Final Project

Data Centre Management System

Section 4

Team 8

Name	ID
Kaushal Joshi	201901199
Satyam Chhatrala	201901209
Vishrut Mehta	201901213
Nemin Shah	201901280

Mentor TA - Vaishnavi

Organization:

Created on: 27th November 2021

Table of Contents

[7] Section 7: Final DDL Scripts and Insert Statements

[8] Section 8: SQL Queries with Snapshots

[9] Section 9: Project Code with Output Screenshots

- Establishing connection with Pgadmin using Python
- Executing Insert statements
- Executing Select statements
- Executing Select statements using custom query
- Output screenshots

Section1: Final version of SRS

[A] Introduction

Purpose and Project Description

What is a Data centre?

A data center is a building with powerful computers that are used to run a company's services or to put it in a better way, it is a physical facility that various organizations use to house their critical applications and data. It's where the information is processed and then made available. All the data is stored, managed and circulated across the computers. A data center's design is based on a network of computing and storage resources that enable the delivery of shared applications and data. The most essential components of a data center design include routers, switches, firewalls, storage systems, servers, and application delivery controllers. In big companies, these components are in tens of thousands of servers, routers etc so that they could process big data, support cloud computing and support billions of users for services such as using Chrome, Gmail, Google cloud and search.

The database is managed by data administrators and data managers.
Their aim is to maintain the flexibility and stability of the database. The in-house servers are not owned by small scale users and hence they seek cloud-based storage and hosting services. Hence, the data uniformity and database management of these cloud-based services is a must for any cloud-based service provider.

Cloud based data management system is intended to provide solutions for organizations as well as individual users to manage their data on a cloud-based platform. The system administrator and employees of the data center can manage the servers and hardware information along with employee information with different privileges. The module will allow the system administrator to check for the profit generated by the servers' hosting websites for different clients. It will also allow the system administrator to manage different users and their data.

The software system which will be produced is called Cloud-based data management system. We will act as a company named – ADCMC (Assist Data center management Company) and the company will manage provide help to the private companies that are interested in managing their data centers and server farms and also cloud-based hosting and storage services to various users. This system is aimed to provide fast, reliable access to the companies' data and information that makes the company profitable. Also, it provides fast access to the clients' information. The system will be cross-platform and will be hosted over Postgres. The companies using this service will have full access to the database and will be able to access different data. However, the company employees using this will have some limitations to the database so that there is no breach of the data.

This is a discrete product that is aimed at providing fast, reliable and efficient service to manage the data of the data center as well as of the clients. Major components in the database will include different queries that will be made to access different data. Data tables will be created and some of them might be connected among themselves.

Real world working flows (Existing and Proposed ones) and their description:

Our database system shall dramatically simplify data centre management by giving data centre operators the ability to run efficient data centre operations and improved data centre infrastructure planning and design. We expect our system to supersede and displace the existing homegrown databases such as MS Excel, as well as reduce the number of on-site employees, eliminate unnecessary manual tasks, and reduce associated business costs.

Working flow examples:

1. Client needs to upgrade their existing hardware and software systems in their data centre

⟶ Existing flow: File system updates may take a long time to synchronize, and may be inconsistent. Excel may be too slow and lack complexity for high frequency of updates. Having no databases shall lead to poor organization.

⟶ Our proposed system: Updates in the data centre are instantaneously reflected in our database. System provides automation of changes to the infrastructure while providing a single source of information and a holistic view across all facility, IT hardware, networks, and applications.

2. Client needs to maintain up-to-date information on of multiple data centre sites & associated equipment, applications and operating systems with different contracts, warranties, process, licenses, patches and upgrade processes.

⟶ Existing flow: File systems may store duplicate information and are highly unreliable when large complex data and their relationships are to be stored. Excel lacks the speed, and cannot function properly on a large scale. Complex relationships between huge data always requires a database system.

⟶ Our proposed system: All such information is securely stored in our database with extremely high reliability. Our database provides live monitoring and visualization of the data centre, the associated status, and the changes within and across all systems and locations.

3. Client must manage multiple data centres with different maturity levels, diverse equipment, processes, and procedures.
⟶ Existing flow: All the existing methods of file system, and home grown databases on excel are inefficient when it comes to maintaining large data spanning multiple data centres.

⟶ Our proposed system: Our database system inherently stores all such information while also maintaining all the important relationships between different entities. Furthermore, our database keeps the track of power, space and capacity information which may be used by the client to determine how to utilize the data centre more efficiently to save energy and space or increase to the utilization of existing equipment

4. Client needs to allocate specific resources during power failure or security breach.

⟶ Existing flow: File system and excel are too sluggish when it comes to high frequency data retrieval in times of system failures. Hence, a dedicated database is required for immediate identification of available resources.

→ Our database makes the identification of available space, power and equipment extremely fast and convenient.

We strive to provide data management solutions which help our clients to minimize their risk, to maximize their service, and where services and applications can be delivered fast with minimal costs.

- Only Certain Authorized people have rights to create, modify and delete database.
- Real time tracking of various data-centers and their associated equipments.
- Automated reports showing maintenance cost of each service and sites of customer-company.
- Easy to use software and interactive user-interface.
- Software should run on low internet usage.
- Software should support maximum version of OS.
- Feedback as well as 24/7 helpline support system must be there.
- Maintaining statistics and history
- Alarming notification system
- PostgreSQL will be familiar to the system administrator using this product
- Admin has the access to the complete database and can modify all the features
- Some free amount of storage is available to the users initially for database storage
- The users and the admin staff should have the basic knowledge of computers.

1. Management Department:

Includes board of directors and CXO's of the company. They can access the whole database.

2. Finance Department:

Includes financial analysts or any other finance concerned authorities. They will use the database for accessing the transactions, the money earned, profit made etc.

3. Data systems reliability engineers:

They require an up-to-date record of all the machines, their current working status, and malfunctioned machines. They need to maintain the machines and fix them regularly, and hence also need the failure and maintenance report of all the machines in the data center.

4. IT Department:

Consists of the CTO and the database administrator. These people manage the database and maintain it.

5. Logistic department:

They require a prompt record regarding machines to dispatch, new machines to be brought to it, and the current machines in the center.

6. Customers:

Our customers are the companies interested in giving us the responsibility of maintaining their database. Their customer id will be and they can access their own databases tores with us by logging in with id and password.

7. Software engineers:

They require an up-to-date list of available machines and their related services, software, and hardware to appropriately schedule their tasks and allocate resources to certain programs.

8. Security engineers:

To prevent security attacks on their data systems, they need information on vulnerable networks and machines, and those which are currently affected by a security breach.

9. Server-side engineers:

They require real-time information on the usage of their servers, the resources allocated, and the number of users currently utilizing their services.

- Employee's information: Detailed information of each employee associated with the data centre management team of the client company shall be included in our product. This includes the employee id, name, age, gender, department, salary and contact number.
- Hardware information: Detailed information for every hardware (eg. data servers, etc.) such as its server id, hardware id, model number, its working status, its processor speed, bandwidth, the software running on in, date of purchase, storage capacity and type, hardware vendor, and its location in the data centre.
- Software information: Detailed information related to the software running of a specific hardware. This includes the corresponding server hosting the software, i.e. the server id, software id, language used, memory, date of creation, the type of software, its latest update patch number and date, and its parent process id.
- Adding/removing/updating a server: Server information such as their presence in the data centre, their working condition, date of purchase, date of removal, last update, repair required, and related features shall be in the database.
- Users' information (web hosting): In case of servers used for web hosting, detailed information regarding the sever id, information of current utilizing the server, the services used by the user, the duration of usage, and the charges per hour of leasing that server.
- User information (cloud storage): In case of servers which provide cloud storage, information containing the server id, the user information, the storage allocated to the user, the current plan the user has subscribed to, the memory used and memory available are some of the features for this category of users.
- Profit gained: Using the information such as charger per hour of leasing a server/service, the duration of usage, and the type of server used, a function shall be devised which can calculate the profit made by the company.
- Server information: It includes server id, status, capacity and number of active clients.

- Processor information: It includes processor id, manufacturer, speed of processor, status and memory.
- Repair information: It includes the information regarding the repair that needs to be done i.e., repair id, type, start date and finish date of repair.
- Software update information: It has its update id, update date and memory through which we come to know about the updates required in any particular software.

Intended audience and reading suggestions

The target audience for our management system consists of all those organizations which have their own data centers. These organizations range from software/hardware providers, public/private cloud providers, or any company which hosts an in-house data center.

Within an organization, the following group of people related directly or indirectly to the data center shall make extensive use of our management system

1. Data systems reliability engineers: They require an up-to-date record of all the machines, their current working status, and malfunctioned machines. They need to maintain the machines and fix them regularly, and hence also need the failure and maintenance report of all the machines in the data center. They also require the cost of repair of all such machines.

2. Logistic department: They require a prompt record regarding machines to dispatch, new machines to be brought to it, and the current machines in the center.

3. Standards engineer: They need to maintain a certain standardized set of machines in the center, and hence require the complete record of machines and their specifications.

4. Software engineers: They require an up-to-date list of available machines and their related services, software, and hardware to appropriately schedule their tasks and allocate resources to certain programs.

5. Security engineers: To prevent security attacks on their data systems, they need information on vulnerable networks and machines, and those which are currently affected by a security breach.

6. Server-side engineers: They require real-time information on the usage of their servers, the resources allocated, and the number of users currently utilizing their services.

7. Revenue team: They need to know the profit made by renting their servers, and certain financial decisions pertaining to that.

8. Sales team: They need a report describing the geographical locations of their users which utilize their cloud service, to make informed sales and advertising decisions.

Product Scope:

Benefits:

1. IT companies: customer-companies can manage and modify data related to their software and hardware availability, their services and cost associated.

2. Features like profit, cost etc. will be added so that a user using a particular service has to pay for the service he uses and the company could track its profit obtained through their provided service.

3. Special privileges to the users will be provided so that a no user can access other user's information. It would be available only if that user gives his/her access, in order to avoid any data breaches. And if he/she didn't give the access, the other user cannot access to his/her services or any other information. The whole system would be transparent in a way that the concerned authority could get access to the required amount of information.

Goals:

1. Security and backup management: Today enterprises and IT companies face an unprecedented number of threats (both internal and external) as well as server crashes. These in turn creates huge demand of data security and backup in their databases.

2. Efficient Accessibility: To be able to provide easy concurrent access as well as modification in database.

3. Accountability: Most of the IT people are professional and trustworthy but we cannot neglect the need for accountability in the data center to track the interactions people have with it. Data centers should log entry details via badge access which should probably be held by Security department and maybe at 2-3 more places. Visitors should sign in and sign out and remain under supervision at all times.

4. Monitoring: Monitoring is a key aspect that needs to be taken care of for a healthy system. We need to monitor most of the commodities that our data center provides such as bandwidth in use, storage, physical space etc.

Objectives:

1. Cost of maintenance: Each data center machine maintenance report is calculated automatically which includes the cost of maintaining each site individually.

2. Maintaining data of customer-company's employee details and customer-company's customer details.

3. Maintaining data of customer-company's software, hardware and associated services details.

4. Provide hierarchieal accessibility model where rights to access certain data are given to certain employees based on their hierarchy.

[B] Documenting the requirements collection / Fact finding phase:

Background readings:

We tried to find and understand the concept of data management of various well-known companies and researched about their work. Unfortunately, the topic of data management is such that companies would keep everything confidential and so we could not find much about it online. But we sure did find and thought about what changes could be done in our company (imaginary). We covered almost every good quality in our company which the other companies provide. We also noted some of the issues with their data management and we have tried to overcome them in our own way.

We also referred to some external materials to make this SRS.

1. https://www.sunbirddcim.com/what-is-data-center-management#:~:text=Data%20Center%20Management%20refers%20to,and%20security%20of%20the%20data : We took this as reference for understanding what is data center management and what is their work flow as well as the issues they are facing.

2. IEEE SRS Template: We learnt what particular part of SRS should be included.

3. Medium Blog 1: How to write a good SRS

4. Medium Blog 2: Reference for how to design a program.

Interview-1

Sunbird Data-center: (Role Play) Interview Plan **System:** Assist Data-center management system **Project Reference:** SF/SJ/2021/12

Interviewee: 1) Nemin shah (Role Play) Designation: CEO at Sunbird data center.

Interviewer: 1) Satyam Chhatrala
Designation: Business strategist – Assist Data-center management company

2) Kaushal Joshi
Designation: Developer – Assist Data-center management company

3) Vishrut Mehta
Designation: Developer – Assist Data-center management company

Date: 14/8/2021 **Time:** 14:30

Duration: 30 minutes **Place:** Sunbird's Office

- **Purpose of Interview:**

Preliminary meeting to get insights about working and infrastructure of Sunbird's data-center management system.

Agenda:

- Environmental concerns.
- Security concerns and efficiency concerns
- Initial ideas
- Follow-up actions

Documents to be brought to the interview:

Any documents relating to current security procedures

Q1) As we know, majority of the equipments like servers, routers, switches, security cameras etc. in a data center rely heavily on electricity and the electricity provided should also be reliable. Sometimes because of power cuts/interruptions, the equipments stop working and it takes a lot of time to restart them and take them back online. So, how do you manage to power your data center keeping in mind the above problems?
Ans. To overcome all the above problems, we have a two-layer system for uninterrupted power supply. In this system, firstly, there is a large battery-backup system. So, whenever there is a power cut for even a little amount of time, the power flows through this battery backup due to which the equipments will not go down and they would get continuous power supply from the batteries. But these batteries can supply power only for about 5 –20 mins. So, for that, secondly, we have huge diesel-powered generators which can provide consistent supply of power for a long duration of time. They get activated the moment there is a power cut. And furthermore, we have 40% of renewable sources providing power supply. So, we are not facing any issues regarding power supply.

Q2) What challenges did you face and how did you solve it? Ans. These were the challenges that we faced:

1. Excessive power consumption and Cost:

Initially, we thought that if we would have more resources and equipments, it would help with the problem of servers going down. But eventually, it resulted into loss of excess energy, power, space and resources. This even resulted to excess waste of money. So, when we came to know about DCIM (Data center infrastructure management system), it became very easy to optimize the space, reduce costs, reduce

energy wastage and even avoid servers down problem. Using this system, we can easily monitor the energy consumption and if required, can optimize it.

2. Unexpected and Unknown failures

As a data center has lots of applications like cables, network, power supply, generators, servers, cooling systems etc. going on at the same time, it is very obvious to get errors and failures leading to big problems. So, we faced the same issues till we came to know about DCIM systems. Of course, staff can look into those problems but it is too heavy and next to impossible task for them. To overcome these failures, constant monitoring and reporting is required. DCIM provides track of the performance metrics and records of the data center. So, we get an idea about the actions going on.

3. Excessive heat and temperature

As numerous equipments are running at the same time, there was a lot of heat generated due to that and it is very important to see that the things don't get too much warm. So, we decided to keep environment monitors/sensors that would take temperature readings at intervals of time. We also placed Air Cooling units which is a must for every data center. It can be air coolers- conditioners etc. Air Cooler mainly aims at keeping the data center at proper normal required temperature.

Q3) Why most of your data-centers are located in cold-climate regions?
Ans. As you that data-center generates huge amount of heat. So the idea behind installing data- centers in cold-climate region is that the air outside the data-center is considerably cooler than air inside the data-center, Allowing cooler natural air to enter the data center (after filtering and humidifying as needed) makes it possible for data-center operators to reduce their dependence on power hungry cooling systems and cut energy costs.

Q4) As we know that data-centers all over the world faces immense amount of data breach and information security related threats, so how sunbird cope up with this issue?
Ans. It's quite confidential but from bird's eye view, our Security Management solution provides a centralized system that manages physical access to different parts of the data center, including rooms, rows, and individual cabinets. It keeps a close track of people entering and exiting, and provides real-time audit reporting and surveillance feeds.

Interview-2

System: Assist Data -center management system

Project Reference: SF/SJ/2021/12

Interviewee:

1) Satyam Chhatrala(Role Play)

Designation: CEO at kyra data center (Our client)

Interviewer:

1) Nemin Shah

Designation: Business strategist – Assist Data-center management company

2) Kaushal Joshi
Designation: Developer – Assist Data-center management company

3) Vishrut Mehta

Designation: Developer – Assist Dara -center management company

Date: 15/8/2021 **Time:** 14:30

Duration: 30 minutes **Place:** Kyra's data center Office

Purpose: To identify the clients' requirements and expectations to develop an all-encompassing data centers management system which shall appeal to a larger mass in the current market.

Agenda:

- Issues with current industry standards in data center management systems.
- Requirements of clients, and their expectations.
- Additional suggestions, if any.

Documents to be brought in interview: An outline describing the clients' requirements and suggestions for our data center management system software.

Interview Questions:

[1]. Please tell us the reason you are not satisfied by the data center management system you are currently using.
⟶ The current systems are restricted to rudimentary file systems, and in some cases Excel. We currently use our in house Excel based database, which is extremely inefficient, requires a lot of manual housekeeping tasks, and increases our business cost. The existing system is very

rudimentary in terms of its speed, complexity, and scalability. They are too slow and lack complexity for high frequency updates and take a long time to synchronize. The current systems are prone to store duplicates and are highly unreliable when large complex data and their relationships are to be stored. Furthermore, no backup (or recovery) provisions are provided by the current system.

[2] Please give us an idea of your day-to-day activities revolving around managing the database of your data centers

⟶ A lot of our activities revolve around managing the database of our data centers, such as:

- Appropriate changes in the database whenever we upgrade (or remove) existing hardware and software systems in our data center.
-We need to maintain an up-to-date record of our various data centers and their associated equipment, applications, and operating systems. This includes different contracts, warranties, process, licenses, patches, and upgrade processes.
-In case of power failures, or a security breach, we must need our database intact to allocate specific resources during such tense conditions.

[3] Please let us know which operating system you prefer to use.
→ We shall be most comfortable with Windows. Though, Mac OS, and Linux shall be welcoo.

[4] Please mention the different groups of people who are authorized to access the database and make changes to it.
→ I shall provide you an exhaustive list of roles in our future correspondence, however a brief overview is as follows:

- All executives above a certain level may be given full authority to make changes in the database.
- The technical department of the data center may only view and make changes to attributes which are related to software, hardware, failures, uptimes, etc.
- The logistic department pertaining to the data center may only view and change entries related to the current, dispatched, and arrived machines, etc.

The final list of roles with their precise authority shall be forwarded to you in a very short time.

[5] Please share your concerns regarding accessibility, and privacy of information related to your data centers.
⟶ Please make sure that our information is not accessible by any of your employees, and that at any given time we have full working access to it. System down times at your ends shall not be tolerated. Furthermore, at any instance we shall have all the rights to opt out of your services. Sharing our information, or accessing it explicitly without our consent shall be considered as a legal matter, and shall be presented in the court.

Interview-3

System: Assist Data-center management system

Project Reference: SF/SJ/2021/12

Interviewee:

1) Vishrut Mehta (Role Play)

Designation: CTO at kyra data center (Our client)

Interviewer:

1) Nemin Shah

Designation: Business strategist – Assist Data-center management company

2) Kaushal Joshi

Designation: Developer – Assist Data-center management company

3) Satyam Chhatrala

Designation: Developer – Assist Data-center management company

Date: 15/9/2021 **Time:** 14:30

Duration: 30 minutes **Place:** Our company's (ADCMC) Office

Purpose: To get feedback of our currently developed data center management system from our client.

Agenda:

- To get a rough idea from our clients on how to make our database better and to know whether or not they are satisfied with the service we provide.
- Additional suggestions, if any.

Interview Questions:

1. **Are you satisfied with our currently modified (keeping in mind your suggestions) data center management system?**
   Ans. Yes, it is way better than the previous system.

2. **Are you satisfied with the speed of our current data center management system (speed of querying)?**
   Ans. Speed as increased but it is not up to the mark that we expected.

3. **Are you satisfied with the current operating system that we are using?**

   Ans. Yes.

4. **What do you think about the backup provisions that we provided to our system?**
   Ans. The backup provisions that you are providing is quite decent but it has few drawbacks that you need to take care of:

   1. Space problem: If your database is small, there won't be an issue with the current system but as the database grows, it becomes very difficult to store large amount of data and the disk will run out of space for storing backup. So, you need to keep track of your disk space at intervals of time and also keep a track of database growth to work accordingly. Keep track that newer backup that you need are not getting deleted.

   2. Speed problem: You are also storing backups across the network but these are running extremely slow eventually leading to affecting the SQL servers and it takes a lot of time to query data and shows timeout errors. As mentioned above, store the backups in

disks instead of storing it on network.

5. **Are you able to update the data available in our database? Ans.** Yes, I can easily update the data.

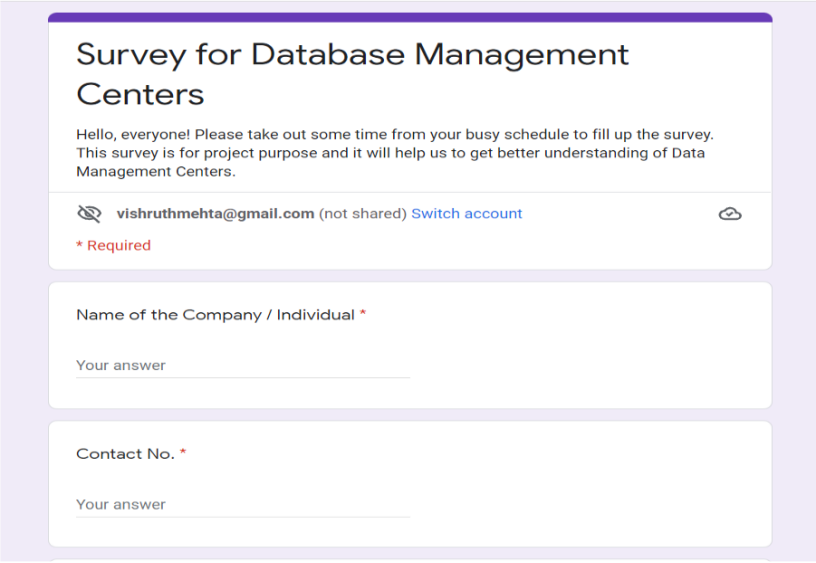
6. **Are you satisfied with the privacy policy provided by our system for all the information and data?**
Ans. Yes, I am satisfied with privacy policy.

7. **Are you able to access all the data and information from our database system? Are you facing any problems?**
Ans. Yes, quite a few of our employees are facing problem accessing the data. They are not able to access certain data. So, I hope you take care of it at the earliest.

8. **Coming to an end, how much ratings would you give to our data management system out of 10?**
Ans. I would give 8/10.

Questionnaire:

Google form link: https://forms.gle/fiQ2k8BeyUrrJBmAA



How often do you inspect your database to look for errors or to review any records? *

- ◯ Everyday
- ◯ Once a week
- ◯ Once a month
- ◯ Other: _____

Do you use a third-party address-verification services? *

- ◯ Yes
- ◯ No
- ◯ Maybe

How many separate databases do you use to hold and analyze customer data? *

- ◯ Less than 5
- ◯ More than 5 but less than 10
- ◯ More than 10

Are there any special privacy or security requirements to uphold? *

- ◯ Yes
- ◯ No

What is your data storage and backup strategy in case if the data were lost or become unusable later?

Your answer

Do you have a single person in charge of data management? *

- ◯ Yes
- ◯ No
- ◯ Maybe

Are there any data sharing requirements? *

- ◯ Yes
- ◯ No
- ◯ Maybe

How long do you want your data to be preserved? *

- ◯ A month
- ◯ 6 months
- ◯ A year
- ◯ Other:

What file formats are you using? *

- ☐ Proprietary file like .xls or .docx
- ☐ Open file like .csv or .rtf
- ☐ Other:

Who will maintain your data for the long term?

Your answer

Are you facing any problems in maintaining your database? *

- ◯ Yes
- ◯ No

If the answer to your above problem is Yes, then what problems are you facing?

- ☐ Lack of organization
- ☐ Complexity issues
- ☐ Financial issues
- ☐ Lack of space
- ☐ Other:

Do you need someone who could store your data in a database? *

- ◯ Yes
- ◯ No
- ◯ Maybe

Do you have any questions for us?

Your answer

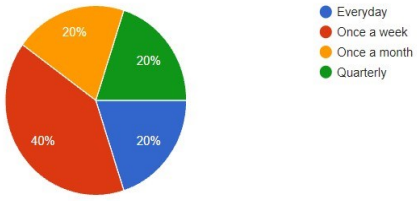
Please note any comments / suggestions you have on Database Management services.

Your answer

Summary of responses:

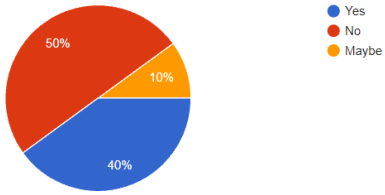
How often do you inspect your database to look for errors or to review any records?

10 responses



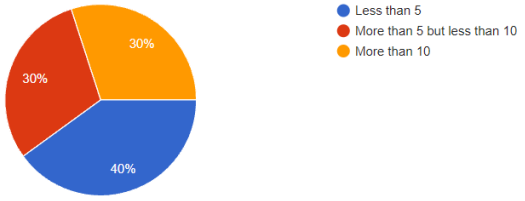
Do you use a third-party address-verification services?

10 responses



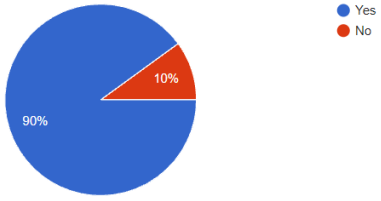
How many separate databases do you use to hold and analyze customer data?

10 responses



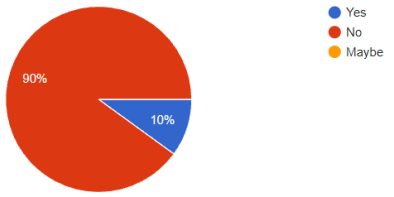
Are there any special privacy or security requirements to uphold?

10 responses



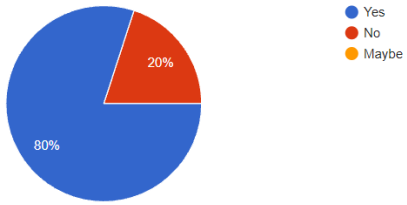
Do you have a single person in charge of data management?

10 responses



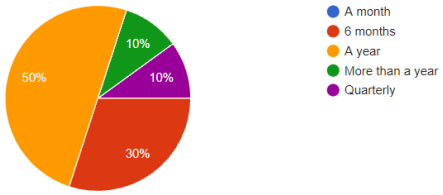
Are there any data sharing requirements?

10 responses



How long do you want your data to be preserved?

10 responses

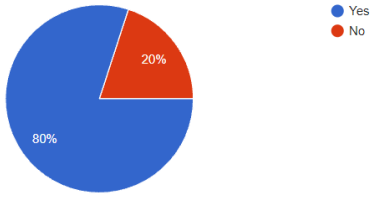


What file formats are you using?

10 responses

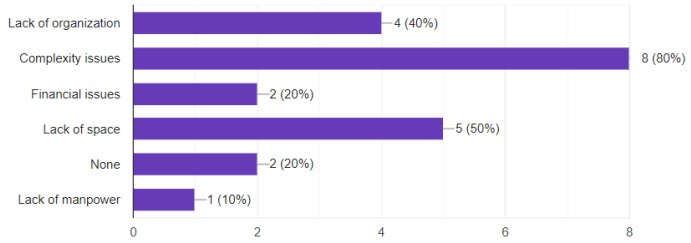
Are you facing any problems in maintaining your database?

10 responses



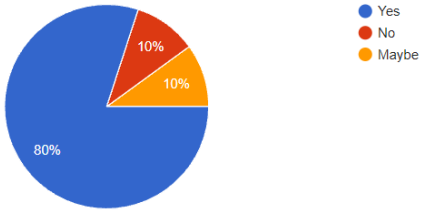
If the answer to your above problem is Yes, then what problems are you facing?

10 responses



Do you need someone who could store your data in a database?

10 responses



Observation:

System: Assist Data-center management system

Project Reference: SF/SJ/2021/12

Observations by: Vishrut Mehta (Role play) (Company employee)

Date: 24/9/2021 **Time:** 14:30

Duration: 45 minutes **Place:** Our company's (ADCMC) Office

Observations (Note: Some of these are our assumptions):

These observations are made by visiting the company and by taking a survey.

1. Chances for data leakage from the site. Privacy should be increased.
2. Any new privacy measures must not offend staff.
3. More filtering features need to be added for the ease of agents.
4. We have to make our query execution time faster by using some data structures to answer queries.
5. Agents should be able to edit multiple customers data at any particular time.
6. Tables having the same details should synchronize with each other in a nicer manner.
7. Majority of clients inspect or review their database every week.
8. Majority of them have less than 5 separate databases but some of them use 5 or more than 5 databases.
9. Many of them have more than one person for managing database as it is quite complex.
10. 80% of clients have data sharing requirements.
11. In total 80% of them requires that their data should be preserved for at least 6 months to a year.
12. Majority of them are facing problems in maintaining their database and it is due to lack of complexity, lack of space, financial issues etc. So, 80% of them requires someone to manage their database.

[C] Fact Finding Chart:

Objective	Technique	Subject(s)	Time Commitment
To get an idea about the working flow of different companies that manages the databases	Background Reading	Company Reports and Web articles	0.25 day
To get an insight about the working of Sunbird's database management system and ideas regarding designing our system	Interview	Director	30 minutes
To get an idea regarding the client's requirements and expectations to develop our database management system	Interview	1 Client	30 minutes
To get feedback about thecurrent developed management system andmake changes as per requirements	Interview	1 Client	30 minutes
To get client's opinion regarding their requirements from the database management system	Survey/Questionnaire	Clients / Audience	1 day
To follow up development of	Observation	1 Creative staff	45 minutes

business understanding			

[D] List Requirements:

- Only Certain Authorized people have rights to create, modify and delete database.
- Real time tracking of various data-centers and their associated equipments.
- Automated reports showing maintenance cost of each service and sites of customer-company.
- Easy to use software and interactive user-interface.
- Software should run on low internet usage.
- Software should support maximum version of OS.
- Feedback as well as 24/7 helpline support system must be there.

[E] User classes and characteristics

1. Management Department:
   Includes board of directors and CXO's of the company. They can access the whole database.

2. Finance Department:
   Includes financial analysts or any other finance concerned authorities. They will use the database for accessing the transactions, the money earned, profit made etc.

3. Data systems reliability engineers:
   They require an up-to-date record of all the machines, their current working status, and malfunctioned machines. They need to maintain the machines and fix them regularly, and hence also need the failure and maintenance report of all the machines in the data center.

4. IT Department:
   Consists of the CTO and the database administrator. These people manage the database and maintain it.

5. Logistic department:
   They require a prompt record regarding machines to dispatch, new machines to be brought to it, and the current machines in the center.

6. Customers:
   Our customers are the companies interested in giving us the responsibility of maintaining their database. Their customer id will be and they can access their own databases stores with us by logging in with id and password.

7. Software engineers:
   They require an up-to-date list of available machines and their related services, software, and hardware to appropriately schedule their tasks and allocate resources to certain programs.

8. Security engineers:
   To prevent security attacks on their data systems, they need information on vulnerable networks and machines, and those which are currently affected by a security breach.

9. Server-side engineers:
   They require real-time information on the usage of their servers, the resources allocated, and the number of users currently utilizing their services.

[F] Operating Environment

We have used pg Admin 4, which is a management tool for PostgreSQL.
PostgreSQL is a powerful, open-source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads.
We use version 13 of PostgreSQL.

Minimum Hardware requirements:

- 1 GHz processor
- 2 GB of RAM
- 512 MB of HDD

Minimum Software Requirements
Windows™ 7 SP1 (desktop) or 2008 R2 (server) and above or macOS 10.12 and above or Debian 9 (Stretch), 10 (Buster) or
Ubuntu 16.04 (Xenial), 18.04 (Bionic), 19.10 (Eoan), 20.04 (Focal) or RHEL/CentOS 7 & 8 (x86_64)
or Fedora 31 & 32 (x86_64)

For running PG admin 4, following browsers are supported:

- Chrome 72+
- Firefox 65+
- Edge 44+
- Safari 12+

[G] Product Functions:

Employee's information: Detailed information of each employee associated with the data centre management team of the client company shall be included in our product. This includes the employee id, name, age, gender, department, salary and contact number.

Hardware information: Detailed information for every hardware (eg. data servers, etc.) such as its server id, model number, its working status, its processor speed, bandwidth, the software running on in, date of purchase, capacity, hardware vendor, and its location in the data centre.

Software information: Detailed information related to the software running of a specific hardware. This includes the corresponding server hosting the software, i.e. the server id, the type of software, its latest update patch number and date, and its parent process id.

Adding/removing/updating a server: Server information such as their presence in the data centre, their working condition, date of purchase, date of removal, last update, repair required, and related features shall be in the database.

Users' information (web hosting): In case of servers used for web hosting, detailed information regarding the sever id, information of current utilizing the server, the services used by the user, the duration of usage, and the charges per hour of leasing that server.
User information (cloud storage): In case of servers which provide cloud storage, information containing the server id, the user information, the storage allocated to the user, the current plan the user has subscribed to, the memory used and memory available are some of the features for this category of users.

Data backup: The most recent backup time and number, this backup shall be stored in a separate database and would be updated regularly.

Profit gained: Using the information such as charger per hour of leasing a server/service, the duration of usage, and the type of server used, a function shall be devised which can calculate the profit made by the company.

[H] Privileges:

Updates in the database: Personnel in the data centre management team of the client company are allowed to make changes in the database, once approved by the system administrator which has full access to the database. The specifics shall be specified in the client company primer.

For instance,

- Updates in software/firmware of a machine: Only the software team of the data centre may make updates.

  Updates in arrival of a new machine in the centre: Only the logistics team of the data centre may make updates.
- Daily profit made by a cloud server: Only the finance team of the may make updates.
- Employee information: Only the client's HR, and the legal team has access to these sections of the database.
- System administrator operating the database may make updates to any entries in the database.

Access to database: Only the respective members of the data centre management team of the client company have access to the data centre. None of our employees are allowed to access the clients datacenter's data.

For instance,

- The software team of the client may not access the data related to logistic and financial attributes of the data centre.
- The logistics team may not get hold of the data pertaining to IT and financial attributes of the data centre.
- Likewise, the financial team may not access any other data except finance related attributes.
- Employee information: Only the client's HR, and the legal team has access to these sections of the database.
- System administrator may access the entire database of the data centre.

Update features/attributes: IT developers at our end shall add/delete/update features of the database service that we provide, however this shall only be committed to the final database once it has been tested and has got the approval of the system administrator.

It is worth noting that the specific privileges shall vary from company to company. One client might want its IT department to view the logistical attributes, while another client may not approve of this. However, we are committed to provide tailor-made database of clients' data centres according to their requirements and demands. The above-mentioned points are rational assumptions from our side.

[I] Assumptions:

- All the authorized users have unique ID and password.
- User's system satisfies OS requirements
- User's system satisfies internet bandwidth requirements.

[J] Business Constraints:

- Prevent multiple logins using the same id and passwords for security reasons.
- New employees would be assigned assets such as email address.
- Clients will be offered discount of 10% if their bills amount greater than $10,000 in any calendar year. (Their bills are dependent on their database size and complexity).
- Due to bounded space and limited financial, we will accept only a certain number of companies as our clients.
- The clients will be given a limited space depending on the contract they sign and they will be allowed to store their database in the space provided. Any additional space requirement will be charged.
- In case of any illegitimate act by the client, our company could/would not be held responsible and in such cases, we have the rights to terminate the contract.

Section2: Noun Analysis

NOUN-ANALYSIS TABLES

1. Noun and Verb Analysis

Noun	Verbs
Data center	Used
Building	Run
Computer	Put
Service	Maintains
Organization	Take
Name	Is processed
A Physical Facility	is
House	Made
Application	Is Stored
Data	Managed
Information	Circulated
Design	Is based
Network	Enabled
Computing	Shared
Storage resource	Include
Delivery	Could
Application	Google
Users	Own
Data Center Design	Using

Servers	Is Managed
Components	Is maintained
Switches	Are
Storage system	Owned
Application delivery controls	Seek
Companies	Must
Components	Is intended
Controllers	Provide
Process	Manage
Data	Will allow
Support cloud computing	Check
Tens	Generated
Billions	Can manage
Chrome	Hosting
Gmail	Maintain
Cloud	Will be produced
Search	Is called
Each employee	Named
Feedback	Will manage
Status	Host
Active clients	Provide
Processor ID	Produce
Manufacturer	Maintain

Database	Provide
Data administrators	Help
Data managers	Are interested
Aim	Managing
Flexibility	Is aimed
Stability	Hosting
Small scale user	Makes
Hosting services	provides
Data uniformity	Will be
Database Management	Hosted
Service provider	Using
Cloud	Will have
Data management system	is
Solutions	Aimed
Cloud-based platform	Providing
System administrator	Manage
Employees	Will include
Hardware information	Made
Employee information	Existing
Hardware ID	Is
Date of purchase	Run
Vendor	Host
Storage capacity and type	Are

Privileges	Created
Module	Might be
Profit	Connected
Servers	Shall
Clients	Simplify
System administrator	Giving
User data	Run
Software System	Improve
Data management system	Expect
Company	Supersede
Data center	Displace
Server farms	Existing
Storage service	Reduce
Assist Data center management Company	Eliminate
Postgres	Associated
Employees	Needs
Limitations	Upgrade
Product	Take
Breach	Synchronize

Clients	May be
Major Components	Having
Queries	Shall
Data tables	Lead
Data center operators	Proposed
Infrastructure Planning	Are
Facility	Reflected
Design	Providing
MS Excel	Needs
Number	Maintain
Employees	Associated
Tasks	Operating
Business	Upgrade
Costs	Stored
Client	Lacks
Hardware	Cannot
Software	Duplicate
Equipment	Existing
Relationships	May
Start Date	Utilizing
Finish Date	Provide
Type	Allocated

Noun	Verbs
Flow	Are
File	Giving
Organization	Maintaining
Updates	Will be
Time	Can
Excel	Logging
Lack	Require
Complexity	Related
Frequency	Allocate
Updates	Prevent
Automation	Need
Changes	Affected
View	Utilizing
Contracts	Manage
Warranties	Prompt
File systems	Regarding
Licenses	Require
Patches	Dispatch
Processes	Brought
UpdateID	Proposed
UpdateDate	Are
Memory	Reflected

Management Department	Includes
Board of Directors	Concerned
CXO	Can
Access	Will use
Finance Department	Earned
Analysts	Made
Finance	Required
Authorities	Accessing
Transactions	Working
Money earned	Need
Profits made	Maintain
Data system reliability engineers	Fix
Records	Consists
Machines	Detailed
Status	Associated
Failure	Shall be included
Maintenance Reports	Running
IT Department	Adding
CTO	Removing
Space	Requires
Capacity	Proposed
Time	Stored

Database administrator	Updating
People	Related
Logistic Department	Hosting
Machines	Regarding
Customers	Server
Date of creation	Utilizing
Responsibility	Provide
Database stores	Allocated
Software engineers	Containing
List	Has subscribed
Software	Shall be stored
Hardware	Would be updated
Schedule	Shall be devised
Tasks	Can calculate
Resources	Comes
Programs	Mantaining
Security engineers	Spanning
Security attacks	Centers
Networks	Proposed
Relationships	Keeps
Entities	May
Energy	Be
Track	Used
Power	Determine

Security breach	Provide
Server-side engineers	Live
Usage	Associated
Number of users	Must
Servers	Manage
Services	Centers
Employee's information	Existing
Data centre management team	Existing
Client company	File
Product	Excel
Employee ID	Are
Name	Utilize
Age	Save
Gender	Existing
Department	Needs
Salary	Allocate
Contact Number	Existing
Hardware information	Excel
Server ID	Are
Automated	Comes
Reports	Dedicated
People	Is
Rights	Can

Model number	Required
Status	Makes
Bandwidth	Strive
Software information	Provide
Date of purchase	Make
Capacity	Help
Hardware vendor	Maximize
Location	Minimize
Type	Modify
Patch number	Delete
Parent process ID	Tracking
Condition	Associated
Date of removal	Showing
Repair ID	Delivered
Features	Authorized
User's information	Have
Web hosting	Create
Duration of usage	
Charges per hour	
Identification	
Times	
Data Retreival	
Sites	
Customer-company	

Cloud storage	
Memory	
Category of users	
Data backup	
Backup time	
Leasing	
Scale	
Relationship	
Reliability	
Monitoring	
Visualization	
Status	
Changes	
Systems	
Location	
Client	
Data	
Maturity	
Levels	
Equipment	
Processes	
Procedure	
Flow	
Method	
Language used	
Software ID	

2. Rejected Nouns & Verbs list

Noun	Verbs
Data center	Duplicates
Building	Vague
Computer	General
Service	Irrelevant
Organization	Duplicates
Name	Duplicates
A PhysicalFacility	Irrelevant
House	Vague
Application	Duplicates
Data	Duplicates
Information	Duplicates
Design	Irrelevant
Network	Duplicates
Computing	Irrelevant
Storage resource	General
Delivery	Vague
Application	Duplicates
Users	Attribute/Duplicates
Data Center Design	General

Servers	Attribute/Duplicates
Components	Duplicates
Switches	Irrelevant
Storage system	General
Application delivery controls	Irrelevant
Companies	Duplicates
Components	Duplicates
Controllers	Vague
Process	Duplicates
Data	Duplicates
Support cloud computing	Irrelevant
Tens	Vague
Billions	Vague
Chrome	Irrelevant
Gmail	Irrelevant
Cloud	Duplicates
Search	Vague
Each employee	Duplicates
Feedback	Irrelevant

Database	Duplicates
Data administrators	Attribute
Data managers	Attribute
Aim	Vague
Flexibility	Irrelevant
Stability	Irrelevant
Small scale user	Duplicates
Hosting services	Irrelevant
Data uniformity	Irrelevant
Database Management	Duplicates
Service provider	Irrelevant
Solutions	Vague
Cloud-based platform	Irrelevant
System administrator	Attribute
Hardware information	Attribute
Employee information	General
Privileges	Irrelevant
Module	Vague
Profit	General
Clients	Attribute
User data	General
Software System	Duplicates
Server farms	Irrelevant
Storage service	General
Assist Data center management Company	Irrelevant
Postgres	Vague

Limitations	Vague
Breach	Vague

Major Components	Irrelevant
Queries	Irrelevant
Data tables	General
Data center operators	General
Infrastructure Planning	Irrelevant
Facility	General
MS Excel	Vague
Number	Irrelevant
Tasks	Duplicates
Business	General
Costs	Irrelevant
Hardware	Duplicates
Software	Duplicates
Equipment	Duplicates
Relationships	Irrelevant

Flow	Vague
File	General
Updates	Duplicates
Time	Duplicates
Lack	Vague
Complexity	Irrelevant
Frequency	Vague
Automation	Irrelevant
Changes	General
View	Vague
Contracts	Irrelevant
Warranties	Irrelevant
File systems	Duplicates
Licenses	General
Patches	Irrelevant
Processes	Duplicates

Management Department	Attribute
Board of Directors	Irrelevant
CXO	General
Access	Duplicate
Finance Department	Attribute
Analysts	general
Finance	general
Authorities	General
Transactions	General
Money earned	Irrelevant
Profits made	General
Data system reliability engineers	Irrelevant
Records	Irrelevant
Machines	General
Failure	Duplicate
Maintenance Reports	General
IT Department	Attribute
CTO	Vague
Space	Duplicate
Time	Duplicate
Database administrator	Attribute

People	Duplicate
Logistic Department	General
Customers	Attribute
Companies	Duplicate
Responsibility	Irrelevant
Database stores	General
Software engineers	General
List	Duplicate
Software	Duplicate
Hardware	Duplicate
Schedule	General
Tasks	Irrelevant
Resources	Duplicate
Programs	Duplicate
Security engineers	Attribute
Security attacks	General
Networks	Duplicate
Relationships	Duplicate
Entities	Duplicate
Energy	Duplicate
Track	Vague

Power	General
Security breach	General
Server-side engineers	Attribute
Usage	Irrelevent
Number of users	Vague
Servers	Duplicate
Services	Duplicate
Employee's information	General
Data centre management team	General
Product	General
Employee ID	Attribute
Name	General
Age	General
Gender	General
Department	Duplicate
Salary	General
Contact Number	Attribute
Automated	Irrelevant
Reports	Irrelevant
Rights	General
Model number	General

Bandwidth	General
Software information	Duplicate
Location	Duplicate
Patch number	General
Parent process ID	General
Condition	Vague
Date of removal	General
Features	General
User's information	Duplicate
Web hosting	General
Identification	Duplicate
Times	Vague
Data Retreival	Duplicate
Sites	Duplicate
Cloud storage	Duplicate
Memory	Duplicate
Category of users	Attribute
Data backup	Duplicate
Backup time	General
Leasing	Irrelevant
Scale	Irrelevant

Reliability	General
Monitoring	Duplicate
Visualization	Duplicate
Changes	General
Systems	Duplicate
Data	Duplicate
Maturity	Irrelevant
Levels	General
Equipment	Duplicate
Processes	Duplicate
Procedure	Duplicate
Flow	General
Method	General

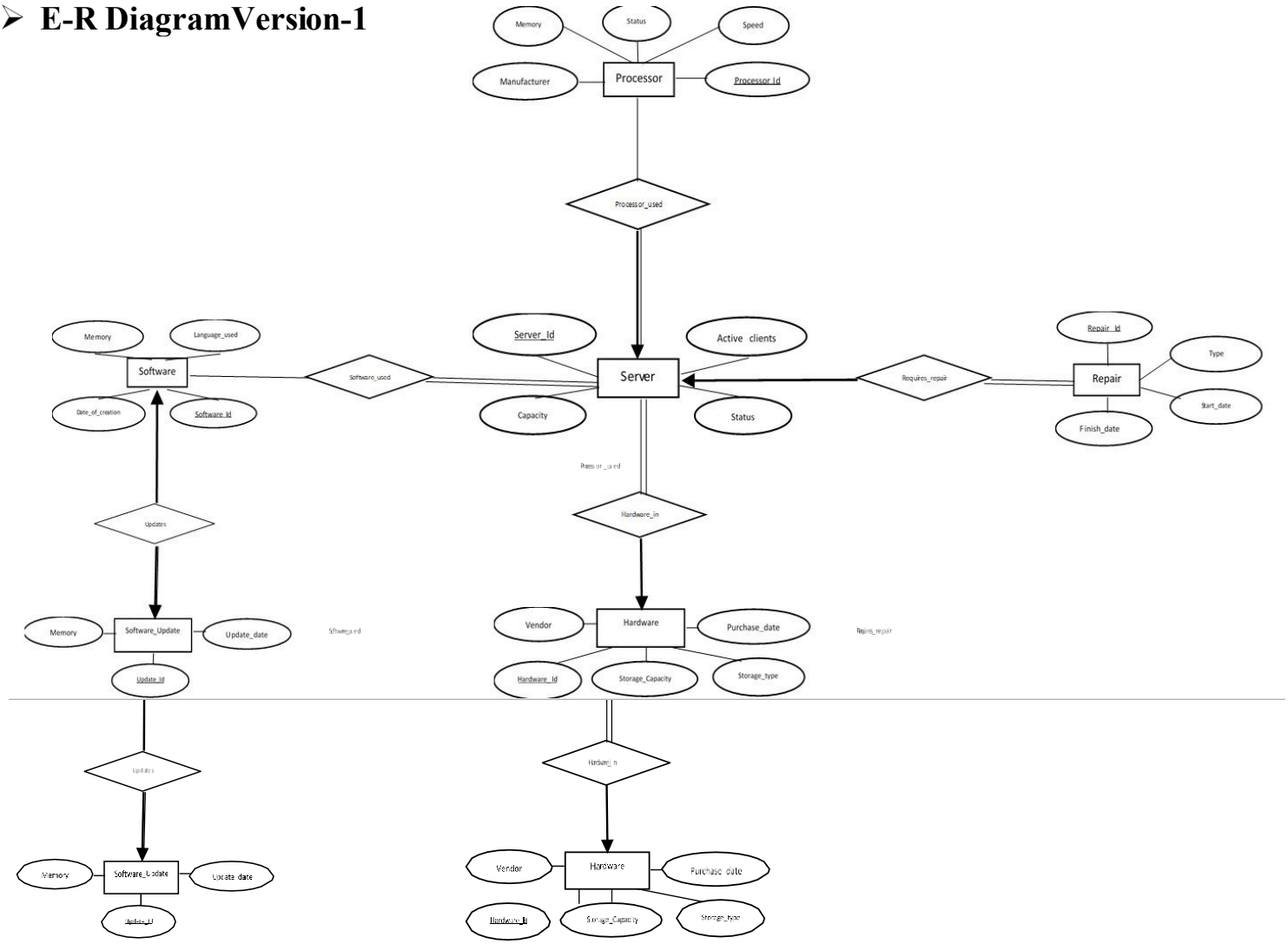
3. Accepted Noun & Verbs list

Candidate entity set	Candidate attribute set	Candidate Relationship set	Cardinality	Participation constraint
Server	Server_Id Status Capacity Active_clients	Provides AccessTo ProcessorUsed HardwareIn SofwareUsed RequiresRepair	One Many One Many Many One	Partial Total Total Partial Partial Total
Processor	Processor id Manufacturer Speed Status Memory	ProcessorUsed	Many	Partial
Hardware	Hardware id Date of purchase Vendor Storage capacity Storage type	HarwareIn	One	Partial
Software	Software id Language_used Memory Date_of_creation	SoftwareUsed Updates	Many One	Partial Parital
Repair	Repair_Id Type Start_date Finish_date	RequiresRepair	Many	Total
Software Update	Update_id Update_date Memory	Updates	One	Total

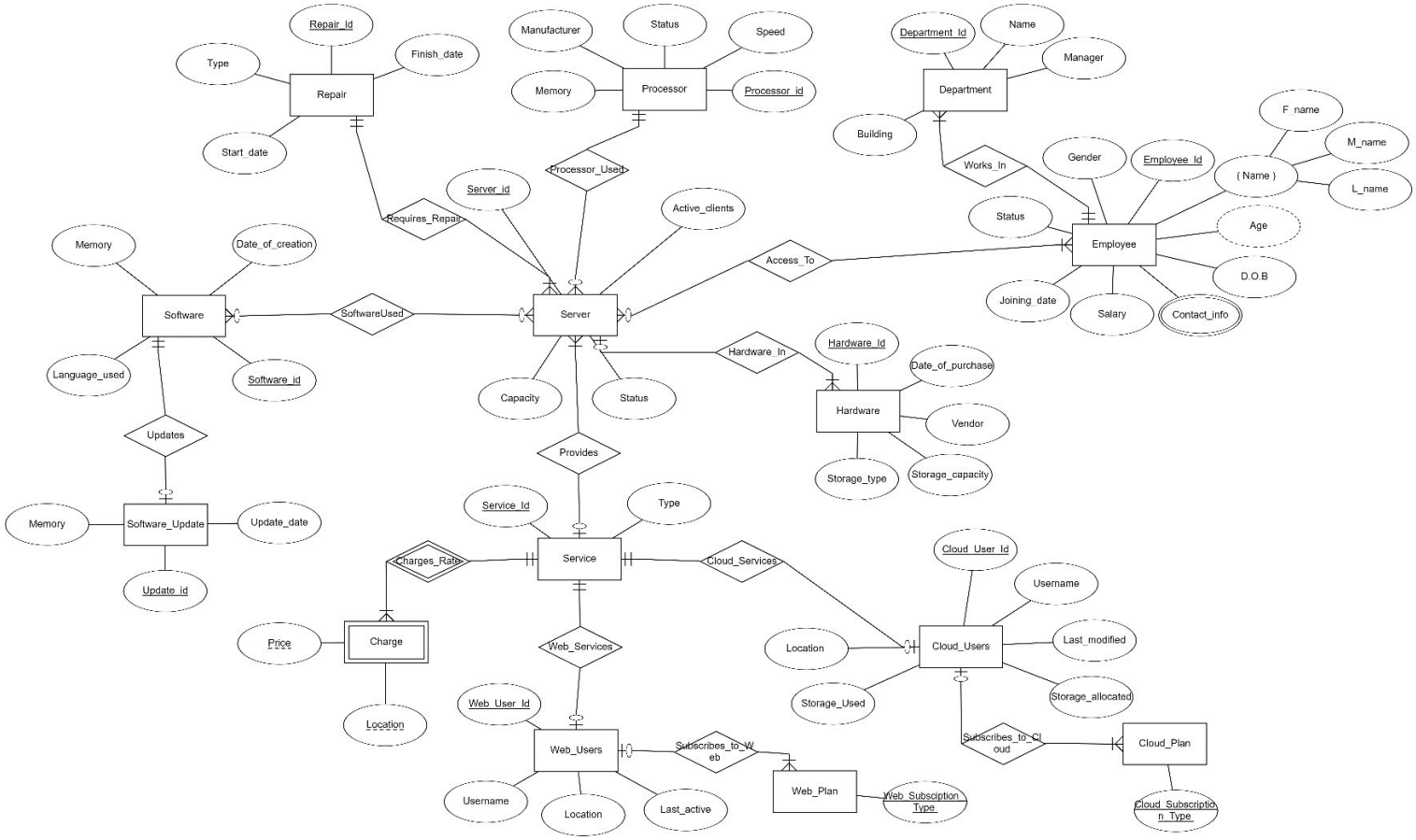
Service	Service_Id Type	Provides WebService CloudService ChargesRate	Many One One Many	Total Partial Partial Total
Web_users	Web_User_Id Username Last_active Location	WebService SubscribesToWeb	One Many	Total Total
Cloud_users	Cloud_User_Id Username Last_modified Storage_allocated Storage_used Location	CloudService SubscribesToCloud	One Many	Total Total
Employee	Employee_IdName Age Gender DOB Contact_info Salary Joining_date Status	AccessTo WorksIn	Many Many	PartialTotal
Department	Department_Id Name Manager Building	WorksIn	One	Total
Charge (WEAK)	Price Location	ChargesRate	One	Partial
WebPlan	Web_Subscription_Type	SubscribesToWeb	One	Partial
CloudPlan	Cloud_Subscription_Type	SubscribesToCloud	One	Partial

Section3: ER-Diagrams all versions

- **E-R DiagramVersion-1**

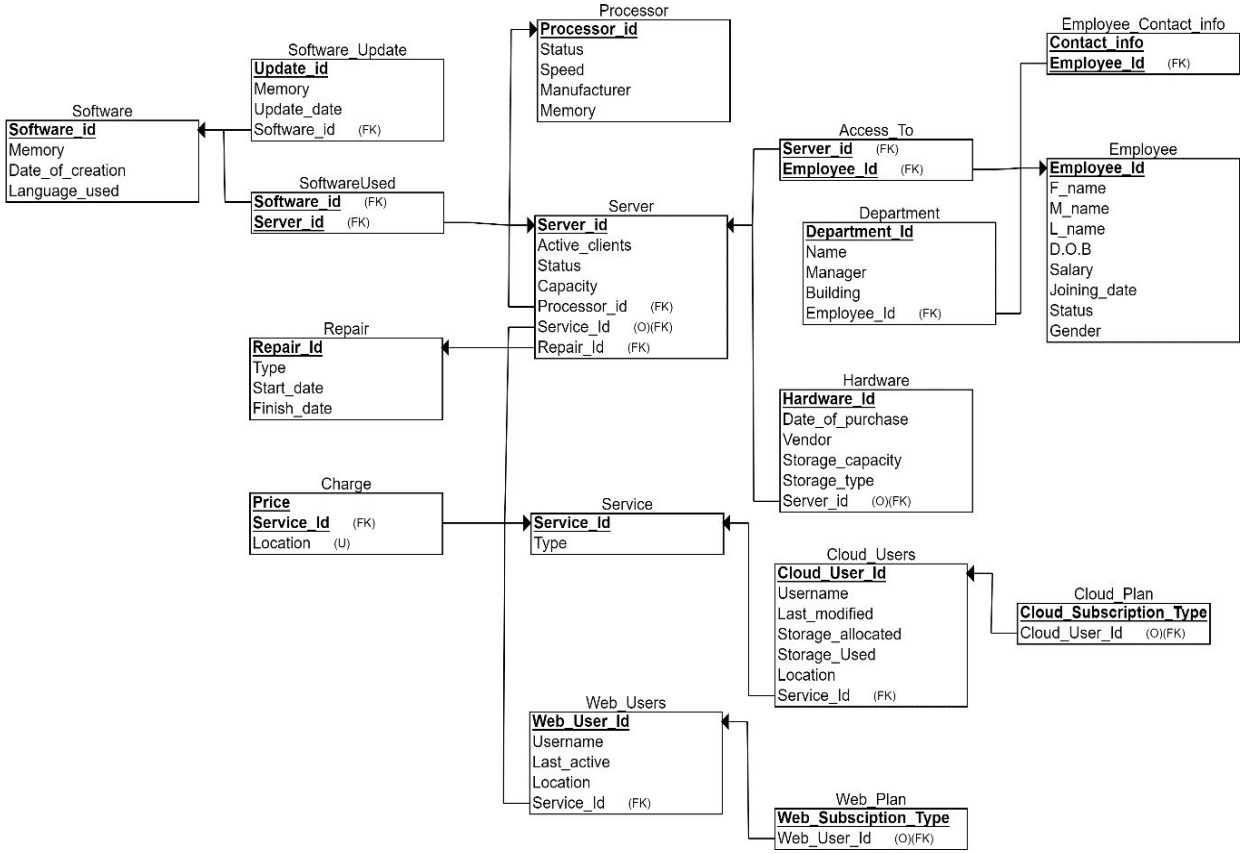


➢ E-R Diagram Version 2 (Final version)



Section4: Conversion of Final ER-Diagram to Relational Model

Relational mapping Diagram



Relational Model

1. **Software (Software_id**, Memory, Date_of_Creation, Language_used)
2. **Software_Update (Update_Id**, Memory, Update_date,Software_Id)
3. **Processor(Processor_Id**,Status,Speed,Manufacturer,Memory)
4. **Service(Service_Id**, Type)
5. **Web_Users(Web_User_Id**,Username,Last_active,Location,Service_Id)
6. **Cloud_Users (Cloud_User_Id**, Username, Last_modified, Storage_allocated, Location, Service_Id)
7. **Charge(Price, Service_Id**,Location)
8. **Cloud_Plan(Cloud_Subscription_Type**,**Cloud_User_Id**)
9. **Employee (Employee_Id**, F_name,M_name,L_name, D.O.B, Salary, Joining_date, Status, Gender)
10. **Repair(Repair_Id**,Type,Start_date,Finish_date)
11. **Web_Plan(Web_Subscription_Type**,**Web_User_Id**)
12. **Department(Department_Id**,Name,Manager,Building,Employee_Id)

13. **Employee_Contact_Info(Contact_Info,Employee_Id)**
14. **Server(Server_Id**,Active_clients,Status,Capacity,Processor_Id,Service_Id,Repair_Id)
15. **Hardware (Hardware_Id**, Date_of_purchase, Vendor, Storage_Capacity, Storage_type, Server_Id)
16. **SoftwareUsed(Software_Id,Server_Id)**
17. **Access_To (Server_Id,Employee_Id)**

Section5:Normalization and Schema Refinement.

Normalization

[1] Converting to 1NF

1. Software (Software_id, Memory, Date_of_Creation, Language_used)
2. Software_Update (Update_Id, Memory, Update_date, Software_Id)
3. Processor (Processor_Id, Status, Speed, Manufacturer, Memory)
4. Service (Service_Id, Type)
5. Web_Users (Web_User_Id, Username, Last_active, Location, Service_Id)
6. Cloud_Users (Cloud_User_Id, Username, Last_modified, Storage_allocated, Location, Service_Id)
7. Charge (Price, Service_Id, Location)
8. Cloud_Plan (Cloud_Subscription_Type, Cloud_User_Id)
9. Employee (Employee_Id, F_name, M_name, L_name, D.O.B, Salary, Joining_date, Status, Gender)

Previously, contact_info was a multivalued attribute in employee. We removed that and made a new table named Employee_Contact_Info to make it 1NF.

10. Employee_Contact_Info (Contact_Info, Employee_Id)
11. Repair (Repair_Id, Type, Start_date, Finish_date)
12. Web_Plan (Web_Subscription_Type, Web_User_Id)
13. Department (Department_Id, Name, Manager, Building, Employee_Id)
14. Server (Server_Id, Active_clients, Status, Capacity, Processor_Id, Service_Id, Repair_Id)
15. Hardware (Hardware_Id, Date_of_purchase, Vendor, Storage_Capacity, Storage_type, Server_Id)

Note: Except 9th relation all the other relations were single valued and atomic and thereby conditions for 1NF are satisfied.

[2] Converting to 2NF

For 2NF of any relation, there are two necessities:

1. It should be in 1NF.
2. There should be no partial dependencies present.

   If the proper subset of a candidate key determines any non-prime attribute, it is a partial dependency.

1. Software

FD = { Software_id → Memory, Software_id → Date_of_Creation, Software_id →

Language_used }

The candidate key for this relation is **Software_id**. **It is**

already in 2NF form.

As we can see that Software_id is a single attribute which is a candidate key. So, there is only 1 proper subset of Software_id that is null set which cannot determine anything. So, it does not have any partial dependencies. Hence, Software is already in 2NF form.

2. Processor

FD = { Processor_Id → Status, Processor_Id → Speed, Processor_Id → Manufacturer, Processor_Id → Memory }

The candidate key for this relation is **Processor_Id**. **It is**

already in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

3. Service

FD = { Service_id → Type }

The candidate key for this relation is **Service_id**. **It is**

already in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

4. Web_Users

FD = { Web_User_Id → Username, Web_User_Id → Last_active, Web_User_Id → Location, Web_User_Id →

Service_Id, Username → Location }

The candidate key for this relation is **Web_User_Id**. **It is**

already in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

5. Cloud_Users

FD = { Cloud_User_Id → Username, Cloud_User_Id → Last_modified, Cloud_User_Id → Location, Cloud_User_Id → Service_Id, Cloud_User_Id → Storage_allocated, Username → Location }

The candidate key for this relation is **Cloud_User_Id**. **It is**

already in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

6. Cloud_Plan

FD = { Cloud_Subscription_Type → Cloud_User_Id}

The candidate key for this relation is **Cloud_Subscription_Type**. **It is already**

in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

7. Employee

FD = { Employee_Id → F_name, Employee_Id → M_name, Employee_Id → L_name, Employee_Id → D.O.B, Employee_Id → Salary, Employee_Id → Joining_date, Employee_Id → Status, Employee_Id → Gender }

The candidate key for this relation is **Employee_Id**. **It is already in 2NF form.**

As there are no partial dependencies present, it is already in 2NF form.

8. Repair

FD = { Repair_Id → Type, Repair_Id → Start_date, Repair_Id → Finish_date }

The candidate key for this relation is **Repair_Id**. **It is already in 2NF form.**

As there are no partial dependencies present, it is already in 2NF form.

9. Web_Plan

FD = { Web_Subscription_Type → Web_User_Id}

The candidate key for this relation is **Web_Subscription_Type**. **It is already in 2NF form.**

As there are no partial dependencies present, it is already in 2NF form.

10. Department

FD = { Department_Id → Name, Department_Id → Manager, Department_Id → Building, Department_Id → Employee_Id, Name → Manager, Name → Building, Employee_Id → Name}

The candidate key for this relation is **Department_Id**. **It is already in 2NF form.**

As there are no partial dependencies present, it is already in 2NF form.

11. Server

FD = { Server_Id → Active_clients, Server_Id → Status, Server_Id → Capacity, Server_Id → Processor_Id, Server_Id → Service_Id, Server_Id → Repair_Id, Service_Id → Status, Repair_Id → Status }

The candidate key for this relation is **Server_Id**. **It is**

already in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

12. Hardware

FD = { Hardware_Id → Date_of_purchase, Hardware_Id → Vendor, Hardware_Id → Storage_Capacity, Hardware_Id → Storage_type, Hardware_Id → Server_Id Server_Id → Storage_Capacity, Server_Id → Storage_type, Server_Id → Vendor }

The candidate key for this relation is **Hardware_Id**. **It is**

already in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

13. Software_Update

FD = { Update_Id → Memory, Update_Id → Update_date, Update_Id → Software_Id }

The candidate key for this relation is **Update_Id**. **It is**

already in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

14. Charge

FD = {Service_Id ⟶ Price, Service_Id ⟶ Idate} The candidate key for this relation is **Service_Id**.

It is already in 2NF form.

As there are no partial dependencies present, it is already in 2NF form.

[3] Converting to 3NF

For 3NF of any relation, there are two necessities:

1. It should be in 1NF and 2NF.
2. No non-PK attribute is transitively dependent on the PK

1. Software

FD = { Software_id → Memory, Software_id → Date_of_Creation, Software_id →

Language_used }

The candidate key for this relation is **Software_id**. **It is**

already in 3NF form.

There are no transitive dependencies present.

2. Processor

FD = { Processor_Id → Status, Processor_Id → Speed, Processor_Id → Manufacturer, Processor_Id → Memory }

The candidate key for this relation is **Processor_Id**. **It is**

already in 3NF form.

There are no transitive dependencies present.

3. Service

FD = { Service_id → Type }

The candidate key for this relation is **Service_id**.

It is already in 3NF form.

There are no transitive dependencies present.

4. Web_Users

FD = { Web_User_Id → Username, Web_User_Id → Last_active, Web_User_Id → Location, Web_User_Id →

Service_Id, Username → Location }

The candidate key for this relation is **Web_User_Id**.

Here, Web_User_Id → Username, Web_User_Id → Location makes a transitive functional dependency of Username → Location troublesome. So, we removed that in the final list of functional dependencies.

After removing the transitive dependency, it gets converted into 3NF form

5. Cloud_Users

FD = { Cloud_User_Id → Username, Cloud_User_Id → Last_modified, Cloud_User_Id → Location, Cloud_User_Id → Service_Id, Cloud_User_Id → Storage_allocated, Username → Location }

The candidate key for this relation is **Cloud_User_Id**.

Here, Cloud_User_Id → Username, Cloud_User_Id → Location makes a transitive functional dependency of Username → Location troublesome. So, we removed that in the final list of functional dependencies.

After removing the transitive dependency, it gets converted into 3NF form

6. Cloud_Plan

FD = { Cloud_Subscription_Type → Cloud_User_Id}

The candidate key for this relation is **Cloud_Subscription_Type**. **It is already in 3NF form.**

There are no transitive dependencies present.

7. Employee

FD = { Employee_Id → F_name, Employee_Id → M_name,

Employee_Id → L_name, Employee_Id → D.O.B,

Employee_Id → Salary, Employee_Id → Joining_date,

Employee_Id → Status, Employee_Id → Gender }

The candidate key for this relation is **Employee_Id**.

It is already in 3NF form.

There are no transitive dependencies present.

8. Repair

FD = { Repair_Id → Type, Repair_Id → Start_date, Repair_Id → Finish_date }

The candidate key for this relation is **Repair_Id**. **It is**

already in 3NF form.

There are no transitive dependencies present.

9. Web_Plan

FD = { Web_Subscription_Type → Web_User_Id}

The candidate key for this relation is **Web_Subscription_Type**. **It is already**

in 3NF form.

There are no transitive dependencies present.

10. Department

FD = { Department_Id → Name, Department_Id → Manager, Department_Id → Building, Department_Id →

Employee_Id, Name → Manager, Name → Building, Employee_Id → Name} The candidate key for this relation is

Department_Id.

Here, there were 3 transitive functional dependencies that caused trouble.

1. Name → Manager
2. Name → Building
3. Employee_Id → Name

So, we removed that in the final list of functional dependencies.

After removing the transitive dependency, it gets converted into 3NF form.

11. Server

FD = { Server_Id → Active_clients, Server_Id → Status, Server_Id → Capacity, Server_Id →

Processor_Id, Server_Id → Service_Id, Server_Id → Repair_Id, Service_Id → Status, Repair_Id →

Status }

The candidate key for this relation is **Server_Id**.

Here, there were 2 transitive functional dependencies that caused trouble.

1. Service_Id → Status
2. Repair_Id → Status

So, we removed that in the final list of functional dependencies.

After removing the transitive dependency, it gets converted into 3NF form.

12. Hardware

FD = { Hardware_Id → Date_of_purchase, Hardware_Id → Vendor,

Hardware_Id → Storage_Capacity, Hardware_Id → Storage_type, Hardware_Id → Server_Id Server_Id →

Storage_Capacity, Server_Id → Storage_type, Server_Id → Vendor }

The candidate key for this relation is **Hardware_Id**.

Here, there were 3 transitive functional dependencies that caused trouble.

1. Server_Id → Storage_Capacity
2. Server_Id → Storage_type
3. Server_Id → Vendor

So, we removed that in the final list of functional dependencies.

After removing the transitive dependency, it gets converted into 3NF form

13. Software_Update

FD = { Update_Id → Memory, Update_Id → Update_date, Update_Id → Software_Id }

The candidate key for this relation is **Update_Id**. **It is**

already in 3NF form.

There are no transitive dependencies present.

14. Charge

FD = {Service_Id ⟶ Price, Service_Id ⟶ Location} The

candidate key for this relation is **Service_Id**.

It is already in 3NF form.

There are no transitive dependencies present.

List of redundancies removed in the process of 2NF

- Username → Location
- Name → Manager
- Name → Building
- Employee_Id → Name
- Service_Id → Status
- Repair_Id → Status
- Server_Id → Storage_Capacity
- Server_Id → Storage_type
- Server_Id → Vendor

[4] Converting to BCNF

For BCNF of any relation, the necessity is:

1. A relation is in BCNF if and only if only determinants are candidatekeys

1. Software

FD = { Software_id → Memory, Software_id → Date_of_Creation, Software_id →

Language_used }

The candidate key for this relation is **Software_id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

2. Processor

FD = { Processor_Id → Status, Processor_Id → Speed, Processor_Id → Manufacturer, Processor_Id → Memory }

The candidate key for this relation is **Processor_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

3. Service

FD = { Service_id → Type }

The candidate key for this relation is **Service_id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

4. Web_Users

FD = { Web_User_Id → Username, Web_User_Id → Last_active, Web_User_Id → Location, Web_User_Id →

Service_Id }

The candidate key for this relation is **Web_User_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

5. Cloud_Users

FD = { Cloud_User_Id → Username, Cloud_User_Id → Last_modified, Cloud_User_Id → Location,

Cloud_User_Id → Service_Id, Cloud_User_Id → Storage_allocated, Username → Location }

The candidate key for this relation is **Cloud_User_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

6. Cloud_Plan

FD = { Cloud_Subscription_Type → Cloud_User_Id}

The candidate key for this relation is **Cloud_Subscription_Type**. **It is already**

in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

7. Employee

FD = { Employee_Id → F_name, Employee_Id → M_name, Employee_Id → L_name, Employee_Id →

D.O.B, Employee_Id → Salary, Employee_Id → Joining_date, Employee_Id → Status, Employee_Id →

Gender }

The candidate key for this relation is **Employee_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

8. Repair

FD = { Repair_Id → Type, Repair_Id → Start_date, Repair_Id → Finish_date }

The candidate key for this relation is **Repair_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

9. Web_Plan

FD = { Web_Subscription_Type → Web_User_Id}

The candidate key for this relation is **Web_Subscription_Type**. **It is already**

in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

10. Department

FD = { Department_Id → Name, Department_Id → Manager, Department_Id → Building, Department_Id → Employee_Id}

The candidate key for this relation is **Department_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

11. Server

FD = { Server_Id → Active_clients, Server_Id → Status, Server_Id → Capacity, Server_Id → Processor_Id, Server_Id → Service_Id, Server_Id → Repair_Id, Service_Id → Status, Repair_Id → Status }

The candidate key for this relation is **Server_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

12. Hardware

FD = { Hardware_Id → Date_of_purchase, Hardware_Id → Vendor, Hardware_Id → Storage_Capacity, Hardware_Id → Storage_type, Hardware_Id → Server_Id Server_Id → Storage_Capacity, Server_Id → Storage_type, Server_Id → Vendor }

The candidate key for this relation is **Hardware_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

13. Software_Update

FD = { Update_Id → Memory, Update_Id → Update_date, Update_Id → Software_Id }

The candidate key for this relation is **Update_Id**. **It is**

already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

14. Charge

FD = {Service_Id ⟶ Price, Service_Id ⟶ Location} The candidate key for this relation is **Service_Id**.

It is already in BCNF form.

As there is only 1 Candidate key, 3NF and BCNF are equivalent.

Section6: Final Functional Dependencies

Final Functional Dependencies

1. **Software**

   Software_id ⟶ Memory

   Software_id ⟶ Date_of_Creation

   Software_id ⟶ Language_used

2. **Processor**

   Processor_Id ⟶ Status

   Processor_Id ⟶ Speed

   Processor_Id ⟶ Manufacturer

   Processor_Id ⟶ Memory

3. **Service**

   Service_id ⟶ Type

4. **Web_Users**

   Web_User_Id ⟶ Username

   Web_User_Id ⟶ Last_active

   Web_User_Id ⟶ Location

   Web_User_Id ⟶ Service_Id Username

   ⟶ Location

5. **Cloud_Users**

   Cloud_User_Id ⟶ Username

   Cloud_User_Id ⟶ Last_modified

   Cloud_User_Id ⟶ Storage_allocated

   Cloud_User_Id ⟶ Location

   Cloud_User_Id ⟶ Service_Id

   Username ->Location

6. **Cloud_Plan**

   Cloud_Subscription_Type ⟶ Cloud_User_Id

7. Employee

Employee_Id ⟶ F_name

Employee_Id ⟶ M_name

Employee_Id ⟶ L_name

Employee_Id ⟶ D.O.B

Employee_Id ⟶ Salary

Employee_Id ⟶ Joining_date

Employee_Id ⟶ Status

Employee_Id ⟶ Gender

8. Repair

Repair_Id ⟶ Type Repair_Id

⟶ Start_date Repair_Id ⟶ Finish_date

9. Web_Plan

Web_Subscription_Type ⟶ Web_User_Id

10. Department

Department_Id ⟶ Name

Department_Id ⟶ Manager

Department_Id ⟶ Building

Department_Id ⟶ Employee_Id

Name ⟶ Manager

Name ⟶ Building

Employee_Id ⟶ Name

11. Server

Server_Id ⟶ Active_clients

Server_Id ⟶ Status

Server_Id ⟶ Capacity

Server_Id ⟶ Processor_Id

Server_Id ⟶ Service_Id

Server_Id ⟶ Repair_Id

Service_Id ⟶ Status

Repair_Id ⟶ Status

12. Hardware

Hardware_Id ⟶ Date_of_purchase

Hardware_Id ⟶ Vendor

Hardware_Id ⟶ Storage_Capacity

Hardware_Id ⟶ Storage_type

Hardware_Id ⟶ Server_Id

Server_Id ⟶ Storage_Capacity

Server_Id ⟶ Storage_Type

Server_Id ⟶ Vendor

13. Software_Update

Update_Id ⟶ Memory

Update_Id ⟶ Update_date

Update_Id ⟶ Software_Id

14. Charge

Service_Id ⟶ Price

Service_Id ⟶ Location

Section7: Final DDL Scripts with Insert statements

DDL SCRIPTS

➢ Software

```
CREATE TABLE Software(
Memory INT NOT NULL,
Date_of_creation DATE, Language_used VARCHAR, Software_id INT NOT NULL, PRIMARY KEY (Software_id)
);

INSERT INTO Software (Software_id,Memory,date_of_creation,Language_used)
VALUES
  (1000,0,'2022-02-22','PYTHON'),
  (1001,2,'2021-05-11','JAVA'),
  (1002,4,'2021-09-19','Go'),
  (1003,6,'2022-07-10','PYTHON'),
  (1004,8,'2021-01-02','Go'),
  (1005,10,'2021-05-17','PYTHON'),
  (1006,12,'2021-08-24','C'),
  (1007,14,'2021-08-19','Kotlin'),
  (1008,16,'2021-06-15','Kotlin'),
  (1009,18,'2021-08-22','C');
```

➢ Processor

```
CREATE TABLE Processor(
Status VARCHAR, Speed VARCHAR,
Processor_id INT NOT NULL, Manufacturer VARCHAR, Memory INT NOT NULL,
PRIMARY KEY (Processor_id)
);

INSERT INTO Processor (Status,Speed,Processor_id,Manufacturer,Memory)
VALUES
  ('Repair',3.6,10000,'Magna LLC',0),
  ('Repair',2.6,10001,'Aliquam Vulputate Incorporated',2),
  ('Repair',1.6,10002,'Ante Lectus Corporation',4),
  ('Repair',2.6,10003,'Mattis Ltd',6),
  ('Working',3.6,10004,'Amet Ante Ltd',8),
  ('Working',3.6,10005,'Semper Rutrum LLC',10),
  ('Working',3.6,10006,'Pretium Limited',12),
  ('Repair',2.6,10007,'Sed Eu Nibh Industries',14),
  ('Repair',2.6,10008,'Dui Fusce Ltd',16),
  ('Repair',1.6,10009,'Pellentesque Habitant Inc.',18);
```

➢ Repair

```
CREATE TABLE Repair(
Repair_Id INT NOT NULL,
Type VARCHAR,
Start_date DATE,
End_date DATE,
PRIMARY KEY (Repair_Id)
);

INSERT INTO Repair (Repair_Id,Type,Start_date,End_date)
VALUES
  (0,'Maintenance','2009-05-27','2013-01-14'),
  (1,'Intensive','2009-07-01','2013-01-04'),
  (2,'Intensive','2011-07-28','2013-03-07'),
  (3,'Maintenance','2008-11-21','2013-04-04'),
```

```
 (4,'Maintenance','2009-04-14','2012-12-15'),
 (5,'Intensive','2010-08-14','2013-05-15'),
 (6,'Maintenance','2010-09-21','2013-01-19'),
 (7,'Intensive','2010-01-07','2013-07-26'),
 (8,'Intensive','2009-08-19','2013-10-21'),
 (9,'Intensive','2009-07-28','2013-10-25');
```

- **Employee**

```
CREATE TABLE Employee (
Employee_Id INT NOT NULL, F_name VARCHAR,
M_name VARCHAR,
L_name VARCHAR,
DOB DATE,
Salary VARCHAR, Joining_date DATE, Status VARCHAR,
Gender VARCHAR,
PRIMARY KEY (Employee_Id)
);

INSERT INTO Employee (Employee_Id,F_name,M_name,L_name,DOB,Salary,Joining_date,Status,Gender)
VALUES
 (100,'Quintessa','Hermione','Mayo','1991-09-24','$6,147','2015-04-18','Sales-head','Female'),
 (101,'Carson','Aileen','Little','1991-01-08','$9,858','2012-02-02','Marketing-head','Female'),
 (102,'Murphy','Alea','French','1978-01-09','$8,186','2015-02-25','Developer','Female'),
 (103,'Colette','Hammett','Palmer','1980-09-30','$8,853','2010-12-26','Sales-head','Female'),
 (104,'Hermione','Brock','Dickson','1992-06-20','$7,649','2019-06-04','SDE-III','Male'),
 (105,'Sylvia','Slade','Barr','1987-12-02','$5,978','2021-03-16','Developer','Male'),
 (106,'Venus','Hall','Benjamin','1993-12-17','$7,071','2020-12-25','Vice-President','Female'),
 (107,'Nissim','Roth','Donaldson','1991-08-13','$6,827','2017-06-22','Product-manager','Male'),
 (108,'Amelia','Wang','Orr','1976-08-10','$8,223','2015-04-07','Marketing-head','Female'),
 (109,'Jerome','Lara','Anthony','1989-07-31','$5,067','2017-08-13','Associate-Engineer','Female');
```

- **Software_Update**

```
CREATE TABLE Software_Update (
Memory INT NOT NULL,
Update_date DATE,
Update_id INT NOT NULL, Software_id INT NOT NULL, PRIMARY KEY (Update_id),
FOREIGN KEY (Software_id) REFERENCES Software (Software_id)
ON DELETE CASCADE
ON UPDATE CASCADE
);

INSERT INTO Software_Update (Memory,Update_date,Update_id,Software_id)
VALUES
 (1,'2021-11-01',100,1000),
 (2,'2021-09-02',101,1001),
 (3,'2022-09-27',102,1002),
 (4,'2021-07-10',103,1003),
 (5,'2022-08-10',104,1004),
 (6,'2021-03-13',105,1005),
 (7,'2022-08-22',106,1006),
 (8,'2021-08-14',107,1007),
 (9,'2021-02-24',108,1008),
 (10,'2021-10-23',109,1009);
```

- **Service**

```
CREATE TABLE Service (
Service_Id INT NOT NULL, Type VARCHAR,
PRIMARY KEY (Service_Id)
```

);

INSERT INTO Service (Service_Id,Type)
VALUES
  (0,'Web'),
  (1,'Web'),
  (2,'Web'),
  (3,'Web'),
  (4,'Web'),
  (5,'Web'),
  (6,'Web'),
  (7,'Web'),
  (8,'Web'),
  (9,'Web');

- **Charge**

CREATE TABLE Charge (
Price VARCHAR,
Location VARCHAR,
Service_Id INT NOT NULL, PRIMARY KEY (Price, Service_Id),
FOREIGN KEY (Service_Id) REFERENCES Service(Service_Id)
);

INSERT INTO Charge (Price,Location,Service_Id)
VALUES
  ('$79','Germany',0),
  ('$81','Pakistan',1),
  ('$54','Costa Rica',2),
  ('$78','China',3),
  ('$83','United States',4),
  ('$97','Indonesia',5),
  ('$65','Mexico',6),
  ('$82','Netherlands',7),
  ('$92','United States',8),
  ('$77','Ireland',9);

- **Cloud_Users**

CREATE TABLE Cloud_Users (
Cloud_User_Id INT NOT NULL, Username VARCHAR, Last_modified VARCHAR, Storage_allocated INT NOT NULL, Storage_Used INT NOT NULL, Location VARCHAR,
Service_Id INT NOT NULL, PRIMARY KEY (Cloud_User_Id),
FOREIGN KEY (Service_Id) REFERENCES Service(Service_Id)
);

INSERT INTO Cloud_Users (Cloud_User_Id,Username,Last_modified,Storage_allocated,Storage_Used,Location,Service_Id)
VALUES
  (100,'Molly','1:00 AM',100,54,'Chile',97),
  (101,'Gemma','11:33 AM',100,14,'Colombia',54),
  (102,'Vincent','7:14 AM',100,70,'Netherlands',65),
  (103,'Yardley','1:10 PM',100,76,'Italy',88),
  (104,'Rebekah','3:22 AM',100,26,'Vietnam',62),
  (105,'Tamara','12:30 AM',100,68,'Belgium',70),
  (106,'Ashely','8:39 AM',100,18,'Australia',51),
  (107,'Kane','4:21 AM',100,32,'Poland',89),
  (108,'Alvin','4:24 PM',100,67,'France',63),
  (109,'Linus','1:45 PM',100,27,'Canada',52);

- **Cloud_Plan**

```
CREATE TABLE Cloud_Plan (
Cloud_Subscription_Plan VARCHAR,
Cloud_User_Id INT,
PRIMARY KEY (Cloud_Subscription_Plan, Cloud_User_Id),
FOREIGN KEY (Cloud_User_Id) REFERENCES Cloud_Users(Cloud_User_Id)
ON DELETE CASCADE
ON UPDATE CASCADE
);

INSERT INTO Cloud_Plan (Cloud_Subscription_Plan,Cloud_User_Id)
VALUES
  ('Diamond',100),
  ('Diamond',101),
  ('Diamond',102),
  ('Platinum',103),
  ('Platinum',104),
  ('Diamond',105),
  ('Silver',106),
  ('Silver',107),
  ('Diamond',108),
  ('Silver',109);
```

- **Server**

```
CREATE TABLE Server (
Server_id INT NOT NULL, Active_clients INT NOT NULL, Status VARCHAR,
Capacity INT NOT NULL, Processor_id INT NOT NULL, Service_Id INT,
Repair_Id INT NOT NULL, PRIMARY KEY (Server_id),
FOREIGN KEY (Processor_id) REFERENCES Processor(Processor_id), FOREIGN KEY (Service_Id) REFERENCES Service(Service_Id),
FOREIGN KEY (Repair_Id) REFERENCES Repair(Repair_Id)
ON DELETE CASCADE
ON UPDATE CASCADE
);

INSERT INTO Server (Server_Id,Active_clients,Status,Capacity,Service_Id,Processor_Id,Repair_Id)
VALUES
  (1000,40726,'Repair',147,0,10000,0),
  (1001,50808,'Repair',111,1,10001,1),
  (1002,8647,'Working',163,2,10002,2),
  (1003,41370,'Working',144,3,10003,3),
  (1004,18267,'Working',114,4,10004,4),
  (1005,88689,'Working',154,5,10005,5),
  (1006,5827,'Repair',180,6,10006,6),
  (1007,21850,'Working',178,7,10007,7),
  (1008,345,'Working',182,8,10008,8),
  (1009,95441,'Repair',174,9,10009,9);
```

- **Web_Users**

```
CREATE TABLE Web_Users (
Web_User_Id INT NOT NULL, Usernames VARCHAR,
Last_active VARCHAR,
 Location VARCHAR,
Service_Id INT NOT NULL, PRIMARY KEY (Web_User_Id),
FOREIGN KEY (Service_Id) REFERENCES Service(Service_Id)
);

INSERT INTO Web_Users (Web_User_Id,Usernames,Last_active,Location,Service_Id)
VALUES
  (100,'Cara','1:22 AM','Ireland',0),
```

```
 (101,'Damian','10:24 AM','South Korea',1),
 (102,'Dean','12:41 AM','Mexico',2),
 (103,'Denton','2:09 PM','Sweden',3),
 (104,'Hayfa','8:04 PM','Netherlands',4),
 (105,'Mariam','2:16 AM','India',5),
 (106,'Rowan','3:26 AM','Netherlands',6),
 (107,'Allistair','10:18 AM','India',7),
 (108,'Hakeem','7:16 AM','Peru',8),
 (109,'Katell','4:09 AM','Italy',9);
```

- **Web_Plan**

```
CREATE TABLE Web_Plan(
Web_Subscription_Type VARCHAR,
Web_User_Id INT,
PRIMARY KEY (Web_Subscription_Type,Web_User_Id),
FOREIGN KEY (Web_User_Id) REFERENCES Web_Users(Web_User_Id)
ON DELETE CASCADE
ON UPDATE CASCADE
);

INSERT INTO Web_Plan (Web_Subscription_Type,Web_User_Id)
VALUES
 ('Gold',100),
 ('Diamond',101),
 ('Platinum',102),
 ('Diamond',103),
 ('Silver',104),
 ('Gold',105),
 ('Diamond',106),
 ('Silver',107),
 ('Gold',108),
 ('Platinum',109);
```

- **Access_To**

```
CREATE TABLE Access_To (
Server_id INT NOT NULL, Employee_Id INT NOT NULL,
PRIMARY KEY (Server_id, Employee_Id),
FOREIGN KEY (Server_id) REFERENCES Server(Server_id),  FOREIGN KEY (Employee_Id) REFERENCES Employee(Employee_Id)
ON DELETE CASCADE
ON UPDATE CASCADE
);

INSERT INTO Access_To (Server_id,Employee_Id)
VALUES
 (1000,100),
 (1001,101),
 (1002,102),
 (1003,103),
 (1004,104),
 (1005,105),
 (1006,106),
 (1007,107),
 (1008,108),
 (1009,109);
```

- **Employee_Contact_info**

```
CREATE TABLE Employee_Contact_info (
Contact_info VARCHAR, Employee_Id INT NOT NULL,
```

```
PRIMARY KEY (Contact_info, Employee_Id),
FOREIGN KEY (Employee_Id) REFERENCES Employee(Employee_Id)
ON DELETE CASCADE
ON UPDATE CASCADE
);

INSERT INTO Employee_Contact_info (Contact_info,Employee_Id)
VALUES
  ('033-672-6605',100),
  ('016-042-7999',101),
  ('046-368-3591',102),
  ('046-216-4158',103),
  ('049-558-2605',104),
  ('059-578-7542',105),
  ('043-275-4568',106),
  ('074-101-6821',107),
  ('035-066-4583',108),
  ('027-625-4377',109);
```

- **Hardware**

```
CREATE TABLE hardware (
Hardware_Id INT NOT NULL, Date_of_purchase DATE, Vendor VARCHAR,
Storage_capacity INT NOT NULL,
Storage_type VARCHAR,
Server_id INT,
PRIMARY KEY (Hardware_Id),
FOREIGN KEY (Server_id) REFERENCES Server(Server_id)
ON DELETE CASCADE
ON UPDATE CASCADE
);

INSERT INTO hardware (Hardware_Id,Date_of_Purchase,Vendor,Storage_capacity,Storage_type,Server_id)
VALUES
  (100,'2022-06-28','Risus Quis Corp.',243,'Cooling',1000),
  (101,'2022-03-30','Facilisis Magna Ltd',186,'Cooling',1001),
  (102,'2021-02-14','Elit Sed Incorporated',166,'Cooling',1002),
  (103,'2021-03-17','Mauris Vestibulum Industries',212,'Ultra-cooling',1003),
  (104,'2022-07-25','Nec Metus Facilisis Corp.',196,'Cooling',1004),
  (105,'2022-01-17','Maecenas Iaculis LLC',204,'Regular',1005),
  (106,'2021-06-24','Nonummy Fusce Limited',176,'Ultra-cooling',1006),
  (107,'2022-11-01','Eu Metus Incorporated',190,'Ultra-cooling',1007),
  (108,'2021-03-10','Auctor Quis Inc.',196,'Ultra-cooling',1008),
  (109,'2022-11-12','A Odio Semper PC',221,'Regular',1009);
```

- **SoftwareUsed**

```
CREATE TABLE SoftwareUsed (
Software_id INT NOT NULL, Server_id INT NOT NULL,
PRIMARY KEY (Software_id, Server_id),
FOREIGN KEY (Software_id) REFERENCES Software(Software_id), FOREIGN KEY (Server_id) REFERENCES Server(Server_id)
ON DELETE CASCADE
ON UPDATE CASCADE
);

INSERT INTO SoftwareUsed (Software_id,Server_id)
VALUES
  (1000,1000),
  (1001,1001),
  (1002,1002),
  (1003,1003),
```

```
(1004,1004),
(1005,1005),
(1006,1006),
(1007,1007),
(1008,1008),
(1009,1009);
```

- **Department**

```
CREATE TABLE Department (
Department_Id INT NOT NULL, Name VARCHAR,
Manager VARCHAR,
Building VARCHAR, Employee_Id INT NOT NULL,
PRIMARY KEY
(Department_Id),
FOREIGN KEY (Employee_Id) REFERENCES Employee(Employee_Id)
);

  INSERT INTO Department (Department_Id, Name, Manager, Building, Employee_Id)
VALUES
  (0, 'IT', 'Rhiannon', 'B0', 100),
  (1, 'HR', 'Stacy', 'B1', 101),
  (2, 'Finance', 'Darryl', 'B2', 102),
  (3, 'Sales', 'Merrill', 'B3', 103),
  (4, 'Marketing', 'Mariam', 'B4', 104),
  (5, 'Management', 'Todd', 'B5', 105),
  (6, 'R&D', 'Aurelia', 'B6', 106);
```

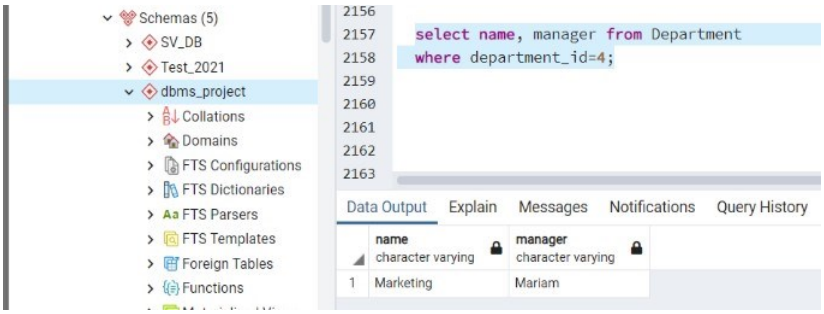
Section8: SQL Queries

SQL Queries

1. **Write a query to find name and manager from table Department where department_id is 4.**

   select name, manager from Department

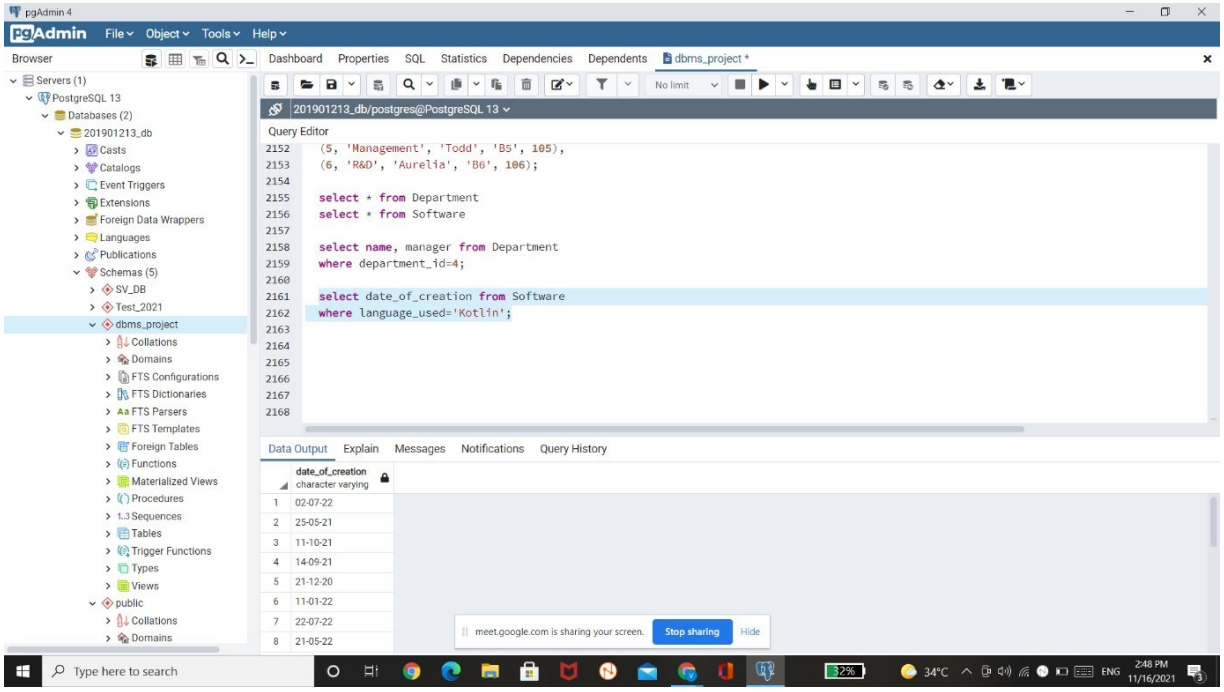
   where department_id=4; --1 (tuple)



2. **Write a query to find the creation date from table Software where language is Kotlin.**

   select date_of_creation from Software where

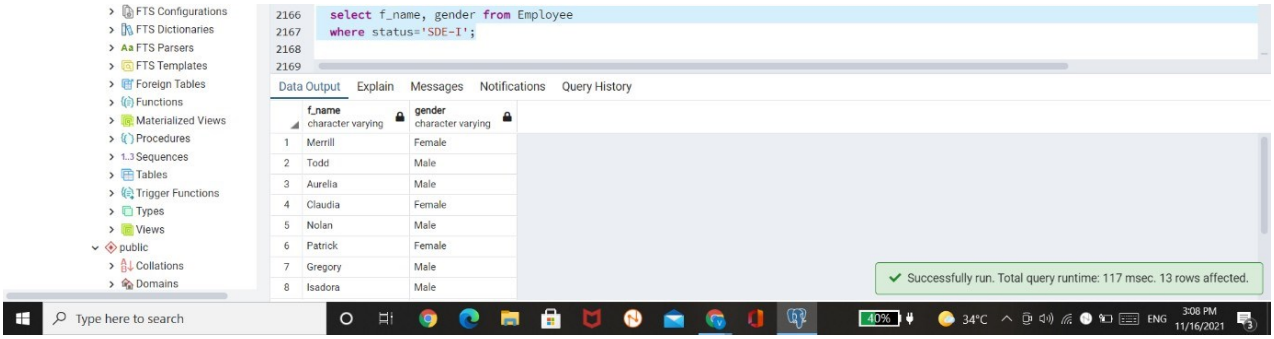
   language_used='Kotlin'; --23



3. **Write a query to find first name and gender from Table Employee where status is SDE-I.**

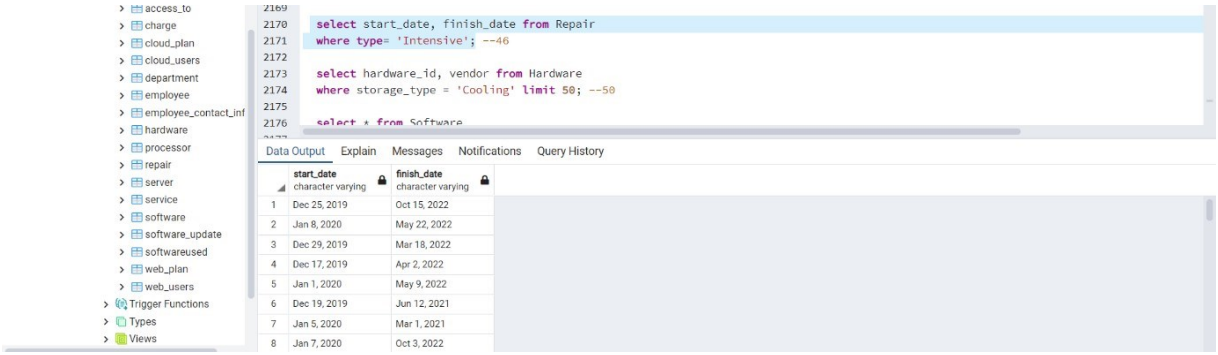
   select f_name,gender from Employee where

   status='SDE-I';--13



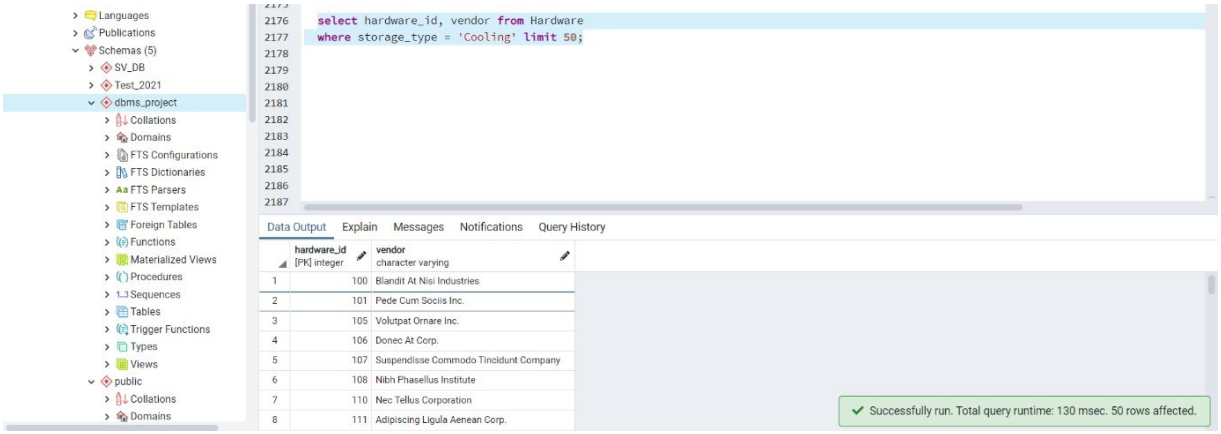
4. **Write a query to find start and finish date from table Repair where type is Intensive.**

```
select start_date, finish_date from Repair where
type='Intensive';--46
```



5. **Write a query to find first 50 hardware id and vendor name from table Hardware where type of storage is Cooling.**

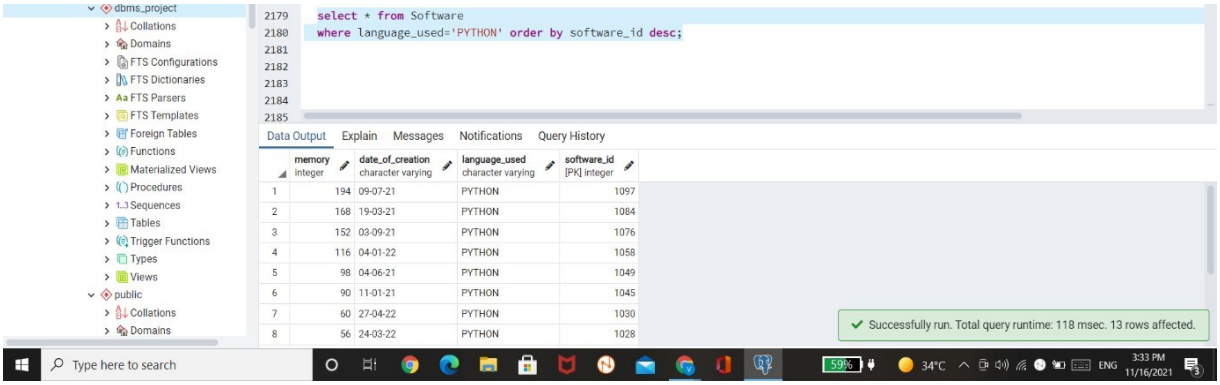
```
select hardware_id, vendor from Hardware where
storage_type = 'Cooling' limit 50;--50
```



6. **Write a query to find all tuples from table Software where language is PYTHON and order the tuples in descending order.**

select * from Software

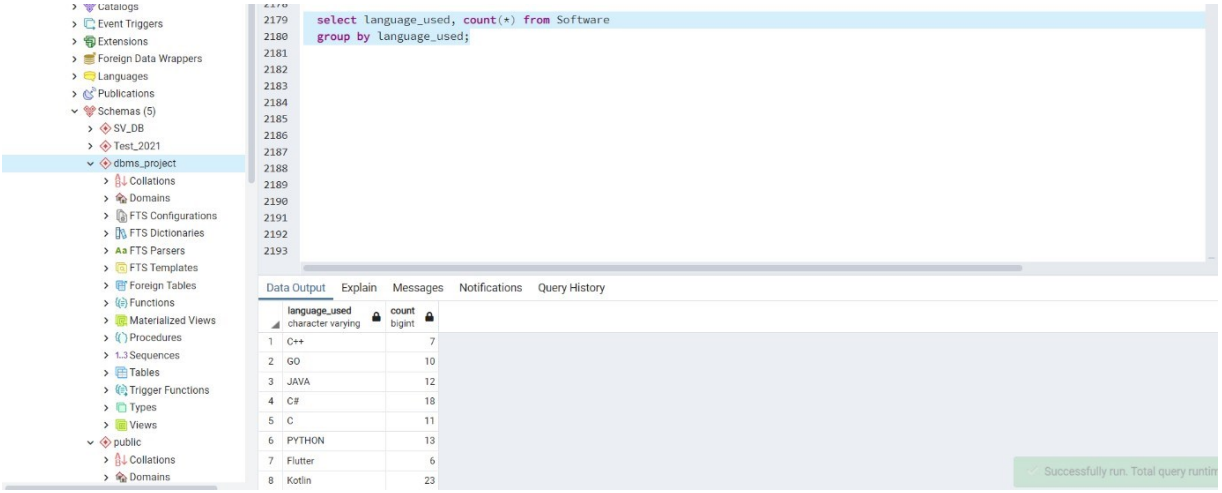
where language_used='PYTHON' order by software_id desc; --13



7. Write a query to find language used and count the number of times a language appears in a table Software.

select language_used, count(*) from Software group by

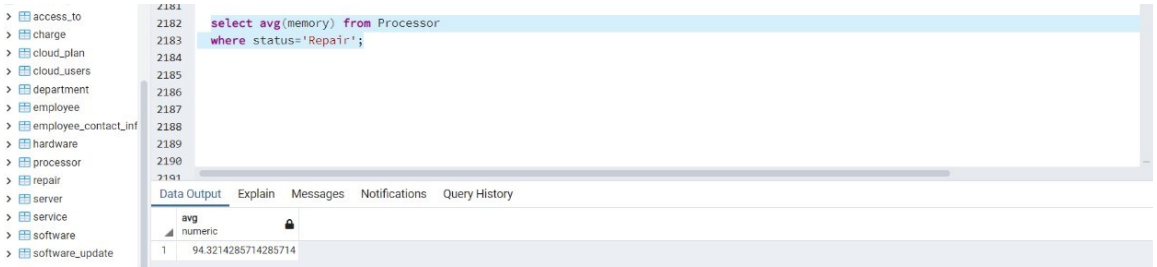
language_used; --8



8. Write a query to find the average value of attribute memory from table Processor where status is Repair.

select avg(memory) from Processor where

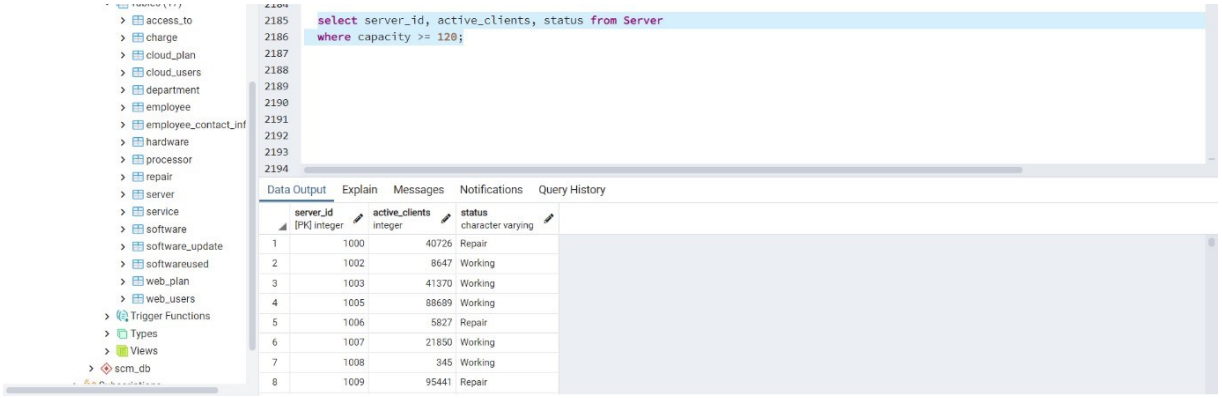
status='Repair'; --1



9. Write a query to find server id, active clients and status from table Server where capacity is greater or equal to 120.

select server_id, active_clients, status from Server where

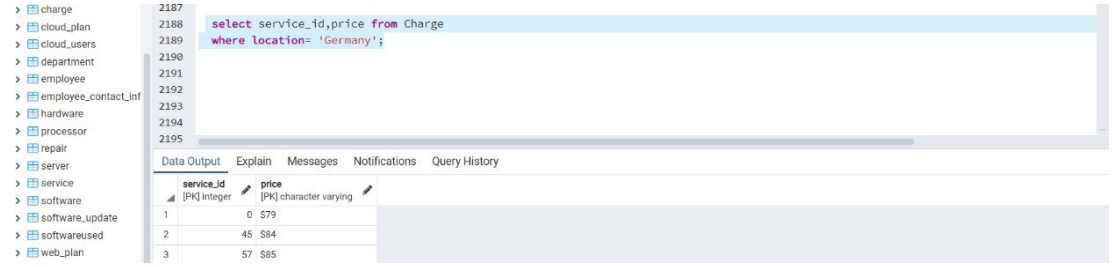
capacity >= 120; --83



10. Write a query to find server id and price from table Charge where location is Germany.

select service_id,price from Charge. where

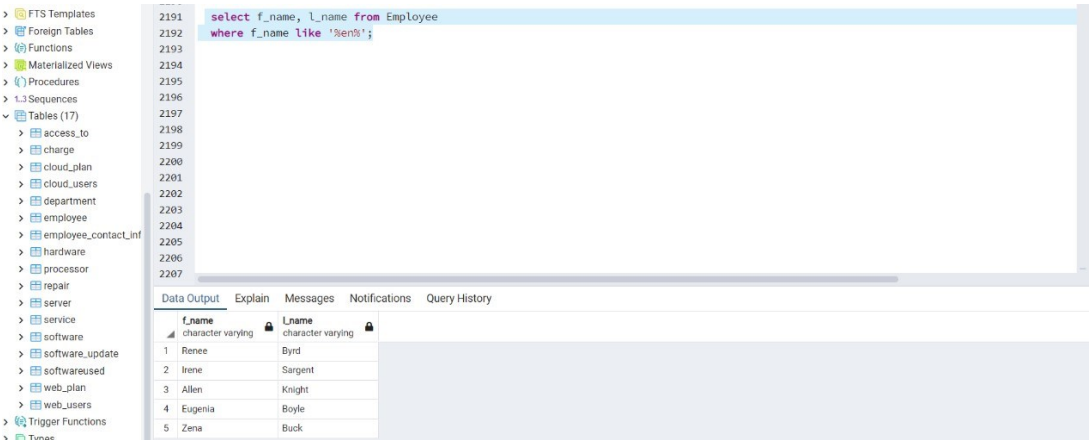
location='Germany'; --3



11. Write a query to find first and last name of employee where first name has a substring 'en'.

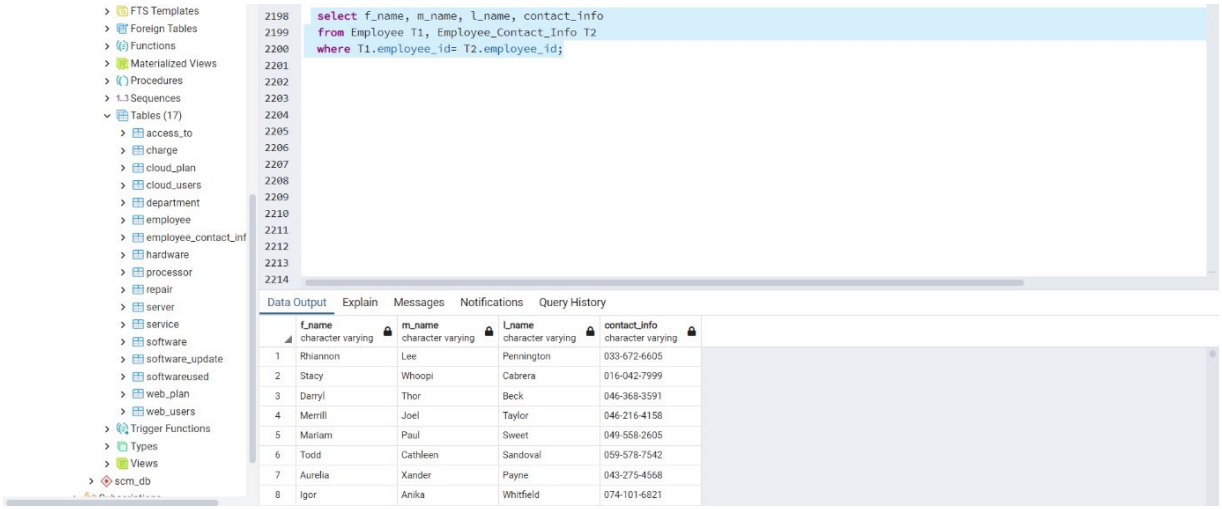
select f_name,l_name from Employee where

f_name like '%en%'; --5



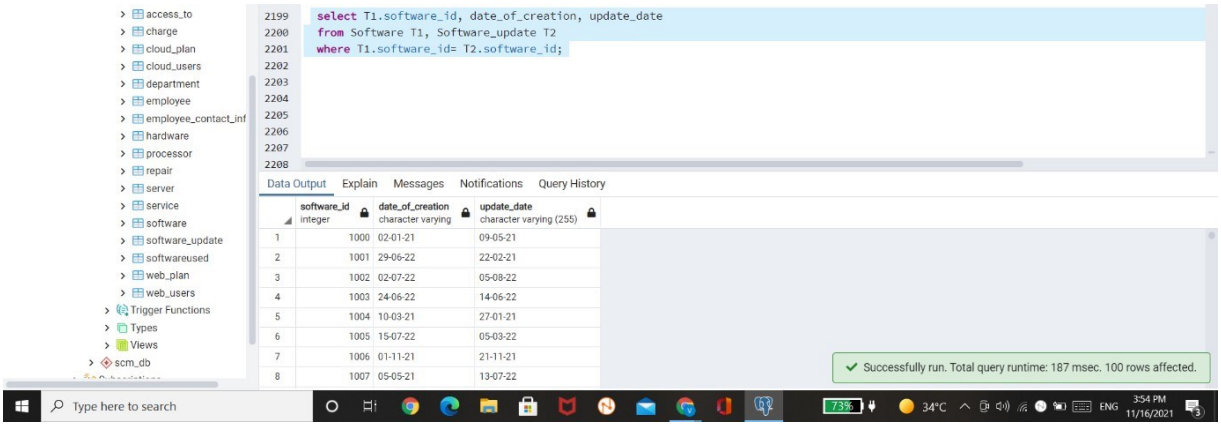
12. Write a query to find first, middle, last name and contact information of employees.

```
select f_name, m_name, l_name, contact_info
from Employee T1, Employee_Contact_Info T2
where T1.employee_id= T2.employee_id; --100
```



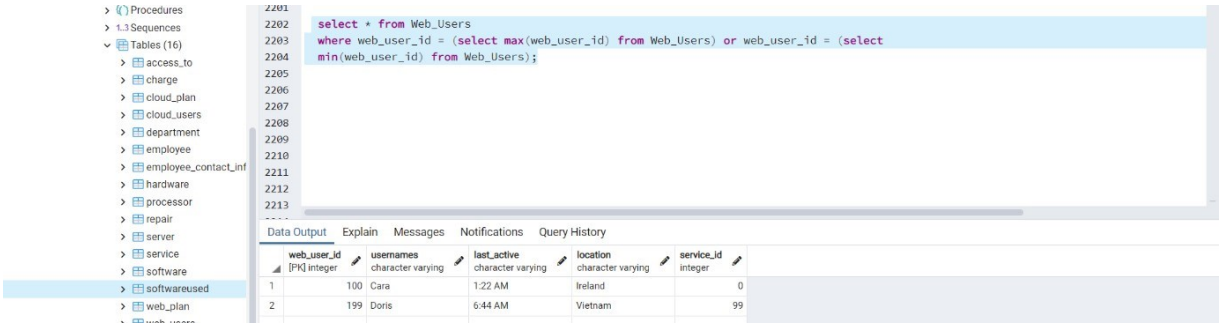
13. Write a query to find software if, date of creation and software update date.

```
select T1.software_id, date_of_creation, update_date from
Software T1, Software_update T2
where T1.software_id= T2.software_id; --100
```



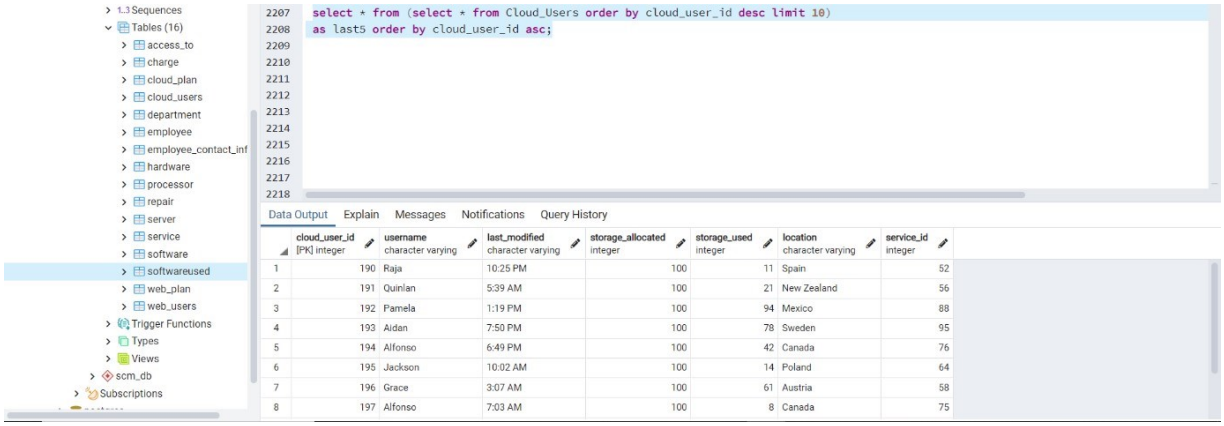
14. Write a query to find first and last record of Web users table.

```
select * from Web_Users
where web_user_id =(select max(web_user_id) from Web_Users) orweb_user_id =(select min(web_user_id) from
Web_Users);--2
```



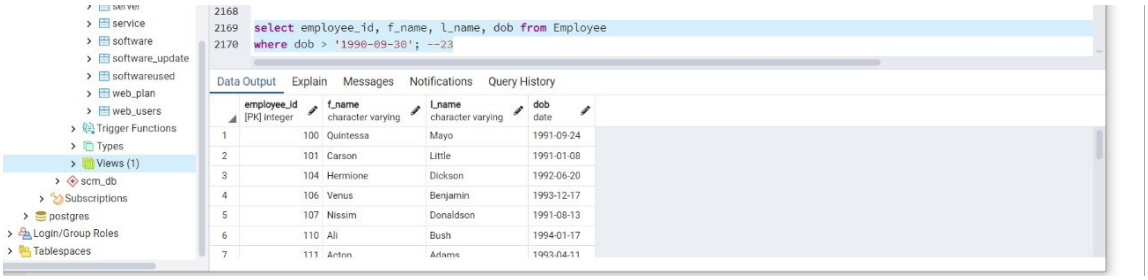
15. Write a query to find last 5 records of table Cloud users.

select * from (select * from Cloud_Users order by cloud_user_id desc limit 10) as last5 order by cloud_user_id asc; --10



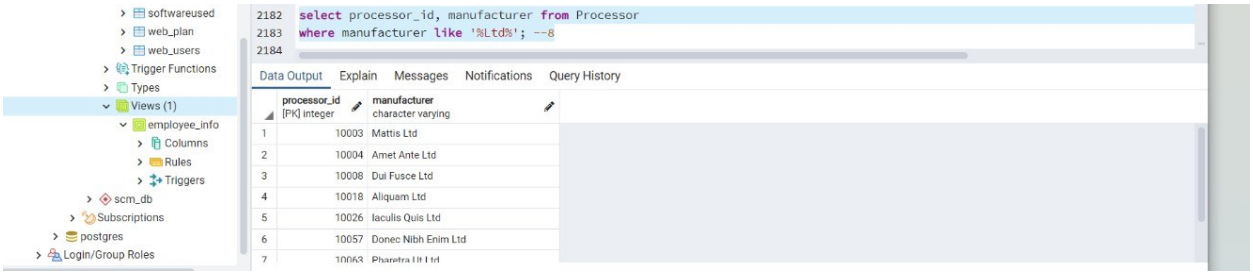
16. Write a query to find employee id, first name, last name and date of birth of Employees whose date of birth is greater than '1990-09-30'.

select employee_id, f_name, l_name, dob from Employee where dob >'1990-09-30'; --23



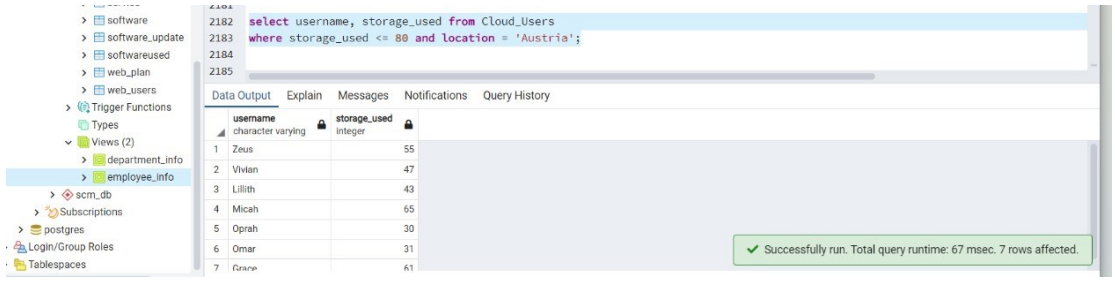
17. Write a query to retrieve processor id and manufacturer from Processor for all the manufacturers containing 'Ltd'.

select processor_id, manufacturer from Processor where manufacturer like '%Ltd%'; --8



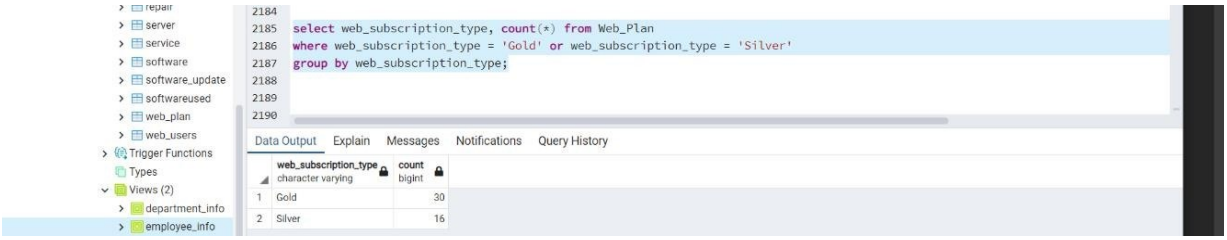
18. Write a query to retrieve username and storage used, of all the cloud users in Austria with storage usage less than or equal to 80.

select username, storage_used from Cloud_Users where storage_used <= 80 and location = 'Austria'; --7



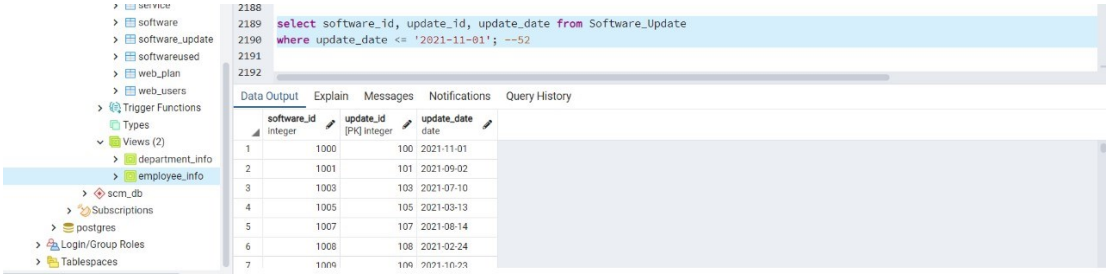
19. Write a query to find total number of records for 'Gold' and 'Silver' subscription type.

select web_subscription_type, count(*) from Web_Plan
where web_subscription_type = 'Gold' or web_subscription_type = 'Silver' group by
web_subscription_type;--2



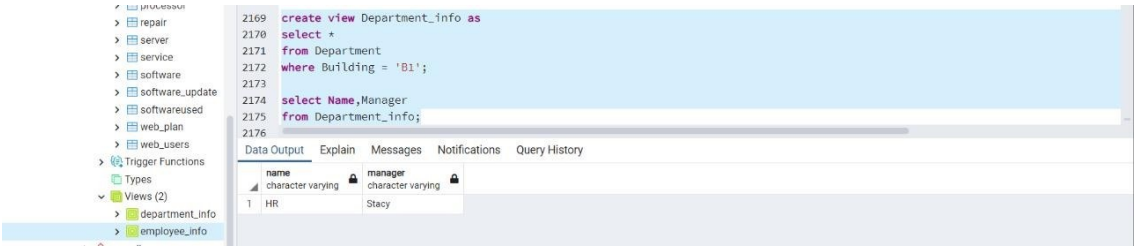
20. Write a query to retrieve software id, update id and update date from Software_Update table where update date is less than or equal to '2021-11-01'.

select software_id, update_id, update_date from Software_Update where update_date
<='2021-11-01';--52



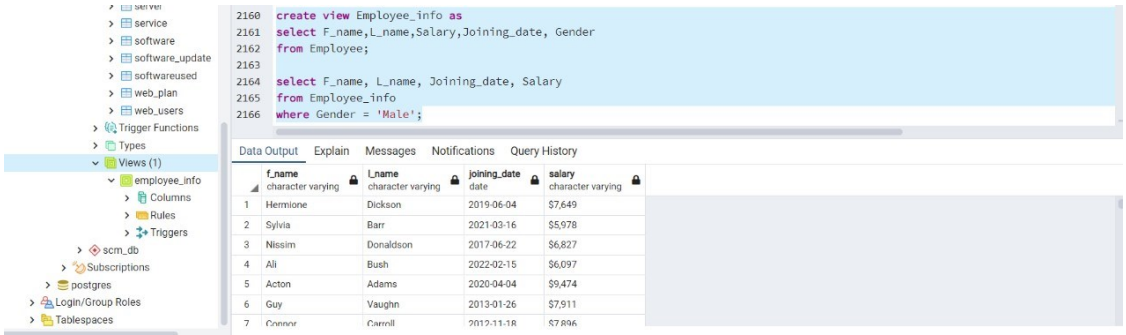
21. Write a query to display name and manager from Department where Building = 'B1'.

```
create view Department_info as select
*
from Department where
Building = 'B1';

select Name,Manager from
Department_info; --1
```



22. Write a query to retrieve first name, last name, joining date and salary of all male employees.

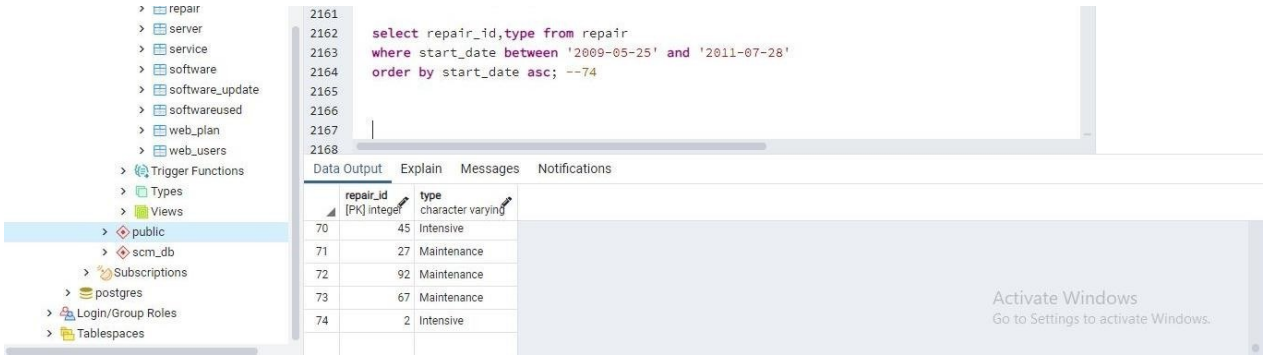
```
create view Employee_info as

select F_name,L_name,Salary,Joining_date, Gender from

Employee;


select F_name,L_name, Joining_date, Salary from

Employee_info

where Gender='Male'; --53
```



23. Write query to find repair_id and its type of all repairs that have occurred between 25-05-2009 to 28-07-2011, sorted by start_date.

```
select repair_id,type from repair

where start_date between '2009-05-25' and '2011-07-28'
```

order by start_date asc; --74 Tuples



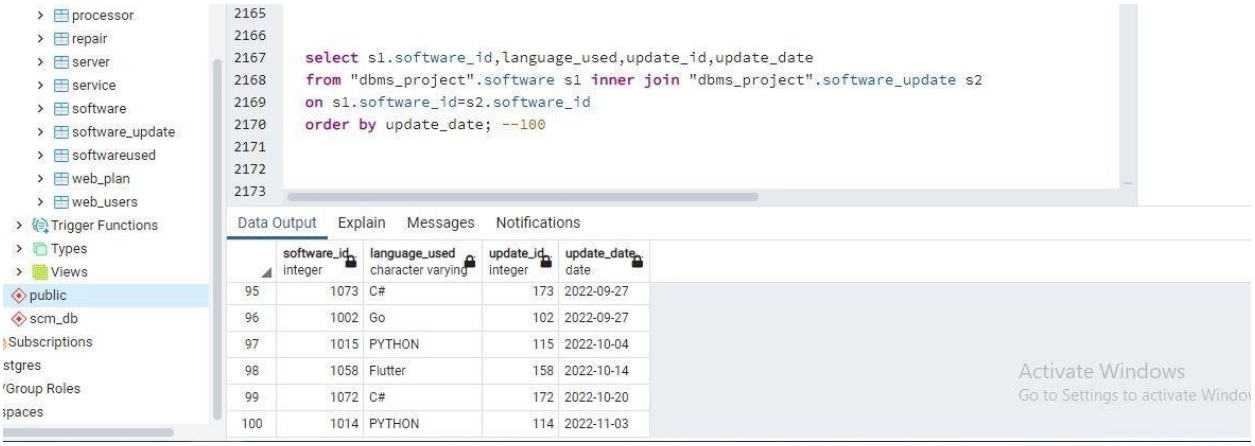
24. Write a query to find language_used in particular software and it last update_date.

select s1.software_id,language_used,update_id,update_date

from "dbms_project".software s1 inner join "dbms_project".software_update s2 on

s1.software_id=s2.software_id

order by update_date; --100

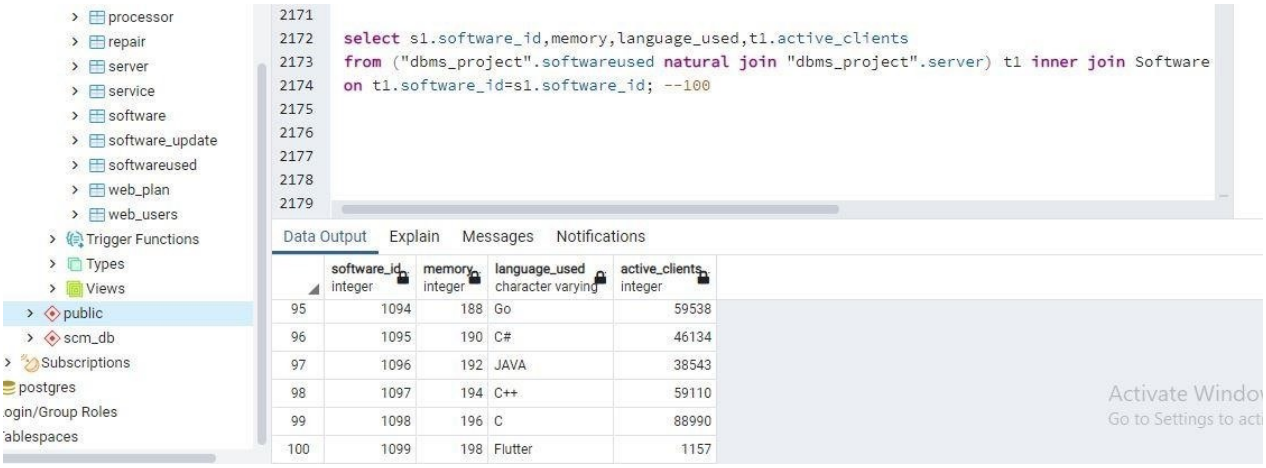


25. Write a query to find software_id, its corresponding language_used to deploy that software,server on which software runs and number of active_clients on that server.

select s1.software_id,memory,language_used,t1.active_clients

from ("dbms_project".softwareused natural join "dbms_project".server) t1 inner join Software s1 on

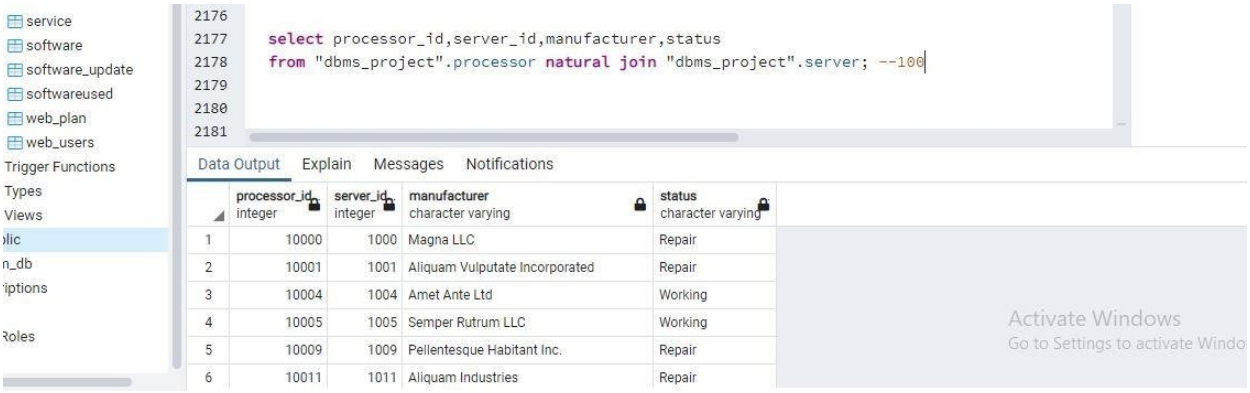
t1.software_id=s1.software_id; --100



26. Write a query to find processor_id corresponding to each server_id, manufacturer of that processor as well as current status of server.

select processor_id,server_id,manufacturer,status

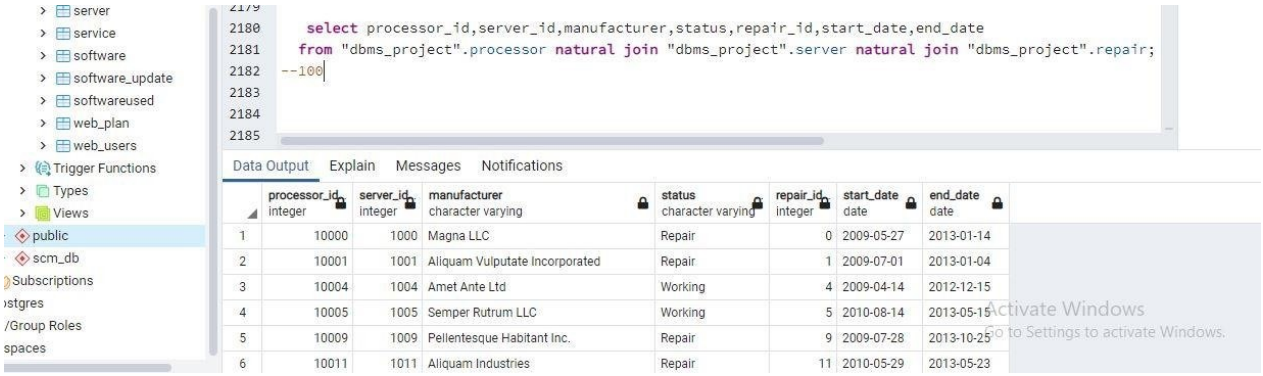
from "dbms_project".processor natural join "dbms_project".server; --100



27. Write a query to display server_id, its status, its corresponsing processor_id, manufacturer of that processor, its corresponding repair_id and other details of repair.

select processor_id,server_id,manufacturer,status,repair_id,start_date,end_date

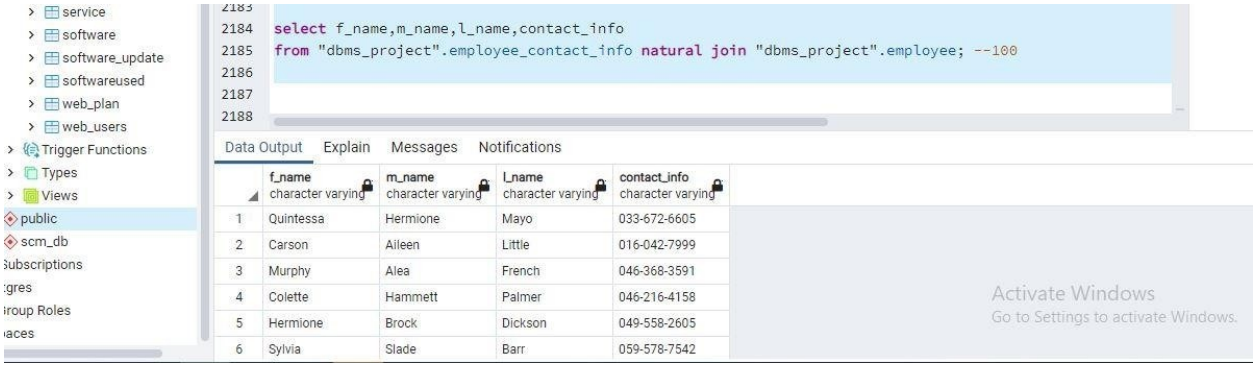
from "dbms_project".processor natural join "dbms_project".server natural join "dbms_project". Repair; --100



28. Write a query to fetch employee name and their contact info.

select f_name,m_name,l_name,contact_info

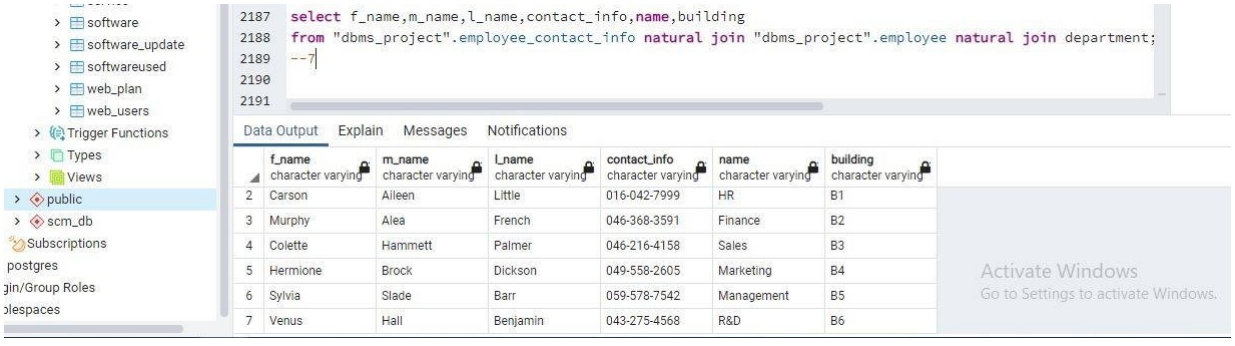
from "dbms_project".employee_contact_info natural join "dbms_project".employee; --100



29. Write a query to fetch full names and their contact-info of each manager present in department relation.

select f_name,m_name,l_name,contact_info,name,building

from "dbms_project".employee_contact_info natural join "dbms_project".employee natural join department; --7

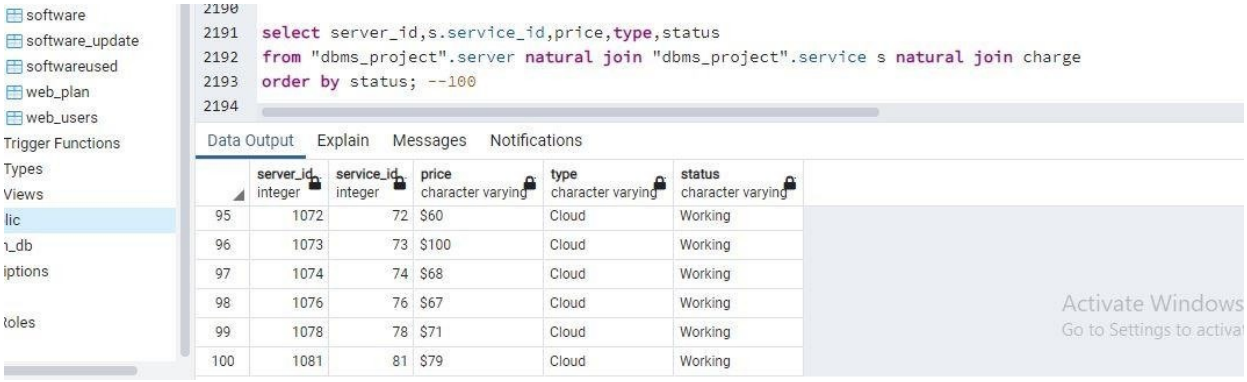


30. Write a query to fetch all details of each service corresponding to each server, order by its server status.

select server_id,s.service_id,price,type,status

from "dbms_project".server naturaljoin "dbms_project".service s naturaljoin charge order by status;--

100

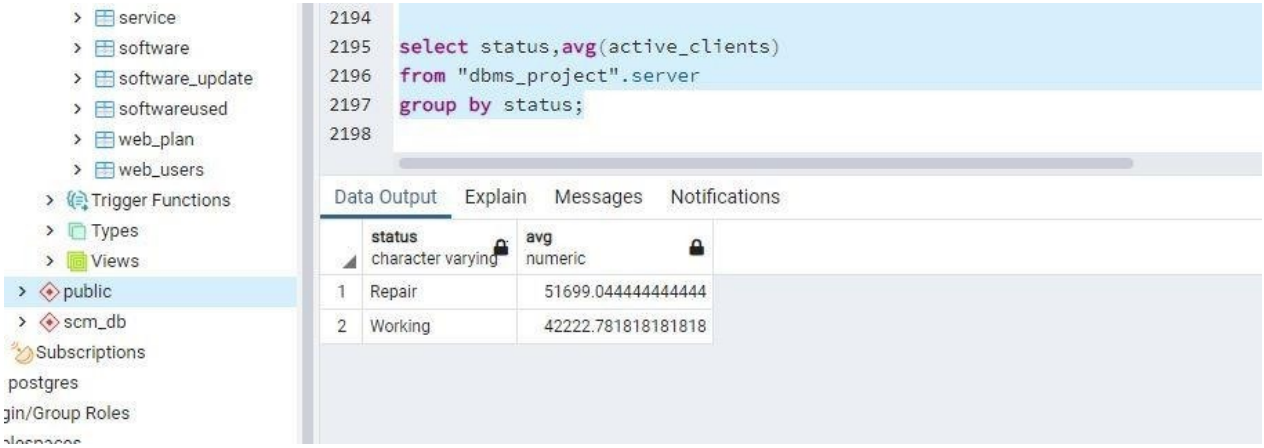


31. Write a query to find average number of active_clients in server under repair as well as under working.

select status,avg(active_clients) from

"dbms_project".server group by

status; --2



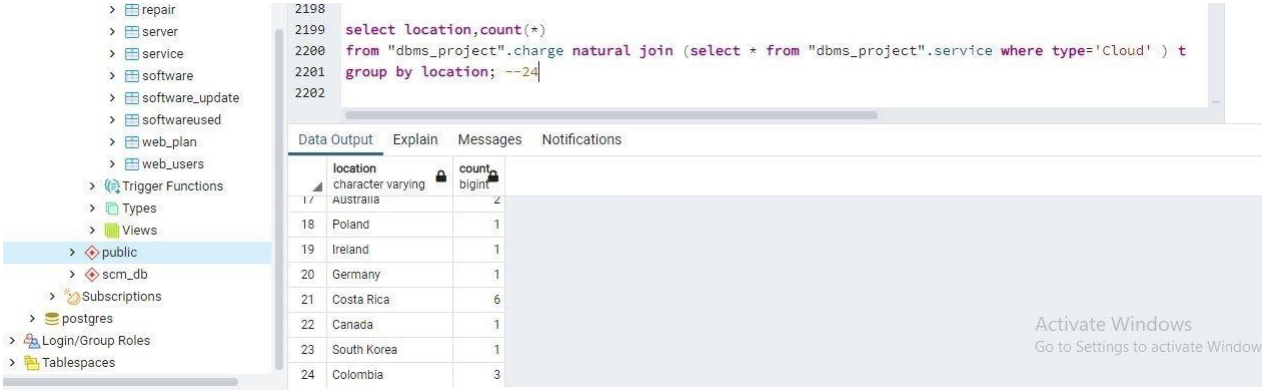
32. Write a query to find total number of users who are using cloud-based services at every location.

select location,count(*)

from "dbms_project".charge natural join (select * from "dbms_project".service where type=

'Cloud')t

group by location; --24



33. Write a query to find total number of users who have subscribed for "Platinum" web-plan at every location.

select location,count(*)

from "dbms_project".Web_Users natural join (select * from "dbms_project".Web_plan where

web_subscription_type='Platinum') t

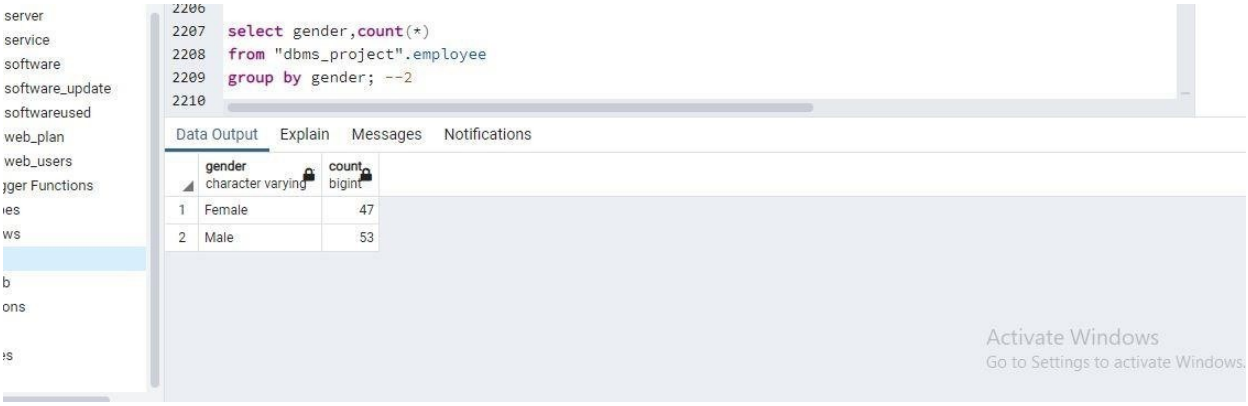
group by location; --17

34. Write a query to find count of male-employees and female-employees.

select gender,count(*)

from "dbms_project".employee

group by gender; --2

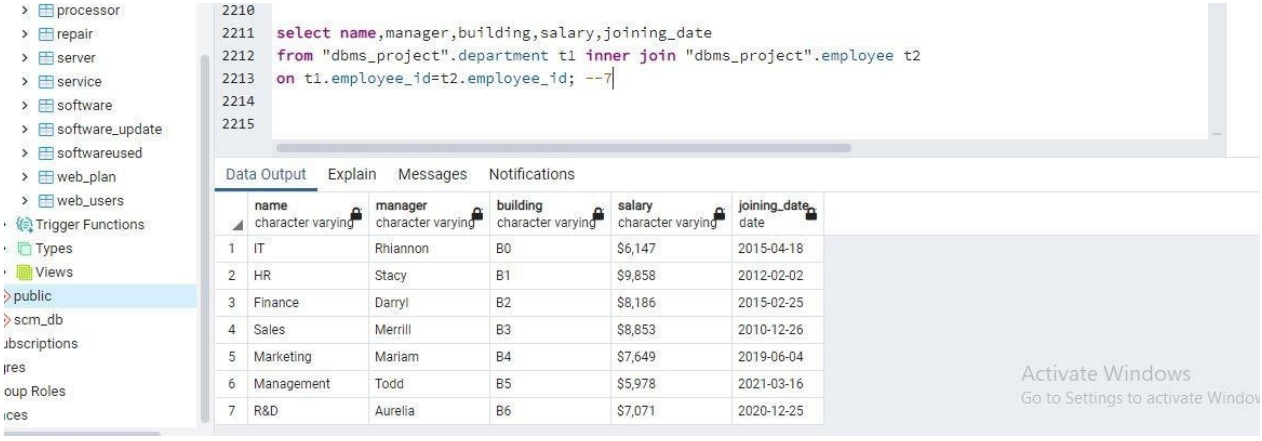


35. Write a query to find salary and joining_date of each manager present in department relation.

select name,manager,building,salary,joining_date

from "dbms_project".department t1 inner join "dbms_project".employee t2 on

t1.employee_id=t2.employee_id; --7

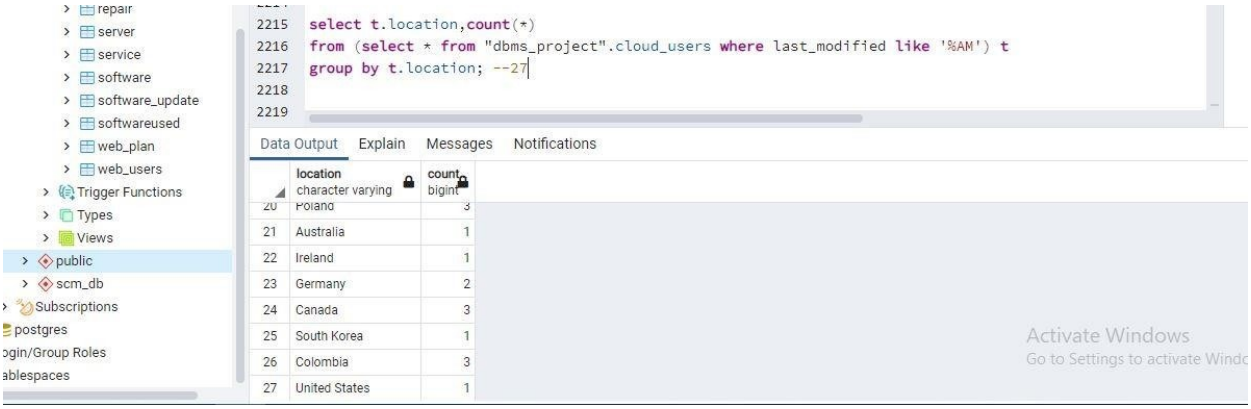


36. Write a query to find total number of cloud_users who were last active in Ante Meridiem at every location.

select t.location,count(*)

from (select * from "dbms_project".cloud_users where last_modified like '%AM') t group by t.location;

--27



37. Write a query to find total number of users who have subscribed for "Diamond" web-plan at every location using 'Having' clause.

```
select location,count(*)
from "dbms_project".web_users natural join "dbms_project".web_plan group by
location,web_subscription_type
having web_subscription_type='Diamond'; --20
```

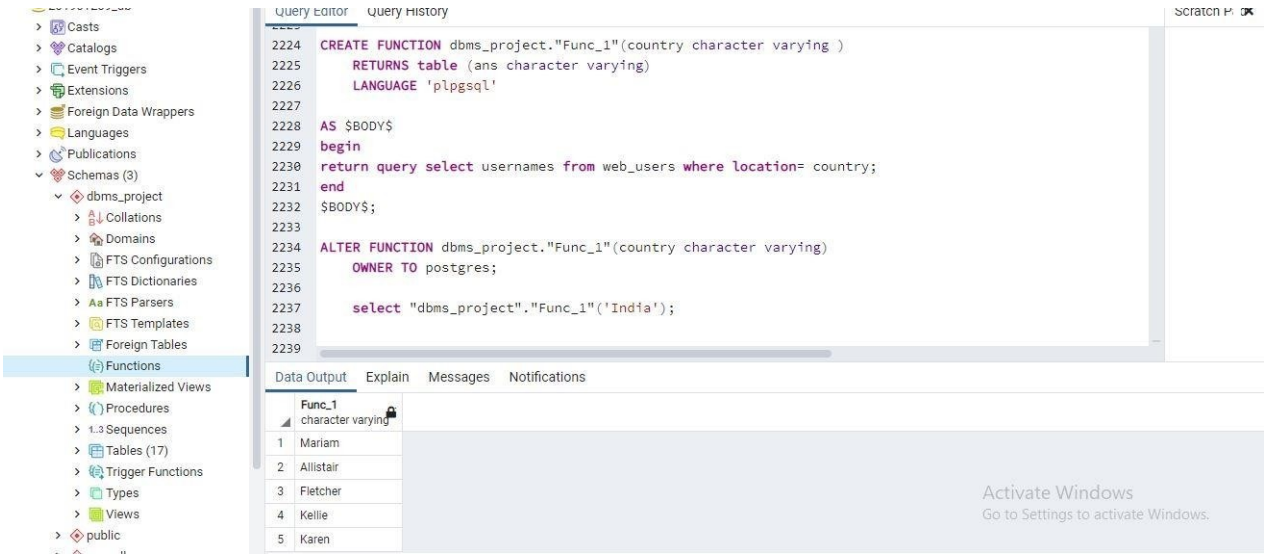
38. Write a function which lists usernames of all the web-users at provided location.

```
CREATE FUNCTION dbms_project."Func_1"(country character varying ) RETURNS
    table (ans character varying)
    LANGUAGE 'plpgsql'

AS $BODY$
begin
return query select usernames from web_users where location= country; end
```

$BODY$;

ALTER FUNCTION dbms_project."Func_1"(country character varying) OWNER TO
postgres;

select "dbms_project"."Func_1"('India');

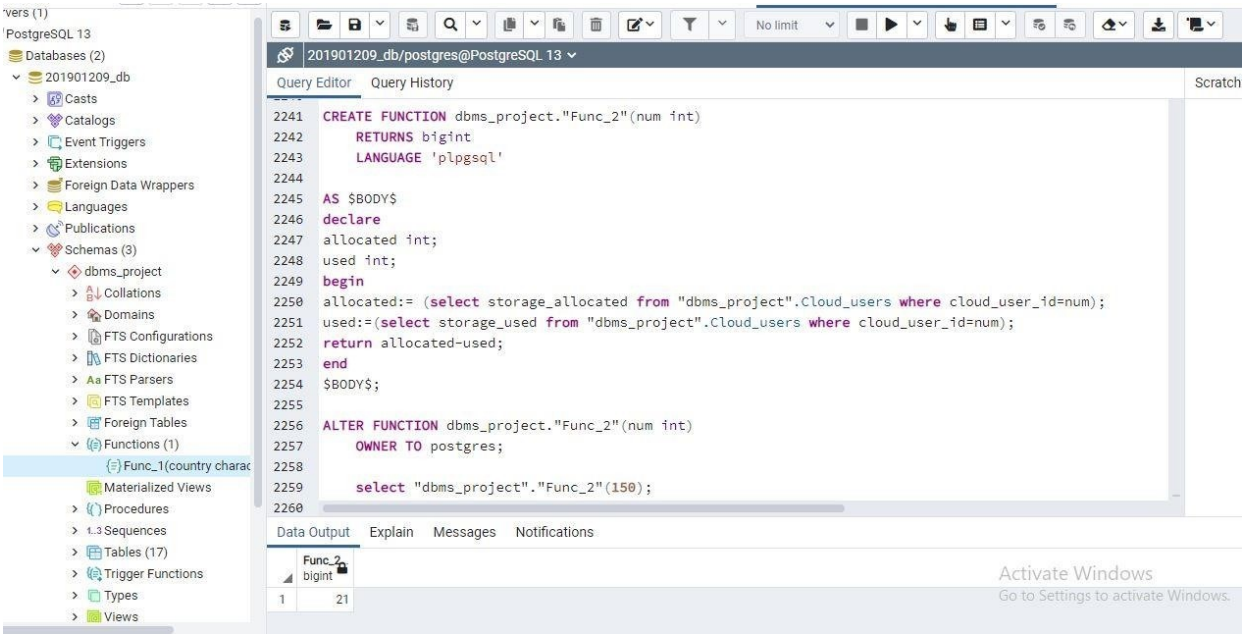


39. Write a function to find remaining cloud storage of given cloud-user.

CREATE FUNCTION dbms_project."Func_2"(num int)
RETURNS bigint
LANGUAGE 'plpgsql'

AS $BODY$
declare
allocated int;
used int; begin
allocated:= (select storage_allocated from "dbms_project".Cloud_users where cloud_user_id=num); used:=(select
storage_used from "dbms_project".Cloud_users where cloud_user_id=num);
return allocated-used;

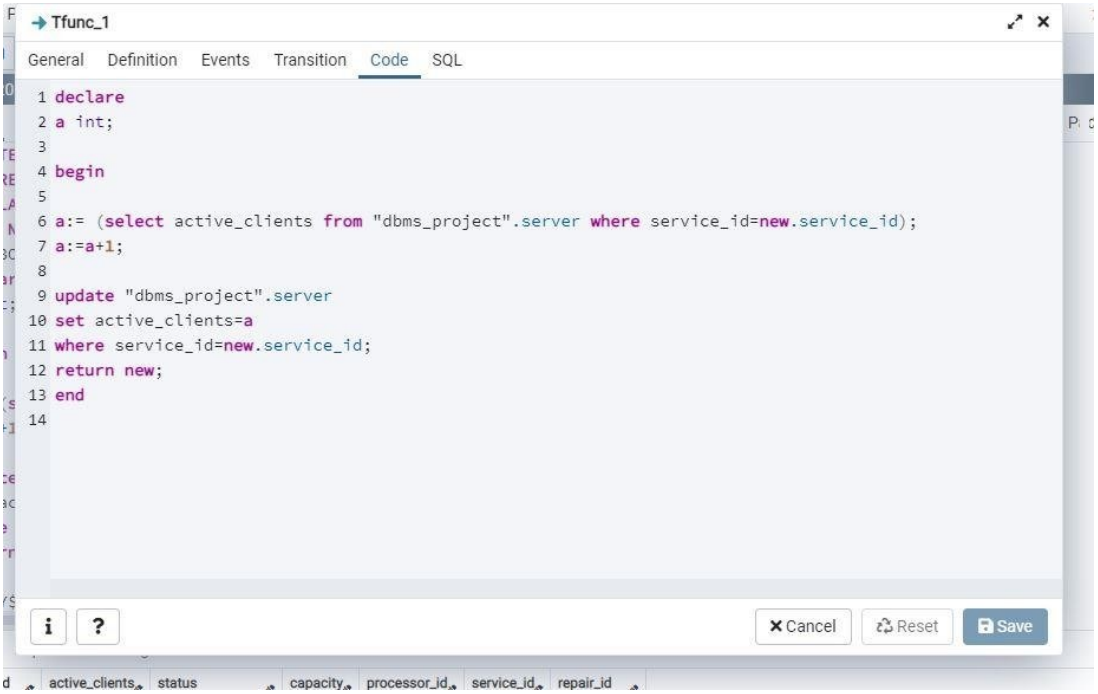
```
end
$BODY$;


ALTER FUNCTION dbms_project."Func_2"(num int) OWNER
    TO postgres;


        select "dbms_project"."Func_2"(150);
```



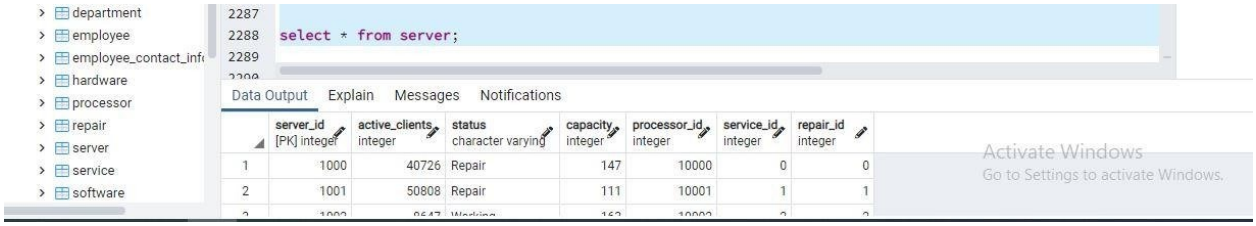
40. Write a trigger function to implement this functonality: when a new cloud user is inserted in cloud_Users relation who started using a particular service-x then number of active_clients on server-y corresponding to service-x should be incremented as well.

```
CREATE FUNCTION dbms_project."Tfunc_1"()
    RETURNS trigger
    LANGUAGE 'plpgsql'
    NOT LEAKPROOF
AS $BODY$
declare a
int;


begin
```

```
a:=(select active_clients from "dbms_project".server where service_id=new.service_id);  a:=a+1;

update "dbms_project".server  set
active_clients=a
where service_id=new.service_id;  return
new;
end
$BODY$;

ALTER  FUNCTION  dbms_project."Tfunc_1"()
   OWNER  TO  postgres;

Insert into "dbms_project".cloud_users(cloud_user_id,username,last_modified,storage_allocated,storage_used
,location,service_id)
values (200,'Satyam','12:00 AM',100,80,'India',0);

select * from server;
```



Before insertion: Take a note that number of active_clients = 40726 corresponding to service_id=0



After insertion: Take a note that number of active_clients =40727 corresponding to service_id=0

Section9: Project Code with output screenshots

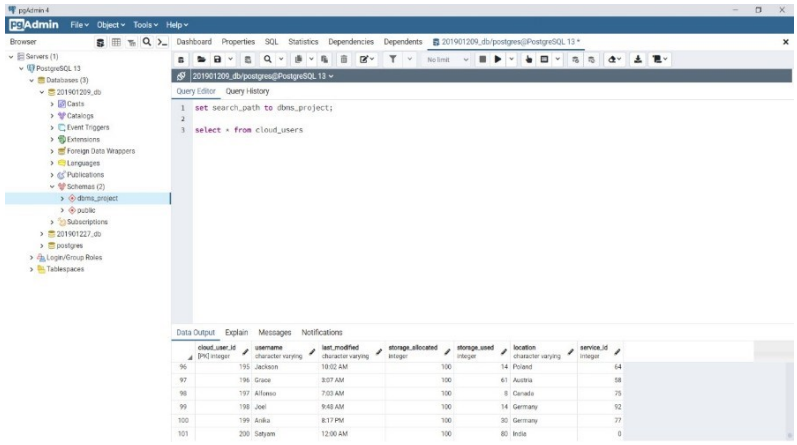
- **Establishing connection with Pgadmin using python**

```
import psycopg2


#establishing the connection
conn = psycopg2.connect(
   host='localhost',database="201901209_db", user='postgres', password='admin',
)
#Creating a cursor object using the cursor() method
cursor =conn.cursor()


#Executing an MYSQL function using the execute() method
cursor.execute("selectversion()")


# Fetch a single row using fetchone() method.
data =cursor.fetchone()
print("Connection established to: ",data)
```

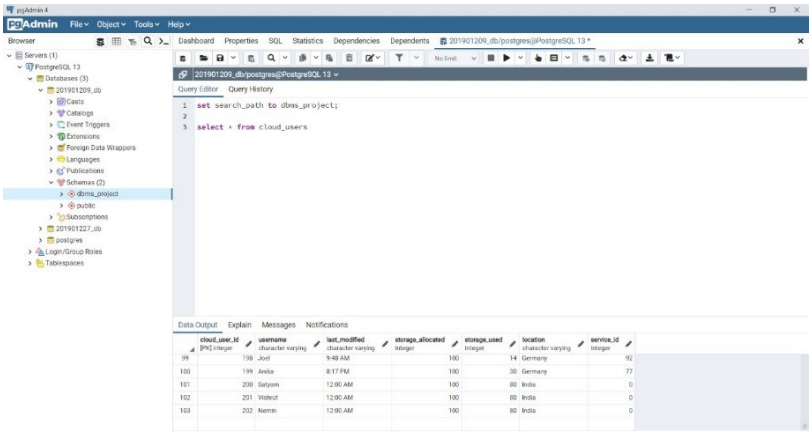
- **Executing insert statement**

```
cursor.execute("'Insert into
"dbms_project".cloud_users(cloud_user_id,username,last_modified,storage_allocated,storage_used
,location,service_id)
values (201,'Vishrut','12:00 AM',100,80,'India',0);
"")
cursor.execute("'Insert into
"dbms_project".cloud_users(cloud_user_id,username,last_modified,storage_allocated,storage_used
,location,service_id)
values (202,'Nemin','12:00 AM',100,80,'India',0);
"")
conn.commit()
```

- **Before Insertion**



- **After Insertion**



- **Executing selectstatement**

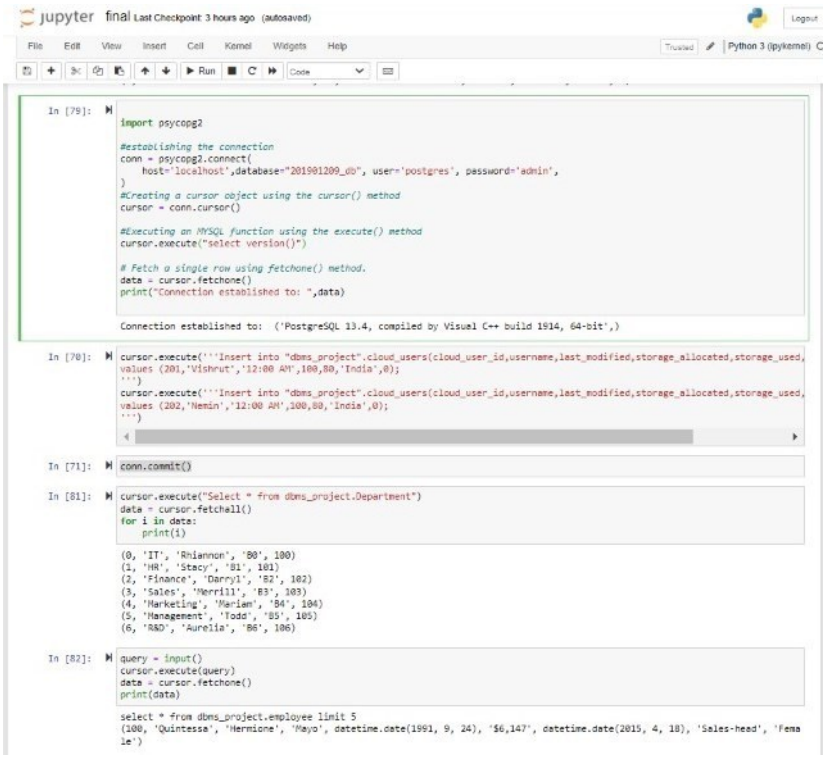
```
cursor.execute("Select  * from dbms_project.Department")
data = cursor.fetchall()
for i in data:
   print(i)
```

```
In [81]: cursor.execute("Select * from dbms_project.Department")
         data = cursor.fetchall()
         for i in data:
             print(i)

(0, 'IT', 'Rhiannon', 'B0', 100)
(1, 'HR', 'Stacy', 'B1', 101)
(2, 'Finance', 'Darryl', 'B2', 102)
(3, 'Sales', 'Merrill', 'B3', 103)
(4, 'Marketing', 'Mariam', 'B4', 104)
(5, 'Management', 'Todd', 'B5', 105)
(6, 'R&D', 'Aurelia', 'B6', 106)
```

- **Executing select statement through custom query**

```
query = input()
cursor.execute(query)
data  = cursor.fetchone()
print(data)
```

Screenshots of jupyter notebook



Thank You!