Dhirubhai Ambani
Institute of Information and Communication Technology

Team: S3_T2

Project: Automobile Service Centre

Course: IT214 Database Management System

Instructor: Prof. Minal Bhise

Team Member: 201901177 – Manav Jain

201901207 – Bhagirath Talaviya

201901221 – Tarun Parashar

201901472 – Aditya Arya

Mentor TA: Raj Shah

Submission Date: 27-Nov-2021

Table of Content

Section1: Final SRS

1. Introduction

With the advancement of technology and the arrival of the internet age, the world is now running at people's fingertips. Gone are the days where you had to call the intended person for an appointment and the service manager would have to flip through their record books to confirm an appointment. All of these are now done directly on the internet. And the automobile sector is keeping up as well. From the sales department to local garages, database management systems have been a great substitute for the record books. Here we will be documenting the blueprint for an online portal and database for an automobile service center.

1.1 Current System in Use, its working and related issues

This is the way how most of the automobile service centers work right now. The flow is as below:

- Search for contact information on Google or contact someone who might give us the phone number of the service center.
- Give your car's registration number.
- The manager or agent will match the numbers in the record book and look for an empty slot and confirm appointment.
- Client will bring the car to the center and the agent will note down the issues and their queries.
- The agent will then pass the note to technician and he will check and repair and service the queries mentioned.
- The agent will do a check drive and once the service is done, he/she will call the client to pick up the car.
- The client will come, pay the bill and pick up the car.
- More books will be used to maintain the financial records.
- The client will also have the facility for feedback but there it has to be written in the feedback book.

Here, a lot of issues come to spotlight:

a) First of all, we observed that a lot record books are being used. Not only that you need a storage room to store all these record books in some order. In a case where the customer does not know what is wrong with the car, the record of services done in the car in the past might come very handy. But in the existing system you have to manually flip through the pages to find the history of the vehicle.
b) Moreover, there is no system for notifying customers for their time-to-time services. It has to be remembered by the customers.
c) One of the major concerns here is in the case of emergency, the customer has to go through a lot of steps as mentioned above. Given the magnitude of the situation, the customer might not want to go through all those steps and would definitely wish for a faster operation.
d) Not only this, there is no service for pickup of the customers too in these kind of crunch situations. And if in the unknown territory, it might be hard for the customer to describe current location. But with the help of our software the agent will directly be able to come to the pinpoint location with help of in-built GPS.
e) There can also be a case where it is not an emergency and the customer needs to service some parts or just fine-tune it. In this case, the customers will go to the service center and may find out that required service or parts or not available and this can be counted as a useless trip. Thus, in our system we will also add whether the required service or parts are available at the moment or not.
f) Now we will look from the admin side. Admin or manager generally keeps a record book for all the employee details. But one of the issues here is to bifurcate the customers' feedback in accordance with the responsible agent or technician which seems like a tedious job.
g) As mentioned in the point d) the manager can also keep a track of the parts or components available at the center. If not, more can be ordered.
h) Now looking at the financial management, a lot of books or slips are used to keep a record of the payments. It is very easy to misplace or forget a tiny piece of paper which might lead to some misunderstandings between the parties. Our system will allow the customers to look at their payment history and also show the current status of the current payment. But the customers will not be able to edit anything. Only managers will able to update or delete or edit the payment related queries.

1.2 Purpose

The purpose of the project here is to create a smooth operation interface for the automobile service sector by gathering all the customer and client data. The database will allow the customers to book the slot for servicing as well as the charges corresponding to the same. The customers will also be allowed to give their feedback for better communication between them and the company. Not only that, the customers will get regular notifications regarding service alerts. For the management side, they will get access to employee details, customer detail, daily business updates, and more importantly, they will get rid of the traditional manual appointment booking for customers.

1.3 Intended Audience and Reading Suggestions

This document is made for the end-users (i.e., automobiles service station owners, customers) to know what our system is meant for doing and how they can use it efficiently.

- This document describes to the customers the features that our system provides and how they can utilize them efficiently and manage their automobiles service. They can pre-register appointments for the service and thus avoid the hassle of queuing in a row at the service center to get their vehicle up to date. On successful booking, customers get an appointment at a convenient time slot.
- The employee's database is also maintained so that the employees can keep track of their payments or work-related information such as how many appointments are registered for a day and how much work is pending yet. The manager can get all the data related to all appointments booked.
- The manager can access the employee database and make changes to it, according to the requirements.
- Apart from these managers and employees can keep track of how many payments from customers are pending and the payment is done in advance. The system does send the notification to the customer for which the next service date has arrived.

Thus, this simple structure keeps all the information well organized.

1.4 Product Scope

The automobile industry works in three phases: Manufacturing, Sales and Services. Once the first two phases are done, the service centers come into the play for the third phase. The brands which are well known all over the world have one thing in common- Efficient customer service. Customer service is considered to be the major factor in the minds of customers before choosing a product. Better customer service means customers' loyalty which will ensure more and profitable business in the future. The project is automobile service databases that can be used by big automobile companies to provide better customer service to their clients and maintain their records in a simple way.

Automobile Service Centre Management system is composed of two main components: a client- side application and a company-side application.

- Client-side application - side we can see the charges and slot availability for various jobs.
- Company side Application - employee details, as well as a feedback system, can be maintained. Also consider regular service alerts, emergency breakdown pickup, and drop facilities.

The model is simple to use and provides a wide range of functionalities for the best client- company partnership. Not only this, we can keep track of the previous services done for the respective automobiles. This will help the service provider to assess the current condition and problems of the automobile more precisely.

1.5 Description

This project is about designing an automobile service system application. This system is made to facilitate the user experience in getting automobile service. The system maintains a database for keeping customer details and bookings made by customers.

- **Registration of Customer and Service Station:**
  Any vehicle owner or service station need to register themselves and verify details in order to be able to access the platform. Verification part is important in order to prevent fraud. After successful registration vehicle owner can add, update and delete their vehicle, can book appointment and use many other features of the software. Service station will receive new customers on this platform and can manage customer, employee and payments easily by this platform.

- **Vehicle Management:**
  - **Adding new vehicle:**
    After completing sign up part, customer will be able to add their new vehicles. The system will maintain its details. Customer need to provide details about the vehicle such as VIN Number, registered number plate, etc. In case of offline customer Manager or employee can also add new vehicle in their Database.

  - **Update detail of vehicle:**
    In any case if customer or manager wants to update the vehicle detail, they should be able to do that. This is because the entries will be done by humans and it is very likely people makes mistakes while entering details.

  - **List a vehicle of particular owner:**
    This function is important in order to access vehicle by owner's name. Access here means when people want to book service offline, they generally don't remember VIN number. To facilitate such case, we will show all the vehicle entries of particular customer by their Phone number.

  - **Show history of vehicle:**
    This function will show all the previous services done for the particular vehicle. This will be helpful for employee to know about vehicle and its problems in case owner don't know.

- **Appointment Management:**
  - **Book new appointment:**
    This is the primary purpose of whole system to provide easy appointment booking system to customer. Customers will be able to book new appointment for their vehicle and manager or employee can also book an appointment in case of offline entry at service station.

  - **Cancel or reschedule Appointment:**
    Again, this is for the facility of customer in any case if it is not possible for customer to arrive at service station on the booked time slot. The customer must be able to cancel or reschedule it.

  - **List booked slot:**
    The employee and manger will be able to see booked slot. This will help them to manage their time accordingly and to know if they can take any customer who did not register any appointment.

  - **List available slot:**
    Again, this is helpful for service station employee and manager to manage offline customer load. Here they will be able to see date wise list of slots which aren't yet booked.
  - **List pending service:**
    This is for employee and manager to know about current status of booked appointments. It will allow them to know workload.

- **List available services for particular Service station:**
  This is for customer to know detail about service station like what particular service is available at any service station. List will also show prices and time required to complete the particular service. So, they will manage their time and choose service station accordingly.

- **List all station providing given service:**
  This is to provide best choice to customer for particular service. The customer will be able to see different station which provide the same service and can compare the price and time to choose best suitable to them.

- **Employee Management:**
  - **List of employees and its detail:**
    This will provide a list of employees for a particular station with all the details. The manager will be able to see all the detail such as name, rating, salary, etc. This will help him to analyze the work done by the employees. Manager can also update employee detail such as salary if the particular employee gets increment.

  - **List of employees without confidential details.**
    This list will show detail of the employee which do not involve any confidential detail such as salary. This is to show the customers which employee is going to serve them and can see their rating to see how good the particular assigned employee is.

- **Payment Management:**
  - **Reliable payment methods:**
    The customer will have different choice to select method of payment. We need to provide them reliable payment method. The data security in case of online payment is very primary requirement.

  - **Managing daily payment reports:**
    This is for the manager to maintain daily account detail of the service station. This function will show the list of payment corresponding to different service, sending payment detail of the day.

  - **Manage Bills:**
    This is also for manager's use. This will help him to maintain the record of the income and managing bills. In case of implementing database on mobile application we can develop system to send the bills to customer on their email.

- **Inventory Management**
  - **Manage Spare parts:**
    This will help manager know how much stock of the spare parts is currently available in the garage. The parts used for a particular service will be automatically updated in the inventory.

- **Feedback System:**
  - **Customer feedback collection for each service:**
    Feedback for each service will be collected from the customer in order to know quality of the service provided by a particular assigned employee. This feedback will also be reflected on employee's profile.
  - **Update rating of Employee assigned according to the feedback:**
    Based on the customer's feedback the employees rating will be updated. Rating System will be based on the star rating in the range of 1-5, higher the rating of the employee indicates good quality of service.

- **Notification System:**
  - **Sending Notification for Service Alert:**
    System will send a periodic alert notification to the customer regarding regular Service. The notification will be sent to customer based on last date of service. This will help them for maintenance of the vehicle and the service stations will get improvement in business.

- **Emergency Service:**
  - **Contact detail will be provided based on nearest location:**
    According to the survey most of the people do not have any contact detail in case of emergency. The software will provide them the contact detail of the service station in particular location which provide emergency services.

2. Document the Requirements Collection/ Fact- Finding Phase

2.1 Background Reading

The automobile industry is one of the well-developed sectors in India. We have gone through many online registration / Appointments booking Portals. There are many independent business websites which provide registrations for all the company and local registered garage, while most of the well-established companies provide their platform to register appointment for their showroom.

- **GoMechanic:**
  - Select location (will choose automatically if access granted by user)
  - Customer will provide vehicle model detail and contact detail.
  - Based on vehicle detail list of service will be shown with prices in different section.
  - 11 different Section or types of service are provided. Periodic Service, Denting & Painting, Batteries, Car Spa & Cleaning, AC Service & Repair, Tires & Wheel Care, Insurance Claim, Detailing Services, Custom Services, Windshields &Lights, Clutch & Fitments.
  - Select Date and Time from available slots.
  - Add/ Select Address.
  - Confirm appointment by completing Payment.
- **KIA-Service Booking:**
  - Customer need to select showroom/dealer by state/city filter.
  - Provide and verify customer detail (Name, Mobile number, email).
  - Provide Vehicle Detail (VIN number, Odometer reading).
  - Select service type (Free Service / Paid Service / Running Repair).
  - Select Method (Pickup Service / Pick & Drop Service / Self Drive In).
  - Select Time Slot from available slots.
  - Submit detail.

References:

- GoMechanic →gomechanic.in
- KIA →ServiceBookingSection
- Droom →droom.in
- Bosch Service→boschcarservice.com
- Maruti Suzuki→ServiceSection
- Mahindra →mahindrafirstchoiceservices.com

Requirements Gathered by Background Reading:

- Customer detail and verified contact details are required
- Vehicle VIN number, Registered Vehicle Number, Odometer reading is required.
- List of service sorted by different category is need to be provide to customer.

- Customer want to see date and time slots available.
- Different payment method should be available to customer.

2.2 Interviews

We have done the role play interview among ourself in a group. One for each Manager and another for customer. Here is plan and summary for both.

2.2.1Customer

Automobile Service: Interview Plan

System: Automobile Service Center

Interviewee: 1) Tarun Parashar (Role Play) **Designation:** Customer

2) Manav Jain (Role Play) **Designation:** Customer

Interviewer: 1) Bhagirath Talaviya **Designation:** Student

2) Aditya Arya **Designation:** Student

Date: 2/10/2021 **Time:** 21:00

Duration: 30 minutes **Place:** Video Conference

Purpose of Interview:

Preliminary meeting to identify problems and requirements regarding customer requirement and facility at Automobile Service Station.

Agenda:

- Automobile Service
- Expectation from online booking Added Features
- Emergency Related to Vehicle

Documents to be brought to the interview: No documents required

Interview Questions:

1) Do you own a car or any type of vehicle? If yes, how big was the customer service a factor in yourself purchasing the particular car?

Ans: Yes. I have Maruti Wagon-R. Apart from knowing for their long-time durability, availability of Maruti service centers all over India was definitely one of the most important factors while buying it.

2) How long it has been since you bought the car? What is the frequency of your car services?

Ans: It's been 8 years now. Earlier it you used to be just the three to four free service provided by the company for 1 year. Not anything major. But after 3-4 years changing of battery, oiling the mechanical parts, repairing of jumpers and brakes, replacing worn out parts and rubbers increased.

3) Have you ever serviced your car at a local garage? What is the difference between any local car garage and a well-known company garage?

Ans: Yes, a couple of times. In local garages you get to know that your car is given more care and attention whereas in my opinion that is missing in well-known car service centre. However, they are able to detect and solve problems better and faster. So local garage should step up and know how to do things more efficiently in terms of quality and time management whereas in bigger garages, it would be better if you get to know your technician and his/her technical abilities better.

4) How satisfied are you with your current customer service? What are the facilities they provide you as a customer?

Ans: I am happy with the customer service they provide. Every time I give the car for service, they do a complete check of the car, do the necessary and call me once it is done. Earlier it used to be within a day, but now it takes more than couple of days.

5) What are the changes or updates you would like to see in their existing way of providing services? Any new facilities you would like them to add?

Ans: They provided 4 free services in the first year. I forgot half of them. So would be nice for new users if they sent the alerts for those services. Not only that it is recommended to have a checkup of the car from time to time in order to run the car smoothly for a longer interval. So, they should still send the alerts to the old customer as well. Also, there should be a facility for locating service centers in new area or have someone pick up and drop the car. This is something really bothering if you are stuck somewhere at work or if you are busy.

6) How would you like to get all these things done on internet or at one stop portal? What are the things you would like to see there?

Ans: Yeah, it would be great. Would love to do the booking the appointment by sitting on the couch. Should be able to postpone or prepone the appointment in case of some other work coming up. Also, there should be a list of services provided at the selected service centre. I would also like to know how many cars are already there at the centre for service so I can know how much time it is roughly going to take.

7) And what about payment methods?

Ans: Online payments are definitely easy and hassle free. But one of the major issues with online payment is the security factors. We hear a lot of cases of online fraud. Not only the payment but the app or portal should also be authentic.

Mock Interview Summary:

- Ambiguity regarding traffic at service center
- No knowledge about some station, do they provide particular service or not.
- Booking should be hassle free, should not ask too many questions.
- Must be able to cancel or shift the appointment.
- Customers want to have reliable payment methods.
- The service station on the App must be authentic, no fraud station should be there.
- People do forget about dates for the service.
- Feedback system is good, but it should not bother people.

2.2.2 Manager

Automobile Service System: Interview Plan

System: Automobile Service Centre

Interviewee: 1) Bhagirath Talaviya **Designation:** Manager

2) Aditya Arya **Designation:** Manager

Interviewer: 1) Tarun Parashar **Designation:** Student

2) Manav Jain **Designation:** Student

Date: 05/10/2021 **Time:** 21:00

Duration: 30 minutes **Place:** Online Video Conference

Purpose of Interview:

Preliminary meeting to identify problems and requirements regarding management at the service centre.

Agenda:

- Employee record management of service system.
- Ideas about workflow management.
- And corresponding actions to be taken.

Documents to be brought to the interview: No documents required

Interview Questions:

1) How do you work the entire service centre? What is your role here?

Ans: As you know, I am the manager. I look over everything that is happening at the centre. From keeping in track of how many vehicles are serviced in a time interval, their payments, and parts to order for the inventory. Not only this, I also manage my technicians and agents, their work hours, salaries, helping them improve. I also keep track of the customers' feedback and work on them with my co-workers.

2) How do you keep track of records?

Ans: As of now, everything is done manually. Every single entry of vehicle, payment, feedback, employee details are done manually and written in the record books.

3) If we were to make an online portal, what are things you would like to see there?

Ans: That would be great. It would save us from hours of paperwork and save lots of storage space. I would like to be able to add or check my employee details, their working hours, and their feedback and manage salary accordingly. If we can see the history of the car instantly, it would save us a lot of time checking unnecessary things in case customer does not know what the issues are. But as a manager, we have to keep security over our record books. So if you were to make an online portal, you should be able to provide the guarantee against

hackers and I should be in control of how much different users can access different parts of your portal or database.

4) Lastly, what are your views if we add an online payment method?

Ans: It would be a great help for customers and would also help us keep track of payment easily. But online payments are tricky and we care about customers so there should be a sense of security. We will not implement it as long as you ensure us that there would be no fraud because that ruins our reputation and customers' money as well.

Mock Interview Summary:

- The requirements of the automobile service centre management system.
- To know how to manage the payment
- How accessibility rights should be given to the different classes of users.
- To get the complete information about all appointments, all services done in the past.
- How to manage payments to all employees of the service centre.
- Add or remove a new employee record to the database.

2.2.3 Requirement Gathered by Interview:

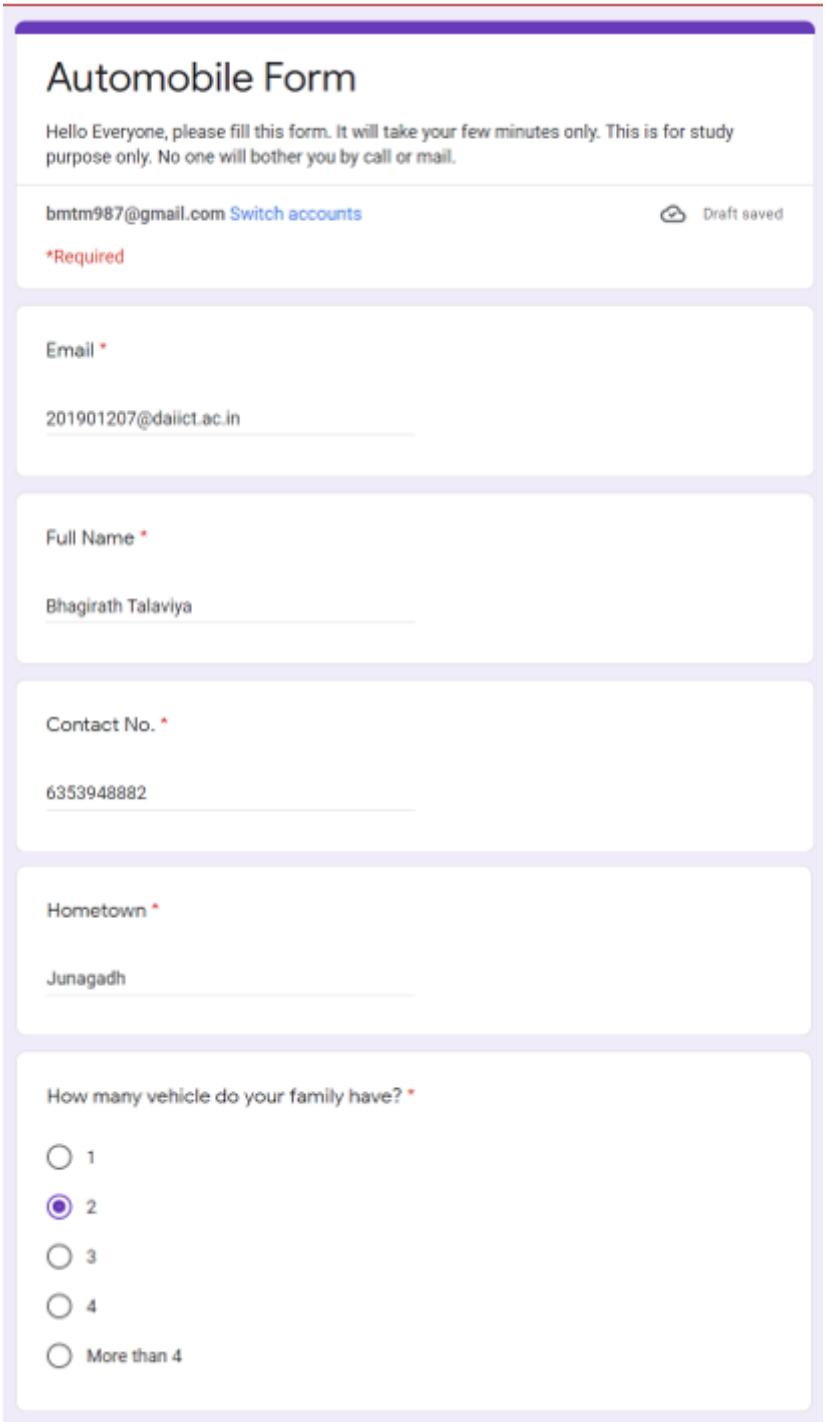
- Reliable payment method for customer.
- Notification for regular service reminder for customer.
- Feedback system for customer to give review and for manager to track employee's performance
- List of slots available for particular station with time is necessary.
- Payment related data should be very clear and no malfunctioning is acceptable in that case.
- Manager and employee should be able to keep track of appointments (i.e., how many is booked, how many service has been completed and all other detail of the same kind)

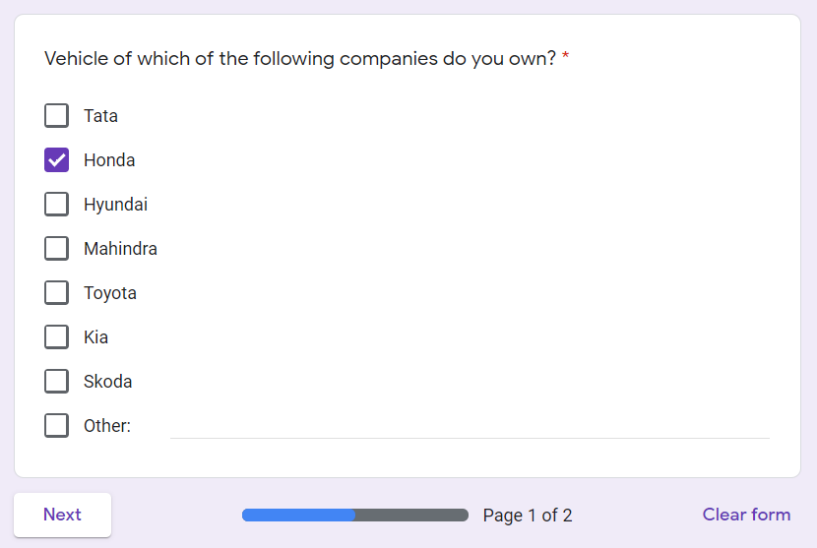
2.3 Questionnaire

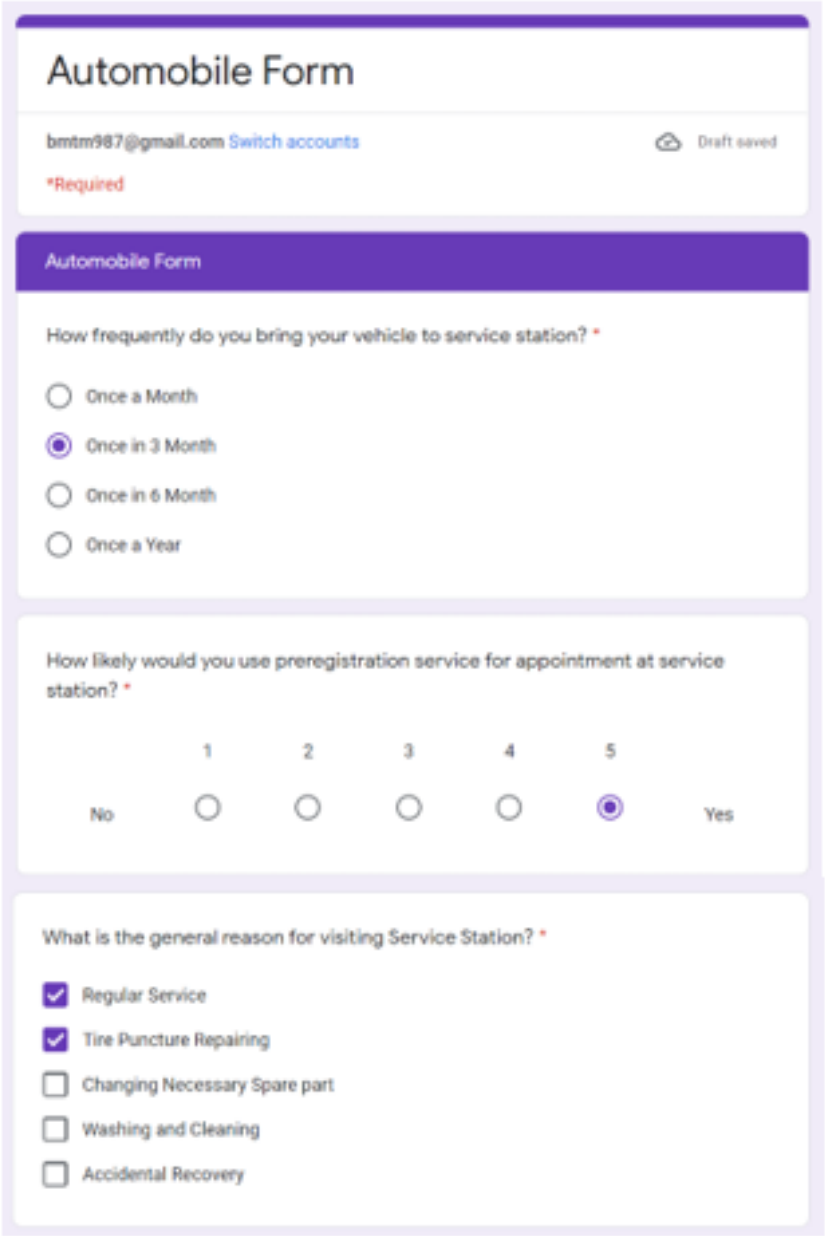
2.3.1 Google Form

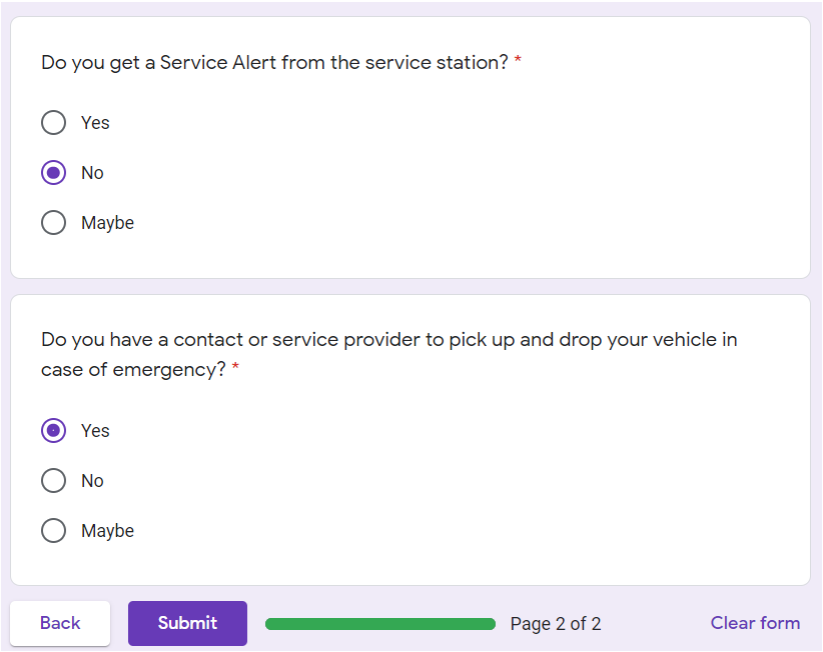
We have created a google form with 11 different questions to know about what does most of the people think. We will come to know by analyzing statistics we get from response. We circulate the google form link among the students of our college and many other colleges. This is the link for the Response Sheet.

Screenshot of the Google form circulated among the student:

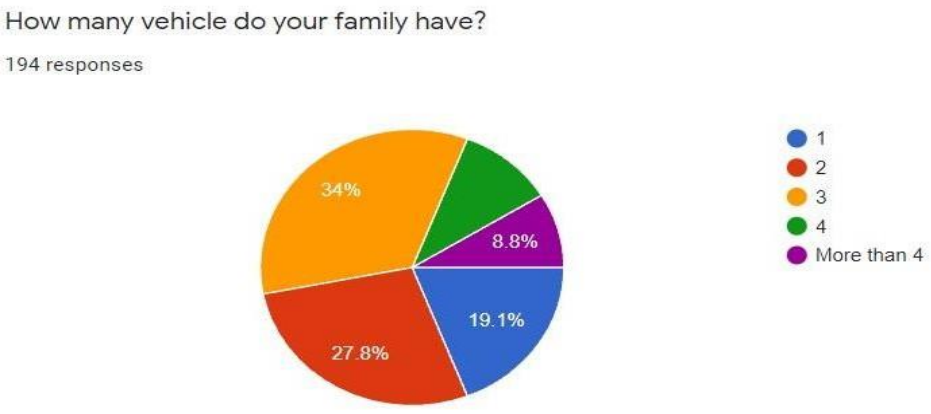


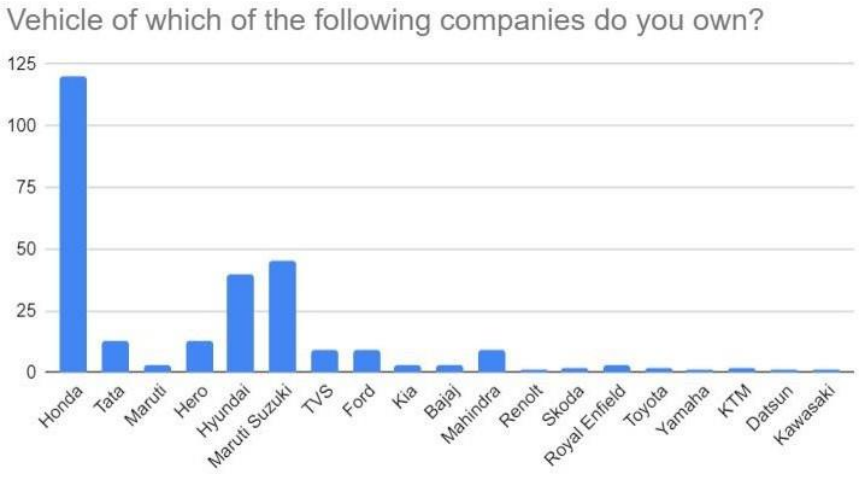

Vehicle of which of the following companies do you own? *
Tata
Honda
Hyundai
Mahindra
Toyota
Kia
Skoda
Other:
Next
Page 1 of 2
Clear form


Automobile Form
bmtm987@gmail.com Switch accounts
Draft saved
*Required
Automobile Form
How frequently do you bring your vehicle to service station? *
Once a Month
Once in 3 Month
Once in 6 Month
Once a Year
How likely would you use preregistration service for appointment at service station? *
1
2
3
4
5
No
Yes
What is the general reason for visiting Service Station? *
Regular Service
Tire Puncture Repairing
Changing Necessary Spare part
Washing and Cleaning
Accidental Recovery


Do you get a Service Alert from the service station? *
Yes
No
Maybe
Do you have a contact or service provider to pick up and drop your vehicle in case of emergency? *
Yes
No
Maybe
Back
Submit
Page 2 of 2
Clear form

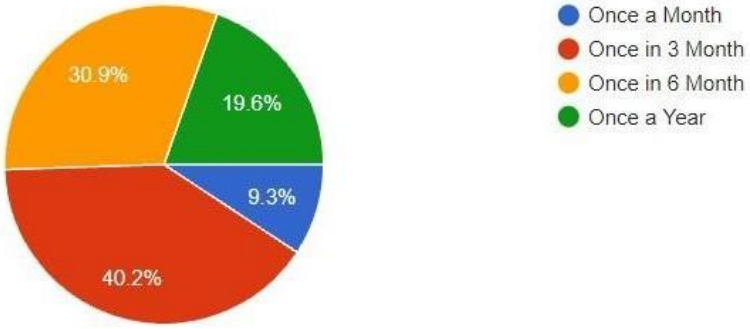
2.3.2 Summery:


How many vehicle do your family have?
194 responses
34%
8.8%
19.1%
27.8%
1
2
3
4
More than 4


Vehicle of which of the following companies do you own?
125
100
75
50
25
0
Honda
Tata
Maruti
Hero
Hyundai
Maruti Suzuki
TVS
Ford
Kia
Bajaj
Mahindra
Renolt
Skoda
Royal Enfield
Toyota
Yamaha
KTM
Datsun
Kawasaki

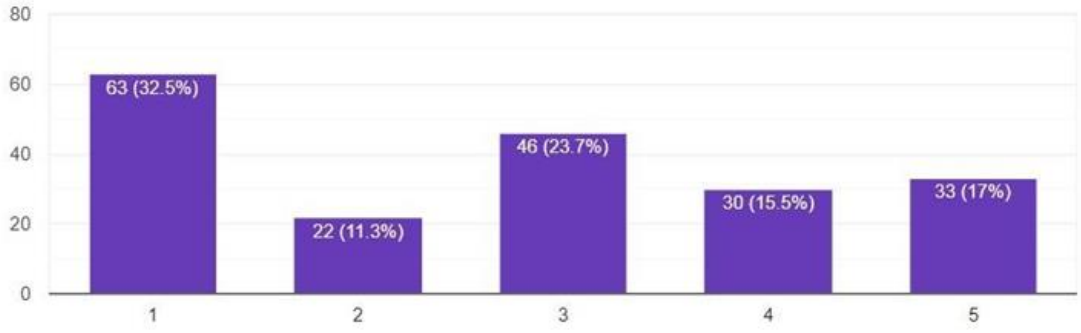
How frequently do you bring your vehicle to service station?

194 responses



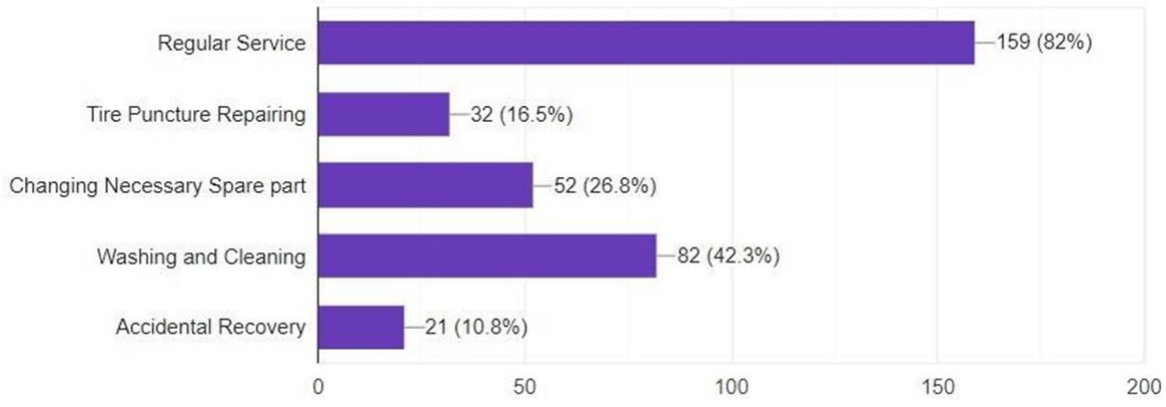
How likely would you use preregistration service for appointment at service station?

194 responses



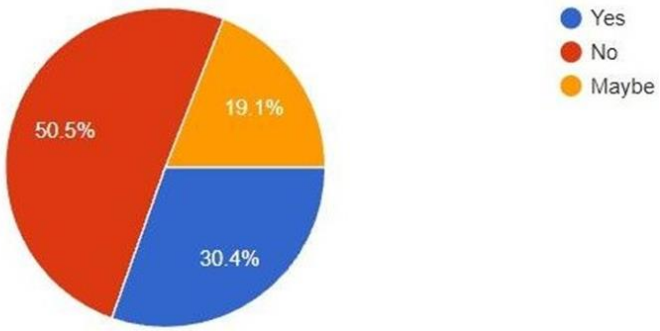
What is the general reason for visiting Service Station?

194 responses



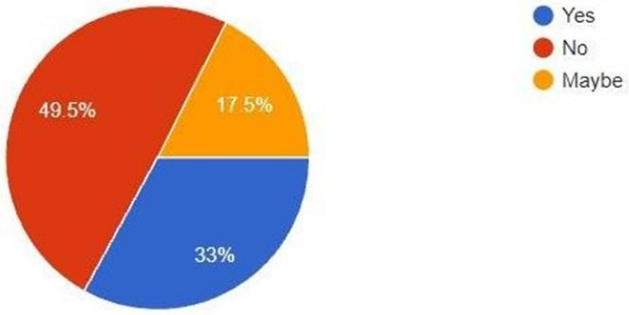
Do you get a Service Alert from the service station?

194 responses



Do you have a contact or service provider to pick up and drop your vehicle in case of emergency?

194 responses



2.3.3 Requirement Gathered by Questionnaire

- As observed in pie chart in summary 50.5% people do not get any notification or alert for regular service. So, this feature is required.
- It is also observable that 49.5 % people do not have any contact in case of vehicle emergency (i.e., On way tire puncture, accidents, etc.). So, we should provide contact number of each service station which provide the emergency pickup service.
- We got to know different company's name whose service station should be included as service provider to attract more user.
- Most people have multiple vehicles at home. So, customer must be allowed to book multiple slots at a time for different vehicle.

2.4 Observation

Automobile Service System: Observations

System: Automobile Service Centre

Observations by: Manav Jain (Student)

Date: 7/10/2021 **Time:** 21:00

Duration: 30 minutes **Place:** Site

Observations:

- Customer should be allowed to reschedule or cancel its appointment.
- Data access for particular class is very important aspect of the software from data security point of view.
- Easy and clear directions is to be there for customer to book appointments.
- Proper data integrity should be there in the database. There should not be any mismatched in appointments detail on customer and service station side. It is also necessary for payment management.
- Customer should be able to see Employee's open detail such as Name and rating. But, not salary.
- Not clear exactly how staff uses car park (Ask staff to fill in the questionnaire on car park usage)

3. Fact Finding Chart

Objective	**Technique**	**Subject(s)**	**Time Commitment**
To get an idea about the working flow of different companies that manages the service centre	Background Reading	Company Reports and Web articles	0.25 day
To get an insight about the working of a local car Service Centre and ideas regarding designing our system	Interview	Manager and technician	45minutes
To get an idea regarding the client's requirement sand expectations to develop our database management system	Interview	1Client	30minutes
To get feedback about the current developed management system and make changes as per requirements	Interview	1Client	30minutes
To get client's opinion regarding their requirements from the database management system	Survey/Questionnaire	Clients/ Audience	1 day
To follow up development of	Observation	1Creativestaff	0.5day

4. Requirements List:

We find the requirements for the automobile service center software by using different method such as reading, interview, questionnaire, and observation. This is the final list of requirements gathered using all this.

- List of service sorted by different category is need to be provide to customer before registering appointment.
- List of slots available for particular station on particular date with time is to be provided to customer.
- Customer should be allowed to reschedule or cancel its appointment.
- Customer should be able to see non confidential detail of employee(agent) assigned to him.
- Most people have multiple vehicles at home. So, customer must be allowed to book multiple slots at a time for different vehicle.
- Reliable Payment system with more than one method of payment is required. Payment related data should be well maintained and no malfunctioning is acceptable in that case.
- Customer detail and verified contact details of customer are required to book slots
- Vehicle type, vehicle's VIN number, Registered Vehicle Number, Odometer reading is necessary to book an appointment.
- Notification for regular service reminder for customer.
- Manager and employee should be able to keep track of appointments (i.e., how many is booked, how many service has been completed and all other detail of the same kind)
- Feedback system for customer to give review and for manager to track employee's performance.
- Notification can be sent to customer for regular service.
- We should provide contact detail of each service station which provide the emergency pickup service.
- The manager should know how much stock of spare parts are available in their garage and thus will be able to tell customers if the required part is available or not.

5. User Classes and Characteristics

To facilitate the implementation of the database and its functionalities, we divide the user interaction to the database into different classes. Each class of users has different roles and privileges assigned to them. Different users have different views of the database. This layered model is very useful, as it enables complex functions to be implemented in a structured way. Following is a detailed description of each class of users.

- **Customer:**
  Customers are the primary users of all the functionalities that the database offers. In this class, users are allowed to add their own entry into the customer table. They can modify their own details, but can't edit or access other customers' details. They are also allowed to view available slots and types of services offered, details of services. The online payment option is also available for end-users.

- **Employee:**
  Users of this class are associated with the service centre. So, they have more privileges compared to the users in the customer class. In this class, users can view all customer details, vehicle details of the customers, view the details of appointments, and can also view their profile. Another important task of employees is to manage payments done by the customers.

- **Manager/Administrator:**
  For the given service centre, the manager is a superuser. He/she has full access to all database functions. The administrator can view details of customers, employees, services available, details of all appointments done in the past, and so on. Along with that, the administrator can also edit the employee details, i.e., add a new employee, modify records of some employees, remove an employee, etc. The manager can add a new service offered by the service centre.

6. Operating Environment

- Database - We are using Pg_Admin, a management tool for PostgreSQL
- Operating system requirements – Windows 7 or higher versions, macOS 10.12 and above
- Minimum Hardware requirements - 2 GB RAM 2.4 GHz 1GB HDD Debian 9 (Stretch), 10 (Buster) or Ubuntu 16.04 (Xenial), 18.04 (Bionic), 19.10 (Eoan), 20.04 (Focal) or RHEL/CentOS 7 & 8 (x86_64) or Fedora 31 & 32 (x86_64)
- The browser support of Chrome 72+, Firefox 65+, Edge 44+, Safari 12+

6.1 User Interfaces

The GUI for the Customer side will be made simple consisting of basic functions that perform various task required by the Customer The home page also consist of information about the database and the creators. It is also having contact information and mailing address for better experience.

The GUI for the service center will display the task assigned, days left, and other features in an easy-to-use model Each servicer/mechanic will also be graded by the customers for the growth and development of the service centers.

6.2 Software Interface

- The database will communicate with customer to get all the basic details regarding the customer
- The database will communicate with vehicle information center to store the information about vehicle
- The database will communicate with slot booking dataset to get the details of slots booking, providing information about availability of slots, the total cost
- The database will communicate with the service station to provide the customer information about the service station
- The database will communicate with manager to get information about employee that is handling a particular customer
- The database will communicate with customer to get feedback post servicing the vehicle

6.2.1 Hardware Interfaces

Since the Application will be web-based service a steady internet connection will be required which can be accessed by connecting all the hardware e.g.-WAN – LAN, Ethernet Cross-Cable.

6.3Communications Interfaces

The system shall use the HTTP protocol for communication over the internet and for the intranet communication will be through TCP/IP protocol suite.

7. Product Functions

- **Registration** of Customer and Service Station: Each person and station need to register them self and verify detail in order to be able to access platform.

- **Vehicle Management:**
  - Adding new vehicle (with required detail)
  - Update detail of vehicle

- List a vehicle of particular owner
- Show history of vehicle (i.e., Previous Services)

- **Appointment Management:**
  - Book new appointment (Slot)
  - Cancel or reschedule Appointment
  - List booked slot
  - List available slot
  - List pending service
  - List available services for particular
  - List all station providing given service

- **Employee Management:**
  - List of employees and its detail
  - List of employees without confidential details

- **Payment Management:**
  - Reliable payment methods
  - Managing daily payment reports
  - Manage Bills

- **Inventory Management**
  - List spare parts available at garage
  - Update the list after every service

- **Feedback System:**
  - Customer feedback collection for each service
  - Update rating of Agent assigned according to the feedback

- **Notification System:**
  - Sending Notification for Service Alert

- **Emergency Service:**
  - Contact detail will be provided based on nearest location

8.Privileges

- **Registration** of Customer and Service Station: Each person and station need to register themselves and verify details in order to be able to access the platform.

- **Vehicle Management:**
  - Adding a new vehicle (with required detail): This function is for customers. So that he/she can add the new vehicle. Employees and manager too can add new vehicle in case of offline registration.
  - Update detail of vehicle: Update in the vehicle details is managed by employees after the service is done. So that the vehicle database remains up to date.
  - List a vehicle of particular owner: This functionality is mainly used by the service center employees when a customer possesses more than one vehicle. All of his/her vehicles can be serviced.
  - Show history of vehicle (i.e., Previous Services): This will be helpful to customers as well as employees to know about the history of the vehicle. Notification system will be based on this history only.

- **Appointment Management:**
  - Book new appointment (Slot): As mentioned, this is the function that is primarily used by the customers in order to get our services.
  - Cancel or reschedule Appointment: This is for customers to find and reschedule the appointment or cancel it. This makes the user experience great and acts in favor of the customer.
  - List booked slot: This is for employees and managers to know the schedule of the day.
  - List available slots: This is for customers to book appointments accordingly.
  - List pending service: This is also for Managers and employees' class to manage workload.
  - List available services for a particular service station: This is for customers to find available service
  - List all station providing a given service: This is for customer to select the best price service in their locality.

- **Employee Management:**
  - List of employees and their details: This is for managers to keep track of their employees.
  - List of employees without confidential details: This is for customers to know about rating and other details of agents assigned to them.

- **Payment Management:**
  - Reliable payment methods: Reliable payments are the necessity of all whether customer or employee or manager. So, this is ensured by the account's employees. The option of online payment is also made available for the ease of the customer.
  - Managing daily payment reports: This function comes under employees as well as managers can also keep track of this.
  - Manage Bills: Bill is generated after every successful payment and all bills for a particular day are managed by employees under the accounts department.

- **Inventory Management**
  - List spare parts available at garage: This is for managers so that they can keep order new parts if there are not enough available.
  - Update the list after every service: This saves managers' time by directly updating the list by deducting the number of parts used in a particular service.

- **Feedback System:**
  - Customer feedback collection for each service: Feedback is very important in terms of the improvements of our services. So after every service customer feedback is taken and some valuable suggestions are taken care of.
  - Update rating of Agent assigned according to the feedback: This task is accomplished by the customer feedback. Rating of the work done by some employees is immediately assigned as soon as feedback is given.

- **Notification System:**
  - Sending Notification for Service Alert: Notification for upcoming service is issued by the system as soon as the time for the next service comes. This will be helpful for good maintenance and good business of the service station.

- **Emergency Service:**
  - Contact details will be provided based on the nearest location: This service will be useful for vehicle owners in case of emergency. This will provide them with the first point of contact.

9. Assumption

- There is a stable electricity and internet connection for all parties so that they can access the system
- All parties satisfy the OS requirement
- All authorized personals have unique IDs.
- The information provided to the system is correct
- All the parties have the basic software and hardware
- All parties have basic knowledge of operating a computer
- The server of the system should be working for all days and at any given time
- All parties should have basic English knowledge so that they can fill the form appropriately

10. Business Constraints

- The enquired slot must be available.
- The required part is available to service center.
- The management is associated with service center near your area of requirement.
- The vehicle is a registered vehicle under the government registration with legitimate paperwork.
- A client is only allowed to create a single account on the website.
- If there is any malpractice by either the service provider or the client, they would be held responsible.

Section2: Noun & Verb Analysis

1. Find the are nouns (entities) or verbs (relationships) in sentences of the problem description using Noun Analysis Method.

NOUN	VERB
Project	Design
System	Facilitate
Database	Maintain
Customer	Bookings
Vehicle Owner	Register
Platform	Access
Vehicle Owner	Add
Vehicle Owner	Update
Vehicle Owner	Delete
Vehicle Owner	Book
Service Station	Receive
Service Station	Manage
Customers	Sign Up
Vehicles	Add
System	Maintain
Credential	Security
Customer	Provide
VIN number	Provide
Registered plate number	Provide
Customer	Update
Manager	Update
VIN number	Don’t remember
Phone number	Vehicle entries
Previous services	Done
Appointment	Provide easily

Customers	book
Appointment	Cancel
appointment	Reschedule
Time slot	Manage
Time slot	Book
Time slot	See
Current status	Know
Available services	List
Prices	List
Time required	List
Password	Safety
Prices	Compare
Different stations	Provide
Location	Provide
Employees’ details	Provide
Manager	See
Work done	Analyze
Manager	Help
Salary	Increase
Manager	Update
Customers	See
Non-confidential details	Access
Agent	Assigned
Payment Methods	Reliable
Online payment	Secure
Customer	Select
Daily payments	Manage
Payment	Corresponding

Data security	Required
Payment	Pending
Database	Implement
Record of income	maintain
Bills	E-Mail
System	Develop
Customer	Feedback
Employees' profile	Reflect
Quality of the service	Rate
Ratings	Update
Ratings	Indicate
Notification	Send
Customers	Alert
Last date	Based
Vehicle	Maintain
Business	Improve
Contact Detail	Provide
Location	Based
Survey	Accord
Emergency Services	Provide

2. Accepted Noun & Verbs list

Candidate Entity Set	**Candidate Attribute set**	**Candidate relationship set**	**Cardinality**	**Participation Constraint**
Feedback	Feedback ID Star Text	- Receive	- One	- Total
Service Station	Station ID Manager Name Location Contact Number of Workstation Emergency	- Provide - Has (with Booked Slot) - Has (withEmployee) - Has (with Sparepart)	- Many - Many - Many - Many	- Total - Partial - Total - Partial
Customer	Customer ID Customer Name Contact City	- Owns - has	- Many - One	- Partial -Total
Slot Booked	Slot_ID DropDate DropTime PickDate PickTime Work Station OdometerRead wStation Amount Amount paid Mode of Payment	- Assigned - Receive - Has(with ServiceStation) - Use Has (with Vehicle)	- One - One - One - Many - One	- Total - Partial - Total - Total - Total
Employee	Employee ID Employee Name Contact Salary Rating	- Assigned - Has	- Many - One	- Partial - Total

Vehicle	Vehicle ID Company Vehicle_Model Veicle_Type Plat Number VIN Last Service	- Owns - Has (with Booked Slot)	- One - Many	- Total - Partial
Service	Service ID Type Price	- Provide - Use (with BookedSlot) - Use (with Sparepart)	- One - Many - Many	- Total - Partial - Partial
Credentials	Email password	- Has (with Customer)	- One	- Total
Sparepart	ID_Sparepart (PK) Name_part Company	- Use (with Service) - Has (ServiceStation)	- Many - Many	- Total - Total

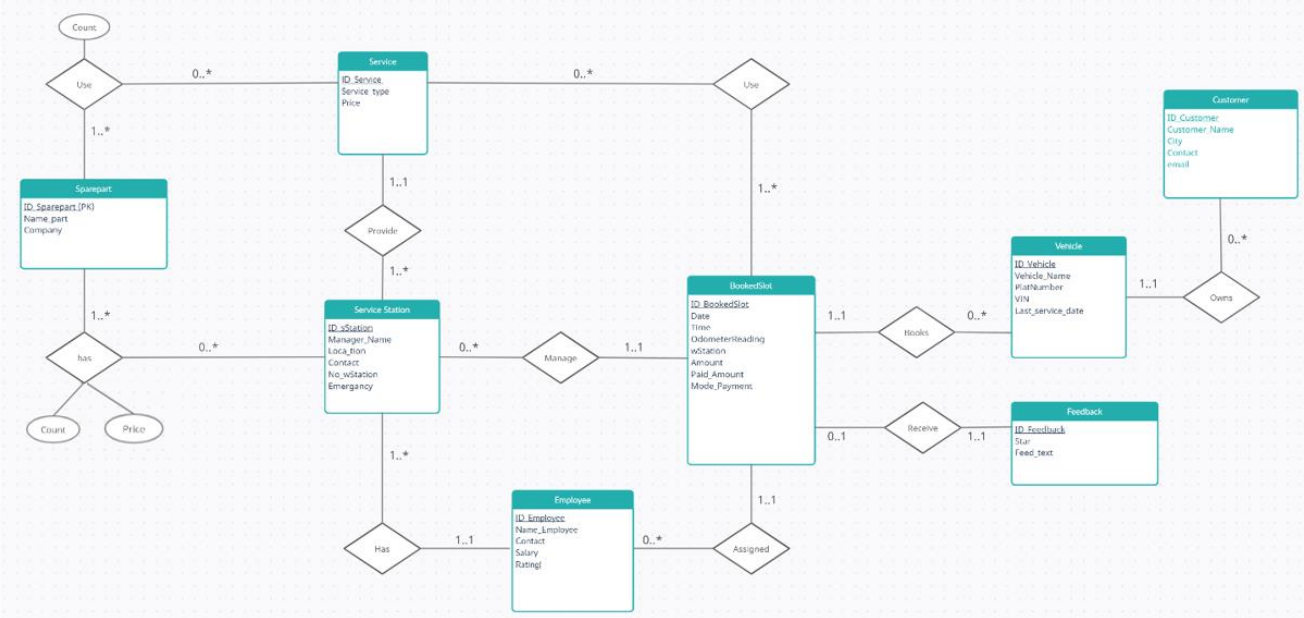
3. Rejected Noun & Verbs List

Noun	**Reject Reason**
Agent	Irrelevant
Data Security	Too general
Survey	Vague
Business	Association
Notifications	general
Quality of the Service	Duplicate
Last date	Irrelevant
Appointment	General
Employee profile	Vague
System	Association
Bills	Too general
Record of Income	Irrelevant
Work Done	Association
Customer	Duplicate
Customer	Duplicate
Notification	Association
Details	Vague
Payment	General
Database	General
Payments	Irrelevant
Payment Methods	Attribute
VIN Number	Irrelevant
Project	Vague
Registered Plate number	Irrelevant
Non-confidential details	Association
Vehicle	Duplicate

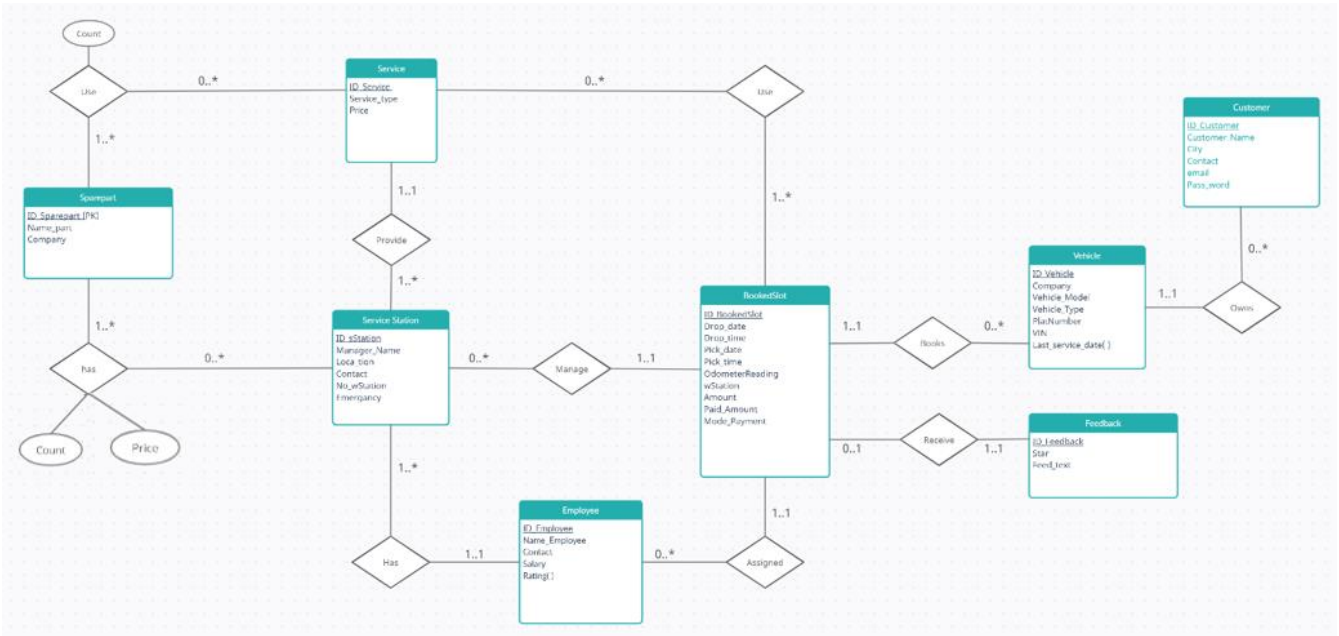
Project	Association
Vehicle Owner	Duplicate
Manager	Association
Manager	Duplicate
Time Required	Irrelevant
Survey	Association
Last date	Attribute
Record	Vague
Income	General
Salary	Attribute
Quality	Vague
Profile	General
Ratings	Attribute

Section3: ER-Diagram all versions

1. ER Diagram: Version-1



2. ER Diagram: Version-2



Section4: Final ER-Diagram to Relational Model Conversion

1. Entity to relational table:

- All the entity in our ER Diagram is strong entity set. So, we will keep all the attribute as it is and add the Foreign Key as per the relationship mentioned in 2nd point.
- We have two derived attributes. Rating in Employee entity and Last_service_date in Vehicle entity.
- So, we will not include it in Relational Schema as it will cause redundancy in data.

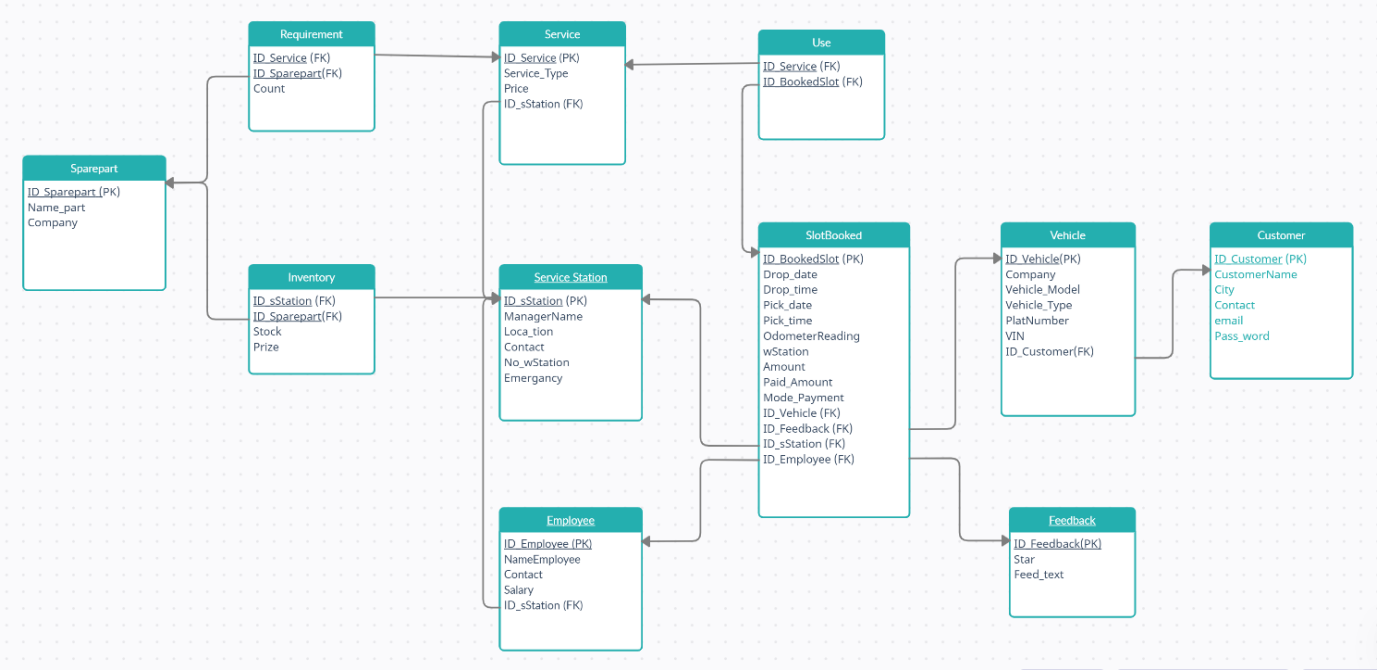
- **ServiceStation (**ID_sStation, Manager_Name, Loca_tion, Contact, No_wStation, Emergency**)**
- **Service (**ID_Service, Service_Type, Price, ID_sStation (FK)**)**
- **Employee (**ID_Employee, Name_Employee, Contact, Salary, ID_sStation (FK)**)**
- **Sparepart (**ID_Sparepart, Name_part, Company**)**
- **Credentials (**email, Pass_word**)**
- **Customer (**ID_Customer, Customer_Name, City, Contact, email (FK)**)**
- **Vehicle (**ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, PlatNumber, VIN, ID_Customer (FK)**)**
- **Feedback (**ID_Feedback, Star, Feed_text**)**
- **SlotBooked (**ID_BookedSlot, Drop_date, Drop_time, Pick_date, Pick_time, OdometerReading, wStation, Amount, Paid_Amount, Mode_Payment, ID_Vehicle (FK), ID_Feedback (FK), ID_Employee (FK), ID_sStation (FK)**)**

2. Relationship to relational table:

- For One-Many and Many-One we have added a primary key of relation which has cardinality “one” as FK in relation with cardinality “many” .
- Also, for one-one relationship we consider either side as “many”.
- For Many-Many type of relationship we introduce a new relation in relational schema.
- In our case we have three Many-Many relations.

- **Use: between Service and Bookedslot**
  - **Use (**ID_Service, ID_BookedSlot**)**
- **Use: between Service and Sparepat**
  - **Requirement (**ID_Service, ID_Sparepart, Count**)**
- **Has: between Sparepart and ServiceStation**
  - **Inventory (**ID_sStation, ID_Sparepart, Stock, Price**)**

3. Relational Schema:



Section5: Normalization and Schema Refinement

1. List of redundancies existing for every schema which is part of the database

- **ServiceStation (**ID_sStation, Manager_Name, Loca_tion, Contact, No_wStation, Emergency**)**
- **Service (**ID_Service, Service_Type, Price, ID_sStation (FK)**)**
- **Employee (**ID_Employee, Name_Employee, Contact, Salary, ID_sStation (FK)**)**
- **Customer (**ID_Customer, Customer_Name, City, Contact, email, password**)**
- **Vehicle (**ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, PlatNumber, VIN, ID_Customer (FK)**)**
- **Feedback (**ID_Feedback, Star, Feed_text**)**
- **SlotBooked (**ID_BookedSlot, Drop_date, Drop_time, Pick_date, Pick_time, OdometerReading, wStation, Amount, Paid_Amount, Mode_Payment, ID_Vehicle (FK), ID_Feedback (FK), ID_Employee (FK), ID_sStation (FK)**)**
- **Use (**ID_Service (FK), ID_BookedSlot (FK)**)**
- **Sparepart (**ID_Sparepart, Name_part, Company**)**
- **Inventory (**ID_sStation (FK), ID_Sparepart (FK)**,** Stock, Prize**)**
- **Requirement (**ID_Service (FK), ID_Sparepart (FK), Count**)**

➢ **No Redundancies found to be existing in the current schemas in database.**

2. List of update, delete, and insert anomalies for every schema

- **ServiceStation (**ID_sStation, Manager_Name, Loca_tion, Contact, No_wStation, Emergency**)**
  - **PK Dependency** ID_sStation → Manager_Name, Loca_tion, Contact, No_wStation, Emergency
  - **Functional Dependency:**
    ID_sStation → Manager_Name
    ID_sStation → Loca_tion
    ID_sStation →Contact
    ID_sStation →No_wStation
    ID_sStation → Emergency
  - **Partial Dependency:** None
  - **Transitive Dependency:** None
  - **Update Anomaly**: There is no update anomaly
  - **Insert Anomaly:** There is no insertion anomaly
  - **Delete Anomaly:** We cannot delete Service station until all the employee information is deleted

- **Service (**ID_Service, Service_Type, Price, ID_sStation (FK)**)**
  - **PK Dependency** ID_Service ->Service_Type, Price, ID_sStation (FK)
  - **Functional Dependency**
    ID_Service -> Service_Type
    ID_Service ->Price
    ID_Service ->ID_sStation
  - **Partial Dependency:** None
  - **Transitive Dependency:** None
  - **Update Anomaly**: We cannot update ID_sStation to Null as it is not null attribute for Service station
  - **Insert Anomaly:** None
  - **Delete Anomaly:** We cannot delete type until the count is deleted

- **Employee (**ID_Employee, Name_Employee, Contact, Salary, ID_sStation (FK)**)**
  - **PK Dependency** ID_Employee ->Name_Employee, Contact, Salary, ID_sStation (FK)
  - **Functional Dependency:**
    ID_employee ->Name_employee
    ID_employee ->Contact

ID_employee ->Salary
ID_employee ->ID_sStation

- **Partial Dependency:** None
- **Partial Dependency:** None
- **Transitive Dependency:** None
- **Update Anomaly**: We cannot update ID_sStation to Null as it is not null attribute for Service_station
- **Insert Anomaly:** There is no insertion anomaly
- **Delete Anomaly:** There is no deletion anomaly

- **Customer (**ID_Customer, Customer_Name, City, Contact, email, pass_word**)**
  - **PK Dependency** ID_Customer ->Customer_Name, City, Contact, email, pass_word
  - **Functional Dependency**
    ID_Customer -> Customer_Name
    ID_Customer -> City
    ID_Customer ->Contact
    ID_Customer ->email
    ID_Customer ->Pass_word
  - **Partial Dependency:** None
  - **Transitive Dependency:** None
  - **Update Anomaly**: No update anomaly exists
  - **Insert Anomaly:** ID_Customer can only be inserted if it exists in vehicle
  - **Delete Anomaly:** No delete anomaly exists

- **Vehicle (**ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, PlatNumber, VIN, ID_Customer (FK)**)**
  - **PK Dependency** ID_Vehicle ->Company, Vehicle_Model, Vehicle_Type, PlatNumber, VIN, ID_Customer (FK)**)**
  - **Functional Dependency**
    ID_Vehicle -> Company
    ID_Vehicle ->Vehicle_Model
    ID_Vehicle -> Vehicle_Type
    ID_Vehicle ->PlatNumber
    ID_Vehicle -> VIN
    ID_Vehicle -> ID_Customer
  - **Partial Dependency:** None
  - **Transitive Dependency:** None
  - **Update Anomaly**: ID_Customer cannot be a null value as it has Not Null attribute for Customer
  - **Insert Anomaly:** There is no insertion anomaly
  - **Delete Anomaly:** ID_Customer can only be deleted when customer database is deleted

- **Feedback (**ID_Feedback, Star, Feed_text**)**
  - **PK Dependency** ID_Feedback ->Star, Feed_text
  - **Functional Dependency**
    ID_Feedback -> Star
    ID_Feedback -> Feed_text
  - **Partial Dependency:** None
  - **Transitive Dependency:** None
  - **Update Anomaly**: There is no update anomaly
  - **Insert Anomaly:** There is no insertion anomaly
  - **Delete Anomaly:** There is no deletion anomaly

- **SlotBooked (**ID_BookedSlot, Drop_date, Drop_time, Pick_date, Pick_time, OdometerReading, wStation, Amount, Paid_Amount, Mode_Payment, ID_Vehicle (FK), ID_Feedback (FK), ID_Employee (FK), ID_sStation (FK)**)**
    - **PK Dependencies** ID_BookedSlot -> Drop_date, Drop_time, Pick_date, Pick_time, OdometerReading, wStation, Amount, Paid_Amount, Mode_Payment, ID_Vehicle (FK), ID_Feedback (FK), ID_Employee (FK), ID_sStation (FK)
    - **Functional Dependency**
        ID_BookedSlot -> Drop_date
        ID_BookedSlot ->Drop_time
        ID_BookedSlot ->Pick_date
        ID_BookedSlot ->OdometerReading
        ID_BookedSlot -> wStation
        ID_BookedSlot -> Amount
        ID_BookedSlot -> Mode_payment
        ID_BookedSlot -> ID_vehicle
        ID_BookedSlot ->ID_feedback
        ID_BookedSlot ->ID_Employee
        ID_BookedSlot ->ID_sStation
    - **Partial Dependency:** None
    - **Transitive Dependency:** None
    - **Update Anomaly**: ID_Vehicle, ID_Feedback, ID_Employee cannot be updated
    - **Insert Anomaly:** ID_Employee that does not exist cannot be inserted
    - **Delete Anomaly:** ID_Vehicle can only be deleted after deleting the vehicle information

- **Use (**ID_Service (FK), ID_BookedSlot (FK)**)**
    - **Partial Dependency:** None
    - **Transitive Dependency:** None
    - **Update Anomaly**: ID_BookedSlot cannot be NULL as it has a NOT NULL attribute from SlotBooked
    - **Insert Anomaly:** ID_BookSlot can only be inserted if it is present in slotbooked
    - **Delete Anomaly:** IS_service cannot be deleted until service is completed

- **SparePart (**ID_Sparepart, name_part, company**)**
    - **PK Depedency** ID_spareparts -> name_part,company
    - **Functional Dependency**
        ID_spareparts -> name_part
        ID_spareparts -> Company
    - **Partial Dependency:** None
    - **Transitive Dependency:** None
    - **Update Anomaly**: There is no update anomaly
    - **Insert Anomaly:** There is no insertion anomaly
    - **Delete Anomaly:** There is no deletion anomaly

- **Inventory (**ID_sStation (FK), ID_Sparepart (FK), Stock, Prize**)**
    - **PK Dependency** ID_sStation,ID_spareparts -> stock,Prize
    - **FunctionalDependency**
        ID_sStation,ID_spareparts -> stock
        ID_sStation,ID_spareparts -> Prize
    - **Partial Dependency:** None
    - **Transitive Dependency:** None
    - **Update Anomaly**: NO update anomaly exists

- **Insert Anomaly:** ID_Sparepart can only be inserted if it preset in spare parts
- **Delete Anomaly:** ID_Sparepart can only be deleted when the information is deleted for inventory table

- **Requirement (**ID_Sevice (FK), ID_Sparepart (FK), Count**)**
  - **PK Dependency** ID_Sevice,ID_spareparts ->count
  - **Functional Dependency**
    ID_Sevice, ID_spareparts ->count
  - **Partial Dependency:** None
  - **Transitive Dependency:** None
  - **Update Anomaly:** ID_Sparepart cannot be null as it is a not NULL attribute for inventory
  - **Insert Anomaly:** There is no insertion anomaly
  - **Delete Anomaly:** ID_Sparepart can only be deleted if it not presents in inventory

3. Document the logic of how you arrived at the 3NF/BCNF design step by step starting from the original design.

1) 1NF:

- We have only 4 relation which have composite attribute. So all other relations are in 1NF form. Below is the 4 relation which has composite attribute.
  - **ServiceStation (**ID_sStation, Manager_Name, Loca_tion, Contact, No_wStation, Emergency**)**
    - **Manager_Name:** This attribute is composite attribute with two different attribute Manager_FirstName and Manager_LastName.
    - So, we will change schema to,
      ServiceStation (ID_sStation, Manager_FirstName, Manager_LastName, Loca_tion, Contact, No_wStation, Emergency**)**

  - **Employee (**ID_Employee, Name_Employee, Contact, Salary, ID_sStation (FK)**)**
    - **Name_Employee:** This attribute is composite attribute with two different attribute Employee_FirstName and Employee_LastName.
    - So, we will change schema to,
      Employee (ID_Employee, Employee_FirstName, Employee_LastName, Contact, Salary, ID_sStation (FK)**)**

  - **Customer (**ID_Customer, Customer_Name, City, Contact, email (FK)**)**
    - **Customer_Name:** This attribute is composite attribute with two different attribute Customer_FirstName and Customer_LastName.
    - So, we will change schema to,
      Customer (ID_Customer, Customer_FirstName, Customer_LastName, City, Contact, email (FK)**)**

  - **Vehicle (**ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, PlatNumber, VIN, ID_Customer (FK)**)**
    - **PlatNumber:** This attribute is composite attribute with three different attribute Plat_StateCode, Plat_DistCode and Plat_CodeNumber
    - So, we will change schema to,
      Vehicle (ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, Plat_StateCode, Plat_DistCode, Plat_CodeNumber, VIN, ID_Customer (FK)**)**

2) Updated Schema(1NF)

- To convert relational schema in to 1NF, we have replaced all the composite attribute with the atomic attributes.

- These are the new schema:
  - **ServiceStation (**ID_sStation, Manager_FirstName, Manager_LastName, Loca_tion, Contact, No_wStation, Emergency**)**
  - **Service (**ID_Service, Service_Type, Price, ID_sStation (FK)**)**
  - **Employee (**ID_Employee, Employee_FirstName, Employee_LastName, Contact, Salary, ID_sStation (FK)**)**
  - **Customer (**ID_Customer, Customer_FirstName, Customer_LastName, City, Contact, email, pass_word**)**
  - **Vehicle (**ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, Plat_StateCode, Plat_DistCode, Plat_CodeNumber, VIN, ID_Customer (FK)**)**
  - **Feedback (**ID_Feedback, Star, Feed_text**)**
  - **SlotBooked (**ID_BookedSlot, Drop_date, Drop_time, Pick_date, Pick_time, OdometerReading, wStation, Amount, Paid_Amount, Mode_Payment, ID_Vehicle (FK), ID_Feedback (FK), ID_Employee (FK), ID_sStation (FK)**)**
  - **Use (**ID_Service (FK), ID_BookedSlot (FK)**)**
  - **Sparepart (**ID_Sparepart, Name_part, Company**)**
  - **Inventory (**ID_sStation (FK), ID_Sparepart (FK)**,** Stock, Prize**)**
  - **Requirement (**ID_Service (FK), ID_Sparepart (FK), Count**)**

3) 2NF:

- To make the schema of Second Normal Form, we need to remove all the Partial Dependency.
- As we know all the relations with single attribute PK are already in 2NF.
- So, in this database all the relations except Use, Inventory and Requirement are having single attribute PK. Hence, they are in 2NF.
- In case of relation “Use”, it has no other attribute than PK. So, it does not have any partial dependency and hence “Use” is in the 2NF.

4) Updated Schema (after 2NF) will remain same

5) 3NF

- To make the schema of Third Normal form we need to remove all the transitive Dependencies
- As we can see from above the schema was in 2NF form we will now see transitive dependencies
- Now by checking all the relation we found no transitive dependencies
- Hence the schema is in 3 NF form

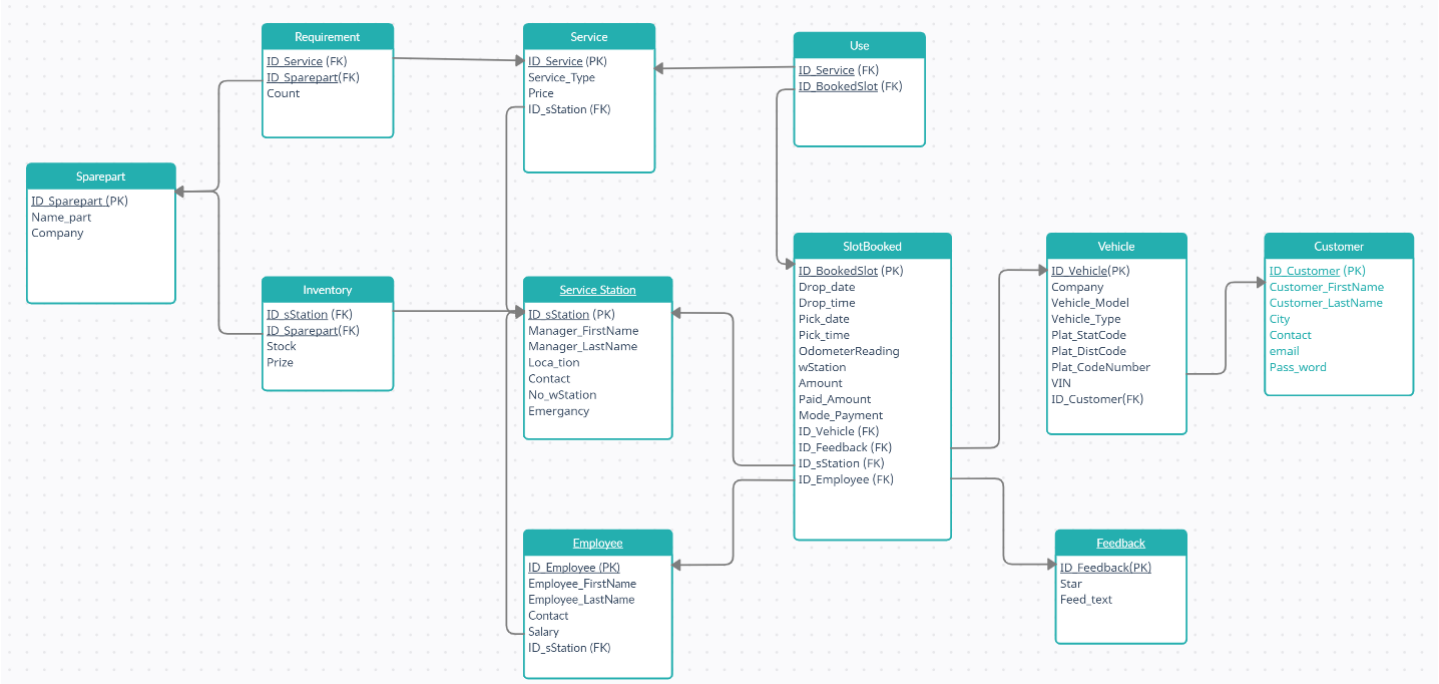
6) Updated Schema (after 3NF) will remain same

7) Final Schema

- **ServiceStation (**ID_sStation, Manager_FirstName, Manager_LastName, Loca_tion, Contact, No_wStation, Emergency**)**
- **Service (**ID_Service, Service_Type, Price, ID_sStation (FK)**)**
- **Employee (**ID_Employee, Employee_FirstName, Employee_LastName, Contact, Salary, ID_sStation (FK)**)**
- **Customer (**ID_Customer, Customer_FirstName, Customer_LastName, City, Contact, email, pass_word**)**
- **Vehicle (**ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, Plat_StateCode, Plat_DistCode, Plat_CodeNumber, VIN, ID_Customer (FK)**)**
- **Feedback (**ID_Feedback, Star, Feed_text**)**

- **SlotBooked (**ID_BookedSlot, Drop_date, Drop_time, Pick_date, Pick_time, OdometerReading, wStation, Amount, Paid_Amount, Mode_Payment, ID_Vehicle (FK), ID_Feedback (FK), ID_Employee (FK), ID_sStation (FK)**)**
- **Use (**ID_Service (FK), ID_BookedSlot (FK)**)**
- **SparePart (**ID_Spareparts, Name_part, Company**)**
- **Inventory** (ID_sStation (FK), ID_Spareparts (FK), Stock, Prize**)**
- **Requirement (**ID_Service (FK), ID_Sparepart (FK), Count**)**

4. Final Relational Schema



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

1. DDL queries:

- **ServiceStation (**ID_sStation, Manager_FirstName, Manager_LastName, Loca_tion, Contact, No_wStation, Emergency**)**

  - As each service station should have different contact number, we have added unique constraint on it.

  - **DDL Query:**

```
Create table if not exists "ASC_DB".ServiceStation
(
        ID_sStation             integer   NOT NULL,
        Manager_FirstName       text,
        Manager_LastName        text,
        Loca_tion               text,
        Contact                 char(10),
        No_wStation             integer,
        Emergency               bool,

        PRIMARY KEY (ID_sStation),

        UNIQUE(Contact)
);
```

- **Service (**ID_Service, Service_Type, Price, ID_sStation (FK)**)**

  - On update in ID_sStation in ServiceStation, we set cascade in this entity. So that it will automatically change here too.
  - On delete we set Foreign Key to NULL as we need to show the service type and other detail on demand for vehicle history.

  - **DDL Query:**

```
Create table if not exists "ASC_DB".Service
(
        ID_Service              integer   NOT NULL,

        Service_Type            text,

        Price                   integer,

        ID_sStation             integer,


        PRIMARY KEY (ID_Service),


        FOREIGN KEY (ID_sStation) REFERENCES "ASC_DB".ServiceStation
        (ID_sStation) ON UPDATE CASCADE ON DELETE SET NULL

);
```

- **Employee (**ID_Employee, Employee_FirstName, Employee_LastName, Contact, Salary, ID_sStation (FK)**)**

  - Here we have set cascade on update and on delete. So, it will do update corresponding to update in ServiceStation entity and on delete as it means service station no longer exist therefore delete employee.

  - **DDL Query:**

```
Create table if not exists "ASC_DB".Employee
(
        ID_Employee             integer   NOT NULL,
        Employee_FirstName      text,
        Employee_LastName       text,
        Contact                 char(10),
        Salary                  integer,

        ID_sStation             integer,


        PRIMARY KEY (ID_Employee),


        FOREIGN KEY (ID_sStation) REFERENCES "ASC_DB".ServiceStation
        (ID_sStation) ON UPDATE CASCADE ON DELETE CASCADE,


        UNIQUE(Contact)
);
```

- **Customer (**ID_Customer, Customer_FirstName, Customer_LastName, City, Contact, email, pass_word**)**

  - As Contact is unique for each customer, we add UNIQUE constraint on it.

  - **DDL Query:**

```
Create table if not exists "ASC_DB".Customer
(
        ID_Customer             integer   NOT NULL,

        Customer_FirstName      text,

        Customer_LastName       text,

        City                    text,

        Contact                 char(10),

        email                   text,


        PRIMARY KEY (ID_Customer),

        UNIQUE(Contact),
        UNIQUE(email)
);
```

- **Vehicle (**ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, Plat_StateCode, Plat_DistCode, Plat_CodeNumber, VIN, ID_Customer (FK)**)**

  - VIN is strictly of the size 17, therefore we keep its type fixed.
  - VIN and Plate number is unique for each vehicle. So, UNIQUE constraint is added for them.
  - Foreign key is kept cascade on delete and on update. So, the corresponding data will be removed once the customer get removed from database.

  - **DDL Query:**

```
Create table if not exists "ASC_DB".Vehicle
(
        ID_Vehicle          integer   NOT NULL,
        Company             text,
        Vehicle_Model       text,
        Vehicle_Type        text,
        Plat_StateCode      text,
        Plat_DistCode       integer,
        Plat_CodeNumber     text,
        VIN                 char(17),
        ID_Customer         integer,

        PRIMARY KEY (ID_Vehicle),

        FOREIGN KEY (ID_Customer) REFERENCES "ASC_DB".Customer
        (ID_Customer) ON UPDATE CASCADE ON DELETE CASCADE,

        UNIQUE(VIN),
        UNIQUE(Plat_StateCode, Plat_DistCode, Plat_CodeNumber)
);
```

- **Feedback (**ID_Feedback, Star, Feed_text**)**

  - We want customer to give feedback on the scale from 1 to 5. Therefore, the CHECK condition is added for it.

  - **DDL Query:**

```
Create table if not exists "ASC_DB".Feedback
(
        ID_Feedback         integer   NOT NULL,
        Star                integer   NOT NULL,
```

```
    Feed_text                text        NOT NULL,


    PRIMARY KEY (ID_Feedback),

    CHECK(Star>=1 and Star<=5)

);
```

- **SlotBooked (**ID_BookedSlot, Drop_date, Drop_time, Pick_date, Pick_time, OdometerReading, wStation, Amount, Paid_Amount, Mode_Payment, ID_Vehicle (FK), ID_Feedback (FK), ID_Employee (FK), ID_sStation (FK)**)**

  - We have keep the DEFAULT mode of payment in Cash.
  - Check condition to verify that manager or employee have enter valid Pickup and drop time/date. The condition is pickup time_date should be greater then that of the dropoff.
  - We will also make sure that Paid Amount is not greater than the bill.
  - On delete for ID_Vehicle only we will cascade in this table. On delete for other foreign key the data for vehicle will not be dropped in order to maintain History.
  - On Update in case of each foreign key we have set cascade. So, corresponding changes will be reflected.

  - **DDL Query:**

```
Create table if not exists "ASC_DB".SlotBooked

(
    ID_BookedSlot        integer   NOT NULL,
    Drop_date            date,
    Drop_time            time,
    Pick_date            date,
    Pick_time            time,
    OdometerReading      integer,
    wStation             integer,
    Amount               integer,
    Paid_Amount          integer,
    Mode_Payment         text      DEFAULT 'Cash',
    ID_Vehicle           integer,
    ID_Feedback          integer,
    ID_Employee          integer,
    ID_sStation          integer,


    PRIMARY KEY (ID_BookedSlot),


    FOREIGN KEY (ID_Vehicle) REFERENCES "ASC_DB".Vehicle (ID_Vehicle) ON
    UPDATE CASCADE ON DELETE CASCADE,
```

```
FOREIGN KEY (ID_Feedback) REFERENCES "ASC_DB".Feedback
(ID_Feedback) ON UPDATE CASCADE ON DELETE SET NULL,


FOREIGN KEY (ID_Employee) REFERENCES "ASC_DB".Employee
(ID_Employee) ON UPDATE CASCADE ON DELETE SET NULL,


FOREIGN KEY (ID_sStation) REFERENCES "ASC_DB".ServiceStation
(ID_sStation) ON UPDATE CASCADE ON DELETE SET NULL,


CHECK (Drop_date <= Pick_date),


CHECK (Paid_Amount <= Amount),


CHECK (case
            when Drop_date = Pick_date
            then Drop_time < Pick_time
      end),


CHECK (Mode_Payment = 'Card' or Mode_Payment = 'Cheque' or
Mode_Payment = 'Cash' or Mode_Payment = 'UPI')
);
```

- **Use (**ID_Service , ID_BookedSlot**)**

  - On update we have set cascade and therefore corresponding changes will be reflected.
  - On Delete for service we will not allowed that operation. As that is required to maintain history of the vehicles.
  - On Delete for SlotBooked, we have set cascade as the data will be redundant once the appointment will be deleted.

  - **DLD Query:**

```
Create table if not exists "ASC_DB".Use
(
        ID_Service              integer,
        ID_BookedSlot           integer   NOT NULL,


        PRIMARY KEY (ID_Service, ID_BookedSlot),


        FOREIGN KEY (ID_Service) REFERENCES "ASC_DB".Service(ID_Service) ON
        UPDATE CASCADE,
```

```
        FOREIGN KEY (ID_BookedSlot) REFERENCES "ASC_DB".SlotBooked
        (ID_BookedSlot) ON UPDATE CASCADE ON DELETE CASCADE
);
```

- **SparePart (**ID_Spareparts, Name_part, Company**)**

  - **DLD Query:**

```
Create table if not exists "ASC_DB".Sparepart
(
        ID_Sparepart            integer,
        Name_part               text,
        Company                 text,

        PRIMARY KEY (ID_Sparepart)
    );
```

- **Inventory** (ID_sStation (FK), ID_Spareparts (FK), Stock, Prize**)**
  - On delete of the Service station which is pointed by FK ID_sStation we will delete the pair as it would be not of any use. So, we use CASCADE on delete.
  - On update changes will be reflected in case of both the FK.
  - We will not allow deletion of Sparepart tuple till there exist service in Service relation.
  - **DLD Query:**

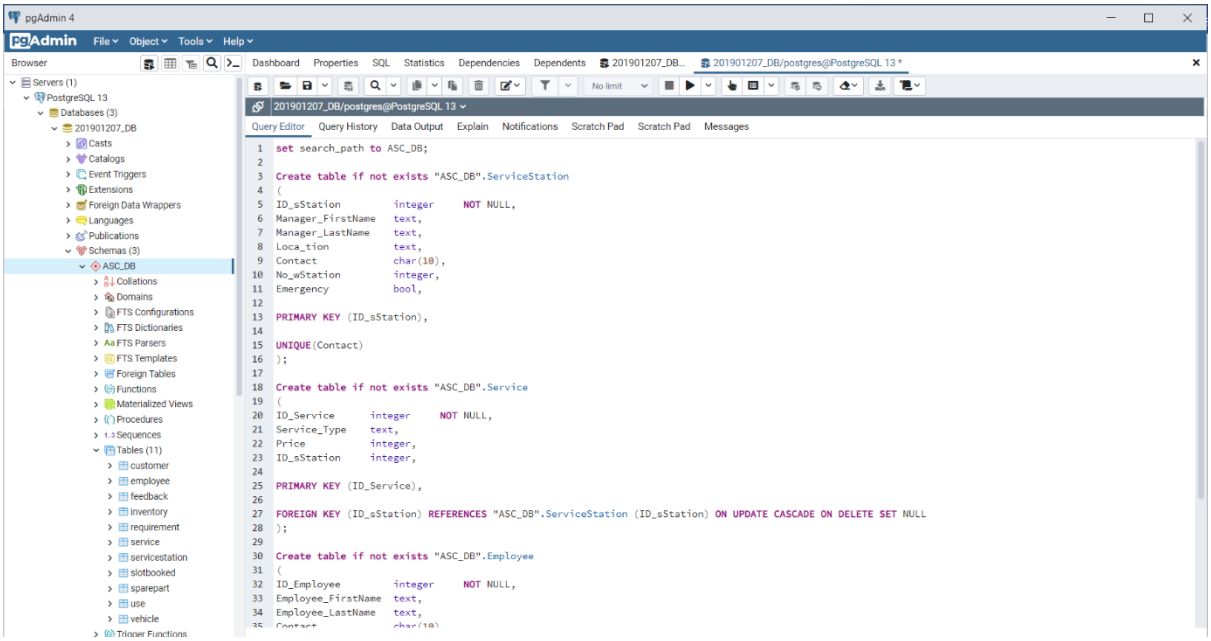
```
Create table if not exists "ASC_DB".Inventory
(
        ID_sStation             integer,
        ID_Sparepart            integer,
        Stock                   integer,
        Prize                   integer,

        PRIMARY KEY (ID_sStation, ID_Sparepart),

        FOREIGN KEY (ID_sStation) REFERENCES
        "ASC_DB".ServiceStation(ID_sStation) ON UPDATE CASCADE ON
        DELETE CASCADE,
```

```
        FOREIGN KEY (ID_Sparepart) REFERENCES "ASC_DB".Sparepart
        (ID_Sparepart) ON UPDATE CASCADE

);
```

- **Requirement (**ID_Service (FK), ID_Sparepart (FK), Count**)**
  - On delete of the service which is pointed by FK ID_Service we will delete the pair as it would be not of any use. So we use CASCADE on delete.
  - On update changes will be reflected in case of both the FK.
  - We will not allow deletion of Sparepart tuple till there exist service in Service relation.
  -
  - **DLD Query:**

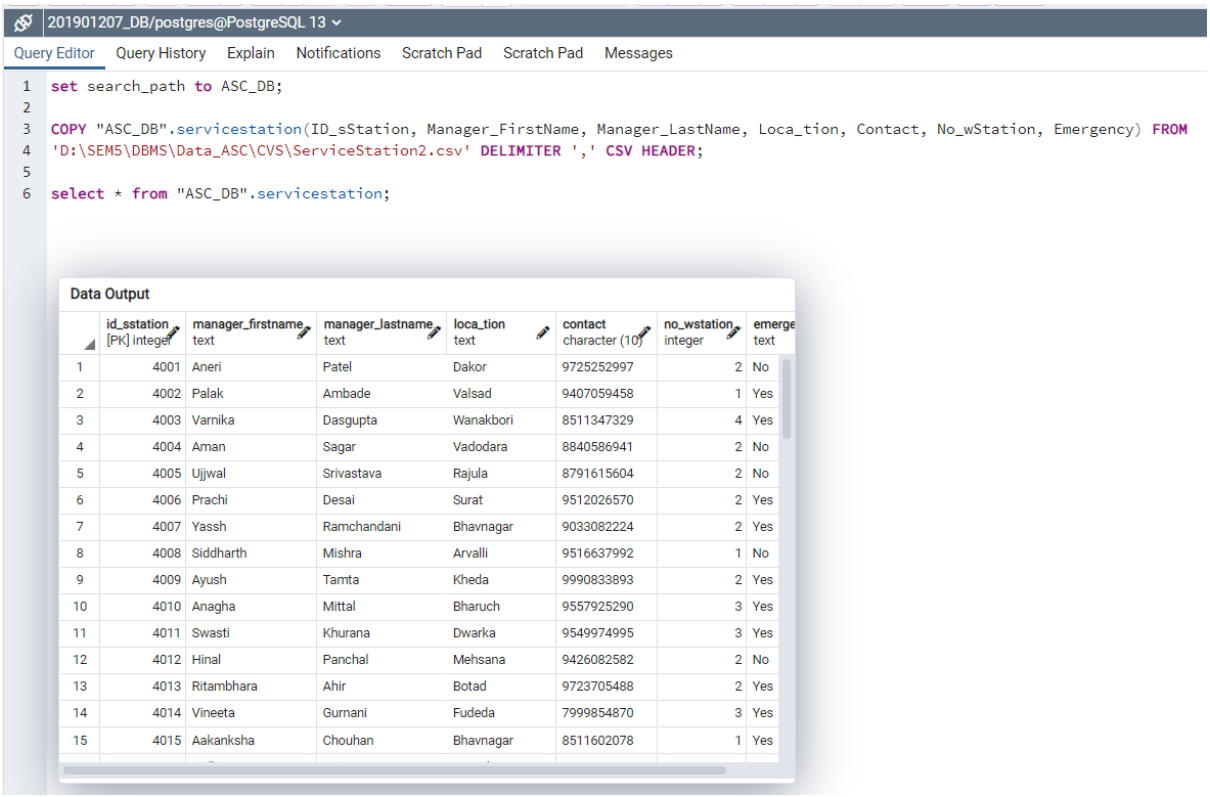
```
Create table if not exists "ASC_DB".Requirement

(

        ID_Service              integer,

        ID_Sparepart            integer,

        Count                   integer,


        PRIMARY KEY (ID_Service, ID_Sparepart),


        FOREIGN KEY (ID_Service) REFERENCES "ASC_DB".Service(ID_Service)
        ON UPDATE CASCADE ON DELETE CASCADE,


        FOREIGN KEY (ID_Sparepart) REFERENCES "ASC_DB".Sparepart
        (ID_Sparepart) ON UPDATE CASCADE

);
```

2. DDL Snapshot:



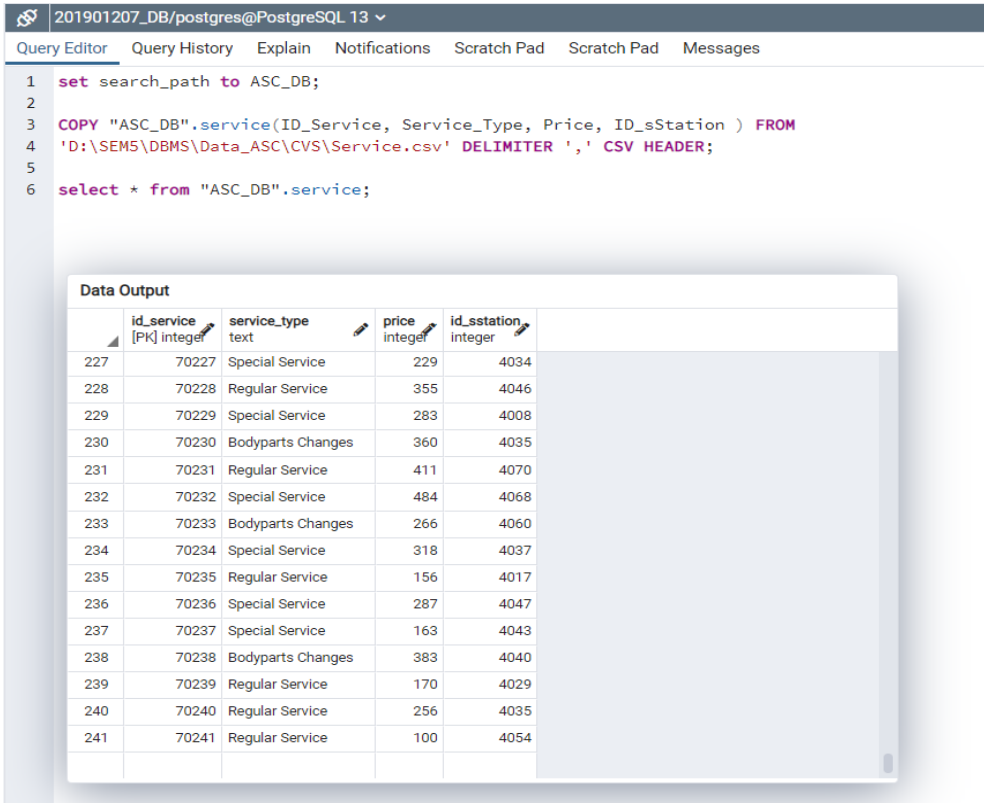
3. DATA Insert Snapshot:

- **ServiceStation (**ID_sStation, Manager_FirstName, Manager_LastName, Loca_tion, Contact, No_wStation, Emergency**)**



	id_sstation [PK] integer	manager_firstname text	manager_lastname text	loca_tion text	contact character (10)	no_wstation integer	emerge text
1	4001	Aneri	Patel	Dakor	9725252997	2	No
2	4002	Palak	Ambade	Valsad	9407059458	1	Yes
3	4003	Varnika	Dasgupta	Wanakbori	8511347329	4	Yes
4	4004	Aman	Sagar	Vadodara	8840586941	2	No
5	4005	Ujjwal	Srivastava	Rajula	8791615604	2	No
6	4006	Prachi	Desai	Surat	9512026570	2	Yes
7	4007	Yassh	Ramchandani	Bhavnagar	9033082224	2	Yes
8	4008	Siddharth	Mishra	Arvalli	9516637992	1	No
9	4009	Ayush	Tamta	Kheda	9990833893	2	Yes
10	4010	Anagha	Mittal	Bharuch	9557925290	3	Yes
11	4011	Swasti	Khurana	Dwarka	9549974995	3	Yes
12	4012	Hinal	Panchal	Mehsana	9426082582	2	No
13	4013	Ritambhara	Ahir	Botad	9723705488	2	Yes
14	4014	Vineeta	Gurnani	Fudeda	7999854870	3	Yes
15	4015	Aakanksha	Chouhan	Bhavnagar	8511602078	1	Yes

- **Service (**ID_Service, Service_Type, Price, ID_sStation (FK)**)**

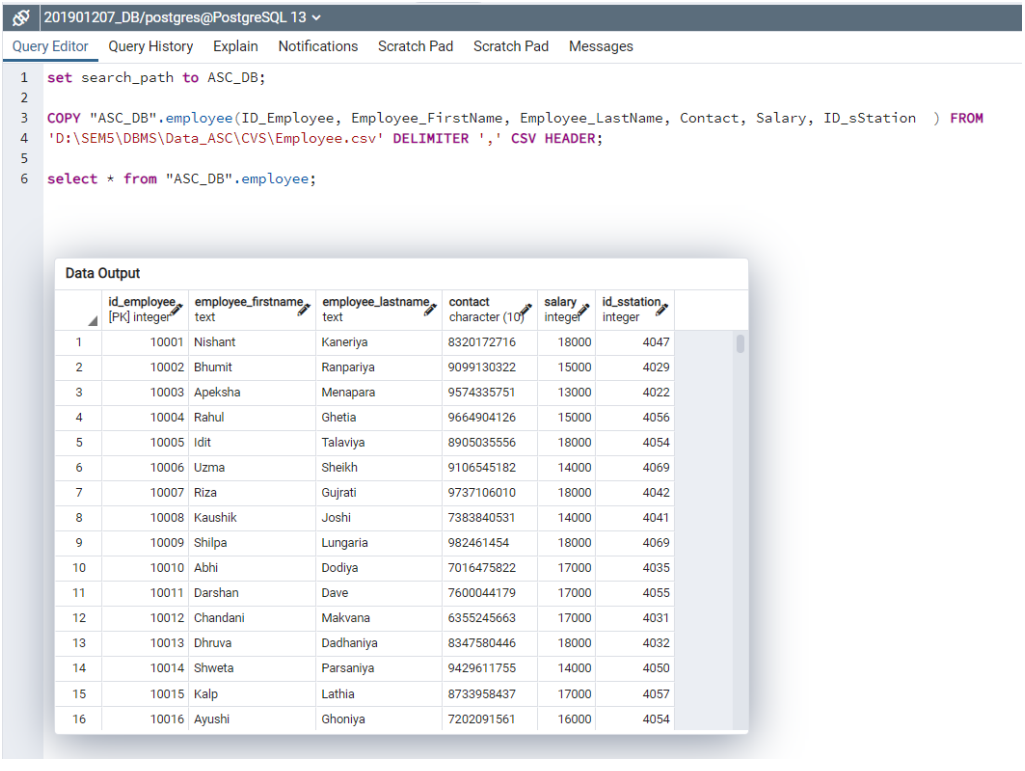


```
set search_path to ASC_DB;

COPY "ASC_DB".service(ID_Service, Service_Type, Price, ID_sStation ) FROM
'D:\SEM5\DBMS\Data_ASC\CVS\Service.csv' DELIMITER ',' CSV HEADER;

select * from "ASC_DB".service;
```

Data Output

	id_service [PK] integer	service_type text	price integer	id_sstation integer
227	70227	Special Service	229	4034
228	70228	Regular Service	355	4046
229	70229	Special Service	283	4008
230	70230	Bodyparts Changes	360	4035
231	70231	Regular Service	411	4070
232	70232	Special Service	484	4068
233	70233	Bodyparts Changes	266	4060
234	70234	Special Service	318	4037
235	70235	Regular Service	156	4017
236	70236	Special Service	287	4047
237	70237	Special Service	163	4043
238	70238	Bodyparts Changes	383	4040
239	70239	Regular Service	170	4029
240	70240	Regular Service	256	4035
241	70241	Regular Service	100	4054

- **Employee (**ID_Employee, Employee_FirstName, Employee_LastName, Contact, Salary, ID_sStation (FK)**)**

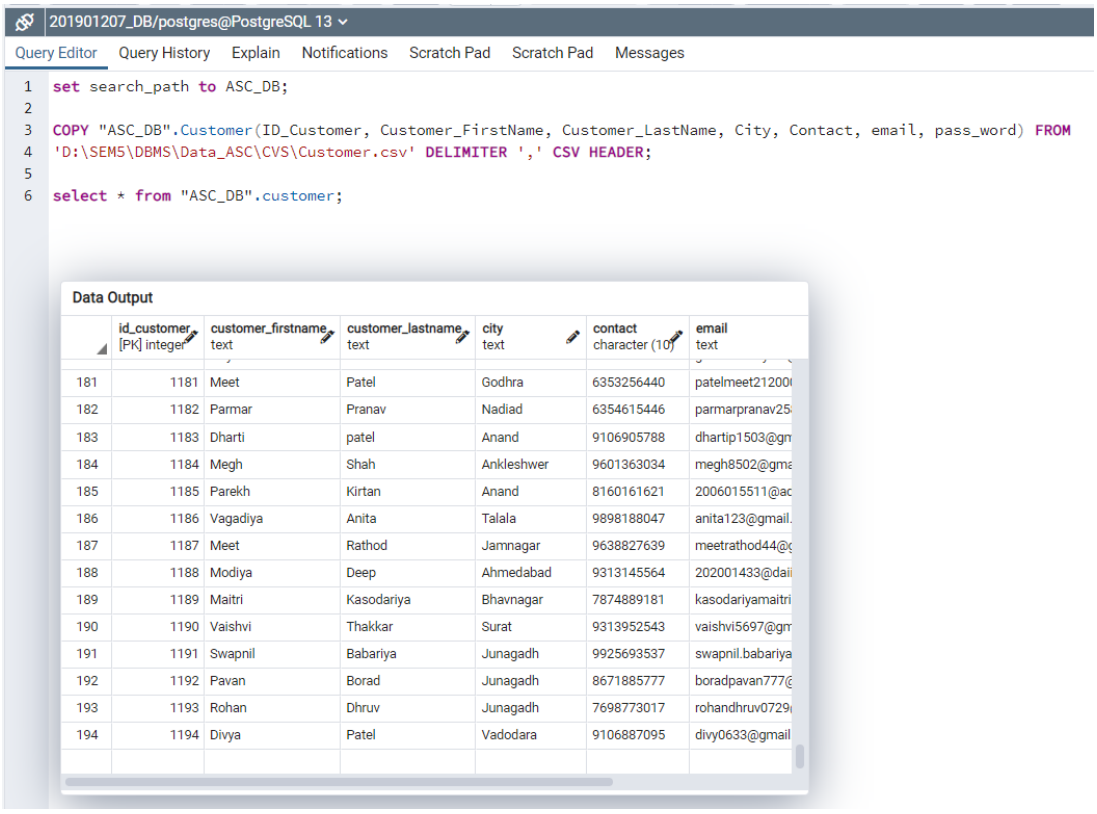


```
set search_path to ASC_DB;

COPY "ASC_DB".employee(ID_Employee, Employee_FirstName, Employee_LastName, Contact, Salary, ID_sStation  ) FROM
'D:\SEM5\DBMS\Data_ASC\CVS\Employee.csv' DELIMITER ',' CSV HEADER;

select * from "ASC_DB".employee;
```

Data Output

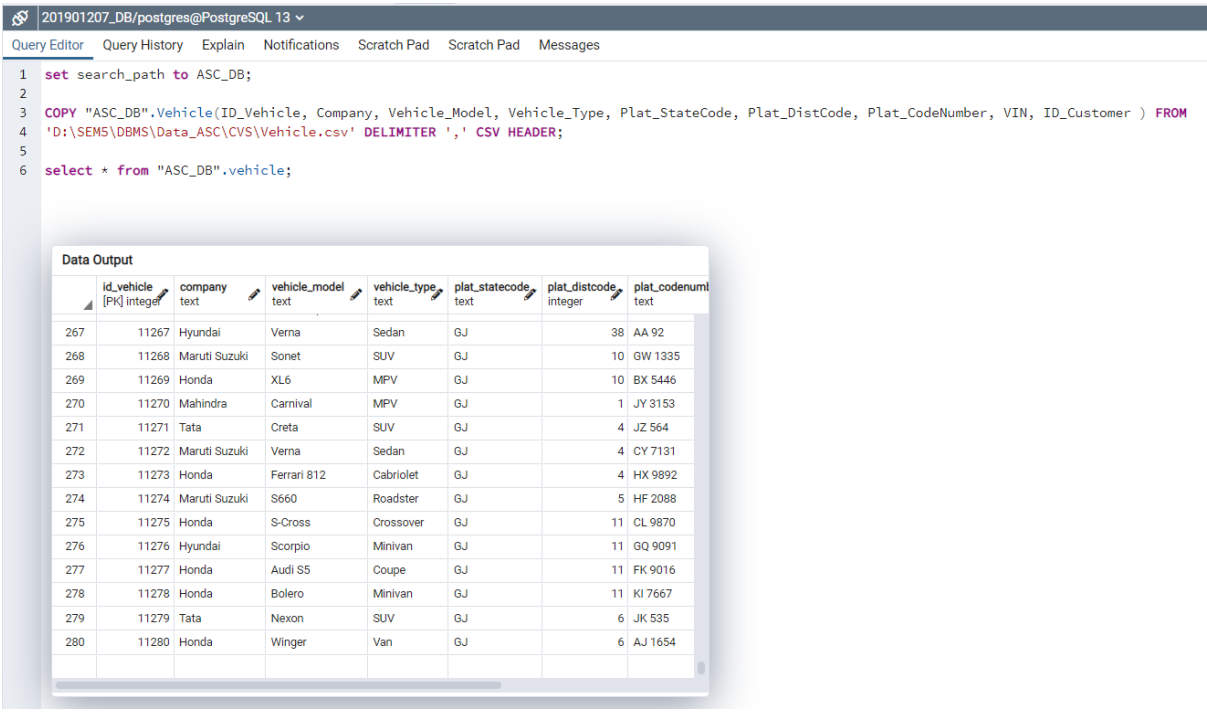
	id_employee [PK] integer	employee_firstname text	employee_lastname text	contact character (10)	salary integer	id_sstation integer
1	10001	Nishant	Kaneriya	8320172716	18000	4047
2	10002	Bhumit	Ranpariya	9099130322	15000	4029
3	10003	Apeksha	Menapara	9574335751	13000	4022
4	10004	Rahul	Ghetia	9664904126	15000	4056
5	10005	Idit	Talaviya	8905035556	18000	4054
6	10006	Uzma	Sheikh	9106545182	14000	4069
7	10007	Riza	Gujrati	9737106010	18000	4042
8	10008	Kaushik	Joshi	7383840531	14000	4041
9	10009	Shilpa	Lungaria	982461454	18000	4069
10	10010	Abhi	Dodiya	7016475822	17000	4035
11	10011	Darshan	Dave	7600044179	17000	4055
12	10012	Chandani	Makvana	6355245663	17000	4031
13	10013	Dhruva	Dadhaniya	8347580446	18000	4032
14	10014	Shweta	Parsaniya	9429611755	14000	4050
15	10015	Kalp	Lathia	8733958437	17000	4057
16	10016	Ayushi	Ghoniya	7202091561	16000	4054

- **Customer (**ID_Customer, Customer_FirstName, Customer_LastName, City, Contact, email, pass_word**)**



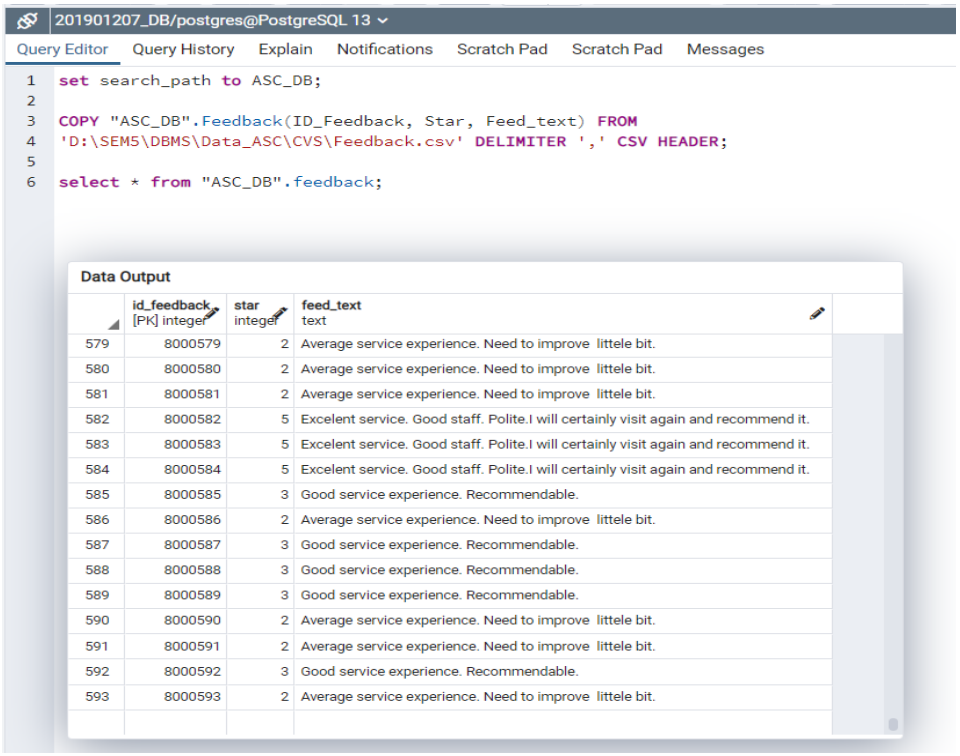
	id_customer [PK] integer	customer_firstname text	customer_lastname text	city text	contact character (10)	email text
181	1181	Meet	Patel	Godhra	6353256440	patelmeet21200
182	1182	Parmar	Pranav	Nadiad	6354615446	parmarpranav25
183	1183	Dharti	patel	Anand	9106905788	dhartip1503@gn
184	1184	Megh	Shah	Ankleshwer	9601363034	megh8502@gma
185	1185	Parekh	Kirtan	Anand	8160161621	2006015511@ac
186	1186	Vagadiya	Anita	Talala	9898188047	anita123@gmail.
187	1187	Meet	Rathod	Jamnagar	9638827639	meetrathod44@
188	1188	Modiya	Deep	Ahmedabad	9313145564	202001433@dai
189	1189	Maitri	Kasodariya	Bhavnagar	7874889181	kasodariyamaitri
190	1190	Vaishvi	Thakkar	Surat	9313952543	vaishvi5697@gm
191	1191	Swapnil	Babariya	Junagadh	9925693537	swapnil.babariya
192	1192	Pavan	Borad	Junagadh	8671885777	boradpavan777@
193	1193	Rohan	Dhruv	Junagadh	7698773017	rohandhruv0729
194	1194	Divya	Patel	Vadodara	9106887095	divy0633@gmail

- **Vehicle (**ID_Vehicle, Company, Vehicle_Model, Vehicle_Type, Plat_StateCode, Plat_DistCode, Plat_CodeNumber, VIN, ID_Customer (FK)**)**



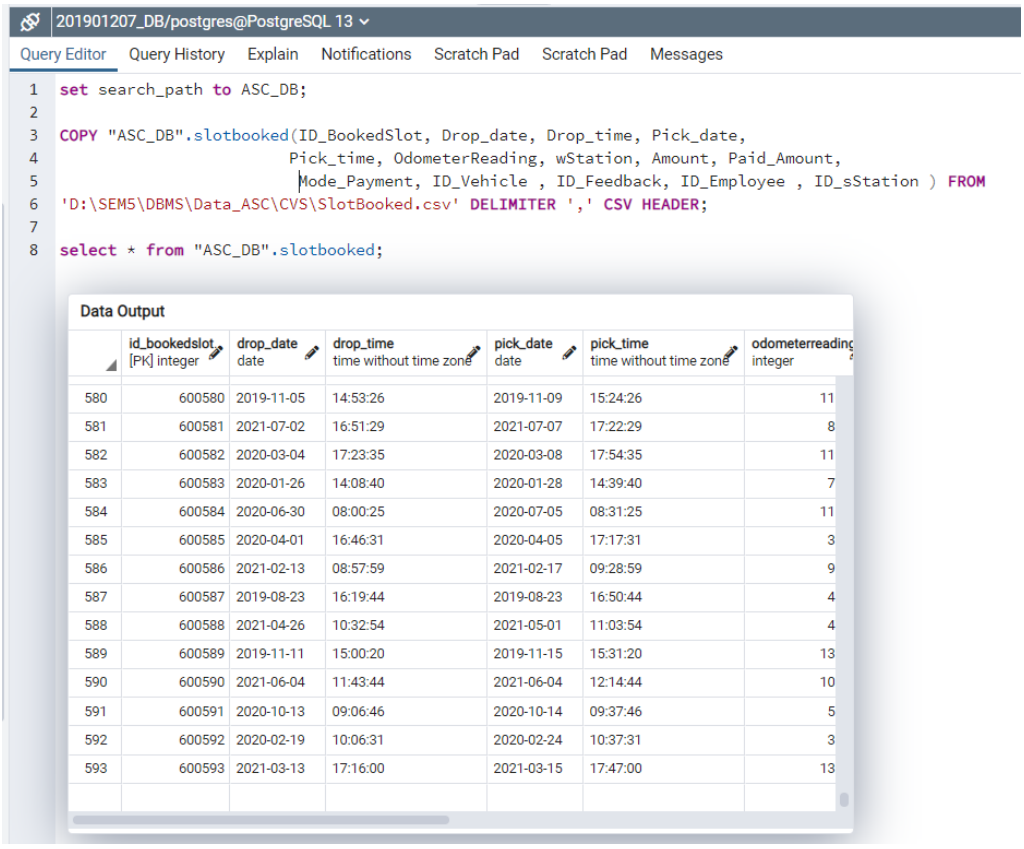
	id_vehicle [PK] integer	company text	vehicle_model text	vehicle_type text	plat_statecode text	plat_distcode integer	plat_codenumb text
267	11267	Hyundai	Verna	Sedan	GJ	38	AA 92
268	11268	Maruti Suzuki	Sonet	SUV	GJ	10	GW 1335
269	11269	Honda	XL6	MPV	GJ	10	BX 5446
270	11270	Mahindra	Carnival	MPV	GJ	1	JY 3153
271	11271	Tata	Creta	SUV	GJ	4	JZ 564
272	11272	Maruti Suzuki	Verna	Sedan	GJ	4	CY 7131
273	11273	Honda	Ferrari 812	Cabriolet	GJ	4	HX 9892
274	11274	Maruti Suzuki	S660	Roadster	GJ	5	HF 2088
275	11275	Honda	S-Cross	Crossover	GJ	11	CL 9870
276	11276	Hyundai	Scorpio	Minivan	GJ	11	GQ 9091
277	11277	Honda	Audi S5	Coupe	GJ	11	FK 9016
278	11278	Honda	Bolero	Minivan	GJ	11	KI 7667
279	11279	Tata	Nexon	SUV	GJ	6	JK 535
280	11280	Honda	Winger	Van	GJ	6	AJ 1654

- **Feedback (**ID_Feedback, Star, Feed_text**)**



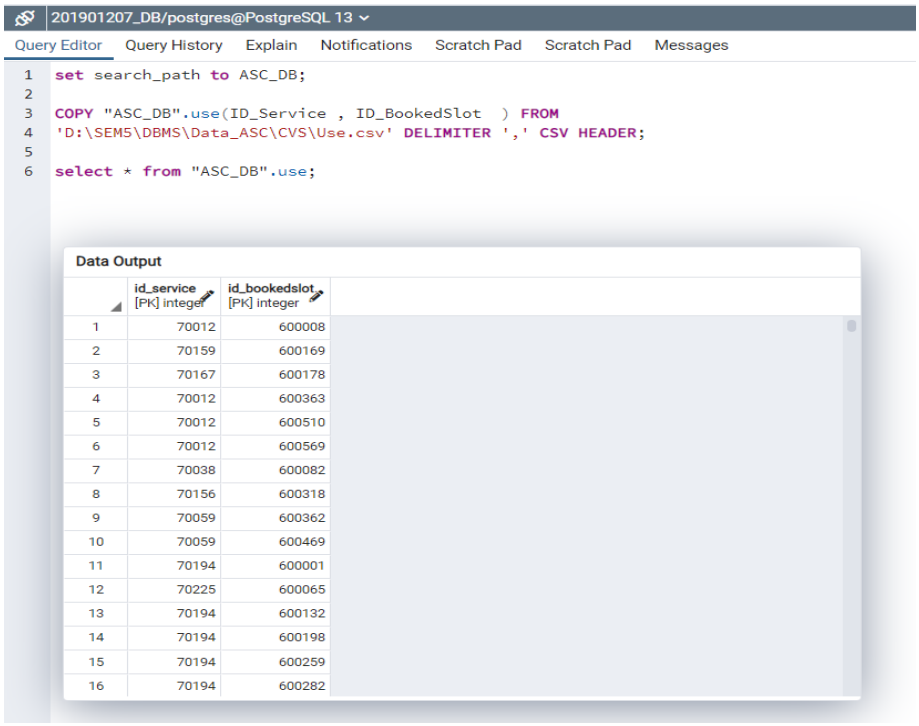
	id_feedback [PK] integer	star integer	feed_text text
579	8000579	2	Average service experience. Need to improve littele bit.
580	8000580	2	Average service experience. Need to improve littele bit.
581	8000581	2	Average service experience. Need to improve littele bit.
582	8000582	5	Excelent service. Good staff. Polite.I will certainly visit again and recommend it.
583	8000583	5	Excelent service. Good staff. Polite.I will certainly visit again and recommend it.
584	8000584	5	Excelent service. Good staff. Polite.I will certainly visit again and recommend it.
585	8000585	3	Good service experience. Recommendable.
586	8000586	2	Average service experience. Need to improve littele bit.
587	8000587	3	Good service experience. Recommendable.
588	8000588	3	Good service experience. Recommendable.
589	8000589	3	Good service experience. Recommendable.
590	8000590	2	Average service experience. Need to improve littele bit.
591	8000591	2	Average service experience. Need to improve littele bit.
592	8000592	3	Good service experience. Recommendable.
593	8000593	2	Average service experience. Need to improve littele bit.

- **SlotBooked (**ID_BookedSlot, Drop_date, Drop_time, Pick_date, Pick_time, OdometerReading, wStation, Amount, Paid_Amount, Mode_Payment, ID_Vehicle (FK), ID_Feedback (FK), ID_Employee (FK), ID_sStation (FK)**)**



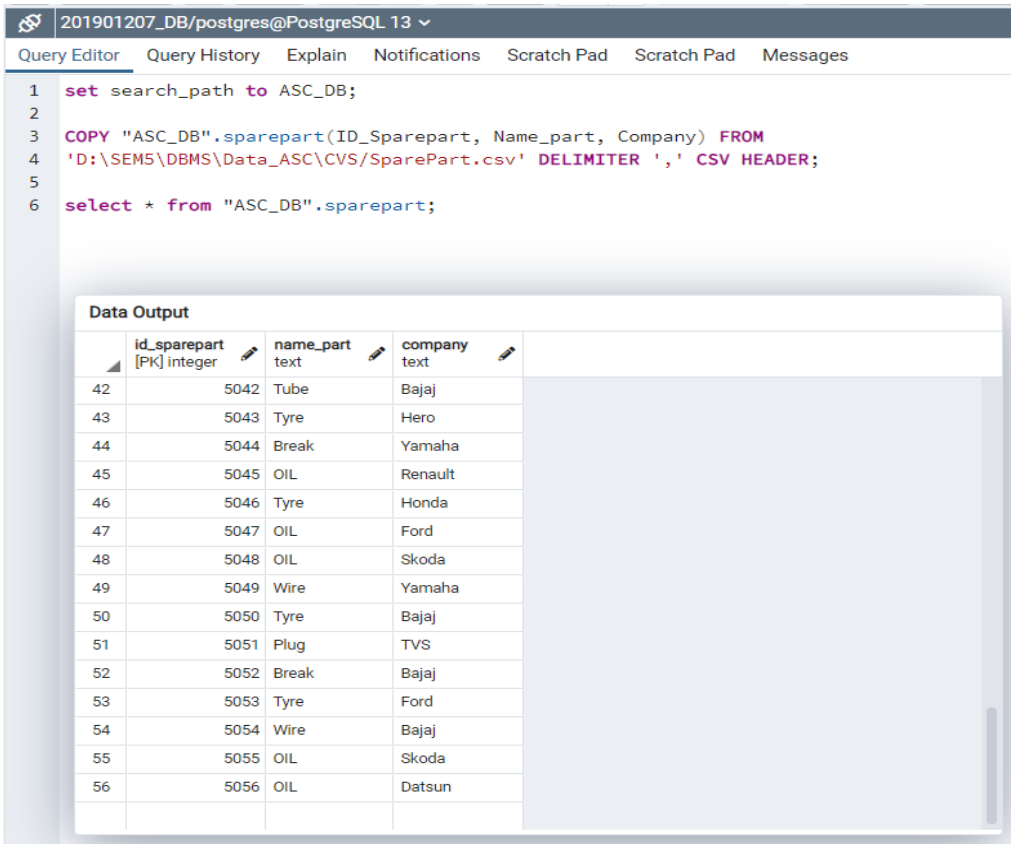
	id_bookedslot [PK] integer	drop_date date	drop_time time without time zone	pick_date date	pick_time time without time zone	odometerreading integer
580	600580	2019-11-05	14:53:26	2019-11-09	15:24:26	11
581	600581	2021-07-02	16:51:29	2021-07-07	17:22:29	8
582	600582	2020-03-04	17:23:35	2020-03-08	17:54:35	11
583	600583	2020-01-26	14:08:40	2020-01-28	14:39:40	7
584	600584	2020-06-30	08:00:25	2020-07-05	08:31:25	11
585	600585	2020-04-01	16:46:31	2020-04-05	17:17:31	3
586	600586	2021-02-13	08:57:59	2021-02-17	09:28:59	9
587	600587	2019-08-23	16:19:44	2019-08-23	16:50:44	4
588	600588	2021-04-26	10:32:54	2021-05-01	11:03:54	4
589	600589	2019-11-11	15:00:20	2019-11-15	15:31:20	13
590	600590	2021-06-04	11:43:44	2021-06-04	12:14:44	10
591	600591	2020-10-13	09:06:46	2020-10-14	09:37:46	5
592	600592	2020-02-19	10:06:31	2020-02-24	10:37:31	3
593	600593	2021-03-13	17:16:00	2021-03-15	17:47:00	13

- **Use (**ID_Service (FK), ID_BookedSlot (FK)**)**



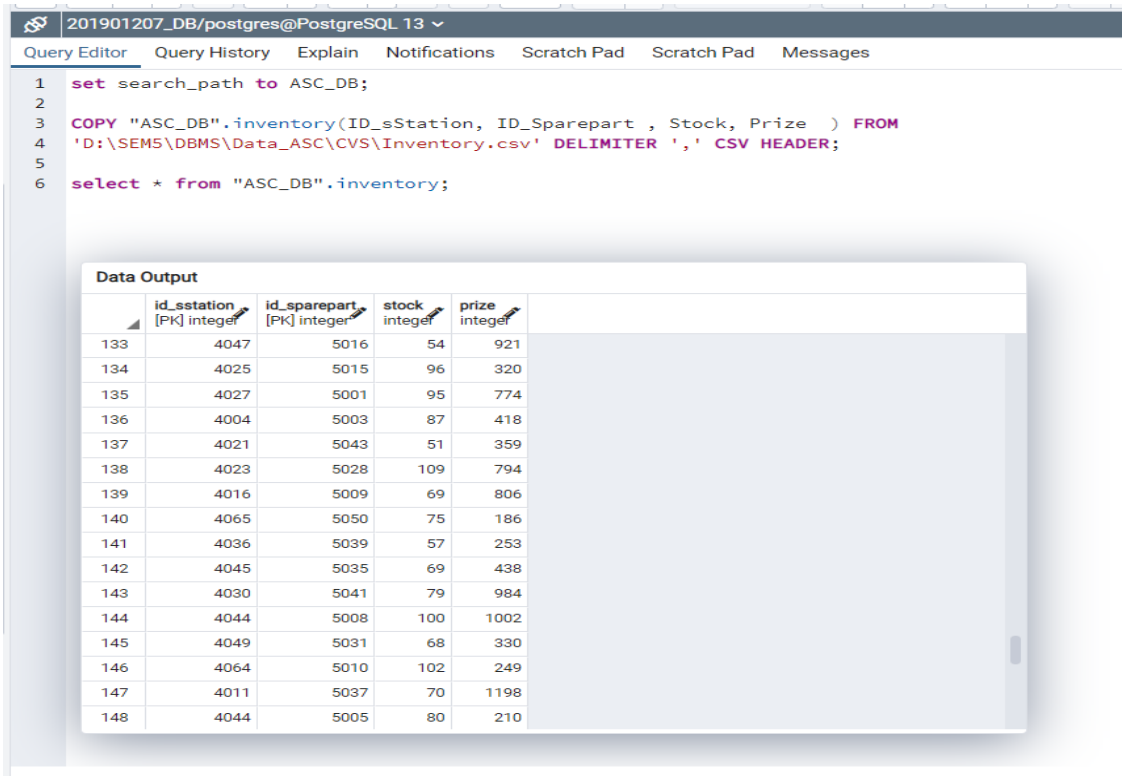
	id_service [PK] integer	id_bookedslot [PK] integer
1	70012	600008
2	70159	600169
3	70167	600178
4	70012	600363
5	70012	600510
6	70012	600569
7	70038	600082
8	70156	600318
9	70059	600362
10	70059	600469
11	70194	600001
12	70225	600065
13	70194	600132
14	70194	600198
15	70194	600259
16	70194	600282

- **SparePart (**ID_Spareparts, Name_part, Company**)**



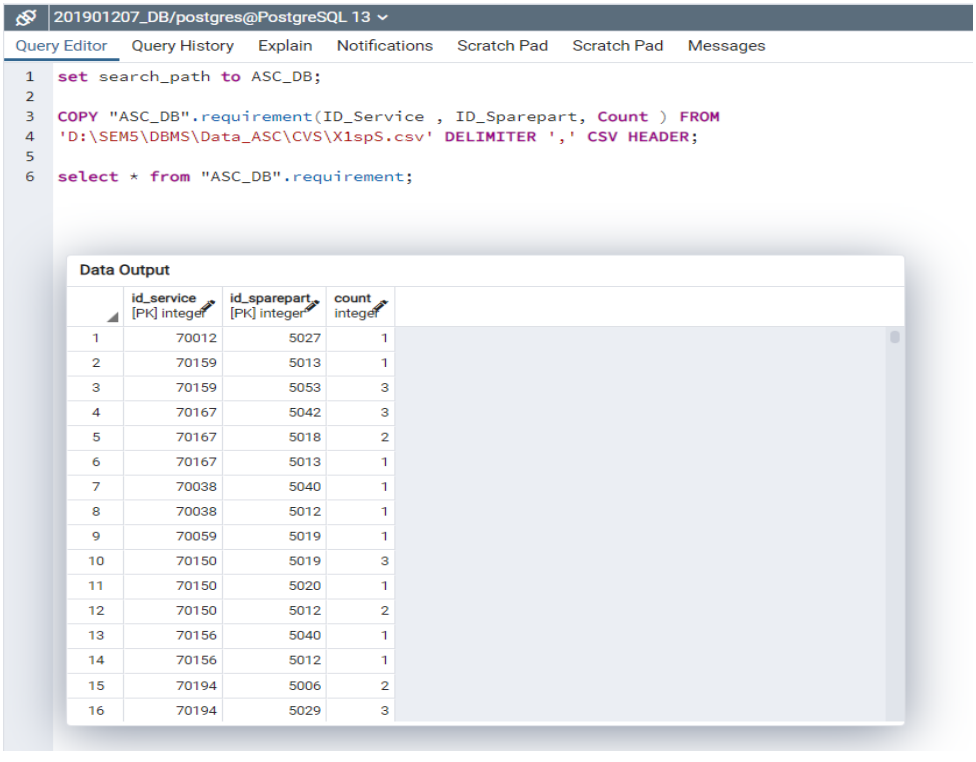
	id_sparepart [PK] integer	name_part text	company text
42	5042	Tube	Bajaj
43	5043	Tyre	Hero
44	5044	Break	Yamaha
45	5045	OIL	Renault
46	5046	Tyre	Honda
47	5047	OIL	Ford
48	5048	OIL	Skoda
49	5049	Wire	Yamaha
50	5050	Tyre	Bajaj
51	5051	Plug	TVS
52	5052	Break	Bajaj
53	5053	Tyre	Ford
54	5054	Wire	Bajaj
55	5055	OIL	Skoda
56	5056	OIL	Datsun

- **Inventory** (ID_sStation (FK), ID_Spareparts (FK), Stock, Prize**)**



	id_sstation [PK] integer	id_sparepart [PK] integer	stock integer	prize integer
133	4047	5016	54	921
134	4025	5015	96	320
135	4027	5001	95	774
136	4004	5003	87	418
137	4021	5043	51	359
138	4023	5028	109	794
139	4016	5009	69	806
140	4065	5050	75	186
141	4036	5039	57	253
142	4045	5035	69	438
143	4030	5041	79	984
144	4044	5008	100	1002
145	4049	5031	68	330
146	4064	5010	102	249
147	4011	5037	70	1198
148	4044	5005	80	210

- **Requirement (**ID_Service (FK), ID_Sparepart (FK), Count**)**



	id_service [PK] integer	id_sparepart [PK] integer	count integer
1	70012	5027	1
2	70159	5013	1
3	70159	5053	3
4	70167	5042	3
5	70167	5018	2
6	70167	5013	1
7	70038	5040	1
8	70038	5012	1
9	70059	5019	1
10	70150	5019	3
11	70150	5020	1
12	70150	5012	2
13	70156	5040	1
14	70156	5012	1
15	70194	5006	2
16	70194	5029	3

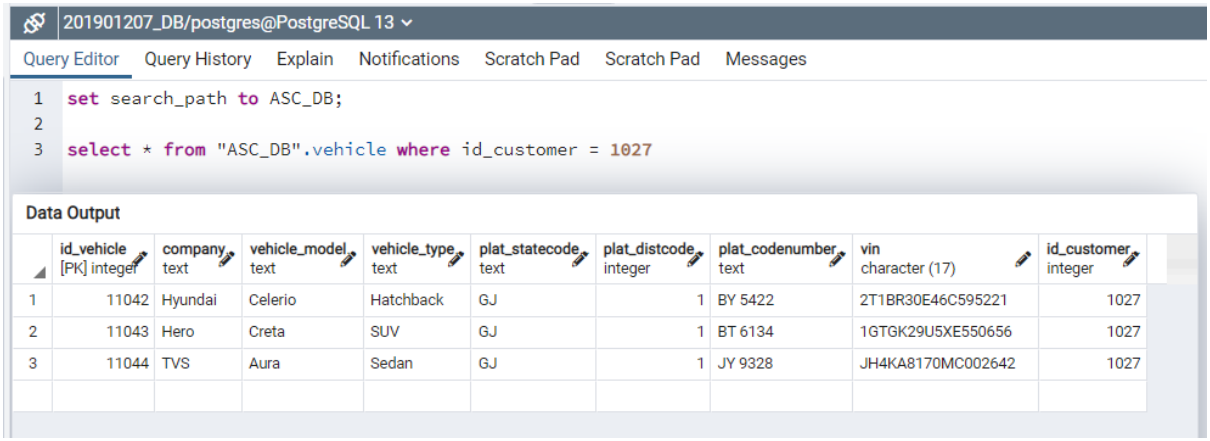
4. SQL Query:

1. **Retrieve all the detail of vehicles own by customer with ID 1027.**

SQL Query:

```
select * from "ASC_DB".vehicle where id_customer = 1027
```

No. of Tuple: 3
Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select * from "ASC_DB".vehicle where id_customer = 1027
```

Data Output

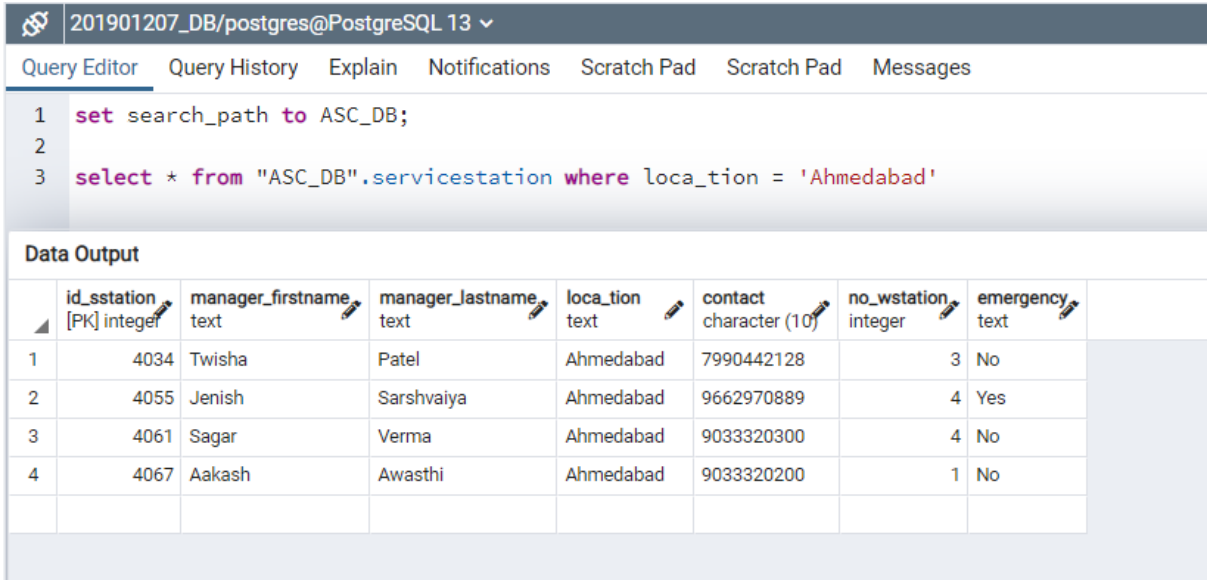
	id_vehicle [PK] integer	company text	vehicle_model text	vehicle_type text	plat_statecode text	plat_distcode integer	plat_codenumber text	vin character (17)	id_customer integer
1	11042	Hyundai	Celerio	Hatchback	GJ	1	BY 5422	2T1BR30E46C595221	1027
2	11043	Hero	Creta	SUV	GJ	1	BT 6134	1GTGK29U5XE550656	1027
3	11044	TVS	Aura	Sedan	GJ	1	JY 9328	JH4KA8170MC002642	1027

2. **Retrieve all the information about Service station located in Amedabad city.**

SQL Query:

```
select * from "ASC_DB".servicestation where loca_tion = 'Ahmedabad'
```

No. of Tuple: 4
Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select * from "ASC_DB".servicestation where loca_tion = 'Ahmedabad'
```

Data Output

	id_sstation [PK] integer	manager_firstname text	manager_lastname text	loca_tion text	contact character (10)	no_wstation integer	emergency text
1	4034	Twisha	Patel	Ahmedabad	7990442128	3	No
2	4055	Jenish	Sarshvaiya	Ahmedabad	9662970889	4	Yes
3	4061	Sagar	Verma	Ahmedabad	9033320300	4	No
4	4067	Aakash	Awasthi	Ahmedabad	9033320200	1	No

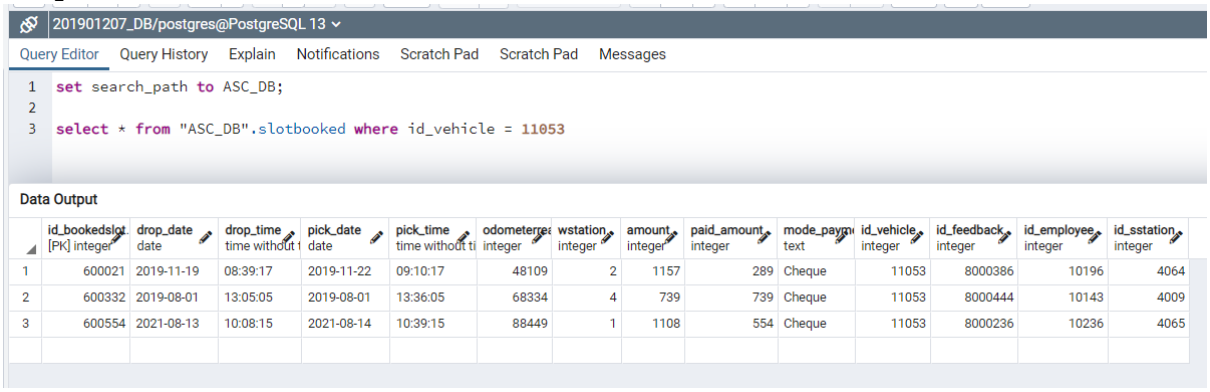
3. **Retrieve all the information about service booked for the vehicle with ID 11053**

SQL Query:

```
select * from "ASC_DB".slotbooked where id_vehicle = 11053
```

No. of Tuple: 3

Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select * from "ASC_DB".slotbooked where id_vehicle = 11053
```

Data Output

	id_bookedslot [PK] integer	drop_date date	drop_time time without t	pick_date date	pick_time time without ti	odometerrea integer	wstation integer	amount integer	paid_amount integer	mode_paym text	id_vehicle integer	id_feedback integer	id_employee integer	id_sstation integer
1	600021	2019-11-19	08:39:17	2019-11-22	09:10:17	48109	2	1157	289	Cheque	11053	8000386	10196	4064
2	600332	2019-08-01	13:05:05	2019-08-01	13:36:05	68334	4	739	739	Cheque	11053	8000444	10143	4009
3	600554	2021-08-13	10:08:15	2021-08-14	10:39:15	88449	1	1108	554	Cheque	11053	8000236	10236	4065

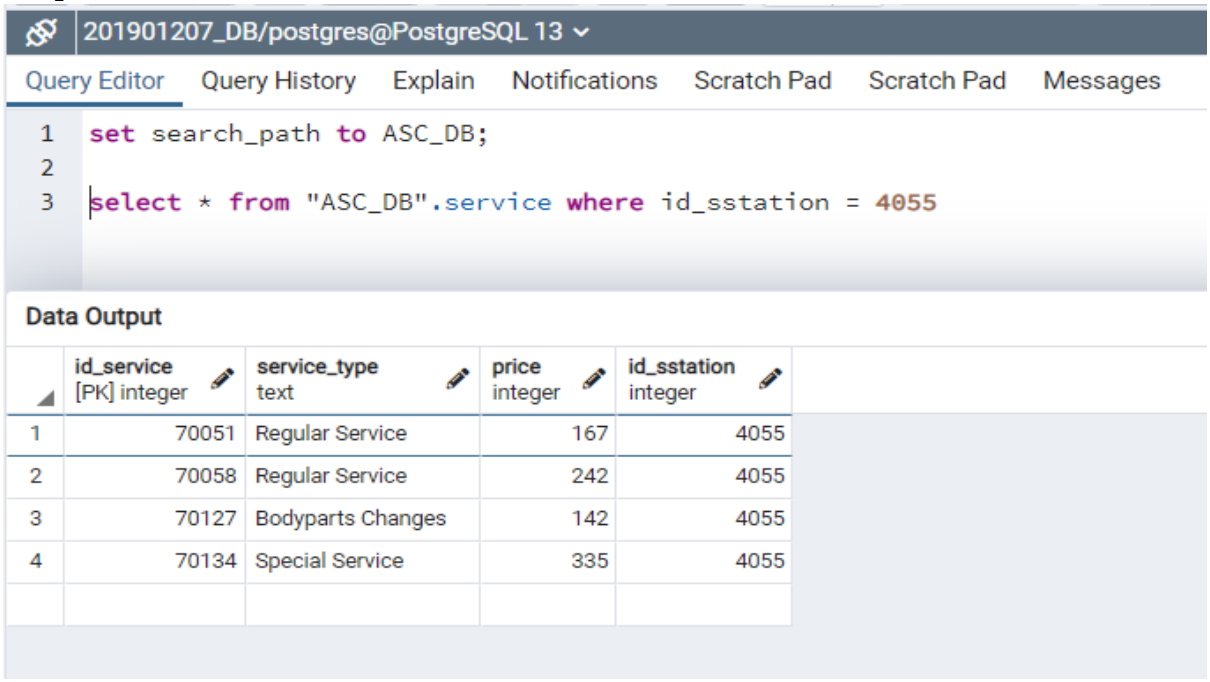
4. **List detail of all the service provided by the service station with ID 4044**

SQL Query:

```
select * from "ASC_DB".service where id_sstation = 4055
```

No. of Tuple: 4

Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select * from "ASC_DB".service where id_sstation = 4055
```

Data Output

	id_service [PK] integer	service_type text	price integer	id_sstation integer
1	70051	Regular Service	167	4055
2	70058	Regular Service	242	4055
3	70127	Bodyparts Changes	142	4055
4	70134	Special Service	335	4055

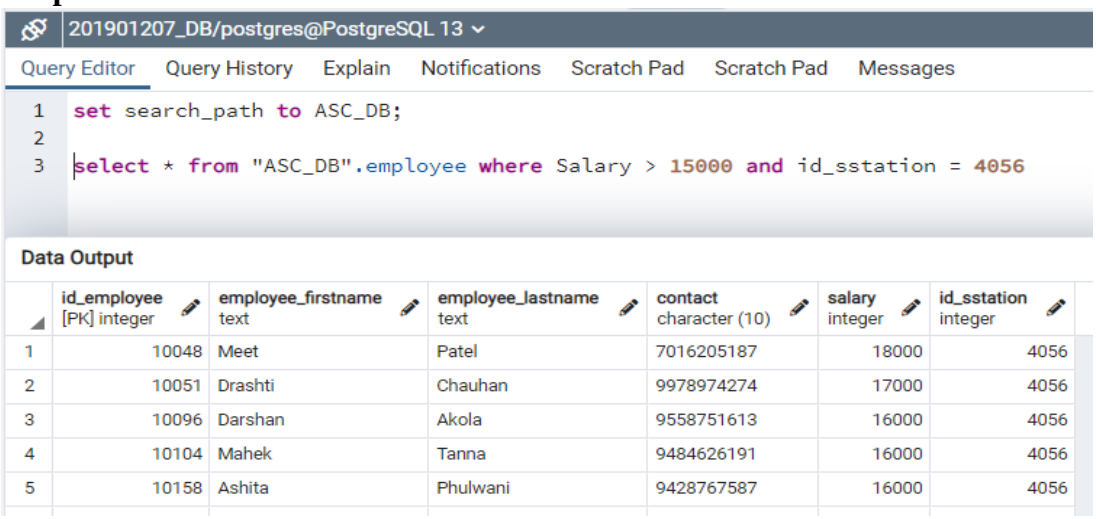
5. **List all the employees of service station with ID 4056 whose salary is more than 15,000**

SQL Query:

```
select * from "ASC_DB".employee where Salary > 15000 and id_sstation = 4056
```

No. of Tuple: 5

Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select * from "ASC_DB".employee where Salary > 15000 and id_sstation = 4056
```

Data Output

	id_employee [PK] integer	employee_firstname text	employee_lastname text	contact character (10)	salary integer	id_sstation integer
1	10048	Meet	Patel	7016205187	18000	4056
2	10051	Drashti	Chauhan	9978974274	17000	4056
3	10096	Darshan	Akola	9558751613	16000	4056
4	10104	Mahek	Tanna	9484626191	16000	4056
5	10158	Ashita	Phulwani	9428767587	16000	4056

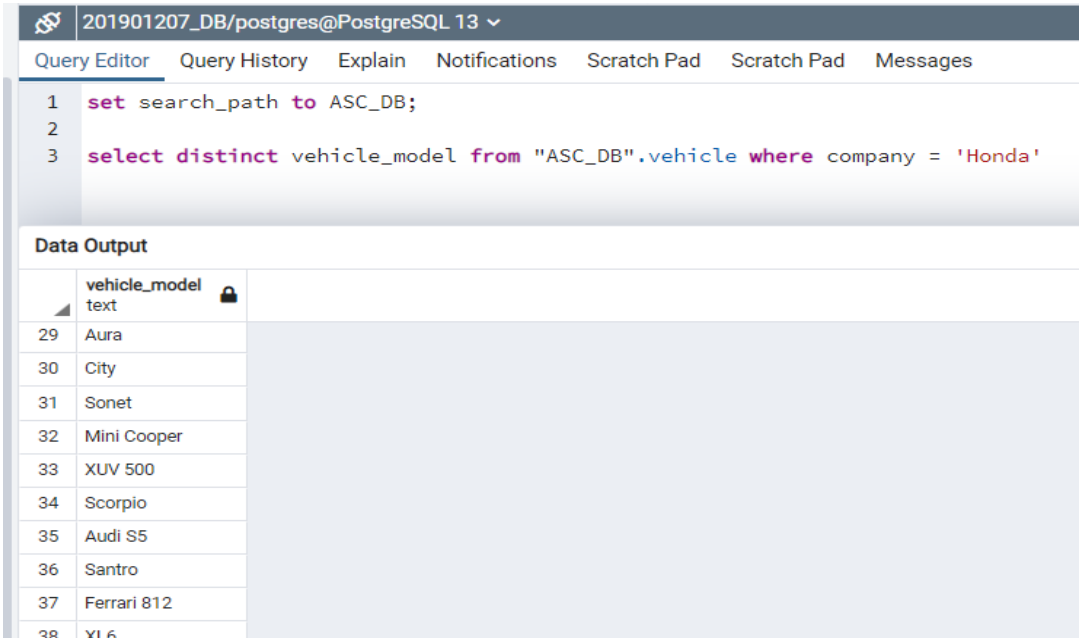
6. **List different model of vehicle from ‘Honda’ company.**

SQL Query:

```
select distinct vehicle_model from "ASC_DB".vehicle
where company = 'Honda'
```

No. of Tuple: 44

Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select distinct vehicle_model from "ASC_DB".vehicle where company = 'Honda'
```

Data Output

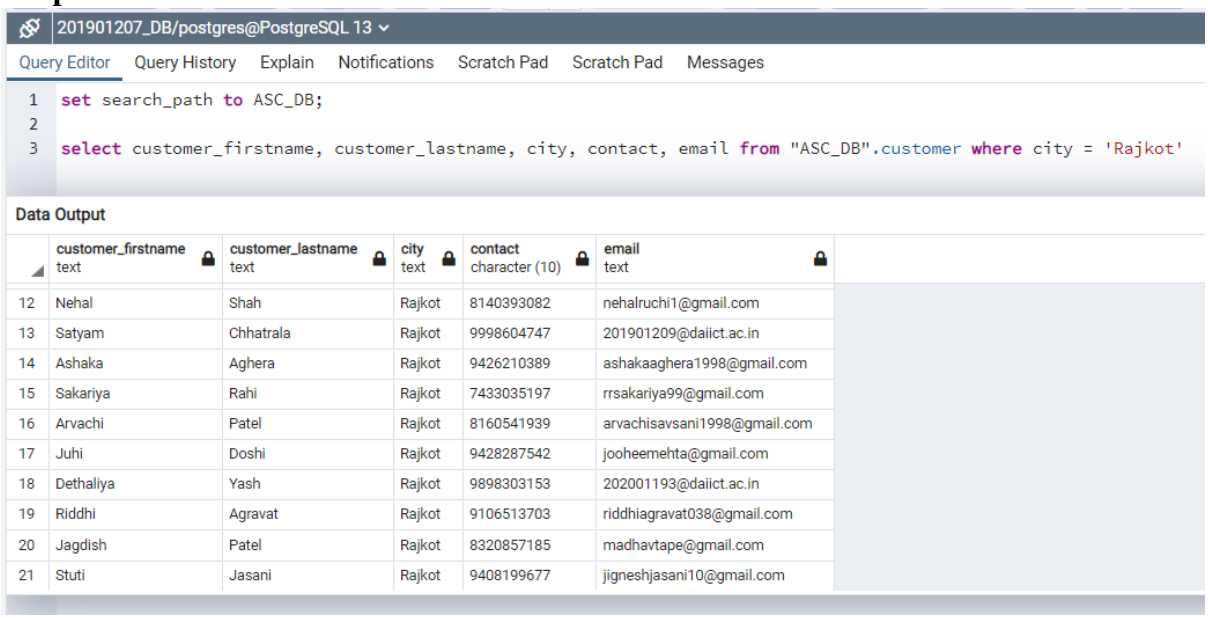
	vehicle_model text
29	Aura
30	City
31	Sonet
32	Mini Cooper
33	XUV 500
34	Scorpio
35	Audi S5
36	Santro
37	Ferrari 812
38	XL6

7. **List the first name, last name, city, contact and email of the customer from city Rajkot**

SQL Query:
```
select customer_firstname, customer_lastname, city, contact, email
    from "ASC_DB".customer where city = 'Rajkot'
```

No. of Tuple: 21

Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select customer_firstname, customer_lastname, city, contact, email from "ASC_DB".customer where city = 'Rajkot'
```

Data Output

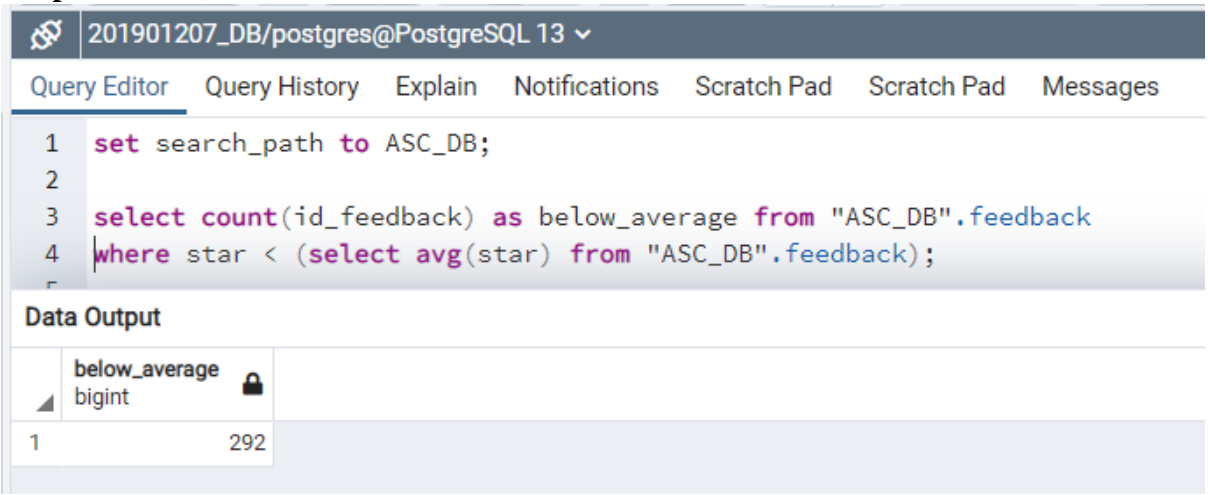
	customer_firstname text	customer_lastname text	city text	contact character (10)	email text
12	Nehal	Shah	Rajkot	8140393082	nehalruchi1@gmail.com
13	Satyam	Chhatrala	Rajkot	9998604747	201901209@daiict.ac.in
14	Ashaka	Aghera	Rajkot	9426210389	ashakaaghera1998@gmail.com
15	Sakariya	Rahi	Rajkot	7433035197	rrsakariya99@gmail.com
16	Arvachi	Patel	Rajkot	8160541939	arvachisavsani1998@gmail.com
17	Juhi	Doshi	Rajkot	9428287542	jooheemehta@gmail.com
18	Dethaliya	Yash	Rajkot	9898303153	202001193@daiict.ac.in
19	Riddhi	Agravat	Rajkot	9106513703	riddhiagravat038@gmail.com
20	Jagdish	Patel	Rajkot	8320857185	madhavtape@gmail.com
21	Stuti	Jasani	Rajkot	9408199677	jigneshjasani10@gmail.com

8. **Find the number of feedback with rating lower than average rating to the service.**

SQL Query:
```
select count(id_feedback) as below_average
    from "ASC_DB".feedback
    where star < (select avg(star) from "ASC_DB".feedback);
```

No. of Tuple: 1

Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select count(id_feedback) as below_average from "ASC_DB".feedback
4  where star < (select avg(star) from "ASC_DB".feedback);
```

Data Output

	below_average bigint
1	292

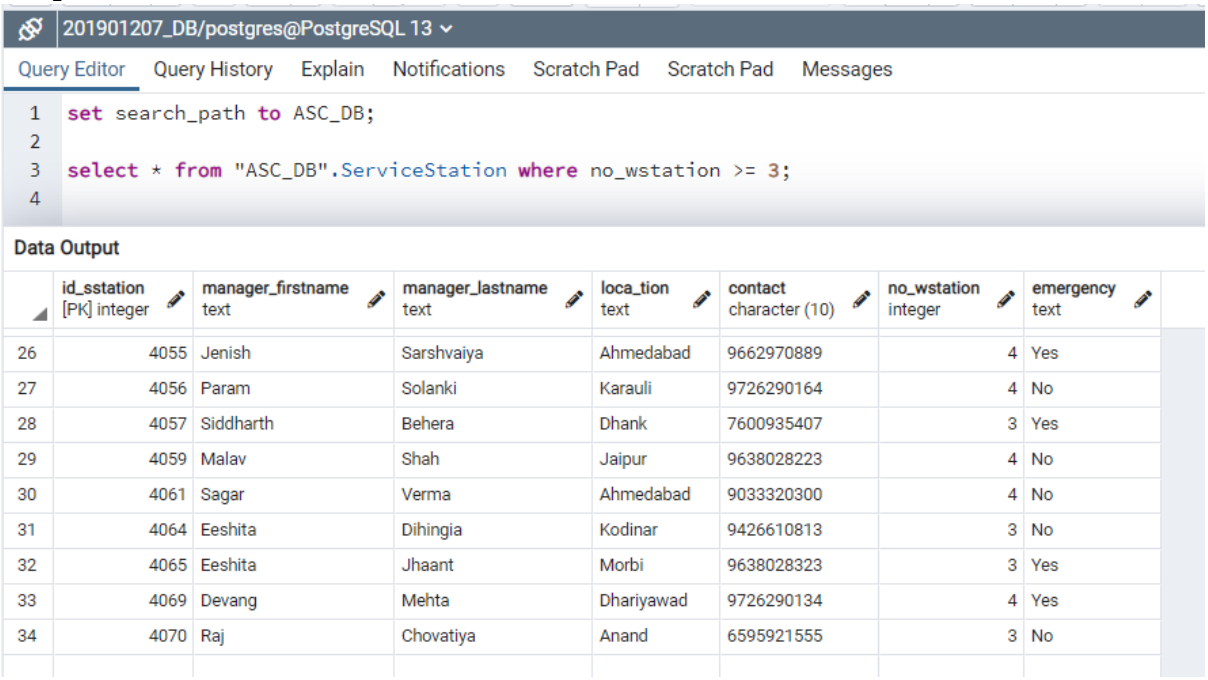
9. **Find the details about service stations where the number of workstations is more than or equal to three.**

SQL Query:

```
select * from "ASC_DB".ServiceStation
where no_wstation >= 3;
```

No. of Tuple: 34

Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select * from "ASC_DB".ServiceStation where no_wstation >= 3;
4
```

Data Output

	id_sstation [PK] integer	manager_firstname text	manager_lastname text	loca_tion text	contact character (10)	no_wstation integer	emergency text
26	4055	Jenish	Sarshvaiya	Ahmedabad	9662970889	4	Yes
27	4056	Param	Solanki	Karauli	9726290164	4	No
28	4057	Siddharth	Behera	Dhank	7600935407	3	Yes
29	4059	Malav	Shah	Jaipur	9638028223	4	No
30	4061	Sagar	Verma	Ahmedabad	9033320300	4	No
31	4064	Eeshita	Dihingia	Kodinar	9426610813	3	No
32	4065	Eeshita	Jhaant	Morbi	9638028323	3	Yes
33	4069	Devang	Mehta	Dhariyawad	9726290134	4	Yes
34	4070	Raj	Chovatiya	Anand	6595921555	3	No

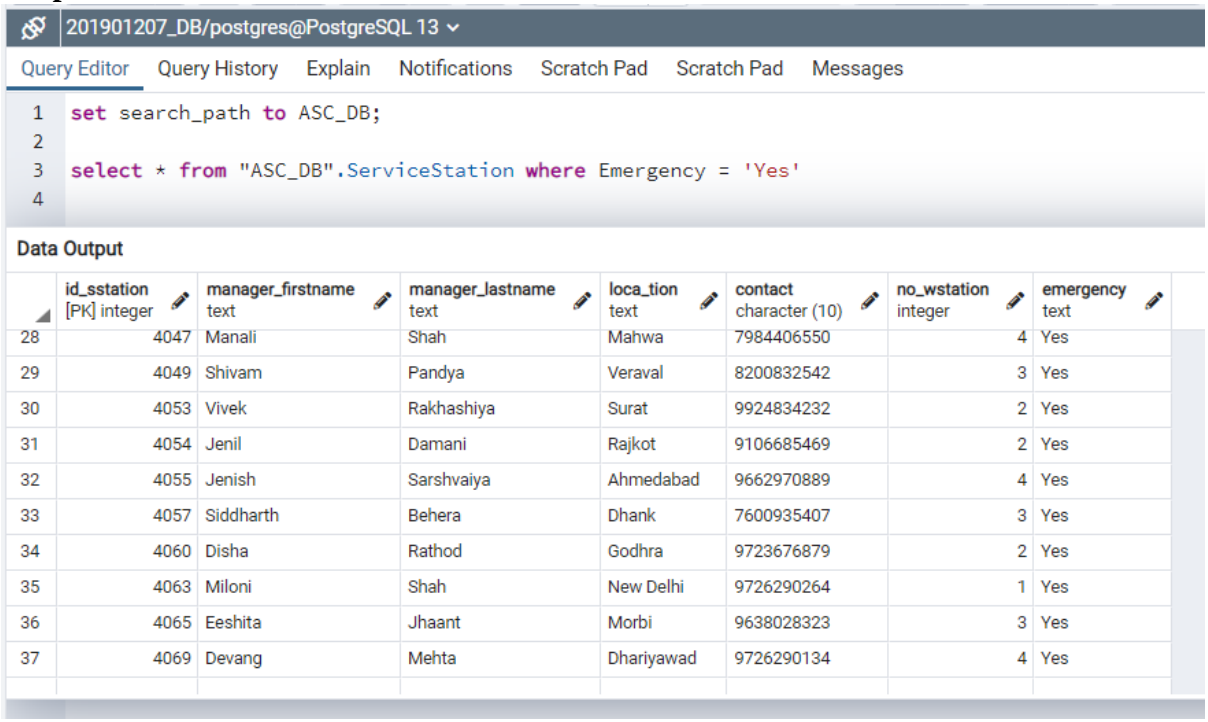
10. **Find all the details about the service stations which offers emergency services.**

SQL Query:

```
select * from "ASC_DB".ServiceStation
where Emergency = 'Yes'
```

No. of Tuple: 37

Snapshot:



```
set search_path to ASC_DB;

select * from "ASC_DB".ServiceStation where Emergency = 'Yes'
```

Data Output

	id_sstation [PK] integer	manager_firstname text	manager_lastname text	loca_tion text	contact character (10)	no_wstation integer	emergency text
28	4047	Manali	Shah	Mahwa	7984406550	4	Yes
29	4049	Shivam	Pandya	Veraval	8200832542	3	Yes
30	4053	Vivek	Rakhashiya	Surat	9924834232	2	Yes
31	4054	Jenil	Damani	Rajkot	9106685469	2	Yes
32	4055	Jenish	Sarshvaiya	Ahmedabad	9662970889	4	Yes
33	4057	Siddharth	Behera	Dhank	7600935407	3	Yes
34	4060	Disha	Rathod	Godhra	9723676879	2	Yes
35	4063	Miloni	Shah	New Delhi	9726290264	1	Yes
36	4065	Eeshita	Jhaant	Morbi	9638028323	3	Yes
37	4069	Devang	Mehta	Dhariyawad	9726290134	4	Yes

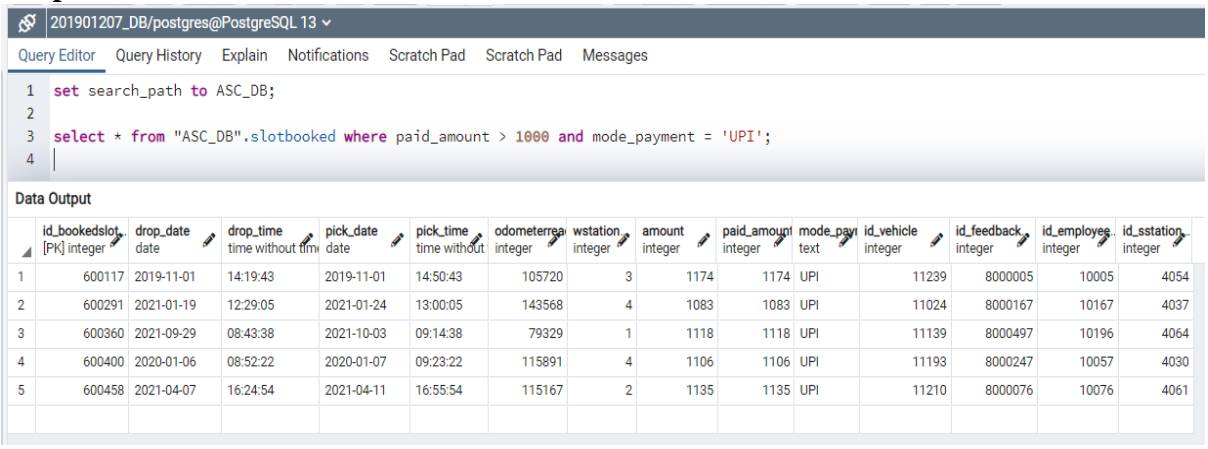
11. Find the information about slots booked with total amount to pay more than 1000 /- and whose payment is done via UPI.

SQL Query:

```
select * from "ASC_DB".slotbooked
where paid_amount > 1000 and mode_payment = 'UPI';
```

No. of Tuple: 5

Snapshot:



```
set search_path to ASC_DB;

select * from "ASC_DB".slotbooked where paid_amount > 1000 and mode_payment = 'UPI';
```

Data Output

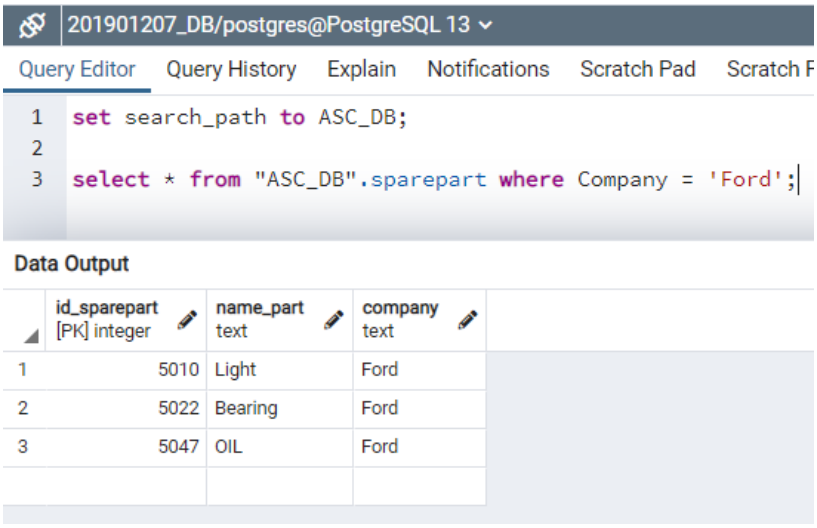
	id_bookedslot [PK] integer	drop_date date	drop_time time without tim	pick_date date	pick_time time without	odometerrea integer	wstation integer	amount integer	paid_amount integer	mode_pay text	id_vehicle integer	id_feedback integer	id_employee integer	id_sstation integer
1	600117	2019-11-01	14:19:43	2019-11-01	14:50:43	105720	3	1174	1174	UPI	11239	8000005	10005	4054
2	600291	2021-01-19	12:29:05	2021-01-24	13:00:05	143568	4	1083	1083	UPI	11024	8000167	10167	4037
3	600360	2021-09-29	08:43:38	2021-10-03	09:14:38	79329	1	1118	1118	UPI	11139	8000497	10196	4064
4	600400	2020-01-06	08:52:22	2020-01-07	09:23:22	115891	4	1106	1106	UPI	11193	8000247	10057	4030
5	600458	2021-04-07	16:24:54	2021-04-11	16:55:54	115167	2	1135	1135	UPI	11210	8000076	10076	4061

12. List all the details of distinct spare parts in the market of the company Ford.

SQL Query:

```
select * from "ASC_DB".sparepart where Company = 'Ford';
```

No. of Tuple: 3
Snapshot:

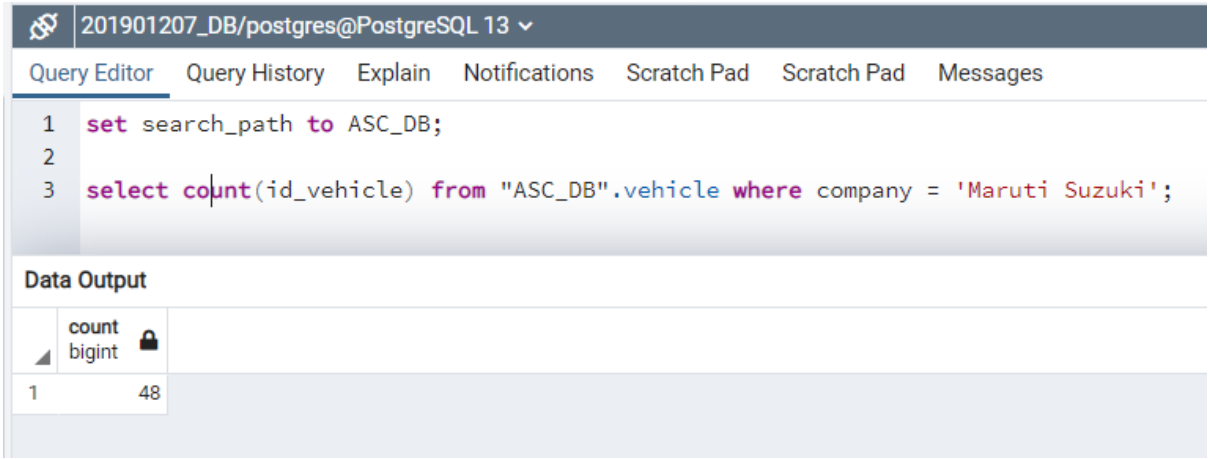


13. Find the total count of vehicles of Maruti Suzuki in the record.

SQL Query:

```
select count(id_vehicle) from "ASC_DB".vehicle where company = 'Maruti Suzuki';
```

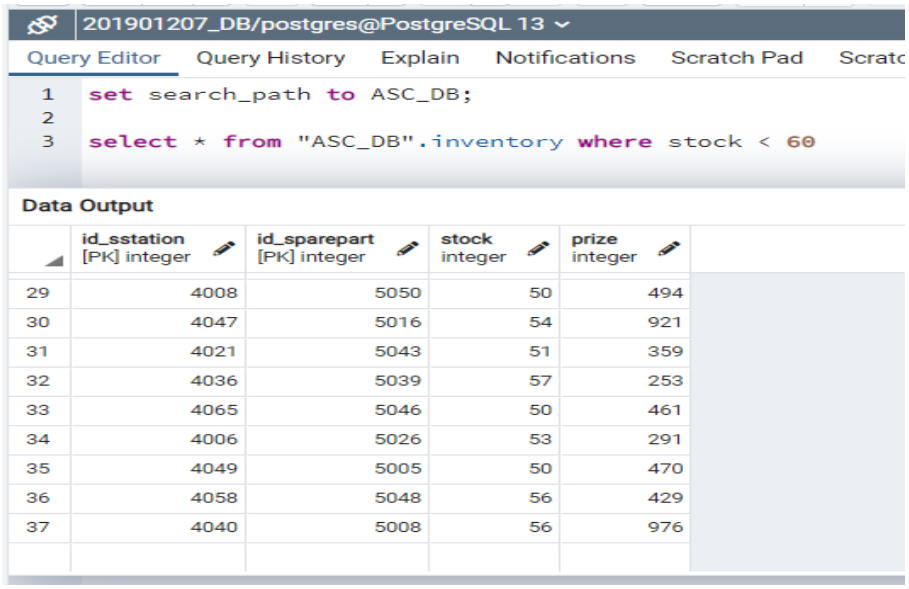
No. of Tuple: 1
Snapshot:



14. List the ID of Sparepart corresponding to all the service station with stock less than 60.

SQL Query:
select * from "ASC_DB".inventory where stock < 60

No. of Tuple: 37
Snapshot:

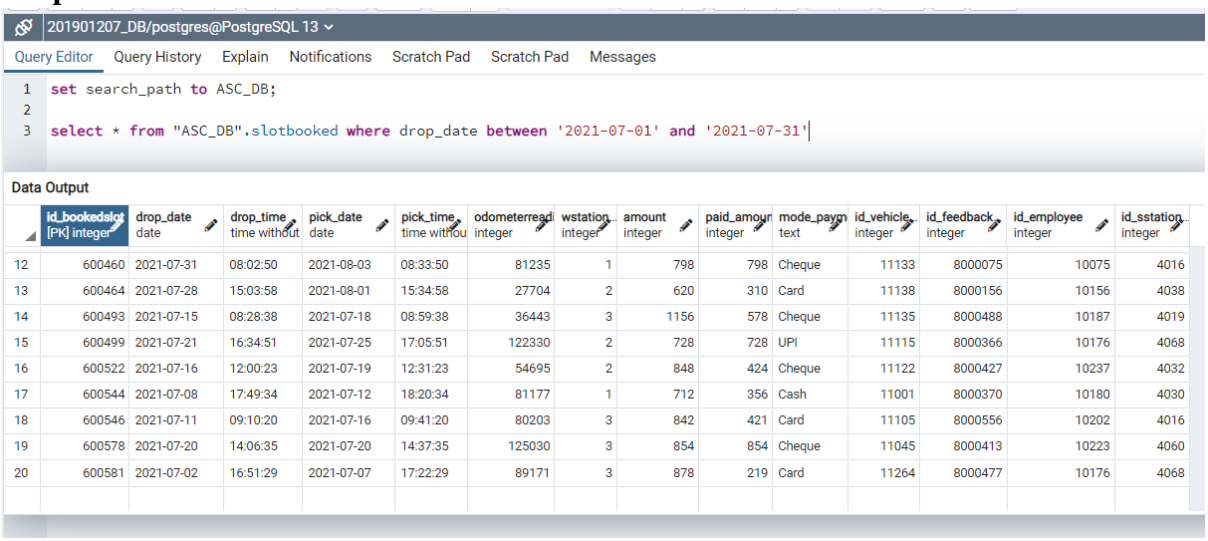


	id_sstation [PK] integer	id_sparepart [PK] integer	stock integer	prize integer
29	4008	5050	50	494
30	4047	5016	54	921
31	4021	5043	51	359
32	4036	5039	57	253
33	4065	5046	50	461
34	4006	5026	53	291
35	4049	5005	50	470
36	4058	5048	56	429
37	4040	5008	56	976

15. List all the details about slots booked in the month of July 2021.

SQL Query: select * from "ASC_DB".slotbooked
where drop_date between '2021-07-01' and '2021-07-31';

No. of Tuple: 20
Snapshot:



	id_bookedslot [PK] integer	drop_date date	drop_time time without	pick_date date	pick_time time withou	odometerread integer	wstation integer	amount integer	paid_amoun integer	mode_paym text	id_vehicle integer	id_feedback integer	id_employee integer	id_sstation integer
12	600460	2021-07-31	08:02:50	2021-08-03	08:33:50	81235	1	798	798	Cheque	11133	8000075	10075	4016
13	600464	2021-07-28	15:03:58	2021-08-01	15:34:58	27704	2	620	310	Card	11138	8000156	10156	4038
14	600493	2021-07-15	08:28:38	2021-07-18	08:59:38	36443	3	1156	578	Cheque	11135	8000488	10187	4019
15	600499	2021-07-21	16:34:51	2021-07-25	17:05:51	122330	2	728	728	UPI	11115	8000366	10176	4068
16	600522	2021-07-16	12:00:23	2021-07-19	12:31:23	54695	2	848	424	Cheque	11122	8000427	10237	4032
17	600544	2021-07-08	17:49:34	2021-07-12	18:20:34	81177	1	712	356	Cash	11001	8000370	10180	4030
18	600546	2021-07-11	09:10:20	2021-07-16	09:41:20	80203	3	842	421	Card	11105	8000556	10202	4016
19	600578	2021-07-20	14:06:35	2021-07-20	14:37:35	125030	3	854	854	Cheque	11045	8000413	10223	4060
20	600581	2021-07-02	16:51:29	2021-07-07	17:22:29	89171	3	878	219	Card	11264	8000477	10176	4068

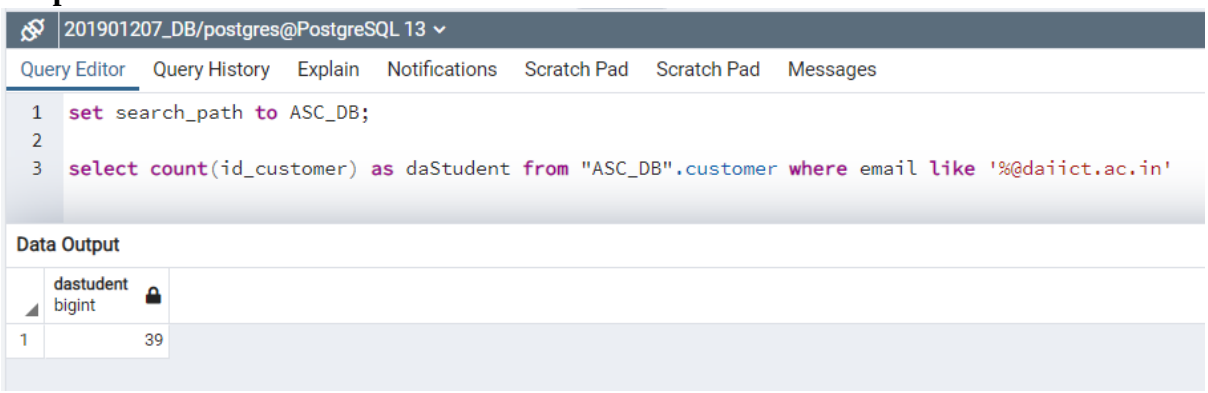
16. Find count of all the customer with email domain '@daiict.ac.in'

SQL Query:

```
select count(id_customer) as daStudent from "ASC_DB".customer
where email like '%@daiict.ac.in';
```

No. of Tuple: 1

Snapshot:



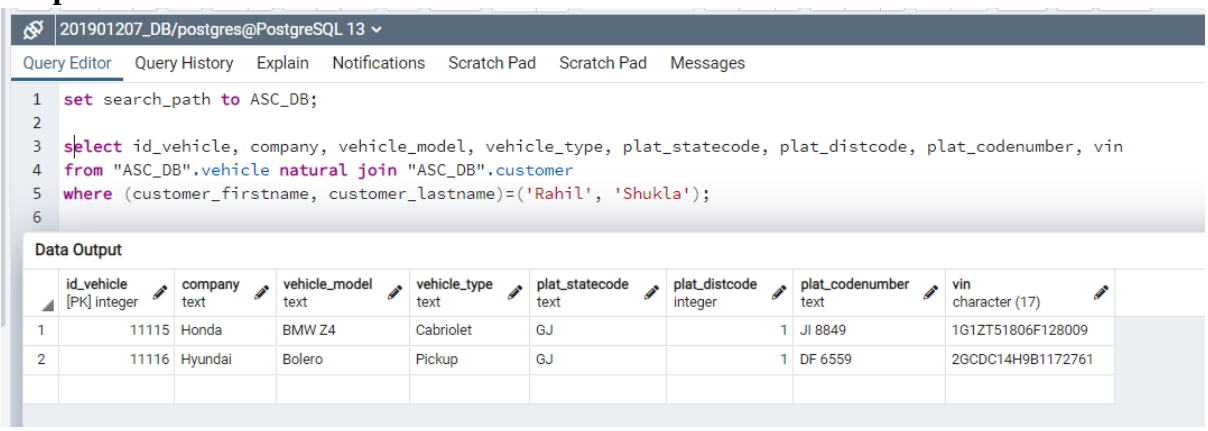
17. Vehicle details of Rahil Shukla

SQL Query:

```
select id_vehicle, company, vehicle_model, vehicle_type, plat_statecode,
plat_distcode, plat_codenumber, vin
from "ASC_DB".vehicle natural join "ASC_DB".customer
where (customer_firstname, customer_lastname) = ('Rahil', 'Shukla');
```

No. of Tuple: 2

Snapshot:



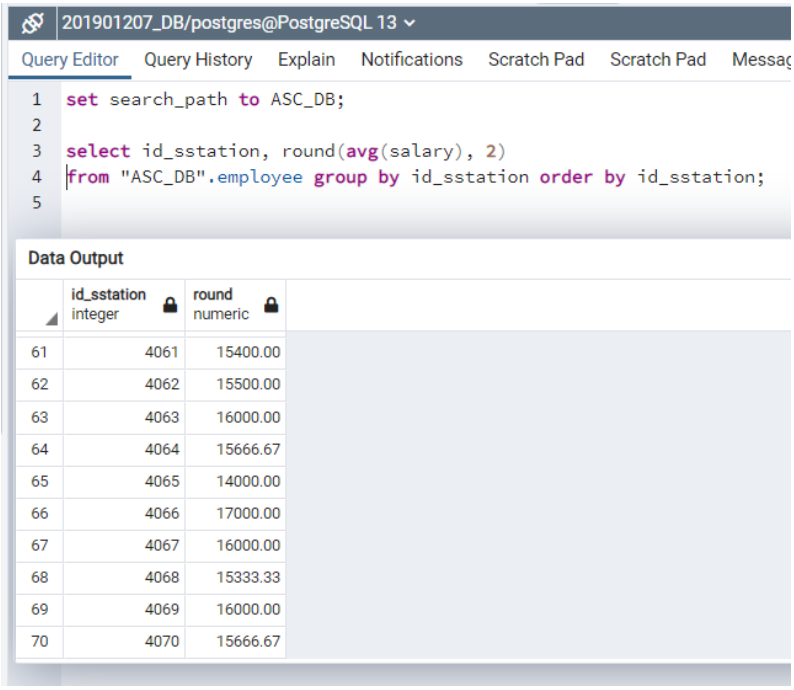
18. Find the average salary of all employees for all service stations.

SQL Query:

```
select id_sstation, round(avg(salary), 2)
from "ASC_DB".employee group by id_sstation order by id_sstation;
```

No. of Tuple: 70

Snapshot:



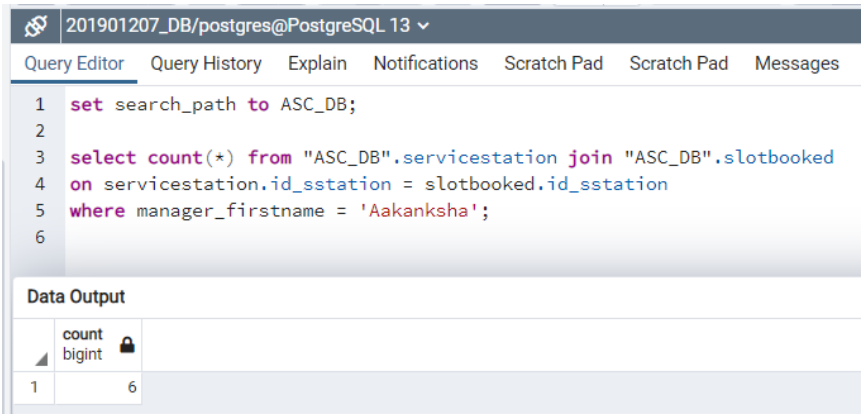
19. Find the total number of services done in the service station which is managed by Aakanksha.

SQL Query:

```
select count(*)
from "ASC_DB".servicestation join "ASC_DB".slotbooked
                on servicestation.id_sstation = slotbooked.id_sstation
where manager_firstname = 'Aakanksha';
```

No. of Tuple: 1

Snapshot:



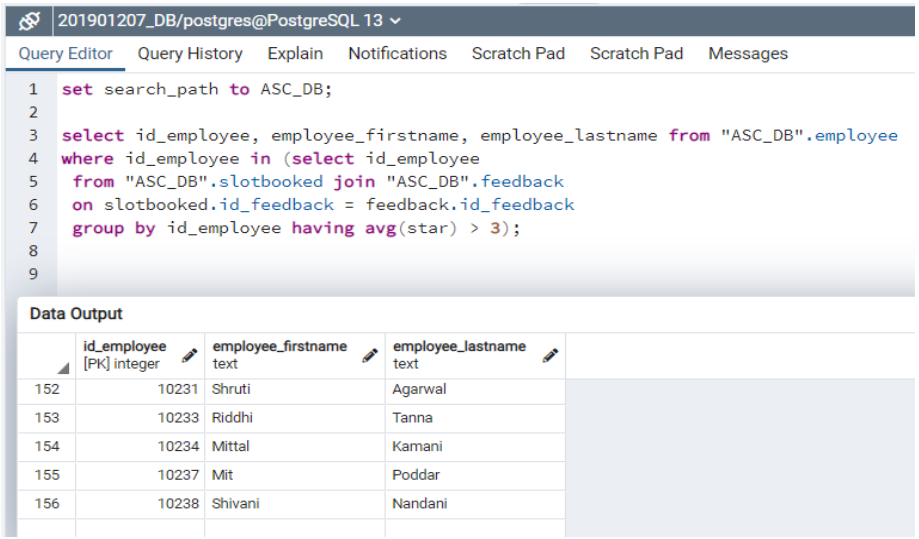
20. List down ID and names of all employees whose work has been rated for more than 3 on average stars in all the services done by him/her.

SQL Query:

```
select id_employee, employee_firstname, employee_lastname
from "ASC_DB".employee
where id_employee in (select id_employee
                      from "ASC_DB".slotbooked join "ASC_DB".feedback
                           on slotbooked.id_feedback = feedback.id_feedback
                      group by id_employee having avg(star) > 3);
```

No. of Tuple: 156

Snapshot:



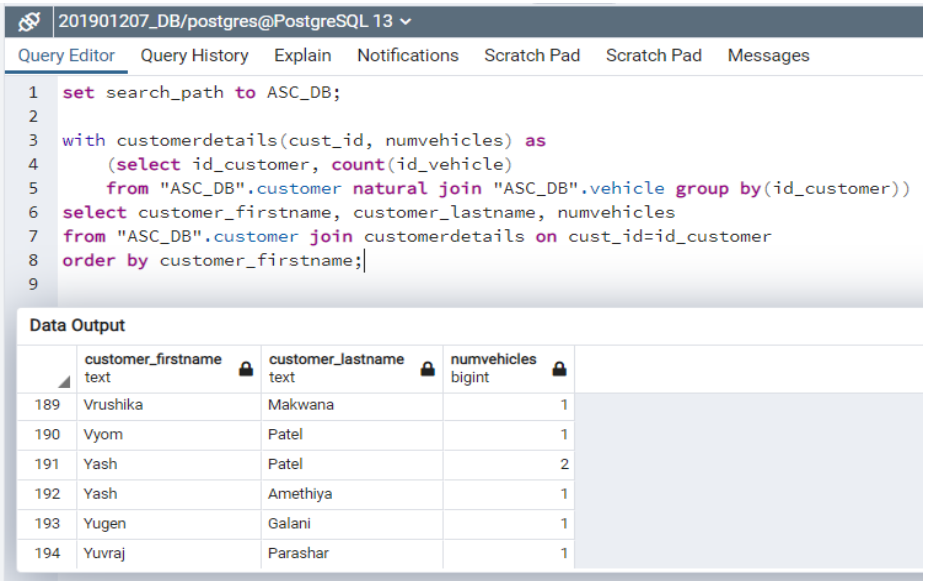
21. Display all customer names with the total number of vehicles they own.

SQL Query:

```
with customerdetails(cust_id, numvehicles)
        as (select id_customer, count(id_vehicle)
           from "ASC_DB".customer natural join "ASC_DB".vehicle
           group by(id_customer))
select customer_firstname, customer_lastname, numvehicles
from "ASC_DB".customer join customerdetails
        on cust_id=id_customer
order by customer_firstname;
```

No. of Tuple: 194

Snapshot:



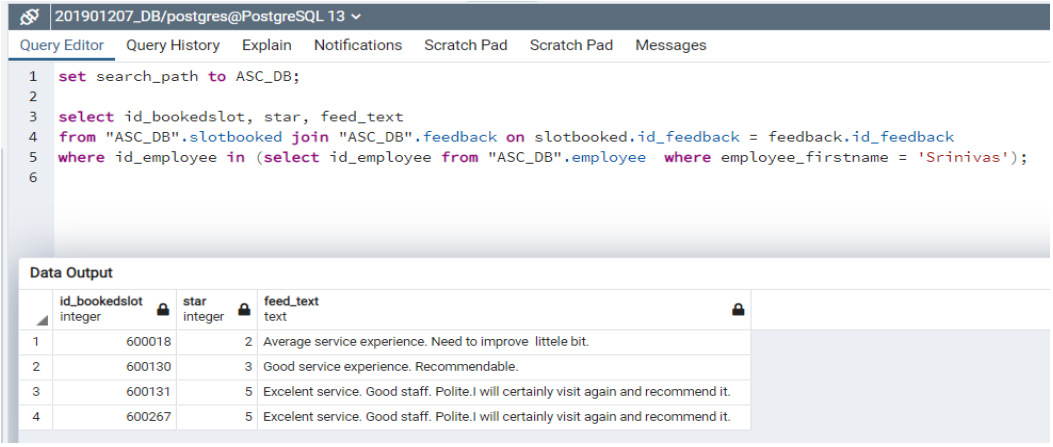
22. List the feedback for the services done by the employee with the first name Srinivas.

SQL Query:

```
select id_bookedslot, star, feed_text
from "ASC_DB".slotbooked join "ASC_DB".feedback
        on slotbooked.id_feedback = feedback.id_feedback
where id_employee in
                    (select id_employee
                    from "ASC_DB".employee
                    where employee_firstname = 'Srinivas');
```

No. of Tuple: 4

Snapshot:



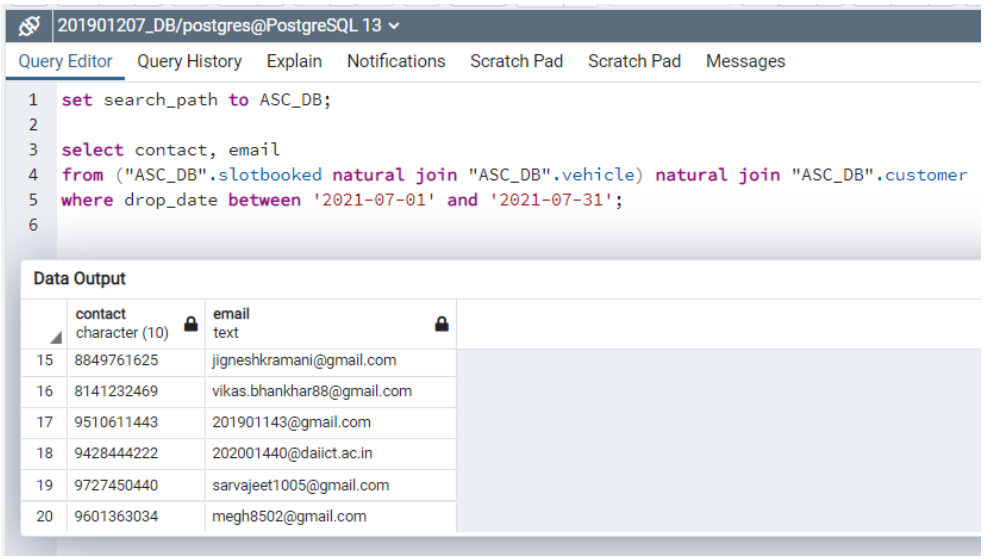
23. Contact information of the customers dropping their vehicle in the month of July 2021

SQL Query:

```
select contact, email
from ("ASC_DB".slotbooked natural join "ASC_DB".vehicle) natural join
"ASC_DB".customer
where drop_date between '2021-07-01' and '2021-07-31';
```

No. of Tuple: 20

Snapshot:



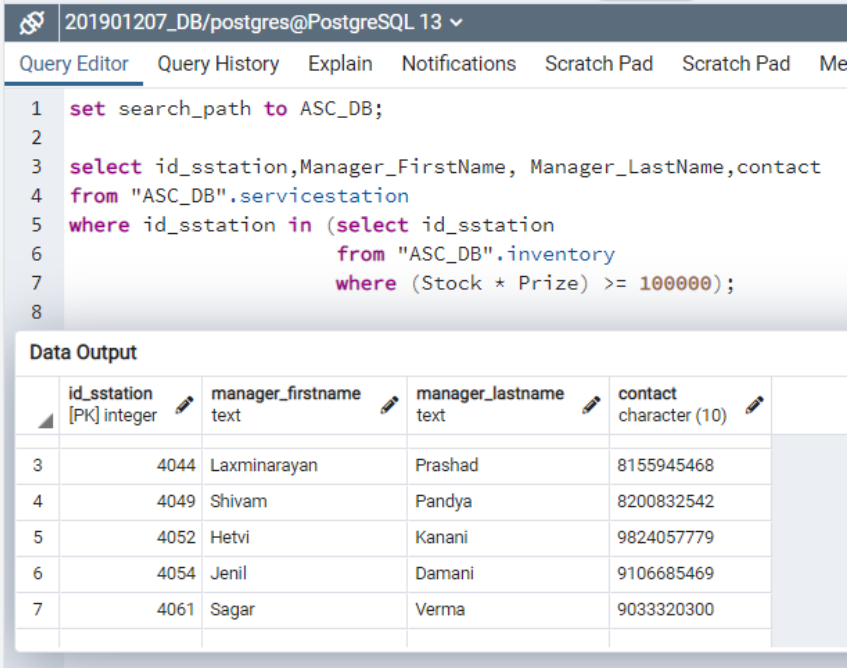
24. List name and contact of the managers whose service station has inventory worth at least 100000.

SQL Query:

```
select id_sstation, Manager_FirstName, Manager_LastName, contact
from "ASC_DB".servicestation
where id_sstation in (select id_sstation
                      from "ASC_DB".inventory
                      where (Stock * Prize) >= 100000);
```

No. of Tuple: 7

Snapshot:



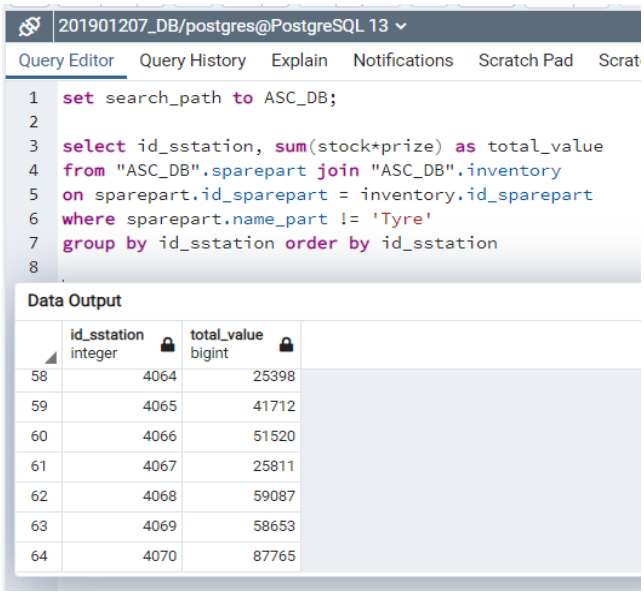
25. Total value of the inventory corresponds to each service station if all the tyre are sold out.

SQL Query:

```
select id_sstation, sum(stock*prize) as total_value
from "ASC_DB".sparepart join "ASC_DB".inventory
        on sparepart.id_sparepart = inventory.id_sparepart
where sparepart.name_part != 'Tyre'
group by id_sstation order by id_sstation
```

No. of Tuple: 64

Snapshot:



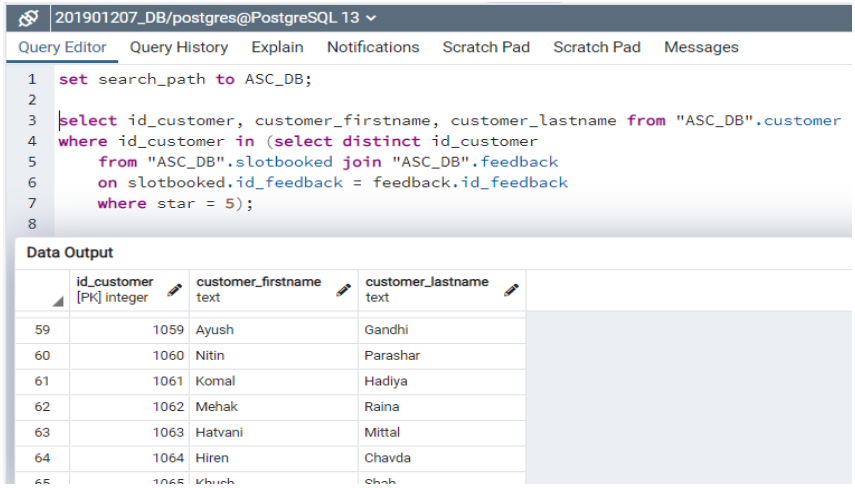
26. Name of the customer who gave 5 star ratings.

SQL Query:

```
select id_customer, customer_firstname, customer_lastname
from "ASC_DB".customer
where id_customer in
        (select distinct id_customer
        from "ASC_DB".slotbooked join "ASC_DB".feedback
                on slotbooked.id_feedback = feedback.id_feedback
        where star = 5);
```

No. of Tuple: 194

Snapshot:



201901207_DB/postgres@PostgreSQL 13

Query Editor | Query History | Explain | Notifications | Scratch Pad | Scratch Pad | Messages

```
1  set search_path to ASC_DB;
2
3  select id_customer, customer_firstname, customer_lastname from "ASC_DB".customer
4  where id_customer in (select distinct id_customer
5      from "ASC_DB".slotbooked join "ASC_DB".feedback
6      on slotbooked.id_feedback = feedback.id_feedback
7      where star = 5);
8
```

Data Output

	id_customer [PK] integer	customer_firstname text	customer_lastname text
59	1059	Ayush	Gandhi
60	1060	Nitin	Parashar
61	1061	Komal	Hadiya
62	1062	Mehak	Raina
63	1063	Hatvani	Mittal
64	1064	Hiren	Chavda

27. How many bearings of 'Ford' company are available at Jaipur service station.

SQL Query:

```
select servicestation.id_sstation, stock
from "ASC_DB".inventory natural join "ASC_DB".sparepart,
"ASC_DB".servicestation
where servicestation.id_sstation = inventory.id_sstation
        and (loca_tion, name_part, company)  =  ('Jaipur', 'Bearing', 'Ford');
```

No. of Tuple: 1

Snapshot:

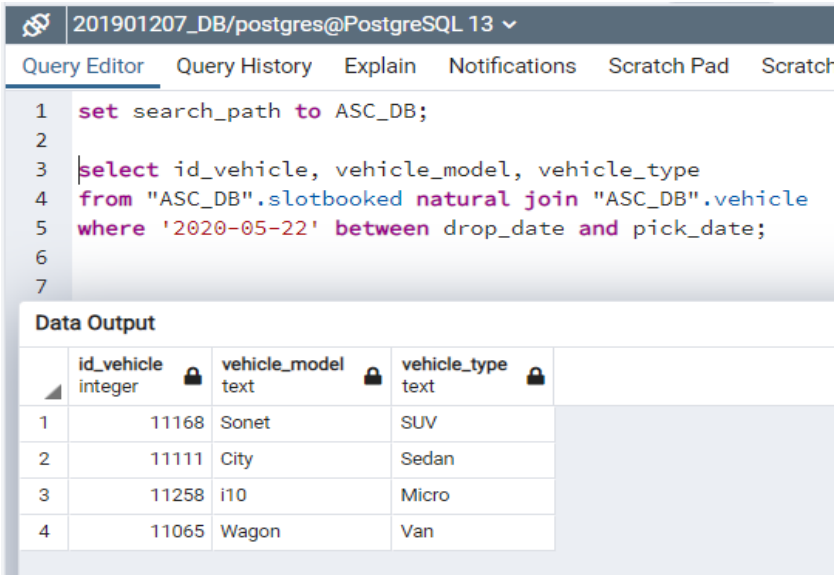
28. Which car requires oil changing on 22nd May 2020?

SQL Query:

```
select id_vehicle, vehicle_model, vehicle_type
from "ASC_DB".slotbooked natural join "ASC_DB".vehicle
where '2020-05-22' between drop_date and pick_date;
```

No. of Tuple: 4

Snapshot:



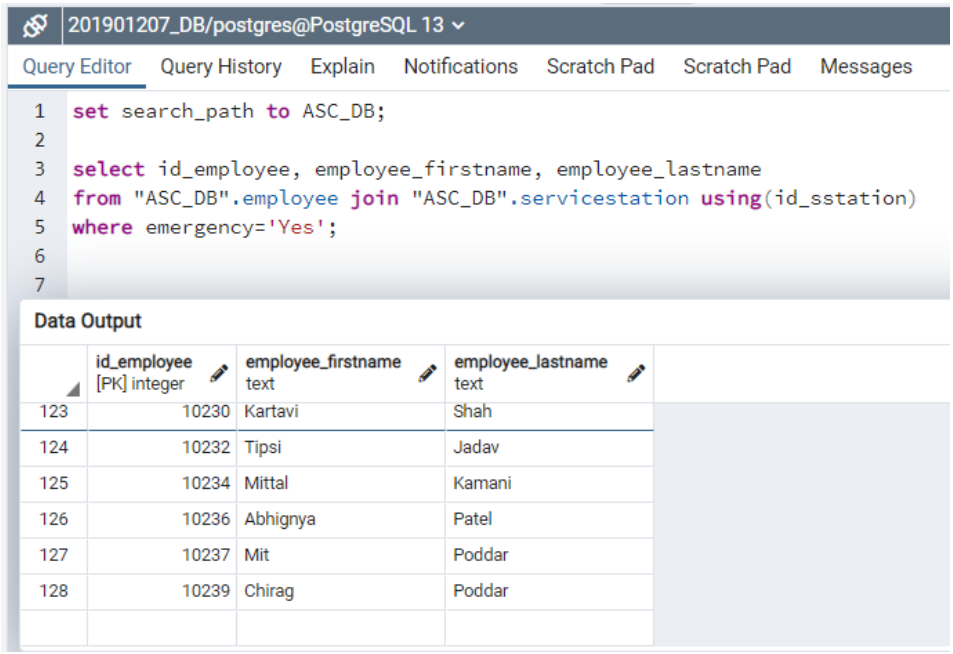
29. Name of employees who work at the station where emergency service is provided.

SQL Query:

```
select id_employee, employee_firstname, employee_lastname
from "ASC_DB".employee join "ASC_DB".servicestation using(id_sstation)
where emergency = 'Yes';
```

No. of Tuple: 128

Snapshot:



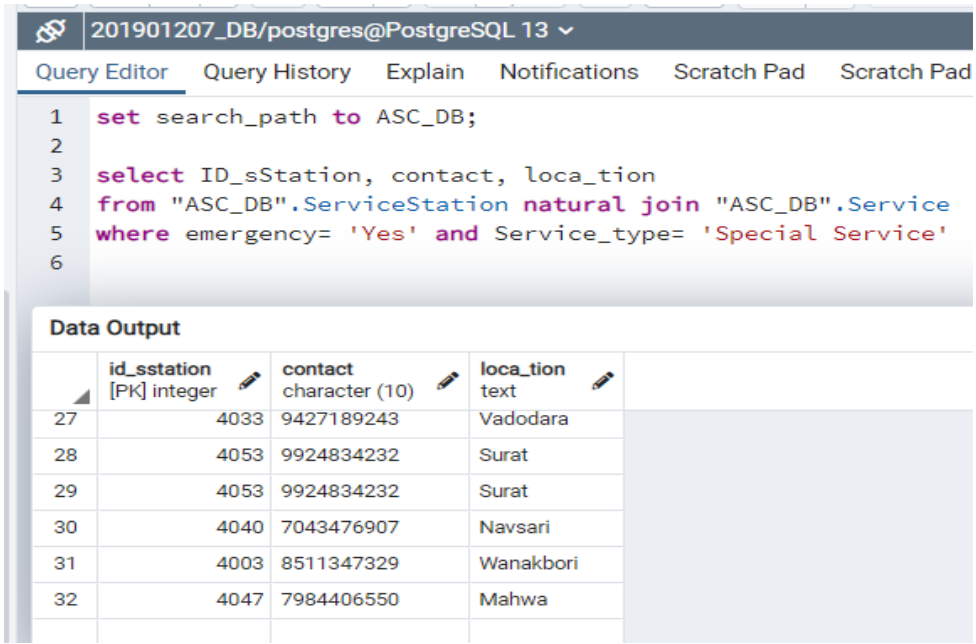
30. List the service stations that provide emergency services as well as special service.

SQL Query:

```
select ID_sStation, contact, loca_tion
from "ASC_DB".ServiceStation natural join "ASC_DB".Service
where emergency = 'Yes' and Service_type = 'Special Service'
```

No. of Tuple: 32

Snapshot:



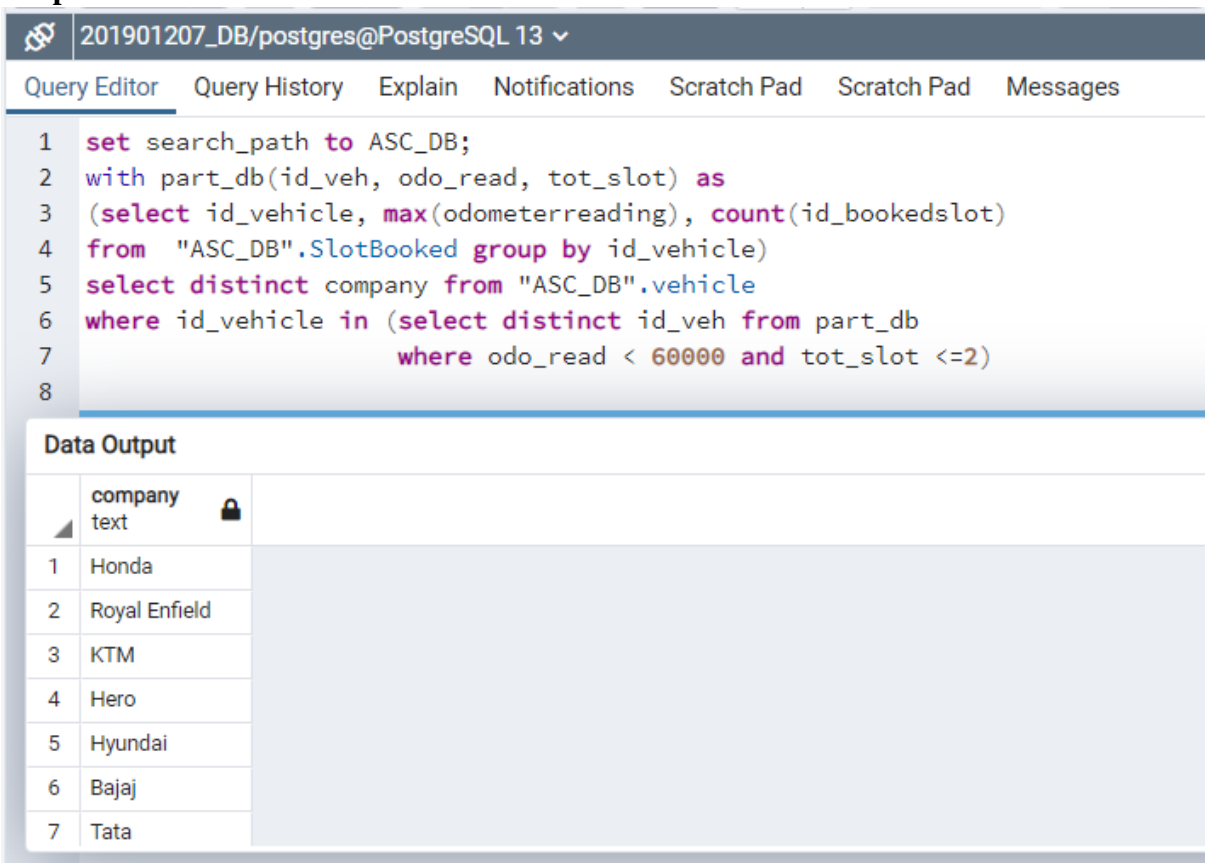
31. Company name of the vehicle which has travelled more than 5000 km and less than 5 services.

SQL Query:

```
with part_db(id_veh, odo_read, tot_slot)
        as (select id_vehicle, max(odometerreading), count(id_bookedslot)
        from  "ASC_DB".SlotBooked group by id_vehicle)
select distinct company from "ASC_DB".vehicle
where id_vehicle in (select distinct id_veh from part_db
                        where odo_read < 60000 and tot_slot <= 2)
```

No. of Tuple: 9

Snapshot:



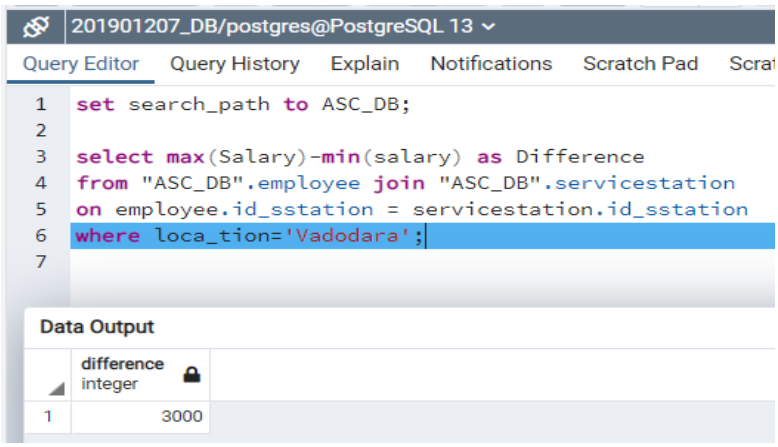
32. Difference in the highest salary of the employees working at different service stations of Vadodara.

SQL Query:

```
select max(Salary)-min(salary) as Difference
from "ASC_DB".employee join "ASC_DB".servicestation
        on employee.id_sstation = servicestation.id_sstation
where loca_tion = 'Vadodara';
```

No. of Tuple: 1

Snapshot:



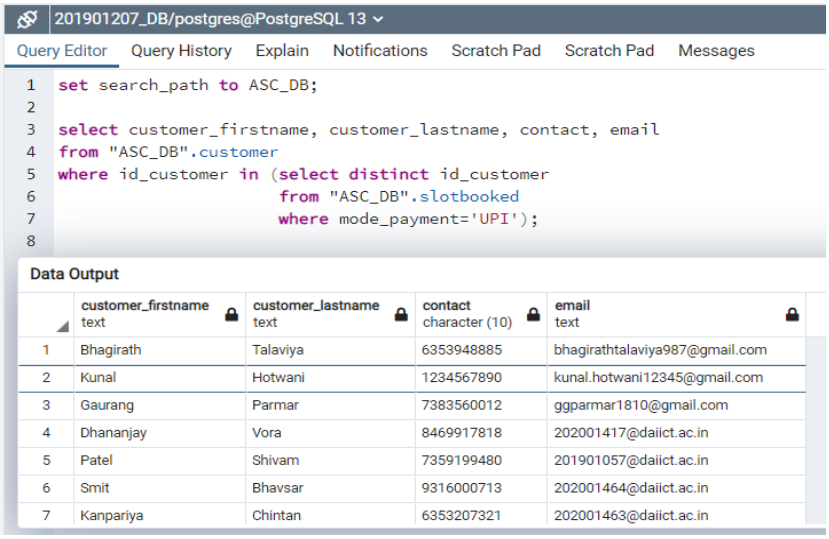
33. Name and contact information of customers who pay through UPI

SQL Query:

```
select customer_firstname, customer_lastname, contact, email
from "ASC_DB".customer
where id_customer in (select distinct id_customer
                      from "ASC_DB".slotbooked
                      where mode_payment = 'UPI');
```

No. of Tuple: 194

Snapshot:

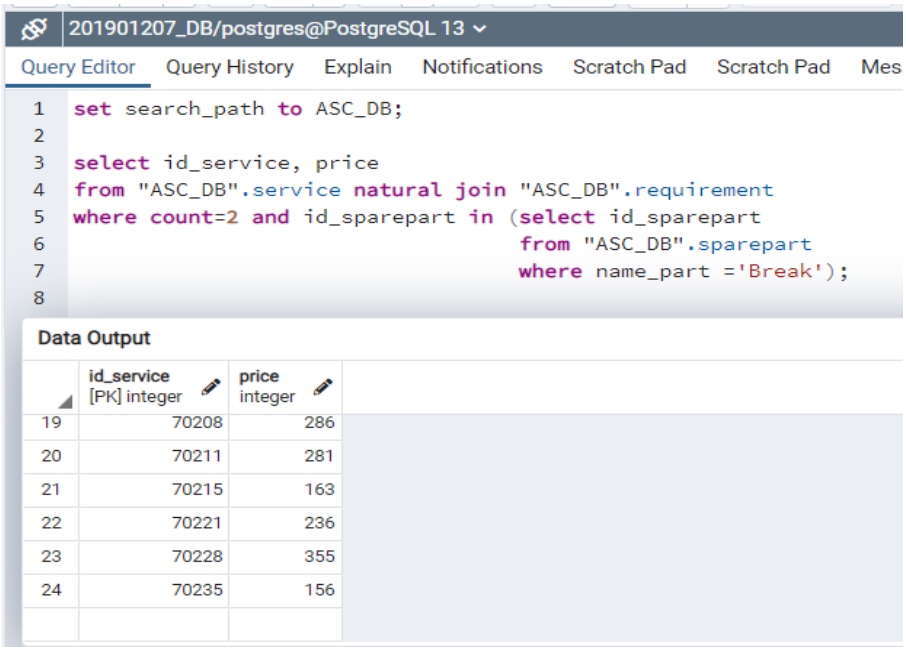


34. Find ID and price of the services which required changing 2 brakes.

SQL Query:

```
select id_service, price
from "ASC_DB".service natural join "ASC_DB".requirement
where count = 2 and id_sparepart in (select id_sparepart
                                     from "ASC_DB".sparepart
                                     where name_part = 'Break');
```

No. of Tuple: 24
Snapshot:



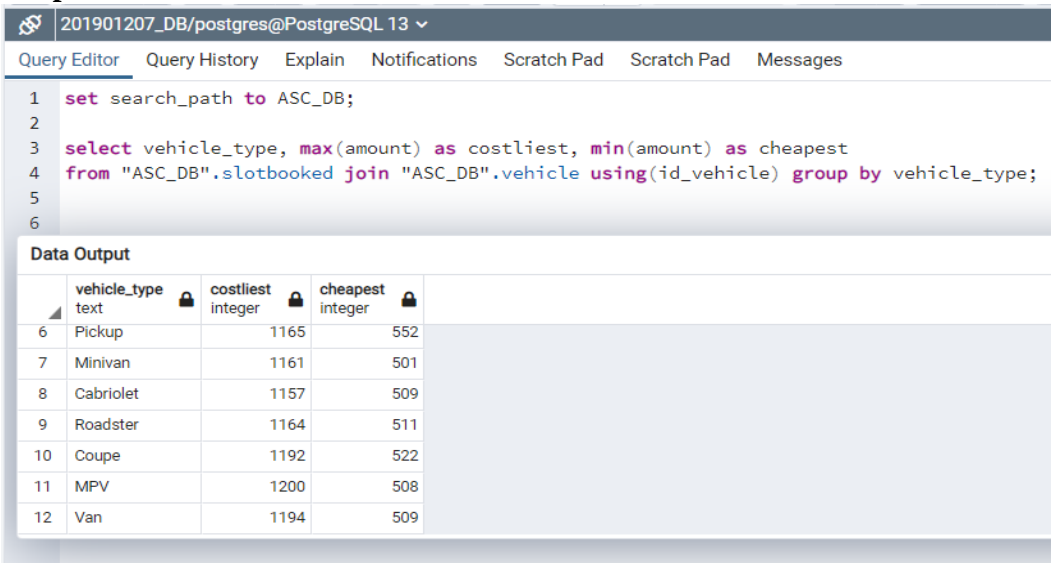
35. Type of the vehicle which has costliest service and type of the vehicle which has cheapest service. Show result in a single table.

SQL Query:

```
select vehicle_type, max(amount) as costliest, min(amount) as cheapest
from "ASC_DB".slotbooked join "ASC_DB".vehicle using(id_vehicle)
group by vehicle_type;
```

No. of Tuple: 12

Snapshot:



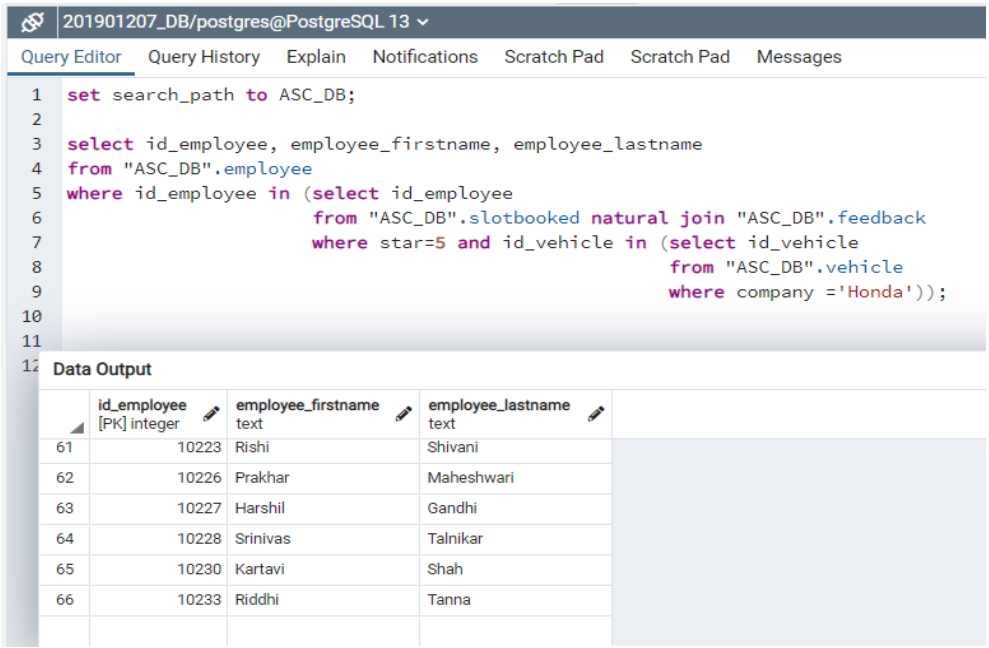
36. Name of employees who have done service of the Honda vehicle with feedback of 5 star.

SQL Query:

```
select id_employee, employee_firstname, employee_lastname
from "ASC_DB".employee
where id_employee in (select id_employee
                      from "ASC_DB".slotbooked natural join "ASC_DB".feedback
                      where star = 5 and id_vehicle in (select id_vehicle
                                                        from "ASC_DB".vehicle
                                                        where company = 'Honda'));
```

No. of Tuple: 66

Snapshot:



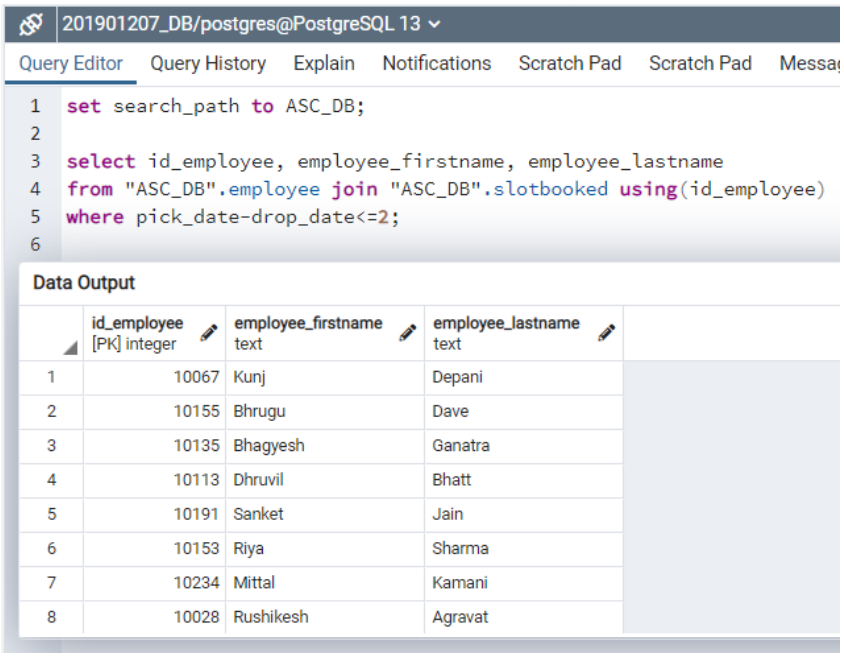
37. Employees who completed the service within 2 day.

SQL Query:

```
select id_employee, employee_firstname, employee_lastname
from "ASC_DB".employee join "ASC_DB".slotbooked using(id_employee)
where pick_date-drop_date <= 2;
```

No. of Tuple: 267

Snapshot:



38. Number of vehicles in ahmedabad whose service costed more than 1000 rupees.

SQL Query:

```
select count(id_vehicle)
from "ASC_DB".vehicle natural join "ASC_DB".slotbooked
where plat_stateCode = 'GJ' and plat_distcode = 1 and amount > 1000;
```

No. of Tuple: 1

Snapshot:

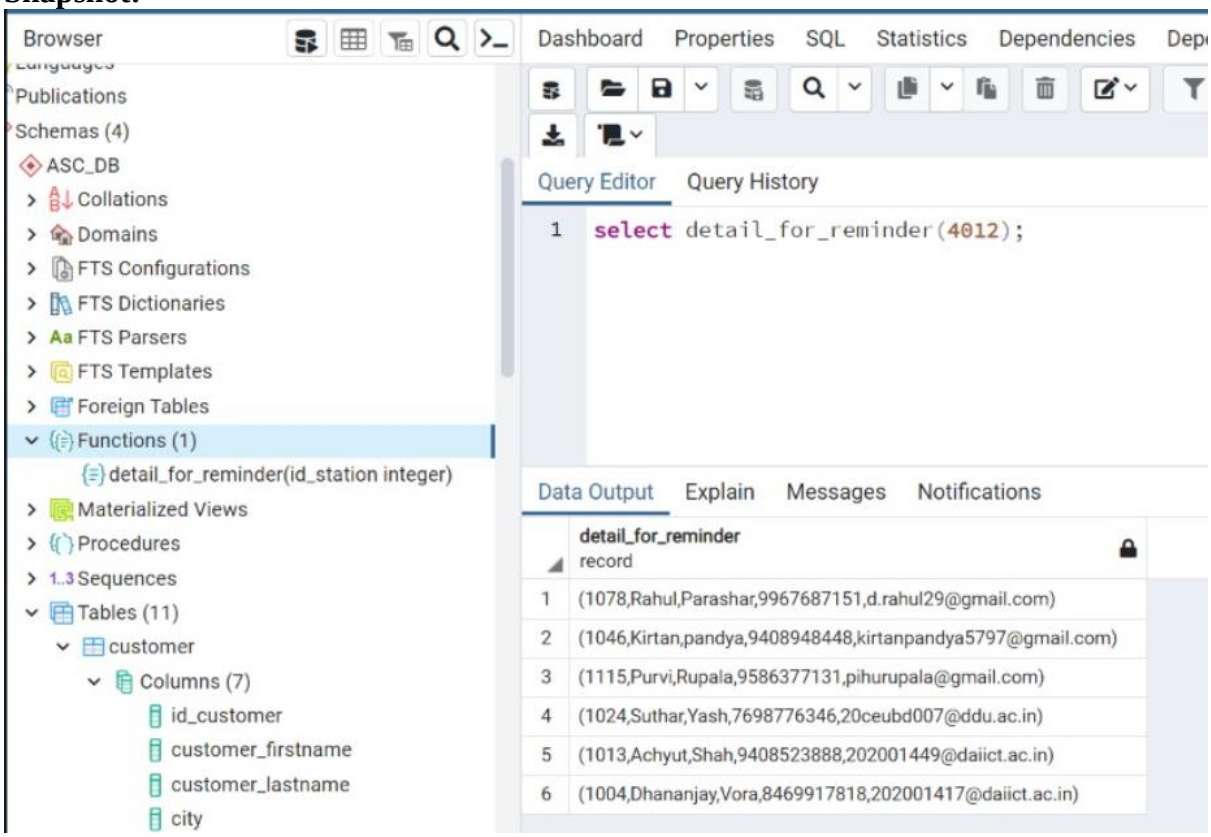
39. Write a function which takes ID of service station as input and return ID, Name, and contact detail of the customer whose last Regular Service is done more than 90 days ago.

SQL Query:

```
CREATE OR REPLACE FUNCTION
"ASC_DB".detail_for_reminder(id_station integer)
RETURNS TABLE(id_customer integer, name_first text, name_last text, mobile text,
email text)
LANGUAGE plpgsql

AS $BODY$
begin
create temp table details(id_customer integer, name_first text, name_last text, mobile
text, email text) on commit drop;

insert into details (
select customer.id_customer, customer_firstname, customer_lastname, contact,
customer.email
from "ASC_DB".customer
where customer.id_customer
        in (select vehicle.id_customer
            from "ASC_DB".vehicle
            where id_vehicle
                in (with last_date(id_veh,date_last)
                        as (select id_vehicle, max(pick_date)
                            from (select * from "ASC_DB".slotbooked
                                    where id_sstation = id_station) as tbl
                                    group by id_vehicle)
```

```
                    select id_veh from last_date
                    where (current_date - date_last) > 90)));

return query table details;
end
$BODY$;
```

No. of Tuple: 6
Snapshot:



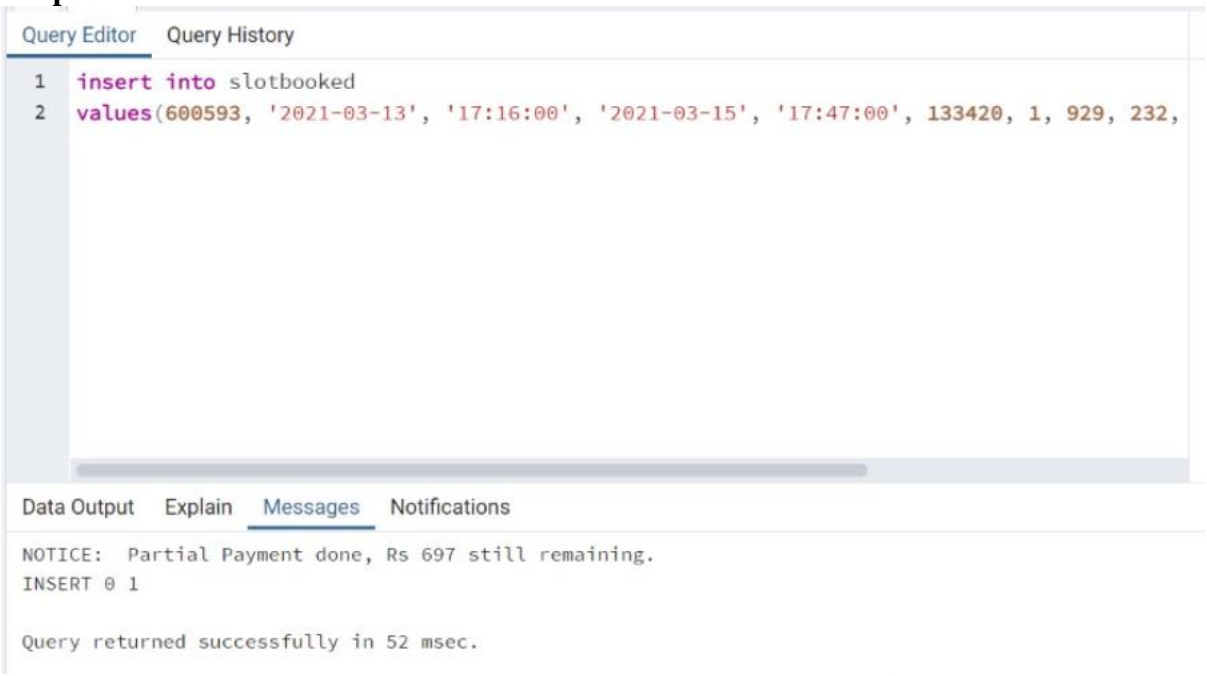
40. Write a trigger function that raise a notice about payment detail, if cashier need to give the change, partial payment is pending or perfect payment is done. Apply it on appropriate relation.

SQL Query:

```
CREATE or replace FUNCTION "ASC_DB".after_payment()
 RETURNS trigger
 LANGUAGE 'plpgsql'
 NOT LEAKPROOF

AS $BODY$
BEGIN
If new.paid_amount < new.amount
Then
RAISE notice 'Partial Payment done, Rs % still remaining.', new.amount -
new.paid_amount;
Else
```

```
if paid_amount > amount
then
Raise notice 'Excess payment, Return Rs %', new.paid_amount - new.amount;
Else
Raise notice 'Fully Paid';
end if;
End if;
End
$BODY$;
```

Snapshot:



Section7: Project Code with output screenshots

Code:

We have created an android application to develop a UI for database. It implements Vehicle table of our relational schema using Room persistence library. Room persistence library is implemented over SQLite database.

We have used a local database to perform CRUD operation over it and saving data.

We implement application by MVVM (Model View ViewModel) architecture.

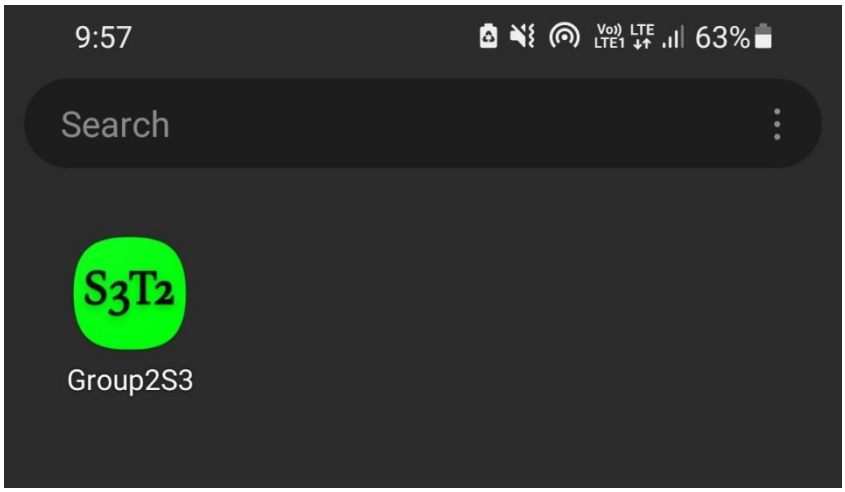
As the code can't be included in the report, we have attached a ZIP file of the code along with the report. Repository of the code with APK is also on the GitHub and link is:

GitHub Repository Link: https://github.com/201901207/VehicleProjectFinal

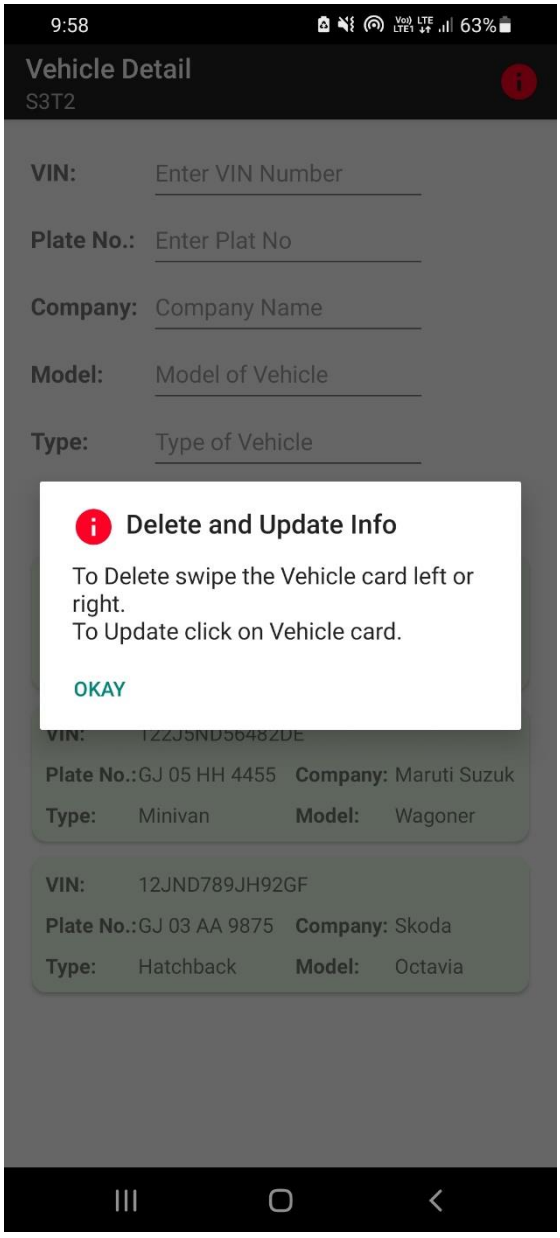
APK Link: GitHub-VehicleProjectAPK

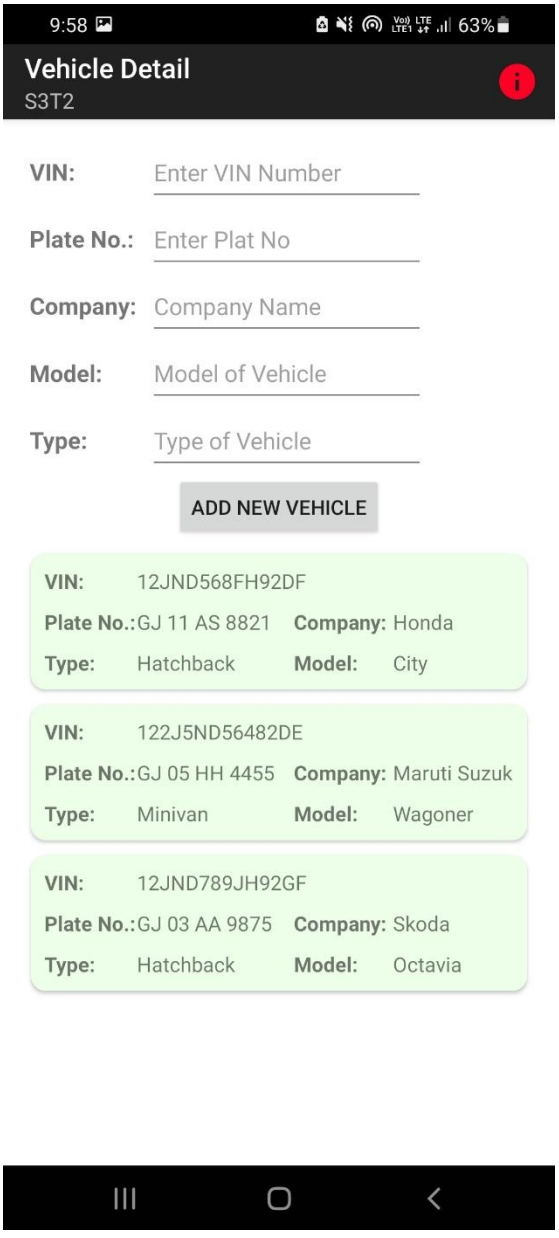
Screenshot of the APP:

- App on Screen grid.

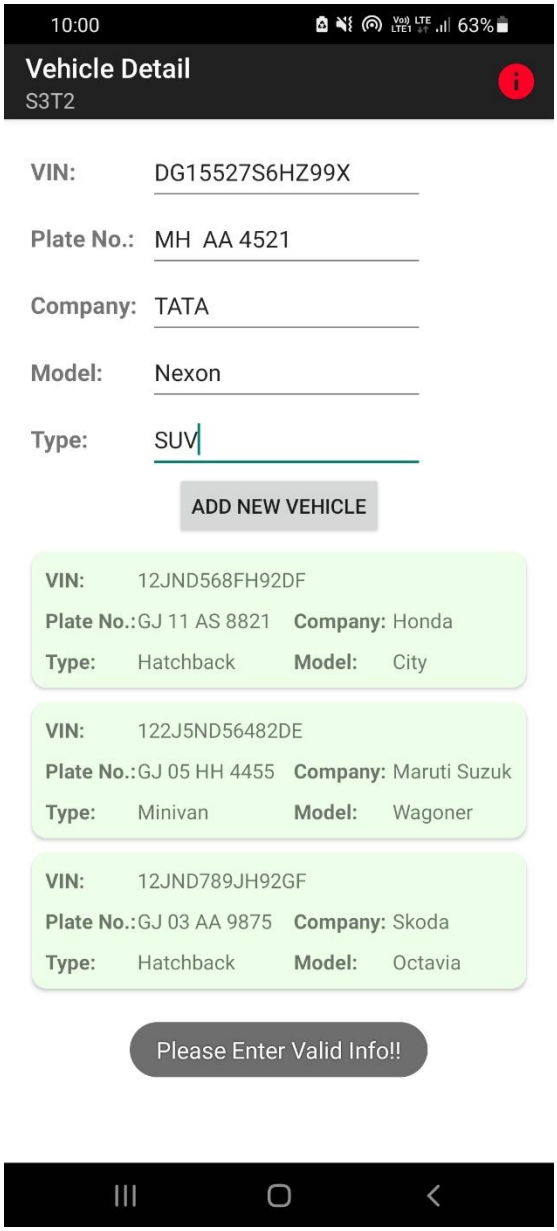


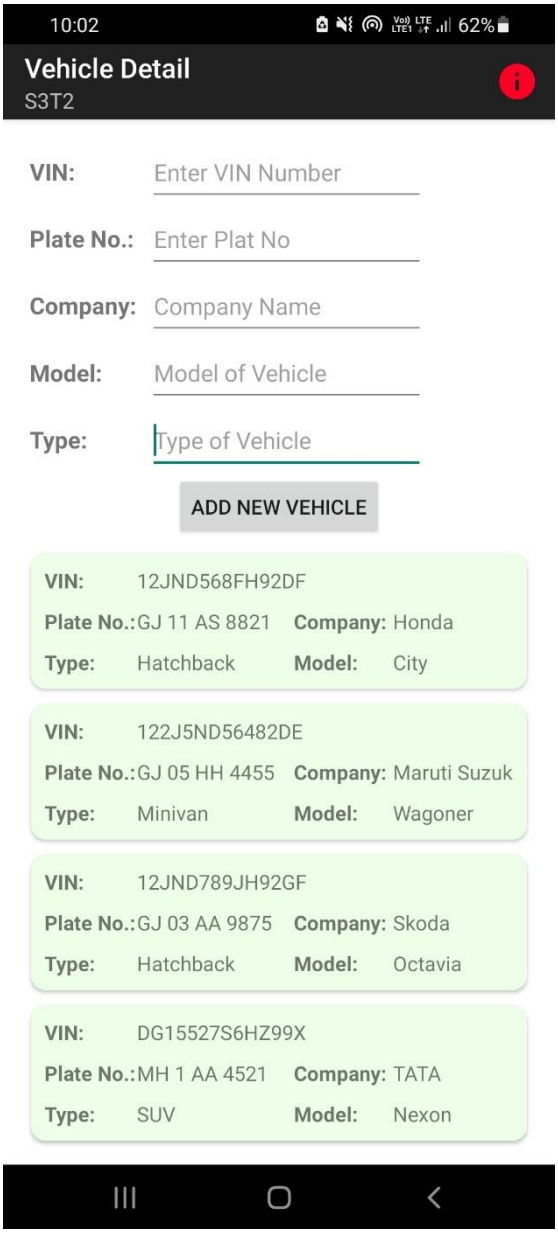
- Launch Screen after First time Installation on the device.
  - It will show Information about various functionality. Later user can see it by pressing info button on top right corner.
  - We have added 3 tuples already just to initialize DB.



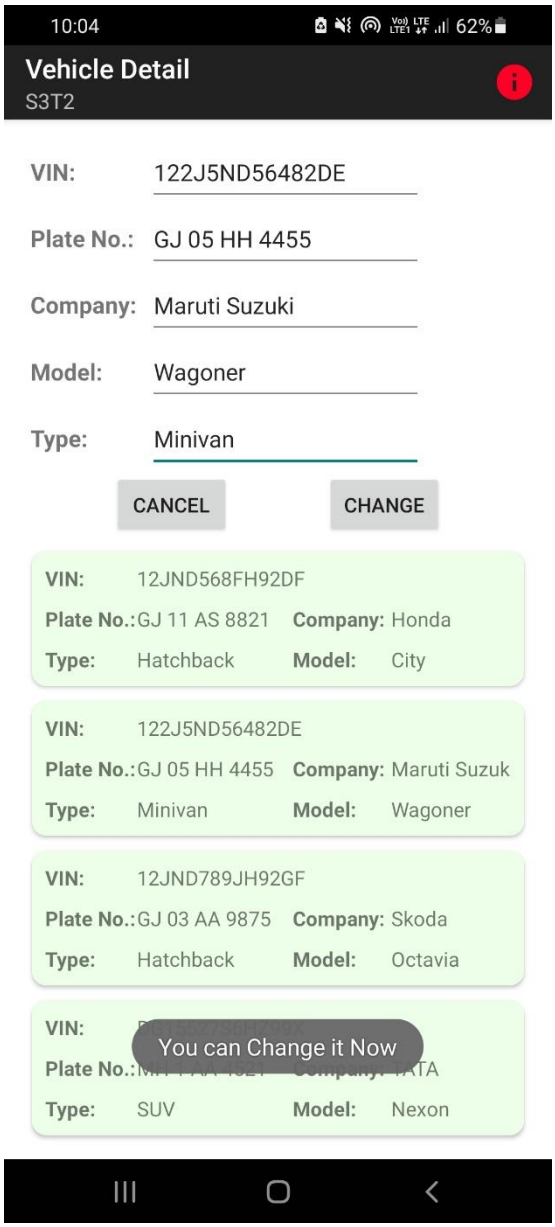


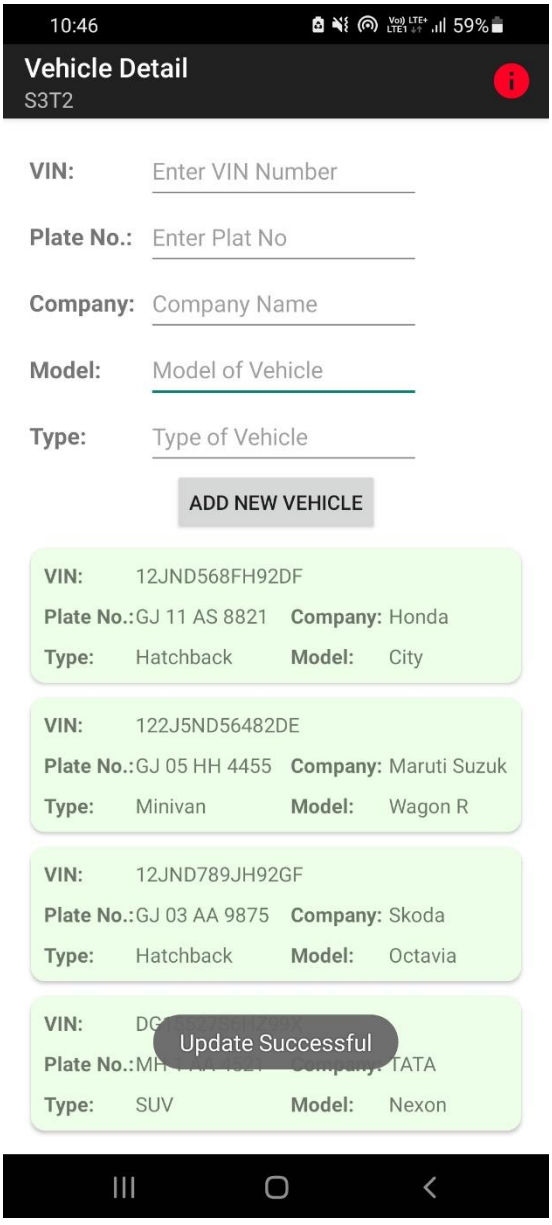
- Fill the detail and press ADD NEW VEHICLE button
  - On adding new vehicle new vehicle will be added after Validation of the data. We will verify that none of the field is empty and Plate Number is in valid form.
  - If there are mistakes in data filling, we will not add data in DB and show message to enter valid data.
  - On entering valid input, on clicking button the data will be added to Database and changes according to database will be appeared on the screen.
  - The list is scrollable if the number of vehicles is large.
  - Here we added the new vehicle to Database and it is also visible on UI now.



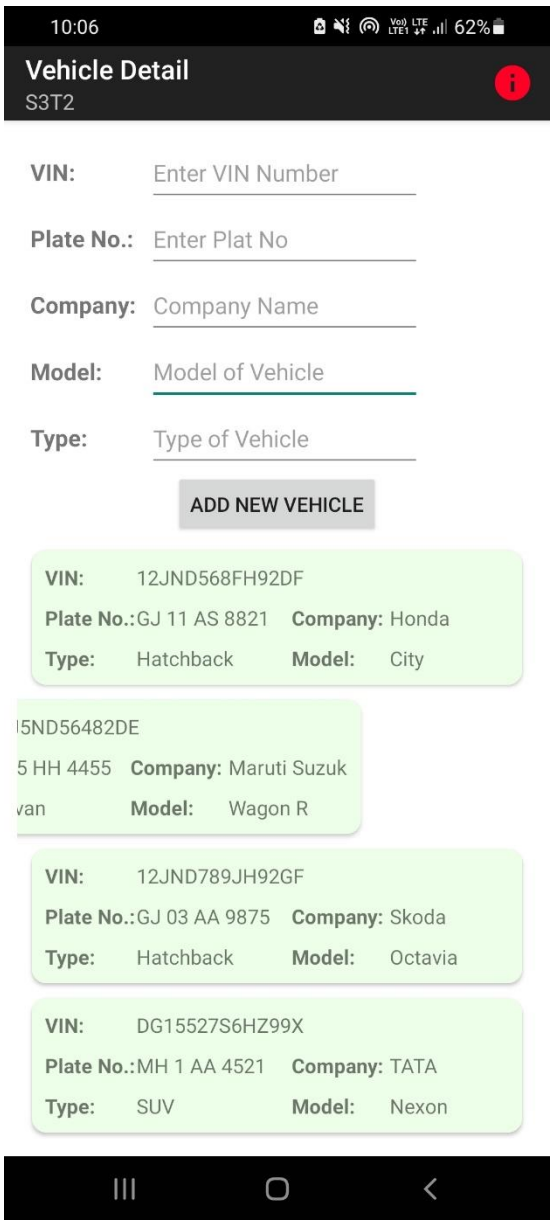


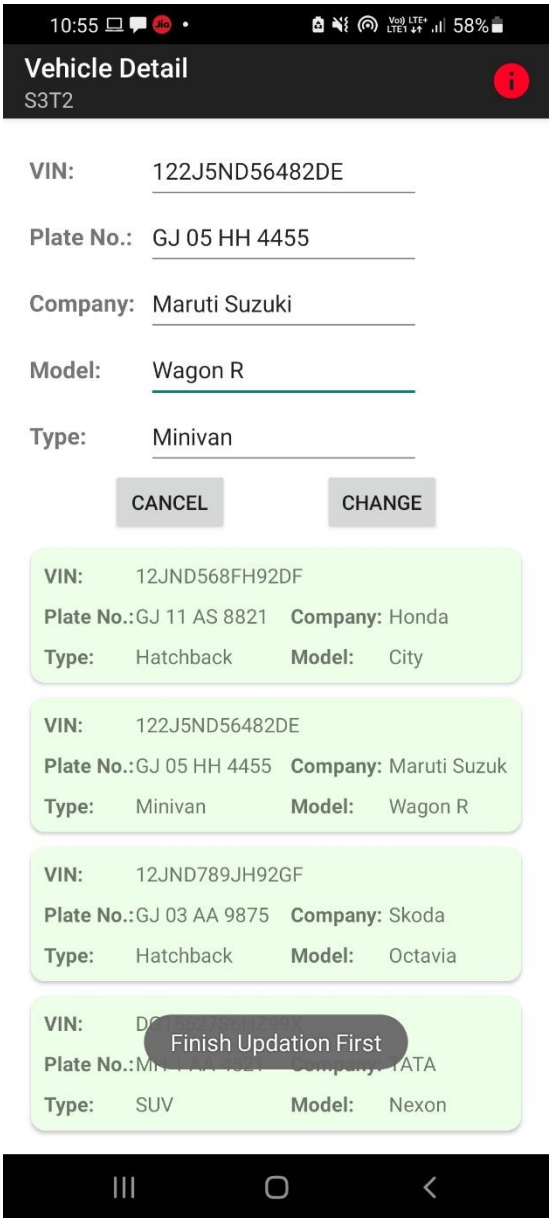
- To Update existing Vehicle's detail, touch on the particular vehicle card.
  - On click over vehicle card, vehicle's detail will be added to form above.
  - Cancel and Change button will appear on screen.
  - Choosing cancel will not make any changes and UI will come in to the original state.
  - Clicking on Change will update the detail of vehicle according to the detail changed in form and the changes will also be reflected on screen. Data validation will be applied here too.
  - Ex. Changing 3rd card from Wagoner → Wagon R





- Deletion can be performed swiping left or right the corresponding card of vehicle.
  - Deletion can’t be performed while updating.





- On swiping away card alert will be raised for user to confirm deletion.
- If User click DELETE, only then data will be deleted and the changes will be display on Screen accordingly.
- If User Cancel deletion, we will abort the operation.