Dhirubhai Ambani

Institute of Information And Communication Technology

IT214 - Database Management Systems

Automobile Service Center

Prepared by Section 3: Team 3

ID – 201901118 NAME – Anjali Prajapati

ID – 201901127 NAME – Foram Mehta

ID – 201901171 NAME – Madhvi Padshala

ID – 201901236 NAME – Parita Makvana

Mentor TA - Raj Shah

Created on: 21st September 2021
Last Modified: 27th November 2021

Table of Contents:

Section 1: Final Version of SRS 5

1. Understand and Complete the Description of the Case study/Problem Domain 5

1.1 Purpose 5

1.2 Document Conventions 6

1.3 Intended Audience and Reading Suggestions 6

1.4 Product Scope 7

1.4.1 Advantages 7

1.5 Description 7

1.5.1 REAL-WORLD working & issues 8

1.5.2 Entities 9

2. Fact Finding Phase 11

2.1 Description of Background Readings/listings: 11

2.2 Interviews 13

2.2.2 Combined requirements gathered from Interviews 19

2.3 Questionnaires 20

2.4 Observations 27

3. Fact-Finding Chart 28

4. Combined Requirements 29

5. User Classes and Characteristics 32

6. Operating Environment 33

7. Product Functions 34

8. Privileges 35

9. Assumptions 35

10. Business Constraints 36

Section 2: Noun Analysis 37

1. Extracted Nouns & Verbs from Problem Description 38
2. Accepted Noun & Verbs list 41
3. Rejected Noun and Verbs list 43

Section 3: ER-Diagrams all versions 46

1. ER diagram version - 1 47
2. Improvisation in ER Diagram 48
3. ER diagram version 2 (Final Version) 50

Section 4: Conversion of Final ER-Diagram to Relational Model **51**

1. **Mapping ER to Relational Model** **52**
2. **Relational Model (before normalization)** **53**

Section 5: Normalization and Schema Refinement **54**

1. **Normalization** **55**
2. **Revised Relational Model Schema After Normalization** **61**
3. **Revised Relational Model After Normalization** **64**

Section-6: Final DDL Scripts and SQL Queries **65**

1. **DDL Script (updated)** **66**
2. **Data Snapshot** **75**
3. **SQL Queries** **93**

4. Triggers and Functions (Stored Procedure) **134**

4.1 Functions **134**

4.2 Triggers function **138**

Section-7: Project Code with output screenshots **151**

1. **Front-End Development** **152**

Section 1: Final Version of SRS

1. Understand and Complete the Description of the Case study/Problem Domain

1.1 Purpose

Product: Automobile Service Centre

Scope of the Project: System for booking & paying for automobile services

Buying cars is always fascinating but maintaining them is not always easy. How can one in this busy world keep track of car servicing? The situation is exacerbated if one owns a single car and uses it daily. Hence, it's a crucial part from the side of automobile centres to ensure that their customers get regular service that is up to their satisfaction.

Since the automobile centres deal with an abundant number of cars it's not easy to keep the records manually and hence databases play an important role here. This could help the employees and project manager of the automobile centre to ensure the excellent progress of the company.

To help the company overcome the difficulties faced by the customers, we have come up with a logical and conceptual design of a database for the Automobile Service centre.

The sole purpose would lie in the welfare of the customers. We intend to make the software that would have booking and paying features for the customers based on the availability of the slots. Customers would be able to choose any centre that would have an agent assigned to follow up the further process. On the admin side, one could see employee details and also the feedback by the users.

Eventually, the software would consider regular service alerts, emergency breakdown pickup, and drop facilities.

1.2 Document Conventions

This document is formed using the IEEE template for System Requirement Specifications (SRS) documents. The Document font convention is Calibri and our typographical convention will be using bolding of words to emphasize words that are of special significance.

1.3 Intended Audience and Reading Suggestions

This system for is for following people:

- **developers** - minimizes the amount of time and effort developers have to expend to achieve desired software goals. It thus reduces development cost.

- **project managers** - Project Managers uses this to ensure all requirements are met by employees and everything is running smoothly.

- **Agents** - Agents use this to retrieve whatever information is related to their projects, so they could work on given projects.

- **Marketing staff** - This system improves communication between team members by saving and displaying the product feature description in one central location that everybody can easily access. So, there won't be any confusion during marketing. That's why marketing staff use this.

1.4 Product Scope

This Automobile service centre would help in efficient management of past as well as new customer's needs perfectly and vividly. It enables the company to track its progress, helps the project manager to ensure whether the employees are working properly, and provides employees a mechanism through which they could give their customers regular service alerts.

The online system would enable the customers to visit any service centre irrespective of its location. This would be helpful in case of emergencies. There would also be an individual employee for a particular location to assist the customers. One location would have multiple employees to ensure smooth functioning.

1.4.1 Advantages

→ Fast, efficient and reliable system
→ Excellent in tracking old customers
→ Provides security in case of fraud as every customer is uniquely identified.
→ Highly Secure and Portable application
→ Provides facilities to check availability of slots.
→ Helps customer to find its nearest servicing centre
→ Provides best service plan to the customer

1.5 Description

In a manual system we have to maintain an employee's detail book, accounting Books, bills etc.

There are many drawbacks of manual system like:-

- Difficult to maintain records.
- Difficult to generate a report where we need to combine things from different books.
- Time consuming and difficult to search for a record.
- Multiple concurrent access not possible

Every organization has the challenge of managing service centres, their clients, payments etc and has different needs.

1.5.1 REAL-WORLD working & issues

- In the real world, authorization becomes very complex. Manually matching the credential may lead to failure.
- In real life a person may have more than one vehicle and this may lead to duplication of data.
- There might be difficulty in the real world to keep track of hard working employees.
- A customer might love a particular employee’s service and hence they need to be praised.

Hence we need to design a system that would consider these challenges and for this we performed a background reading analysis of websites like mericar.com, app - Gomechanic and blog - Glenn's auto.

After the analysis we got to know several plus points like

- The website was offering breakdown services to their customers.
- The Aap GoMechanic was excellent in every domain like online service booking and payments,

  affordable pickup and drop facility online booking.

What we learnt from the analysis is that we should incorporate this in our system for better results. Hence our system is designed so that this would help the company in strategic planning and would help to ensure that the company is equipped with correct details of the future goals.

Our system would automate the existing manual system with the help of computerized equipment. We have designed exclusive employee and customer management systems that are adapted to your managerial requirements. The system would have remote access features and would ultimately allow you to manage resources in a better way.

We have incorporated the features from existing system and have also added few more features from our side like -

- There would be relations that would keep track of all the services the company has provided till date. This would help in determining the next service date and would then give service alerts to the employees.
- The relations are supported by triggers that would ensure there is no duplication of tuples . The triggers would also help to set the service date and time automatically whenever a customer comes up with a servicing request.
- Many times this might be the case that some employees/agents might be stealing company's resources for their personal use without any permission. This would be against the rule and so to keep track on whether there is no misuse of the stock we would have a table that would keep the quantity information of the company's general resources.
- Our system would have two way authorization to verify the customer. The customer would be cross checked via his ID and then an OTP would be sent to the phone number associated with the ID. The service would be given only by verifying through the OTP. This can help reduce fraud.
- We would also include feedback from users' side, the amount paid by the customer for servicing and the number of times the customer has visited the service center for car service. So that this information can be useful to us in future.
- The system would keep track of the service along with the agent who followed up the process. So the issue of not recording the customer’s feedback would not arise.

The above are some different points that we decided to implement in our system.

1.5.2 Entities

- **Customer =** This relation would contain personal data of the customers.It will have **customer_id**, cust_name, age, address and email.

- **Vehicle =** The relation would have all the registered vehicle description including its name (KIA, Hyundai etc), its type(scooty, bike, bus etc.) and the duration after which the corresponding vehicle needs service. This would include **v_no**, vehicle_name, vehicle_type, duration and RC_No.

- **Employee =** This relation would contain personal data of the employees. This includes **employee_id**, e_name, e_contact, service_location, salary and designation.

- **Stock =** This table would keep stock information of Basic things required at the service center like Engine Oil, Coolant, Cleaning water etc and it will also include **loc_id**.

- **Slots =**This would help customers identify whether the slots are available in their preferred location for their vehicle type. This includes **loc_id** ,vehicle_type, availability.

- **Track=**This would help employees to add all the service records till date along with the date of the service. This would also include feedback from users' side, the amount paid by the customer for servicing. This includes **track_id**, loc_id, v_no, s_date, end_date, amount, feedback. Here we will create a trigger which would count feedback points for a particular loc_id.

- **Alert =**This would help employees determine which customer needs to have its car service done and also keeps the past records like what kind of pick-up service was adopted in the past by the past customers. This includes **v_no**, alert (if date – end_date> duration then update), pick _up_type {home pickup OR emergency}).

- **Progress =** This relation would keep track of the progress of every service center across the nation. This would be accessible by the CEO where he/she would be able to see attributes like **loc_id**, **current_year** and Points. The center having maximum points would be succeeding and others would be lagging.

- **Service** = This table would keep information of vehicle_no, which employee serviced this vehicle, which services were done and when it was done. Also service_charges and how many times the vehicle got serviced would be here. This includes date, **v_no**, e_id, s_type, charges and no_of_services. While entering data about service in this table we will create a trigger that would generate bill for particular v_no and services.
- **Loc_info =** This table has information about location of service centres. This includes **loc_id** and service_location.

2. Fact Finding Phase

2.1 Description of Background Readings/listings:

Website: MeriCAR.com

- On this Website, they mentioned the importance of trusted mechanics. People only want services from trusted mechanics. They had doorstep services, so customers can see everything before their eyes. They have trusted skilled service providers that come to your house for automobile services. They provided fully equipped van with trained professionals.
- Their system had car breakdown on road service. Cars could break down anywhere and sometimes won't start anytime soon. In this circumstance they will come to your rescue and repair your vehicle instantly.
- They also provided denting painting services at lower rates than market rates.
- They also had weekend offers, where they provide complete service with discounted rates. So, it is important to have offers to attract more customers.
- They also provided a manual guide to get battery fuel efficiency.They also had a section where they answer customers' questions, like when you should service your car, what's involved in services, how clean air filters works. So, to make a reliable service centre database, customer communication is important.
- They seemed to have very nice services with good price points. But their website was not secure so they should probably upgrade their website with a secure connection.

Play Store App : GoMechanic

- This app provides services like denting & painting, AC service & repair, Car spa & Cleaning, Batteries, Tyres & wheel Care, Clutch Fittings etc.
- This app also provides scheduled packages like basic service, standard service etc. They have very clear descriptions about every service like how much time taken, when recommended, free pick-up & drop availability with service price. They also have a section where they answer frequently asked questions about automobile services. And also have a customer review section according to your location.
- They have another feature in which they find a workshop near your area. They use genuine parts with very good warranty policies. Moreover, they provide affordable services with a pickup & drop facility.
- They also have a blog section where they provide articles related to basic car problems with automobile solutions.
- They have SOS services and also they give massive discounts on their annual app membership. By all of this we can say this app has really good services and amazing customer service.

Blog- Glenn's auto

- This blog focuses on different automobile service sections and guidelines for that, which seemed pretty helpful. They stated the importance of having certified and experienced mechanics for servicing centres.
- They also had a customer review section. There were some maintenance tips for vehicle owners. Then they listed services which their service centre provides.
- Then there were social handles of their service centres. Also they had this section called Gallery, where they posted their service centre pics with customers. Which indicates good customer interaction and services.

References

1. http://www.mericar.com/car_service_center/
2. https://play.google.com/store/apps/details?id=gomechanic.retail&hl=en_IN&gl=US
3. https://glennsauto.com/blog/

Requirements gathered from Background Readings

- ➔ Customers use online platforms in times of emergency. So to make our system more helpful, we will provide convenient service scheduling in our system.
- ➔ It is important to provide reasonable prices for our automobile service centre. Also it is required to provide a tracking facility, so that customers would have an idea about when mechanics would reach on their location for repairing.
- ➔ Customers will pay when services are easy and comfortable. So, we will provide a variety of paying options including debit card, credit card, net banking and also cash-on-delivery. Also it should be secure and encrypted.
- ➔ We will have a rating and feedback section where customers can review for services and quality. Also new members can check them for reference.
- ➔ The system should provide all the useful information about automobile servicing so that customers don't have to do web searches to extract information.
- ➔ This system should also create a sense of credibility in the business transactions.

2.2 Interviews

Interview 1

Role Play Interview Plan

System: Automobile service center

Interviewee: **1) Brijesh Pachouri** (Role Play)

Designation: Service Advisor at Ignition Automobile Service Center

Interviewer: 1) Anjali Prajapati (Student at DAIICT)

2) Madhvi Padshala (Student at DAIICT)

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

Duration: 45 minutes **Place:** Google Meet

Purpose of Interview:

Preliminary meeting to identify problems and requirements regarding the system being created for Automobile Services.

Agenda:

Software Facilities / requirements

Managing daily work flow and schedule

Complete Servicing style

Documents to be brought to the interview:

- One Rough plan of the the system / software

Mock Interview Summary 1

System: Automobile service center

Interviewee: 1) Brijesh Pachouri (Role Play)

Designation: Service Advisor at Ignition Automobile Service Center

Interviewer: 1) Anjali Prajapati (Student at DAIICT)

2) Madhvi Padshala (Student at DAIICT)

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

Duration: 45 minutes **Place:** Google Meet

Purpose of Interview:

Preliminary meeting to identify problems and requirements regarding the system being created for Automobile Services.

1. Our purpose is to reduce the work to Managing the details of vehicle, customers , payment , registrations and auto spare parts , suppliers.

2. Parking space problem at service center , and there is enough customer waiting area but needs to include children play area.There is Hygienic Pantry, neat and clean Washrooms and other facilities also available at the service center.

3. We need a system that can give regular service alerts to the customers according to their vehicle service duration and which provides customers with Pick up / drop facilities anytime , anywhere . so that Customers get satisfied with our services.

4. We need a good follow-up system too , so that customers can give their feedback for our services.

5. We should Maintain a database for a stock of the Auto Oil , fuel , cleaning water etc so that we can inform a supplier of the necessary things whenever our stock is empty.

Interview 2

Role Play Interview Plan

System: Automobile service center

Interviewee: 1) Kashyap Tiwari (Role Play)

Designation: Manager at Ignition Automobile Service Center

Interviewer: 1) Foram Mehta (Student at DAIICT)

2) Parita Makvana (Student at DAIICT)

Date: 6/10/2021 **Time:** 16:30

Duration: 45 minutes **Place:** Google Meet

Purpose of Interview:

Preliminary meeting to identify problems and requirements regarding services provided to Customers at Ignition Automobile service center.

Agenda:

Current security procedures

Dealing with Coworkers

Customer's Satisfactions

Follow-up actions

Documents to be brought to the interview:

- One Rough plan of the the system / software

Mock Interview Summary 2

System: Automobile service center

Interviewee: 1) Kashyap Tiwari (Role Play)

Designation: Manager at Ignition Automobile Service Center

Interviewer: 1) Foram Mehta (Student at DAIICT)

2) Parita Makvana (Student at DAIICT)

Date: 6/10/2021 **Time:** 16:30

Duration: 45 minutes **Place:** Google Meet

Purpose of Interview:

Preliminary meeting to identify problems and requirements regarding services provided to Customers at Ignition Automobile service center.

1. It is very important to understand customers' problems and make timely, practical decisions.
2. Sometimes it happens that someone wrong person access the customer details and tries to use it for their benefits , so we have to make sure that only authentic people can access the details so that we reduce these frauds.
3. We should keep all the details of the Co-workers who work there , and regularly check their progress report. Employee management system required for this.
4. We are looking for ways to provide value to customers , so that We have to take regular feedback from customers to improve servicing procedures.

Interview 3

Role Play Interview Plan

System: Automobile service center

Interviewee: 1) Harikrishn Murti(Role Play)

Designation: CEO of Ignition Automobile Service Center

Interviewer: 1) Madhvi Padshala (Student at DAIICT)

2) Parita Makvana (Student at DAIICT)

Date: 7/10/2021 **Time:** 10:30

Duration: 45 minutes **Place:** Google Meet

Purpose of Interview:

Preliminary meeting to identify problems and requirements regarding management and services provided to customers at Ignition Automobile service center.

Agenda:

Initial ideas

Problems with security and any other concerns

Strategies to escalate business

Documents to be brought to the interview:

- One Rough plan of the the system / software

Mock Interview Summary 3

System: Automobile service center

Interviewee: 1) Harikrishn Murti(Role Play)

Designation: CEO of Ignition Automobile Service Center

Interviewer: 1) Madhvi Padshala (Student at DAIICT)

2) Parita Makvana (Student at DAIICT)

Date: 7/10/2021 **Time:** 10:30

Duration: 45 minutes **Place:** Google Meet

Purpose of Interview:

Preliminary meeting to identify problems and requirements regarding management and services provided to customers at Ignition Automobile service center.

1. We need a system with remote access features , which will allow us to manage the system workforce anytime , anywhere.

2. It needs a better system or a full fledged computer software so that they can concentrate on their services and other requirements rather than concentrating on the record keeping.

3. Need to keep records of all the Co-workers , employees and managers who are currently working in the service center and also people who worked there in the past.

4. It is necessary that some information should be kept private so that no unauthorized person can misuse it and the information can be well preserved.

2.2.2 Combined requirements gathered from Interviews

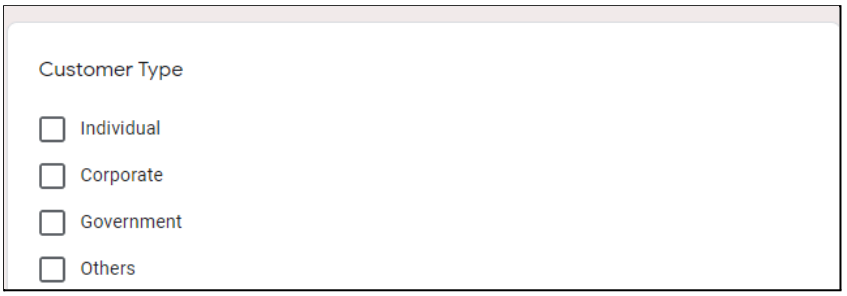
- The Database should be compact , neat and easily accessible and should be user friendly and Window primarily based totally in order that no specialised and great mastering is required.
- It should contain all information about vehicles , customers , employees, employee's salary , stock , rating system , pick up / drop facilities options etc.
- The system needs to maintain quantity records, customers records , update and delete records , search area and also security system to prevent misuse of data.
- We should manage efficient service order processing and billing designed for any given number of orders.
- There should be proper registration or login / signup system for the convenience of the customer.
- There should be a separate database for the workers/employees so that the details of the employees can be stored, also their progress report can be viewed and their salary and needs can be taken care of.

- In the case of insurance we should make split payments for car owners and insurance companies. Also provides necessary documents for it.
- The system has adequate scope for modification in future if it is necessary.
- System should give appropriate access to the authorized users depending on their permission .
- During the run time of the system if a user has input any wrong data or some error has occurred then the system should inform him/her.

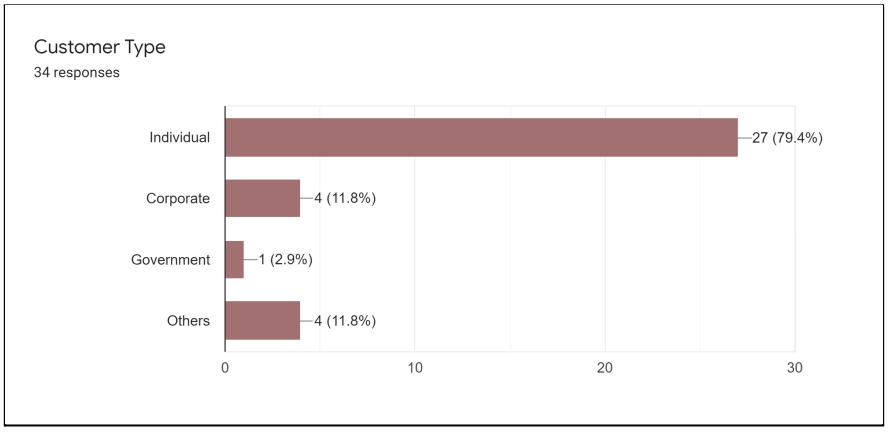
2.3 Questionnaires

We have collected 34 responses for our SRS project through google form. Following are the outcomes of our survey with some observations and google forms structure.

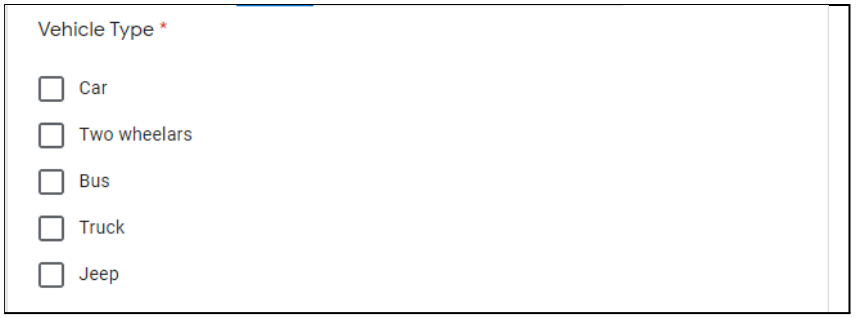
1. Customer Type:



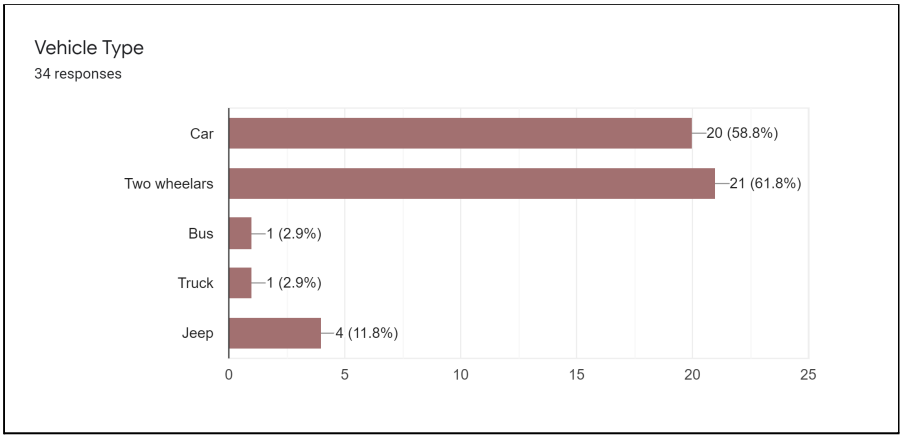
Response:



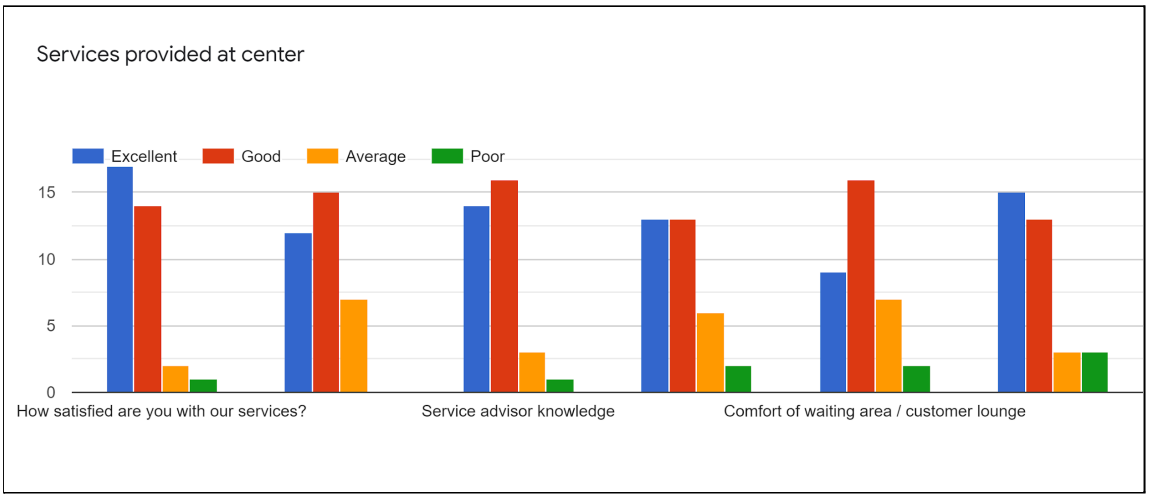
2. Vehicle type:



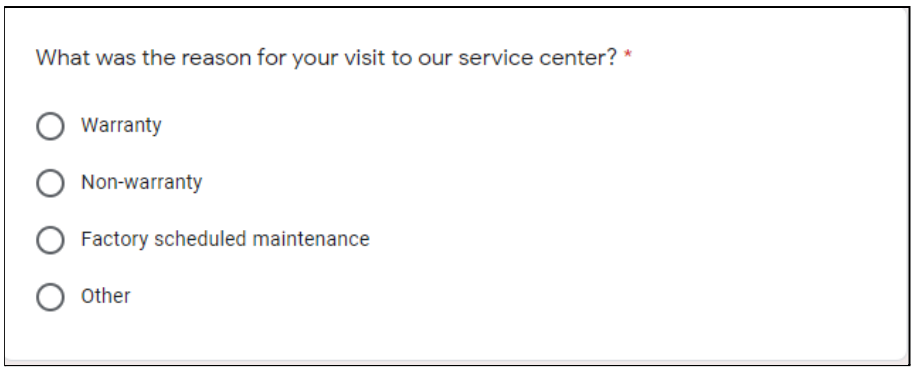
Response:



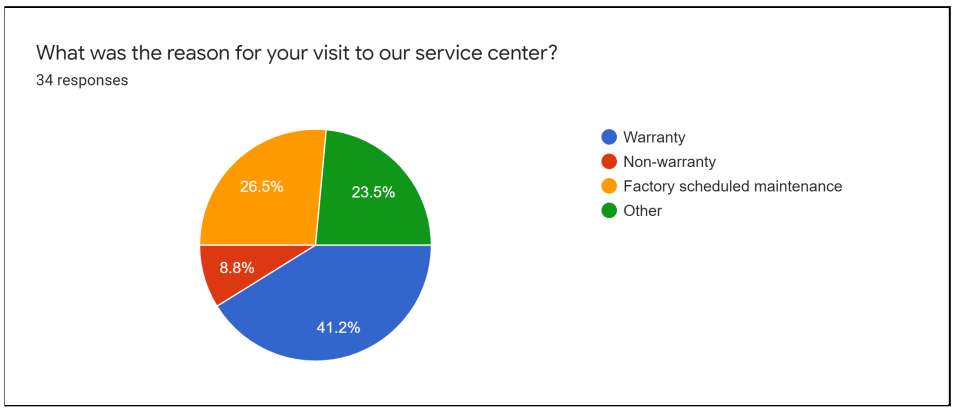
3. Service Provided at our center:



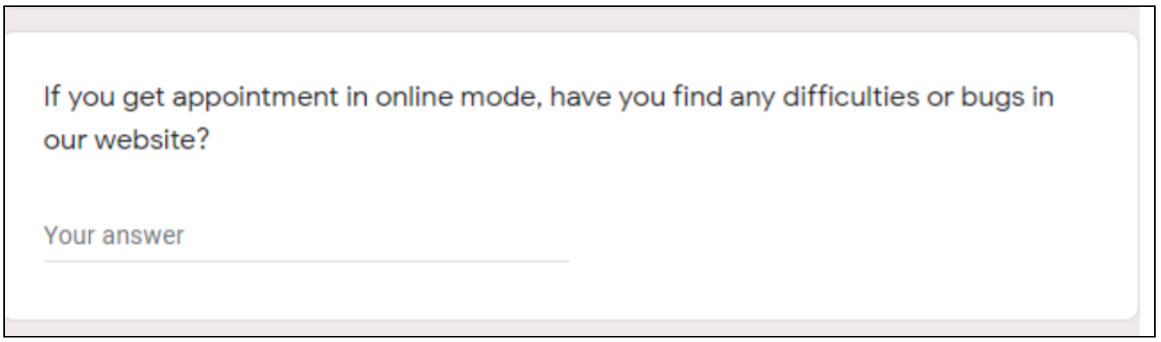
4. What was the reason for your visit to our service center?



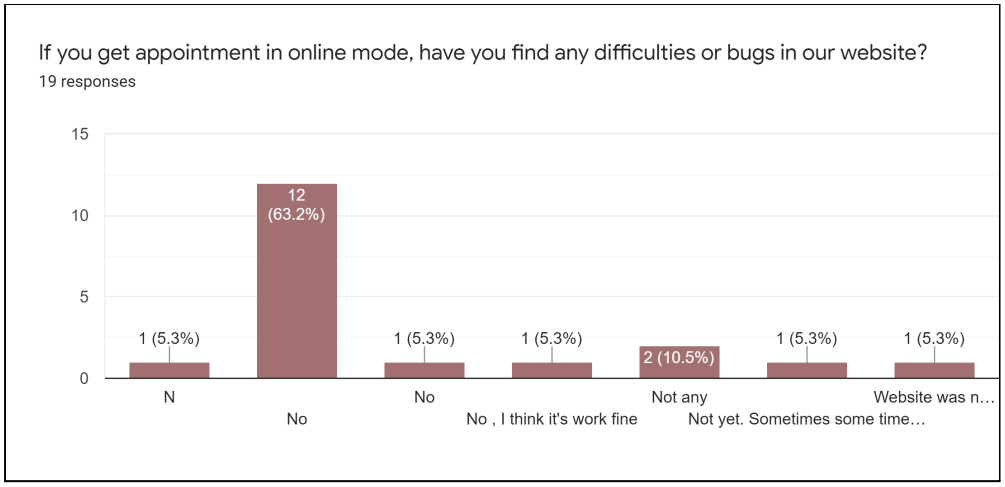
Response:



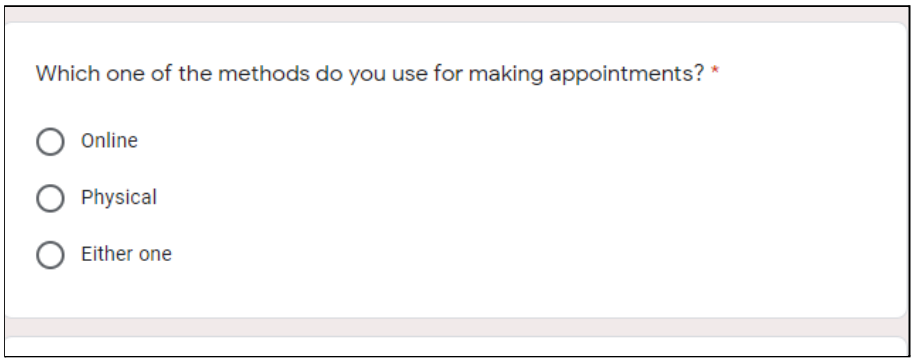
5. If you get an appointment in online mode, have you found any difficulties or bugs in our website?



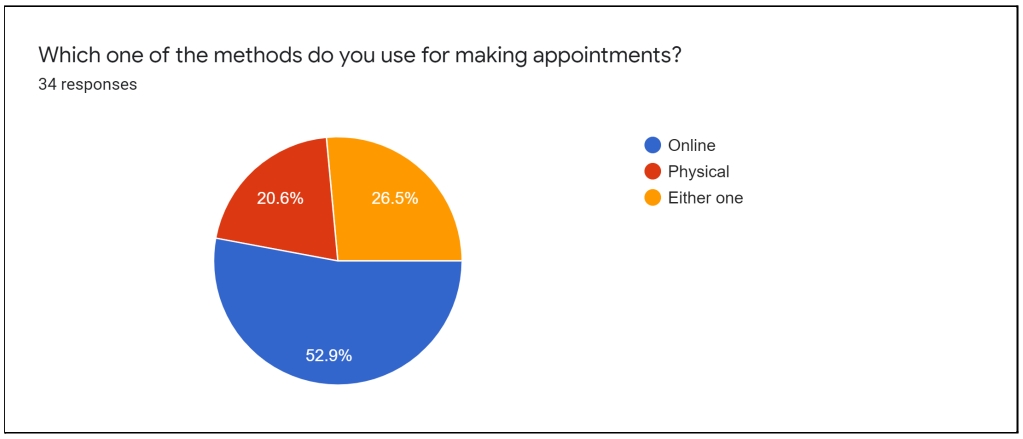
Response:



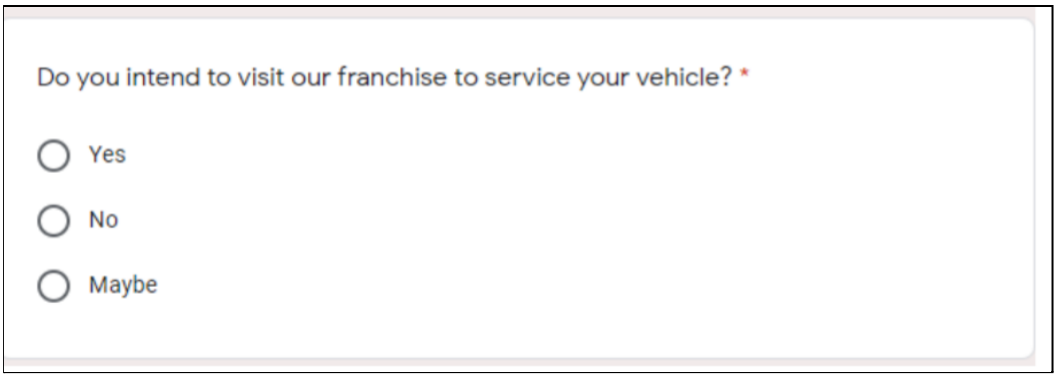
6. Which one of the methods do you use for making appointments?



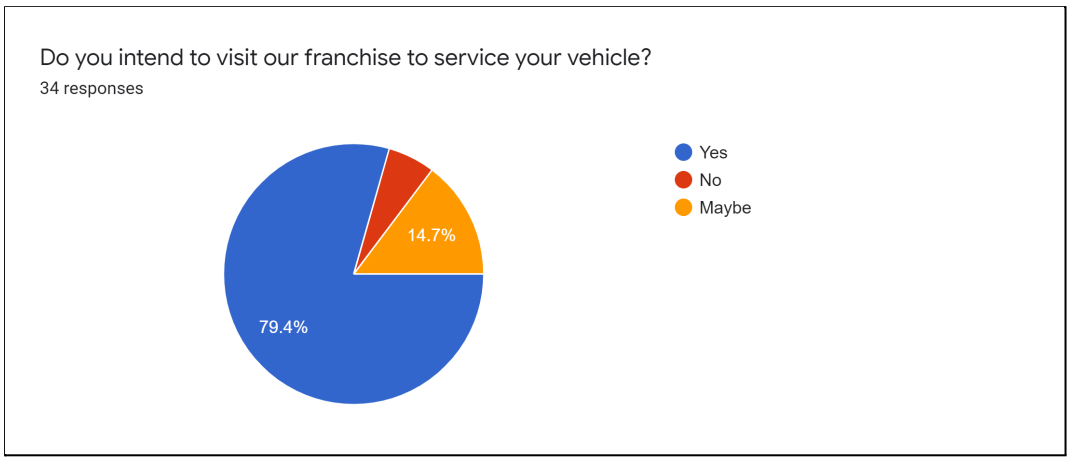
Response:



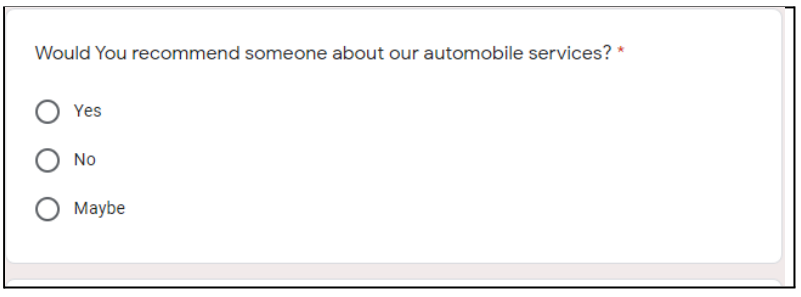
7. Do you intend to visit our franchise to service your vehicle?



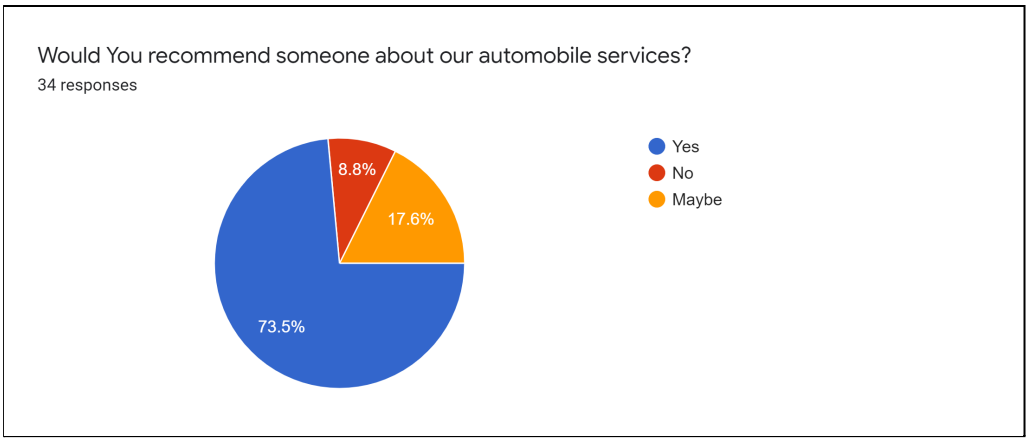
Response:



8. Would you recommend someone about our automobile services?

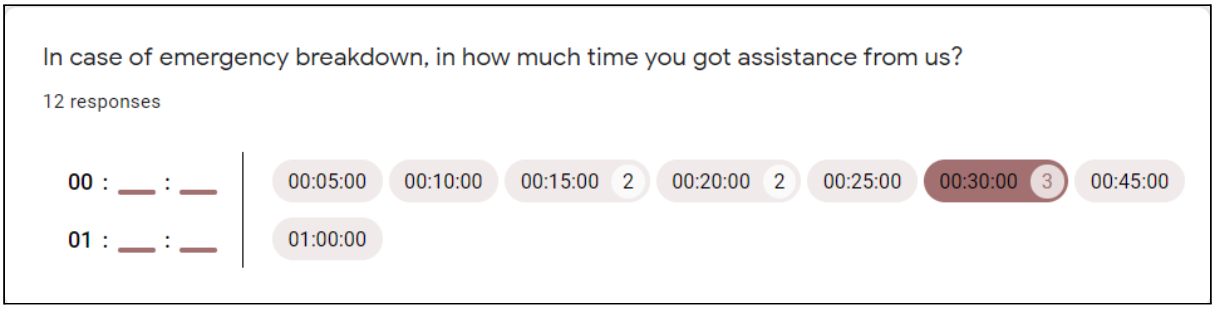


Response:



9. In case of emergency breakdown, in how much time you got assistance from us?

Response:



Short answer-based Question and their responses:

- Question:
    - What are the things that you like most about our service?
- Answers:
    - Service management done so nicely
    - Low cost
    - Quick response. Helpful and polite guidance
    - Online bookings and payments
    - Double verification and regular service alerts
    - Regular service alerts
    - Breakdown service
    - Fast service
    - Fast servicing and easy registration process
    - Vehicle home delivery and pick up

- Question:
    - What would you recommend we do to improve our services or improve the customer's experience:

- Answers:
    - Need to provide photographs of vehicle parts which were replaced
    - List contact details for all nearby service centers

2.4 Observations

System : Automobile Service Centre

Observations by: Parita Makvana (Business Marketing Manager)

Date: 10/10/2021 **Time:** 14:30

Duration: 30 minutes **Place:** Telephonic call

Observations:

- Most of the users are satisfied with this automobile service.
- They are impressed by the online booking system and quick response given by service providers in case of emergency breakdown.
- Large number of customers have serviced their cars and two wheelers and are happy to continue further service. They are also in favour of suggesting this service to others.
- There was high demand for authorization of the customers before servicing.
- The CEO would love to know about individual centres progress for a healthy competition.
- Hardworking employees should be found and promoted/appreciated.

3. Fact-Finding Chart

Objective	Technique	Subject(s)	Time Commitment
To get background on Automobile Service centers working Procedures	Background Reading	Website - Mericar.com and Blog	1 day
To get information about App features and working	Background Reading	Play Store App GoMechanic	0.5 day
To know about the requirements and software facilities	Interview	Role Play(Service Advisor)	45 minutes
To establish security system and fulfilling additional requirements	Interview	Role Play (Manager)	1 hour
To get information about business strategies, Working Flow	Interview	Role Play(CEO)	1 hour
Survey regarding Customers feedback	Questionnaire	Google form	2 days
To get one on one experience about the operations of an existing system	Observation	All Above	30 minutes

4. Combined Requirements

4.1 Product Requirements

- A table with the personal information of all the past customers in the schema “Customer”.
- A table for all the vehicles whose service has been done in the past along with their owner details in the schema “Vehicle”
- Separate table for employee personal details and the points they have earned so as to get a rough idea of their performance.
- Table for showing the available slots for a type of vehicle in the schema “Slots”.
- Table for keeping track of the past services with their date and time information in the schema “Track”.
- A table for keeping stock information of the materials required in the servicing center in the schema “Stock”.
- A table that would help the CEO to know which centre is doing better at the national level in the schema “progress”.
- A table that would help the agents to give regular servicing alerts to the customer along with taking feedback in the schema “Alert”.

4.2 Safety Requirements

If due to some catastrophic failure there is extensive damage to a large portion of the database then the recovery method would restore a past copy of database that was saved to any archival storage before and would reconstruct the current state by again applying and doing the operations of committed transactions from saved log, upto the time of failure.

4.3 Security Requirements

- The special requirement from the perspective of the security is that the user's interaction with the database is such that the developers are able to hide internal irrelevant details from the users. For this there are three levels of abstraction: Physical level, conceptual level and view level. The physical level describes the actual storage in the memory. These are often hidden from the programmers.
- Hence we would create the system in such a way that only the CEO would be able to see the progress of various centres across the nation and the individual centres won't have access to this information. so

that this would not demotivate the centres lagging behind and would not make the succeeding centres overconfident. These would have some privacy as well.

- Also the agents of a particular centre would only have access to their centres customers information only.

4.4 Interview Requirements

- Requirements like services given till date, present stock, regular service alerts, centres progress, agent points, slots availability etc.
- Separate schema for customers to access data that has no confidential information of the company.
- Proper privileges to be given to the company's CEO and the centre manager.
- Data regarding the available stock of materials like engine oil, cleansing water etc to be accessible by the agents.
- A table to add new customers by assigning them a new id and checking for duplicates.
- Privileg to the Managers to check whether the agents are not giving any free service to their closed ones and are complying to the rules of the company.
- Verification to cross check the authorized customers via ID and phone number (OTP).

4.5 Observations Requirements

➔ Maximum customers (52.9%) preferred online mode for booking services.

➔ Most of the people were satisfied with the prior knowledge of the services and charges explained by the agent.

➔ The agents too were satisfied by the feedback system that would help them in their promotion.

➔ Everyone had their own thoughts like:

1. Some were extremely satisfied by the regular servicing alerts while others found this not so attractive.
2. Some customers were loving the OTP cross verification system while agents might not be happy with this as they were compelled to follow the rules.
3. The managers were happy knowing the progress of each agent.
4. Maximum customers were getting breakdown services within 30 minutes which was very useful for them.
5. The company was happy with the stock alerts given by the system regularly.

4.6 Software Quality Attributes

- **Availability:** The system should be available to multiple users concurrently as many customers might be booking slots and paying at same time.
- **Usability:** The booking slots and related information should satisfy a maximum number of customer needs.
- **Correctness:** The client should get correct information about their serving and the employees should also get correct information regarding stock.
- **Maintainability:** The system and the managers in charge should maintain correct records of past servicing and customers.

4.7 Removing duplicates and prioritizing the requirements

PRIORITY	REQUIREMENTS	FREQUENCY
1	Regular service alerts based on the last service history	3
2	Break down service to the customer	2
3	Feedback system	2
4	Double cross verification via customer ID and OTP to avoid frauds	2
5	Stock updates and alerts	1
6	Progress of every center at national level	1

5. User Classes and Characteristics

User classes are listed below:

- CEO of the company-
  This user class will have all privileges and can access every detail about every service centre. So that he/she could check customer reviews and make necessary changes in the service centre and also track employees' records.

- Developers-
  Developers will upgrade the database system as per new arising requirements. They will maintain the system and keep it secure so that unauthorized people don't access our database.

- Project managers-
  They will make sure employees and agents are working smoothly and customers don't have to face more difficulties.

- Marketing staff-
  They will analyse customers' reviews and will update services accordingly.

- Users-
  Users won't have all privileges but they can give feedback about whatever service they received.

6. Operating Environment

This Database can be deployed on Linux or Windows operating systems with Postgresql server with some basic hardware and software requirements like RAM, System version, Browser etc.

Server Side: Hardware and software

- Processor: Intel core processor 2 GHz
- RAM: 2 GB RAM
- Hard Disk: 80 GB HDD
- Monitor: Compatible Printing Device
- Internet Connectivity with Ports configured
- Keyboard, Mouse: Any Mouse Software
- Operating System: Microsoft Windows 7/8/8.1/10
- Database: Postgresql
- Browser: Google Chrome, Mozilla Firefox etc.

- We will need an efficient and optimized software program application in order that we can perform query operations for instant data retrieval from the database.
- The **system application has to** be **highly** scalable, as **we're planning to apply this system** to facilitate a service that tracks the **information** of **people everywhere in the** country.

Client side: Hardware and software

- Processor: Intel core i3 or higher processor 2 GHz
- RAM: 520 MB RAM
- Hard Disk: 40 GB HDD
- Monitor: Compatible Printing Device
- Internet Connectivity with Ports configured

● Postgresql Server

● Browser: Mozilla Firefox or Google Chrome, etc.

7. Product Functions

- **Access() -** Our System has been developed to troubleshoot Manual system Problems. This System keeps the data in a centralized way which is available to all the users simultaneously.

- **User_registration() -** The system is reduced as much as possible to avoid errors while entering the data. First the customer need to login on the site and we check that the particular user is already exist in our database or not by their customer id, and if it is exist then we do not need to update its entry , but if it's not exist then he/she has to register first on the site then we add customer details in our customer table and after that we proceed further.

- **Check_availability() -** Our database keeps records of the customers, vehicle , service , payment, Automobile spare parts stock, employee details, service center location , service duration etc. For the customers , there will be a search bar for searching for availability of the slots at a particular service center.

- **To_alert() -** We will have a feature that lets some data be shared with other service centers so that all employees can also access other people's data. This would help employees determine which customer needs to have its car service done so that they can send regular alerts for services.

- **User_feedback() -** We would also include feedback from users' side, the amount paid by the customer for servicing and the number of times the customer has visited the service center for car service. So that this information can be useful to us whenever the need arises.

- **Review_progress() -** We will have a feature that The CEO of the Company can Review the Progress report of the particular Center.

8. Privileges

- Customers can see only the available slots at the particular service center. If slots are available then only they can login/register and book their slot.

- Customers, Vehicles and stock details will be accessible by Employees/Service agents. They are able to see the past history of the customers, information about regular services alerts and customer feedback. Customers can't read/update this information.

- Managers have rights to read and update the employee details according to their needs. Manager Can't access Progress Table.

- Only The CEO of the company has the privilege to know about the progress of different service centers. He can access the whole database and can change it if needed. No other user classes have permission for it.

9. Assumptions

- We assumed that we would have a separate database interface for every sub-objective. For example, customers, employees, agents, managers, CEO of the franchise.
- We did this to make customer service easier, instant roadside assistance and to make a more organized and well defined structure of the service center.
- For the purpose of its workability, we will input random data.
- Since this will contain web based appointments there is a need for the internet browser. It is assumed that the user is familiar with the internet browser.
- The service center will service all types of vehicles, so we assume that the employees have knowledge and can service all of them.
- In case of an emergency breakdown faced by a customer, car towing service will be provided from a nearby service provider, that is assumed.
- We based our questionnaire only on those customers who had taken our service at least one time.

10. Business Constraints

- The most important would be online booking and paying features for the multiple customers concurrently 24x7. Its uptime cost will be heavy, because if implemented it will be heavily used by customers.
- The other one that we figured out was “Security”. As a lot of personal information will be in it , it will be constantly attacked by hackers. Therefore a big costly 24x7 security team is a must for surveliiance.

Section 2: Noun Analysis

Extracted Nouns & Verbs from Problem Description

Table 1

Noun	Verb
Customer	Books
Vehicle	Service
Employee	Authentication
Slot	Notify
Stock	Complain
Track	Updates
Pick Up Type	Appreciation
Progress	Maintains
Manager	Available
CEO	checks
Agent	Gives
Customer Id	manages
Customer Contact	Review
Points	Owns
Authorization	Verify
Company	contains
Service ID	Supply
Reports	Connect
Payment	received

Insurance	Uploading
Vehicle No.	Read
Time	Implement
Customer waiting area	facilitate
Spareparts	Collect
Location	Provide
Email	Limit
Bills	Pick Up
Accounts	Create
Manual System	Work
Database	Alert
Age	Assistance
Address	Keeps
Internet	Records
No. of centers	consumes
Duration	Changes
System	Gets
Credentials	
Vehicle Name	
Managerial Requirements	
Drivers	
Location ID	
Salary	

Date	
Vehicle Type	
Employee ID	
Employee Name	
Designation	
Emp Contact	
Availability	
Track ID	
Start Date	
End Date	
Starting Time	
Ending Time	
Amount	
No of Services	
Service Type	
Engine Oil	
Coolant	
Cleaning Water	
Vehicle Id	
Employee Contact	
Feedback	

Status	

Accepted Noun & Verbs list

Table 2

Candidate Entity Set	Candidate Attribute Set	Candidate Relationship Set
Customer	Customer_id Customer_name Customer_age Customer_address Customer_email	owns
Vehicle	Vehicle_no Vehicle_name Vehicle_type Duration RC_No	gets
Employee	Employee_id Employee_name Employee_contact Service_location Salary Designation	emp_loc emp_service
Slot	Loc_id	

	Vehicle_type Availability	checks
Track	Track_id Loc_id Vehicle_no Start_date End_date Amount Feedback	service_track
Progress	Loc_id Current_year Points	rates
Stock	Loc_id Coolant Engine_oil Cleaning_water	updates
Alert	V_no Alert Pickup_type	gives
Sevice	Date Vehicle_no Employee_id Service_type Charges No_of_services	gives gets
Loc_info	Loc_id Service_location	contain

Rejected Noun and Verbs list

Table 3(Rejected Nouns)

Noun	Reject Reason
Manager	General
CEO	General
Agent	Duplicates
Authorization	Vague
Company	General
Reports	Duplicates
Payment	General
Insurance	Irrelevant
Accounts	Duplicates
Time	Duplicates
Customer waiting area	Vague
Spareparts	Vague
Manual System	General
Database	General
Internet	General
System	Vague
No of Centers	Irrelevant
Credentials	Irrelevant
Managerial Requirements	General

Drivers	Irrelevant
Service Id	General

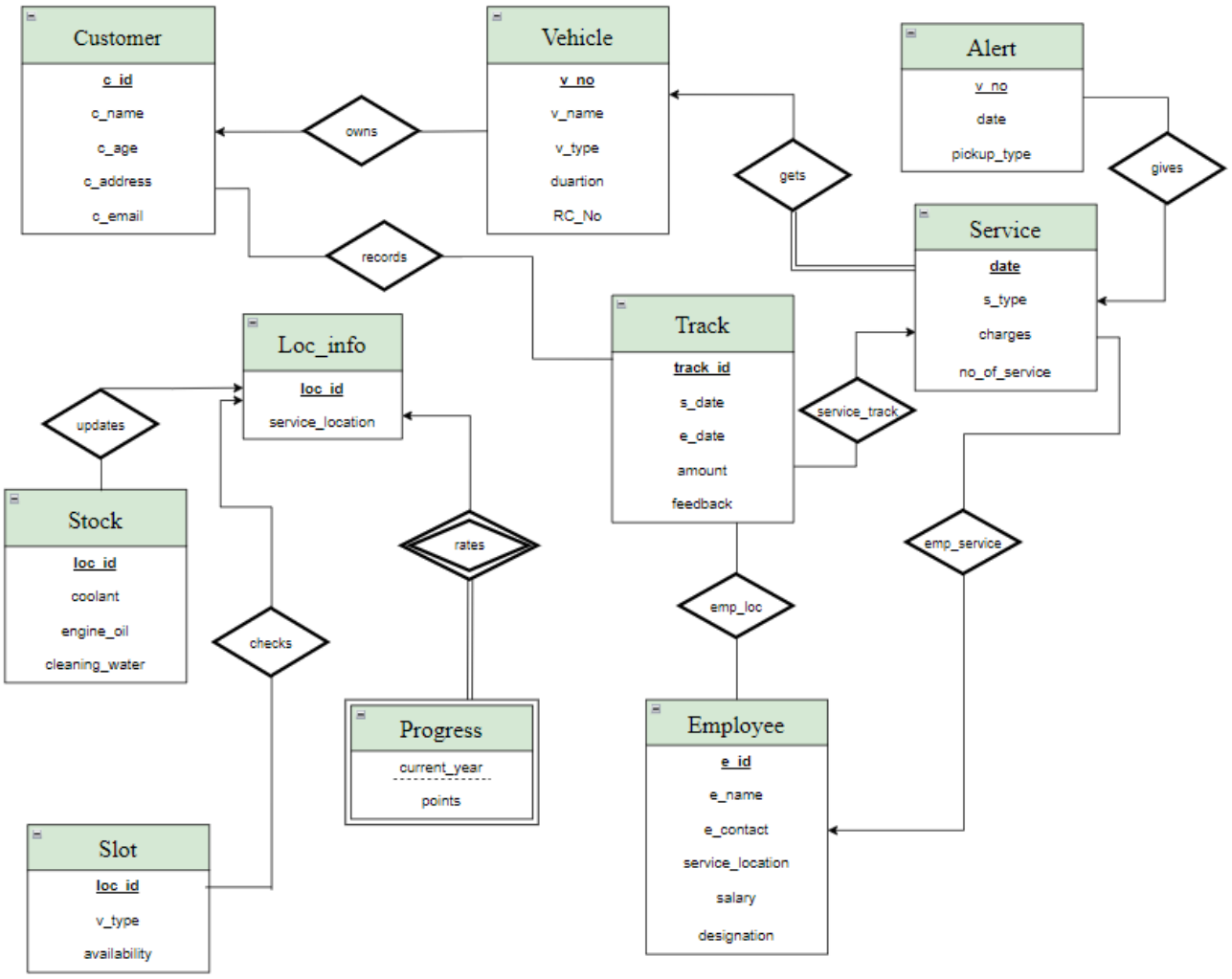
Table 3 (Rejected Verbs)

Verb	Reject Reason
Authentication	Vague
Complain	General
Appreciation	Irrelevant
Available	Vague
Manages	Vague
Review	General
Verify	General
Connect	Vague
Supply	Duplicates
Received	Vague
Provide	Vague
Read	General
Implement	Irrelevant
Facilitate	General
Collect	Vague
Limit	Irrelevant
Pick Up	General

Create	General
Work	Duplicates
Uploading	Irrelevant
Maintain	Duplicates
Notify	Duplicates
Consumes	Irrelevant
Keeps	General
Changes	General
Assistance	Irrelevant

Section 3
ER-Diagrams all versions

ER diagram version - 1



Improvisation in ER Diagram

1. Identify Entity Types

A. Types of Entities

- In our ER model version 1, ‘Progress’ is the only weak entity set dependent on the strong entity ‘Loc_info’ via identifying relationship ‘rates’.
- However we realised that stock and slot are such entities that have a common attribute ‘loc_id’ and both inherit this from the parent table ‘Loc_info’. Therefore in the ER version 2 we converted ‘stock’ and ‘slot’ to weak entity sets dependent on the ‘Loc_info’ associated via the identifying relationship ‘updates’ and ‘checks’ respectively.
- Similarly ‘Alert’ entity gives alert to vehicle number according to its last service and hence this was converted to weak entity dependent on the ‘Service’ entity.

B. Types of Relationships

- Aggregation - In our ER model, there are no two relations between the same entity sets. Hence we need not form any aggregation.
- Recursive - In our ER model, there is no relation on an entity set itself. Hence there is no recursive relation.
- Simple association links
    - Customer and Vehicle are related by one-many relationship as a single customer may have multiple cars but a particular vehicle number would belong to one customer only.
    - An employee will service multiple vehicles and hence it would be repeated many times in the track table. Also multiple past service (track) of a customer might have been followed up by different employees. Hence there is a many-many relationship between employee and track.

2. Identify Relationship Types

A. Entity vs Attribute

- Entities are a set of attributes. If an attribute of a user class/entity needs to have its own different properties/structure, the attribute is converted into an entity having a relationship with the corresponding user class/entity. In some cases, even if an attribute needs to be multivalued, the attribute is converted into an entity.
- In our ER diagram, a ‘vehicle’ could be an attribute of the Customer’ entity set, but we converted it into an entity because every vehicle will have its own properties and specifications. Also a single customer may have multiple vehicles.

B. Entity vs Relationships

- Entities and relationships both can have their own attributes, but relationships are associations between two or more entities.
- ‘Customer’ entity set is self sufficient whereas we need a relationship between ‘Customer’ and ‘Vehicle’ to understand which customer owns which vehicle.
- Similarly, there is a relationship between ‘Employee’ and ‘Service’ entity sets to tell which employee followed up service on a particular date.
- ‘Track’ entity set has a relationship with ‘Customer’ as well as ‘Employee’ because a particular customer's service record would be stored in the track table but the details of the employee who followed that service would be there in the ‘employee’ entity. this would help in case of feedback.

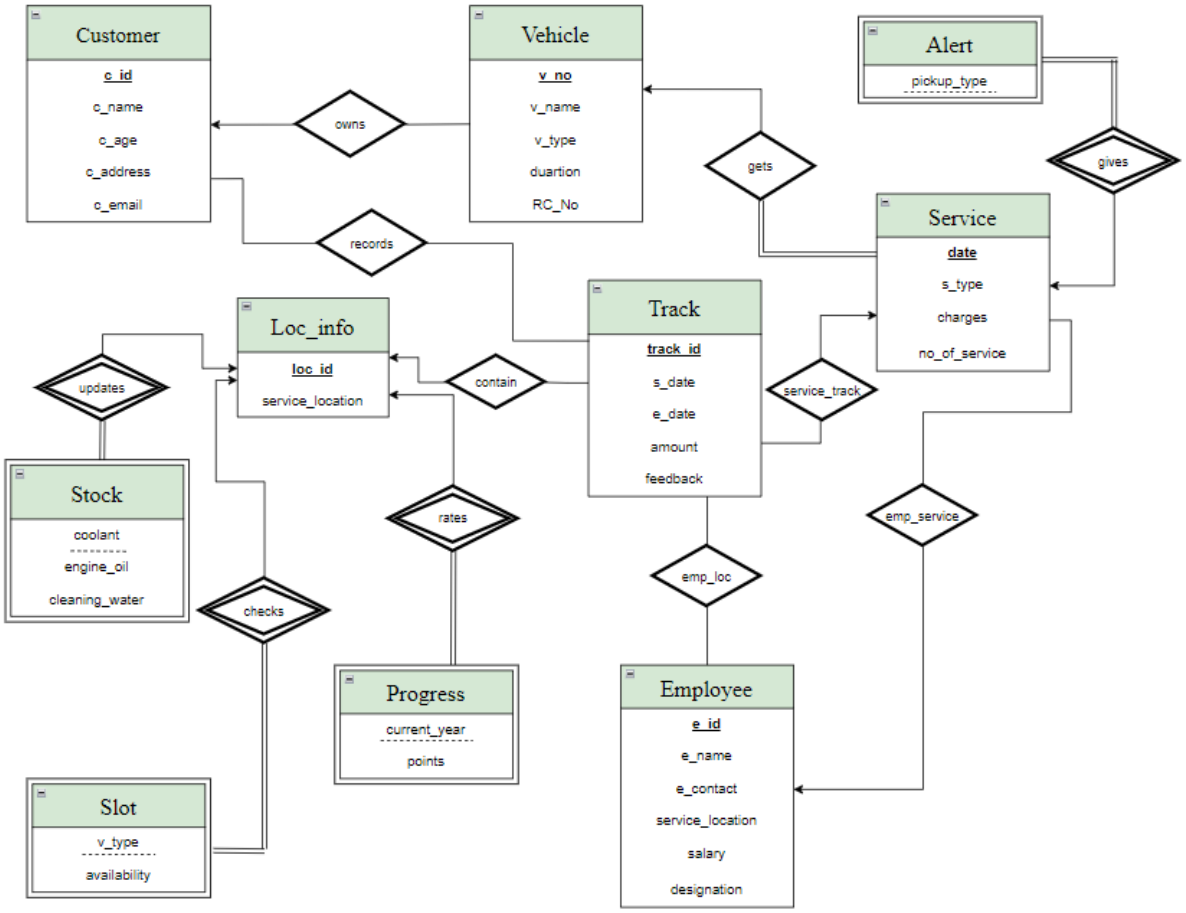
C. Binary vs Ternary Relationships vs Aggregation

In our ER model, there are no ternary relationships or any aggregation. Since we had utmost 2 tables required to be associated with each other in every relationship therefore we required only binary relationships.

D. Total Participation

1) ***Stock*** has total participation with ***Loc_info*** because the location whose stock is mentioned would definitely be present in the ***Loc_info*** relation. Similarly, the ***slot*** and ***Loc_info*** have a total participation. This is also because these entities are related via an identifying relationship between a weak and a storing entity.
2) ***Progress*** has total participation with ***Loc_info*** since whichever service centres are present in the progress tables and have received some points would be definitely there in the ***Loc_info*** table.
3) Similarly alert would be given to only those vehicles that would be surely present in the past service records table. Thus ***alert*** and ***service*** have total participation

ER diagram version 2 (Final Version)



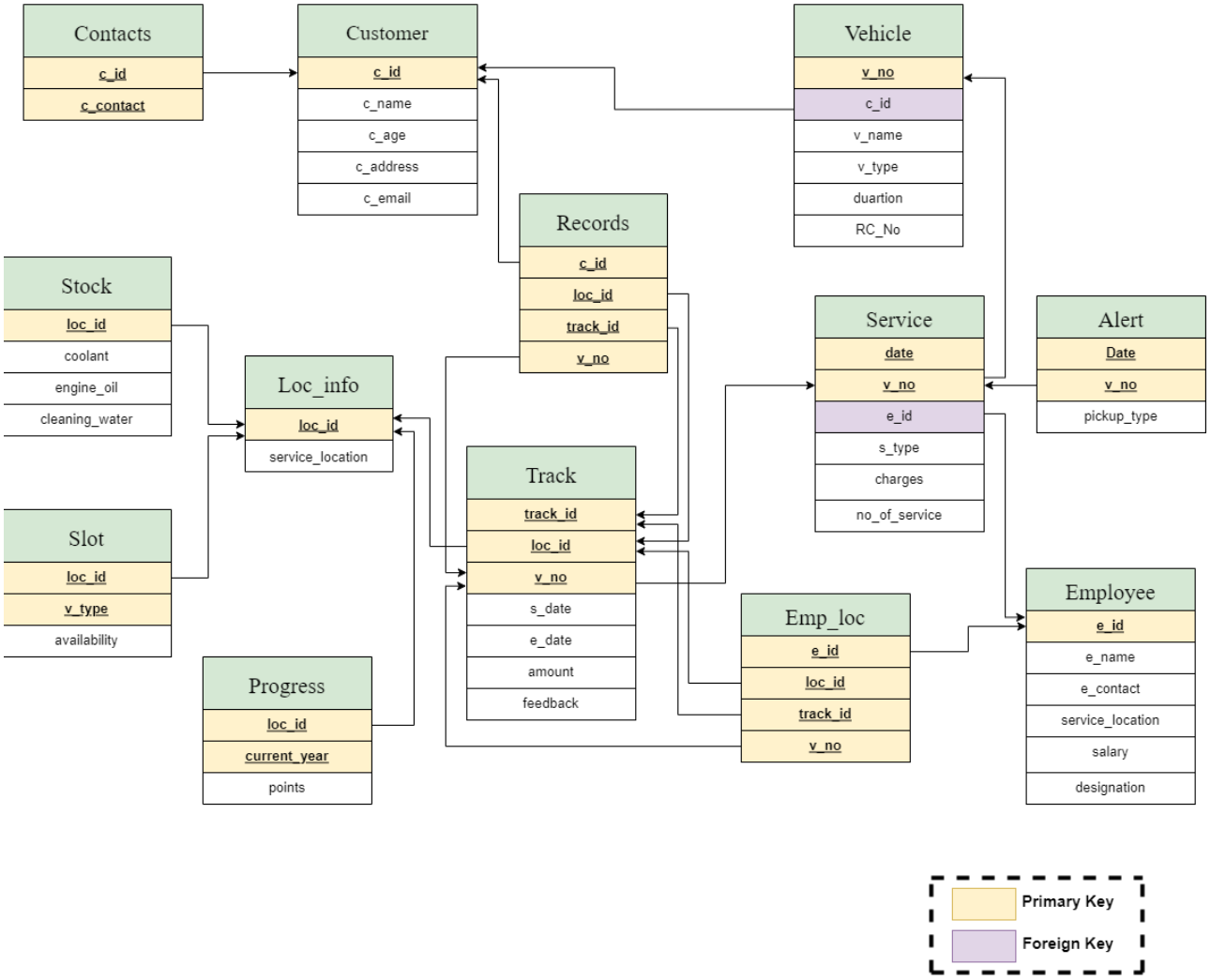
Section 4:

Conversion of Final ER-Diagram to Relational Model

1. Mapping ER to Relational Model

- Customer (**c_id**, c_name, c_age, c_address, c_email)
- Contacts (**c_id**, **c_contact**)
    - FK c_id references customer
- Vehicle (**v_no**, c_id, v_name, v_type, duration, RC_No)
    - FK c_id references customer
- Employee (**e_id**, e_name, e_contact, service_location, salary, designation)
- loc_info(**loc_id**, service_location)
- Slot **(loc_id**, **v_type**, availability)
    - FK loc_id references loc_info
- Stock (**loc_id**, coolant, engine_oil, cleaning_water)
    - FK loc_id references loc_info
- Service (**v_no**, **date**, e_id, s_type, charges, no_of_service)
    - FK v_no references vehicle
    - FK e_id references employee
- Track (**track_id**, **loc_id**, **v_no**, s_date, e_date, amount, feedback)
    - FK loc_id references loc_info
    - FK v_no and s_date references service
- Progress (**loc_id**, **current_year** , points)
    - FK loc_id references loc_info
- Alert (**date**, **v_no**, pickup_type)
    - FK date, v_no references service
- Emp_loc (**e_id**, **loc_id**, **track_id**, **v_no**)
    - FK e_id references employee
    - FK loc_id, track_id, v_no references track
- Records (**c_id**, **track_id**, **loc_id**, **v_no**)
    - FK loc_id, track_id, v_no references track
    - FK c_id references customer

Relational Model (before normalization)



Section 5
Normalization and Schema Refinement

1. Normalization

Normalize the database up to 1NF (scalar values), 2NF(Remove Partial Dependencies), and 3NF/BCNF (Remove Transitive Dependencies)

-> Customer (**c_id**, c_name, c_age, c_address, c_email)

- **Primary Key**: c_id → c_name, c_age, c_address, c_email

Note* - Here we have assumed c_name to be First name only. Hence c_name is not a multivalued attribute.

Thus this is in the BCNF form because

- Since there are no multivalued attributes, it's in the 1NF.
- There is no composite key and every other attribute is dependent on the PK only. Hence this is in 2NF.
- Here none of the non-primary attributes are dependent on any other non-prime attributes. Thus in the 3 NF.
- Since the LHS in the dependencies is candidate key = primary key = c_id only.

  Thus this is in its BCNF form.

-> Contacts (**c_id**, **c_contact**)

- **Primary Key:** (c_id, c_contact) → (c_id, c_contact)
- **Foreign Key:** c_id

Thus this is in the BCNF form because

- Since there are no multivalued attributes, it's in the 1NF.
- There is a composite key but no other attributes exist. Hence this is in 2NF.
- Here both attributes are prime attributes. There is no transitive dependency. Thus in the 3 NF.
- Since the LHS in the dependencies is candidate key = primary key = (c_id, c_contact) only.

  Thus this is in its BCNF form.

-> Vehicle (**v_no**, c_id, v_name, v_type, duration, RC_book)

- **Primary Key**: v_no → c_id, v_name, RC_book
- **Foreign Key**: c_id

- Since there are no multivalued attributes, every attribute is in its atomic form. Hence it's in the 1NF.
- There is no composite key hence no partial dependency. Thus in the 2NF.
- But here there is a transitive dependency which violates the 3NF.

- **Transitive**:

  v_no → v_name

  v_name → v_type

  v_type → duration

  So it has three determinants, v_no , v_name, v_type

- **Redundancy**: Here many customers can own the same type of vehicle, hence v_name and v_type are redundant.
- **Insert Anomaly**: We can not insert a particular vehicle type and its name without assigning v_no.
- **Delete Anomaly**: If a sole tuple corresponding to a particular vehicle name gets deleted then all its related information such as v_no , v_type, duration is also lost.
- **Update Anomaly**: Duration for a particular vehicle type appears many times in a table , so If we want to change the duration, it may need changes in so many tuples.

Hence to avoid this , we need to break down Vehicle table into three parts,

-> Vehicle(**v_no**, c_id, v_name, RC_No)

-> Vehicle_type(**v_name**, v_type)

-> Vehicle_duration(**v_type**, duration)

Here, in the above three tables, there are no multivalued attributes , no partial dependency, and no transitive dependency.

And LHS in the dependencies is the candidate key = primary key.

Thus these three relations are in its BCNF form.

-> Employee (**e_id**, e_name, e_contact, service_location, salary, designation)

- **Primary Key**: e_id → e_name, e_contact, service_location, salary, designation

Note* - Here we have assumed an employee has only one official contact number. Hence e_contact is not a multivalued attribute.

Thus this is in the BCNF form because

- Every attribute is atomic hence it's in the 1NF.
- There is no composite key hence every other attribute is dependent on the PK only. Hence this is in 2NF.
- Here none of the non-primary attributes are dependent on any other non-prime attributes. That is, no transitive dependencies exist here. Thus in the 3 NF.
- Since the LHS in the dependencies is candidate key = primary key = e_id only. Thus this is in its BCNF form.

-> loc_info(**loc_id**, service_location)

- **Primary Key:** loc_id → service_location

- Since there are no multivalued attributes, it's in the 1NF.

- There is no composite key and every other attribute is dependent on the PK only.

  Hence this is in 2NF.

- Here none of the non-primary attributes are dependent on any other non-prime attributes.

  Thus in the 3 NF.

- Since the LHS in the dependencies is candidate key = primary key = loc_id only.

  Thus this is in its BCNF form.

-> Slot (**loc_id**, **v_type**, availability)

- **Primary Key: (**loc_id, v_type) → availability

- Here every column would have an atomic value. Thus this is already in its 1 NF.
- There is a composite key and since the availability cannot be determined by the loc_id or v_type 'alone' but by both combined, hence no partial dependency and therefore the relation is in its 2 NF.

- Availability is dependent on candidate keys only hence there is no transitive dependency and this is in its 3rd Normal form.
- Eventually, the LHS of the dependency is a primary key = candidate key and hence the relation is in its BCNF.

-> Stock (**loc_id**,, coolant, engine_oil, cleaning_water)

- **Primary Key:** loc_id → coolant, engine_oil, cleaning_water
- **Foreign Key:** loc_id

Since there are no multivalued attributes, it's in the 1NF.

There is no composite key and every other attribute is dependent on the PK only. Hence this is in 2NF.

Here none of the non-primary attributes are dependent on any other non-prime attributes.

Thus in the 3 NF.

Since the LHS in the dependencies is candidate key = primary key = loc__id only.

Thus this is in its BCNF form.

-> Service (**v_no**, **date**, e_id, s_type, charges, no_of_service)

- **Primary Key:** (v_no, date) → e_id, s_type
- **Foreign Key:** e_id
- **Redundancy:** Here , for every service of several vehicles , the service charges are repeated ,so it's redundant.
- **Insert Anomaly**: We can not insert a service type without assigning v_no.
- **Delete Anomaly**: If a sole tuple corresponding to a particular vehicle number gets deleted then all its related information such as s_type , charges is also lost.
- **Update Anomaly**: Service charges for a particular service type appear many times in a table , so If we want to change the service charge, it may need changes in so many tuples.
- There are no multivalued attributes so we can say it's in 1NF.
- Here, no_of_services is only dependent on v_no. So, this is partial dependency. Hence, we will need to divide it into two tables.

-> Service (**v_no**, **date**, e_id, s_type,charges)

-> Ser_records (**v no**, no_of_service)

- Here , in the above two tables, there is no partial dependency. Hence these both are in 2NF.
- In the service table, non prime attribute charges are dependent on another nonprime attribute s_type. Hence the transitive dependency.
- To avoid this , we need to break down table into two tables,

  -> Service (**v_no**, **date**, e_id, s_type)

  -> Ser_charges(**s_type**, charges)

- Since the LHS in the dependencies is candidate key = primary key only. Thus these are in its BCNF form.

-> Track (**track_id**, **loc_id**, **v_no**, s_date, e_date, amount, feedback)

- **Primary Key:** (track_id, loc_id, v_no) → s_date, e_date, amount, no_of_service, feedback
- **Foreign Key:** v_no
- **Insert Anomaly**: We cannot directly insert vehicle number in the past track record without mentioning the location where it was serviced.
- Each column contains atomic values. Therefore this is in its 1st Normal form.
- Here the amount is partially dependent on v_no and on the servicing date. Hence to avoid this we would have separate table;-

  Track (**track_id**, **loc_id**, **v_no**, s_date, e_date, feedback)

  Bill (**date, v_no,** amount)

- (track_id, loc_id, v_no) → s_date, e_date, feedback
- (date, v_no) → amount
- There is no transitive dependency and hence this is in 3rd Normal form.
- The left hand side of the dependencies is the candidate keys and thus this table is in BCNF form.

-> Progress (**loc_id**, **year** points)

- **Primary Key:** (loc_id , year) → points
- **Foreign Key:** loc_id

Thus this is in the BCNF form because

- Every attribute is atomic hence it's in the 1NF.
- There is a composite key and every other non prime attribute is dependent on the composite PK only. Hence this is in 2NF.
- Here none of the non-primary attributes are dependent on any other non-prime attributes. That is, no transitive dependencies exist here. Thus in the 3 NF.
- Since the LHS in the dependencies is candidate key = primary key = (loc_id,year) only. Thus this is in its BCNF form.

-> Alert (**date**, **v_no**, pickup_type)

- **Primary Key:** (date, v_no) → pickup_type
- **Foreign Key:** v_no

This is in the BCNF form because

- Since there are no multivalued attributes, it's in the 1NF.
- There is a composite key, but every other attribute is dependent on the composite PK only. Hence this is in 2NF.
- Here none of the non-primary attributes are dependent on any other non-prime attributes. Thus in the 3 NF.
- Since the LHS in the dependencies is candidate key = primary key = (date,v_no) only.

  Thus this is in its BCNF form.

-> Emp_loc (**e_id**, **loc_id**, **track_id**, **v_no**)

- **Primary Key:** (e_id, loc_id, track_id, v_no) → (e_id, loc_id, track_id, v_no)
- **Foreign Key:** e_id, loc_id, track_id, v_no

Thus this is in the BCNF form because

- Every attribute is atomic hence it's in the 1NF.
- There is a composite key but no other non prime attributes. So no partial dependency exists. Hence this is in 2NF.
- There are no transitive dependencies here. Thus in the 3 NF.
- Since the LHS in the dependencies is candidate key = primary key only. Thus this is in its BCNF form.

-> Records (**c_id**, **track_id**, **loc_id**, **v_no**)

- **Primary Key:** (c_id, track_id, loc_id, v_no) → (c_id, track_id, loc_id,v_no)
- **Foreign Key:** c_id, track_id, loc_id, v_no

Thus this is in the BCNF form because

- Every attribute is atomic hence it's in the 1NF.
- There is a composite key but no other non prime attributes. So no partial dependency exists. Hence this is in 2NF.
- There are no transitive dependencies here. Thus in the 3 NF.
- Since the LHS in the dependencies is candidate key = primary key = (c_id,track_id,loc_id) only. Thus this is in its BCNF form.

2. Revised Relational Model Schema After Normalization

1. Customer (**c_id**, c_name, c_age, c_address, c_email)

2. Contacts (**c_id**, **c_contact**)
    - FK c_id references Customer

3. Vehicle_duration(**v_type**, duration)

4. Vehicle_type(**v_name**, v_type)
    - FK v_type references Vehicle_duration

5. Vehicle(**v_no**, c_id, v_name, RC_No)
    - FK c_id references Customer
    - FK v_name references Vehicle_type

6. Employee (**e_id**, e_name, e_contact, service_location, salary, designation)

7. loc_info (**loc_id**, service_location)

8. Slot (**loc_id**, **v_type**, availability)
    - FK loc_id references loc_info

9. Stock (**loc_id**, coolant, engine_oil, cleaning_water)
    - FK loc_id references loc_info

10. Ser_records (**v_no**, no_of_service)
    - FK v_no references Vehicle

11. Ser_charges(**s_type**, charges)

12. Service (**v_no**, **date**, e_id, s_type)
    - FK e_id references Employee
    - FK s_type references Ser_charges
    - FK v_no referencesVehicle

13. Track (**track_id**, **loc_id**, **v_no**, s_date, e_date, feedback)
    - FK (v_no, s_date) references service
    - FK loc_id references loc_info

14. Bill (**date, v_no,** amount)
    - FK (v_no,date) references service

15. Progress (**loc_id**, **c_year**, points)

    - FK loc_id references loc_info

16. Alert (**date**, **v_no**, pickup_type)

    - FK (date, v_no) references service

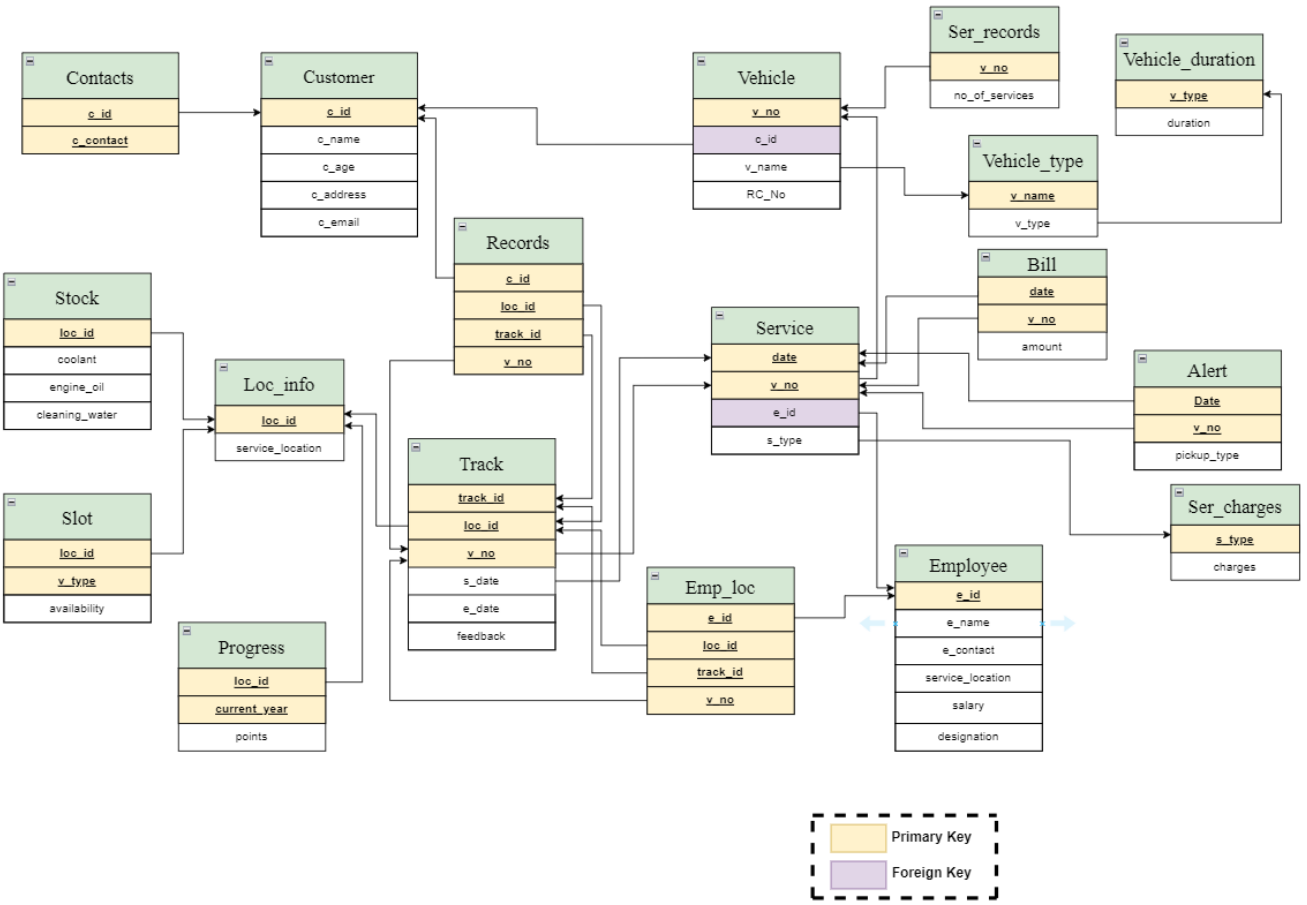
17. Emp_loc (**e_id**, **loc_id**, **track_id**, **v_no**)

    - FK e_id references employee
    - FK (loc_id, track_id, v_no) references track

18. Records (**c_id**, **track_id**, **loc_id**, **v_no**)

    - FK (c_id) references Customer
    - FK (track_id, loc_id ,v_no) references track

3. Revised Relational Model After Normalization



Section-6
Final DDL Scripts and SQL Queries

1. DDL Script (updated)

```
CREATE TABLE IF NOT EXISTS Customer
(
c_id SERIAL PRIMARY KEY,
c_name text NOT NULL,
c_age integer CHECK (c_age>=18),
c_address text,
c_email text
);
```

```
INSERT INTO Customer (c_id, c_name, c_age, c_address, c_email)
VALUES ('Madhvi','20','rajkot','madhvi111@gmail.com')
```

```
CREATE TABLE IF NOT EXISTS Contacts
(
c_id integer,
c_contact text,
PRIMARY KEY (c_id, c_contact),
FOREIGN KEY (c_id) REFERENCES customer(c_id) ON UPDATE CASCADE ON DELETE
CASCADE
);
```

```
INSERT INTO Contacts (c_id, c_contact)
VALUES ('111','9723069472')
```

```
CREATE TABLE IF NOT EXISTS Vehicle_duration
(
v_type text PRIMARY KEY,
duration integer
);
```

```
INSERT INTO Vehicle_duration(v_type, duration)
VALUES ('car','10')
```

```
CREATE TABLE IF NOT EXISTS Vehicle_type
(
v_name text PRIMARY KEY,
v_type text,
FOREIGN KEY (v_type) REFERENCES Vehicle_duration (v_type) ON
UPDATE CASCADE ON DELETE CASCADE
);
```

```
INSERT INTO Vehicle_type(v_name, v_type)
VALUES ('BMW','car')
```

```
CREATE TABLE IF NOT EXISTS Vehicle
(
v_no text PRIMARY KEY,
c_id integer,
v_name text,
RC_No text,
```

```
FOREIGN KEY (c_id) REFERENCES Customer (c_id) ON
UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY (v_name) REFERENCES Vehicle_type (v_name) ON
UPDATE CASCADE ON DELETE CASCADE
);
```

```
INSERT INTO Vehicle(v_no, c_id, v_name, RC_No)
VALUES ('GJ10RK5427','111','BMW', 'AB2106')
```

```
CREATE TABLE IF NOT EXISTS Employee
(
e_id text,
e_name text NOT NULL,
e_contact text,
service_location text,
salary integer,
designation text,
PRIMARY KEY(e_id)
);
```

```
INSERT INTO Employee (e_id, e_name, e_contact, service_location, salary,
                                                    designation)
VALUES ('101','Foram','9510533028', 'Mumbai', '100000', 'mechanic')
```

```
CREATE TABLE IF NOT EXISTS loc_info
(
loc_id text PRIMARY KEY,
service_location text
);
```

```
INSERT INTO loc_info (loc_id, service_location)
VALUES('m1', 'Mumbai')
```

```
CREATE TABLE IF NOT EXISTS Slot
(
loc_id text NOT NULL,
v_type text NOT NULL,
availability text NOT NULL,
PRIMARY KEY(loc_id,v_type),
FOREIGN KEY(loc_id) REFERENCES loc_info(loc_id) ON
UPDATE CASCADE
);
```

```
INSERT INTO Slot (loc_id,  v_type, availability)
VALUES ('m1','car','yes')
```

```
CREATE TABLE IF NOT EXISTS Stock
(
loc_id text PRIMARY KEY,
coolant integer,
engine_oil integer,
```

```
cleaning_water integer,
FOREIGN KEY(loc_id) REFERENCES loc_info(loc_id) ON
UPDATE CASCADE ON DELETE CASCADE
);
```

```
INSERT INTO Stock (loc_id, coolant, engine_oil, cleaning_water)
VALUES ('m1', '1','5','2')
```

```
CREATE TABLE IF NOT EXISTS ser_records
(
v_no text PRIMARY KEY,
no_of_services int,
FOREIGN KEY (v_no) REFERENCES vehicle(v_no) ON UPDATE CASCADE
 );
```

```
INSERT INTO Ser_records (v_no, no_of_services)
VALUES ('GJ10RK5427','6')
```

```
CREATE TABLE IF NOT EXISTS ser_charges
(
s_type text PRIMARY KEY,
charges int NOT NULL
);
```

```
INSERT INTO Ser_charges(s_type, charges)
VALUES ('coolant' , '2500')
```

```
CREATE TABLE IF NOT EXISTS service
(
v_no text,
date date,
e_id text,
s_type text NOT NULL,
PRIMARY KEY(v_no,date),
FOREIGN KEY (e_id) REFERENCES employee(e_id) ON UPDATE CASCADE,
FOREIGN KEY (s_type) REFERENCES ser_charges(s_type) ON UPDATE CASCADE,
FOREIGN KEY (v_no) REFERENCES vehicle(v_no) ON UPDATE CASCADE
);
```

```
INSERT INTO Service (v_no, date, e_id, s_type)
VALUES ('GJ10RK5427', '10-02-2021','101', 'coolant')
```

```
CREATE TABLE IF NOT EXISTS track
(
track_id integer,
loc_id text,
v_no text,
s_date DATE,
e_date DATE,
feedback integer,
PRIMARY KEY(track_id, loc_id, v_no),
FOREIGN KEY (v_no, s_date) REFERENCES service (v_no, date),
FOREIGN KEY (loc_id) REFERENCES loc_info (loc_id),
```

```
CONSTRAINT chk_feed CHECK (feedback>=0 AND feedback<=5),
CONSTRAINT chk_date CHECK ( e_date >= s_date)
);
```

```
INSERT INTO Track (track_id, loc_id, v_no, s_date, e_date, feedback)
VALUES ('1', 'm1', 'GJ10RK5427', '10-02-2021', '11-6-2021', '5')
```

```
CREATE TABLE IF NOT EXISTS Progress
(
loc_id text NOT NULL,
c_year int NOT NULL,
points integer,
PRIMARY KEY (loc_id, c_year),
FOREIGN KEY (loc_id) REFERENCES loc_info(loc_id) ON
UPDATE CASCADE ON DELETE CASCADE
);
```

```
INSERT INTO Progress (loc_id, c_year, points)
VALUES ('m1', '2021', '26')
```

```
CREATE TABLE IF NOT EXISTS emp_loc
(
e_id text,
loc_id text,
track_id integer,
v_no text,
```

```
PRIMARY KEY (e_id, loc_id,track_id, v_no),
FOREIGN KEY (e_id) REFERENCES employee(e_id) ON UPDATE
CASCADE ON DELETE CASCADE,
FOREIGN KEY (loc_id,track_id, v_no) REFERENCES track(loc_id,track_id, v_no) ON
UPDATE CASCADE ON DELETE CASCADE
);
```

```
INSERT INTO Emp_loc (e_id,  loc_id, track_id, v_no)
VALUES ('101','m1','1','GJ10RK5427')
```

```
CREATE TABLE IF NOT EXISTS Bill
(
date date,
v_no text,
amount real,
PRIMARY KEY(date, v_no),
FOREIGN KEY (v_no,date) REFERENCES service (v_no,date)
);
```

```
INSERT INTO Bill (date, v_no, amount)
VALUES ('10-02-2021','GJ10RK5427', '800')
```

```
CREATE TABLE IF NOT EXISTS Alert
(
date DATE,
v_no text,
pickup_type text,
```

```
PRIMARY KEY(date, v_no),

FOREIGN KEY (date,v_no) REFERENCES service (date,v_no)

);
```

```
INSERT INTO Alert (date, v_no, pickup_type)
VALUES ('10-02-2021', 'GJ10RK5427', 'home')
```

```
CREATE TABLE IF NOT EXISTS records

(

track_id integer,

loc_id text NOT NULL,

v_no text,

c_id integer,

PRIMARY KEY(c_id,track_id,loc_id, v_no),

FOREIGN KEY (c_id) REFERENCES Customer (c_id) ON

UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY (track_id,loc_id,v_no) REFERENCES track (track_id,loc_id,v_no) ON UPDATE
CASCADE ON DELETE CASCADE

);
```

```
INSERT INTO Records (c_id, track_id, loc_id, v_no)
VALUES ('111','1','m1','GJ10RK5427')
```

2. Data Snapshot

Customer

201901236_db/postgres@PostgreSQL 13

Query Editor | Query History

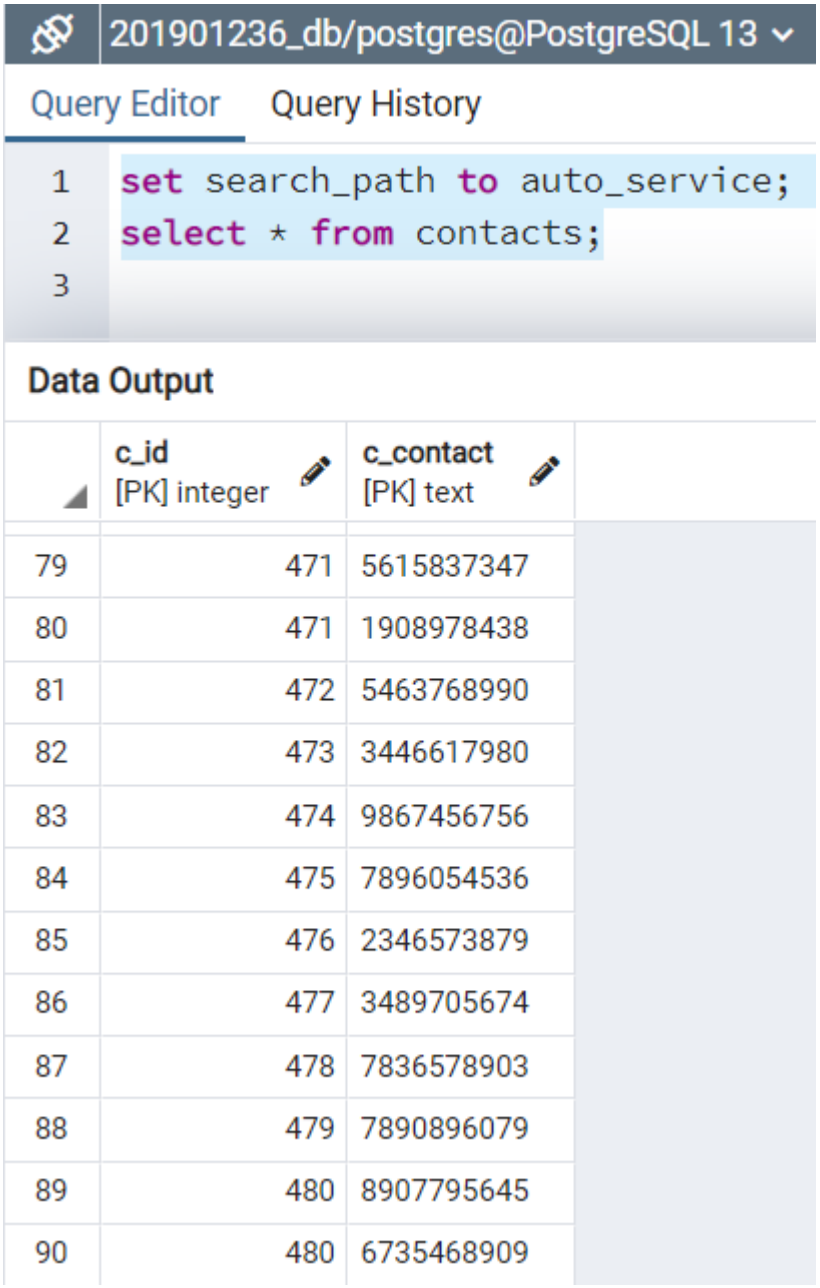
```
1  set search_path to auto_service;
2  select * from customer;
3
```

Data Output

	c_id [PK] integer	c_name text	c_age integer	c_address text	c_email text
69	469	Ferdinand	42	0 West Avenue	ftschirasche1w@java.com
70	470	Kaitlyn	47	76 American Ash Park	ksultana1x@cbsnews.com
71	471	Cordelie	47	470 Sachs Street	cvallery1y@sakura.ne.jp
72	472	Grange	49	8023 Roth Avenue	gcashell1z@alexa.com
73	473	Barbee	40	2 Texas Parkway	bcreagh20@zimbio.com
74	474	Whitney	46	1 Fulton Crossing	wpinkstone21@unblog.fr
75	475	Etti	26	7 Mallard Point	equennell22@elegantthemes.com
76	476	Ardath	49	2161 Bultman Street	awoollacott23@friendfeed.com
77	477	Truman	34	3 Dunning Hill	tsandyford24@reuters.com
78	478	Karalee	40	52233 Graceland Lane	koda25@netvibes.com
79	479	Gennie	39	1 Fairfield Terrace	gottosen26@hp.com
80	480	George	46	6 Miller Alley	gbernardo27@walmart.com

No of tuples = 80

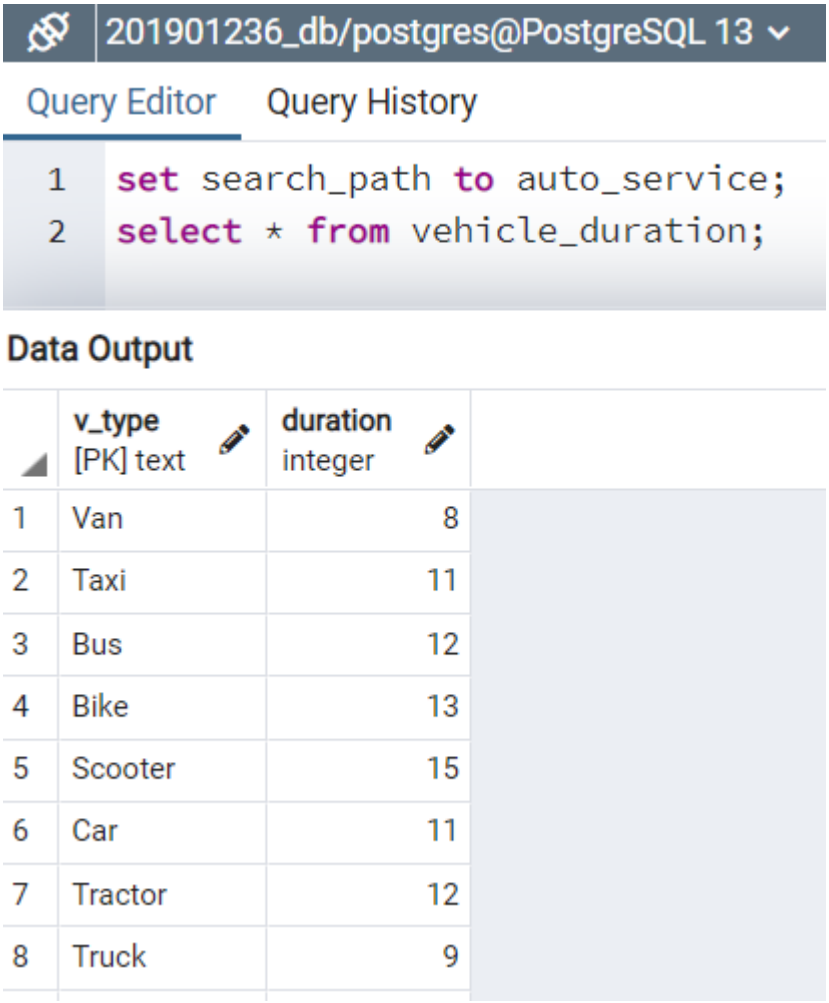
Contacts



	c_id [PK] integer	c_contact [PK] text
79	471	5615837347
80	471	1908978438
81	472	5463768990
82	473	3446617980
83	474	9867456756
84	475	7896054536
85	476	2346573879
86	477	3489705674
87	478	7836578903
88	479	7890896079
89	480	8907795645
90	480	6735468909

No of tuples = 90

Vehicle_duration



	v_type [PK] text	duration integer
1	Van	8
2	Taxi	11
3	Bus	12
4	Bike	13
5	Scooter	15
6	Car	11
7	Tractor	12
8	Truck	9

No of tuples = 8

Vehicle_type

201901171_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2  select * from Vehicle_type;
```

Data Output | Explain | Messages | Notifications

	v_name [PK] text	v_type text
23	Pep plus	Scooter
24	Yamaha YZF	Bike
25	Double decker bus	Bus
26	Activa	Scooter
27	Hero Splendor	Bike
28	Ford Fiesta	Van
29	VW caddy van	Van
30	Honda Shine	Bike
31	Kawasaki	Bike
32	Peugeot Partner	Van
33	Trakstar	Tractor
34	KTM RC	Bike
35	Ford model 8N	Tractor
36	Farmall model M	Tractor
37	Citroen Berlingo	Van
38	Honda Livo	Bike
39	Hyundai i20	Car
40	TVS XL 100	Scooter

No of tuples: 40

Vehicle

201901171_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2  select * from vehicle;
```

Data Output | Explain | Messages | Notifications

	v_no [PK] text	c_id integer	v_name text	rc_no text
87	UP 16HJ5638	444	John Deere	AB01CD0087
88	UP 16HJ5639	405	TATA Punch	AB01CD0088
89	UP 16HJ5640	470	Renault Kwid	AB01CD0089
90	UP 16HJ5641	468	Access	AB01CD0090
91	UP 16HJ5642	455	Hero Pleasure	AB01CD0091
92	UP 16HJ5643	471	Kia	AB01CD0092
93	UP 16HJ5644	427	TATA Tiago	AB01CD0093
94	UP 16HJ5645	418	Fiat Ducato	AB01CD0094
95	UP 16HJ5646	436	StarBus 24	AB01CD0095
96	JK 01HN8678	465	Tata Magna	AB01CD0096
97	JK 01HN8679	420	Maruti Baleno	AB01CD0097
98	JK 01HN8680	408	Jaguar	AB01CD0098
99	JK 01HN8681	423	Ford Transit	AB01CD0099
100	JK 01HN8682	458	Accent	AB01CD0100
101	JK 01HN8683	419	Toyota Tacoma	AB01CD0101
102	JK 01HN8684	429	Bajaj Pulsar	AB01CD0102
103	JK 01HN8685	478	Pep plus	AB01CD0103
104	JK 01HN8686	404	Yamaha YZF	AB01CD0104

No of tuples: 104

Employee

201901236_db/postgres@PostgreSQL 13

Query Editor | Query History

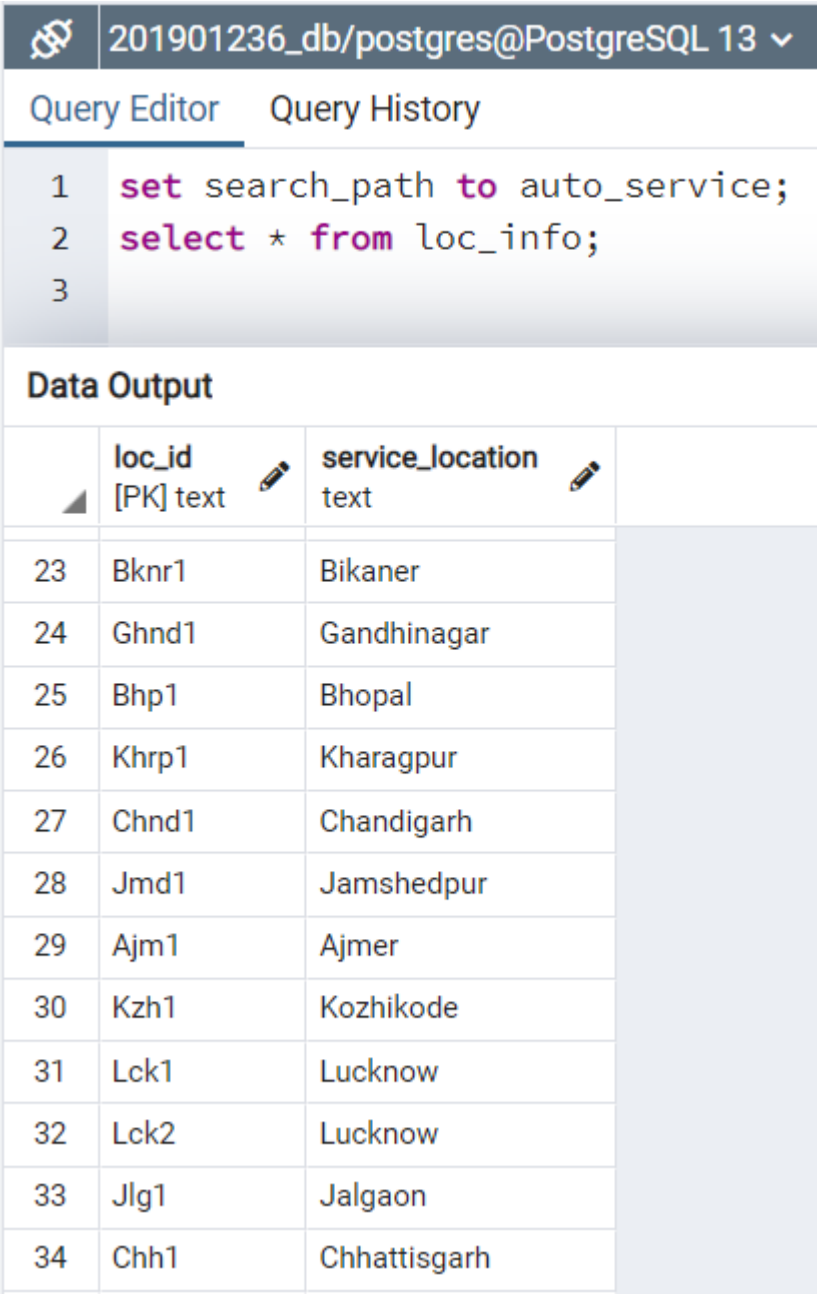
```
1  set search_path to auto_service;
2  select * from employee;
3
```

Data Output

	e_id [PK] text	e_name text	e_contact text	service_location text	salary integer	designation text
149	MA24	Irvine	7648432374	Bikaner	183091	Manager
150	MA25	Bettye	8759261549	Gandhinagar	187108	Manager
151	MA26	Caldwell	9414382368	Bhopal	187397	Manager
152	MA27	Shanon	6728366966	Kharagpur	192144	Manager
153	MA28	Marget	6243967982	Chandigarh	192459	Manager
154	MA29	Elton	7220678819	Jamshedpur	194533	Manager
155	MA30	Margalit	7928213805	Ajmer	194994	Manager
156	MA31	Gerrie	7315183042	Kozhikode	196707	Manager
157	MA32	Cathie	9650417589	Lucknow	197732	Manager
158	MA33	Tommy	8621311702	Lucknow	197773	Manager
159	MA34	Charmine	9015914002	Jalgaon	198752	Manager
160	MA35	Camile	9624321843	Chhattisgarh	199475	Manager

No of tuples = 160

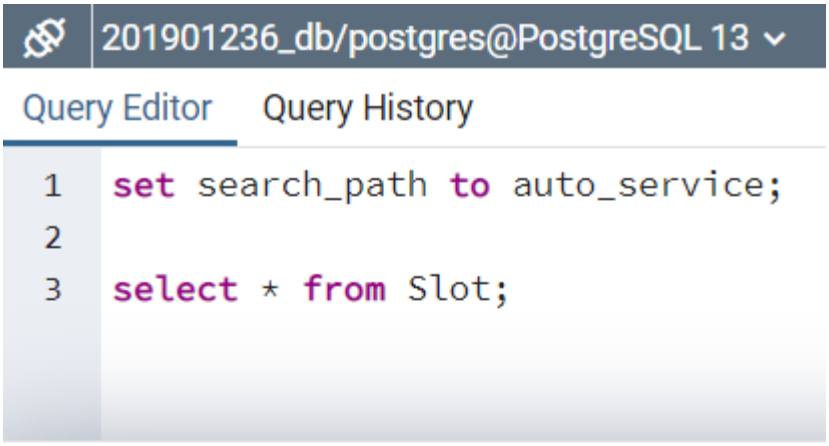
loc_info



	loc_id [PK] text	service_location text
23	Bknr1	Bikaner
24	Ghnd1	Gandhinagar
25	Bhp1	Bhopal
26	Khrp1	Kharagpur
27	Chnd1	Chandigarh
28	Jmd1	Jamshedpur
29	Ajm1	Ajmer
30	Kzh1	Kozhikode
31	Lck1	Lucknow
32	Lck2	Lucknow
33	Jlg1	Jalgaon
34	Chh1	Chhattisgarh

No of tuples = 34

Slot



Data Output

	loc_id [PK] text	v_type [PK] text	availability text
257	Jlg1	Van	NO
258	Jlg1	Taxi	YES
259	Jlg1	Bus	YES
260	Jlg1	Bike	YES
261	Jlg1	Scooter	YES
262	Jlg1	Car	YES
263	Jlg1	Tractor	YES
264	Jlg1	Truck	YES
265	Chh1	Van	YES
266	Chh1	Taxi	YES
267	Chh1	Bus	YES
268	Chh1	Bike	YES
269	Chh1	Scooter	YES
270	Chh1	Car	YES
271	Chh1	Tractor	YES
272	Chh1	Truck	YES

No of tuples: 272

Stock

201901236_db/postgres@PostgreSQL 13

Query Editor | Query History

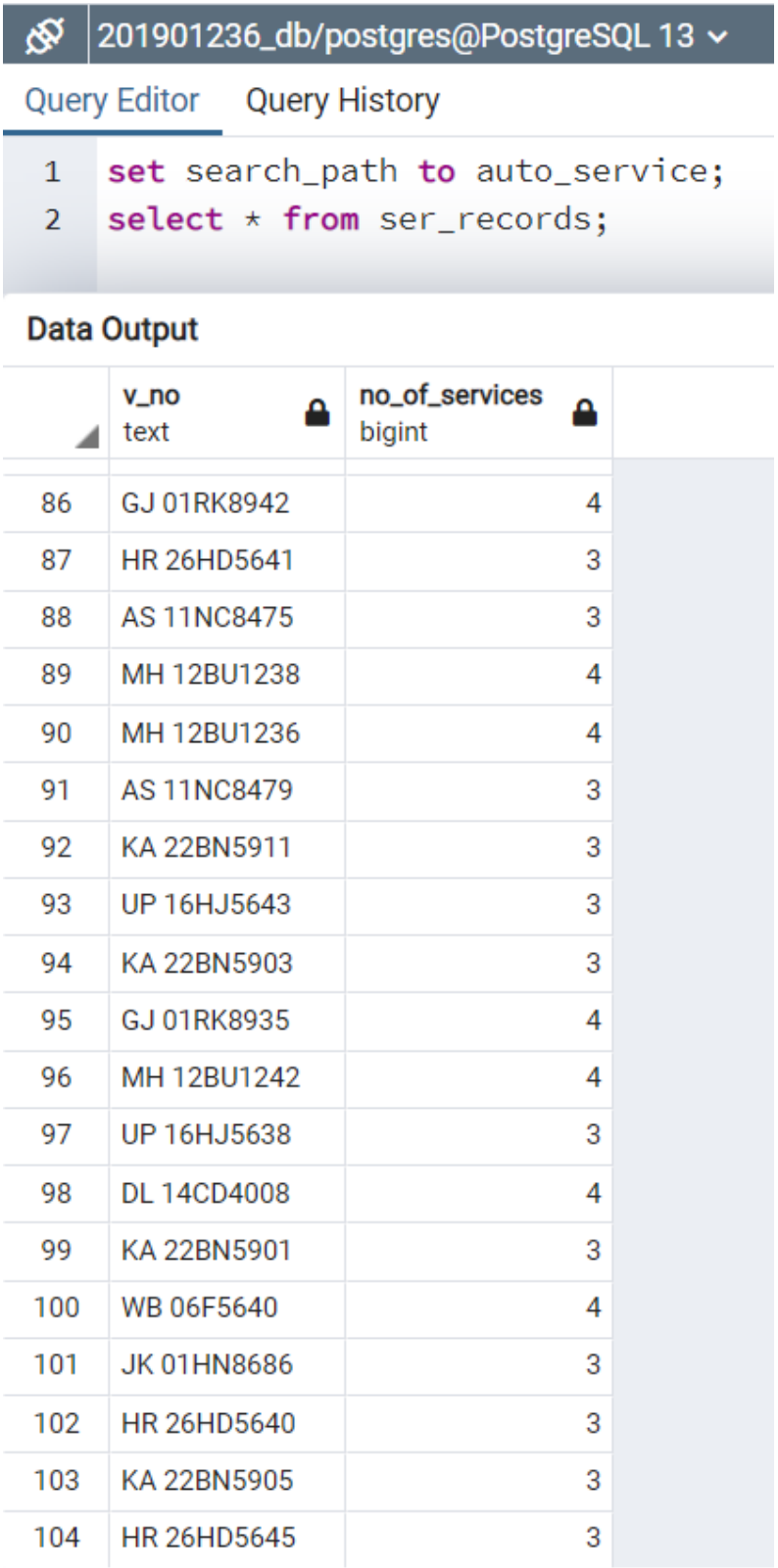
```
1  set search_path to auto_service;
2  select * from stock;
3
```

Data Output

	loc_id [PK] text	coolant integer	engine_oil integer	cleaning_water integer
23	Bknr1	183	296	149
24	Ghnd1	150	48	165
25	Bhp1	176	95	88
26	Khrp1	186	166	317
27	Chnd1	50	192	68
28	Jmd1	32	49	159
29	Ajm1	156	301	129
30	Kzh1	38	188	36
31	Lck1	163	350	153
32	Lck2	112	75	208
33	Jlg1	192	295	196
34	Chh1	141	109	218

No of tuples = 34

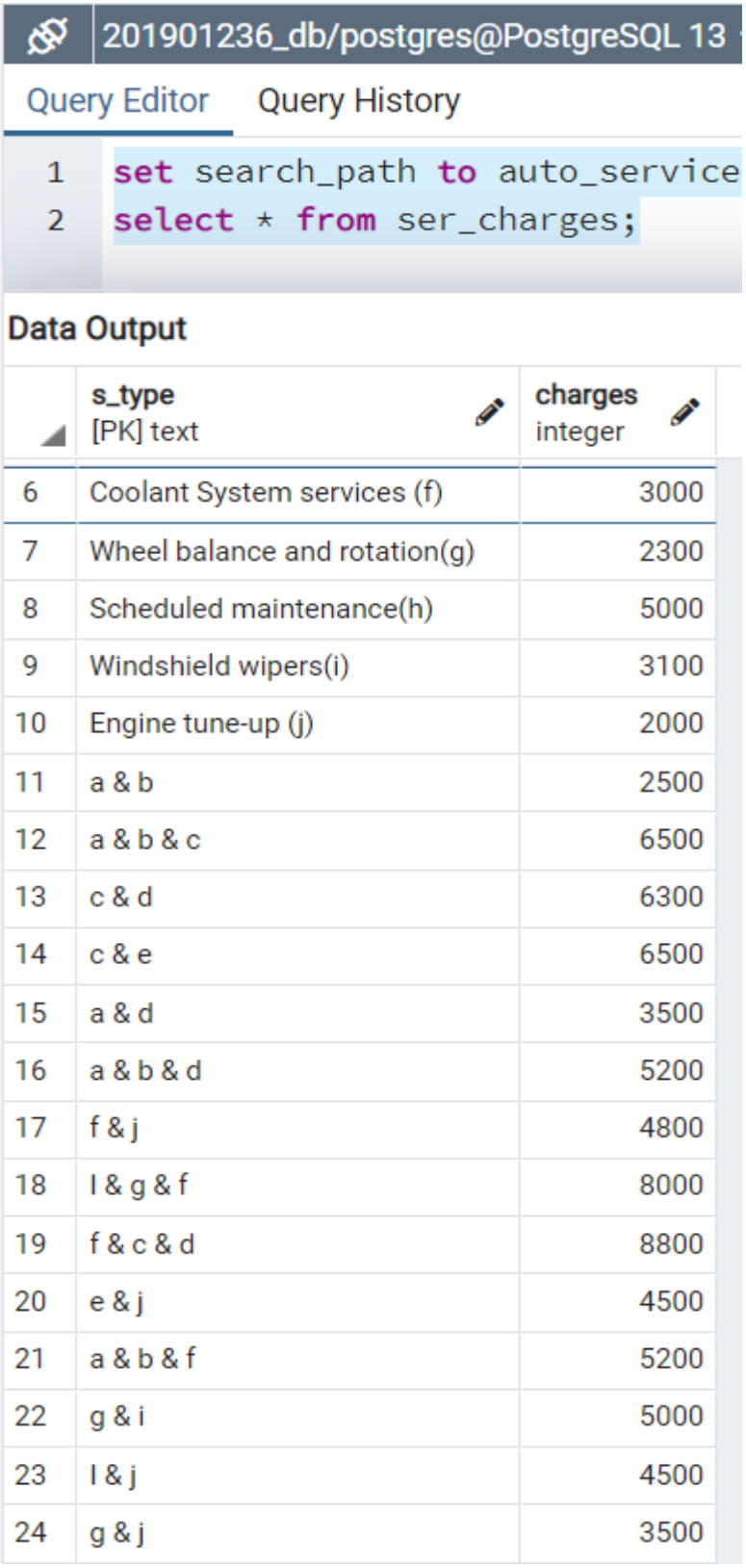
Ser_records



	v_no text	no_of_services bigint
86	GJ 01RK8942	4
87	HR 26HD5641	3
88	AS 11NC8475	3
89	MH 12BU1238	4
90	MH 12BU1236	4
91	AS 11NC8479	3
92	KA 22BN5911	3
93	UP 16HJ5643	3
94	KA 22BN5903	3
95	GJ 01RK8935	4
96	MH 12BU1242	4
97	UP 16HJ5638	3
98	DL 14CD4008	4
99	KA 22BN5901	3
100	WB 06F5640	4
101	JK 01HN8686	3
102	HR 26HD5640	3
103	KA 22BN5905	3
104	HR 26HD5645	3

No of tuples: 104

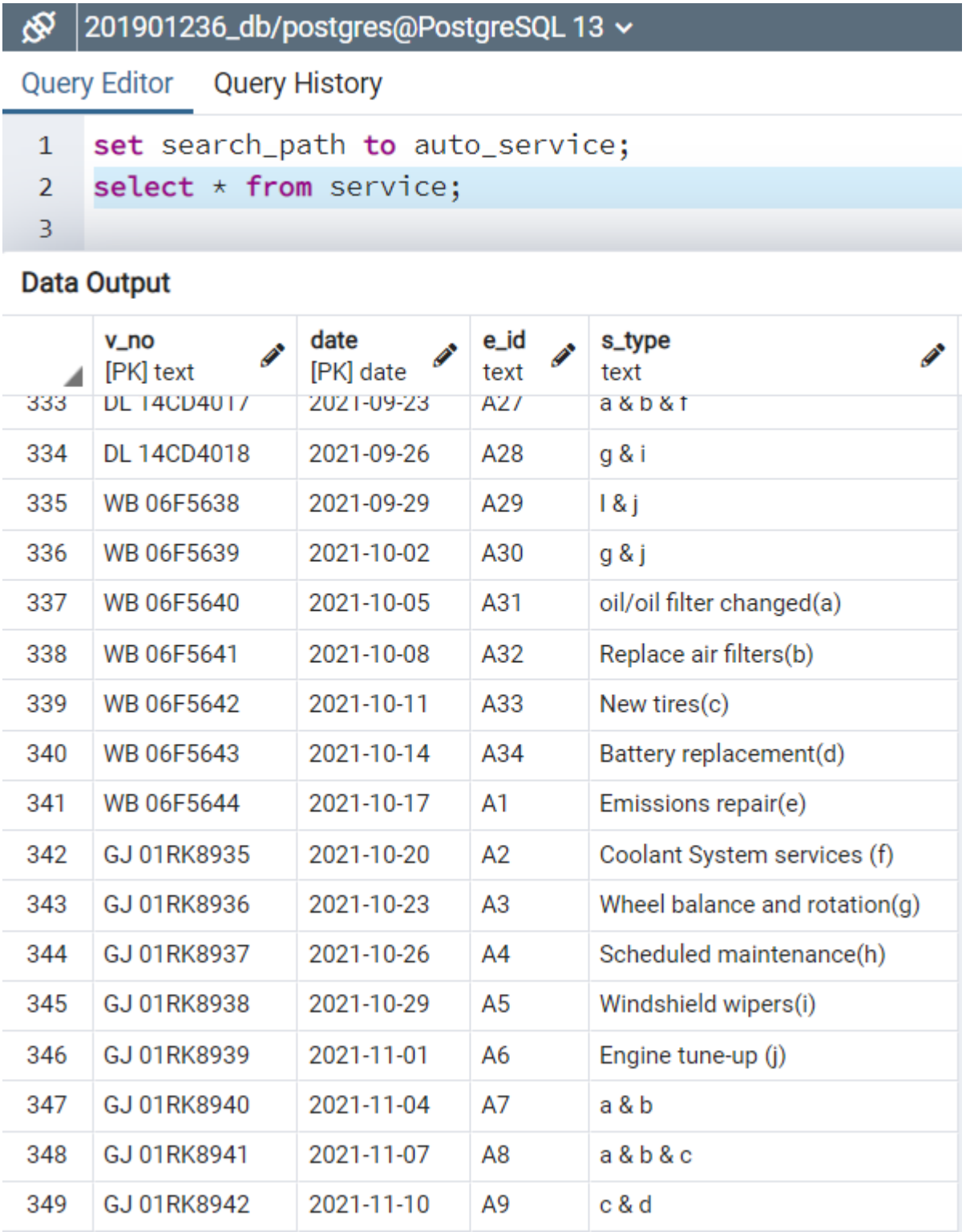
Ser_charges



	s_type [PK] text	charges integer
6	Coolant System services (f)	3000
7	Wheel balance and rotation(g)	2300
8	Scheduled maintenance(h)	5000
9	Windshield wipers(i)	3100
10	Engine tune-up (j)	2000
11	a & b	2500
12	a & b & c	6500
13	c & d	6300
14	c & e	6500
15	a & d	3500
16	a & b & d	5200
17	f & j	4800
18	l & g & f	8000
19	f & c & d	8800
20	e & j	4500
21	a & b & f	5200
22	g & i	5000
23	l & j	4500
24	g & j	3500

No of tuples: 24

Service



	v_no [PK] text	date [PK] date	e_id text	s_type text
333	DL 14CD4017	2021-09-23	A27	a & b & f
334	DL 14CD4018	2021-09-26	A28	g & i
335	WB 06F5638	2021-09-29	A29	l & j
336	WB 06F5639	2021-10-02	A30	g & j
337	WB 06F5640	2021-10-05	A31	oil/oil filter changed(a)
338	WB 06F5641	2021-10-08	A32	Replace air filters(b)
339	WB 06F5642	2021-10-11	A33	New tires(c)
340	WB 06F5643	2021-10-14	A34	Battery replacement(d)
341	WB 06F5644	2021-10-17	A1	Emissions repair(e)
342	GJ 01RK8935	2021-10-20	A2	Coolant System services (f)
343	GJ 01RK8936	2021-10-23	A3	Wheel balance and rotation(g)
344	GJ 01RK8937	2021-10-26	A4	Scheduled maintenance(h)
345	GJ 01RK8938	2021-10-29	A5	Windshield wipers(i)
346	GJ 01RK8939	2021-11-01	A6	Engine tune-up (j)
347	GJ 01RK8940	2021-11-04	A7	a & b
348	GJ 01RK8941	2021-11-07	A8	a & b & c
349	GJ 01RK8942	2021-11-10	A9	c & d

No of tuples: 349

Track

201901236_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2
3  select * from track;
4
```

Data Output

	track_id [PK] integer	loc_id [PK] text	v_no [PK] text	s_date date	e_date date	feedback integer
333	1332	Chnd1	DL 14CD4017	2021-09-23	2021-09-26	4
334	1333	Jmd1	DL 14CD4018	2021-09-26	2021-09-29	3
335	1334	Ajm1	WB 06F5638	2021-09-29	2021-10-02	4
336	1335	Kzh1	WB 06F5639	2021-10-02	2021-10-05	1
337	1336	Lck1	WB 06F5640	2021-10-05	2021-10-08	5
338	1337	Lck2	WB 06F5641	2021-10-08	2021-10-11	2
339	1338	Jlg1	WB 06F5642	2021-10-11	2021-10-14	2
340	1339	Chh1	WB 06F5643	2021-10-14	2021-10-17	0
341	1340	Mum1	WB 06F5644	2021-10-17	2021-10-20	0
342	1341	Mum2	GJ 01RK8935	2021-10-20	2021-10-23	4
343	1342	Del1	GJ 01RK8936	2021-10-23	2021-10-26	2
344	1343	Del2	GJ 01RK8937	2021-10-26	2021-10-29	4
345	1344	Bng1	GJ 01RK8938	2021-10-29	2021-11-01	2
346	1345	Bng2	GJ 01RK8939	2021-11-01	2021-11-04	2
347	1346	Hyd1	GJ 01RK8940	2021-11-04	2021-11-07	5
348	1347	Hyd2	GJ 01RK8941	2021-11-07	2021-11-10	5
349	1348	Ahbd1	GJ 01RK8942	2021-11-10	2021-11-13	1

No of tuples: 349

Bill

201901236_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2  select * from bill;
3
```

Data Output

	date [PK] date	v_no [PK] text	amount real
332	2021-09-20	DL 14CD4016	4500
333	2021-09-23	DL 14CD4017	5200
334	2021-09-26	DL 14CD4018	5000
335	2021-09-29	WB 06F5638	4500
336	2021-10-02	WB 06F5639	3500
337	2021-10-05	WB 06F5640	1300
338	2021-10-08	WB 06F5641	1500
339	2021-10-11	WB 06F5642	4000
340	2021-10-14	WB 06F5643	2500
341	2021-10-17	WB 06F5644	3000
342	2021-10-20	GJ 01RK8935	3000
343	2021-10-23	GJ 01RK8936	2300
344	2021-10-26	GJ 01RK8937	5000
345	2021-10-29	GJ 01RK8938	3100
346	2021-11-01	GJ 01RK8939	2000
347	2021-11-04	GJ 01RK8940	2500
348	2021-11-07	GJ 01RK8941	6500
349	2021-11-10	GJ 01RK8942	6300

No of tuples: 349

Progress

201901236_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2  select * from progress;
```

Data Output

	loc_id [PK] text	c_year [PK] integer	points integer
85	Ajm1	2019	15
86	Ajm1	2020	20
87	Ajm1	2021	25
88	Kzh1	2019	30
89	Kzh1	2020	5
90	Kzh1	2021	10
91	Lck1	2019	15
92	Lck1	2020	20
93	Lck1	2021	25
94	Lck2	2019	30
95	Lck2	2020	5
96	Lck2	2021	10
97	Jlg1	2019	15
98	Jlg1	2020	20
99	Jlg1	2021	25
100	Chh1	2019	30
101	Chh1	2020	35
102	Chh1	2021	10

No of tuples = 102

Alert

201901171_db/postgres@PostgreSQL 13

Query Editor | Query History

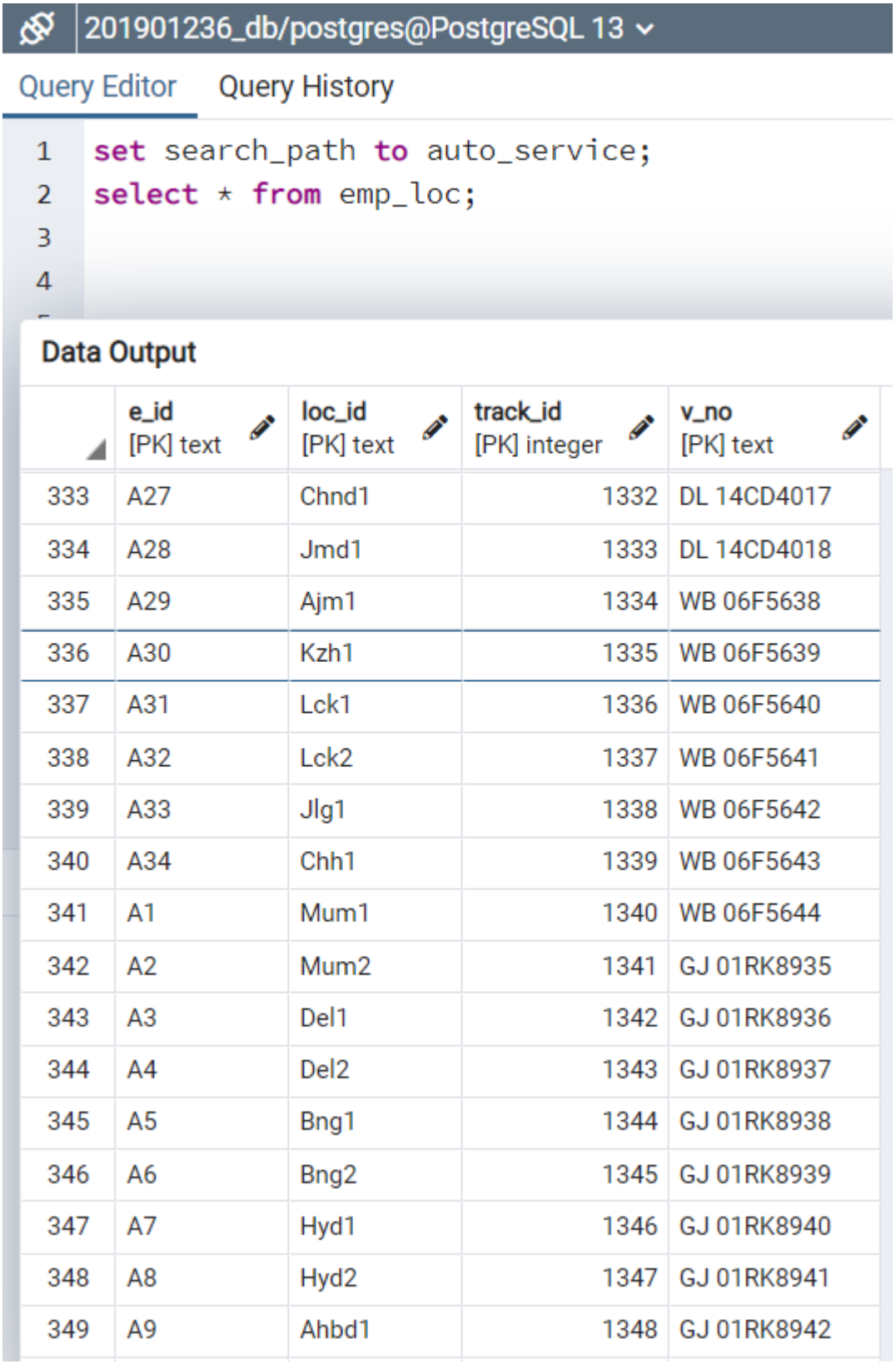
```
1  set search_path to auto_service;
2  select * from alert;
3
```

Data Output | Explain | Messages | Notifications

	date [PK] date	v_no [PK] text	pickup_type text
33	2021-09-20	DL 14CD4016	Normal
34	2021-09-23	DL 14CD4017	Normal
35	2021-09-26	DL 14CD4018	Emergency
36	2021-09-29	WB 06F5638	Emergency
37	2021-10-02	WB 06F5639	Emergency
38	2021-10-05	WB 06F5640	Emergency
39	2021-10-08	WB 06F5641	Emergency
40	2021-10-11	WB 06F5642	Emergency
41	2021-10-14	WB 06F5643	Emergency
42	2021-10-17	WB 06F5644	Emergency
43	2021-10-20	GJ 01RK8935	Normal
44	2021-10-23	GJ 01RK8936	Normal
45	2021-10-26	GJ 01RK8937	Normal
46	2021-10-29	GJ 01RK8938	Normal
47	2021-11-01	GJ 01RK8939	Normal
48	2021-11-04	GJ 01RK8940	Normal
49	2021-11-07	GJ 01RK8941	Emergency
50	2021-11-10	GJ 01RK8942	Emergency

No of tuples: 50

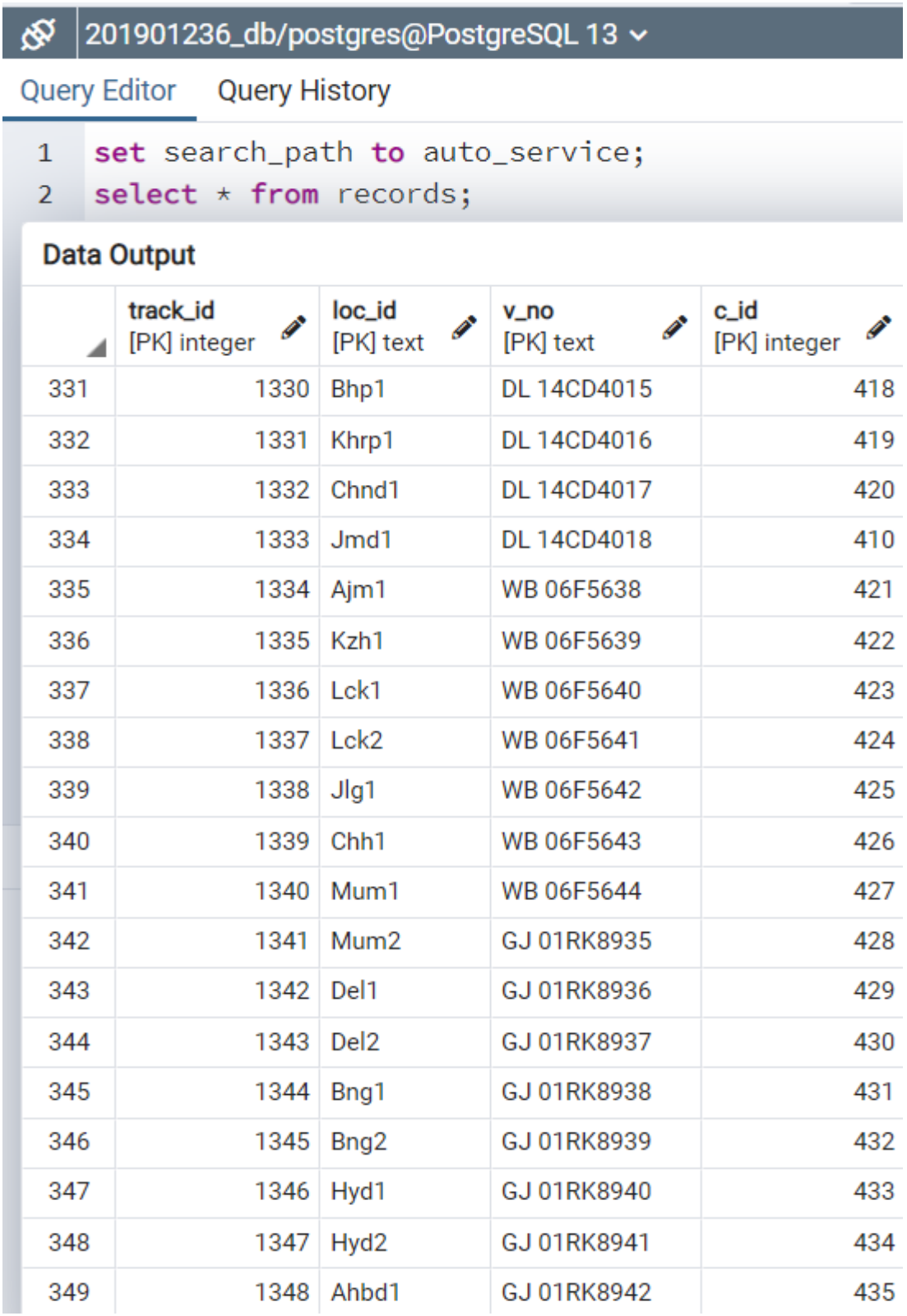
Emp_loc



	e_id [PK] text	loc_id [PK] text	track_id [PK] integer	v_no [PK] text
333	A27	Chnd1	1332	DL 14CD4017
334	A28	Jmd1	1333	DL 14CD4018
335	A29	Ajm1	1334	WB 06F5638
336	A30	Kzh1	1335	WB 06F5639
337	A31	Lck1	1336	WB 06F5640
338	A32	Lck2	1337	WB 06F5641
339	A33	Jlg1	1338	WB 06F5642
340	A34	Chh1	1339	WB 06F5643
341	A1	Mum1	1340	WB 06F5644
342	A2	Mum2	1341	GJ 01RK8935
343	A3	Del1	1342	GJ 01RK8936
344	A4	Del2	1343	GJ 01RK8937
345	A5	Bng1	1344	GJ 01RK8938
346	A6	Bng2	1345	GJ 01RK8939
347	A7	Hyd1	1346	GJ 01RK8940
348	A8	Hyd2	1347	GJ 01RK8941
349	A9	Ahbd1	1348	GJ 01RK8942

No of tuples: 349

Records



	track_id [PK] integer	loc_id [PK] text	v_no [PK] text	c_id [PK] integer
331	1330	Bhp1	DL 14CD4015	418
332	1331	Khrp1	DL 14CD4016	419
333	1332	Chnd1	DL 14CD4017	420
334	1333	Jmd1	DL 14CD4018	410
335	1334	Ajm1	WB 06F5638	421
336	1335	Kzh1	WB 06F5639	422
337	1336	Lck1	WB 06F5640	423
338	1337	Lck2	WB 06F5641	424
339	1338	Jlg1	WB 06F5642	425
340	1339	Chh1	WB 06F5643	426
341	1340	Mum1	WB 06F5644	427
342	1341	Mum2	GJ 01RK8935	428
343	1342	Del1	GJ 01RK8936	429
344	1343	Del2	GJ 01RK8937	430
345	1344	Bng1	GJ 01RK8938	431
346	1345	Bng2	GJ 01RK8939	432
347	1346	Hyd1	GJ 01RK8940	433
348	1347	Hyd2	GJ 01RK8941	434
349	1348	Ahbd1	GJ 01RK8942	435

No of tuples: 349

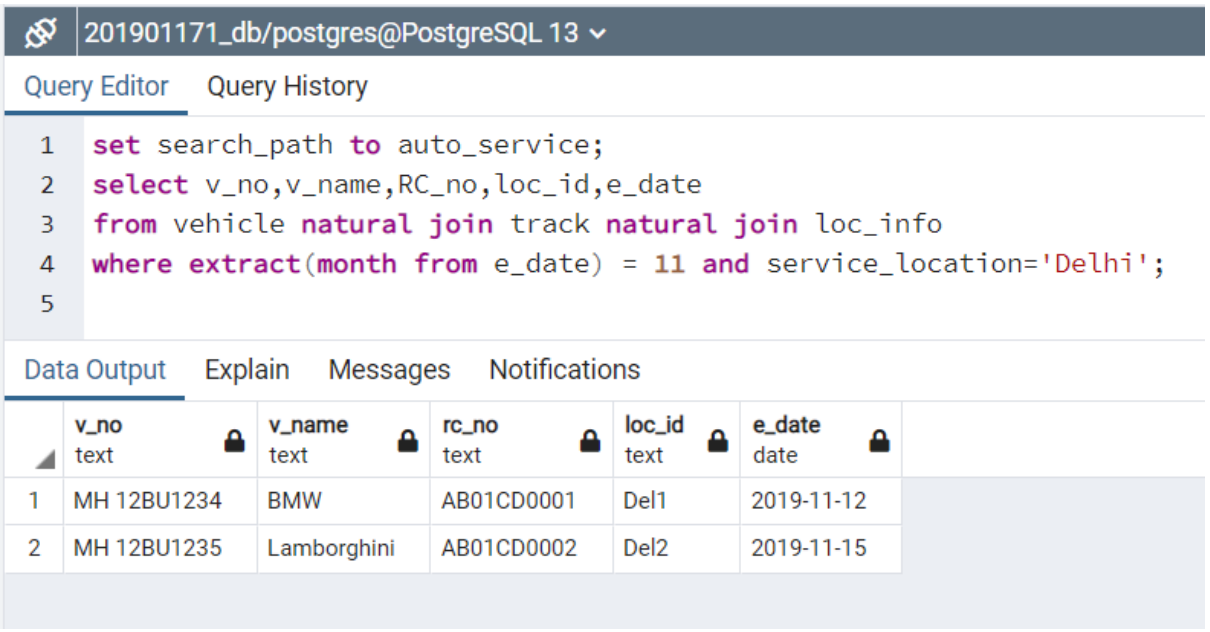
3. SQL Queries

1. **Find the vehicle details, service date and centre where vehicle servicing was done in November , and the location is Delhi.**

- **select v_no,v_name,RC_no,loc_id,e_date from**

  vehicle NATURAL JOIN track NATURAL JOIN loc_info

  where extract(month from e_date) = 11 and service_location='Delhi';



	v_no text	v_name text	rc_no text	loc_id text	e_date date
1	MH 12BU1234	BMW	AB01CD0001	Del1	2019-11-12
2	MH 12BU1235	Lamborghini	AB01CD0002	Del2	2019-11-15

Number of tuples = 2

2. **List the employees who have serviced the maximum number of vehicles till date along with the vehicle details they serviced.**

- ```
  create view maxE as SELECT e_id, count(*)
  FROM service
  GROUP BY e_id
  HAVING COUNT(*) = (SELECT COUNT(*) AS c FROM service GROUP BY e_id  ORDER BY c DESC LIMIT 1)
  ```

```
16  select * from maxE;
```

Explain | Messages | Notifications

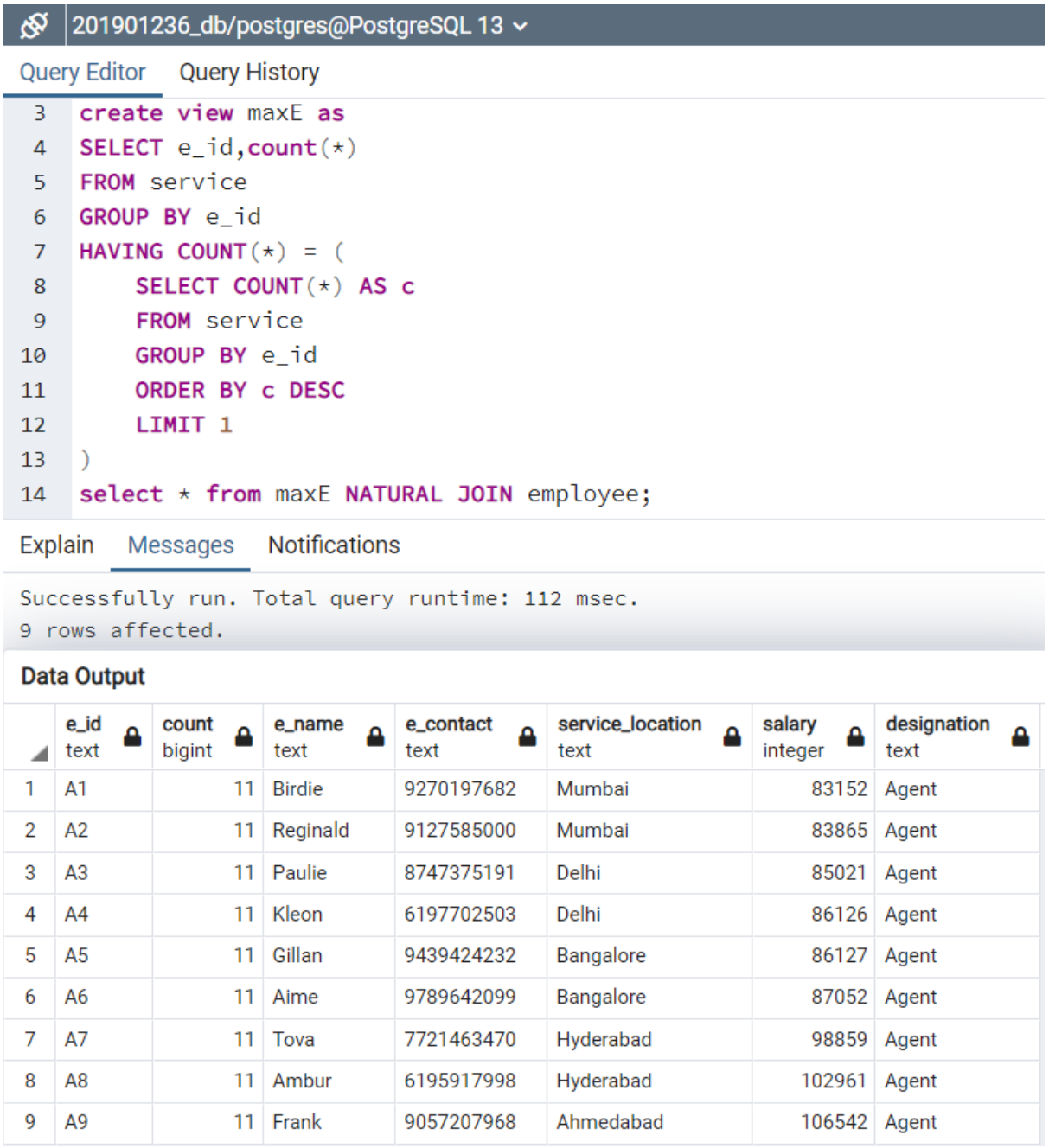
```
Successfully run. Total query runtime: 118 msec.
9 rows affected.
```

Data Output

	e_id text	count bigint
1	A1	11
2	A7	11
3	A3	11
4	A9	11
5	A4	11
6	A6	11
7	A5	11
8	A2	11
9	A8	11

View maxE
```

select * from maxE NATURAL JOIN employee;



	e_id text	count bigint	e_name text	e_contact text	service_location text	salary integer	designation text
1	A1	11	Birdie	9270197682	Mumbai	83152	Agent
2	A2	11	Reginald	9127585000	Mumbai	83865	Agent
3	A3	11	Paulie	8747375191	Delhi	85021	Agent
4	A4	11	Kleon	6197702503	Delhi	86126	Agent
5	A5	11	Gillan	9439424232	Bangalore	86127	Agent
6	A6	11	Aime	9789642099	Bangalore	87052	Agent
7	A7	11	Tova	7721463470	Hyderabad	98859	Agent
8	A8	11	Ambur	6195917998	Hyderabad	102961	Agent
9	A9	11	Frank	9057207968	Ahmedabad	106542	Agent

No Of Tuples: 9

- We can see that out of 35 agents, 9 agents have serviced the maximum number of vehicles (11) till now.
- The above table shows the details of those agents.

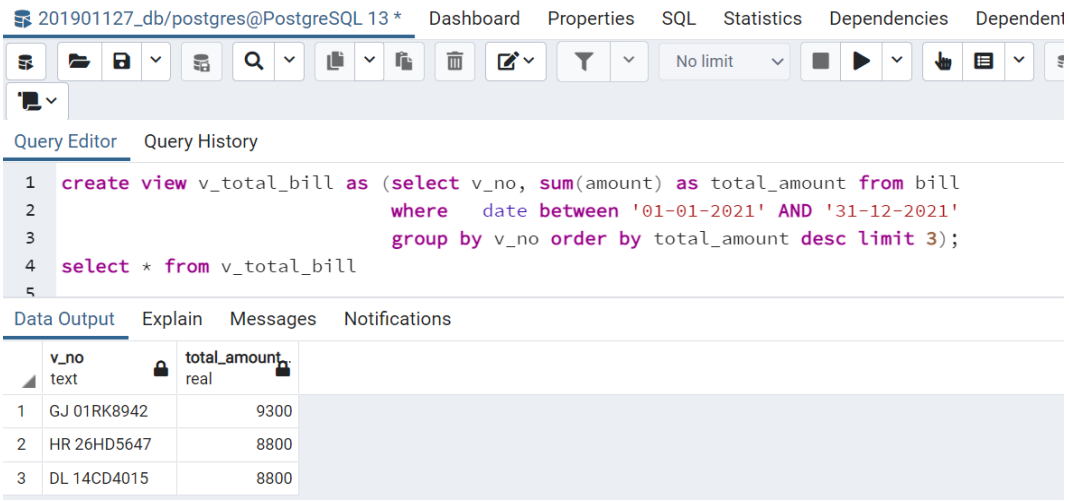
3. **List 3 customers who have paid the highest in the year 2021.**

- **create view v_total_bill as (select v_no, sum(amount) as total_amount from bill**

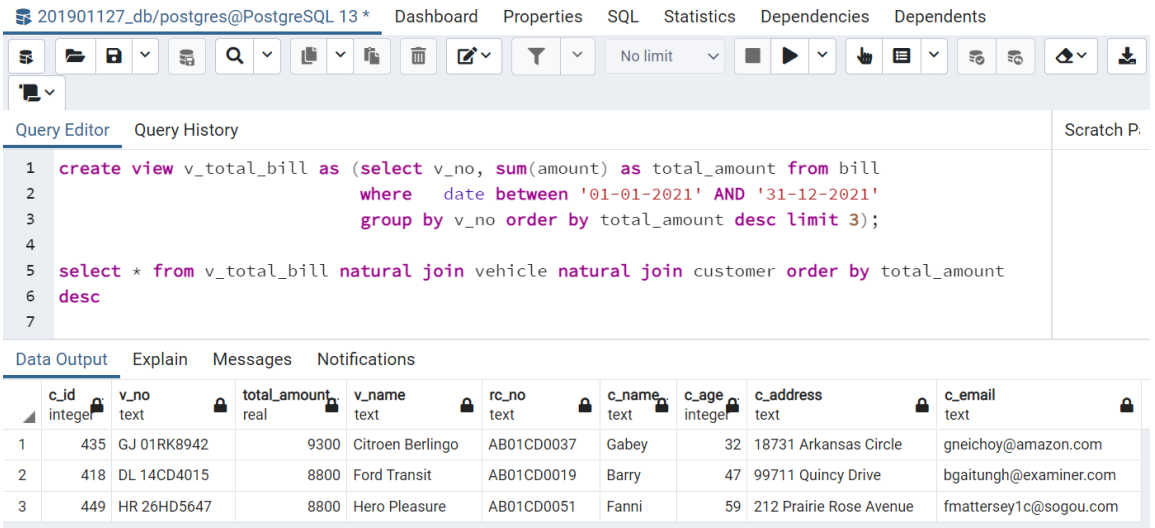
  where date between '01-01-2021' AND '31-12-2021'

  group by v_no order by total_amount desc limit 3);

select * from v_total_bill



select * from v_total_bill natural join vehicle natural join customer order by total_amount desc



Number of tuples = 3

4. **list all vehicle details which have passed their service duration time from the date '30-01-2021'.**

- ```
  create view veh_month_duration as
  (select v_no,v_type, (extract(month from current_date) - extract(month from e_date))
  as month_ from vehicle NATURAL JOIN  track NATURAL JOIN vehicle_type
  where e_date > '30-01-2021')

  select * from veh_month_duration
  ```

201901171_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service
2
3  create view veh_month_duration as
4   (select v_no,v_type, (extract(month from current_date) - extract(month from e_date))
5   as month_ from vehicle NATURAL JOIN  track NATURAL JOIN vehicle_type
6   where e_date > '30-01-2021')
7  select * from veh_month_duration
```

Data Output | Explain | Messages | Notifications

	v_no text	v_type text	month_ double precision
85	AS 11NC8475	Bike	7
86	GJ 01RK8937	Van	1
87	AS 11NC8476	Van	7
88	GJ 01RK8938	Tractor	0
89	AS 11NC8477	Tractor	7
90	GJ 01RK8939	Bike	0
91	AS 11NC8478	Bike	7
92	GJ 01RK8940	Tractor	0
93	AS 11NC8479	Tractor	7
94	GJ 01RK8941	Tractor	0
95	AS 11NC8480	Tractor	6
96	GJ 01RK8942	Van	0
97	AS 11NC8481	Van	6
98	MN 01AG2222	Bike	6
99	MN 01AG2223	Car	6
100	MN 01AG2224	Scooter	6

view veh_month_duration
```

select v_no , v_type

from veh_month_duration NATURAL JOIN vehicle_duration where month_ >= duration;

```
8  select v_no , v_type from veh_month_duration NATURAL JOIN vehicle_duration
9  where month_ >= duration;
```

Data Output | Explain | Messages | Notifications

	v_no text	v_type text
1	KA 22BN5900	Truck

Number of tuples = 1

5. **Find all the bike details that came for service in 2019 in descending order of the amount paid.**

- **select * from bill**

  NATURAL JOIN service

  NATURAL JOIN vehicle

  NATURAL JOIN vehicle_type

  where vehicle_type.v_type='Bike' and service.date between '2019-01-01' and '2019-12-31'

  ORDER BY bill.amount desc;

```
201901236_db/postgres@PostgreSQL 13
Query Editor  Query History
1  set search_path to auto_service;
2  select * from bill
3  NATURAL JOIN service
4  NATURAL JOIN vehicle
5  NATURAL JOIN vehicle_type
6  where vehicle_type.v_type='Bike' and service.date between '2019-01-01' and '2019-12-31'
7  ORDER BY bill.amount desc;
8
Explain  Messages  Notifications
Successfully run. Total query runtime: 156 msec.
20 rows affected.
```

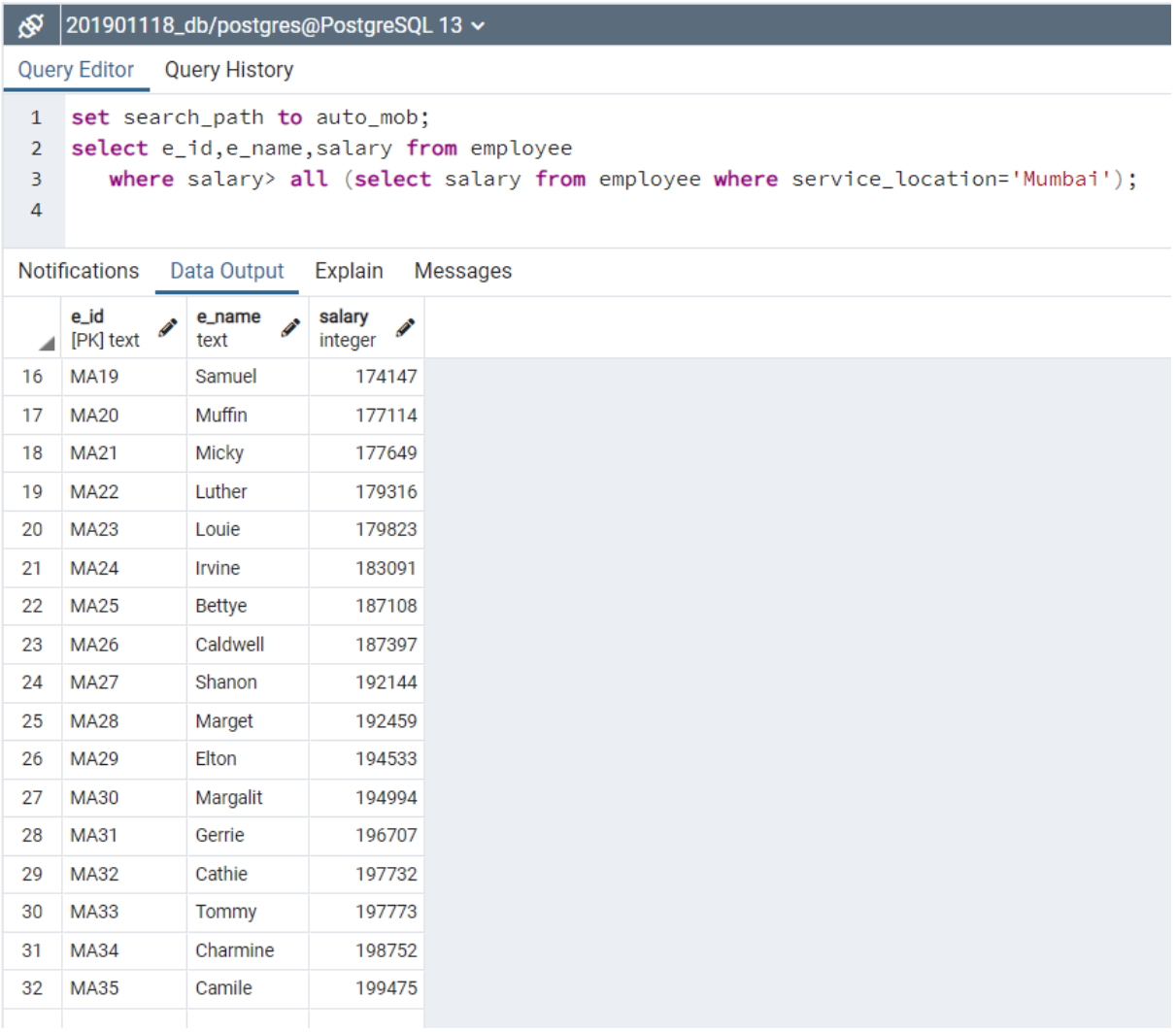
Data Output

	v_name text	v_no text	date date	amount real	e_id text	s_type text	c_id integer	rc_no text	v_type text
10	Royal Enfield	MH 12BU1236	2019-01-07	4000	A3	New tires(c)	403	AB01CD0003	Bike
11	Hero Splendor	WB 06F5642	2019-03-20	4000	A27	New tires(c)	425	AB01CD0027	Bike
12	Yamaha YZF	WB 06F5639	2019-03-11	3500	A24	g & j	422	AB01CD0024	Bike
13	Honda Shine	GJ 01RK8935	2019-03-29	3000	A30	Coolant System services (f)	428	AB01CD0030	Bike
14	Bajaj Pulsar	JK 01HN8684	2019-10-31	3000	A34	Coolant System services (f)	429	AB01CD0102	Bike
15	Honda Livo	MN 01AG2222	2019-08-20	3000	A10	Coolant System services (f)	474	AB01CD0078	Bike
16	Royal Enfield	MH 12BU1236	2019-11-15	2500	A5	a & b	403	AB01CD0003	Bike
17	Royal Enfield	MN 01AG2227	2019-09-04	2500	A15	a & b	479	AB01CD0083	Bike
18	Kawasaki	GJ 01RK8936	2019-04-01	2300	A31	Wheel balance and rotation(g)	429	AB01CD0031	Bike
19	KTM RC	GJ 01RK8939	2019-04-10	2000	A34	Engine tune-up (j)	432	AB01CD0034	Bike
20	KTM RC	AS 11NC8478	2019-08-08	1500	A6	Replace air filters(b)	471	AB01CD0074	Bike

Number of tuples = 20

6. **Find the names of all employees whose salary is greater than the salary of all employees at the Mumbai service location.**

- **select e_id,e_name,salary from employee**
  where salary> all (select salary from employee where service_location='Mumbai');



	e_id [PK] text	e_name text	salary integer
16	MA19	Samuel	174147
17	MA20	Muffin	177114
18	MA21	Micky	177649
19	MA22	Luther	179316
20	MA23	Louie	179823
21	MA24	Irvine	183091
22	MA25	Bettye	187108
23	MA26	Caldwell	187397
24	MA27	Shanon	192144
25	MA28	Marget	192459
26	MA29	Elton	194533
27	MA30	Margalit	194994
28	MA31	Gerrie	196707
29	MA32	Cathie	197732
30	MA33	Tommy	197773
31	MA34	Charmine	198752
32	MA35	Camile	199475

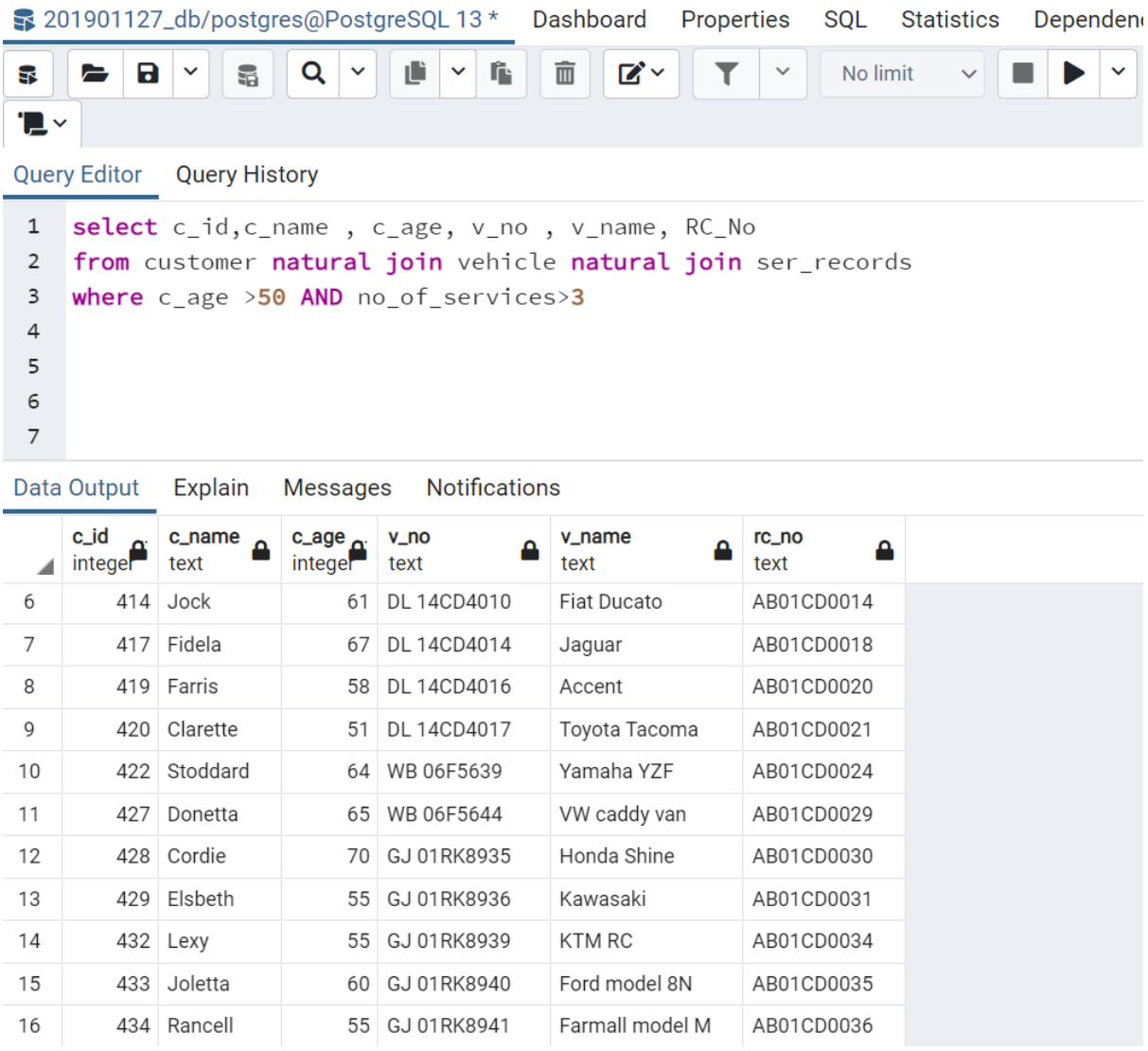
Number of tuples = 32

7. **List all the customers and their vehicle details who have had done services more than 3 times and also are above 50.**

- **select c_id,c_name , c_age, v_no , v_name, RC_No**

  from customer NATURAL JOIN vehicle NATURAL JOIN ser_records

  where c_age >50 AND no_of_services>3

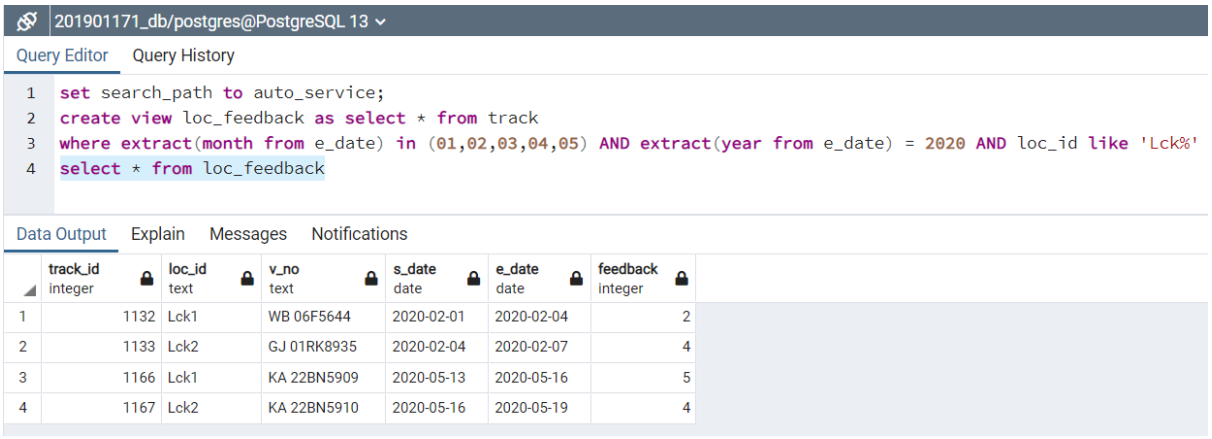


	c_id integer	c_name text	c_age integer	v_no text	v_name text	rc_no text
6	414	Jock	61	DL 14CD4010	Fiat Ducato	AB01CD0014
7	417	Fidela	67	DL 14CD4014	Jaguar	AB01CD0018
8	419	Farris	58	DL 14CD4016	Accent	AB01CD0020
9	420	Clarette	51	DL 14CD4017	Toyota Tacoma	AB01CD0021
10	422	Stoddard	64	WB 06F5639	Yamaha YZF	AB01CD0024
11	427	Donetta	65	WB 06F5644	VW caddy van	AB01CD0029
12	428	Cordie	70	GJ 01RK8935	Honda Shine	AB01CD0030
13	429	Elsbeth	55	GJ 01RK8936	Kawasaki	AB01CD0031
14	432	Lexy	55	GJ 01RK8939	KTM RC	AB01CD0034
15	433	Joletta	60	GJ 01RK8940	Ford model 8N	AB01CD0035
16	434	Rancell	55	GJ 01RK8941	Farmall model M	AB01CD0036

Number of tuples = 16

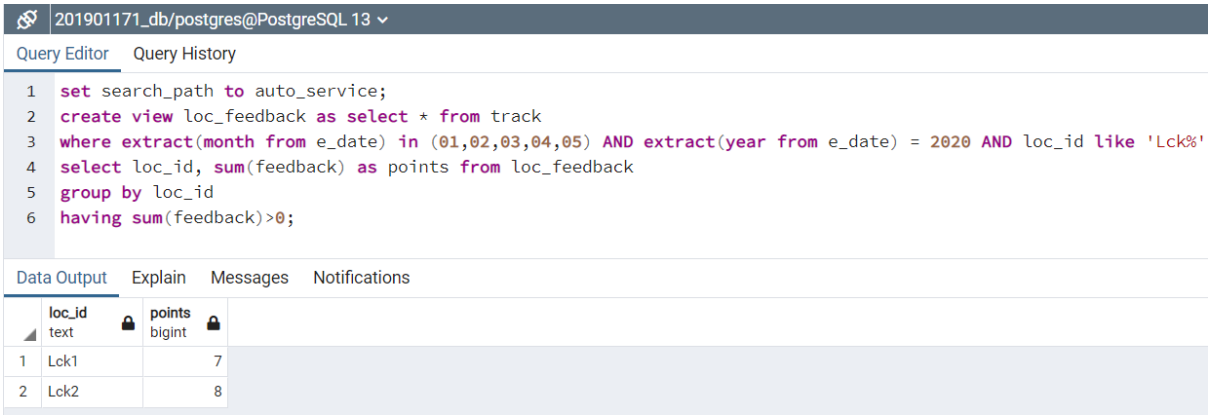
8. **Find total progress of all the centers at Lucknow from January,2020 to May 2020.**

- ```
  create view loc_feedback as select * from track
  where extract(month from e_date) in (01,02,03,04,05) AND
  extract(year from e_date) 2020 AND loc_id like 'Lck%'
  ```



	track_id integer	loc_id text	v_no text	s_date date	e_date date	feedback integer
1	1132	Lck1	WB 06F5644	2020-02-01	2020-02-04	2
2	1133	Lck2	GJ 01RK8935	2020-02-04	2020-02-07	4
3	1166	Lck1	KA 22BN5909	2020-05-13	2020-05-16	5
4	1167	Lck2	KA 22BN5910	2020-05-16	2020-05-19	4

```
select loc_id, sum(feedback) as points from loc_feedback
group by loc_id
having sum(feedback)>0;
```

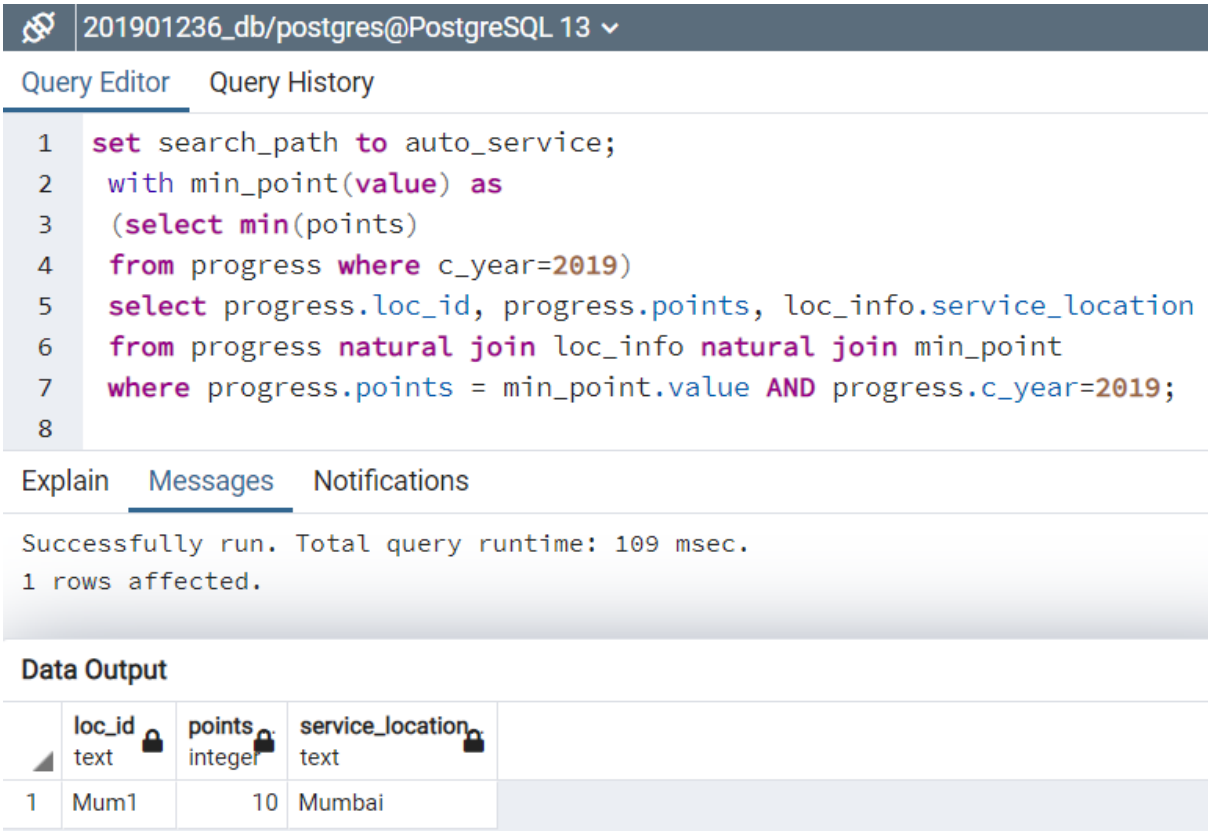


	loc_id text	points bigint
1	Lck1	7
2	Lck2	8

Number of tuples = 2
```

9. **Find the centre with the worst performance in the year 2019 along with its location.**

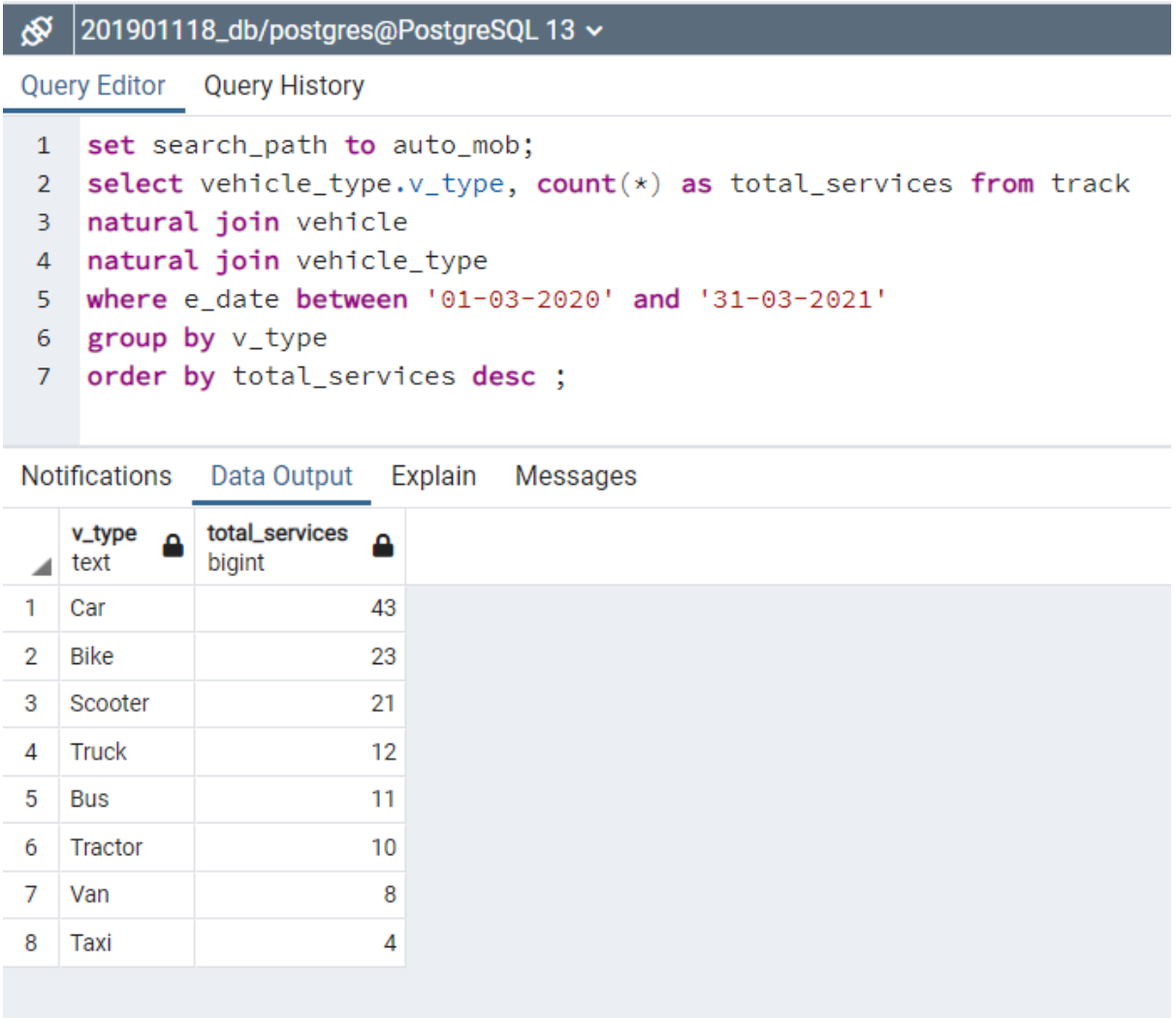
- ```
  with min_point(value) as
  (select min(points) from progress where c_year=2019)
  select progress.loc_id, progress.points, loc_info.service_location
  from progress NATURAL JOIN loc_info NATURAL JOIN min_point
  where progress.points = min_point.value AND progress.c_year=2019;
  ```



Number of tuples = 1
  ```

10. **List all vehicle types by descending order on which they have serviced most between march 2020 to march 2021**

- ```
  select vehicle_type.v_type, count(*) as total_services from track
  NATURAL JOIN vehicle
  NATURAL JOIN vehicle_type
  where e_date between '01-03-2020' AND '31-03-2021'
  group by v_type
  order by total_services desc;
  ```



	v_type text	total_services bigint
1	Car	43
2	Bike	23
3	Scooter	21
4	Truck	12
5	Bus	11
6	Tractor	10
7	Van	8
8	Taxi	4

Number of tuples = 8
  ```

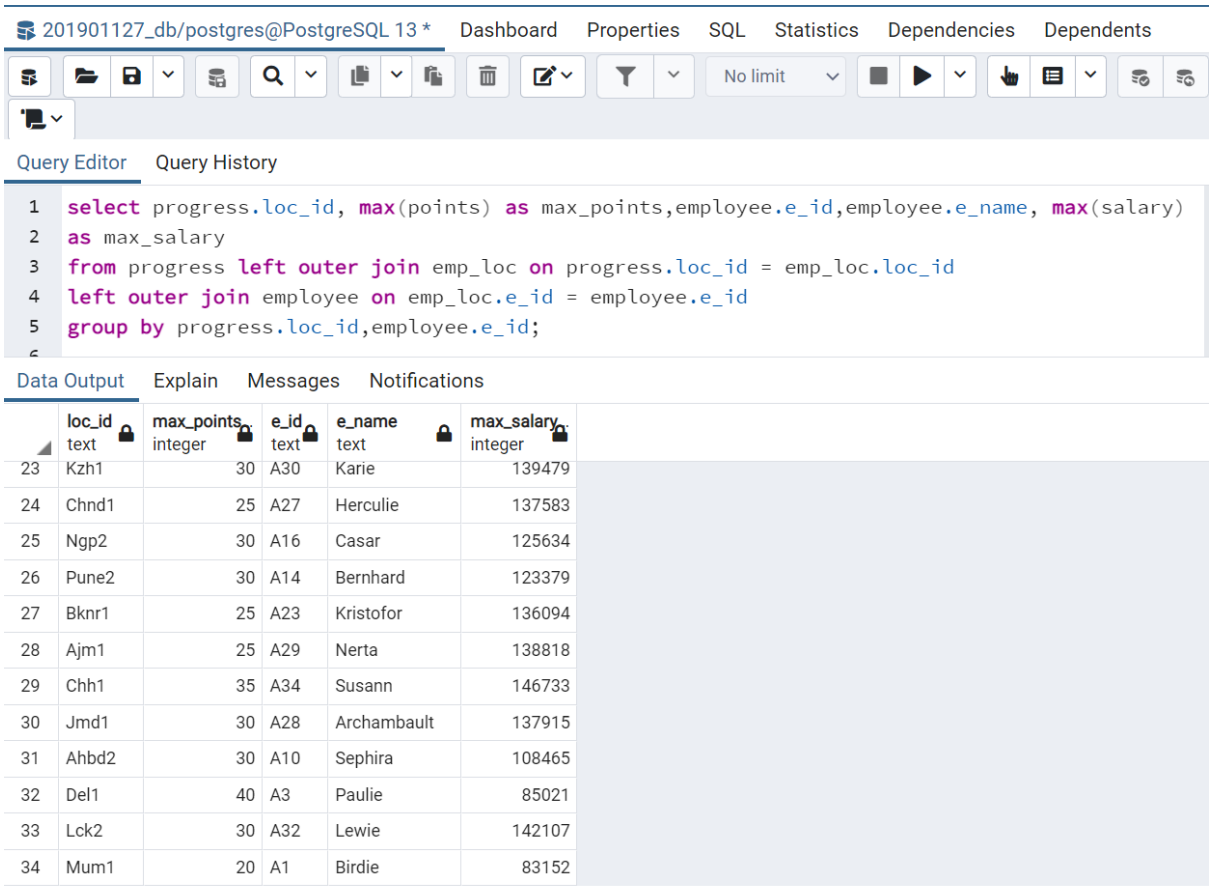
11. **List all the service centers with the highest progress points at any year and also find their employees' highest salary and name and ID.**

- **select progress.loc_id, max(points) as max_points,employee.e_id,employee.e_name, max(salary) as max_salary**

  from progress LEFT OUTER JOIN emp_loc on progress.loc_id = emp_loc.loc_id

  LEFT OUTER JOIN employee on emp_loc.e_id = employee.e_id

  group by progress.loc_id,employee.e_id;



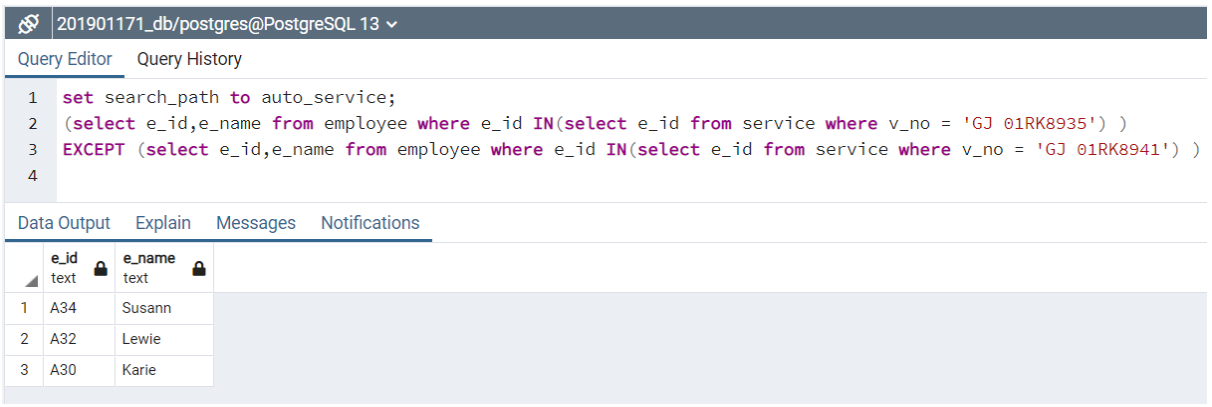
	loc_id text	max_points integer	e_id text	e_name text	max_salary integer
23	Kzh1	30	A30	Karie	139479
24	Chnd1	25	A27	Herculie	137583
25	Ngp2	30	A16	Casar	125634
26	Pune2	30	A14	Bernhard	123379
27	Bknr1	25	A23	Kristofor	136094
28	Ajm1	25	A29	Nerta	138818
29	Chh1	35	A34	Susann	146733
30	Jmd1	30	A28	Archambault	137915
31	Ahbd2	30	A10	Sephira	108465
32	Del1	40	A3	Paulie	85021
33	Lck2	30	A32	Lewie	142107
34	Mum1	20	A1	Birdie	83152

Number of tuples = 34

12. **Find the agent who followed up the service for GJ 01RK8935 but not for GJ 01RK8941.**

- **(select e_id,e_name from employee where e_id IN(select e_id from service where v_no = 'GJ 01RK8935')) EXCEPT**

  (select e_id,e_name from employee where e_id IN(select e_id from service where v_no = 'GJ 01RK8941'))



Number of tuples = 3

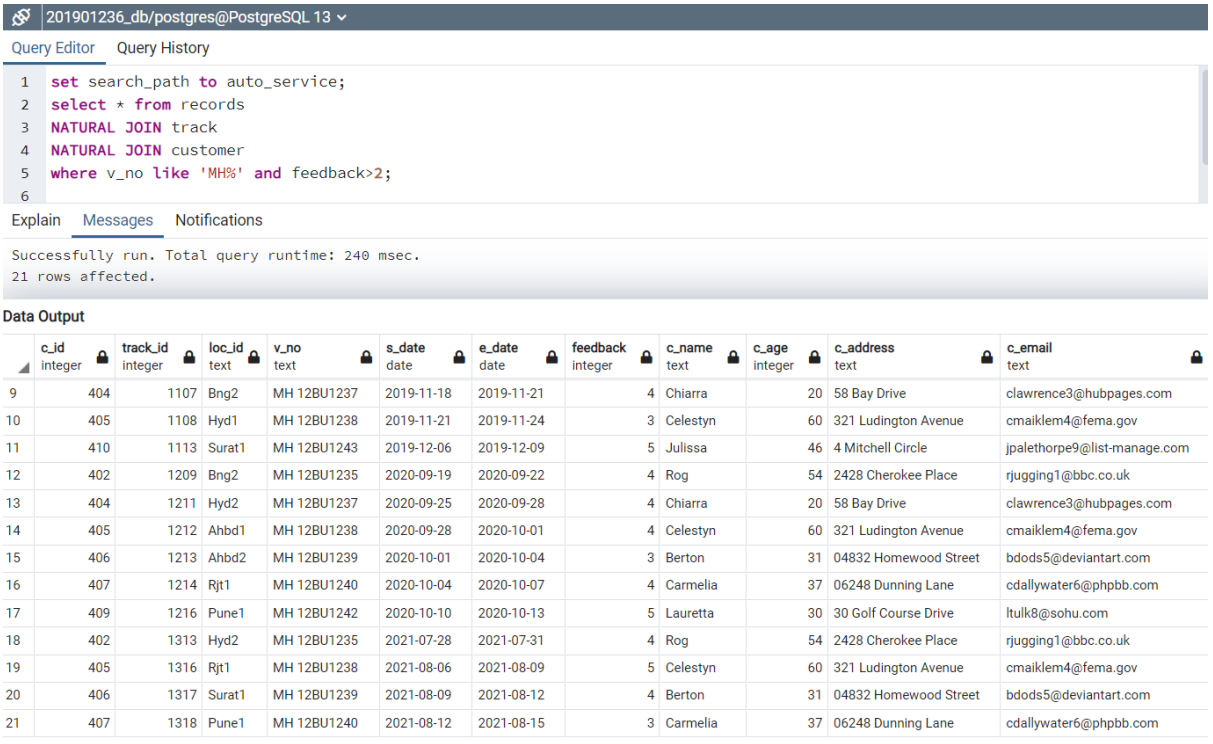
13. **List all the Maharashtra passing vehicles that have given a good feedback of more than 2 along with their service history.**

- **select * from records**

  NATURAL JOIN track

  NATURAL JOIN customer

  where v_no like 'MH%' and feedback>2;



201901236_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2  select * from records
3  NATURAL JOIN track
4  NATURAL JOIN customer
5  where v_no like 'MH%' and feedback>2;
6
```

Explain | Messages | Notifications

```
Successfully run. Total query runtime: 240 msec.
21 rows affected.
```

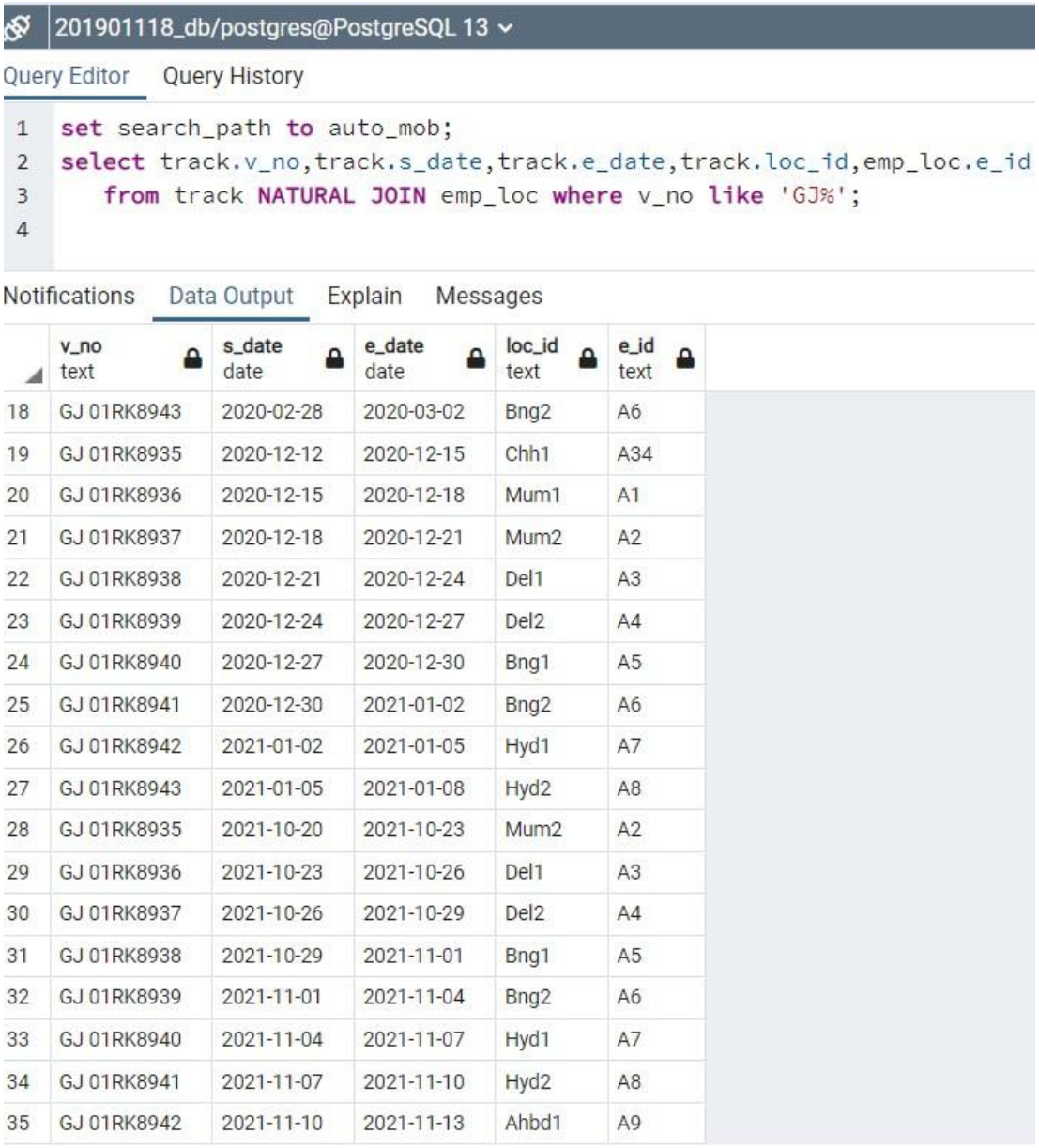
Data Output

	c_id integer	track_id integer	loc_id text	v_no text	s_date date	e_date date	feedback integer	c_name text	c_age integer	c_address text	c_email text
9	404	1107	Bng2	MH 12BU1237	2019-11-18	2019-11-21	4	Chiarra	20	58 Bay Drive	clawrence3@hubpages.com
10	405	1108	Hyd1	MH 12BU1238	2019-11-21	2019-11-24	3	Celestyn	60	321 Ludington Avenue	cmaiklem4@fema.gov
11	410	1113	Surat1	MH 12BU1243	2019-12-06	2019-12-09	5	Julissa	46	4 Mitchell Circle	jpalethorpe9@list-manage.com
12	402	1209	Bng2	MH 12BU1235	2020-09-19	2020-09-22	4	Rog	54	2428 Cherokee Place	rjugging1@bbc.co.uk
13	404	1211	Hyd2	MH 12BU1237	2020-09-25	2020-09-28	4	Chiarra	20	58 Bay Drive	clawrence3@hubpages.com
14	405	1212	Ahbd1	MH 12BU1238	2020-09-28	2020-10-01	4	Celestyn	60	321 Ludington Avenue	cmaiklem4@fema.gov
15	406	1213	Ahbd2	MH 12BU1239	2020-10-01	2020-10-04	3	Berton	31	04832 Homewood Street	bdods5@deviantart.com
16	407	1214	Rjt1	MH 12BU1240	2020-10-04	2020-10-07	4	Carmelia	37	06248 Dunning Lane	cdallywater6@phpbb.com
17	409	1216	Pune1	MH 12BU1242	2020-10-10	2020-10-13	5	Lauretta	30	30 Golf Course Drive	ltulk8@sohu.com
18	402	1313	Hyd2	MH 12BU1235	2021-07-28	2021-07-31	4	Rog	54	2428 Cherokee Place	rjugging1@bbc.co.uk
19	405	1316	Rjt1	MH 12BU1238	2021-08-06	2021-08-09	5	Celestyn	60	321 Ludington Avenue	cmaiklem4@fema.gov
20	406	1317	Surat1	MH 12BU1239	2021-08-09	2021-08-12	4	Berton	31	04832 Homewood Street	bdods5@deviantart.com
21	407	1318	Pune1	MH 12BU1240	2021-08-12	2021-08-15	3	Carmelia	37	06248 Dunning Lane	cdallywater6@phpbb.com

Number of tuples = 21

14. **Select vehicle which has passing state 'Gujarat' and their track id, loc id, service start date and end date with associated employee id.**

- ```
  select track.v_no,track.s_date,track.e_date,track.loc_id,emp_loc.e_id
  from track NATURAL JOIN emp_loc where v_no like 'GJ%';
  ```



201901118_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_mob;
2  select track.v_no,track.s_date,track.e_date,track.loc_id,emp_loc.e_id
3     from track NATURAL JOIN emp_loc where v_no like 'GJ%';
4
```

Notifications | Data Output | Explain | Messages

	v_no text	s_date date	e_date date	loc_id text	e_id text
18	GJ 01RK8943	2020-02-28	2020-03-02	Bng2	A6
19	GJ 01RK8935	2020-12-12	2020-12-15	Chh1	A34
20	GJ 01RK8936	2020-12-15	2020-12-18	Mum1	A1
21	GJ 01RK8937	2020-12-18	2020-12-21	Mum2	A2
22	GJ 01RK8938	2020-12-21	2020-12-24	Del1	A3
23	GJ 01RK8939	2020-12-24	2020-12-27	Del2	A4
24	GJ 01RK8940	2020-12-27	2020-12-30	Bng1	A5
25	GJ 01RK8941	2020-12-30	2021-01-02	Bng2	A6
26	GJ 01RK8942	2021-01-02	2021-01-05	Hyd1	A7
27	GJ 01RK8943	2021-01-05	2021-01-08	Hyd2	A8
28	GJ 01RK8935	2021-10-20	2021-10-23	Mum2	A2
29	GJ 01RK8936	2021-10-23	2021-10-26	Del1	A3
30	GJ 01RK8937	2021-10-26	2021-10-29	Del2	A4
31	GJ 01RK8938	2021-10-29	2021-11-01	Bng1	A5
32	GJ 01RK8939	2021-11-01	2021-11-04	Bng2	A6
33	GJ 01RK8940	2021-11-04	2021-11-07	Hyd1	A7
34	GJ 01RK8941	2021-11-07	2021-11-10	Hyd2	A8
35	GJ 01RK8942	2021-11-10	2021-11-13	Ahbd1	A9

Number of tuples = 35
```

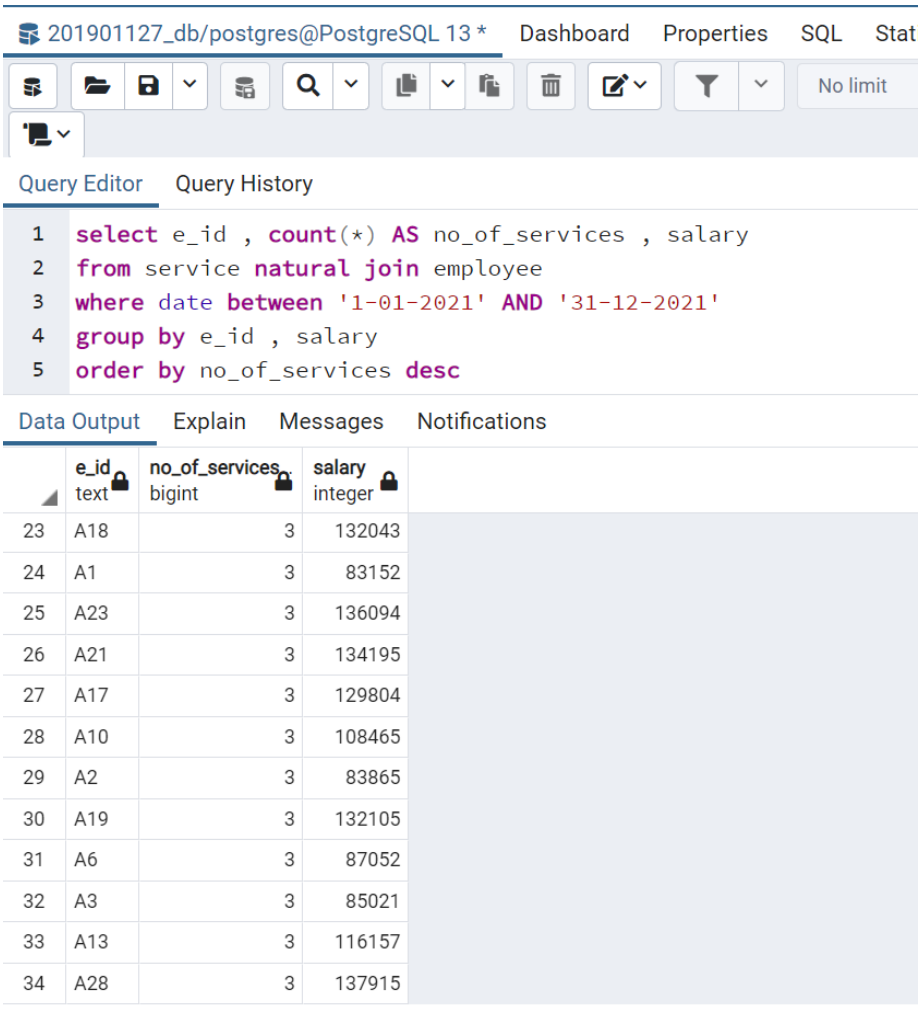
15. list all the employees in descending order of the num_of_services they did in year 2021 and their salary.

- **select e_id , count(*) AS no_of_services , salary from service NATURAL JOIN employee**

  where date between '1-01-2021' AND '31-12-2021'

  group by e_id , salary

  order by no_of_services desc



	e_id text	no_of_services bigint	salary integer
23	A18	3	132043
24	A1	3	83152
25	A23	3	136094
26	A21	3	134195
27	A17	3	129804
28	A10	3	108465
29	A2	3	83865
30	A19	3	132105
31	A6	3	87052
32	A3	3	85021
33	A13	3	116157
34	A28	3	137915

Number of tuples = 34

16. **Give the customer a special discount 10% if his bill is greater than 1000 rupees else give 5% discount. His vehicle number = GJ 01RK8935 and service type = oil/oil filter change. Offer valid for today(18-11-2021) only. Show vehicle number , date and bill amount as a output.**

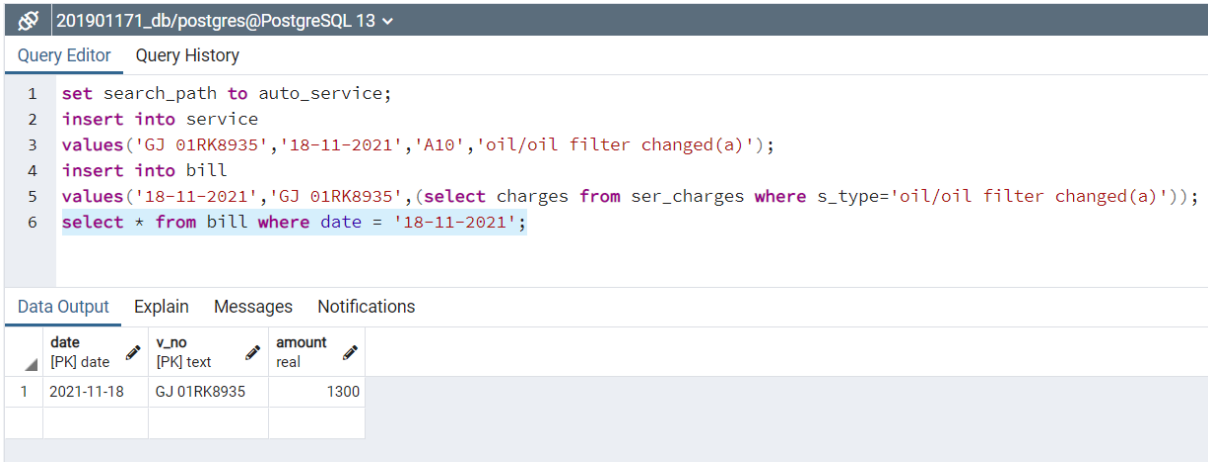
- ```
  insert into service values('GJ 01RK8935','18-11-2021','A10','oil/oil filter changed(a)');

  insert into bill

  values('18-11-2021','GJ 01RK8935',(select charges from ser_charges

  where s_type='oil/oil filter changed(a)'));

  select * from bill where date = '18-11-2021';
  ```



Here , first we have to insert data for that vehicle in the service table , after that we can find the bill for that vehicle.

Then we apply given conditions to find the total amount.

```
update bill

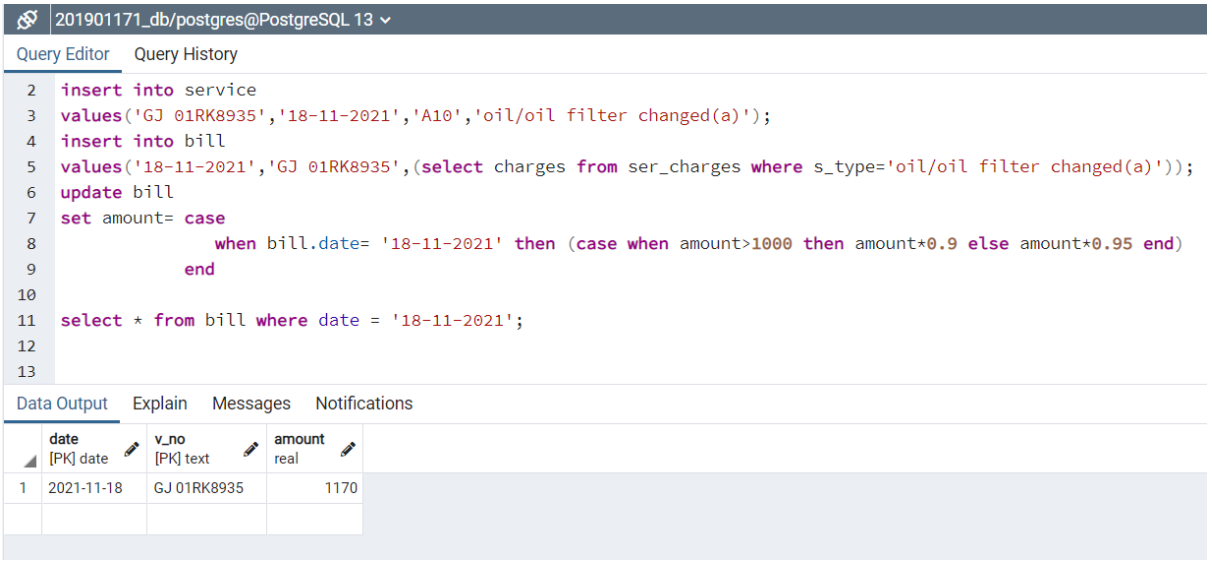
set amount= case

        when bill.date= '18-11-2021' then (case when amount>1000 then

                                        amount*0.9   else amount*0.95 end)

    end

select * from bill where date = '18-11-2021';
```
```



Number of tuples = 1

Here , in the above screen shot we can see that,final amount in the Bill table updated successfully.

17. **Help the company to give awards of AGENT OF THE YEAR to top 5 agents of the company.**

```
create view ag as

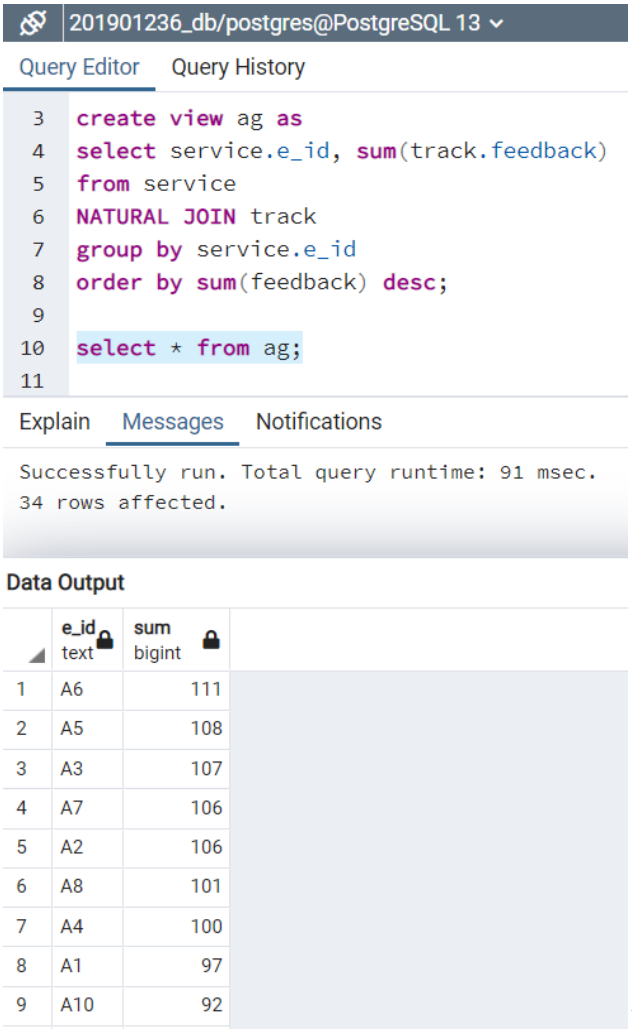
select service.e_id, sum(track.feedback)

from service

NATURAL JOIN track

group by service.e_id

order by sum(feedback) desc;
```



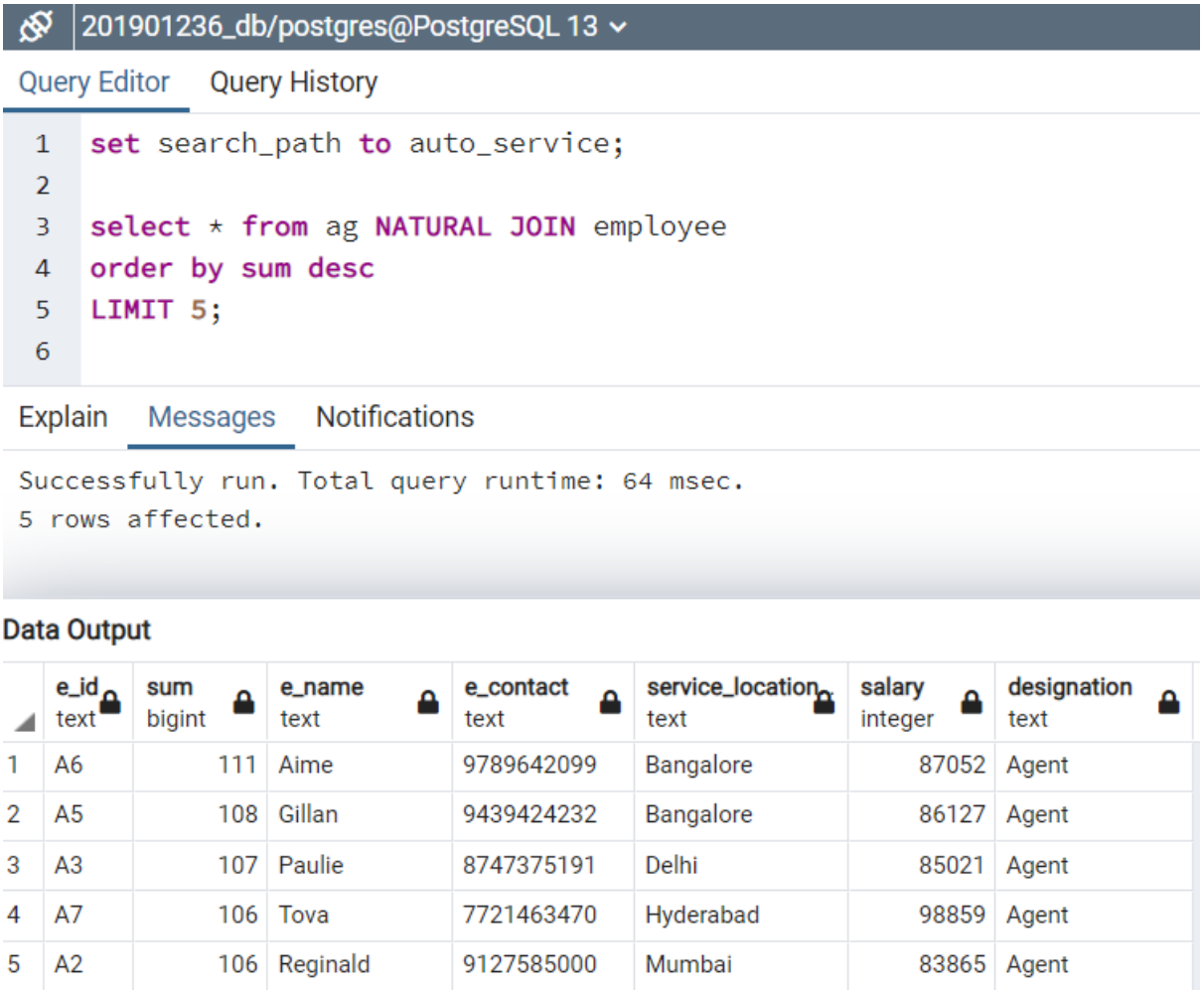
	e_id text	sum bigint
1	A6	111
2	A5	108
3	A3	107
4	A7	106
5	A2	106
6	A8	101
7	A4	100
8	A1	97
9	A10	92

View

```
select * from ag NATURAL JOIN employee

order by sum desc

LIMIT 5;
```



Data Output

	e_id text	sum bigint	e_name text	e_contact text	service_location text	salary integer	designation text
1	A6	111	Aime	9789642099	Bangalore	87052	Agent
2	A5	108	Gillan	9439424232	Bangalore	86127	Agent
3	A3	107	Paulie	8747375191	Delhi	85021	Agent
4	A7	106	Tova	7721463470	Hyderabad	98859	Agent
5	A2	106	Reginald	9127585000	Mumbai	83865	Agent

Number of tuples = 5

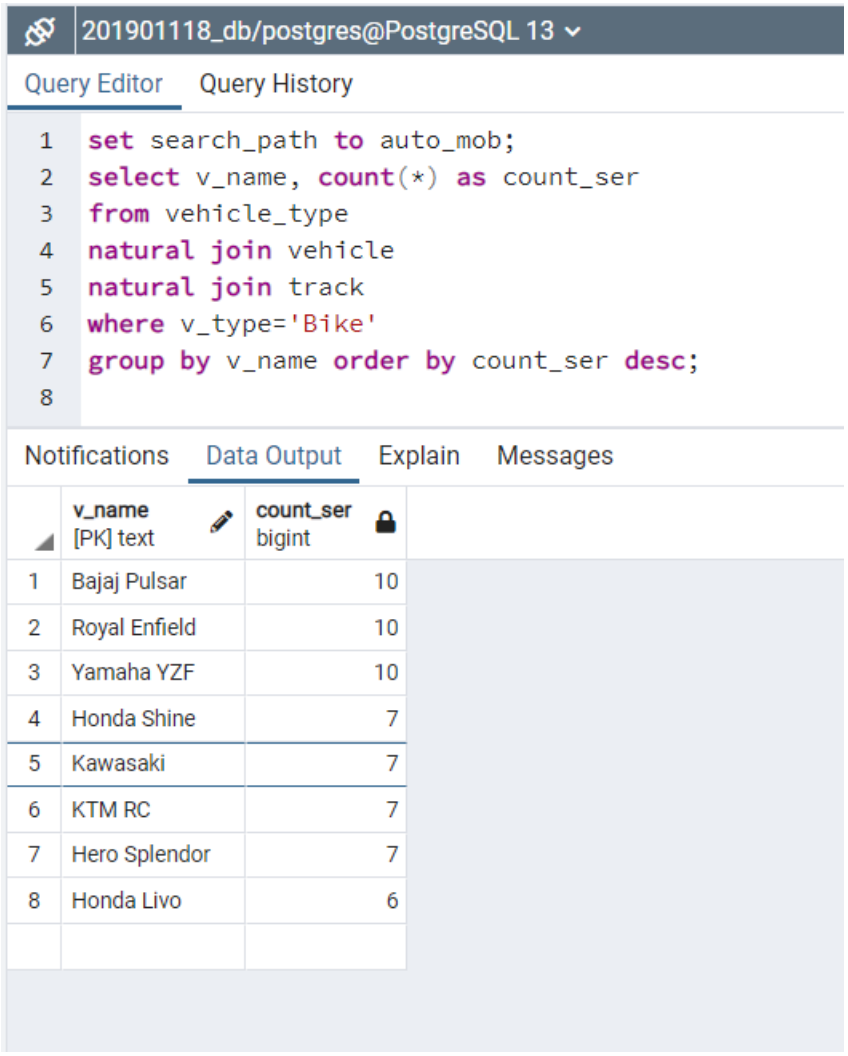
- From this the CEO can infer that these are the top 5 hardworking agents across the nation and hence they might be eligible for promotion later.
- The table gives names and posting information of such agents.

18. **List most serviced bike vehicles' names in descending order of number of service.**

- **select v_name, count(*) as count_ser**

  from vehicle_type NATURAL JOIN vehicle NATURAL JOIN track where v_type='Bike' group by v_name

  order by count_ser desc;



Number of tuples = 8

19. **Find all the centers and the employee name and ID who is working at that particular center where the total feedback points is greater than the average of the total points of all centers.**

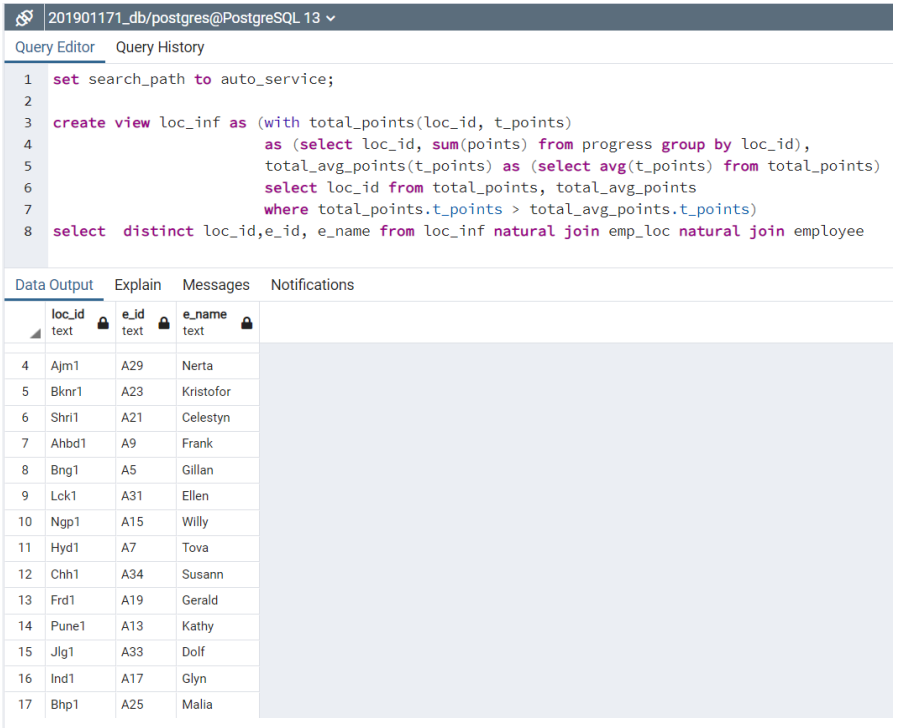
- **create view loc_inf AS**

  (with total_points(loc_id, t_points) as (select loc_id, sum(points) from progress group by loc_id), total_avg_points(t_points) as (select avg(t_points) from total_points)

  select loc_id from total_points, total_avg_points

  where total_points.t_points > total_avg_points.t_points)

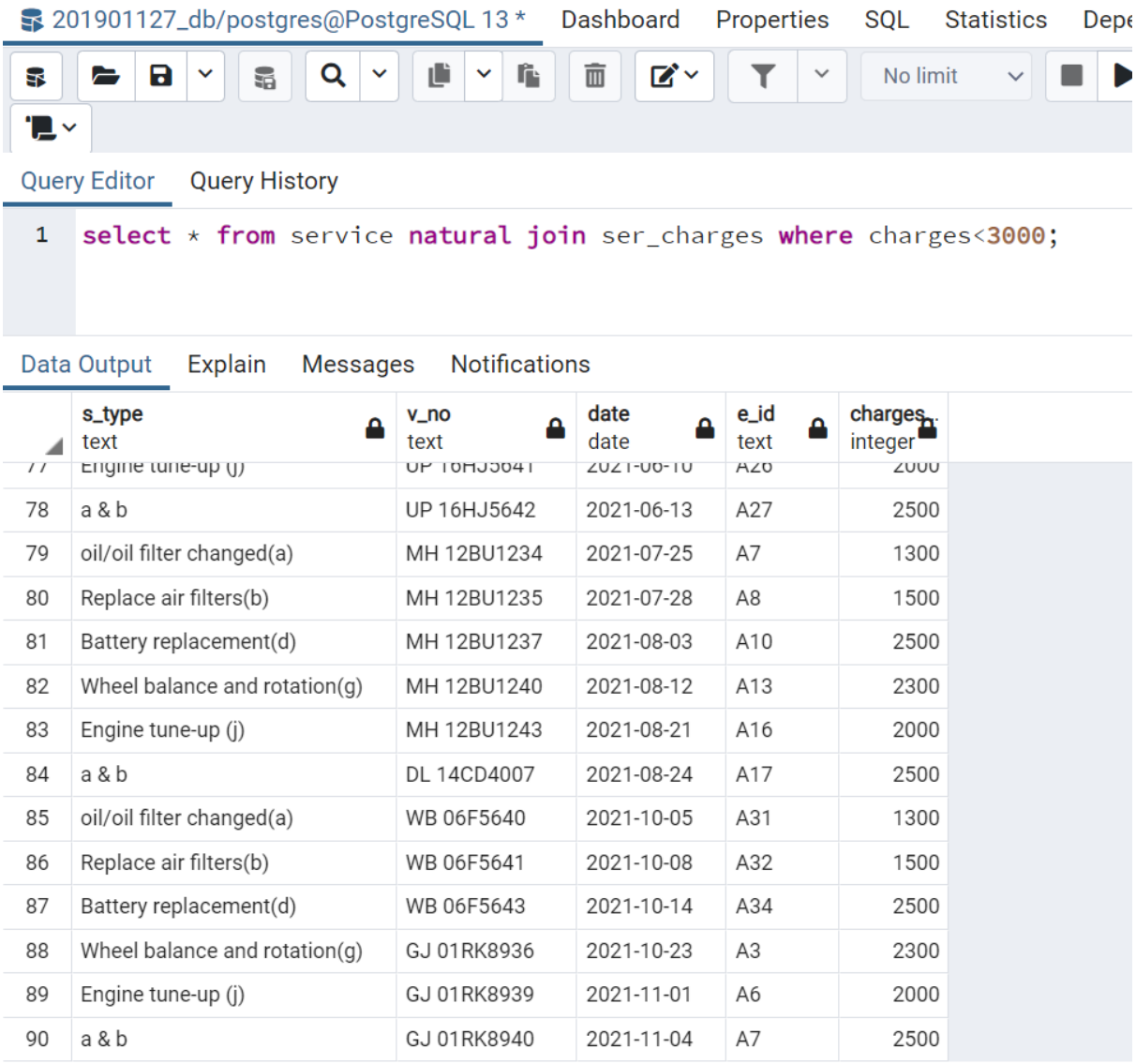
  select distinct loc_id,e_id, e_name from loc_inf natural join emp_loc natural join employee



Number of tuples = 17

20. Find the complete service details with service charges less than 3000.

- **select * from service natural join ser_charges where charges<3000;**

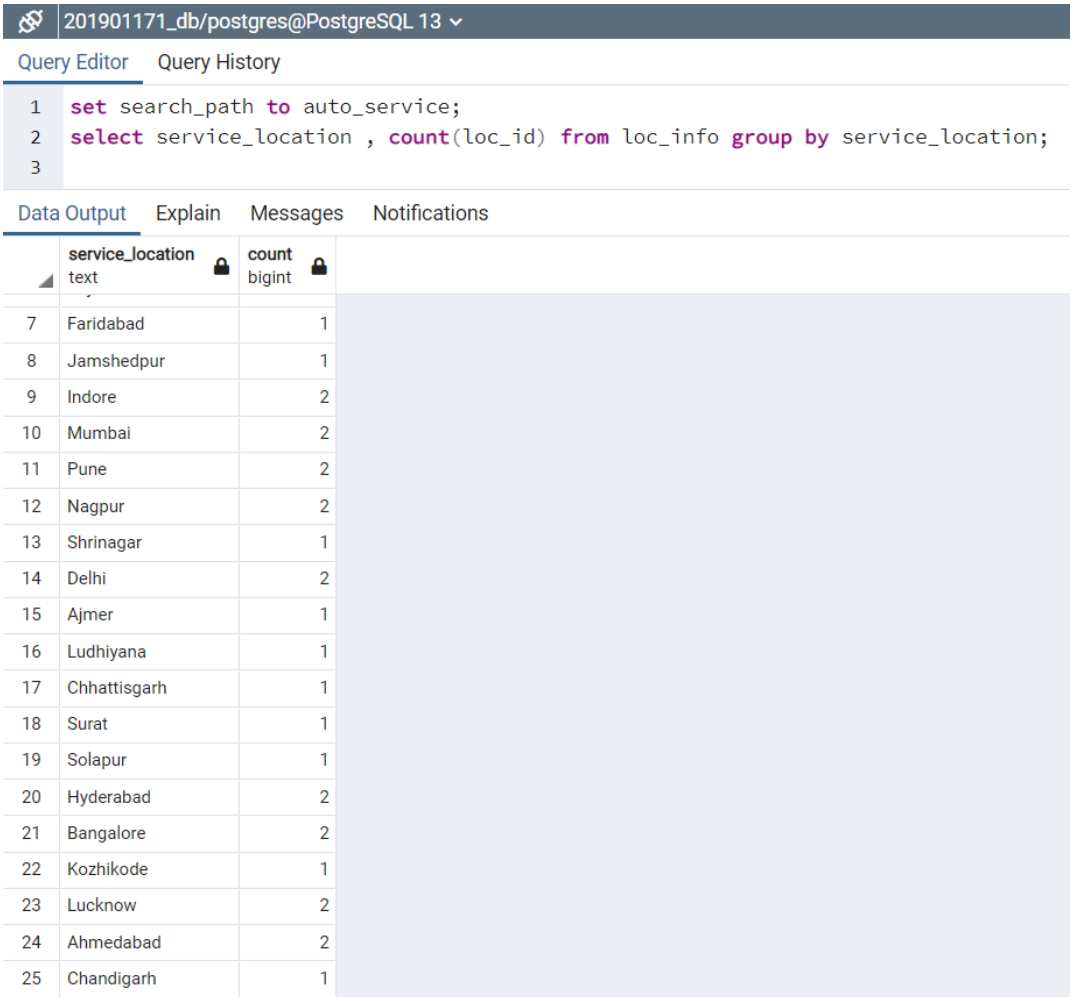


	s_type text	v_no text	date date	e_id text	charges integer
77	Engine tune-up (j)	UP 16HJ5641	2021-06-10	A26	2000
78	a & b	UP 16HJ5642	2021-06-13	A27	2500
79	oil/oil filter changed(a)	MH 12BU1234	2021-07-25	A7	1300
80	Replace air filters(b)	MH 12BU1235	2021-07-28	A8	1500
81	Battery replacement(d)	MH 12BU1237	2021-08-03	A10	2500
82	Wheel balance and rotation(g)	MH 12BU1240	2021-08-12	A13	2300
83	Engine tune-up (j)	MH 12BU1243	2021-08-21	A16	2000
84	a & b	DL 14CD4007	2021-08-24	A17	2500
85	oil/oil filter changed(a)	WB 06F5640	2021-10-05	A31	1300
86	Replace air filters(b)	WB 06F5641	2021-10-08	A32	1500
87	Battery replacement(d)	WB 06F5643	2021-10-14	A34	2500
88	Wheel balance and rotation(g)	GJ 01RK8936	2021-10-23	A3	2300
89	Engine tune-up (j)	GJ 01RK8939	2021-11-01	A6	2000
90	a & b	GJ 01RK8940	2021-11-04	A7	2500

Number of tuples = 90

21. List the locations and the number of centres it has.

- **select service_location , count(loc_id) from loc_info group by service_location;**

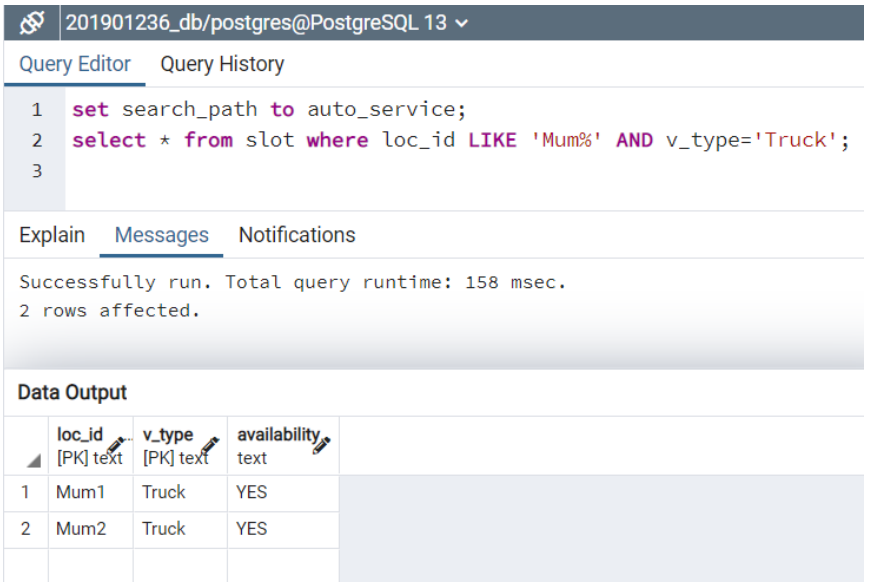


	service_location text	count bigint
7	Faridabad	1
8	Jamshedpur	1
9	Indore	2
10	Mumbai	2
11	Pune	2
12	Nagpur	2
13	Shrinagar	1
14	Delhi	2
15	Ajmer	1
16	Ludhiyana	1
17	Chhattisgarh	1
18	Surat	1
19	Solapur	1
20	Hyderabad	2
21	Bangalore	2
22	Kozhikode	1
23	Lucknow	2
24	Ahmedabad	2
25	Chandigarh	1

Number of tuples = 25

22. **Shows the slot availability status of all the Mumbai centres for trucks.**

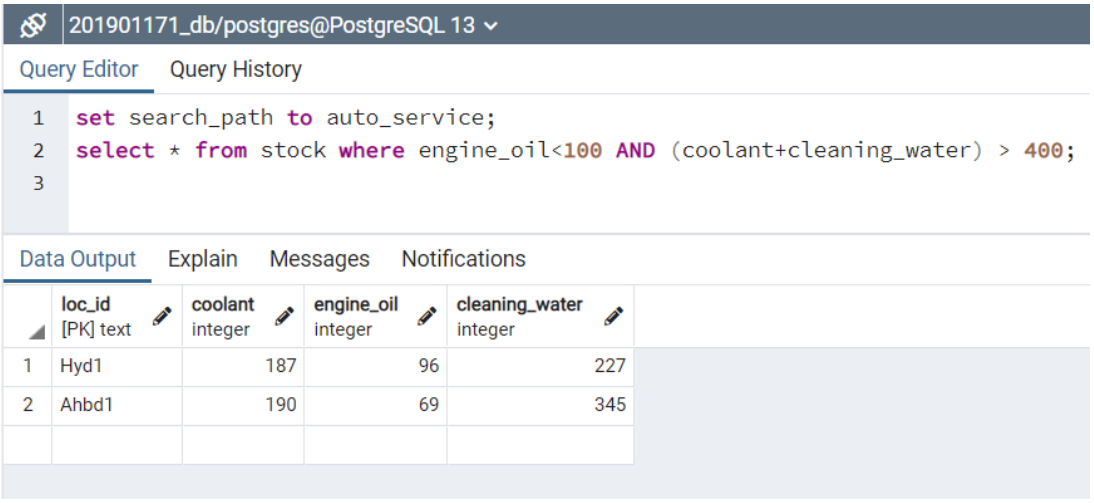
- **select * from slot where loc_id LIKE 'Mum%' AND v_type='Truck';**



Number of tuples = 2

23. **Find all the centres having engine oil less than 100 litres and cleaning water and coolant as a total greater than 400 litres.**

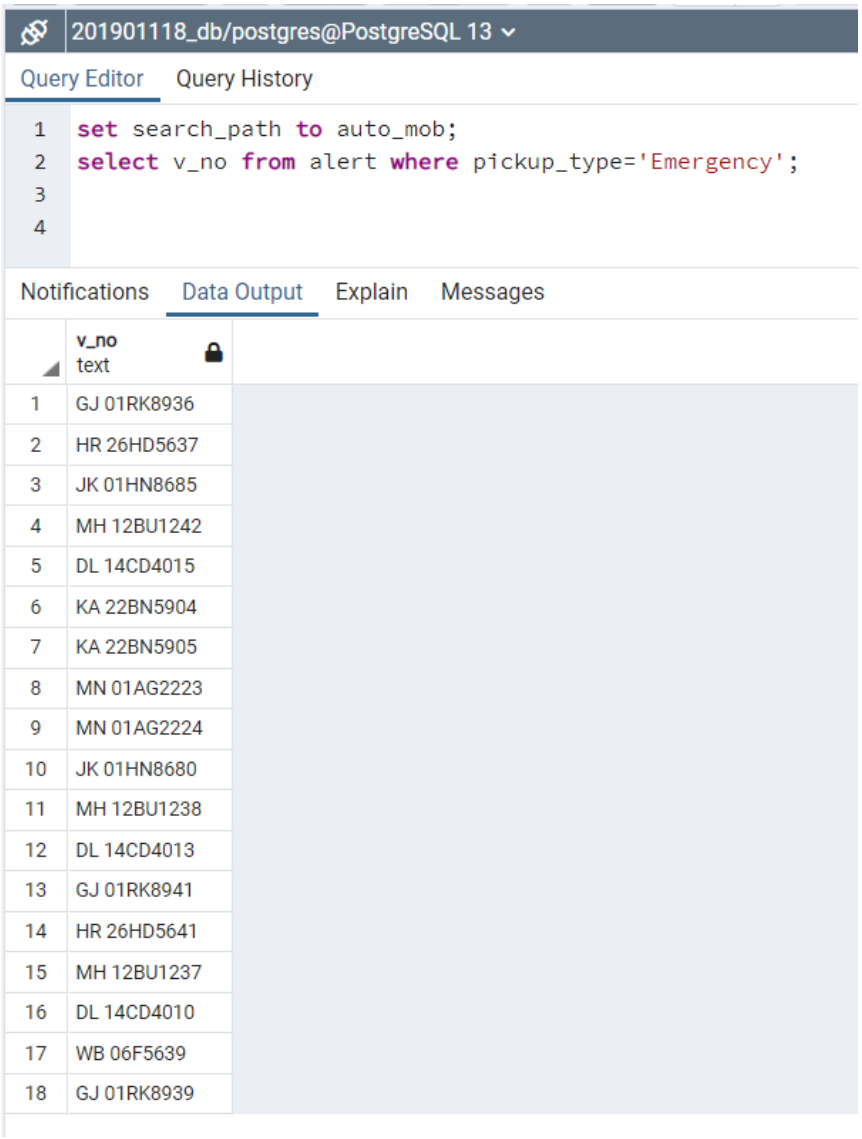
select * from stock where engine_oil<100 AND (coolant+cleaning_water) > 400;



Number of tuples = 2

24. List all the vehicles that came for emergency breakdown service so far.

- **select v_no from alert where pickup_type='Emergency';**



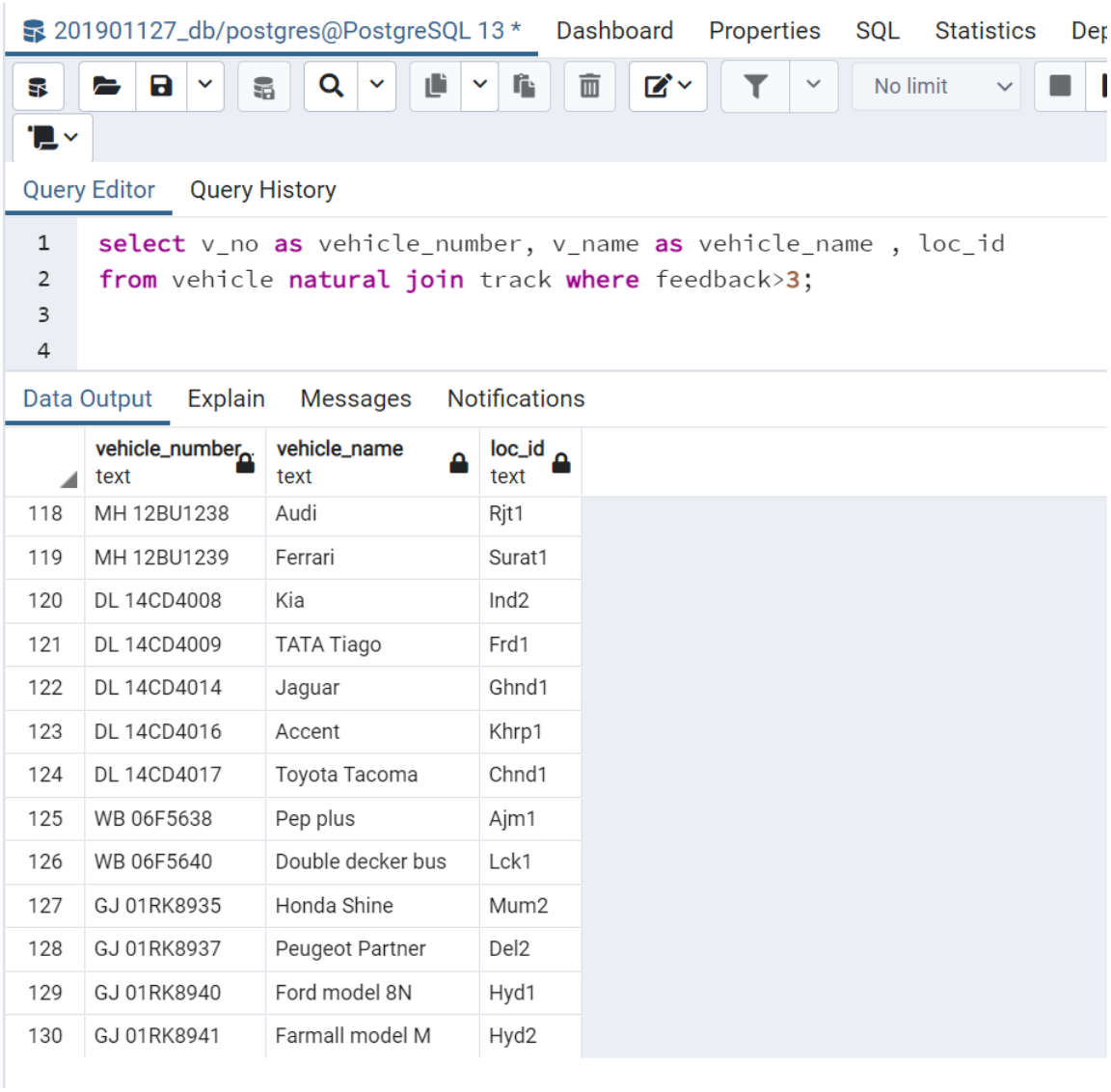
Number of tuples = 18

25. **Find vehicle number, vehicle name and it's service location id , where customer gave more than 3 feedback points.**

- **select v_no as vehicle_number, v_name as vehicle_name , loc_id**

  from vehicle NATURAL JOIN track

  where feedback>3;

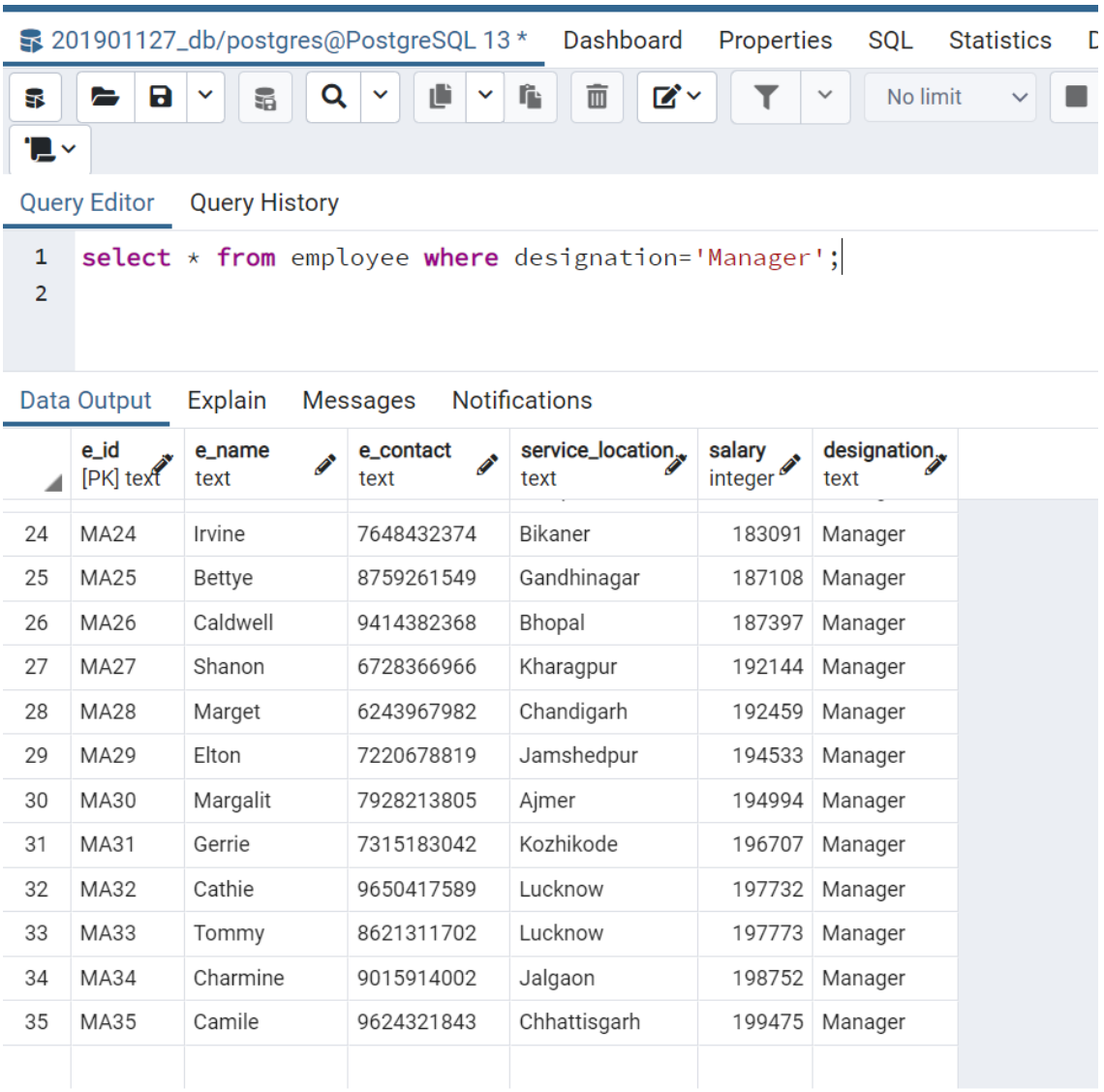


	vehicle_number text	vehicle_name text	loc_id text
118	MH 12BU1238	Audi	Rjt1
119	MH 12BU1239	Ferrari	Surat1
120	DL 14CD4008	Kia	Ind2
121	DL 14CD4009	TATA Tiago	Frd1
122	DL 14CD4014	Jaguar	Ghnd1
123	DL 14CD4016	Accent	Khrp1
124	DL 14CD4017	Toyota Tacoma	Chnd1
125	WB 06F5638	Pep plus	Ajm1
126	WB 06F5640	Double decker bus	Lck1
127	GJ 01RK8935	Honda Shine	Mum2
128	GJ 01RK8937	Peugeot Partner	Del2
129	GJ 01RK8940	Ford model 8N	Hyd1
130	GJ 01RK8941	Farmall model M	Hyd2

Number of tuples = 130

26. List the personal details of all the managers working in the company across India.

- **select * from employee where designation='Manager';**



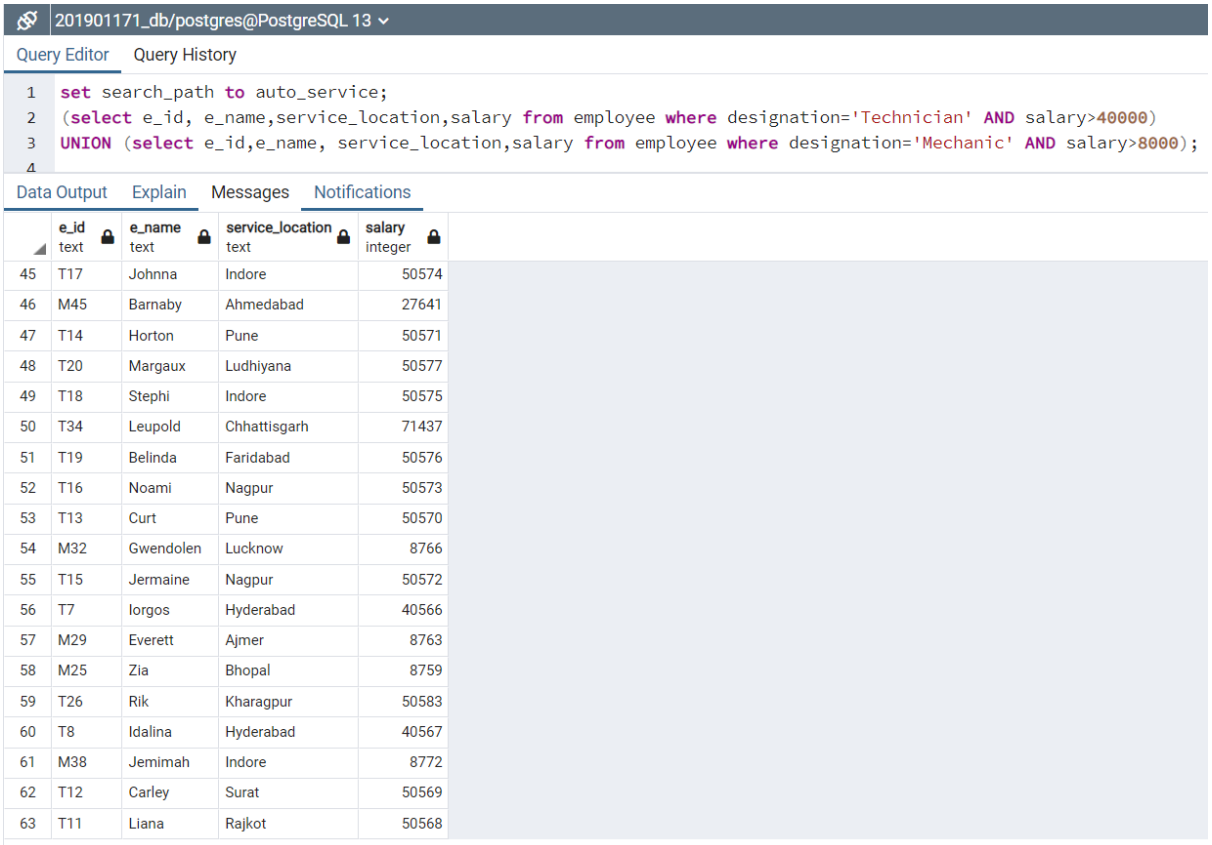
	e_id [PK] text	e_name text	e_contact text	service_location text	salary integer	designation text
24	MA24	Irvine	7648432374	Bikaner	183091	Manager
25	MA25	Bettye	8759261549	Gandhinagar	187108	Manager
26	MA26	Caldwell	9414382368	Bhopal	187397	Manager
27	MA27	Shanon	6728366966	Kharagpur	192144	Manager
28	MA28	Marget	6243967982	Chandigarh	192459	Manager
29	MA29	Elton	7220678819	Jamshedpur	194533	Manager
30	MA30	Margalit	7928213805	Ajmer	194994	Manager
31	MA31	Gerrie	7315183042	Kozhikode	196707	Manager
32	MA32	Cathie	9650417589	Lucknow	197732	Manager
33	MA33	Tommy	8621311702	Lucknow	197773	Manager
34	MA34	Charmine	9015914002	Jalgaon	198752	Manager
35	MA35	Camile	9624321843	Chhattisgarh	199475	Manager

Number of tuples = 35

27. **Find employee id , name, service_location and salary of the employee whose designation is Technician and salary greater than 40000 or whose designation is Mechanic and salary greater than 8000.**

- **(select e_id as ID, name as Name, service_location ,salary from employee where designation= 'Technician' AND salary>40000) UNION**

  (select e_id as ID, name as Name, service_location , salary from employee where designation= 'Mechanic' AND salary>8000);



201901171_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2  (select e_id, e_name,service_location,salary from employee where designation='Technician' AND salary>40000)
3  UNION (select e_id,e_name, service_location,salary from employee where designation='Mechanic' AND salary>8000);
4
```

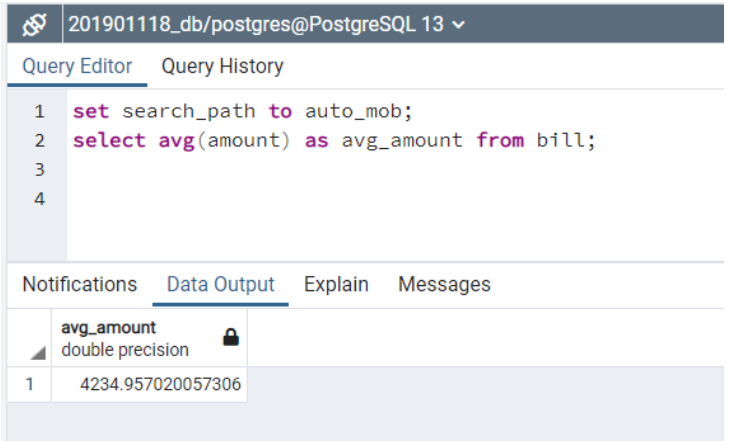
Data Output | Explain | Messages | Notifications

	e_id text	e_name text	service_location text	salary integer
45	T17	Johnna	Indore	50574
46	M45	Barnaby	Ahmedabad	27641
47	T14	Horton	Pune	50571
48	T20	Margaux	Ludhiyana	50577
49	T18	Stephi	Indore	50575
50	T34	Leupold	Chhattisgarh	71437
51	T19	Belinda	Faridabad	50576
52	T16	Noami	Nagpur	50573
53	T13	Curt	Pune	50570
54	M32	Gwendolen	Lucknow	8766
55	T15	Jermaine	Nagpur	50572
56	T7	Iorgos	Hyderabad	40566
57	M29	Everett	Ajmer	8763
58	M25	Zia	Bhopal	8759
59	T26	Rik	Kharagpur	50583
60	T8	Idalina	Hyderabad	40567
61	M38	Jemimah	Indore	8772
62	T12	Carley	Surat	50569
63	T11	Liana	Rajkot	50568

Number of tuples = 63

28. Find the Average bill amount of the company till date.

- **select avg(amount) as avg_amount from bill;**

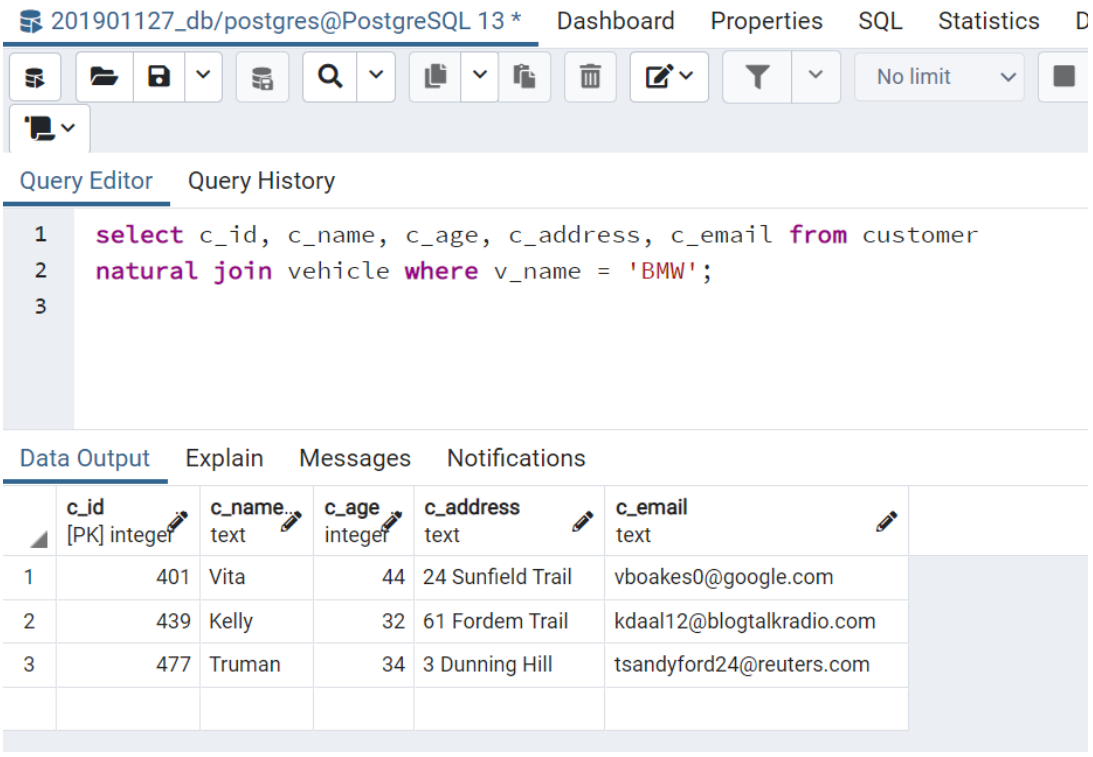


Number of tuples = 1

29. List all the customers owning a BMW car.

- **select c_id, c_name, c_age, c_address, c_email**

  from customer NATURAL JOIN vehicle where v_name = 'BMW';

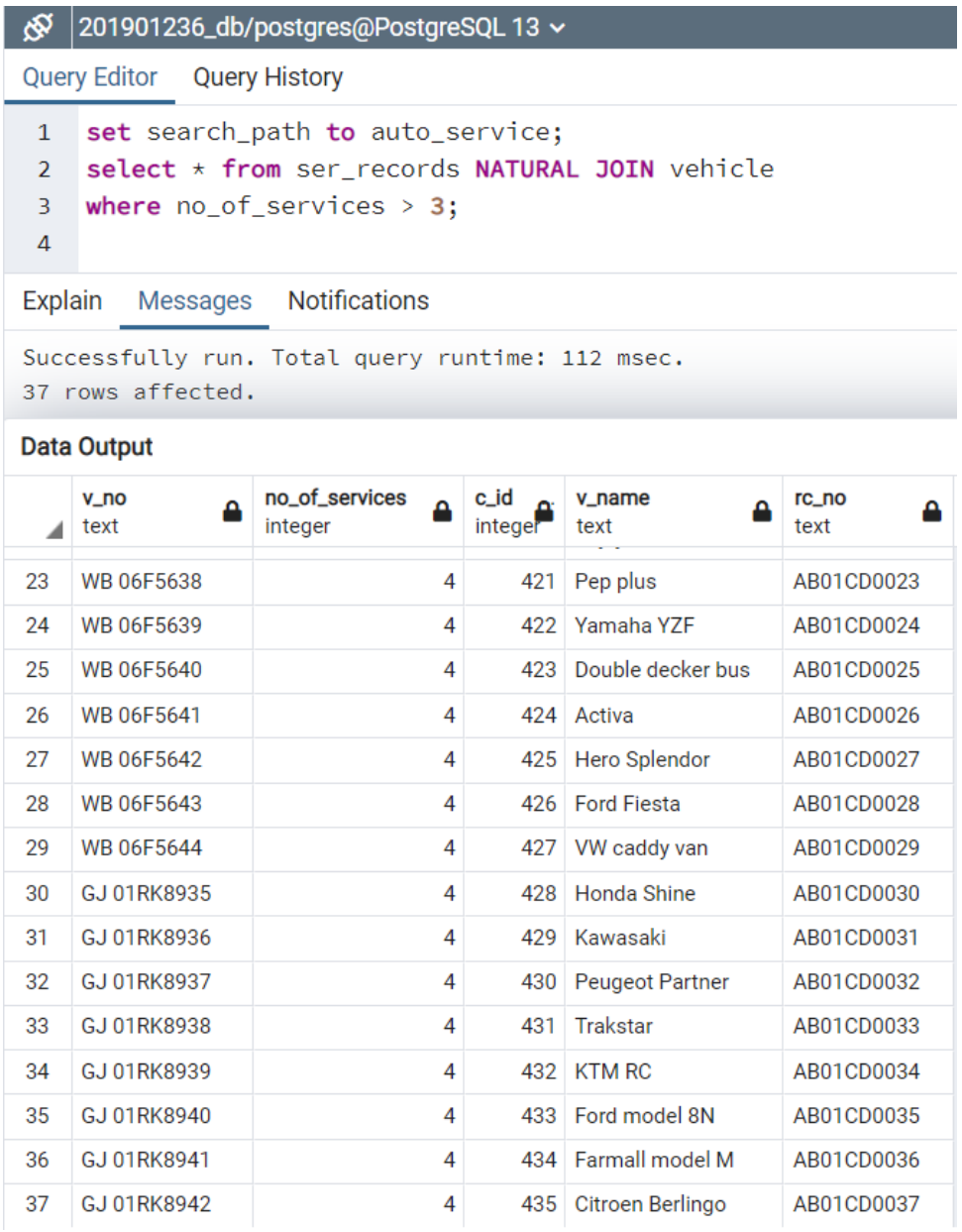


Number of tuples = 3

30. List the vehicles that came for service more than three times so far.

- **select * from ser_records NATURAL JOIN vehicle**

  where no_of_services > 3;



201901236_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2  select * from ser_records NATURAL JOIN vehicle
3  where no_of_services > 3;
4
```

Explain | Messages | Notifications

```
Successfully run. Total query runtime: 112 msec.
37 rows affected.
```

Data Output

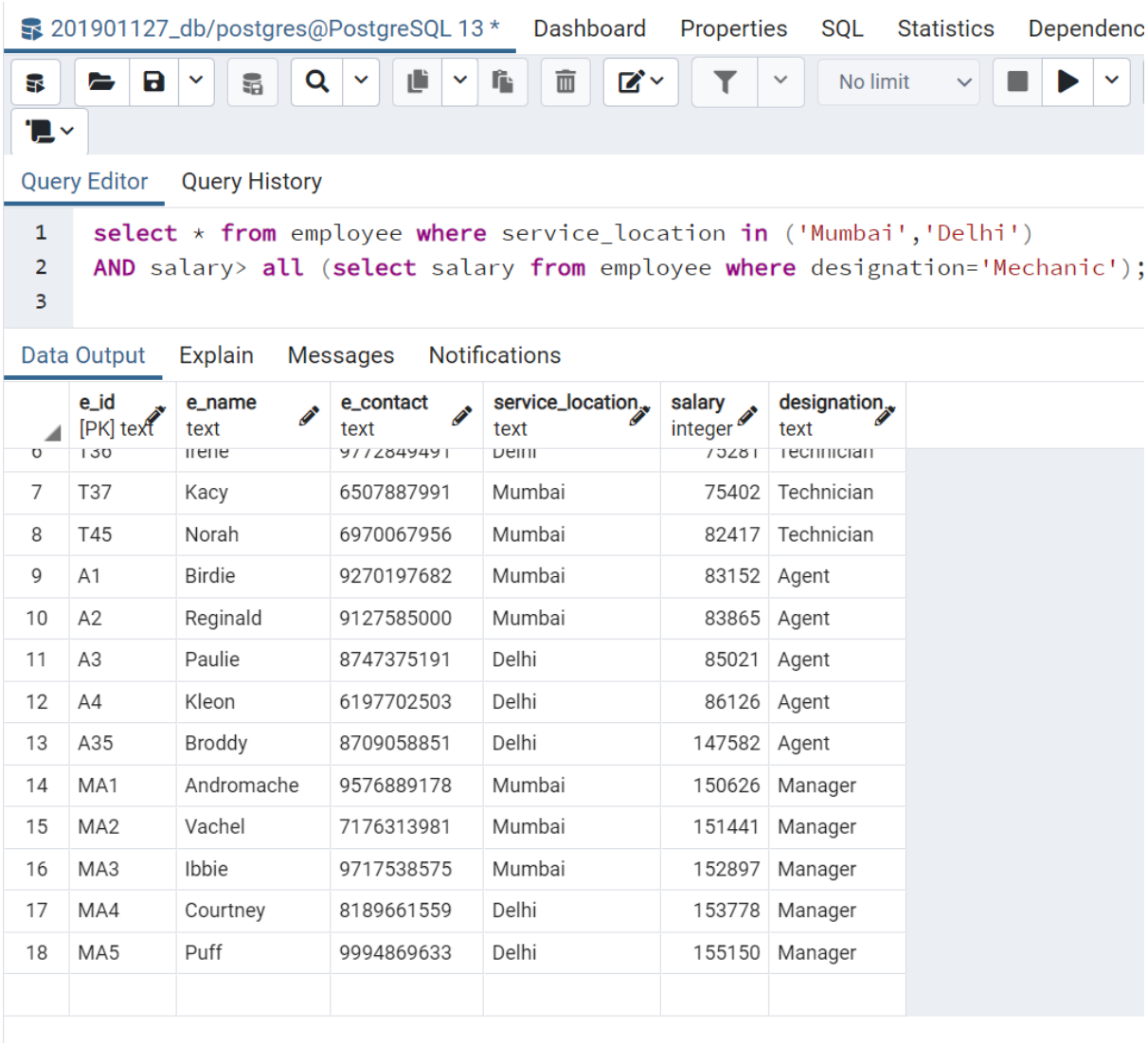
	v_no text	no_of_services integer	c_id integer	v_name text	rc_no text
23	WB 06F5638	4	421	Pep plus	AB01CD0023
24	WB 06F5639	4	422	Yamaha YZF	AB01CD0024
25	WB 06F5640	4	423	Double decker bus	AB01CD0025
26	WB 06F5641	4	424	Activa	AB01CD0026
27	WB 06F5642	4	425	Hero Splendor	AB01CD0027
28	WB 06F5643	4	426	Ford Fiesta	AB01CD0028
29	WB 06F5644	4	427	VW caddy van	AB01CD0029
30	GJ 01RK8935	4	428	Honda Shine	AB01CD0030
31	GJ 01RK8936	4	429	Kawasaki	AB01CD0031
32	GJ 01RK8937	4	430	Peugeot Partner	AB01CD0032
33	GJ 01RK8938	4	431	Trakstar	AB01CD0033
34	GJ 01RK8939	4	432	KTM RC	AB01CD0034
35	GJ 01RK8940	4	433	Ford model 8N	AB01CD0035
36	GJ 01RK8941	4	434	Farmall model M	AB01CD0036
37	GJ 01RK8942	4	435	Citroen Berlingo	AB01CD0037

Number of tuples = 37

31. **List details of employees working at Mumbai and Delhi and whose salary is greater than the salary of all the employees whose designation is mechanic.**

- **select * from employee where service_location in ('Mumbai','Delhi') AND**

  salary> all (select salary from employee where designation='Mechanic');



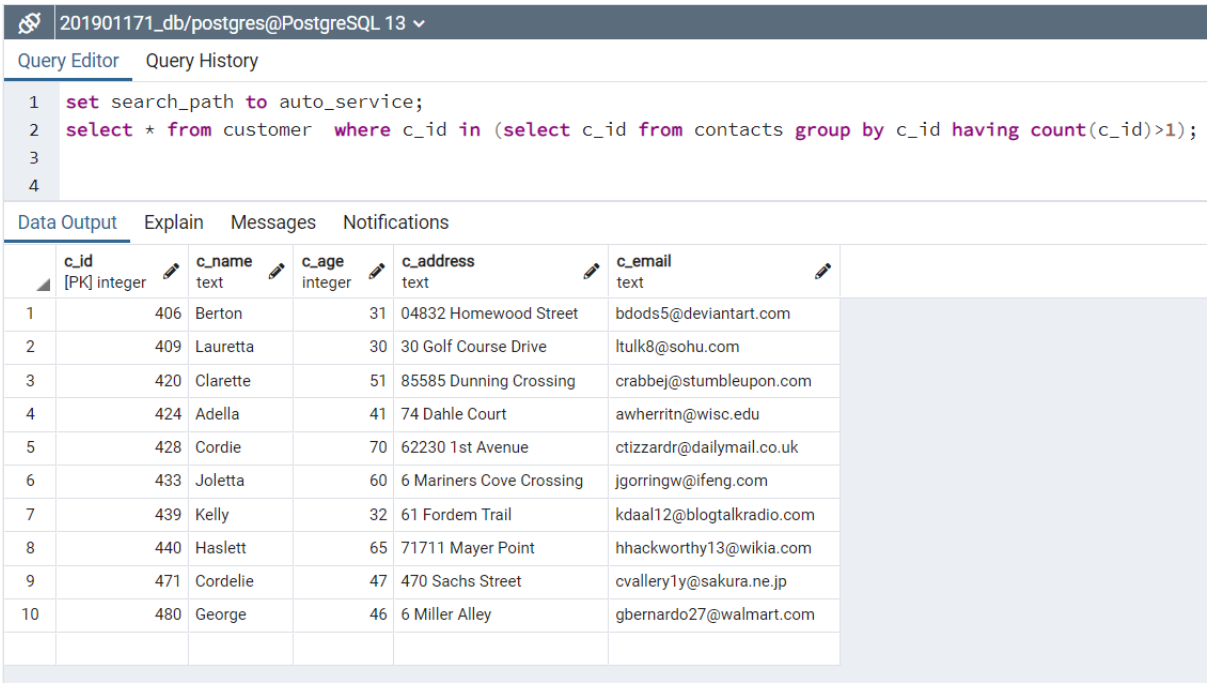
	e_id [PK] text	e_name text	e_contact text	service_location text	salary integer	designation text
6	T36	Irene	9772849491	Delhi	75281	Technician
7	T37	Kacy	6507887991	Mumbai	75402	Technician
8	T45	Norah	6970067956	Mumbai	82417	Technician
9	A1	Birdie	9270197682	Mumbai	83152	Agent
10	A2	Reginald	9127585000	Mumbai	83865	Agent
11	A3	Paulie	8747375191	Delhi	85021	Agent
12	A4	Kleon	6197702503	Delhi	86126	Agent
13	A35	Broddy	8709058851	Delhi	147582	Agent
14	MA1	Andromache	9576889178	Mumbai	150626	Manager
15	MA2	Vachel	7176313981	Mumbai	151441	Manager
16	MA3	Ibbie	9717538575	Mumbai	152897	Manager
17	MA4	Courtney	8189661559	Delhi	153778	Manager
18	MA5	Puff	9994869633	Delhi	155150	Manager

Number of tuples = 18

32. List the customers having more than one contact number.

- **select * from customer**

  where c_id in (select c_id from contacts group by c_id having count(c_id)>1);



201901171_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_service;
2  select * from customer  where c_id in (select c_id from contacts group by c_id having count(c_id)>1);
3
4
```

Data Output | Explain | Messages | Notifications

	c_id [PK] integer	c_name text	c_age integer	c_address text	c_email text
1	406	Berton	31	04832 Homewood Street	bdods5@deviantart.com
2	409	Lauretta	30	30 Golf Course Drive	ltulk8@sohu.com
3	420	Clarette	51	85585 Dunning Crossing	crabbej@stumbleupon.com
4	424	Adella	41	74 Dahle Court	awherritn@wisc.edu
5	428	Cordie	70	62230 1st Avenue	ctizzardr@dailymail.co.uk
6	433	Joletta	60	6 Mariners Cove Crossing	jgorringw@ifeng.com
7	439	Kelly	32	61 Fordem Trail	kdaal12@blogtalkradio.com
8	440	Haslett	65	71711 Mayer Point	hhackworthy13@wikia.com
9	471	Cordelie	47	470 Sachs Street	cvallery1y@sakura.ne.jp
10	480	George	46	6 Miller Alley	gbernardo27@walmart.com

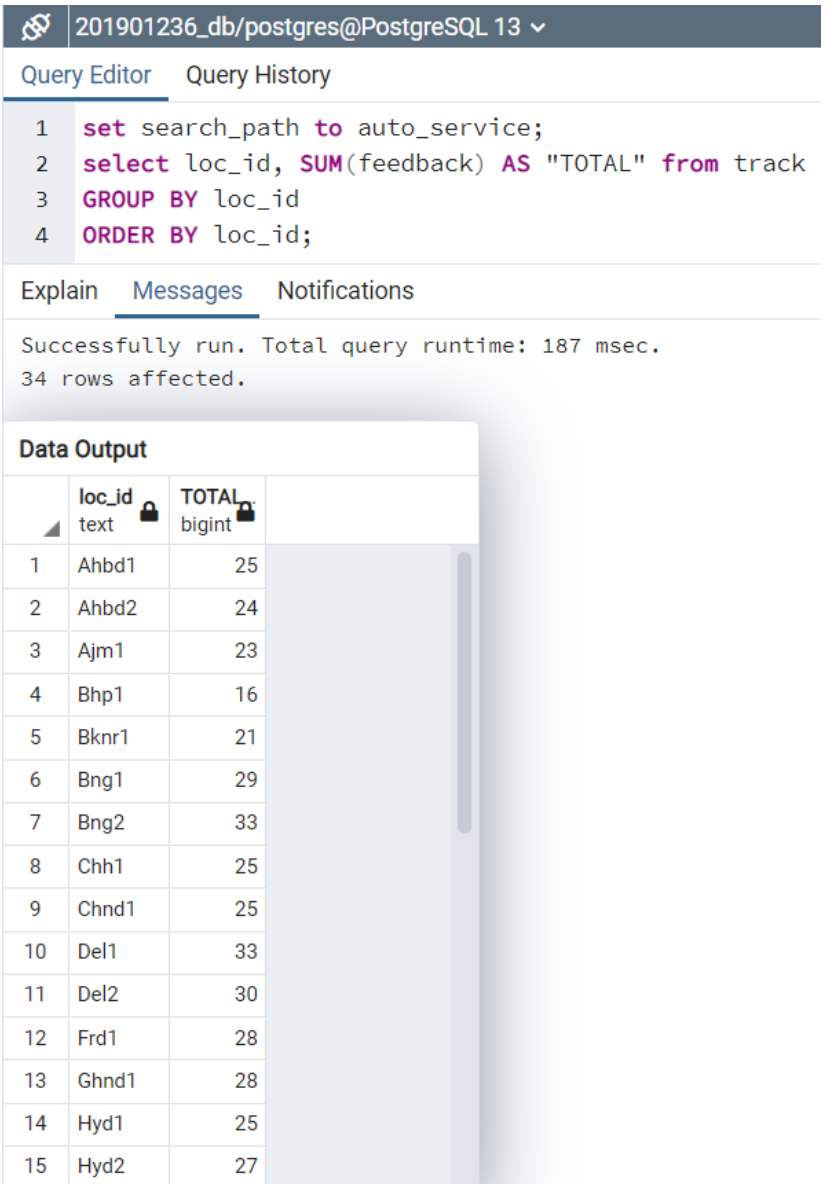
Number of tuples = 10

33. Calculate the individual centre's total feedback points across the nation.

- **select loc_id, SUM(feedback) AS "TOTAL" from track**

  GROUP BY loc_id

  ORDER BY loc_id;



	loc_id text	TOTAL bigint
1	Ahbd1	25
2	Ahbd2	24
3	Ajm1	23
4	Bhp1	16
5	Bknr1	21
6	Bng1	29
7	Bng2	33
8	Chh1	25
9	Chnd1	25
10	Del1	33
11	Del2	30
12	Frd1	28
13	Ghnd1	28
14	Hyd1	25
15	Hyd2	27

Number of tuples = 15

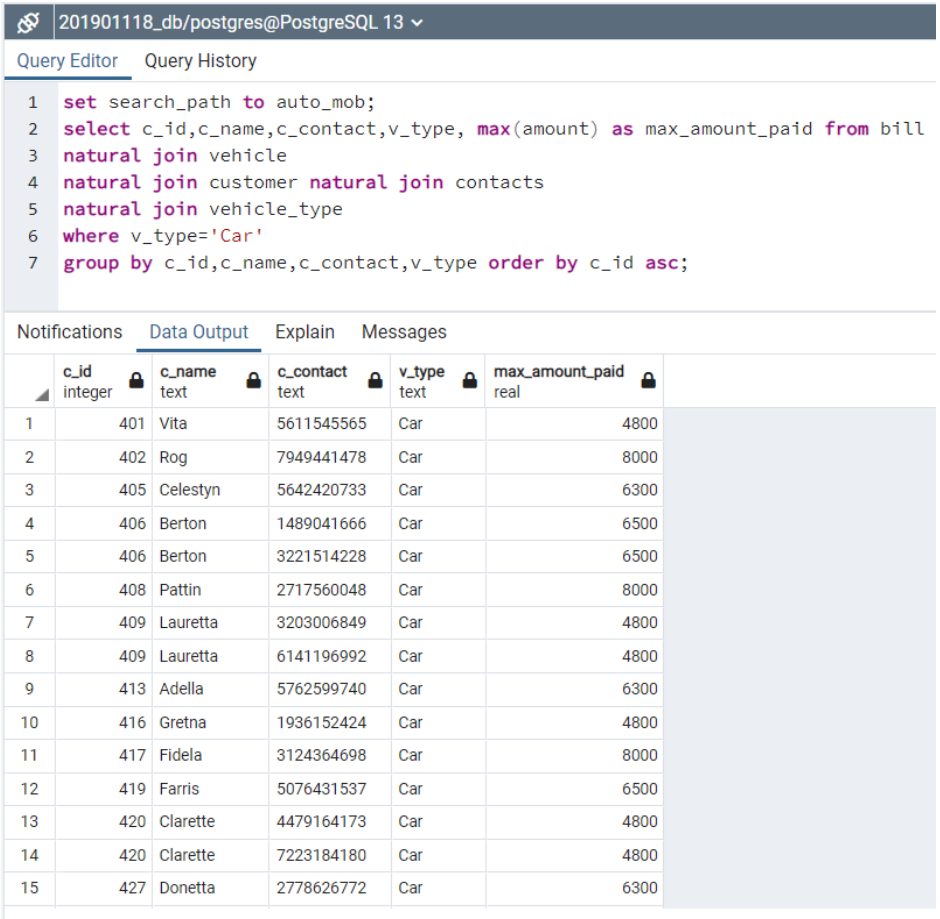
34. **Find all the customers with their contact details who have serviced their 'Car' and have done maximum payment from their multiple services.**

- **select c_id,c_name,c_contact,v_type, max(amount) as max_amount_paid**

  from bill NATURAL JOIN vehicle NATURAL JOIN customer NATURAL JOIN contacts NATURAL JOIN vehicle_type where v_type='Car'

  group by c_id,c_name,c_contact,v_type

  order by c_id asc;



201901118_db/postgres@PostgreSQL 13

Query Editor | Query History

```
1  set search_path to auto_mob;
2  select c_id,c_name,c_contact,v_type, max(amount) as max_amount_paid from bill
3  natural join vehicle
4  natural join customer natural join contacts
5  natural join vehicle_type
6  where v_type='Car'
7  group by c_id,c_name,c_contact,v_type order by c_id asc;
```

Notifications | Data Output | Explain | Messages

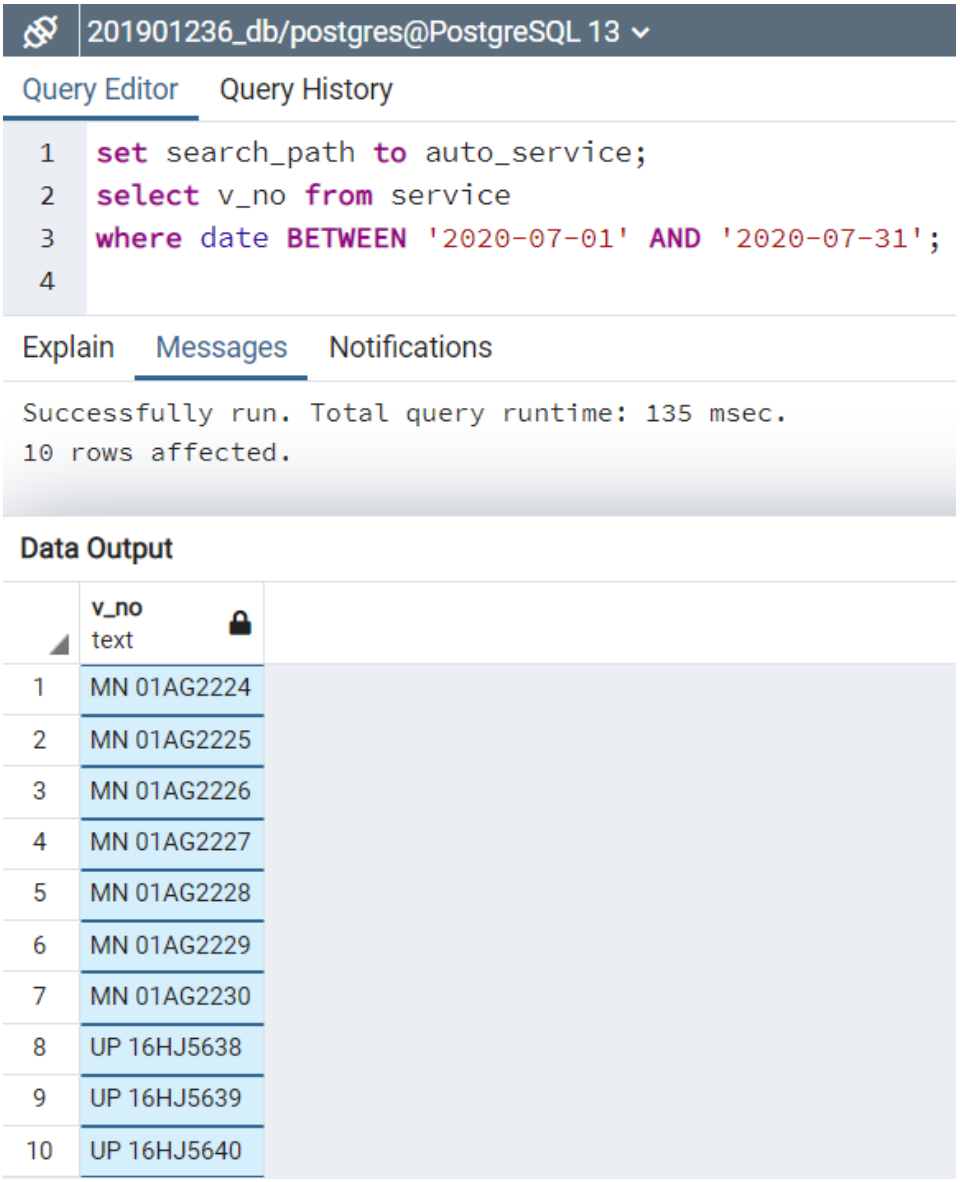
	c_id integer	c_name text	c_contact text	v_type text	max_amount_paid real
1	401	Vita	5611545565	Car	4800
2	402	Rog	7949441478	Car	8000
3	405	Celestyn	5642420733	Car	6300
4	406	Berton	1489041666	Car	6500
5	406	Berton	3221514228	Car	6500
6	408	Pattin	2717560048	Car	8000
7	409	Lauretta	3203006849	Car	4800
8	409	Lauretta	6141196992	Car	4800
9	413	Adella	5762599740	Car	6300
10	416	Gretna	1936152424	Car	4800
11	417	Fidela	3124364698	Car	8000
12	419	Farris	5076431537	Car	6500
13	420	Clarette	4479164173	Car	4800
14	420	Clarette	7223184180	Car	4800
15	427	Donetta	2778626772	Car	6300

Number of tuples = 15

35. Show the vehicle numbers serviced in the July month of 2020.

- **select v_no from service**

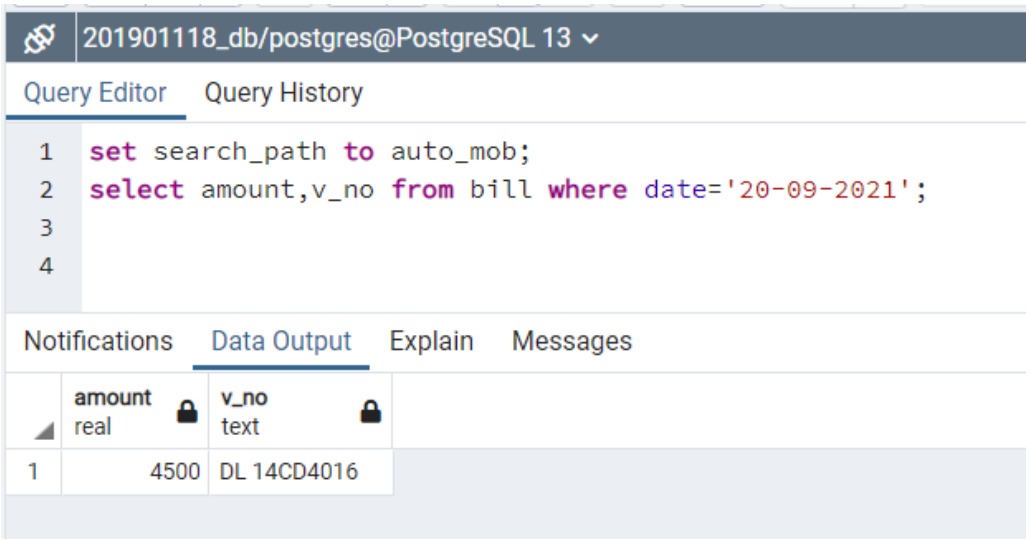
  where date BETWEEN '2020-07-01' AND '2020-07-31';



Number of tuples = 10

36. Find the bill amount paid on 20th September 2021 and by which vehicle number.

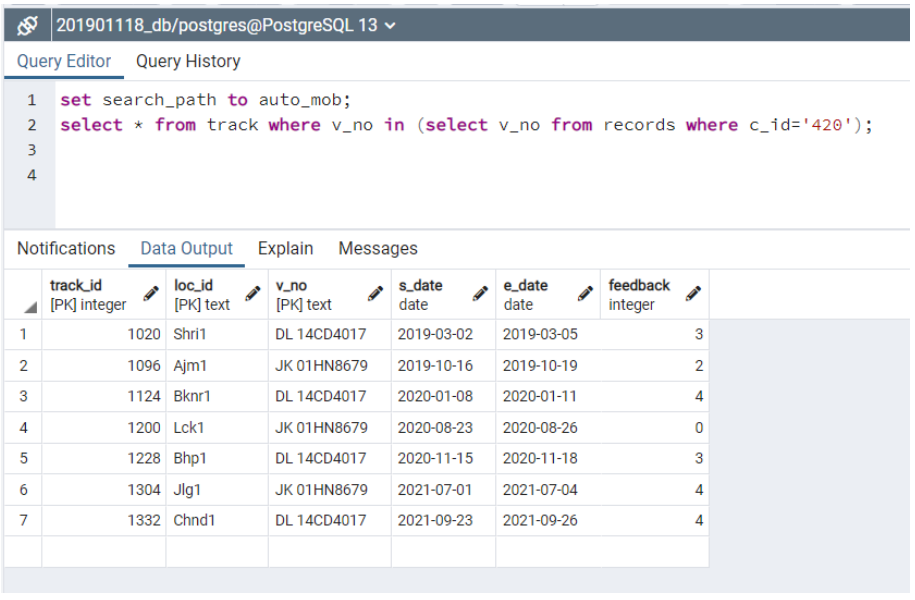
- **select amount,v_no from bill where date='20-09-2021';**



Number of tuples = 1

37. List the past service history of the customer with ID = 420.

- **select * from track where v_no IN (select v_no from records where c_id='420');**



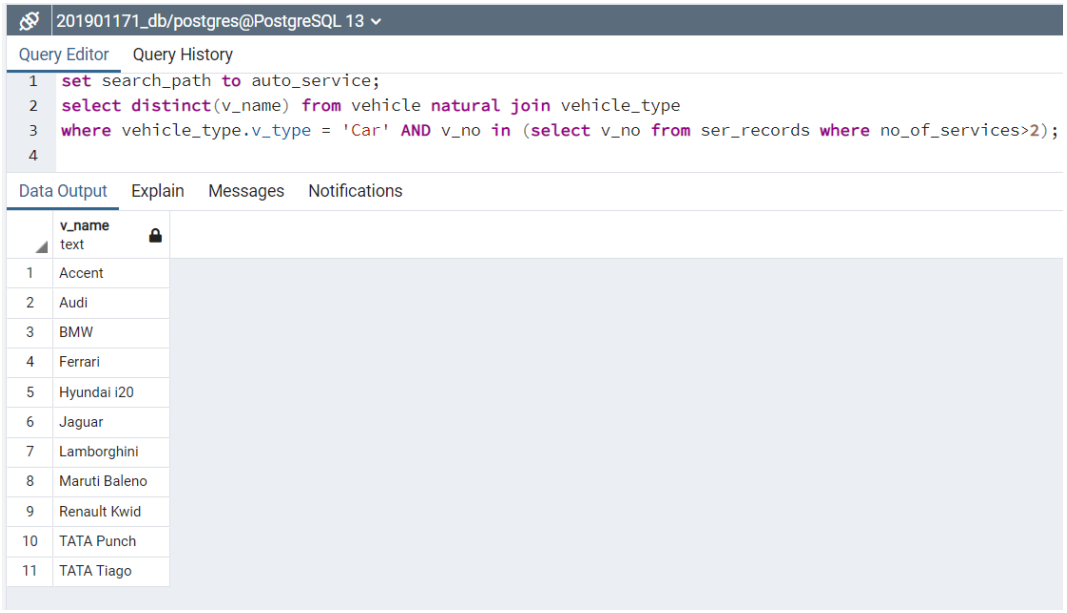
Number of tuples = 7

38. List the names of the cars serviced more than 2 times so far.

- **select distinct (v_name) from vehicle NATURAL JOIN vehicle_type**

  where vehcle_type.v_type = 'Car' AND v_no in

  (select v_no from ser_records where no_of_services>2);



Number of tuples = 11

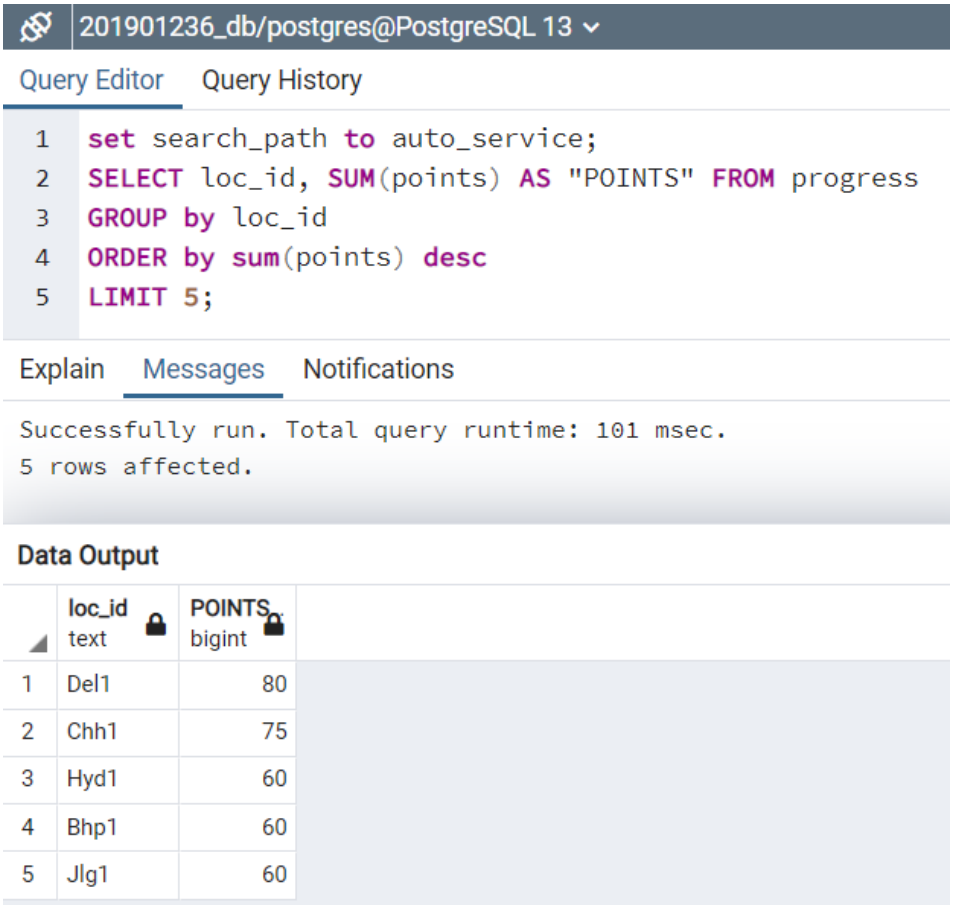
39. List top 5 centres across the nation.

- **SELECT loc_id, SUM(points) AS "POINTS" FROM progress**

  GROUP by loc_id

  ORDER by sum(points) desc

  LIMIT 5;



	loc_id text	POINTS bigint
1	Del1	80
2	Chh1	75
3	Hyd1	60
4	Bhp1	60
5	Jlg1	60

Number of tuples = 5

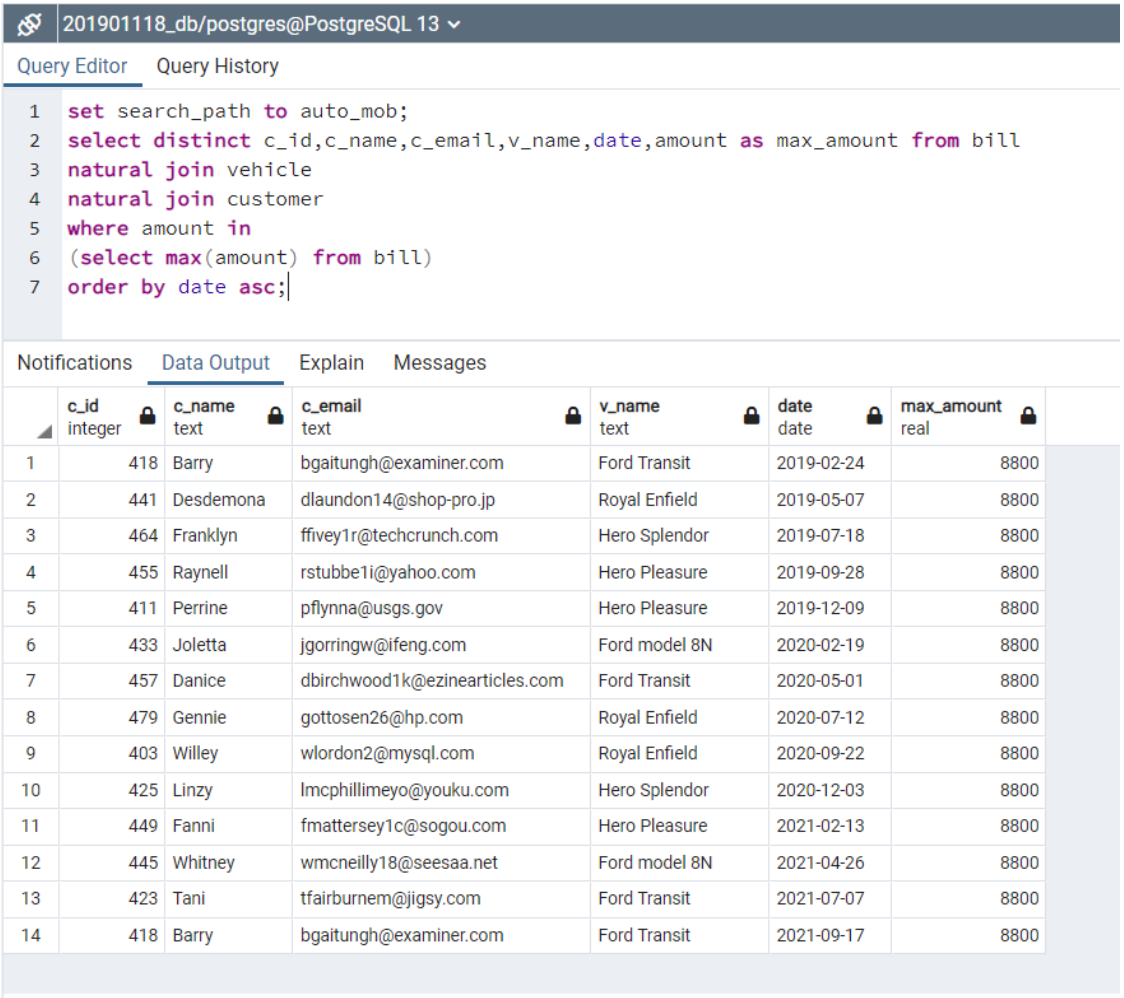
40. **List all the customer details with the respective date who have paid the maximum amount of bill paid till now.**

- **select distinct c_id,c_name,c_email,v_name,date, amount as max_amount from bill**

  NATURAL JOIN vehicle NATURAL JOIN customer

  where amount in (select max(amount) from bill)

  order by date asc;



	c_id integer	c_name text	c_email text	v_name text	date date	max_amount real
1	418	Barry	bgaitungh@examiner.com	Ford Transit	2019-02-24	8800
2	441	Desdemona	dlaundon14@shop-pro.jp	Royal Enfield	2019-05-07	8800
3	464	Franklyn	ffivey1r@techcrunch.com	Hero Splendor	2019-07-18	8800
4	455	Raynell	rstubbe1i@yahoo.com	Hero Pleasure	2019-09-28	8800
5	411	Perrine	pflynna@usgs.gov	Hero Pleasure	2019-12-09	8800
6	433	Joletta	jgorringw@ifeng.com	Ford model 8N	2020-02-19	8800
7	457	Danice	dbirchwood1k@ezinearticles.com	Ford Transit	2020-05-01	8800
8	479	Gennie	gottosen26@hp.com	Royal Enfield	2020-07-12	8800
9	403	Willey	wlordon2@mysql.com	Royal Enfield	2020-09-22	8800
10	425	Linzy	lmcphillimeyo@youku.com	Hero Splendor	2020-12-03	8800
11	449	Fanni	fmattersey1c@sogou.com	Hero Pleasure	2021-02-13	8800
12	445	Whitney	wmcneilly18@seesaa.net	Ford model 8N	2021-04-26	8800
13	423	Tani	tfairburnem@jigsy.com	Ford Transit	2021-07-07	8800
14	418	Barry	bgaitungh@examiner.com	Ford Transit	2021-09-17	8800

Number of tuples = 14

4. Triggers and Functions (Stored Procedure)

4.1 Functions

1. **Create a function that would take the location center and the vehicle type as an input and would tell whether the slot is available or not for the required specifications.**

```
CREATE OR REPLACE FUNCTION "auto_mob"."check_slot"(location_id text, veh_type text)
 RETURNS text
 LANGUAGE 'plpgsql'
AS $BODY$
declare
check_yn text;
BEGIN
select slot.availability into check_yn from slot where slot.loc_id=location_id and slot.v_type=veh_type;
if(check_yn ='YES') then
return 'Slot is available';
else return 'Slot is not available';
end if;
END
$BODY$;
```

```
SELECT "auto_mob"."check_slot"('Mum2','Scooter');
```

```
17  SELECT "auto_mob"."check_slot"('Mum2','Scooter');
18
```

Notifications | Data Output | Explain | Messages

	check_slot text
1	Slot is not available

```
SELECT "auto_mob"."check_slot"('Del1','Bus');
```

```
16
17  SELECT "auto_mob"."check_slot"('Del1','Bus');
18
```

Notifications | Data Output | Explain | Messages

	check_slot text
1	Slot is available

- After calling the function check_slot(), we can see that for scooters the slot is not available at Mumbai center 2 but the slot is available for Bus at Delhi center 2.
- Hence this would help the customers to book the slots if it's available according to their inputs.

2. **Create a function that takes the location ID, vehicle number, service date , end date and feedback as input and adds this as the last entry in the track table with immediate next track ID.**

```
CREATE OR REPLACE FUNCTION "auto_service"."track_insert"(lID text, vno text, sd date, ed date, feed integer)

RETURNS TABLE (track_id integer, loc_id text, v_no text, s_date date, e_date date, feedback integer)

LANGUAGE 'plpgsql'

AS $BODY$

declare

trackId integer;

BEGIN

select max(track.track_id) into trackId from track;

trackId = trackId + 1;

insert into track (track_id, loc_id, v_no, s_date, e_date, feedback)

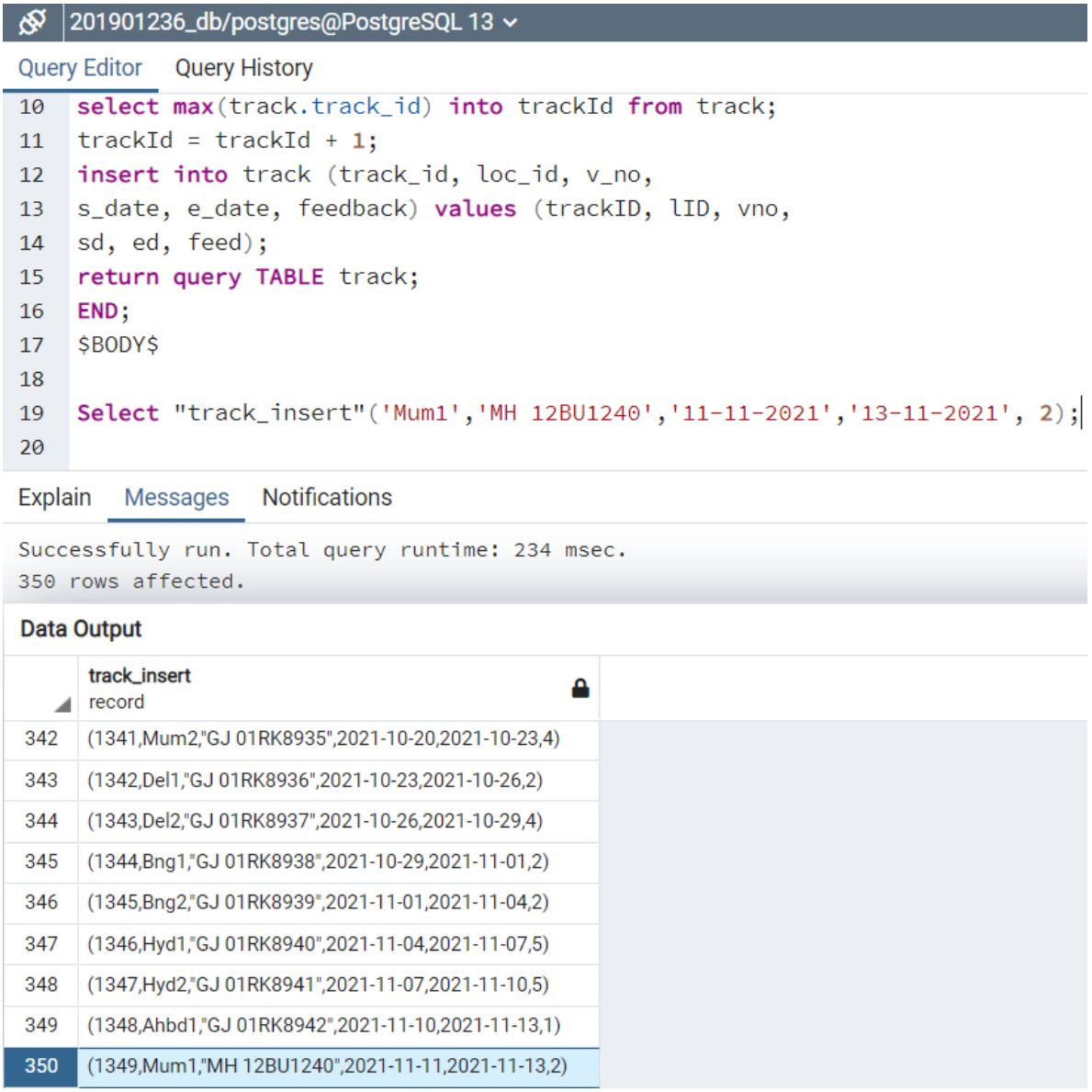
values (trackID, lID, vno, sd, ed, feed);

return query TABLE track;

END;

$BODY$
```

```
select "track_insert"('Mum1','MH 12BU1240','11-11-2021','13-11-2021', 2);
```



- The function updates the track ID and thus no need to check the entire track table repeatedly while inserting a new record.

4.2 Triggers function

1. **Create a trigger function that automatically updates the stock and points in the stock and progress table respectively whenever any vehicle comes for service. This trigger also ensures that the stock never decreases below 0 and gets updated when required.**

```
CREATE OR REPLACE FUNCTION "auto_service".update_stock_points()

RETURNS trigger

LANGUAGE 'plpgsql'

AS $BODY$

declare

lid text;

vno text;

vn text;

vt text;

co integer;

oil integer;

water integer;

feed integer;

y integer;


BEGIN

select track.loc_id into lid from track where track.loc_id = new.loc_id;

select track.v_no into vno from track where track.v_no = new.v_no;

select vehicle.v_name into vn from vehicle where vehicle.v_no = vno;

select vehicle_type.v_type into vt from vehicle_type where vehicle_type.v_name = vn;
```

```
select stock.coolant into co from stock where stock.loc_id=lid;

select stock.engine_oil into oil from stock where stock.loc_id=lid;

select stock.cleaning_water into water from stock where stock.loc_id=lid;

select extract(year from current_date) into y from track where track.loc_id = lid;

select track.feedback into feed from track where track.loc_id = lid;


update progress

 set points = points + feed

 where loc_id = lid AND c_year=y;


if(co<=20) then

 RAISE NOTICE 'Coolant was insufficient. Hence its stock was updated';

 update stock

 set coolant  = coolant +150

where loc_id = lid;

end if;


if(oil<=20) then

 RAISE NOTICE 'Engine Oil was insufficient. Hence its stock was updated';

 update stock

 set engine_oil = engine_oil+150

where loc_id = lid;

end if;


if(water<=20) then

 RAISE NOTICE 'Cleaning water was insufficient. Hence its stock was updated';
```

```
 update stock
 set cleaning_water = cleaning_water+150
where loc_id = lid;
end if;
 if(vt = 'Car') then
 update stock
 set coolant = coolant-5,
 engine_oil = engine_oil - 10,
 cleaning_water = cleaning_water - 20
 where loc_id = lid;

elsif(vt = 'Bike') then
 update stock
 set engine_oil = engine_oil - 5,
 cleaning_water = cleaning_water - 10
 where loc_id = lid;

elsif(vt = 'Scooter') then
 update stock
 set engine_oil = engine_oil - 5,
 cleaning_water = cleaning_water - 10
 where loc_id = lid;

elsif(vt = 'Truck') then
 update stock
  set coolant = coolant-3,
```

```
 engine_oil = engine_oil - 10,
 cleaning_water = cleaning_water - 20
 where loc_id = lid;

elsif(vt = 'Tractor') then
 update stock
 set coolant = coolant-2,
 engine_oil = engine_oil - 10,
 cleaning_water = cleaning_water - 10
 where loc_id = lid;

elsif(vt = 'Bus') then
 update stock
 set coolant = coolant-5,
    engine_oil = engine_oil - 10,
    cleaning_water = cleaning_water - 20
 where loc_id = lid;

elsif(vt = 'Taxi') then
 update stock
 set coolant = coolant-2,
    engine_oil = engine_oil - 5,
    cleaning_water = cleaning_water - 5
 where loc_id = lid;

else                    /*  FOR VAN */
```

```
 update stock
 set coolant = coolant-2,
     engine_oil = engine_oil - 5,
     cleaning_water = cleaning_water - 10
 where loc_id = lid;


end if;
return new;
END
$BODY$;


CREATE TRIGGER "Trigger_insert"
AFTER INSERT
ON "auto_service".track
FOR EACH ROW
EXECUTE PROCEDURE "auto_service".update_stock_points();
```

```
select "track_insert"('Del2', 'DL 14CD4010', '13-11-2021', '16-11-2021', 4);

select * from stock where loc_id='Del2';
```

```
130  select * from stock where loc_id='Del2';
```

Query Editor Query History Explain Notifications

Messages

```
Successfully run. Total query runtime: 50 msec.
1 rows affected.
```

Data Output

	loc_id [PK] text	coolant integer	engine_oil integer	cleaning_water integer
1	Del2	88	258	150

BEFORE

- These are the stock details of Delhi 2 centre before the vehicle 'DL 14CD4010 came for service.
- Its record is inserted with the help of the “track_insert” function as seen in the last question.

```
85  select "track_insert"('Del2', 'DL 14CD4010', '13-11-2021', '16-11-2021', 4);
86
```

Explain Messages Notifications

```
Successfully run. Total query runtime: 157 msec.
352 rows affected.
```

Data Output

	track_insert record
345	(1344,Bng1,"GJ 01RK8938",2021-10-29,2021-11-01,2)
346	(1345,Bng2,"GJ 01RK8939",2021-11-01,2021-11-04,2)
347	(1346,Hyd1,"GJ 01RK8940",2021-11-04,2021-11-07,5)
348	(1347,Hyd2,"GJ 01RK8941",2021-11-07,2021-11-10,5)
349	(1348,Ahbd1,"GJ 01RK8942",2021-11-10,2021-11-13,1)
350	(1349,Mum1,"MH 12BU1240",2021-11-11,2021-11-13,2)
351	(1350,Mum2,"MH 12BU1234",2021-11-12,2021-11-14,3)
352	(1351,Del2,"DL 14CD4010",2021-11-13,2021-11-16,4)

- Whenever any insertion takes place in the track table, the trigger function update_stock_points() starts doing its work. It searches the vehicle type from the vehicle number and according to that updates/decreases the stock in the stock table.
- Moreover, this also adds the feedback given by the vehicle owner in the progress table of that corresponding location centre simultaneously.

```
130  select * from stock where loc_id='Del2';
```

Query Editor | Query History | Explain | Notifications

Messages

```
Successfully run. Total query runtime: 53 msec.
1 rows affected.
```

Data Output

	loc_id [PK] text	coolant integer	engine_oil integer	cleaning_water integer
1	Del2	85	248	130

AFTER

- As we can see from the BEFORE table the coolant, engine oil and cleaning water was 88, 258 and 150 respectively and now in the AFTER table the stock had decreased to 85, 248 and 130. This is because the vehicle number DL 14CD4010 was a 'Truck' and hence according to the trigger function the stock decreased by 3,10,20 respectively.

```
select "track_insert"('Bng2', 'WB 06F5644', '14-11-2021', '16-11-2021', 3);

select * from progress where loc_id='Bng2';
```

```
143  select * from progress where loc_id='Bng2';
144
```

Data Output | Explain | Messages | Notifications

	loc_id [PK] text	c_year [PK] integer	points integer
1	Bng2	2020	5
2	Bng2	2019	33
3	Bng2	2021	10

BEFORE

- Here a new vehicle WB 06F5644 has come for service at the location Bengal center 2 and has given a feedback point of 3 out of 5.
- The points for Bengal centre 'Bng2' in the progress table was 10 for the current year 2021 and now the trigger function would update this to 10 + 3 = 13. This can be seen in the AFTER table.
- Simultaneously its stock would also be updated similarly to previous examples.

```
143  select * from progress where loc_id='Bng2';
144
```

Data Output | Explain | Messages | Notifications

	loc_id [PK] text	c_year [PK] integer	points integer
1	Bng2	2020	5
2	Bng2	2019	33
3	Bng2	2021	13

AFTER

2. **Create a trigger function that would update the number of services in the ser_records table as and when a vehicle comes for service and its record gets entered in the service table.**

```
CREATE OR REPLACE FUNCTION "auto_mob".update_no_of_services()

RETURNS trigger

LANGUAGE 'plpgsql'

AS $BODY$

declare

veh_no text;

BEGIN

select service.v_no into veh_no from service where service.v_no = new.v_no;

/*select ser_records.no_of_services into service_count from ser_records where ser_records.v_no = veh_no;*/

update ser_records

 set no_of_services = no_of_services+1

 where v_no = veh_no;

return new;

END

$BODY$;


CREATE TRIGGER "Trigger_update_services"

AFTER INSERT

ON "auto_mob".service

FOR EACH ROW

EXECUTE PROCEDURE "auto_mob".update_no_of_services();
```

```
insert into service values ('AS 11NC8470','16-11-2021','M15','New tires(c)');  /*ngp2*/

select "track_insert"('Ngp2', 'AS 11NC8470', '16-11-2021', '18-11-2021', 2);

select * from ser_records where v_no='AS 11NC8470';
```

```
25  select * from ser_records where v_no='AS 11NC8470';
26
```

Notifications | Data Output | Explain | Messages

	v_no [PK] text	no_of_services integer
1	AS 11NC8470	3

BEFORE

- From the BEFORE table we can comprehend that the vehicle AS 11NC8470 has come for service three times so far. Therefore this would be its 4th time.
- Hence now when this car is coming for service on 16-11-2021, its record would be inserted into the service table and on insertion the trigger **update_no_of_services()** will automatically update the no_of_services in the ser_records tabel to 3+1 = 4 for that particular vehicle.

```
25  select * from ser_records where v_no='AS 11NC8470';
```

Notifications | Data Output | Explain | Messages

	v_no [PK] text	no_of_services integer
1	AS 11NC8470	4

AFTER

- From the AFTER tabel we can clearly see that the number of services for the vehicle AS 11NC8470 has been updated to 4.

3. **Create a trigger function that would generate the bill (add amount) into the bill tabel according to the service type whenever a vehicle comes for service.**

```
CREATE OR REPLACE FUNCTION "auto_mob".bill_generate()

RETURNS trigger

LANGUAGE 'plpgsql'

AS $BODY$

declare

veh_no text;

bill_date date;

payment real;

ser_type text;

BEGIN

select service.v_no into veh_no from service where service.v_no = new.v_no;

select service.date into bill_date from service where service.date=new.date;

select service.s_type into ser_type from service where service.s_type=new.s_type;

select ser_charges.charges into payment from ser_charges where ser_charges.s_type=ser_type;

/*select ser_records.no_of_services into service_count from ser_records where ser_records.v_no = veh_no;*/

insert into bill values(bill_date,veh_no,payment);

return new;

END

$BODY$;


CREATE TRIGGER "Trigger_generate_bill"

AFTER INSERT

ON "auto_mob".service
```

```
FOR EACH ROW

EXECUTE PROCEDURE "auto_mob".bill_generate();
```

insert into service values ('KA 22BN5905','19-11-2021','M15','Scheduled maintenance(h)');

select * from bill;

```
29  select * from bill;
```

Notifications | Data Output | Explain | Messages

	date [PK] date	v_no [PK] text	amount real
342	2021-10-20	GJ 01RK8935	3000
343	2021-10-23	GJ 01RK8936	2300
344	2021-10-26	GJ 01RK8937	5000
345	2021-10-29	GJ 01RK8938	3100
346	2021-11-01	GJ 01RK8939	2000
347	2021-11-04	GJ 01RK8940	2500
348	2021-11-07	GJ 01RK8941	6500
349	2021-11-10	GJ 01RK8942	6300

BEFORE

```
28  insert into service values ('KA 22BN5905','19-11-2021','M15','Scheduled maintenance(h)');
29  select * from bill;
```

Notifications | Data Output | Explain | Messages

	date [PK] date	v_no [PK] text	amount real
343	2021-10-23	GJ 01RK8936	2300
344	2021-10-26	GJ 01RK8937	5000
345	2021-10-29	GJ 01RK8938	3100
346	2021-11-01	GJ 01RK8939	2000
347	2021-11-04	GJ 01RK8940	2500
348	2021-11-07	GJ 01RK8941	6500
349	2021-11-10	GJ 01RK8942	6300
350	2021-11-19	KA 22BN5905	5000

AFTER

- The car KA 22BN5905 comes for Scheduled maintenance(h) and it's record gets entered into the service table.
- When this insertion takes place in the service tabel the trigger function **bill_generate()** will fetch the service charges of that particular service type from the ser_charges table and would add that amount in the bill table for that vehicle.
- We can see in the AFTER tabel a new record gets inserted by the trigger. Since the cost for Scheduled maintenance(h) was 5000 INR hence that amount gets inserted in the bill table for that vehicle number.

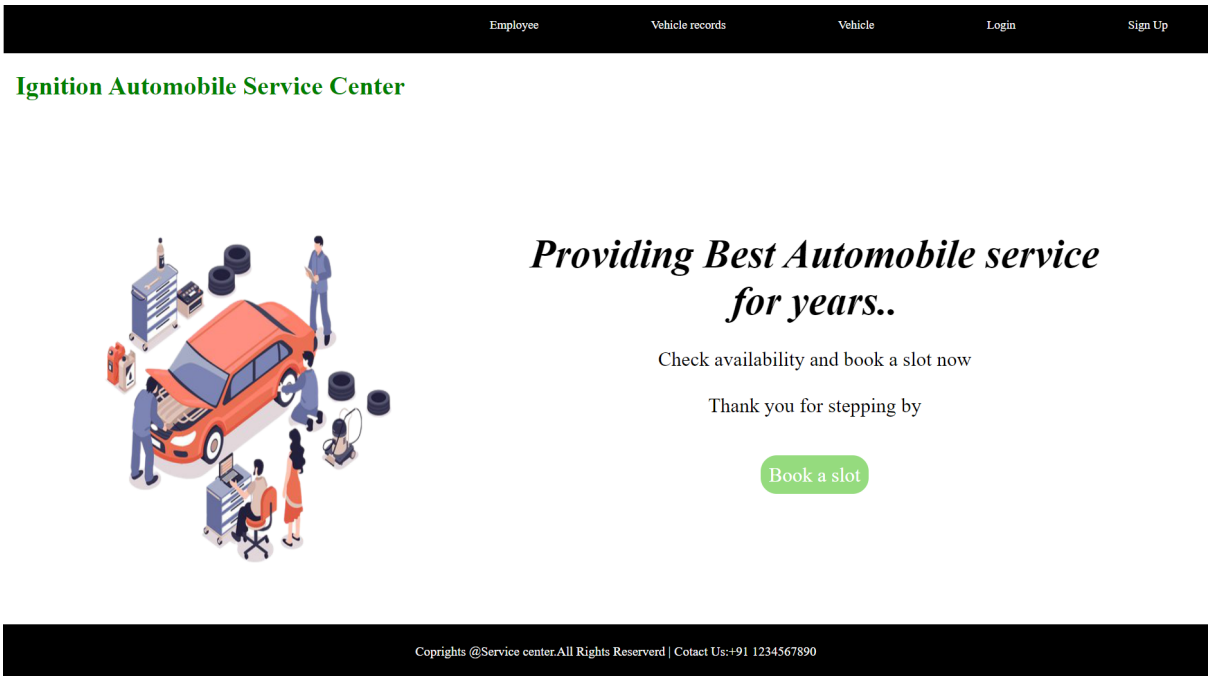
Section-7
Project Code with output screenshots

1. Front-End Development

- We have used HTML, CSS and Python to create this web application
- We have implemented the insert, delete and edit(update) functions in our web application.

Here is the GitHub Repositary Link for the Project: https://github.com/Kp3005/DBMS_Project.git

Front Page:



Explanation:

The above picture is the home screen of our automobile service centre where customers can get all the online benefits from their couch. Any customer who wants to get their vehicle service done would first book a slot online via the green button shown.
After clicking, the customer would be asked the service centre where they want to do their vehicle service and the type of vehicle for which they want service (e.g. car, bus, truck).

In the navigation bar, the ***Employee*** option retrieves the employee details from the database. Also we can update employee details and delete the records if needed.

The ***Vehicle*** would allow the customer to register their vehicle details and the form would insert this into the database.

The ***Vehicle records*** would ask for the c_id and would give their vehicle details from the database.

The **login** would help in verifying the customer and would then allow them to proceed further. If the customer is new, then an error message would pop up and would ask for registering/ sign up first. The ***sign up*** page will insert the new customer in the database table.

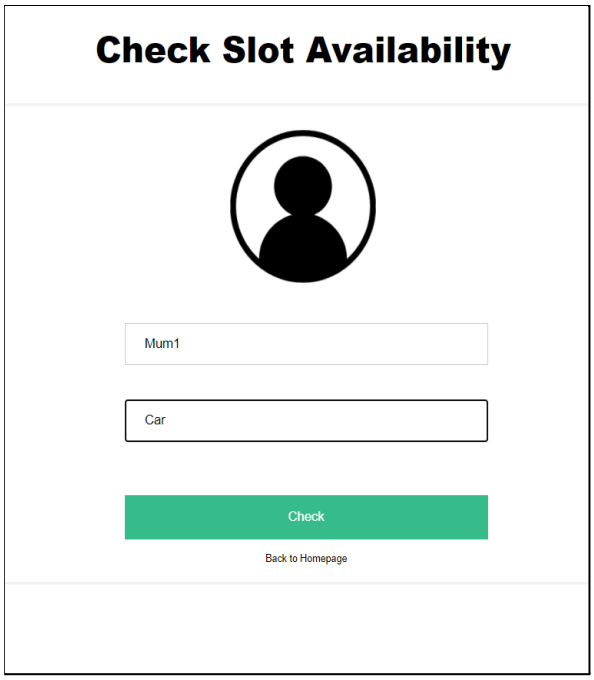
1. **connection.php**

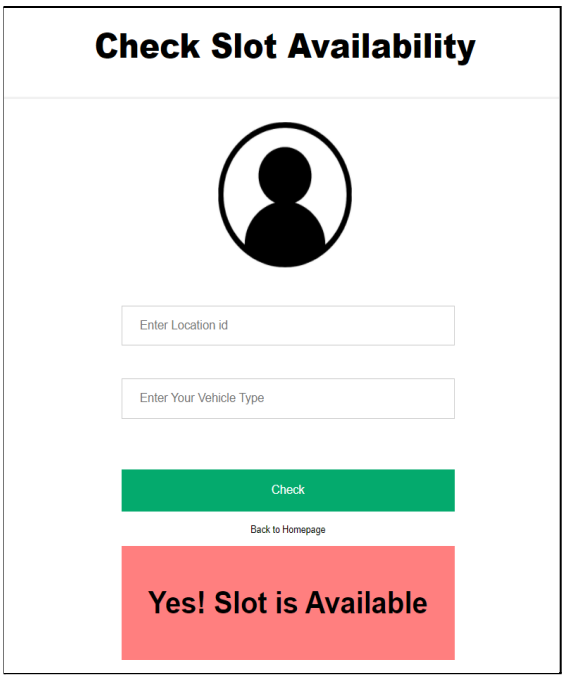
```
<?php
$conn = mysqli_connect('localhost','root','','service_center');
if($conn === false){
   die("ERROR: Could not connect. " . mysqli_connect_error());
   }
?>
```

2. **slot.php**

```
<?php
require_once('connection.php');
if(isset($_POST['Check']))
{
   $loc_id = mysqli_real_escape_string($conn, $_POST['loc_id']);
   $v_type = mysqli_real_escape_string($conn, $_POST['v_type']);
   $result = mysqli_query($conn, "SELECT * from slot where loc_id ='$loc_id' and v_type = '$v_type' ");
   $row = mysqli_fetch_assoc($result);

   if($row['availability'] == 'YES') echo '<div class="alert alert-primary" role="alert" style="margin-top:10px;
padding:20px;background-color:rgba(255,0,0,0.5);letterspacing:2;">
      <h1><center>Yes! Slot is Available</center></h1> </div>';
    else echo '<div class="alert alert-primary" role="alert" style="margin-top:10px;
padding:20px;background-color:rgba(255,0,0,0.5);letterspacing:2; ">
      <h1><center>Opss! Slot is Not Available</center></h1> </div>';
}
?>
```





Here the customer is checking the slot for mumbai center 1 for the vehicle type car. The query executes and checks the availability status in the database table whether it's yes/no. And hence the message says “Yes! Slot is available”.

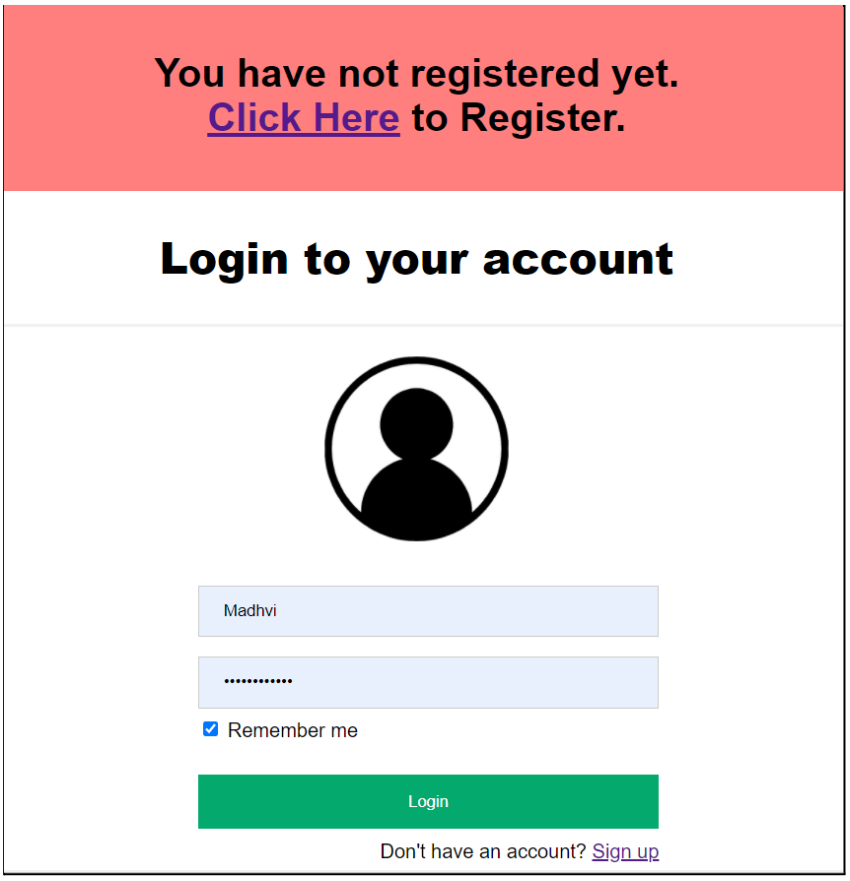
3. login.php

```php
<?php
session_start();
require('connection.php');
if(isset($_POST['login']) ){
   $c_name=$_POST['c_name'];
   $c_id=$_POST['c_id'];
   $sql="select * from customer where 'c_name' = '{$c_name}' and c_id='{$c_id}' ";
   $res=mysqli_query($conn,$sql);
   $row=mysqli_fetch_assoc($res);
   $count=mysqli_num_rows($res);

   if($count==1){
      $_SESSION['USER_LOGIN']='yes';
      $_SESSION['c_name']=$row['c_name'];
      $_SESSION['c_id']=$row['c_id'];
```

```
        header('location:index.php');
        }
    else{
        echo '<div class="alert alert-primary" role="alert" style="margin-top:10px;
padding:20px;background-color:rgba(255,0,0,0.5);letterspacing:2; ">
        <h1><center>You have not registered yet.<br><a href="signup.php">Click Here</a> to
Register.</center></h1>
</div>';
    }
}
?>
```

4. **logout.php**

```
<?php
session_start();
unset($_SESSION['USER_LOGIN']);
unset($_SESSION['c_name']);;
unset($_SESSION['c_id']);
header('location:index.php');
?>
```



Since there was no customer with the name *"Madhvi"*, there is an error message at the top that asks to register first.

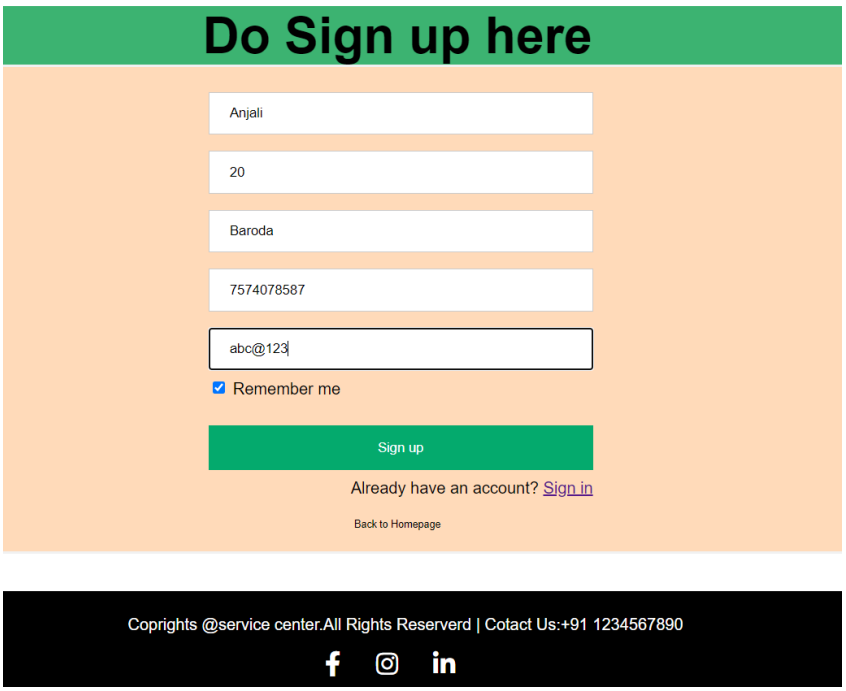
There is an option below "Sign up" where one can register themself first. The new customer would be asked about his/her personal details and that would be inserted in the customer table and a cust_id would be assigned to them by auto increment process.

5. **sign up.php**

```
<?php
require_once('connection.php');

if (isset($_POST['c_name']) && !empty($_POST['c_name']) && isset($_POST['c_age']) &&
!empty($_POST['c_age']) && isset($_POST['c_address']) && !empty($_POST['c_address']) &&
isset($_POST['c_contact']) && !empty($_POST['c_contact']) &&  isset($_POST['c_email']) &&
!empty($_POST['c_email'])){
   $c_name = mysqli_real_escape_string($conn, $_POST['c_name']);
   $c_age = mysqli_real_escape_string($conn, $_POST['c_age']);
   $c_address = mysqli_real_escape_string($conn, $_POST['c_address']);
   $c_email = mysqli_real_escape_string($conn, $_POST['c_email']);

   $insertquery = mysqli_query($conn, "insert into customer (`c_name`,`c_age`,`c_address`,`c_email`) "
            . "values ('{$c_name}','{$c_age}','{$c_address}', '{$c_email}')") or die(mysqli_error($conn));

   //Below Function will Give you last Inserted Record ID
   $lastinsertid = mysqli_insert_id($conn);

      if ($insertquery)
      {
         $response['flag'] = 1;
         //Messege for Succesfully Registration....
         echo '<div style="background-color:#52b350; padding:10px; font-size:26px;
weight:500;"><center>Registered Succesfully.</center></div>';
         echo "Your Customer id: " .$lastinsertid;
      }
      else
      {
         echo "Please Try Again Something Went  Wrong";
      }
}
?>
```

Before Inserting:

c_id	c_name	c_age	c_address	c_email
467	Adella	60	26341 Brentwood Junction	abaxstare1u@archive.org
468	Matt	42	665 Manley Way	mpalombi1v@cdc.gov
469	Ferdinand	42	0 West Avenue	ftschirasche1w@java.com
470	Kaitlyn	47	76 American Ash Park	ksultana1x@cbsnews.com
471	Cordelie	47	470 Sachs Street	cvallery1y@sakura.ne.jp
472	Grange	49	8023 Roth Avenue	gcashell1z@alexa.com
473	Barbee	40	2 Texas Parkway	bcreagh20@zimbio.com
474	Whitney	46	1 Fulton Crossing	wpinkstone21@unblog.fr
475	Etti	26	7 Mallard Point	equennell22@elegantthemes.com
476	Ardath	49	2161 Bultman Street	awoollacott23@friendfeed.com
477	Truman	34	3 Dunning Hill	tsandyford24@reuters.com
478	Karalee	40	52233 Graceland Lane	koda25@netvibes.com
479	Gennie	39	1 Fairfield Terrace	gottosen26@hp.com
480	George	46	6 Miller Alley	gbernardo27@walmart.com



This will insert “Anjali’s” details in the customer database since she is a new customer.

c_id	c_name	c_age	c_address	c_email
469	Ferdinand	42	0 West Avenue	ftschirasche1w@java.com
470	Kaitlyn	47	76 American Ash Park	ksultana1x@cbsnews.com
471	Cordelie	47	470 Sachs Street	cvallery1y@sakura.ne.jp
472	Grange	49	8023 Roth Avenue	gcashell1z@alexa.com
473	Barbee	40	2 Texas Parkway	bcreagh20@zimbio.com
474	Whitney	46	1 Fulton Crossing	wpinkstone21@unblog.fr
475	Etti	26	7 Mallard Point	equennell22@elegantthemes.com
476	Ardath	49	2161 Bultman Street	awoollacott23@friendfeed.com
477	Truman	34	3 Dunning Hill	tsandyford24@reuters.com
478	Karalee	40	52233 Graceland Lane	koda25@netvibes.com
479	Gennie	39	1 Fairfield Terrace	gottosen26@hp.com
480	George	46	6 Miller Alley	gbernardo27@walmart.com
481	Anjali	20	Baroda	abc@123

New record of Anjali inserted on clicking “Sign up” button.

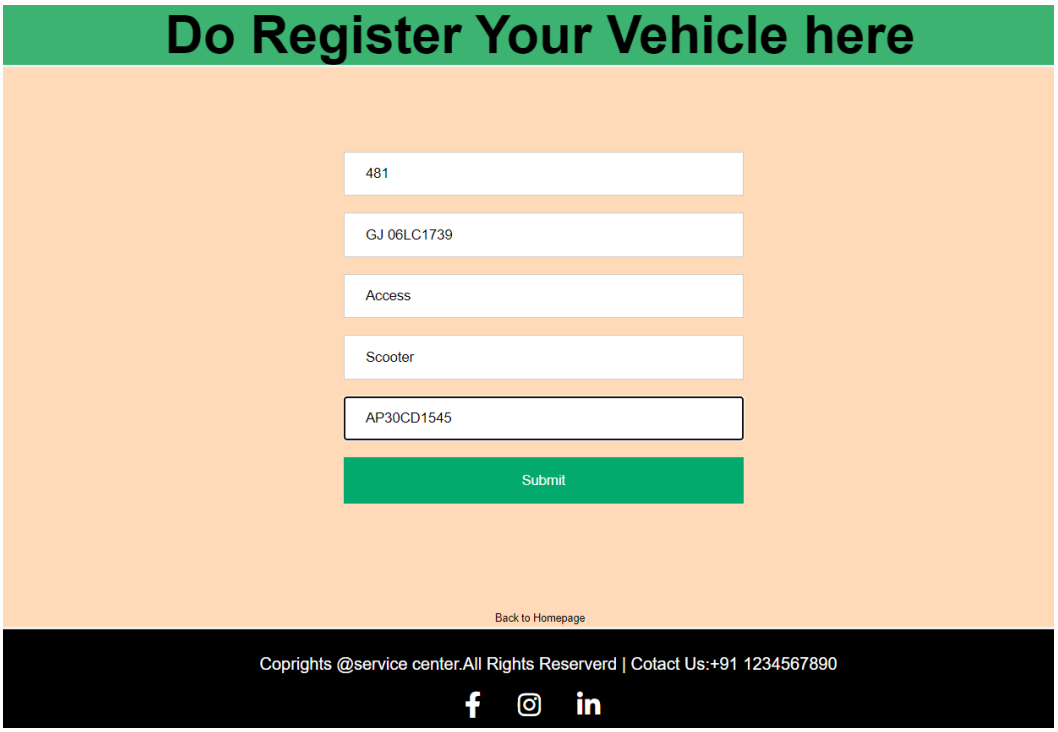
6. **vehicle.php**

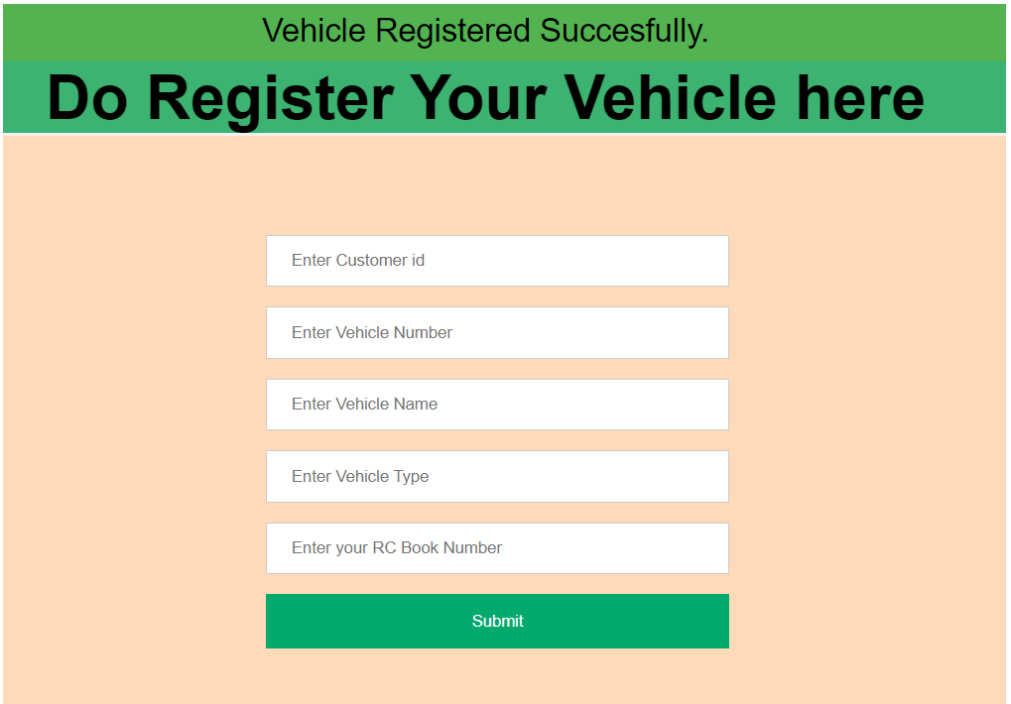
```
<?php
require_once('connection.php');

if (isset($_POST['c_id']) && !empty($_POST['c_id']) && isset($_POST['v_no']) && !empty($_POST['v_no'])
&& isset($_POST['v_name']) && !empty($_POST['v_name']) && isset($_POST['v_type']) &&
!empty($_POST['v_type']) &&  isset($_POST['RC_No']) && !empty($_POST['RC_No'])){
   //Stores Values in Variable
   $c_id = mysqli_real_escape_string($conn, $_POST['c_id']);
   $v_no = mysqli_real_escape_string($conn, $_POST['v_no']);
   $v_name = mysqli_real_escape_string($conn, $_POST['v_name']);
   $v_type = mysqli_real_escape_string($conn, $_POST['v_type']);
   $RC_No = mysqli_real_escape_string($conn, $_POST['RC_No']);

   $insertq = mysqli_query($conn, "insert into vehicle (`v_no`,`c_id`,`v_name`,`RC_No`) "
```

```
            . "values ('{$v_no}','{$c_id}','{$v_name}', '{$RC_No}')") or die(mysqli_error($conn));

    if($insertq)
    {
        $response['flag'] = 1;
        //Messege for Succesfully Registration....
        echo '<div style="background-color:#52b350; padding:10px; font-size:26px;
weight:500;"><center>Vehicle Registered Succesfully.</center></div>';
    }
    else
    {
        echo "Please Try Again Something Went Wrong" ;
    }
}

?>
```

v_no	c_id	v_name	RC_No
UP 16HJ5645	418	Fiat Ducato	AB01CD0094
UP 16HJ5646	436	StarBus 24	AB01CD0095
JK 01HN8678	465	Tata Magna	AB01CD0096
JK 01HN8679	420	Maruti Baleno	AB01CD0097
JK 01HN8680	408	Jaguar	AB01CD0098
JK 01HN8681	423	Ford Transit	AB01CD0099
JK 01HN8682	458	Accent	AB01CD0100
JK 01HN8683	419	Toyota Tacoma	AB01CD0101
JK 01HN8684	429	Bajaj Pulsar	AB01CD0102
JK 01HN8685	478	Pep plus	AB01CD0103
JK 01HN8686	404	Yamaha YZF	AB01CD0104
AB 00CC9990	425	Pep Plus	AB99CD9099
AB 00CC9890	435	Pep Plus	AB99CD9999
AB 00CC9890	425	Activa	AB99CD8899

This is the list of vehicle numbers and their owners. Any customer can register their vehicle via the form below. This would insert the details in the above database table.





v_no	c_id	v_name	RC_No
UP 16HJ5646	436	StarBus 24	AB01CD0095
JK 01HN8678	465	Tata Magna	AB01CD0096
JK 01HN8679	420	Maruti Baleno	AB01CD0097
JK 01HN8680	408	Jaguar	AB01CD0098
JK 01HN8681	423	Ford Transit	AB01CD0099
JK 01HN8682	458	Accent	AB01CD0100
JK 01HN8683	419	Toyota Tacoma	AB01CD0101
JK 01HN8684	429	Bajaj Pulsar	AB01CD0102
JK 01HN8685	478	Pep plus	AB01CD0103
JK 01HN8686	404	Yamaha YZF	AB01CD0104
AB 00CC9990	425	Pep Plus	AB99CD9099
AB 00CC9890	435	Pep Plus	AB99CD9999
AB 00CC9890	425	Activa	AB99CD8899
GJ 06LC1739	481	Access	AP30CD1545

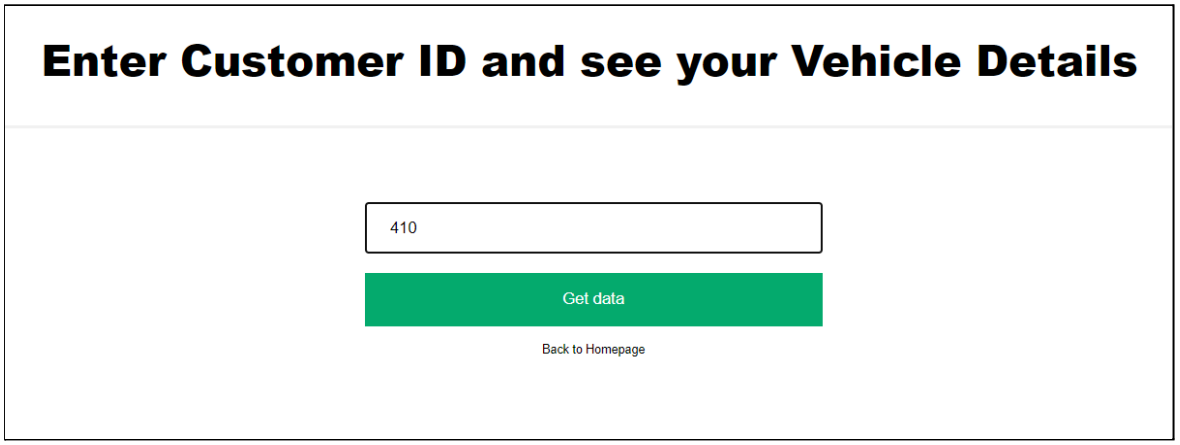
We can see that the record of Access owned by c_id 481 is successfully inserted in the database table.

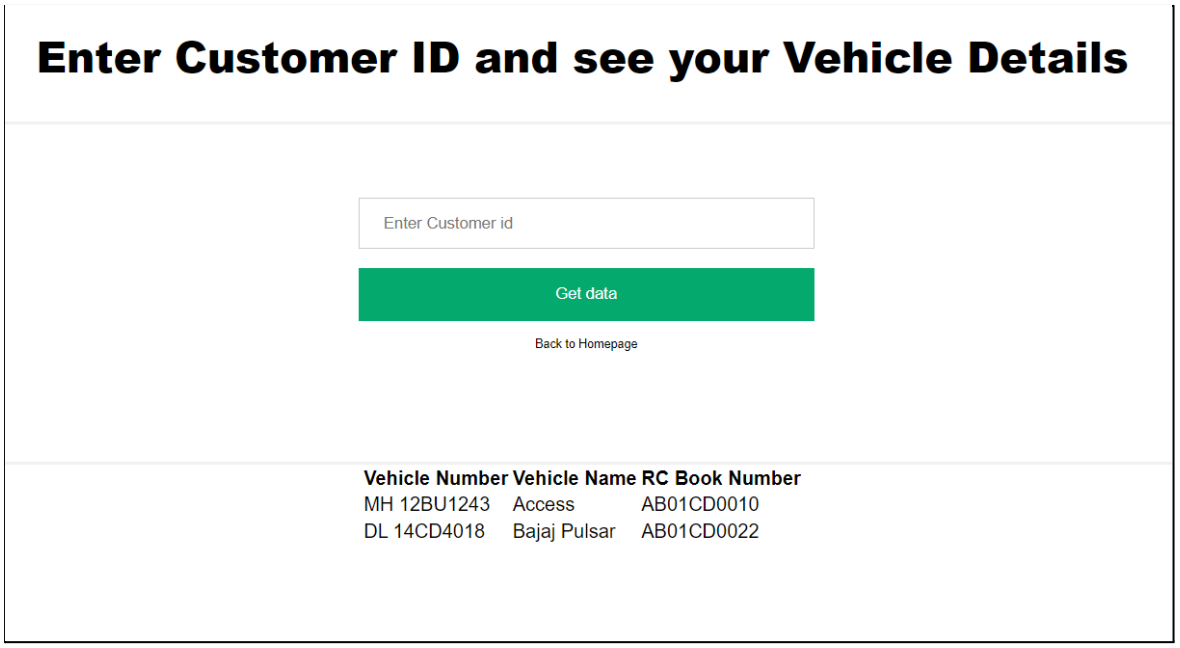
7. **fetchdata.php**

```
<?php
session_start();
require('connection.php');
if(isset($_POST['lOgin']) ){
   $c_id=$_POST['c_id'];
}
?>

<table class="table" >
      <thead style="border: 5px black">
         <tr>
            <th scope="col">Vehicle Number    </th>
            <th scope="col">Vehicle Name     </th>
            <th scope="col">RC Book Number   </th>
         </tr>
      </thead>
      <tbody>
```

```
        <?php
    include('connection.php');
    $query = mysqli_query($conn,"select * from vehicle where c_id='$c_id'");
    while($inx=mysqli_fetch_assoc($query))
    {
    ?>
            <tr>
                <td> <?php echo $inx['v_no']; ?> </td>
                <td><?php echo $inx['v_name'];?></td>
                <td><?php echo $inx['RC_No'];?></td>
            </tr>
            <?php
        }
?>
    </tbody>
    </table>
```

This would take c_id as input and will fetch its vehicle details from the database.



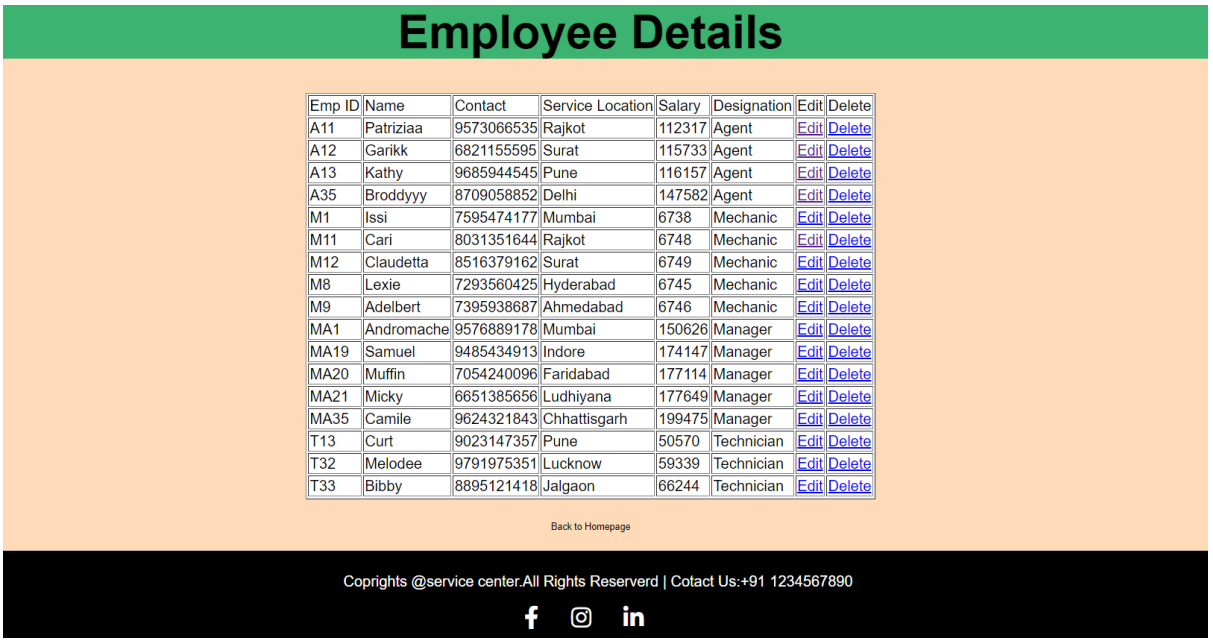


One can understand from the above that the customer with ID 410 owns two vehicles ‘Access’ and ‘Bajaj Pulsar’.

8. employee.php

```
<?php
//session_start();
$conn = mysqli_connect('localhost','root','','service_center');
if($conn === false){
   die("ERROR: Could not connect. " . mysqli_connect_error());
   }
?>



<?php
include "connection.php"; // Using database connection file here
$records = mysqli_query($conn,"select * from employe"); // fetch data from database
while($data = mysqli_fetch_array($records))
{
?>
  <tr>
   <td><?php echo $data['e_id']; ?></td>
```

```
  <td><?php echo $data['e_name']; ?></td>
  <td><?php echo $data['e_contact']; ?></td>
  <td><?php echo $data['service_location']; ?></td>
  <td><?php echo $data['salary']; ?></td>
  <td><?php echo $data['designation']; ?></td>
  <td><a href="edit.php?e_id=<?php echo $data['e_id']; ?>">Edit</a></td>
  <td><a href="delete.php?e_id=<?php echo $data['e_id']; ?>">Delete</a></td>
 </tr>
<?php
}
?>
```



Employee Details

Emp ID	Name	Contact	Service Location	Salary	Designation	Edit	Delete
A11	Patriziaa	9573066535	Rajkot	112317	Agent	Edit	Delete
A12	Garikk	6821155595	Surat	115733	Agent	Edit	Delete
A13	Kathy	9685944545	Pune	116157	Agent	Edit	Delete
A35	Broddyyy	8709058852	Delhi	147582	Agent	Edit	Delete
M1	Issi	7595474177	Mumbai	6738	Mechanic	Edit	Delete
M11	Cari	8031351644	Rajkot	6748	Mechanic	Edit	Delete
M12	Claudetta	8516379162	Surat	6749	Mechanic	Edit	Delete
M8	Lexie	7293560425	Hyderabad	6745	Mechanic	Edit	Delete
M9	Adelbert	7395938687	Ahmedabad	6746	Mechanic	Edit	Delete
MA1	Andromache	9576889178	Mumbai	150626	Manager	Edit	Delete
MA19	Samuel	9485434913	Indore	174147	Manager	Edit	Delete
MA20	Muffin	7054240096	Faridabad	177114	Manager	Edit	Delete
MA21	Micky	6651385656	Ludhiyana	177649	Manager	Edit	Delete
MA35	Camile	9624321843	Chhattisgarh	199475	Manager	Edit	Delete
T13	Curt	9023147357	Pune	50570	Technician	Edit	Delete
T32	Melodee	9791975351	Lucknow	59339	Technician	Edit	Delete
T33	Bibby	8895121418	Jalgaon	66244	Technician	Edit	Delete

Back to Homepage

Coprights @service center.All Rights Reserverd | Cotact Us:+91 1234567890

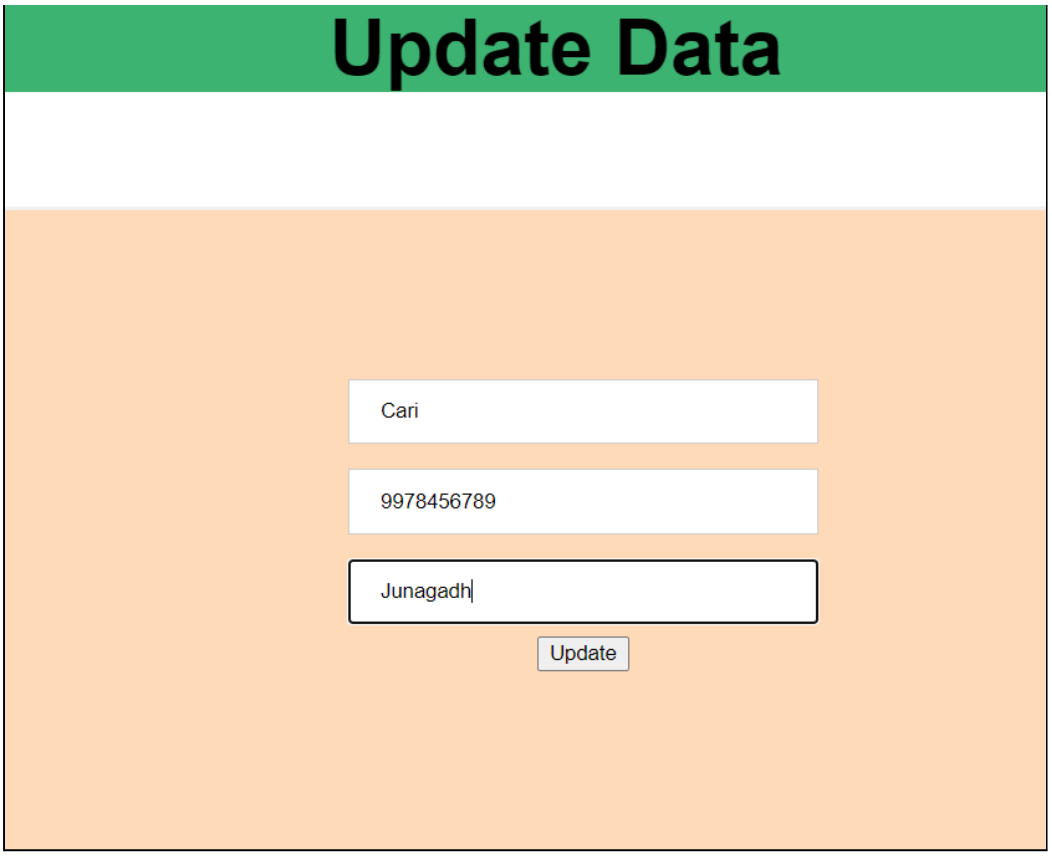
The employee section of the navigation bar would display details of the employees from the database using select query.

9. **update.php**

```php
<?php

include "connection.php"; // Using database connection file here
$e_id = mysqli_real_escape_string($conn, $_GET['e_id']);
$qry = mysqli_query($conn,"select * from employe where e_id='$e_id'"); // select query

$data = mysqli_fetch_array($qry); // fetch data

if(isset($_POST['update'])) // when click on Update button
{
    $e_name = $_POST['e_name'];
    $e_contact = $_POST['e_contact'];
    $service_location = $_POST['service_location'];
    $edit = mysqli_query($conn,"update employe set e_name='$e_name',     e_contact='$e_contact' ,
service_location = '$service_location' where e_id='$e_id'");

    if($edit)
    {
        mysqli_close($conn); // Close connection
        header("location:Employee.php"); // redirects to all records page
        exit;
    }
    else
    {
        echo mysqli_error();
    }
}
?>
```

M11	Cari	8031351644	Rajkot	6748	Mechanic	Edit	Delete

These were the details of Cari before.



M11	Cari	9978456789	Junagadh	6748	Mechanic	Edit	Delete

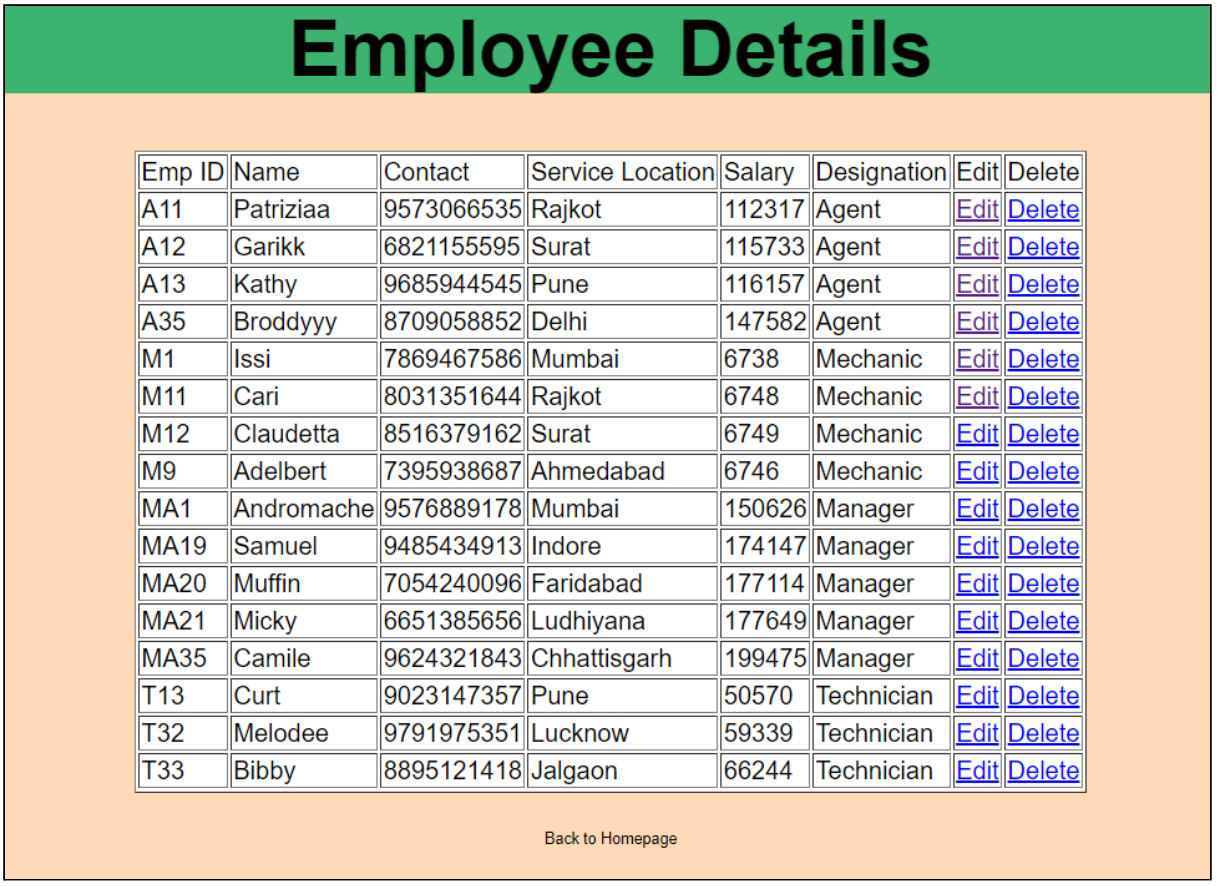
After updating/editing Cari's phone number and service location the above is the tuple changed in the relation.

10. delete.php

```
<?php

include "connection.php"; // Using database connection file here

$e_id = $_GET['e_id']; // get id through query string

$del = mysqli_query($conn,"delete from employe where e_id = '$e_id'"); // delete query
```

```php
if($del)
{
    mysqli_close($conn); // Close connection
    header("location:Employee.php"); // redirects to all records page
    exit;
}
else
{
    echo "Error deleting record"; // display error message if not delete
}
?>
```

Apart from updating, if any employee resigns or leaves the company then we can delete its record by clicking on the blue Delete hyperlink.



Employee Details

Emp ID	Name	Contact	Service Location	Salary	Designation	Edit	Delete
A11	Patriziaa	9573066535	Rajkot	112317	Agent	Edit	Delete
A12	Garikk	6821155595	Surat	115733	Agent	Edit	Delete
A13	Kathy	9685944545	Pune	116157	Agent	Edit	Delete
A35	Broddyyy	8709058852	Delhi	147582	Agent	Edit	Delete
M1	Issi	7869467586	Mumbai	6738	Mechanic	Edit	Delete
M11	Cari	8031351644	Rajkot	6748	Mechanic	Edit	Delete
M12	Claudetta	8516379162	Surat	6749	Mechanic	Edit	Delete
M9	Adelbert	7395938687	Ahmedabad	6746	Mechanic	Edit	Delete
MA1	Andromache	9576889178	Mumbai	150626	Manager	Edit	Delete
MA19	Samuel	9485434913	Indore	174147	Manager	Edit	Delete
MA20	Muffin	7054240096	Faridabad	177114	Manager	Edit	Delete
MA21	Micky	6651385656	Ludhiyana	177649	Manager	Edit	Delete
MA35	Camile	9624321843	Chhattisgarh	199475	Manager	Edit	Delete
T13	Curt	9023147357	Pune	50570	Technician	Edit	Delete
T32	Melodee	9791975351	Lucknow	59339	Technician	Edit	Delete
T33	Bibby	8895121418	Jalgaon	66244	Technician	Edit	Delete

Back to Homepage

Here, we deleted the information of an employee who worked at Hyderabad and whose employee id was M8.