE 'C_ID already exists. Violates the primary key constraints.';
RETURN null;
ELSE
RAISE NOTICE 'Data inserted successfully.';
RETURN new;
END IF;
END
$BODY$

CREATE TRIGGER "Customer_Trigger"
BEFORE INSERT
ON "t4".customer
FOR EACH ROW
EXECUTE PROCEDURE "t4"."Customer_Update"();\


INSERT INTO t4.customer(
    c_id, customername, c_password, gender, contact_no, address, pincode)
    VALUES (301, 'ronak', 'shdvjdhc', 'Male', '54443776', 'hbvfjhvhfbv', '360001');
```

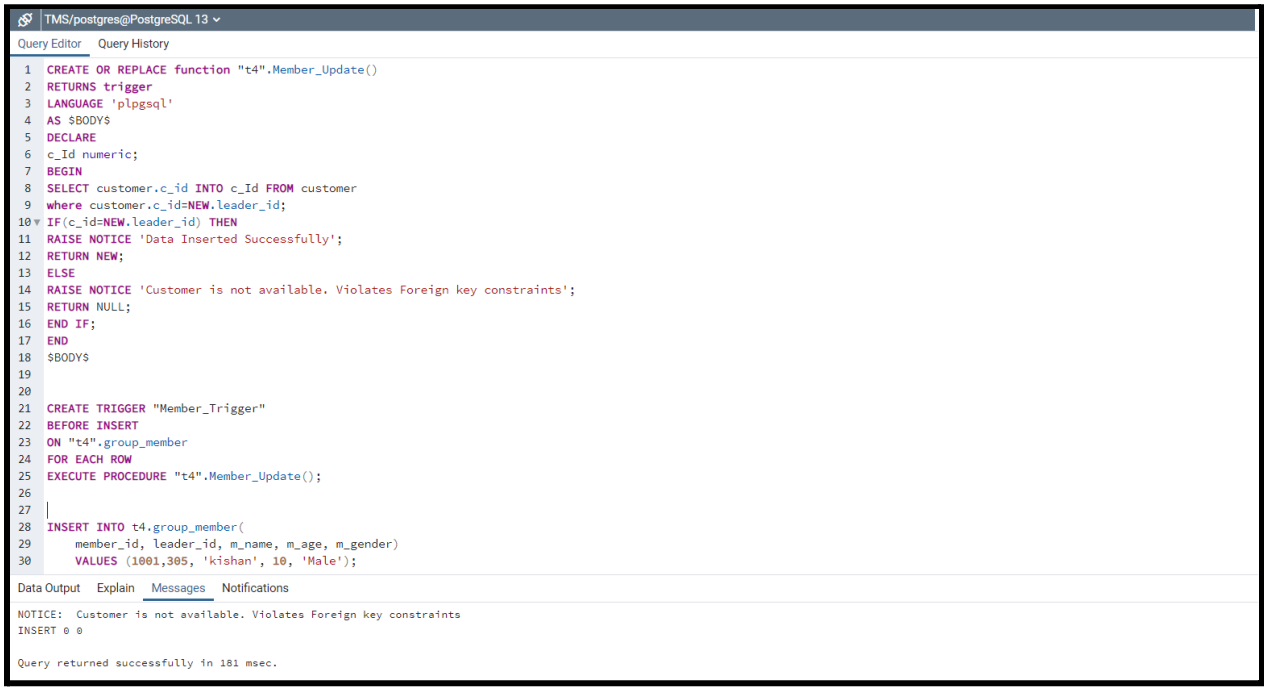
Data Output | Explain | Messages | Notifications

```
NOTICE:  Data inserted successfully.
INSERT 0 1

Query returned successfully in 75 msec.
```

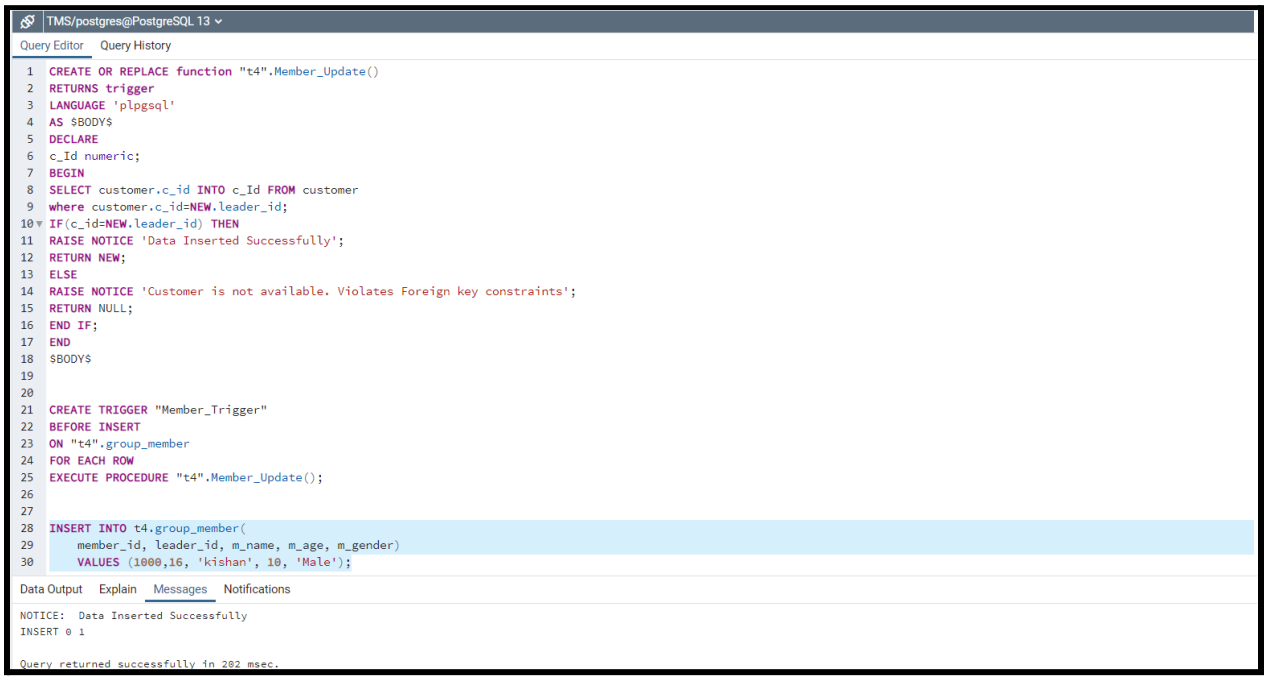
2. Trigger function for inserting new group members.

```
CREATE OR REPLACE function "t4".Member_Update()
RETURNS trigger
LANGUAGE 'plpgsql'
AS $BODY$
DECLARE
c_Id numeric;
BEGIN
SELECT customer.c_id INTO c_Id FROM customer
where customer.c_id=NEW.leader_id;
IF(c_id=NEW.leader_id) THEN
RAISE NOTICE 'Data Inserted Successfully';
RETURN NEW;
ELSE
RAISE NOTICE 'Customer is not available. Violates Foreign key constraints';
RETURN NULL;
END IF;
END
$BODY$


CREATE TRIGGER "Member_Trigger"
BEFORE INSERT
ON "t4".group_member
FOR EACH ROW
EXECUTE PROCEDURE "t4".Member_Update();
```

Unsuccessful Case (violation case)



Successful Case



Section7 : Project Code with output screenshots

- We used python Django to connect our Postgres database and build the following website.
- We have made a short video for working on our web page containing all functionalities and zip files of our codes file.

7.1 Video Link:

https://drive.google.com/file/d/1umoUuxgXvZT9GIai56ijEwbI0dz3VMYN/view?usp=sharing

7.2 Codes File Link:

https://drive.google.com/file/d/1bnb5TYmAv7yN1jLOAIE_-NGJ-13sMH6X/view?usp=sharing

Views.py

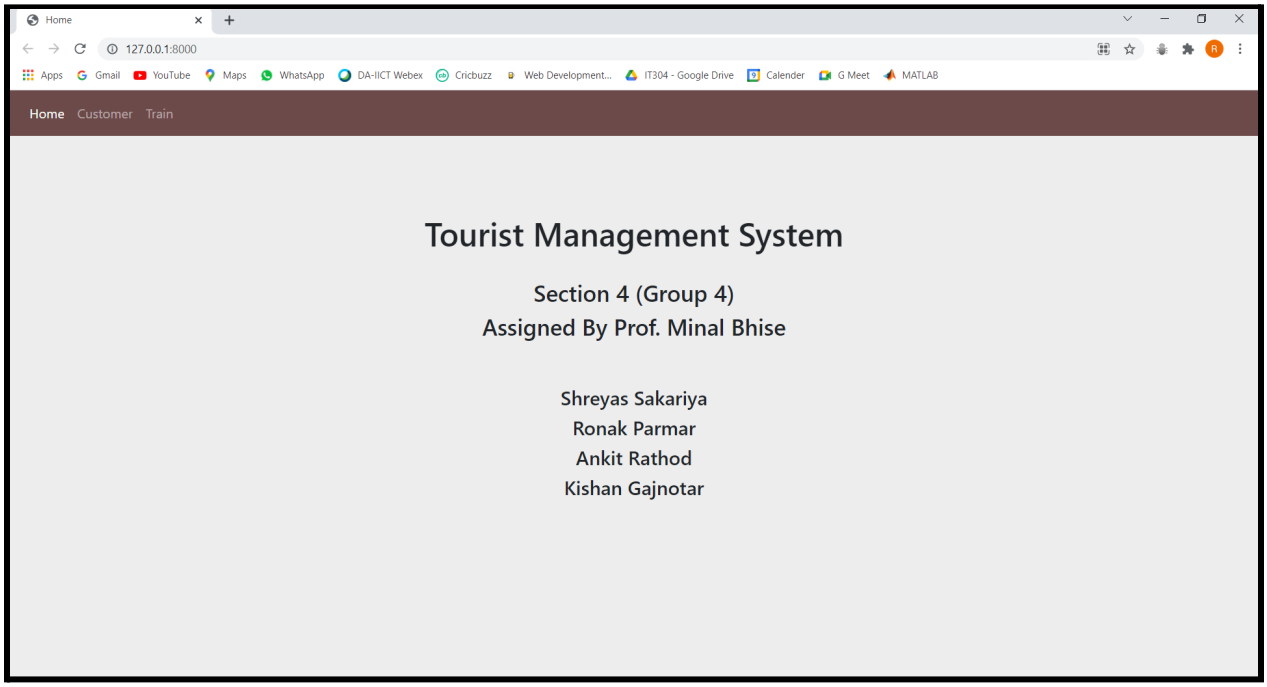
We use the below backhand handling code to connect python Django with our Postgres database. To see all other files go to the google drive link.

```
from django import http
from django.http.response import HttpResponse
from django.shortcuts import render
from TMS.models import EmpModel
from django.contrib import messages
from TMS.forms import Empforms

import psycopg2
connection =
psycopg2.connect(host="localhost",database="TMS",user="postgres",password="admin",options='-
c search_path=t4')
cursor = connection.cursor()

def home(request):
    return render(request,'index.html')

def showemp(request):
    showall=EmpModel.objects.all()
    return render(request,'c_index.html',{"data":showall})

def Insertemp(request):
    if request.method =="POST":
        if request.POST.get('c_id') and request.POST.get('customername') and
request.POST.get('c_password') and request.POST.get('gender') and
request.POST.get('contact_no') and request.POST.get('address') and
request.POST.get('pincode'):
            saverecord=EmpModel()
            saverecord.c_id=request.POST.get('c_id')
            saverecord.customername=request.POST.get('customername')
            saverecord.c_password=request.POST.get('c_password')
```

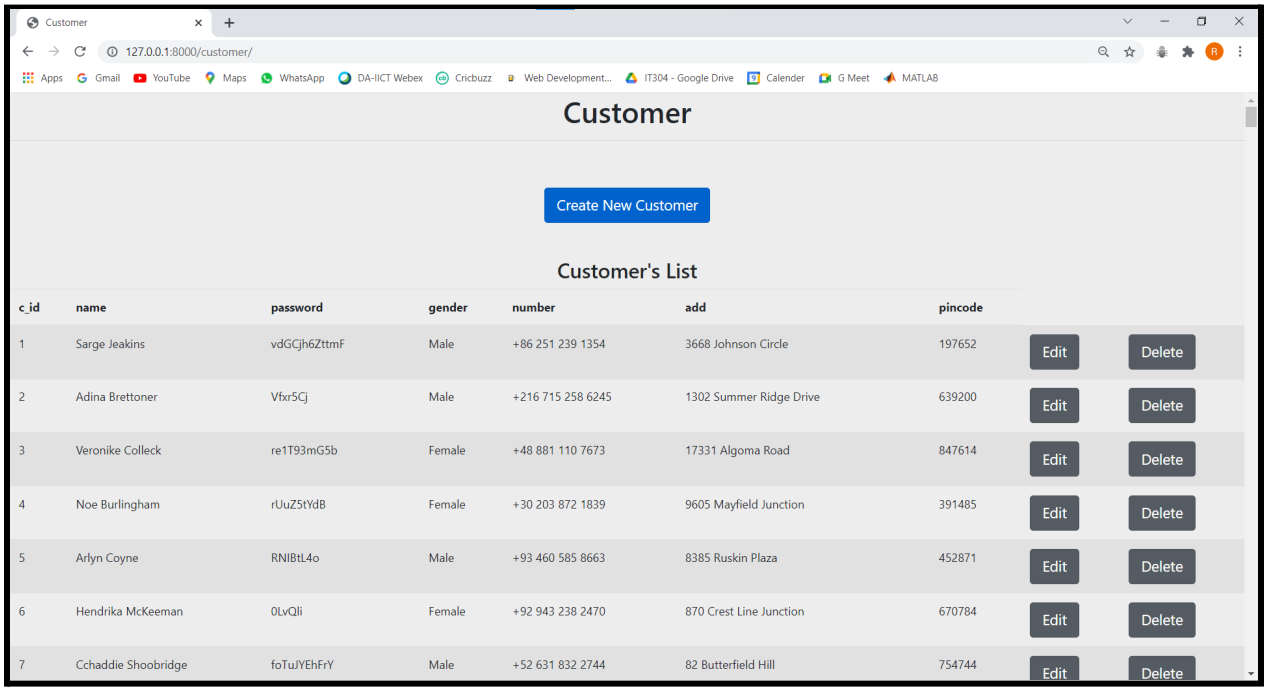
```
            saverecord.gender=request.POST.get('gender')
            saverecord.contact_no=request.POST.get('contact_no')
            saverecord.address=request.POST.get('address')
            saverecord.pincode=request.POST.get('pincode')
            saverecord.save()
            messages.success(request,'Customer ' +saverecord.customername+ ' is added
sucessfully.....')
        return render(request,'c_insert.html')
    else:
        return render(request,'c_insert.html')

def Editemp(request,c_id):
    Editempobj=EmpModel.objects.get(c_id=c_id)
    return render(request,'c_edit.html',{"EmpModel":Editempobj})

def updateemp(request,c_id):
    Updateemp=EmpModel.objects.get(c_id=c_id)
    form=Empforms(request.POST,instance=Updateemp)
    if form.is_valid():
        form.save()
        messages.success(request,'Record Update Successfully....!')
        return render(request,'c_edit.html',{"EmpModel":Updateemp})

def Delemp(request,c_id):
    delemployee=EmpModel.objects.get(c_id=c_id)
    delemployee.delete()
    showdata=EmpModel.objects.all()
    return render(request,'c_index.html',{"data": showdata})


def booking(request):
    if request.POST.get('find'):
        bording_point= "'%"+request.POST.get('bording_point')+"%'"
        destination_location= "'%"+request.POST.get('destination_location')+"%'"
        q=f"select * from train where (bording_point like {bording_point} and
destination_location like {destination_location})"
        cursor.execute(q)
        connection.commit()
        tuples = cursor.fetchall()
        return render(request,'t_index.html',{"data":tuples})
    else:
        q=f"select * from train "
        cursor.execute(q)
        connection.commit()
        tuples = cursor.fetchall()
        return render(request,'t_index.html',{"data":tuples})
```

7.3 Screenshot of output or web pages with data

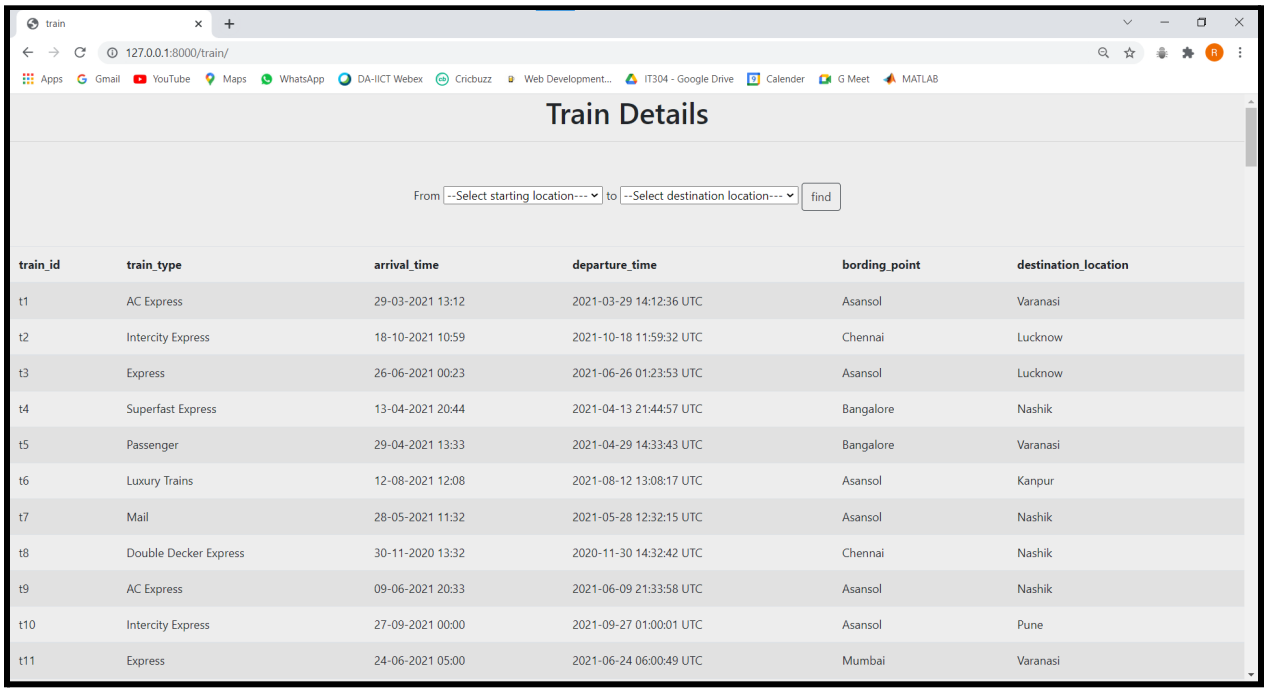
- **The Homepage of our website**



- **Customer interface on our website**



❖ **Train interface on our website**

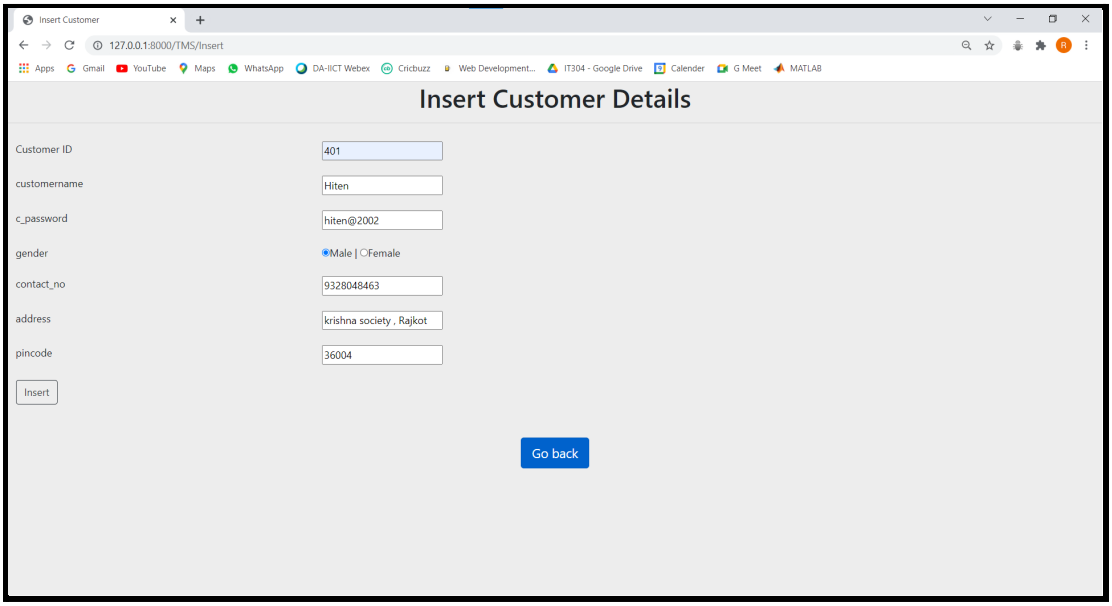


train_id	train_type	arrival_time	departure_time	bording_point	destination_location
t1	AC Express	29-03-2021 13:12	2021-03-29 14:12:36 UTC	Asansol	Varanasi
t2	Intercity Express	18-10-2021 10:59	2021-10-18 11:59:32 UTC	Chennai	Lucknow
t3	Express	26-06-2021 00:23	2021-06-26 01:23:53 UTC	Asansol	Lucknow
t4	Superfast Express	13-04-2021 20:44	2021-04-13 21:44:57 UTC	Bangalore	Nashik
t5	Passenger	29-04-2021 13:33	2021-04-29 14:33:43 UTC	Bangalore	Varanasi
t6	Luxury Trains	12-08-2021 12:08	2021-08-12 13:08:17 UTC	Asansol	Kanpur
t7	Mail	28-05-2021 11:32	2021-05-28 12:32:15 UTC	Asansol	Nashik
t8	Double Decker Express	30-11-2020 13:32	2020-11-30 14:32:42 UTC	Chennai	Nashik
t9	AC Express	09-06-2021 20:33	2021-06-09 21:33:58 UTC	Asansol	Nashik
t10	Intercity Express	27-09-2021 00:00	2021-09-27 01:00:01 UTC	Asansol	Pune
t11	Express	24-06-2021 05:00	2021-06-24 06:00:49 UTC	Mumbai	Varanasi

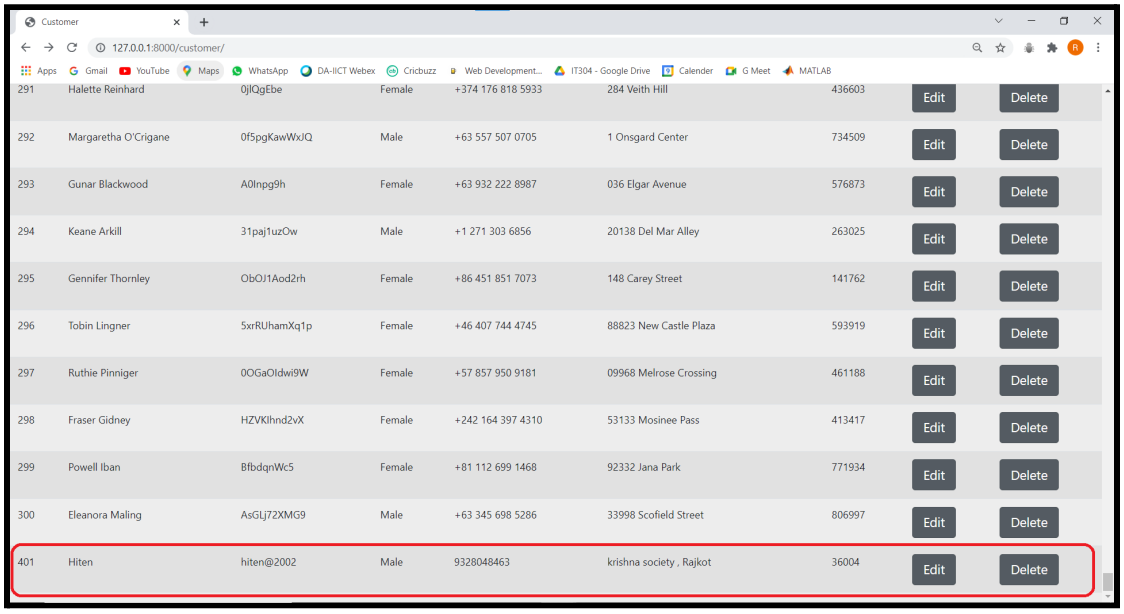
Functionality

1. Insert

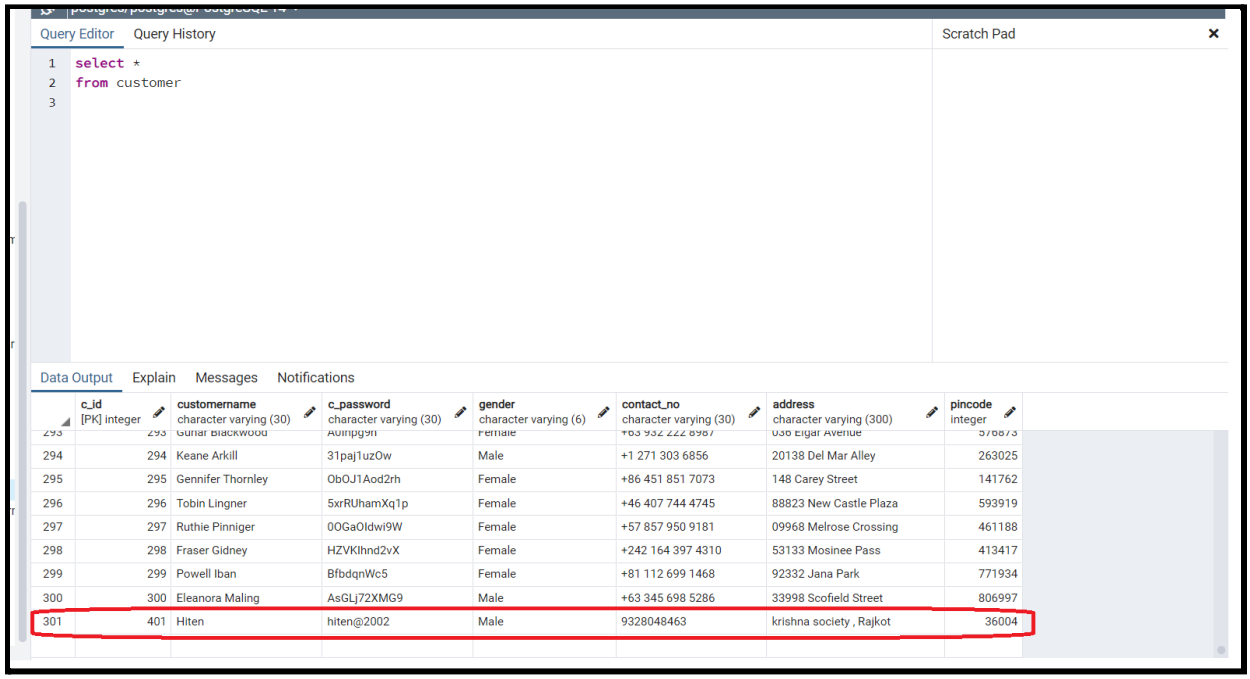
❖ Inserting Customer's detail



❖ Customer record after inserting

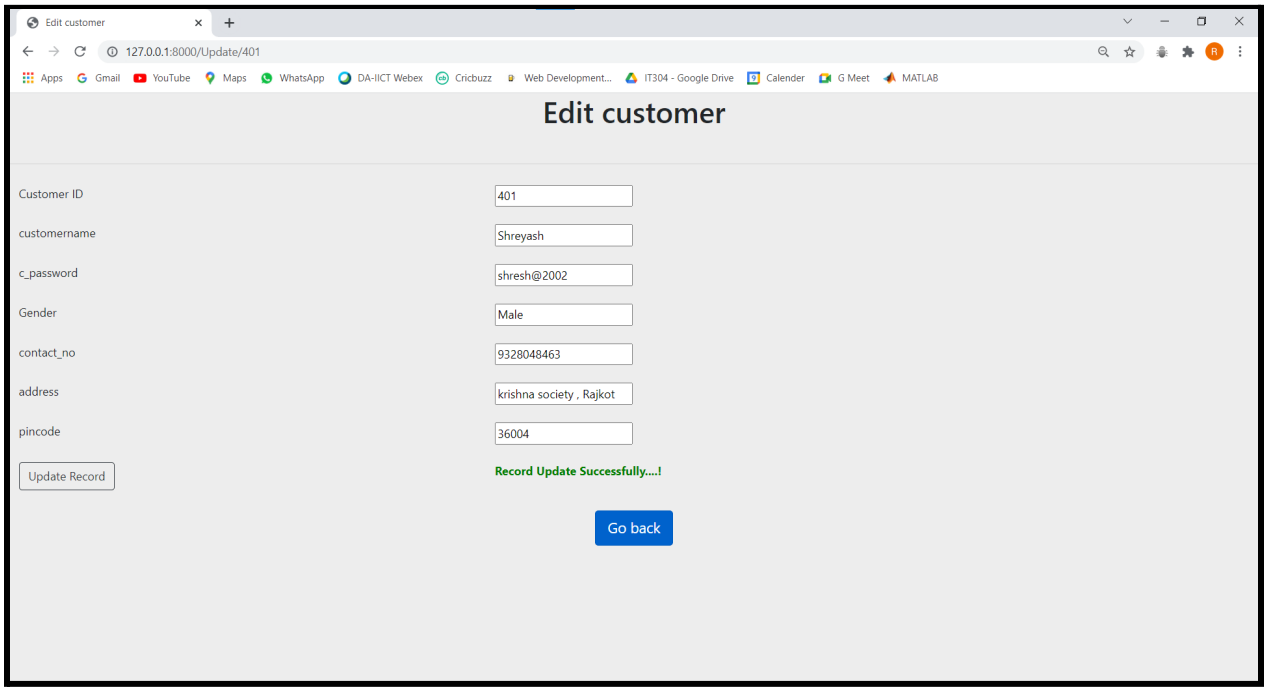


❖ Postgres updated data after inserting it on the website

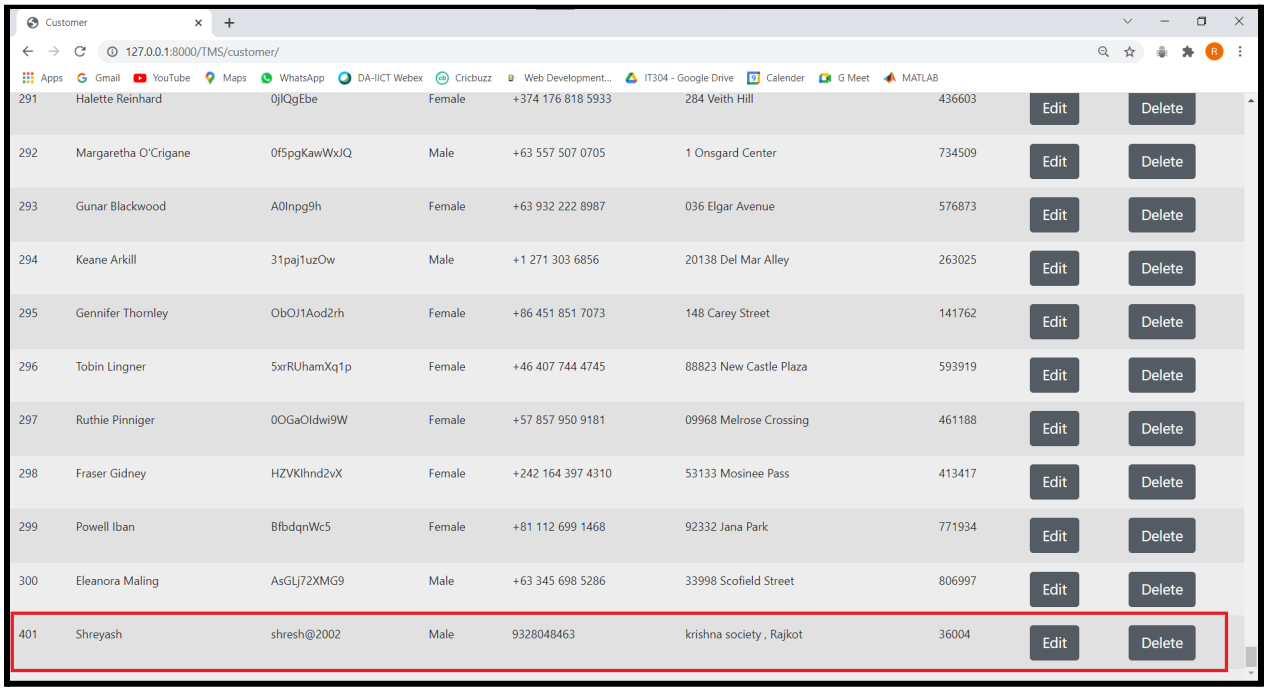


2. Update

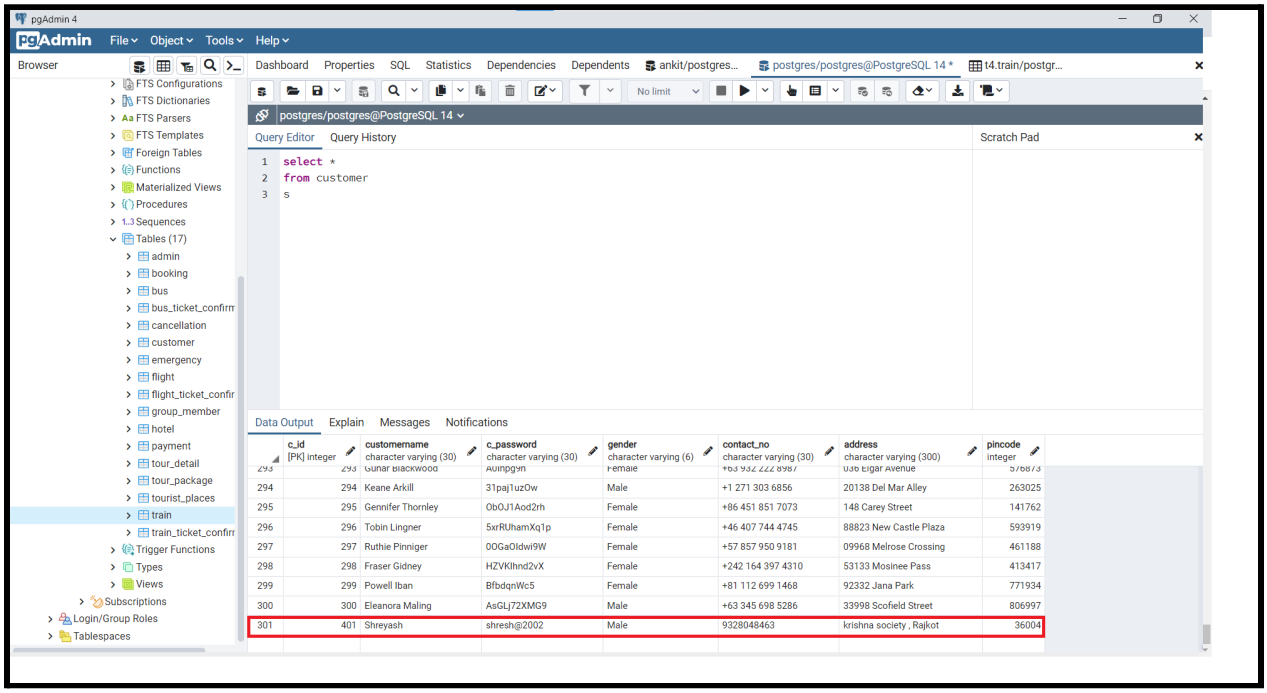
❖ Updating customer's details



❖ Updated Customer record after updating

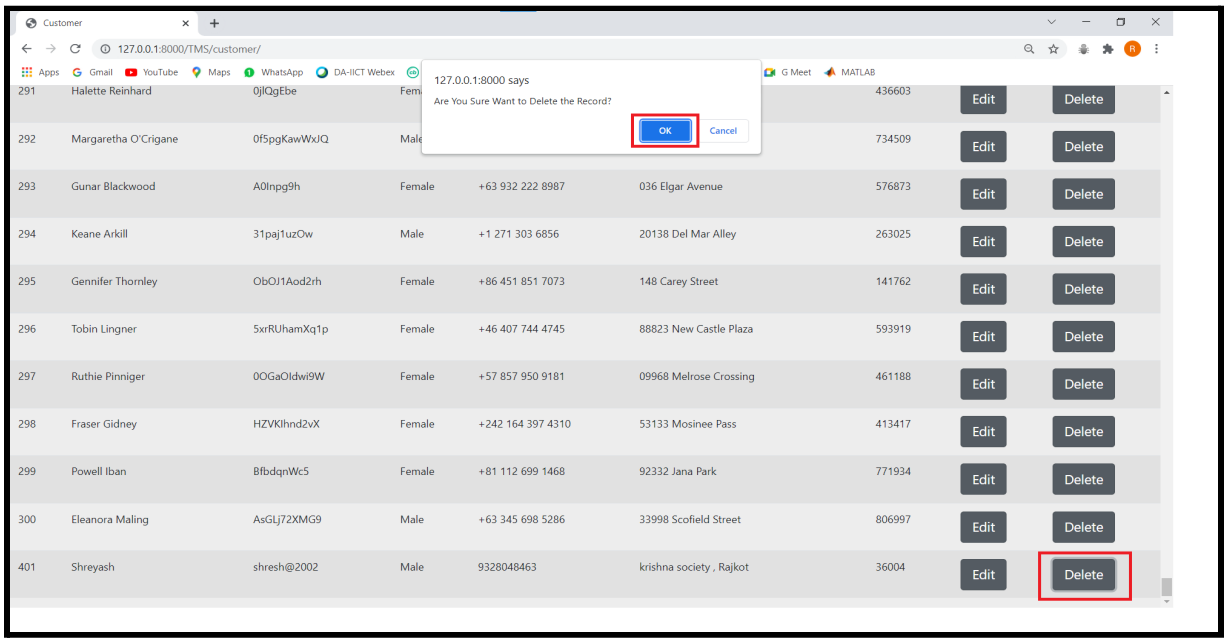


❖ **Postgres: Updated Customer record after updating**

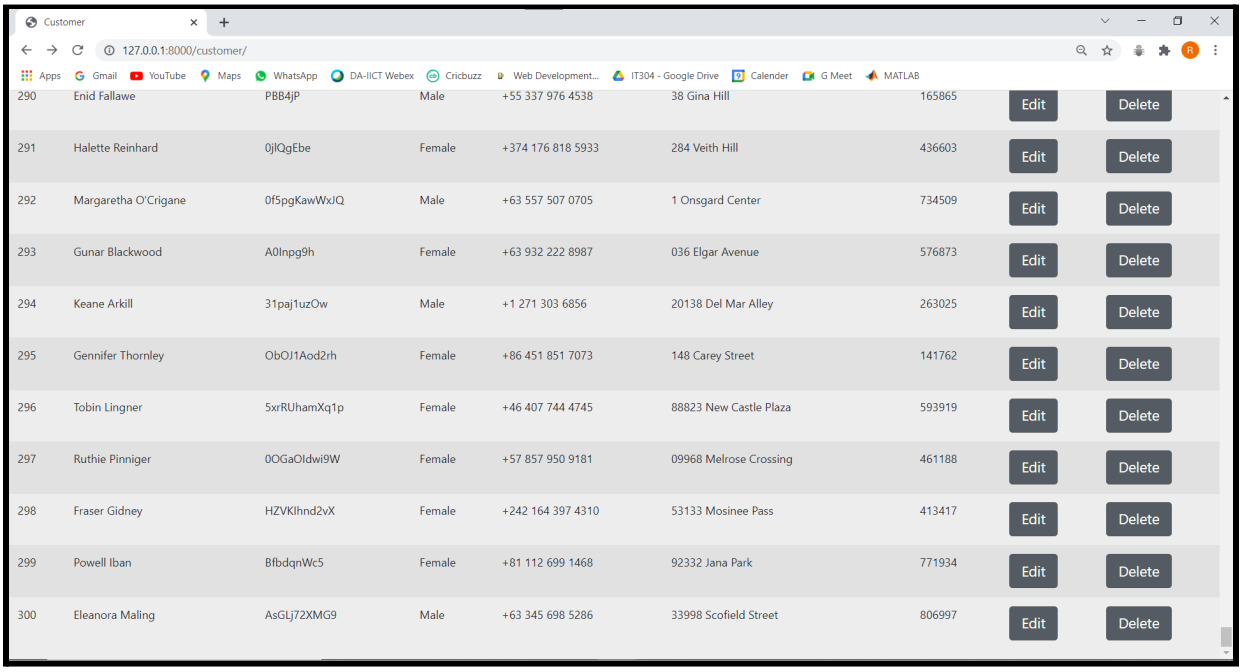


3. Delete

❖ Click on the delete button which customer detail you want to delete then press" OK"

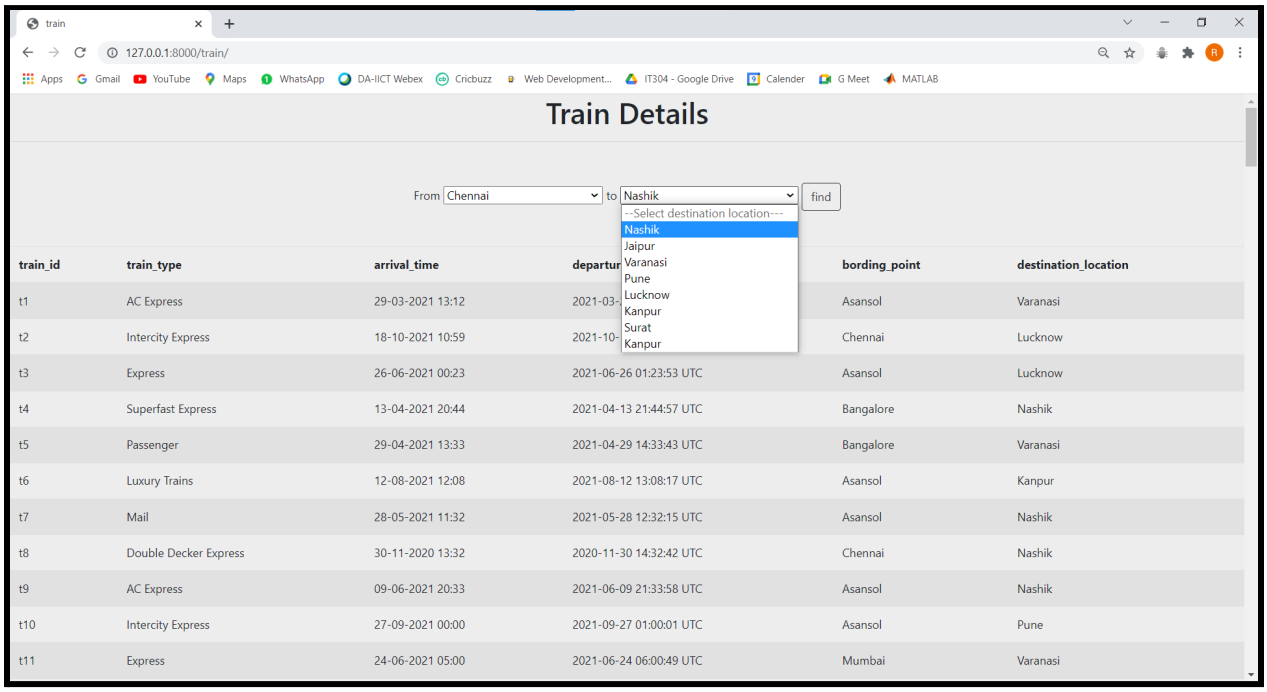


❖ Customer record after deleting



4. Search

❖ Here we use Train table for searching operation in which you select boarding point(starting location) and destination location it will give also train information from starting location to destination location



❖ Result of search query