

**Database Management System (IT-214)**  
**2021**  
**DA-IICT**



**Section 2 - Team 6**  
**Chickenpox Vaccination Drive**  
**Management System**

**Name of group members:**

Nidhi Manek (201901402)

Krupal Patel (201901406)

Vidhi Badrakiya (201901434)

Kavya Parikh (201901474)

Mentor TA: Devanshi Ma'am

# Index

<b>Topic</b>	<b>Page Number</b>
<b>Section 1: Final version of SRS</b>	4 - 24
Introduction	5
Fact Finding Phase	9
Requirements	21
User classes and Characteristics	22
Operating Environment	22
Product Functions	23
Privileges	23
Assumptions	24
Business Constraints	24
<b>Section 2: Noun Analysis</b>	25 - 33
Table 1 : Noun and Verb Analysis	26
Table 2 : Accepted noun and verb List	29
Table 3 : Rejected noun and verb List	31
Development of ER Diagram	32
<b>Section 3: ER-Diagrams all versions</b>	34 - 37
Finalising ER Diagrams	35
ER Diagrams	37
<b>Section 4: Conversion of Final ER-Diagram to Relational Model</b>	38 - 39
Mapping ER model to Relational Model	39
<b>Section 5: Normalization and Schema Refinement</b>	40 - 44
Existing Schema	41
Types of Dependencies	41
Redundancies and anomalies	42
Normalization of database upto 1NF	42

Normalization of database upto 2NF	43
Normalization of database upto 3NF/BCNF	44
Final schema (BCNF)	44
<b>Section 6: SQL: Final DDL Scripts, Insert statements, 40 SQL Queries with Snapshots of output of each query</b>	45 - 109
Final DDL scripts	46
Insert statements and screenshots of loaded data	51
Trigger Functions	59
Functions	65
SQL Queries	68
<b>Section 7: Project Code with output screenshots</b>	110 - 119
Front-End Code	111
Queries and Output Screenshots	115

## Section1:

### Final version of SRS

# **Introduction**

## **Purpose:**

To create a systematic way to provide chickenpox vaccines to all eligible people and keep track of vaccinated and unvaccinated people and other related information using a database which provides a management system for vaccination drives.

## **Intended audience and Reading suggestions:**

Different types of readers: Developers, Project Managers, Testers, Marketing staff, users-1 (administrators and staff), users-2 (people to be vaccinated) and documentation writers .

Sr No.	Reader Type	Reading suggestions
1	Developers	1.The team who develops the software 2. [A] - [J]
2	Project Managers	1.The team who takes care of functional aspects 2. [A] - [J]
3	Testers	1.The team which checks the software for proper functioning 2. [D]
4	Marketing staff	1.The team which promotes the software 2. [E] , [G] , [I] , [J]
5	users-1	1.The people who will run vaccination centers. 2. [A] - [J]
6	users-2	1.The common people who take the vaccine. 2. [A] , [B] (only Flow and Features)
7	Documentation writers	1.The team who makes the structured document 2.[A]-[J]

The rest of the SRS describes a detailed description of the vaccination process, an overview of different fact-finding techniques, product requirements, product functions, user classes and characteristics, privilege classes and business constraints. The structured and organised form of the document helps in efficiently listing the requirements and thus in developing the software further.

## **Product scope:**

The software serves as a one stop destination to keep an account of all domains related to the Chickenpox vaccine. People from various age groups can book slots online or offline at their desired time, convenience and based on availability of different centers. The database keeps track of details of vaccination centers, vaccine stocks, which employees are available for doorstep visits, personal details of people, etc.

## **Description:**

### **About the disease:**

- Varicella (chickenpox) is an acute, highly contagious viral disease with worldwide distribution.
- The highest prevalence occurs in the 4-10 year age group but tends to be more severe in adults.
- Chickenpox is usually treated as a harmless childhood disease, and rarely rated as an important public health problem, but this can be severe and even fatal in otherwise healthy children.
- Chickenpox can cause pneumonia and is an important risk factor for developing severe invasive streptococcal disease.
- Complications of chickenpox include bacterial infections, decreased platelets, arthritis, hepatitis, pneumonia.
- Chickenpox is the leading cause of vaccine-preventable death in children.
- Universal vaccination can cause a dramatic reduction in the incidence of varicella, associated complications, hospitalizations and fatality rates.

### **Precautions:**

Chickenpox vaccine is not suitable for everyone.

The type of people who **can not** register for the vaccine:

- People who are currently affected by chickenpox.
- If people had a life-threatening allergic reaction in response to the previous dose.
- Who are moderately or severely ill at the time the slot is scheduled.
- Pregnant women should not get chickenpox vaccines.
- People who have a severe disease that affects immunity system

The chickenpox vaccine is recommended for **all children under age 13 who have not had chickenpox**. It is also recommended for all adolescents and adults who have not been vaccinated and have not had chickenpox. There is no need to take the vaccine if a person has had chickenpox.

The type of people who **should** necessarily register for the vaccine:

- Healthcare professionals
- Childcare workers
- Residents and staff of nursing homes
- International travellers
- Military personnel
- Non-pregnant women of child-bearing age

### **Registration:**

- Citizens of any age can register for the vaccination. Registration can be done online, offline or door-to-door.
- At the registration time system will take data such as Registration ID, Vaccine ID, Vaccine Name, Registration Mode, Center ID, Dose and Registration Time and Date.
- The people who are currently affected by chickenpox, had a life-threatening allergic reaction in response to the previous dose, women who are pregnant or are severely ill at the time when slot is scheduled should not register for the vaccination.

- Registered user include details about their previous(first) dose details and also guardian details (if any)
- For the online registration, the user has to register on an online portal.
- Here each registered user will have some data such as User Id, User Name,User Age, User Gender, User Address, User Phone Number and User Guardian details.
- In the registration process there are details also: first dose or first dose has already been taken already and second dose.
- There are registered slot details too which contain slot time.
- Government vaccination center takes no charges for vaccinating people while prices for private centers are restricted by the government to prevent black marketing.
- Show the vaccine price while registering from private organizations.
- Registration is based on unique identity. Infants do not have aadhar card or voter card so birth certificates will be used for unique user identity.

### **Scheduling Appointment:**

- One can book a slot online or can reach the designated vaccination center for offline scheduling. If the slot is free, they receive the vaccination on the spot. Or else they will get a dose scheduled on some other near and convenient day.
- Each slot will have its corresponding slot ID. The slot will have details such as time, date, day. The users will be able to see such details while registering. The online registration capacity of each slot will also be shown. No more slots than this capacity can be booked.
- To provide the registered slot rescheduling facility before 12 hours in case a user is not available due to an emergency. We are providing them a facility so that they can change the slots according to their convenience. Also users should not reschedule the slot before less than 12 hours as that will not give enough time to other users to book that slot.
- While booking slots, the details of a consultant doctor will also be given. The patient can contact that doctor in case of complications due to vaccination.
- Sometimes users unnecessarily book the slots or they don't reach to take the vaccine in a given slot, because of that other users can not book that slot. If this behaviour is repeated more than three times by a specific user then his/her online registration process will be cancelled and he/she will not be able to register online (for that dose) again. In this case, the person will have to do the on-site (offline) registration in order to avail the vaccine.
- Another way to take the vaccine is through door-to-door appointment. The government health care workers come to your home and give you the vaccine if you are underprivileged. In case of private institutions, door-to-door service is chargeable.
- While scheduling an appointment for vaccination, the system will show vaccination centre names along with the name of the vaccine that will be administered.
- Vaccination centers based on locality - pincode or area name etc. will be shown. If these details are provided, it becomes convenient for the user to book the nearest center.
- The registration slip can be downloaded after the appointment has been scheduled.
- Once an appointment is scheduled, you will receive the details of the vaccination centre, date and time slot chosen for appointment in an SMS sent to your registered mobile number. You can also download the registration slip and print it or keep it on your smartphone.
- People registering for 2nd dose should have already taken 1st dose and 1st dose should have been taken before at least 1 month. This is because it is necessary to have a gap of 1-3 months before taking the 2nd vaccine for online registrations.

### **Vaccination process:**

- Each vaccination center will have its corresponding center ID, center name, center address. The vaccination center type,i.e, private or government or both, will also be assigned.
- Each vaccination center will also have its maximum staff capacity. This will show the maximum staff that the vaccination center can contain.
- For each vaccination center, there will also be an administrator which oversees all the aspects.

- The Vaccination Centres are directed to ensure that if a citizen is being vaccinated with 2nd dose, they should confirm that the first dose vaccination was done with the same vaccine as is being offered at the time of second dose and that the first dose was administered more than 28 days ago for offline registrations.
- After successful vaccination, a vaccination certificate will be provided to you that states you have successfully taken the vaccine. Unique e-certificate id will be provided using which the user can access e-certificate from the corresponding e-portal.
- The Vaccination Centre is responsible for generating your certificate and for providing a printed copy post vaccination on the day of vaccination itself. Please do insist on receiving the certificate at the Centre.

**Vaccine details:**

**Varivax**

- Contains only chickenpox vaccine
- Is licensed for use in children age 12 months and older, adolescents, and adults
- Can be given to children for their routine two doses of chickenpox vaccine at age 12 through 15 months and age 4 through 6 years

**ProQuad**

- Contains a combination of measles, mumps, rubella, and varicella (chickenpox) vaccines, which is also called MMRV
- Is only licensed for use in children age 12 months through 12 years

**Vaccination Management functionalities:**

- Administrators of each center will manage all the functionalities related to vaccine management at different centers. A common employee table will help to keep track of administrator details, doctor details and staff details of different vaccination centers. It also has details like admin landline number, doctor email address and staff type.
- Dynamic online slots. If a vaccination slot is suitable for many people then there will be a lot of online and offline registration for that particular slot. In this case, if we have a rough idea about the number of people coming offline for a particular slot or we can do an offline slot approximation analysis, then we can set a limit to online registrations providing dynamic online slots. So that we can easily provide vaccines to each registered user.
- Record the number of doses of vaccines available, in real-time and also record the 1st dose and 2nd dose details like e-certificate id, vaccine id. At each center, details of remaining proquad vaccine and varivax vaccine will be stored for every corresponding date.
- Give details of consultant doctors like doctor id, doctor name, doctor number so that users can consult doctors at vaccine centers.
- Allocate a unique vaccine id. It may happen that at the end of a working day, all vaccines may not have been used (some users may have failed to show up) or some vaccines may have gone wasted. Vaccine id helps to keep track of those.
- At the end of each day we can check the staff utilisation using the number of vaccines provided or the number of registrations done by each staff member. This will help us to transfer the staff member with less utilisation to the center with full utilisation (given transferring this staff member will not make another center to cross staff capacity of that center).
- Also, if a new staff member is added we can allot that staff member to the center with full utilisation. In this way, staff utilisation will help us to remove, transfer and add staff members to the different centers.
- To give users the details of allocated staff members like staff id, staff name, staff type from whom they are taking their vaccine for user reference.
- It also records allocated staff details like staff id, center id, date at which the staff allocated to that center, and attended users.
- Update the user's registered data (if required). Sometimes it may happen that the user wants to update some details like name, age, gender etc. We need to allow users to update wrong/invalid data.

- It also records allocated staff details like staff id, center id, date at which the staff allocated to that center, and attended users.
- Each center has some limited space or area which can be used for vaccination drives. For example, if each of the staff members can provide vaccines to 10 users, then 5 staff members will be able to provide vaccines to 50 people. It is necessary to know, in one time slot, whether the center area will be able to cover 50 people or not. If not then the place will be much crowded. To solve this problem, we can set a limit to the number of staff members in the center, this will limit the number of registrations hence solving the problem.
- It may happen that at the end of a working day, all vaccines may not have been used (some users may have failed to show up) or some vaccines may have gone wasted. At the end of each day, remaining vaccines will be returned and a report of wasted vaccines will be given to the authorities. In this way, the center's stock details are kept account of.

## **Fact Finding Phase**

### **1) Background Reading:**

#### **Working and flow:**

We studied the existing vaccination system. The inference of various stages in the vaccination system is as follows:

#### **Registration:**

- For the online registration, log in into the online portal and click on the “Register/Sign In yourself” tab to register for chickenpox vaccination.
- Citizens of any age can register for the vaccination.
- The people who are currently affected by chickenpox, had a life-threatening allergic reaction in response to the previous dose, women who are pregnant or are severely ill at the time when slot is scheduled should not register for the vaccination.
- The registration process will include details about their name, age, previous dose details, guardian details (if any)
- Registration can be done online or offline.
- Government vaccination center takes no charges for vaccinating people.
- Registration on the portal can be done using any of the following ID proofs: a. Aadhaar card b. Driving License c. PAN card d. Passport e. Pension Passbook f. NPR Smart Card g. Voter ID h. Birth Certificate

#### **Scheduling Appointment for 1st dose and 2nd dose:**

- One can book a slot online or can reach the designated vaccination center. If the slot is free, they receive the vaccination on the spot. Or else they will get a dose scheduled on some other near and convenient day.
- Another way to take the vaccine is through door-to-door appointment. The government health care workers come to your home and give you the vaccine if you are underprivileged. In case of private institutions, door-to-door service is chargeable.
- While scheduling an appointment for vaccination, the system will show vaccination centre names along with the name of the vaccine that will be administered.
- The appointment slip can be downloaded after the appointment has been scheduled.

- You cannot cancel an appointment already scheduled. You can also reschedule the appointment and choose another date or time slot of your convenience.
- Once an appointment is scheduled, you will receive the details of the vaccination centre, date and time slot chosen for appointment in an SMS sent to your registered mobile number. You can also download the appointment slip and print it or keep it on your smartphone.
- It is recommended that both doses of vaccine should be taken 1-3 months apart for realising the full benefit of vaccination. Both doses must be of the same vaccine type.

#### **Vaccination:**

- Vaccination is free at Government hospitals and charged up to pre decided amount per dose in Private hospitals. Citizens can get this information while booking an appointment.
- The Vaccination Centres are directed to ensure that if a citizen is being vaccinated with 2nd dose, they should confirm that the first dose vaccination was done with the same vaccine as is being offered at the time of second dose and that the first dose was administered more than 28 days ago.
- After successful vaccination, a vaccination certificate will be provided to you that states you have successfully taken the vaccine.
- The Vaccination Centre is responsible for generating your certificate and for providing a printed copy post vaccination on the day of vaccination itself. Please do insist on receiving the certificate at the Centre.
- You can also download the certificate from the online portal.

#### **Features:**

Any vaccine management software will allow us to manage vaccination processes and workflows with features such as inventory management for managing and maintaining the supply chain, and patient management features for storing patient-related information and tracking vaccination doses.

Below are some common features that any vaccine management software will have:

- **Appointment Management:** Register and schedule/reschedule appointments for vaccination slots and sites. Selection of vaccination centers of convenience. Schedule vaccination Date as per slot availability at a center. Once the appointment is fixed it can be rescheduled at any later stage but before the vaccination appointment day.
- **Medical Records:** Create patient medical records that include vaccination records for first and second doses.
- **Vaccine Information/Guidelines:** Send patients information about vaccines and what they should expect at the vaccination site. Deliver educational content to patients and address their vaccine hesitancy with knowledge bases, virtual assistants, and more.
- **Reminders:** Text/Email for initial and second doses, as well as specific instructions to follow once they get vaccinated.
- **Follow-ups/alerts:** Send instructions to patients via text/email about who to contact in case of emergency. Also record any side effects they experience using an e-form.
- **Waitlist Management:** Manage your patient queues and waitlists based on custom criteria, and set priorities, such as for healthcare workers and other frontline essential workers.

Below are some features of CoWIN app, which is the official COVID-19 vaccination registration app in India for reference:

- CoWIN will give permission to create a Unique Health ID for every user.
- After administration of both doses of a COVID-19 vaccine, a QR code certificate will be generated which can be stored on the government's DigiLocker app.

- The app will provide an automated allocation of vaccination sessions. One of the authentication methods would be the use of the Aadhaar number to prevent malpractice in the vaccination drive, reported NDTV.
- In case of any adverse effect following immunisation, there is a provision for real-time reporting on the digital platform.
- The app includes features like SMS in 12 languages, 24X7 helpline, and others.

#### **Flaws:**

- Some of the problems that were reported on the vaccine management app from several cities across the country are that registered beneficiaries failed to receive prior intimation on attending centers to receive the vaccination and other major glitches that even slowed down the process of delivering vaccines.
- In the wake of the wide-scale COVID-19 vaccination that is in active progression across India, a new development in the form of fake CoWIN vaccination apps are threatening the process of genuine vaccination and slot booking, as was reported by CERT-In.
- Many citizens started facing errors at the OTP stage itself, getting “ Unexpected Error” as well as “Server is facing issues. Please try later,” messages.
- While some are facing issues with the OTP, others are not able to find any slots for weeks after successfully registering.
- Jio network glitch affected the vaccination work as many of the centers depended on mobile internet connection or got the connectivity through a Wi-Fi dongle to get the connectivity with the vaccine management software, especially many centers in the rural areas and some in urban areas. The trouble due to Jio’s network going down many centers faced trouble in the registration of beneficiaries and also affected the target of vaccination.

#### **References:**

- <https://vaccineindia.org/article/chickenpox>
- <https://www.mygov.in/covid-19>
- <https://pubmed.ncbi.nlm.nih.gov/21791972/>
- <https://palpalnewshub.com/india-news/special-features-of-cowin-app-created-for-vaccination-know-the-full-detail/>
- <https://cyberdaily.securelayer7.net/multiple-fake-cowin-vaccination-apps-duping-users-cert-in-advisory/>
- <https://www.msn.com/en-in/news/other/glitches-in-cowin-20-hold-up-vaccination-centre-must-upgrade-app-capacity-to-meet-demand-say-experts/ar-BB1e78xi>

#### **Requirements from Background Reading:**

- To get all the details of the user while registering
- To understand about the preferences of users from online/offline/door-to-door mode
- To see that person is eligible to take the vaccine. Priority classes should get higher preference
- To record the number of doses of vaccines available, in real-time
- To assure that people registering for 2nd dose have already taken 1st dose
- To assure efficient working of app with slot details
- To see that vaccines do not get wasted
- To manage patient queues and waitlists based on custom criteria, and set priorities
- To schedule vaccination date as per slot availability at a center
- To provide selection of vaccination centers of convenience
- To provide adequate staff at each vaccination center

- To provide the slot rescheduling facility
- To show the price of vaccine while registering privately

## 2) Interviews:

### **1. Management Requirements: (Role Play) Interview Plan**

**System:** Vaccination drive management

**Interviewee:** Kavya Parikh (**Role Play**)

**Designation:** Head of Management at city vaccination center

**Interviewer:** Nidhi Manek

**Designation:** Developer of software required for vaccination drive management system

**Date:** 6/10/2021      **Time:** 11:00 AM

**Duration:** 45 minutes

#### **Purpose of Interview:**

To identify the issues and requirements regarding the management of the drive.

#### **Agenda:**

Identify and discuss the problems related to management of users, staff and stock of vaccines.

### **Management Requirements: (Role Play) Interview Summary**

**System:** Vaccination drive management

**Interviewee:** Kavya Parikh (**Role Play**)

**Designation:** Head of Management at city vaccination center

**Interviewer:** Nidhi Manek

**Designation:** Developer of software required for vaccination drive management system

**Date:** 6/10/2021      **Time:** 11:00 AM      **Duration:** 45 minutes

#### **Purpose of Interview:**

To identify the issues and requirements regarding the management of the drive.

1. Unique user id: If a user wants to register for the vaccine then he/she will be able to register using only the birth-certificate. So that we can prevent the system from multiple registrations for the same user.
2. Online registration prohibition: If a user fails to reach the slot at a given time, this failure is allowed for maximum 3 times for that slot. It is to prevent users from unnecessarily booking slots. After it, the person will have to do the on-site (offline) registration in order to avail the vaccine.
3. E-certificate: The users will be given the unique e-certificate id from which they will be able to access the e-certificate from e-portal.

4. Remaining and wasted vaccine management: At the end of each day, remaining vaccines will be returned and a report of wasted vaccines will be given to the authorities.
5. Dynamic online slots: If a vaccination slot is suitable for many people then there will be a lot of online and offline registration for that particular slot. In this case, if we have a rough idea about the number of people coming offline for a particular slot, then we can set a limit to online registrations. So that we can easily provide vaccines to each registered user.
6. Staff capacity of centers: Each center has some limited space or area which can be used for vaccination drives. To cover the registered people in the given area in a given slot (to avoid crowd), there should be a limit to registrations per slot and hence there should be a limit to the number of staff members in the center.
7. Staff utilisation: At the end of each day we can check the staff utilisation using the number of vaccines provided or the number of registrations done by each staff member. This will help us to transfer the staff member with less utilisation to the center with full utilisation (given transferring this staff member will not make another center to cross staff capacity of that center).

## **2. User Requirements: (Role Play) Interview Plan**

**System:** Vaccination Drive Management

**Project Reference:** SF/SJ/2021/13

**Interviewee:** Amisha Sharma(**Role Play**)

**Designation:** User Requirements Head

**Interviewer:** Vidhi Badrakiya

**Designation:** Administrator at Vaccine management drive

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

**Duration:** 30 minutes **Place:** Vaccine center

### **Purpose of Interview:**

Meeting to identify problems and requirements regarding Vaccination drive for each mode from the user's perspective.

### **Agenda:**

Problems with User's experience and their concern about vaccination center

Mobile application concerns

## **User Requirements: (Role Play) Interview Summary:**

**System:** Vaccination Drive Management

**Project Reference:** SF/SJ/2009/13

**Interviewee:** Amisha Sharma(**Role Play**)

**Designation:** User Requirements Head

**Interviewer:** Vidhi Badrakiya

**Designation:** Developer of Vaccine management drive for software

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

**Duration:** 30 minutes **Place:** Vaccine center

**Purpose of Interview:**

Meeting to identify problems and requirements regarding Vaccination drive for each mode from the user's perspective.

1. Registration process should be based on a unique identity.
2. People come to the vaccination center after registering online(slot wise) then also they have to wait for a long time in the queue.
3. Mobile applications should give vaccination centers according to the user's locality.
4. There should be a choice of preference for users.
5. Users should get vaccine id for verification of vaccine.
6. Users want to get details of the person from whom they are getting the vaccine.
7. If in case pre-booked slot time is missed by the user then users should get a chance to book another slot.
8. In receipt there should not be any spelling mistakes.
9. It should give confirmation that the vaccine has been taken as early as possible.

**Requirements from Interview:**

- To implement constraint in such manner so that registration is based on unique identity
- To determine staff utilisation at the end of each day which will help to transfer, remove or add staff members from different centers
- To implement method so that user don't have to wait in queue for longer (set slots capacity)
- To give E-certificate id
- To determine the staff capacity of each centers and allocate staff accordingly
- To implement choice for user preference
- To add unique vaccine id
- To give user details of person from them they are taking their vaccine
- To provide dynamic online slots according to staff capacity of center and approximate idea of offline registrations
- Updation of user's registered data (if required)
- To implement registration in such manner so that they can register again (for user which are registering online if pre-booked slots are missed)
- To set a unique user id that identifies user uniquely
- To manage remaining and wasted vaccines
- To provide unique e-certificate id using which user can access e-certificate from the corresponding website
- To prohibit online registration - If the user is not able to reach in a given time slot for more than three times

### 3) Questionnaire:

[https://docs.google.com/forms/d/1pVBsN62DwzQAJsxfaKUtyUMBnl4q5brlPdgRogRkZ6g/viewform?edit\\_requested=true](https://docs.google.com/forms/d/1pVBsN62DwzQAJsxfaKUtyUMBnl4q5brlPdgRogRkZ6g/viewform?edit_requested=true)



## Vaccination Survey

Please take a few minutes to fill out the form. This data will help us analyze data related to chickenpox vaccination drive.

 201901402@daiict.ac.in (not shared) [Switch account](#) 

\* Required

Name \*

Your answer

Have you ever suffered from Chickenpox? \*

Yes  
 No

Have you ever taken Chickenpox vaccine? \*

Not Taken  
 Both doses  
 1 dose

In which age group do you belong? \*

5-12 Years  
 13 Years and above  
 1-4 Years

Which vaccine have you taken?

Varivax  
 ProQuad

What was your age when you took the first dose?

- 1-4 Years
- 5-12 Years
- 13 Years and above

After how many months of first dose did you take the second dose?

- 1 Month
- 2 Months
- 3 Months
- Not taken yet

Rate the facility at vaccination drive.

1      2      3      4      5

Rating



In which mode did you take the vaccine?

- Offline
- Online
- Door to door

Have you taken vaccine from Government organization or Private organizations?

- Government
- Private

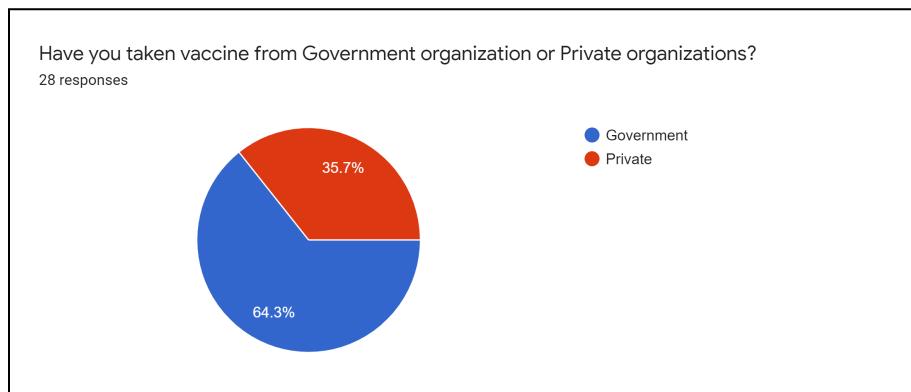
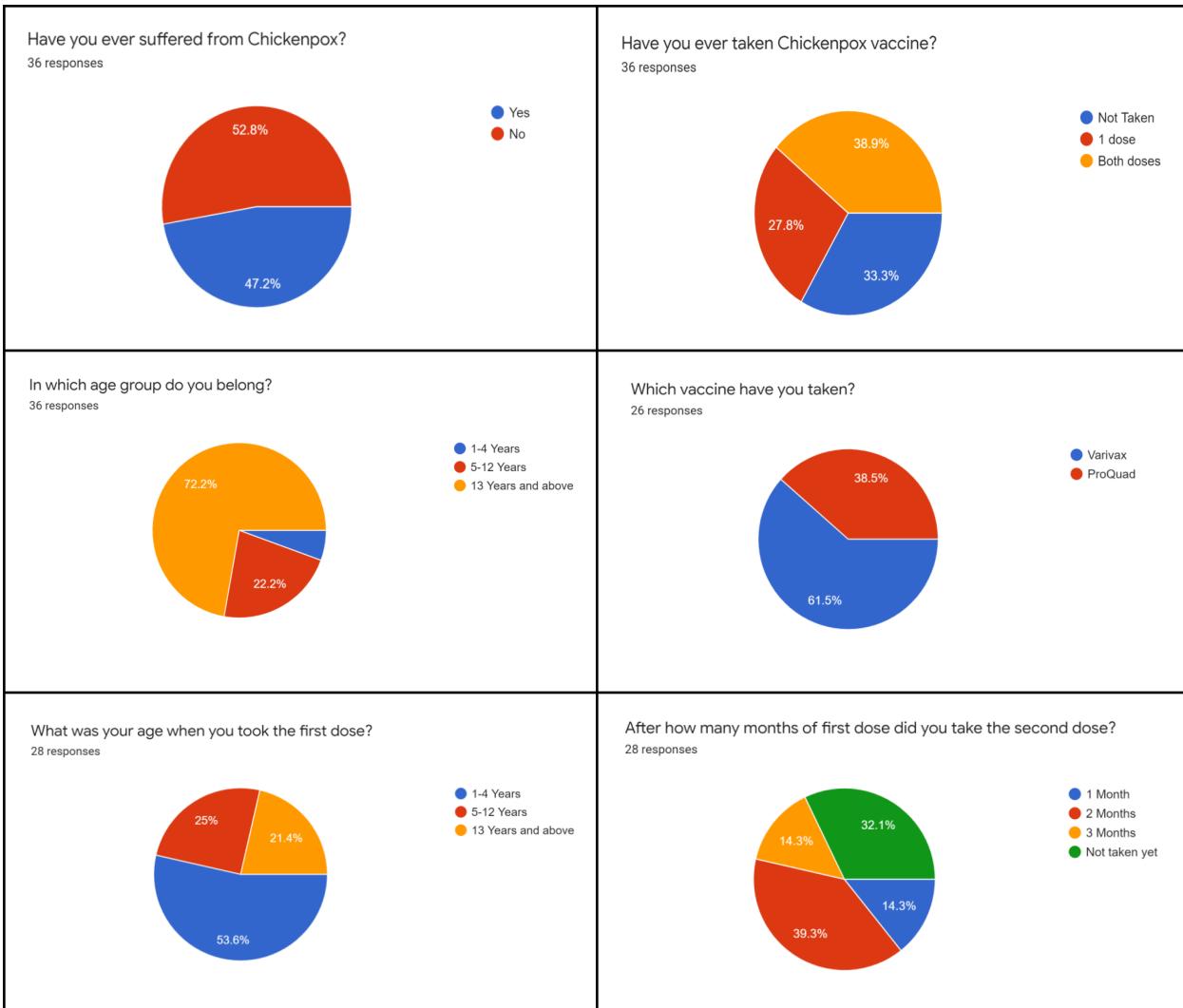
Any suggestions about improvement in vaccination drive.

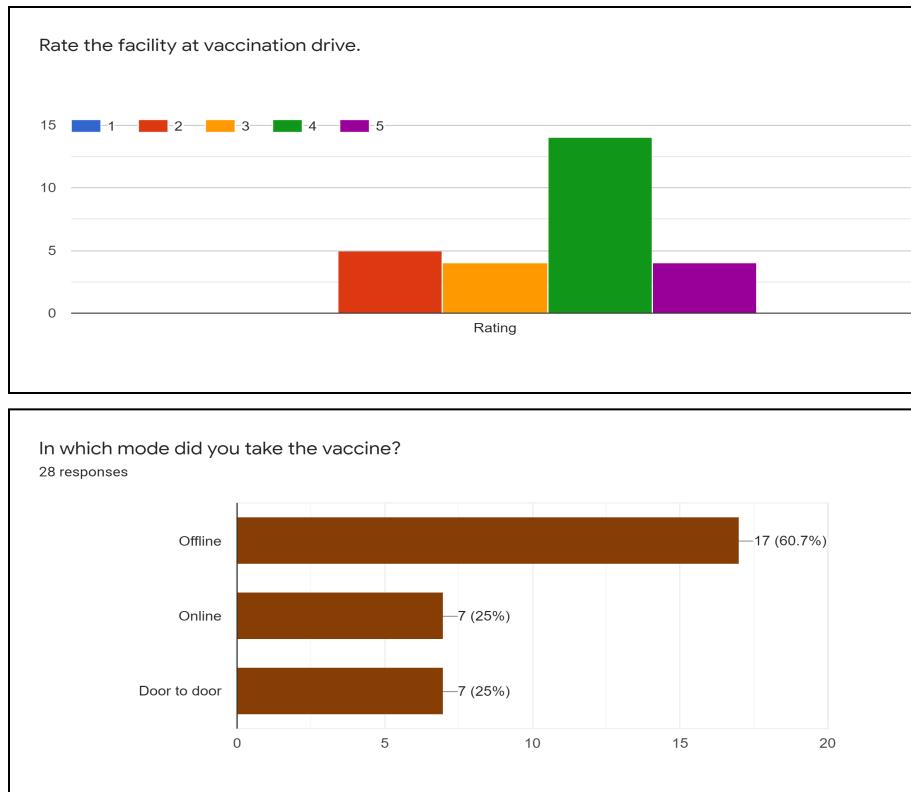
Your answer

**Submit**

**Clear form**

## Graphs and Inferences :





- The people who have taken vaccines are rarely encountered with chickenpox.
- More than half the people surveyed have been affected by chickenpox.
- More than 60% of people have taken the vaccine.
- Audience prefers offline vaccination.
- The audience prefers government vaccination centers as compared to private organisations.
- Door-to-door survey has been conducted successfully, covering 25% of people.

#### **Requirements from Questionnaire:**

- To find how many users take vaccines on time
- To find how much improvement is needed in vaccination center services
- To find if users prefer government or private organisation to take vaccine
- To find how many users have suffered from Chickenpox

#### **4) Observations :**

##### **Vaccine Drive Management: Observations**

**System:** Vaccination Drive Management

**Project Reference:** SF/SJ/2020/14

**Observations By:**

**Date:** 1/10/2021 **Time:** 11:30 AM

**Duration:** 40 minutes    **Place:** Vaccine center

**Observations:**

- Center should have the facility to consult doctors in such cases.
- In some cases if a child is not there with the parent at the vaccination center then based on the child's identity registration can be confirmed.
- Vaccination centers should be neat and clean for the hygiene purpose.
- After registering online, people come to the vaccination center but they have to wait for a long time in the queue.
- In offline registration, beneficiaries do not get a vaccination certificate in real-time as it would be generated after the data is updated on the vaccine management portal.
- Identity proof specified by you at the time of registration on the portal and a printout/screenshot of your appointment slip should be carried by you.
- Facility to users to cancel/reschedule their appointment is given for a fixed number of times
- The vaccinated people should stay at the center itself for half an hour to ensure there are no side-effects from the vaccination, if there occurs any issue they should be able to contact the doctors available at the center.
- Management should somehow regulate the price of vaccines at private centers.
- To show vaccination centers based on locality - pincode or area name etc.

**Requirements from Observation:**

- To give details of consulting doctors
- To ensure efficient use of slot timings.
- To keep users at the vaccination center for sometime after taking the vaccine so that side effects can be tackled.

**Create Fact-Finding Chart**

<b>Objective</b>	<b>Technique(s)</b>	<b>Subjects</b>	<b>Time</b>
To get all the details of the user while registering	Background Reading	Registration Aspect	1 day
To implement constraint in such manner so that registration is based on unique identity	Interview	User requirements	1.25 hours
To understand about the preferences of users from online/offline/door-to-door mode	Background Reading	Registration Aspect	0.5 day
To see that person is eligible to take the vaccine. Priority classes should get higher preference	Background Reading	Staff Responsibility	0.5 day
To implement method so that user don't have to wait in queue for longer (set slots capacity)	Interview	User requirements	1.25 hours
To find how many users take vaccines on	Questionnaire	User	2 days

time		requirements	
To record the number of doses of vaccines available, in real-time	Background Reading	Administrator Management	1 hour
To give details of consulting doctors	Observation	Vaccine drive management	2 hours
To implement choice for user preference	Interview	User requirements	1.25 hours
To assure that people registering for 2nd dose have already taken 1st dose	Background Reading	Staff Responsibility	3 hours
To assure efficient working of app with slot details	Background Reading	Administrator Management	0.5 day
To see that vaccines do not get wasted	Background Reading	Vaccine drive management	1 hour
To add unique vaccine id	Interview	User requirements	1.25 hours
To give user details of person from them they are taking their vaccine	Interview	User requirements	1.25 hours
To provide the slot rescheduling facility	Background Reading	User Facility	1 day
To find how much improvement is needed in vaccination center services	Questionnaire	Vaccine drive management	2 days
To show the price of vaccine while registering privately	Background Reading	Administrator Management	2 hours
To provide adequate staff at each vaccination center	Background Reading	Vaccine drive management	1.5 hours
To ensure efficient use of slot timings. So slots should be divided accurately	Observation	Vaccine drive management	2 hours
To provide selection of vaccination centers of convenience	Background Reading	User requirements	1.25 hours
To schedule vaccination date as per slot availability at a center	Background Reading	Vaccine drive management	0.5 day
Updation of user's registered data (if required)	Interview	User requirements	1.25 hours
To implement registration in such manner so that they can register again (for user which are registering online if pre-booked slots are missed)	Interview	User requirements	1.25 hours

To manage patient queues and waitlists based on custom criteria, and set priorities	Background Reading	Vaccine drive management	1 day
To set a unique user id that identifies user uniquely	Interview	Management requirements	1.5 hours
To prohibit online registration - If the user is not able to reach in a given time slot for more than three times	Interview	Management requirements	1.5 hours
To provide unique e-certificate id using which user can access e-certificate from the corresponding website	Interview	Management requirements	1.5 hours
To manage remaining and wasted vaccines	Interview	Management requirements	1.5 hours
To provide dynamic online slots according to staff capacity of center and approximate idea of offline registrations	Interview	Management requirements	1.5 hours
To determine the staff capacity of each centers and allocate staff accordingly	Interview	Management requirements	1.5 hours
To determine staff utilisation at the end of each day which will help to transfer, remove or add staff members from different centers	Interview	Management requirements	1.5 hours
To give E-certificate id	Interview	User requirements	1.25 hours
To show vaccination centers based on locality - pincode or area name etc.	Interview	User requirements	1.25 hours

## Requirements

- To get all the details of the user while registering (x3)
- To provide unique e-certificate id using which user can access e-certificate from the corresponding website (x2)
- To give details of consulting doctors (x2)
- To understand about the preferences of users from online/offline/door-to-door mode (x2)
- To show vaccination centers based on locality - pincode or area name etc.(x2)
- To give user, the details of staff member from whom they are taking their vaccine (x2)
- To implement constraint in such manner so that registration is based on unique identity (x1)

- To provide dynamic online slots according to staff capacity of center and approximate idea of offline registrations (x1)
- To prohibit online registration - If the user is not able to reach in a given time slot for more than three times (x1)
- To implement method so that user don't have to wait in queue for longer (set slots capacity) (x1)
- To find how many users take vaccines during a given slot. (x1)
- To record the number of doses of vaccines available, in real-time (x1)
- To assure that people registering for 2nd dose have already taken 1st dose (x1)
- To add unique vaccine id (x1)
- To determine staff utilisation (x1)
- To provide the slot rescheduling facility before 12 hours (x1)
- To show the price of vaccine while registering privately (x1)
- To update of user's registered data (if required) (x1)
- To provide the staff capacity of each centers (x1)
- To manage remaining and wasted vaccines(x1)
- To regulate the price of vaccines at private centers.(x1)

## User Classes and Characteristics

The software includes specific classes pertaining to differential classes like:

- **Administrator:** In the system, the role of the Administrator is to supervise the staff, manage the user and staff requirements, manage vaccine stocks and storage. Each center will have administrators to run the drive successfully.
- **Staff:** In the system, staff includes three different types of people.
  1. The people who will take care of the registration and certification related tasks.
  2. Health care staff. The people who will give vaccines to users. The staff in this category should necessarily have the nursing degree or should have adequate medical knowledge and experience.
  3. The doctors. If any emergency will occur then people can consult them.
- **User:** In the system, the role of the user is like a normal person .User is one who is getting vaccinated or a guardian of a vaccinating child. Users can register online/offline themselves in private/government centers and get vaccinated. In specific cases users also have a facility for door-to-door vaccination. Every vaccinated user will have a unique e-certificate for confirmation of vaccination.

Different age groups:

- Age group -1 : People 1-4 years.
- Age group -2 : People 5-12 years.
- Age group -3 : People of age 13 years and older.

## Operating Environment

- The software used is pgAdmin. It is used to perform any sort of database administration required for a Postgres database.
- The system is a Windows 64 machine.
- The application/software should be such that it works efficiently in a heavy load environment.
- This vaccination software stores, analyses and keeps track of data limited to a well developed urban city.
- The app or vaccine management website should be compatible with other systems and work consistently with them.

## Product Functions

- The software being specified provides slot booking. It provides multiple slot timings throughout the working days and weekends.
- Keeps track of registered people and registration details. Registration will be based on unique identity.
- The slots can be booked both online, offline, and door-to-door for private organisations.
- Choice is provided for the vaccine they want to take based on availability. (From 2 certified chickenpox vaccines.)
- Keeps track of fully and partially vaccinated people.
- Keeps track of requirements of vaccines at different centers.
- Keeps track of price of vaccines and profit earned by private sectors.
- Keeps track of management staff, health care staff, consulting doctors etc.
- **Provides dynamic online slots.**
- Keeps track of how many users take vaccines during a given slot.
- Assures that people registering for 2nd dose have already taken 1st dose
- **Provide the slot rescheduling facility before 12 hours**
- Show the price of vaccine while registering privately
- Updates the user's registered data (if required)
- **Prohibits online registration - If the user is not able to reach in a given time slot for more than three times**
- Provide unique e-certificate id using which user can access e-certificate from the corresponding website
- **Provides vaccination center details based on locality - pincode or area name etc.**
- **Determines staff utilisation at the end of each day which will help to transfer, remove or add staff members from different centers**

## Privileges

User Classes	Accessible classes	Read	Insert	Update	Delete
Administrator	Administrator	Yes	No	No	No
	Staff	Yes	Yes	Yes	Yes
	User	Yes	Yes	Yes	Yes
Staff	Staff	Yes	No	No	No
	User	Yes	Yes	Yes	Yes
User	None	-	-	-	-

## **Assumptions**

- The software assumes that each center has adequate staff for managing the vaccination drives during different slots.
- It assumes that the staff is well qualified and works in the interest of patients.
- It assumes that the center has enough material resources like a facility for storing vaccines, adequate number of syringes, etc.
- The vaccine provided is in good condition and not the past expiry date.
- Transportation of vaccines to different centers is available at any given time.
- Employees do not have issues towards working in any vaccination center.

## **Business Constraints**

Business constraint defines business rules or business policy. Each organization has specific rules about the data store.

Some Common business constraints include:

### **Time Constraints:**

- Time constraints include not only the amount of time required to complete a task, but also the amount of time needed to obtain supplies.
- Once identified as a primary constraint, management can take steps to address time factors and improve vaccination performance. For example, larger supply orders might reduce time constraints imposed by long wait times. Similarly, allocating more space to vaccination centers might allow more people to be vaccinated, thereby reducing the total time needed to vaccinate the whole urban city.

### **Management and Staffing:**

- As the vaccination process grows, their staffing and management needs change, as well.
- This can constrain vaccination growth and productivity when staff cannot adapt to new demands.
- Management needs also change over time and sometimes poor management constrains growth by allocating resources inappropriately.

### **Financial Constraints:**

- Capital to run the system may provide delay in the functioning of the system.
- If the allocation of salaries to staff takes time, there may be worker strikes and the system gets shut down.

## Section2:

### Noun Analysis

**Table 1: Noun and Verb Analysis**

Nouns	Verbs
Registration ID	Center Stock
Varivax	Rescheduling
Citizen	Confirmation
Portal	Allocated staff
User ID	Consultant
Vaccine ID	register
e-portal	First dose details
Proquad	responsible
Chicken pox	Offline Approximation Analysis
Vaccine name	Administrator
Registration mode	Update user data
Center ID	Registered slots
Dose	Staff Utilization
Slot ID	Limit Registration
Pincode	scheduling
Emergency	Consulting doctors
Registered time and date	Registered user
User ID	Scheduling Appointment
appointment	Vaccination slot
area name	Second dose details
date	Registered slots
Registration	First dose details
User name	Allocated staff
User age	Consultant
User gender	Registered center

time	
User address	
door-to-door appointment	
User phone number	
User guardian	
Center id	
e-certificate	
People	
registering	
2nd dose	
1st dose	
Center name	
offline	
Center address	
vaccine	
online	
Employee id	
first dose	
Employee Name	
Center type	
Vaccine price	
Doctor ID	
Second dose	
Doctor Email	
Admin Landline Number	
Employee type	
Admin ID	
Maximum staff Capacity	

Slot ID	
Vaccination center	
time	
date	
day	
Online registration capacity	
Offline approximate registration	
E certificate ID	
Vaccine ID	
Staff ID	
Center ID	
date	
Vaccine ID	
Staff ID	
reg_id	
Attended users	
E-certificate ID	
Varivax vaccine	
Proquad vaccine	
Date	
Remaining proquad vaccines	
Remaining Varivax vaccines	
Wasted vaccines	
Doctor id	
Doctor name	
Doctor number	
Admin ID	
Admin name	

Admin number	
Staff ID	
Staff name	
Staff type	
Wasted Vaccine	
Number of doses	
Staff members	
Centers	
Employee number	

**Table 2:Accepted Noun & Verbs list**

Candidate Entity Set	Candidate Attribute Set	Candidate Relationship set
User	Staff ID	Consultant
Vaccination center	Maximum staff Capacity	Registered slots
Stock Details	Center ID	Allocated staff
Doctor Details	Center Name	Administrator
Admin Details	Center Address	Registered Users
Registration Details	Center Type	Offline approximation analysis
Slots	Vaccine Price	Center Stock
Staff Details	Staff Name	Vaccine dose details
Employee	Time	Registered center
	Date	
	Day	
	Proquad Vaccine	
	Online Registration Capacity	
	User ID	
	User Name	

	User age	
	Admin Landline	
	Employee ID	
	Doctor Email	
	User Gender	
	User address	
	User Phone number	
	Staff type	
	User Guardian	
	Vaccine Name	
	Employee type	
	Employee Name	
	Employee number	
	Wasted Vaccine	
	Registration ID	
	Varivax Vaccine	
	Vaccine ID	
	Date	
	Remaining Proquad Vaccine	
	Remaining Varivax Vaccine	
	Registration Mode	
	Center ID	
	Dose	
	Registration Time and Date	

**Table 3: Rejected Noun & Verbs list**

Noun	Rejected reason
Varivax	Irrelevant
Citizen	General
e-portal	Irrelevant
Slot ID	Duplicate
Proquad	Irrelevant
Chicken pox	General
Dose	General
door-to-door appointment	Attribute
Pincode	Irrelevant
People	Vague
Emergency	General
appointment	General
area name	General
offline	General
Registration	Association
User ID	Duplicate
Vaccine ID	Duplicate
center	General
online	Attribute
Staff ID	Duplicate
Center ID	Duplicate
Date	Duplicate
Time	Duplicate
First dose	Duplicate
Second dose	Duplicate

Doctor ID	Duplicate
Doctor name	Duplicate
Doctor number	Duplicate
Staff ID	Duplicate
Staff Name	Duplicate
Admin ID	Duplicate
Admin Name	Duplicate
Admin Number	Duplicate

Verbs	Rejected reason
Rescheduling	General
Confirmation	Irrelevant
Consultant	Duplicate
Register	Vague
First dose details	Duplicate
responsible	Irrelevant
Update user data	Irrelevant
Registered slots	Duplicate
Staff Utilization	General
Limit Registration	Irrelevant
scheduling	Duplicate
Consulting doctors	Duplicate
Scheduling Appointment	Duplicate
Vaccination slot	General

## II) Development of ER diagram

**Note:** For each candidate entity set, details in the bracket of the candidate relationship set indicates the relationship with that particular entity set.

<b>Candidate Entity Set</b>	<b>Candidate Attribute Set</b>	<b>Candidate Relationship set</b>
Vaccination center	Center ID Center Name Center Address Center Type Vaccine Price Maximum staff Capacity	Administrator (Admin details) Consultant (Doctor details) Center Stock (Stock Details) Offline approximation analysis (slots) Allocated staff (Staff Details) Registered center (Registration details)
Slots	Slot ID Time Date Day Online Registration Capacity	Registered slots (Registration details) Offline approximation analysis (Vaccination center)
User	User Id User Name User Age User Gender User Address User Phone Number User Guardian	Registered user (Registration details)
Registration Details	Registration ID Vaccine Name Registration Mode Center ID Dose Registration Time and Date	Registered slots (slots) Registered Users (Users) Registered center (Vaccination center) Vaccine dose details (Staff Details)
Staff Details	Staff type	Allocated staff (Vaccination center) Vaccine dose details (Registration details)
Doctor Details	Doctor Email	Consultant (Vaccination center)
Stock Details	Date Proquad Vaccine Varivax Vaccine Remaining Proquad Vaccine Remaining Varivax Vaccine Wasted Vaccine	Center stock (Vaccination center)
Admin Details	Admin landline number	Administrator (Vaccination center)
Employee	Employee ID Employee Name Employee Number Employee Type	

## Section3:

### ER-Diagrams all versions

## Finalizing ER diagrams.

### Identify weak entity sets:

Weak Entity set	Identifier Entity set	Identifier Entity
Stock details	Vaccination center	Center ID

### Identify Hierarchy:

Parent (Higher level Entity set)	Child (Higher level Entity set)
Employee	Doctor details
	Admin details
	Staff details

### Identify association types:

Relationship	Association type
Center Stock (Stock details to Vaccination center)	Many to one
Consultant (Doctor details to Vaccination center)	One to one
Offline Approx Analysis (Slots to Vaccination center)	Many to many
Administrator (Admin details to Vaccination center)	One to one
Allocated Staff (Staff details to Vaccination center)	Many to many
Registered User (Registration details to slots)	Many to one
Vaccine dose details (Registration details to staff details)	Many to many
Registered center (Registration details to vaccination center)	Many to one
Registered Slots (Slots to Registration details)	One to many

## **Identify Relationship types:**

### **Entity vs Attributes:**

Here doctor details and admin details can be selected as attributes in the vaccination center also, but we have taken them both as entity sets so that we can allow extra information pertaining to them.

### **Entity vs Relationship Set:**

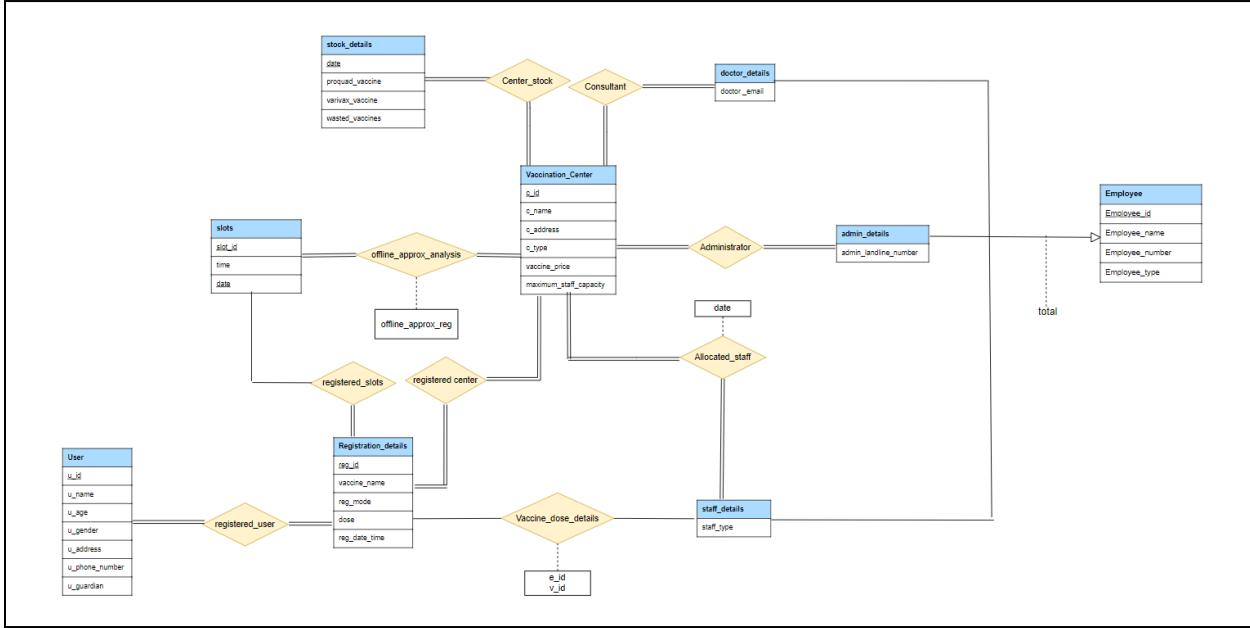
Allocated staff can be taken as an entity set instead of a relationship set. But to decrease redundancy, we have taken allocated staff as a relationship set between vaccination center and staff details. The associated attributes with the allocated staff relationship set are date and attended users.

## **Participation:**

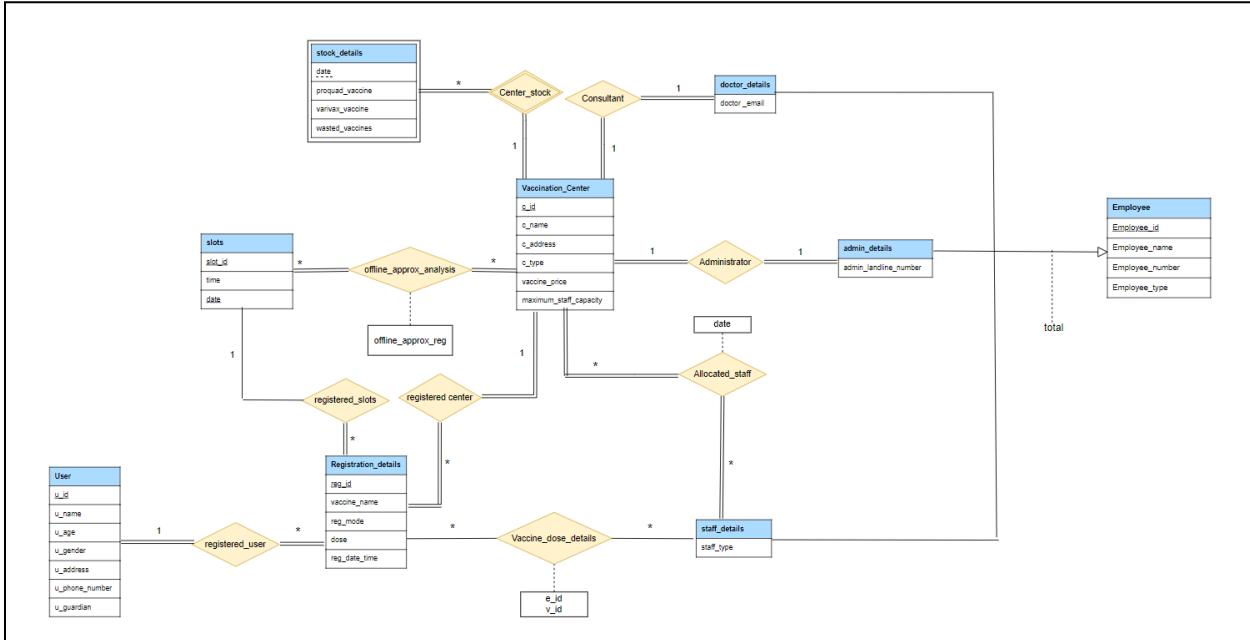
<b>Relationship</b>	<b>Participation</b>
Center Stock (Stock details to Vaccination center)	Total
Consultant (Doctor details to Vaccination center)	Total
Offline Approx Analysis (Slots to Vaccination center)	Total
Administrator (Admin details to Vaccination center)	Total
Allocated Staff (Staff details to Vaccination center)	Total
Registered User (Registration details to slots)	Total
Vaccine dose details (Registration details to staff details)	Partial
Registered center (Registration details to vaccination center)	Total
Registered Slots (Slots to Registration details)	Slots Partial Registration details total

## E-R diagrams:

### Version 1:



### Version 2 (final):



## Section 4:

### Conversion of Final ER-Diagram to Relational Model

## Mapping E-R Model to Relational Model

- user (u\_id, u\_name, u\_age, u\_gender, u\_address, u\_phone\_number, u\_guardian)
- registration\_details (reg\_id, u\_id, slot\_id, date, vaccine\_name, reg\_mode, c\_id, dose, reg\_date\_time)
- vaccine\_dose\_details (reg\_id, employee\_id, e\_id, v\_id)
- stock\_details (c\_id, date, proquad\_vaccine, varivax\_vaccine, wasted\_vaccine)
- slots (slot\_id, date, time)
- vaccination\_center (c\_id, c\_name, c\_address, c\_type, vaccine\_price, maximum\_staff\_capacity)
- staff\_details (employee\_id, staff\_type)
- allocated\_staff (employee\_id, c\_id, date)
- offline\_approx\_analysis (slot\_id, date, c\_id, offline\_approx\_reg)
- doctor\_details (employee\_id, c\_id, doctor\_email)
- admin\_details (employee\_id, c\_id, admin\_landline\_number)
- employee (employee\_id, employee\_name, employee\_type, employee\_number)

## Section 5:

### Normalization and Schema Refinement

## Existing Schema:

- user (u\_id, u\_name, u\_age, u\_gender, u\_address, u\_phone\_number, u\_guardian)
- registration\_details (reg\_id, u\_id, slot\_id, date, vaccine\_name, reg\_mode, c\_id, dose, reg\_date\_time)
- vaccine\_dose\_details (reg\_id, employee\_id, e\_id, v\_id)
- stock\_details (c\_id, date, proquad\_vaccine, varivax\_vaccine, wasted\_vaccine)
- slots (slot\_id, date, time)
- vaccination\_center (c\_id, c\_name, c\_address, c\_type, vaccine\_price, maximum\_staff\_capacity)
- staff\_details (employee\_id, staff\_type)
- allocated\_staff (employee\_id, c\_id, date)
- offline\_approx\_analysis (slot\_id, date, c\_id, offline\_approx\_reg)
- doctor\_details (employee\_id, c\_id, doctor\_email)
- admin\_details (employee\_id, c\_id, admin\_landline\_number)
- employee (employee\_id, employee\_name, employee\_type, employee\_number)

## Dependencies (PK, FK, Functional Dependencies) for each relation:

PK Dependency
$u\_id \rightarrow u\_name, u\_age, u\_gender, u\_address, u\_phone\_number, u\_guardian$
$reg\_id \rightarrow u\_id, slot\_id, date, vaccine\_name, reg\_mode, c\_id, dose, reg\_date\_time$
$\{reg\_id, employee\_id\} \rightarrow e\_id, v\_id$
$c\_id \rightarrow c\_name, c\_address, c\_type, vaccine\_price, maximum\_staff\_capacity$
$\{slot\_id, date\} \rightarrow time$
$\{c\_id, date\} \rightarrow proquad\_vaccine, varivax\_vaccine, wasted\_vaccine$
$employee\_id \rightarrow staff\_type$
$employee\_id \rightarrow c\_id, admin\_landline\_number$
$\{slot\_id, date, c\_id\} \rightarrow offline\_approx\_reg$
$employee\_id \rightarrow c\_id, doctor\_email$
$employee\_id \rightarrow employee\_name, employee\_type, employee\_number$

Functional Dependency
$e\_id \rightarrow v\_id$
$\text{slot id} \rightarrow \text{time}$
$\text{reg\_id} \rightarrow e\_id, v\_id$

## Redundancies and anomalies:

### Relation slots:

**Redundancy :** The same date is repeated for different slot\_id and different times.

For different dates, the same slot\_id and time are repeated.

#### **Anomalies:**

**Insert :** We can not insert a slot time unless it is not assigned to a date.

**Update :** To update timing of a slot\_id, we will have to update timings for all the dates.

**Delete :** If we delete tuples for a particular date, timing information of the slot will also be deleted.

### Relation vaccine\_dose\_details:

**Redundancy :** The details like e\_id, v\_id and reg\_id will be repeated for the details of different types of employees associated with the user (employee\_id).

#### **Anomalies:**

**Insert :** We can not insert dose details unless it is assigned to an employee.

**Update :** To update dose details, we will have to update it for every employee\_id associated with that registration.

**Delete :** If we delete a tuple for a particular employee\_id, other dose details associated with that employee\_id will also be deleted.

The above anomalies and redundancies are due to partial dependencies which will be removed in 2NF form.

## Normalization of the database up to 1NF (scalar values)

### Composite attributes:

- In relation slots, time contains a time range : start\_time : end\_time
- In relation registration\_details, reg\_date\_time contains : reg\_date and reg\_time.

Due to these composite attributes we cannot retrieve the data we want easily. For retrieving that particular data we have to extract the data from the composite data.

To normalize schema up to 1NF form: we can divide the attributes into two parts.

New, relations will be: slots (slot\_id, date, start\_time, end\_time)

and registration\_details (reg\_id, u\_id, slot\_id, date, vaccine\_name, reg\_mode, c\_id, dose, reg\_date, reg\_time).

## Schema in 1NF form:

- user (u\_id, u\_name, u\_age, u\_gender, u\_address, u\_phone\_number, u\_guardian)
- registration\_details (reg\_id, u\_id, slot\_id, date, vaccine\_name, reg\_mode, c\_id, dose, reg\_date, reg\_time)
- vaccine\_dose\_details (reg\_id, employee\_id, e\_id, v\_id)
- stock\_details (c\_id, date, proquad\_vaccine, varivax\_vaccine, wasted\_vaccine)
- slots (slot\_id, date, start\_time, end\_time)
- vaccination\_center (c\_id, c\_name, c\_address, c\_type, vaccine\_price, maximum\_staff\_capacity)
- staff\_details (employee\_id, staff\_type)
- allocated\_staff (employee\_id, c\_id, date)
- offline\_approx\_analysis (slot\_id, date, c\_id, offline\_approx\_reg)
- doctor\_details (employee\_id, c\_id, doctor\_email)
- admin\_details (employee\_id, c\_id, admin\_landline\_number)
- employee (employee\_id, employee\_name, employee\_type, employee\_number)

There are still anomalies and redundancy related to partial dependencies in 1NF form which were also there before 1NF form.

## Relation slots:

**Redundancy :** The same date is repeated for different slot\_id and different times.

For different dates, the same slot\_id and time are repeated.

### Anomalies:

**Insert :** We can not insert a slot time unless it is not assigned to a date.

**Update :** To update timing of a slot\_id, we will have to update timings for all the dates.

**Delete :** If we delete tuples for a particular date, timing information of the slot will also be deleted.

## Relation vaccine\_dose\_details:

**Redundancy :** The details like e\_id, v\_id and reg\_id will be repeated for the details of different types of employees associated with the user (employee\_id).

### Anomalies:

**Insert :** We can not insert dose details unless it is assigned to an employee.

**Update :** To update dose details, we will have to update it for every employee\_id associated with that registration.

**Delete :** If we delete a tuple for a particular employee\_id, other dose details associated with that employee\_id will also be deleted.

## Normalization of the database further to 2NF (Remove Partial Dependencies)

- In the slots table, the attributes ‘start\_time’ and ‘end\_time’ depends only on ‘slot\_id’  
Slots table will be decomposed into two tables:  
slots\_time(slot\_id, start\_time, end\_time) and slots(slot\_id, date)
- In the vaccine\_dose\_details table, the attribute ‘e\_id’ and ‘v\_id’ depend only on ‘reg\_id’  
vaccine\_dose\_details table will be decomposed into two tables:  
vaccine\_employee(reg\_id, employee\_id) and vaccine\_dose\_details(reg\_id, e\_id, v\_id)

## List of redundancies exiting for the schema in 2NF

- There are no such redundancies existing for schema in 2NF and there are no anomalies.

## Normalization of the database further to 3NF/BCNF (Remove Transitive Dependencies)

- All the non prime attributes are directly depending on the primary key in the 2NF schema.
- So, there are no transitive dependencies. Hence, the schema is in 3NF form.
- Also, Primary keys are not dependent on non-prime attributes in any relation and hence, schema is in BCNF form.

## Final Schema (BCNF)

- user (u\_id, u\_name, u\_age, u\_gender, u\_address, u\_phone\_number, u\_guardian)
- registration\_details (reg\_id, u\_id, slot\_id, date, vaccine\_name, reg\_mode, c\_id, dose, reg\_date, reg\_time)
- slots\_time(slot\_id, start\_time, end\_time)
- slots(slot\_id, date)
- vaccine\_employee(reg\_id, employee\_id)
- vaccine\_dose\_details(reg\_id, e\_id, v\_id)
- stock\_details (c\_id, date, proquad\_vaccine, varivax\_vaccine, wasted\_vaccine)
- vaccination\_center (c\_id, c\_name, c\_address, c\_type, vaccine\_price, maximum\_staff\_capacity)
- staff\_details (employee\_id, staff\_type)
- allocated\_staff (employee\_id, c\_id, date)
- offline\_approx\_analysis (slot\_id, date, c\_id, offline\_approx\_reg)
- doctor\_details (employee\_id, c\_id, doctor\_email)
- admin\_details (employee\_id, c\_id, admin\_landline\_number)
- employee (employee\_id, employee\_name, employee\_type, employee\_number)

## Section6:

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

## Final DDL Scripts

### Order in which tables are created:

- user
- slots\_time
- slots
- Vaccination\_center
- Registration\_details
- Employee
- Staff\_details
- Vaccine\_dose\_details
- Vaccine\_employee
- stock\_details
- allocated\_staff
- offline\_approx\_analysis
- Doctor\_details
- Admin\_details

#### user:

```
CREATE TABLE "user"  
(  
    u_id bigint NOT NULL,  
    u_name character(20) NOT NULL,  
    u_age numeric(4,2) NOT NULL,  
    u_gender "char" NOT NULL,  
    u_address character varying(100),  
    u_phone_number character(15) NOT NULL,  
    u_guardian character(20),  
    PRIMARY KEY (u_id)  
)
```

#### slots\_time:

```
CREATE TABLE "slots_time"  
(  
    slot_id integer NOT NULL,  
    start_time time without time zone NOT NULL,  
    end_time time without time zone NOT NULL,  
    PRIMARY KEY (slot_id)  
)
```

**slots:**

```
CREATE TABLE "slots"
(
    slot_id integer NOT NULL,
    date date NOT NULL,
    PRIMARY KEY (slot_id, date),
    FOREIGN KEY (slot_id) REFERENCES slots_time(slot_id)
        ON UPDATE cascade
        ON DELETE cascade
)
```

**vaccination\_center:**

```
CREATE TABLE "vaccination_center"
(
    c_id bigint NOT NULL,
    c_name character(25) NOT NULL,
    c_address character(10) NOT NULL,
    c_type character(10) NOT NULL,
    vaccine_price numeric(6, 2) NOT NULL,
    max_staff_capacity integer,
    PRIMARY KEY (c_id)
)
```

**registration\_details:**

```
CREATE TABLE "registration_details"
(
    reg_id bigint NOT NULL,
    u_id bigint NOT NULL,
    slot_id integer NOT NULL,
    date date NOT NULL,
    vaccine_name character(15) NOT NULL,
    reg_mode character(15) NOT NULL,
    c_id integer NOT NULL,
    dose "char" NOT NULL,
    reg_date date NOT NULL,
    reg_time time without time zone NOT NULL,
    PRIMARY KEY (reg_id),
    FOREIGN KEY (date, slot_id) REFERENCES slots (date, slot_id)
        ON UPDATE cascade
        ON DELETE NO ACTION,
    FOREIGN KEY (c_id) REFERENCES vaccination_center (c_id)
        ON UPDATE cascade
        ON DELETE NO ACTION,
```

```
FOREIGN KEY (u_id) REFERENCES user (u_id)
    ON UPDATE cascade
    ON DELETE NO ACTION,
    CHECK (dose in ('1', '2'))
)
```

### **employee:**

```
CREATE TABLE "employee"
(
    employee_id bigint NOT NULL,
    employee_name character(20) NOT NULL,
    employee_type character(20) NOT NULL,
    employee_number character(15) NOT NULL,
    PRIMARY KEY (employee_id)
)
```

### **staff\_details:**

```
CREATE TABLE "staff_details"
(
    employee_id bigint NOT NULL,
    staff_type character(20) NOT NULL,
    PRIMARY KEY (employee_id),
    FOREIGN KEY (employee_id) REFERENCES employee (employee_id)
        ON UPDATE cascade
        ON DELETE cascade
)
```

### **vaccine\_dose\_details:**

```
CREATE TABLE "vaccine_dose_details"
(
    reg_id bigint NOT NULL,
    e_id bigint NOT NULL,
    v_id bigint NOT NULL,
    PRIMARY KEY (reg_id),
    FOREIGN KEY (reg_id) REFERENCES registration_details (reg_id)
        ON UPDATE cascade
        ON DELETE NO ACTION
)
```

### **vaccine\_employee:**

```
CREATE TABLE "vaccine_employee"
(
    reg_id bigint NOT NULL,
    employee_id bigint NOT NULL,
    PRIMARY KEY (reg_id, employee_id),
    FOREIGN KEY (employee_id) REFERENCES staff_details (employee_id)
        ON UPDATE cascade
        ON DELETE NO ACTION,
    FOREIGN KEY (reg_id) REFERENCES registration_details (reg_id)
        ON UPDATE cascade
        ON DELETE NO ACTION,
    FOREIGN KEY (reg_id) REFERENCES vaccine_dose_details (reg_id)
        ON UPDATE cascade
        ON DELETE NO ACTION
)
```

### **stock\_details:**

```
CREATE TABLE "stock_details"
(
    c_id bigint NOT NULL,
    date date NOT NULL,
    proquad_vaccine bigint NOT NULL,
    varivax_vaccine bigint NOT NULL,
    wasted_vaccines integer NOT NULL,
    PRIMARY KEY (c_id, date),
    FOREIGN KEY (c_id) REFERENCES vaccination_center (c_id)
        ON UPDATE cascade
        ON DELETE NO ACTION
)
```

### **allocated\_staff:**

```
CREATE TABLE "allocated_staff"
(
    employee_id bigint NOT NULL,
    c_id bigint NOT NULL,
    date date NOT NULL,
    PRIMARY KEY (employee_id, c_id, date),
    FOREIGN KEY (c_id) REFERENCES vaccination_center (c_id)
        ON UPDATE cascade
        ON DELETE NO ACTION,
    FOREIGN KEY (employee_id) REFERENCES staff_details (employee_id)
        ON UPDATE cascade
        ON DELETE NO ACTION
)
```

### **offline\_approx\_analysis:**

```
CREATE TABLE "offline_approx_analysis"
(
    slot_id integer NOT NULL,
    date date NOT NULL,
    c_id bigint NOT NULL,
    offline_approx_reg integer NOT NULL,
    PRIMARY KEY (slot_id, date, c_id),
    FOREIGN KEY (date, slot_id) REFERENCES slots (date, slot_id)
        ON UPDATE cascade
        ON DELETE cascade,
    FOREIGN KEY (c_id) REFERENCES vaccination_center (c_id)
        ON UPDATE cascade
        ON DELETE cascade
)
```

### **doctor\_details:**

```
CREATE TABLE "doctor_details"
(
    employee_id bigint NOT NULL,
    c_id bigint NOT NULL,
    doctor_email character(50),
    PRIMARY KEY (employee_id),
    FOREIGN KEY (c_id) REFERENCES vaccination_center (c_id)
        ON UPDATE cascade
        ON DELETE NO ACTION,
    FOREIGN KEY (employee_id) REFERENCES employee (employee_id)
        ON UPDATE cascade
        ON DELETE cascade
)
```

### **admin\_details:**

```
CREATE TABLE "admin_details"
(
    employee_id bigint NOT NULL,
    c_id bigint NOT NULL,
    landline_number character(15) NOT NULL,
    PRIMARY KEY (employee_id),
    FOREIGN KEY (c_id) REFERENCES vaccination_center (c_id)
        ON UPDATE cascade
        ON DELETE NO ACTION,
```

```

FOREIGN KEY (employee_id) REFERENCES employee (employee_id)
    ON UPDATE cascade
    ON DELETE cascade
)

```

## Insert Statements and screenshots of loaded data

```

copy user
from 'D:\dbms\user.csv'
csv header

```

The screenshot shows the pgAdmin 4 interface with the 'user' table selected in the left sidebar. The 'Data Output' tab is active, displaying the results of the query 'select \* from "user";'. The results show 12 rows of data with columns: u\_id, u\_name, and u\_age.

u_id	u_name	u_age
1	Xavier	2
2	Louis	3
3	Beatrisa	2.00 F
4	Chickie	2.25 F
5	Drew	5.00 M
6	Golda	9.50 F
7	Sharleen	4.00 F
8	Lauree	2.50 F
9	Alyn	13.00 M
10	Arther	2.00 M
11	Harman	5.00 M
12	Katuscha	6.00 F

```
copy slots_time
from 'D:\dbms\slots_time.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with a node for 't6\_vc\_mn' expanded, showing various objects like Collations, Domains, FTS Configurations, etc. The main area has a 'Query Editor' tab active with the following SQL code:

```
1 select *
2 from slots_time;
```

The 'Messages' panel indicates the query was successfully run with a runtime of 252 msec and 2 rows affected.

The 'Data Output' panel shows the results of the query:

slot_id	start_time	end_time
1	09:00:00	12:00:00
2	14:00:00	17:00:00

```
copy slots
from 'D:\dbms\slots.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with a node for 't6\_vc\_mn' expanded, showing various objects like Collations, Domains, FTS Configurations, etc. The main area has a 'Query Editor' tab active with the following SQL code:

```
1 select *
2 from slots;
```

The 'Messages' panel indicates the query was successfully run with a runtime of 99 msec and 266 rows affected.

The 'Data Output' panel shows the results of the query:

slot_id	date
1	2021-07-21
2	2021-07-22
3	2021-07-23
4	2021-07-24
5	2021-07-25
6	2021-07-26
7	2021-07-27
8	2021-07-28
9	2021-07-29
10	2021-07-30
11	2021-07-31
12	2021-07-21

```
copy vaccination_center
from 'D:\dbms\vaccination_center.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under 'Tables (14)', the 'vaccination\_center' table is selected. The main area displays a query editor with the following SQL code:

```
1 select *
2 from vaccination_center;
```

Below the query editor, the 'Messages' section shows the output: "Successfully run. Total query runtime: 848 msec. 20 rows affected." The results are presented in a data grid:

c_id	[PK] bigint	c_name	character (25)	c_address	character (10)	c_type	character (10)	vaccine_price	numeric (6,2)	max_staff_capacity	integer
1	1	Ashram road UHC	ahmedabad	ahmedabad		government		0.00		3	
2	3	Satellite UHC	ahmedabad	private		government		1500.00		3	
3	4	Jodhpur UHC	ahmedabad	ahmedabad		government		0.00		3	
4	5	Joshipura UHC	junagadh	private		government		1500.00		3	
5	6	Girnar UHC	junagadh	junagadh		government		0.00		2	
6	7	Kalva Chok UHC	junagadh	government		government		0.00		3	
7	8	Gandhidham UHC	kutch	kutch		government		0.00		3	
8	9	Anjar UHC	kutch	government		government		0.00		2	
9	10	GIDC UHC	rajkot	rajkot		government		0.00		3	
10	11	Nana Mava UHC	rajkot	government		government		0.00		3	
11	12	Ram Krishna Nagar UHC	rajkot	private		government		1500.00		3	
12	13	Mandvi UHC	vadodara	government		government		0.00		3	
13	14	Hari Naar UHC	vadodara	private		government		1500.00		3	

```
copy stock_details
from 'D:\dbms\stock_details.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under 'Tables (14)', the 'stock\_details' table is selected. The main area displays a query editor with the following SQL code:

```
1 select *
2 from stock_details;
```

Below the query editor, the 'Messages' section shows the output: "Successfully run. Total query runtime: 115 msec. 40 rows affected." The results are presented in a data grid:

c_id	[PK] bigint	date	[PK] date	proquad_vaccine	bigint	varivax_vaccine	bigint	wasted_vaccines	integer
1	1	2021-07-21		6		4		0	
2	1	2021-09-10		7		3		0	
3	2	2021-07-21		8		5		1	
4	2	2021-09-10		8		5		0	
5	3	2021-07-21		6		4		0	
6	3	2021-09-10		6		4		0	
7	4	2021-07-21		7		3		1	
8	4	2021-09-10		7		3		0	
9	5	2021-07-21		5		5		0	
10	5	2021-09-10		6		4		0	
11	6	2021-07-21		4		3		0	
12	6	2021-09-10		4		3		0	
13	7	2021-07-21		6		4		1	

```
copy allocated_staff
from 'D:\dbms\allocated_staff.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under 'Servers (1) > PostgreSQL 13 > Databases (5) > S2\_T6\_vaccine\_drive\_management > Tables (14)', the 'allocated\_staff' table is selected. The main pane displays a query editor with the following SQL:

```
1 select *
2 from allocated_staff;
```

The 'Messages' section indicates the query was successfully run with a total runtime of 53 msec and 120 rows affected.

The 'Data Output' tab shows the results of the query:

employee_id	c_id	date
1	1	2 2021-07-21
2	2	18 2021-07-21
3	3	1 2021-07-21
4	4	2 2021-07-21
5	5	3 2021-07-21
6	6	4 2021-07-21
7	7	5 2021-07-21
8	8	6 2021-07-21
9	9	7 2021-07-21
10	10	8 2021-07-21
11	11	9 2021-07-21
12	12	10 2021-07-21
13	13	11 2021-07-21

```
copy offline_approx_analysis
from 'D:\dbms\offline_approx_analysis.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. In the left sidebar, under 'Servers (1) > PostgreSQL 13 > Databases (5) > S2\_T6\_vaccine\_drive\_management > Tables (14)', the 'offline\_approx\_analysis' table is selected. The main pane displays a query editor with the following SQL:

```
1 select *
2 from offline_approx_analysis;
```

The 'Messages' section indicates the query was successfully run with a total runtime of 56 msec and 160 rows affected.

The 'Data Output' tab shows the results of the query:

slot_id	date	c_id	offline_approx_reg
1	1 2021-07-21	1	2
2	1 2021-08-01	1	1
3	1 2021-09-01	1	3
4	1 2021-10-01	1	2
5	2 2021-07-21	1	2
6	2 2021-08-01	1	3
7	2 2021-09-01	1	2
8	2 2021-10-01	1	1
9	1 2021-07-21	2	3
10	1 2021-08-01	2	3
11	1 2021-09-01	2	4
12	1 2021-10-01	2	3
13	2 2021-07-21	2	4

```
copy doctor_details
from 'D:\dbms\doctor_details.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema, including tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', etc. The central area shows a query editor with the following SQL code:

```
1 set search_path to t6_vc_mn;
2 select *
3 from doctor_details;
```

The 'Messages' section indicates the query was successfully run with a total runtime of 202 msec and 20 rows affected.

The 'Data Output' tab shows the results of the query, listing 13 rows of data with columns: employee\_id, c\_id, and doctor\_email. The data is as follows:

	employee_id	c_id	doctor_email
1	61	1	faloshikin0@gmail.com
2	62	2	mmcginn1@gmail.com
3	63	3	rnnutkin2@yahoo.com
4	64	4	lsouthard3@gmail.com
5	65	5	jburnyeat4@gmail.com
6	66	6	jmonshali5@yahoo.com
7	67	7	mbrosch6@gmail.com
8	68	8	zhalows7@gmail.com
9	69	9	bbarbe8@yahoo.com
10	70	10	lokey9@yahoo.com
11	71	11	apennuzia@gmail.com
12	72	12	epretsellb@hotmail.com
13	73	13	nneilanshaili@mit.edu

```
copy admin_details
from 'D:\dbms\admin_details.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema, including tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', etc. The central area shows a query editor with the following SQL code:

```
1 set search_path to t6_vc_mn;
2 select *
3 from admin_details;
```

The 'Messages' section indicates the query was successfully run with a total runtime of 82 msec and 20 rows affected.

The 'Data Output' tab shows the results of the query, listing 13 rows of data with columns: employee\_id, c\_id, and landline\_number. The data is as follows:

	employee_id	c_id	landline_number
1	81	1	3307342317
2	82	2	7756822908
3	83	3	8993920978
4	84	4	7285091896
5	85	5	4333479411
6	86	6	3923969282
7	87	7	1333951354
8	88	8	7099028660
9	89	9	2764463195
10	90	10	9738408729
11	91	11	4765188613
12	92	12	2517112079
13	93	13	8032927386

```
copy employee
from 'D:\dbms\employee.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema, including tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', 'employee', etc. The central area shows a query editor with the following SQL code:

```
1 set search_path to t6_vc_mn;
2 select *
3 from employee;
```

The 'Messages' section indicates the query was successfully run with a total runtime of 84 msec and 100 rows affected.

The 'Data Output' tab shows the results of the query as a table:

employee_id	employee_name	employee_type	employee_number
1	Blinny	staff	5597664832
2	Christie	staff	3436735001
3	Ingram	staff	3153639900
4	Garek	staff	9585018177
5	Cassandra	staff	9402132963
6	Reuben	staff	9069720705
7	Florry	staff	1409203027
8	Henrietta	staff	3534907950
9	Melcent	staff	5912232479
10	Abbie	staff	7391893985
11	Lazaro	staff	2335022149
12	Gayle	staff	7069602520
13	Ara	staff	6131293879

```
copy staff_details
from 'D:\dbms\staff_details.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema, including tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', 'employee', etc. The central area shows a query editor with the following SQL code:

```
1 set search_path to t6_vc_mn;
2 select *
3 from staff_details;
```

The 'Messages' section indicates the query was successfully run with a total runtime of 99 msec and 60 rows affected.

The 'Data Output' tab shows the results of the query as a table:

employee_id	staff_type
1	management
2	management
3	management
4	management
5	management
6	management
7	management
8	management
9	management
10	management
11	management
12	management

```
copy vaccine_dose_details
from 'D:\dbms\vaccine_dose_details.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema, including tables like admin\_details, allocated\_staff, doctor\_details, employee, offline\_approx\_analys, registration\_details, slots, slots\_time, staff\_details, stock\_details, user, vaccination\_center, vaccine\_dose\_details, and vaccine\_employee. The central area shows the SQL query editor with the following command:

```
1 set search_path to t6_vc_mn;
2 select *
3 from vaccine_dose_details;
```

The 'Messages' section indicates the query was successfully run with a total runtime of 83 msec and 104 rows affected.

The 'Data Output' tab is selected, showing the results of the query:

reg_id	e_id	v_id
1	1	70672
2	3	26004
3	4	86094
4	5	13027
5	6	49137
6	7	49771
7	8	17460
8	9	97623
9	10	46544
10	11	90953
11	13	27672
12	14	34213
13	15	49842

```
copy registration_details
from 'D:\dbms\registration_details.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema, including tables like admin\_details, allocated\_staff, doctor\_details, employee, offline\_approx\_analys, registration\_details, slots, slots\_time, staff\_details, stock\_details, user, vaccination\_center, vaccine\_dose\_details, and vaccine\_employee. The central area shows the SQL query editor with the following command:

```
1 set search_path to t6_vc_mn;
2 select *
3 from registration_details;
```

The 'Messages' section indicates the query was successfully run with a total runtime of 113 msec and 110 rows affected.

The 'Data Output' tab is selected, showing the results of the query:

reg_id	u_id	slot_id	date	vaccine_name	reg_mode	c_id	dose	reg_date	reg_time
1	1	1	1 2021-07-21	proquad	online	5	1	2021-07-20	17:05:00
2	2	2	2 2021-07-22	proquad	online	1	1	2021-07-21	05:50:00
3	3	3	2 2021-07-22	proquad	online	17	1	2021-07-21	03:09:00
4	4	4	2 2021-07-24	proquad	online	10	1	2021-07-23	12:13:00
5	5	2	2 2021-07-25	proquad	online	1	1	2021-07-24	06:00:00
6	6	5	2 2021-07-26	proquad	online	2	1	2021-07-25	03:14:00
7	7	6	1 2021-07-27	proquad	online	13	1	2021-07-26	11:50:00
8	8	7	1 2021-07-28	proquad	offline	18	1	2021-07-27	14:55:00
9	9	8	1 2021-07-29	proquad	online	3	1	2021-07-28	08:21:00
10	10	9	2 2021-07-30	varivax	online	11	1	2021-07-29	07:00:00
11	11	10	1 2021-07-31	proquad	online	12	1	2021-07-30	11:26:00
12	12	11	2 2021-08-01	proquad	online	6	1	2021-07-31	18:04:00
13	13	12	1 2021-08-02	noninjard	online	5	1	2021-08-01	03:20:00

```
copy vaccine_employee
from 'D:\dbms\vaccine_employee.csv'
csv header
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure with the 'Tables (14)' node expanded, showing various tables like 'admin\_details', 'allocated\_staff', etc. The 'registration\_details' table is currently selected. The main pane is the 'Query Editor' which contains the following SQL code:

```
1 set search_path to t6_vc_mn;
2 select *
3 from vaccine_employee;
```

Below the query editor, the 'Messages' section indicates a successful run with a total runtime of 81 msec and 287 rows affected.

reg_id	employee_id
1	1
2	1
3	3
4	3
5	4
6	4
7	5
8	5
9	6
10	6
11	7
12	7
13	8

## Trigger Functions

1) Prohibition of online registration if a user tries to register more than 3 times and not reach to take the vaccine.

**Code:**

```
declare
reg_count bigint;

begin

select count(u_id) into reg_count from t6_vc_mn."registration_details"
where "u_id"=New."u_id" and "dose"=New."dose";

if (reg_count=3 and New."reg_mode"='online') then
Raise notice 'You are not allowed to register online because your online registration limit is exceeded. Please visit
your nearest vaccination center to take vaccine.';
return null;
else
return new;
end if;

end
```

**Output:**

59

**2) Prohibition of registration if a user tries to register for the second dose before completion of one month after the first dose.**

**Code:**

```
declare
d date;

begin

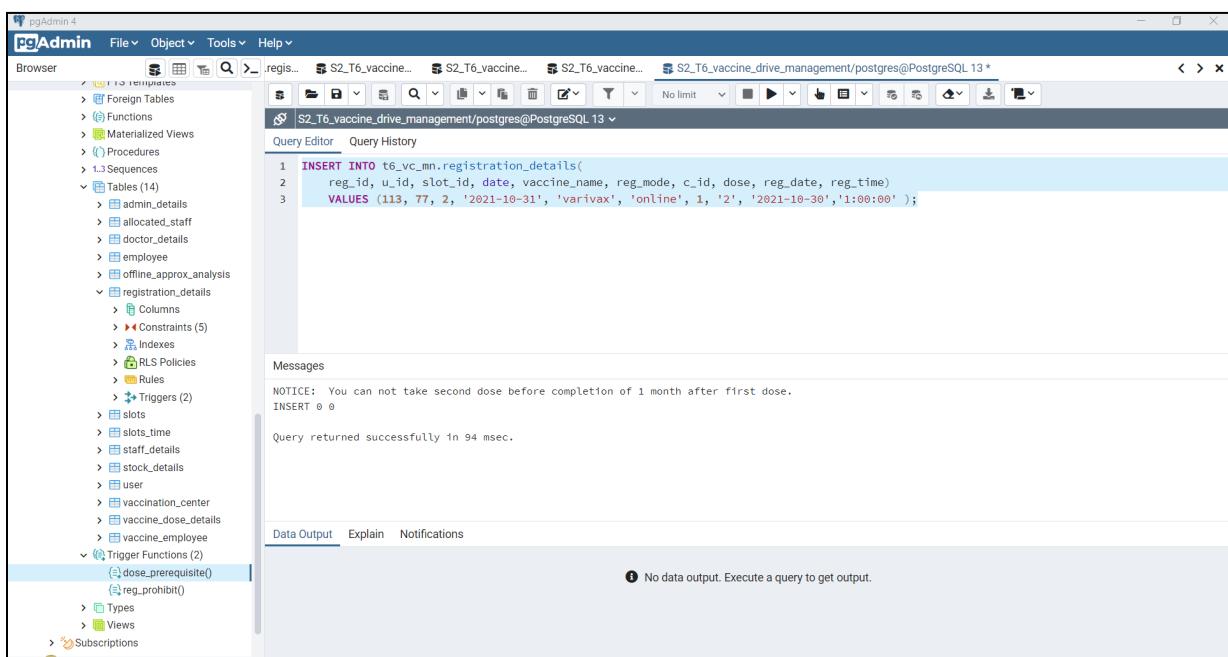
if(new."dose"='2') then
select "date" into d
from t6_vc_mn."registration_details" inner join t6_vc_mn."vaccine_dose_details"
on t6_vc_mn."registration_details"."reg_id" = t6_vc_mn."vaccine_dose_details"."reg_id"
where "u_id"=New."u_id" and "dose"='1';

if(new."date"-d < 30) then
Raise notice 'You can not take second dose before completion of 1 month after first dose.';
return null;
end if;

return new;
else
return new;
end if;

end
```

**Output:**



### 3) Prohibit users from registering for the second dose without taking the first dose.

**Code:**

```
declare  
c int;
```

```
begin
```

```
if (new."dose"='2') then  
select count(t6_vc_mn."vaccine_dose_details"."reg_id") into c  
from t6_vc_mn."registration_details" inner join t6_vc_mn."vaccine_dose_details"  
on t6_vc_mn."registration_details"."reg_id" = t6_vc_mn."vaccine_dose_details"."reg_id"  
where "u_id"=New."u_id" and "dose"='1';
```

```
if (c=0) then  
Raise notice 'You have not taken first dose.';  
return null;  
else
```

```
return new;  
end if;
```

```
else  
return new;
```

```
end if;  
end
```

**Output:**

The screenshot shows the pgAdmin 4 interface with the following details:

- Browser:** Shows the schema tree for the database, including tables like allocated\_staff, doctor\_details, employee, offline\_approx\_analysis, registration\_details, slots, slots\_time, staff\_details, stock\_details, user, vaccination\_center, vaccine\_dose\_details, vaccine\_employee, and trigger functions like dose\_prerequisite2(), dose\_prerequisite(), and reg\_prohibit().
- Query Editor:** Displays the SQL query:

```
1 INSERT INTO t6_vc_mn.registration_details(  
2     reg_id, u_id, slot_id, date, vaccine_name, reg_mode, c_id, dose, reg_date, reg_time)  
3 VALUES (113, 81, 2, '2021-10-31', 'varivax', 'online', 1, '2', '2021-10-30', '1:00:00');
```
- Messages:** Shows the output of the query:

```
NOTICE: You have not taken first dose.  
INSERT 0 0
```

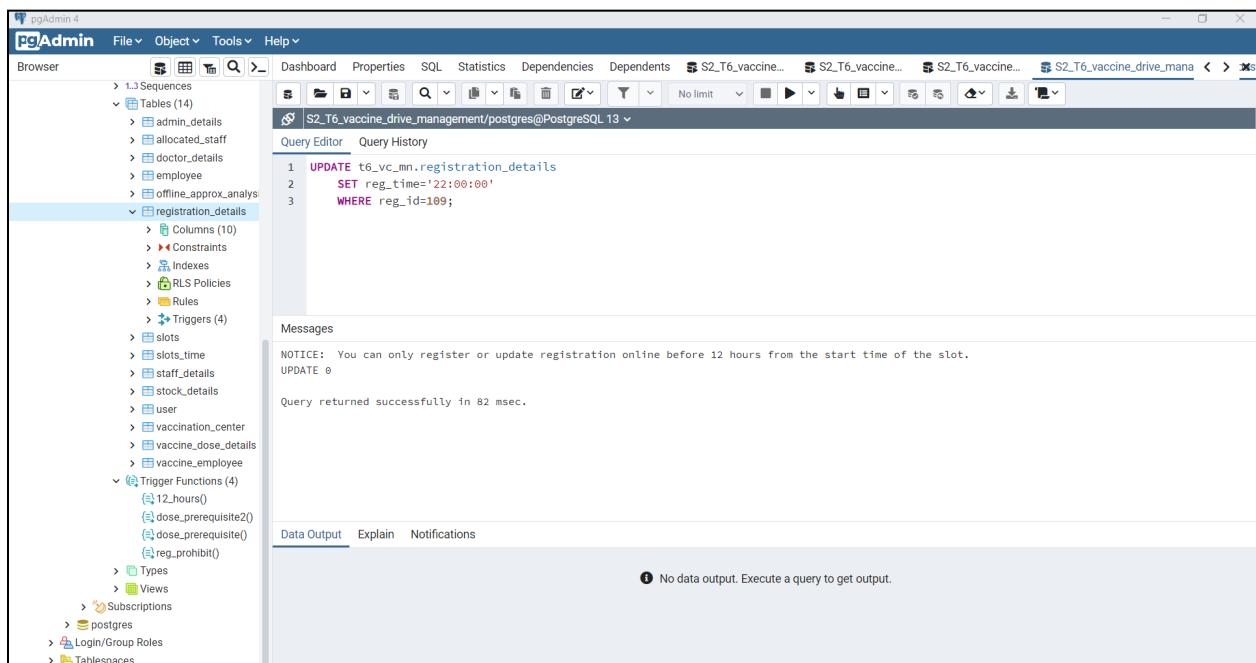
Query returned successfully in 67 msec.
- Data Output:** Shows a message: "No data output. Execute a query to get output."

#### 4) Provide a rescheduling facility to users before 12 hours of registered slot.

**Code:**

```
begin
if(new."reg_mode"='online') then
    if(new."date" - new."reg_date"=1 and new."slot_id"=1 and new."reg_time">>'21:00:00') then
        Raise notice 'You can only register or update registration online before 12 hours from the start
time of the slot.';
        return null;
    end if;
    if(new."date"-new."reg_date"=0 and new."slot_id"=1) then
        Raise notice 'You can only register or update registration online before 12 hours from the start
time of the slot.';
        return null;
    end if;
    if(new."date"-new."reg_date"=0 and new."slot_id"=2 and new."reg_time">>'2:00:00') then
        Raise notice 'You can only register or update registration online before 12 hours from the start
time of the slot.';
        return null;
    end if;
    return new;
else
return new;
end if;
end
```

**Output:**



pgAdmin 4

**Query Editor**

```

1 UPDATE t6_vc_mn.registration_details
2   SET reg_id=109, u_id=31, slot_id=2, reg_time='1:00:00'
3 WHERE reg_id=109;

```

**Messages**

UPDATE 1

Query returned successfully in 91 msec.

**Data Output**

No data output. Execute a query to get output.

pgAdmin 4

**Query Editor**

```

1 UPDATE t6_vc_mn.registration_details
2   SET reg_id=109, u_id=31, slot_id=2, reg_time='3:00:00'
3 WHERE reg_id=109;

```

**Messages**

NOTICE: You can only register or update registration online before 12 hours from the start time of the slot.  
UPDATE 0

Query returned successfully in 74 msec.

**Data Output**

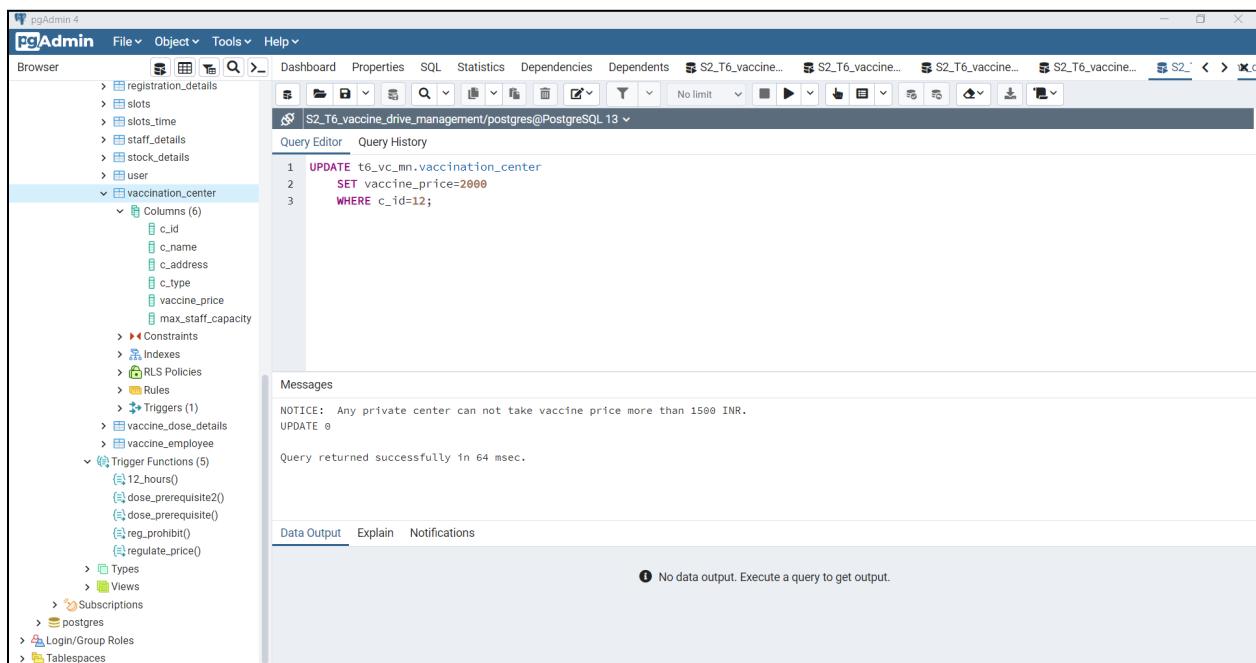
No data output. Execute a query to get output.

## 5) Regulate prices of vaccines at private vaccination centers.

**Code:**

```
begin  
  
if (new."c_type"='private' and new."vaccine_price">>1500) then  
Raise notice 'Any private center can not take vaccine price more than 1500 INR.';  
return null;  
elseif ((new."c_type"='government' or new."c_type"='gov-D2D') and new."vaccine_price">>0) then  
Raise notice 'Any government center can not take vaccine price.';  
return null;  
end if;  
  
return new;  
  
end
```

**Output:**



## Functions:

### 1. Find female to male ratio who have taken at least the first dose of vaccine in a given city.

```
CREATE OR REPLACE FUNCTION female_to_male_ratio(center character(10))
RETURNS float(4)
LANGUAGE 'plpgsql'
AS
$BODY$
declare
a float(4); c float(4);
BEGIN
select count(distinct u_id) into a
from vaccine_dose_details natural join registration_details natural join "user" natural join vaccination_center
where u_gender='F' and c_address=center;
select count(distinct u_id) into c
from vaccine_dose_details natural join registration_details natural join "user" natural join vaccination_center
where u_gender='M' and c_address=center;
return a/c;
END
$BODY$;

select * from female_to_male_ratio('ahmedabad')
```

### Output:

The screenshot shows the pgAdmin 4 interface with the following details:

- Left Panel (Browser):** Shows the database structure with Schemas (2), Tables (14), and other objects like Functions and Procedures.
- Center Panel (Query Editor):** Displays the SQL code for the function creation and a subsequent query execution.

```
19 CREATE OR REPLACE FUNCTION female_to_male_ratio(center character(10))
20 RETURNS float(4)
21 LANGUAGE 'plpgsql'
22 AS
23 $BODY$
24 declare
25 a float(4); c float(4);
26 BEGIN
27 select count(distinct u_id) into a
28 from vaccine_dose_details natural join registration_details natural join "user" natural join vaccination_center
29 where u_gender='F' and c_address=center;
30 select count(distinct u_id) into c
31 from vaccine_dose_details natural join registration_details natural join "user" natural join vaccination_center
32 where u_gender='M' and c_address=center;
33 return a/c;
34 END
35 $BODY$;
36
37 select * from female_to_male_ratio('ahmedabad')
```
- Bottom Panel (Data Output):** Shows the result of the query execution:

female_to_male_ratio
real
1 1.4285715
- Status Bar:** Shows a green checkmark indicating success and the message "Successfully run. Total query runtime: 347 msec. 1 rows affected."

## 2. Find dynamic online registrations available for given c\_id, slot\_id and date.

### Views:

```
create view v as
select c_id,date,count(employee_id)
from allocated_staff natural join staff_details
where staff_type='health-care'
group by c_id,date;

create view offline_data as
select c_id,slots."slot_id",slots."date",offline_approx_reg
from offline_approx_analysis inner join slots
on offline_approx_analysis."slot_id"=slots."slot_id"
where (slots."date"<'2021-08-01' and offline_approx_analysis."date"='2021-07-21')
or ((slots."date" between '2021-08-01' and '2021-08-31') and offline_approx_analysis."date"='2021-08-01')
or ((slots."date" between '2021-09-01' and '2021-09-30') and offline_approx_analysis."date"='2021-09-01')
or ((slots."date" between '2021-10-01' and '2021-10-31') and offline_approx_analysis."date"='2021-10-01');
```

### Function:

```
CREATE OR REPLACE function "dynamic_online_reg"(center bigint, s_id integer, s_date date)
```

```
RETURNS integer
```

```
LANGUAGE 'plpgsql'
```

```
AS $BODY$
```

```
declare
co integer;
```

```
declare
d integer;
BEGIN
```

```
if(s_date<'2021-09-10') then
select count into co
from v,slots
where v."date" = '2021-07-21' and slots."date"=s_date and slot_id=s_id and c_id=center;
else
select count into co
from v,slots
where v."date" = '2021-09-10' and slots."date"=s_date and slot_id=s_id and c_id=center;
end if;
select co*5-offline_approx_reg into d
from offline_data
where "date"=s_date and slot_id=s_id and c_id=center;
```

```
return d;
```

```
END;
```

```
$BODY$;
```

```
select "dynamic_online_reg"(1,1,'2021-07-21');
```

## Output:

The screenshot shows the pgAdmin 4 interface. On the left is the 'Browser' pane, which lists various database objects under the schema 'S2\_T6\_vaccine\_drive\_management'. In the center is the 'Query Editor' pane, containing the following SQL code:

```
22 where v."date" = '2021-07-21' and slots."date"=s_date and slot_id=s_id and c_id=center;
23 else
24 select count into co
25 from v,slots
26 where v."date" = '2021-09-10' and slots."date"=s_date and slot_id=s_id and c_id=center;
27 end if;
28 select co*5-offline_approx_reg into d
29 from offline_data
30 where "date"=s_date and slot_id=s_id and c_id=center;
31
32 return d;
33 END;
34 $BODY$;
35
36 select "dynamic_online_reg"(1,1,'2021-07-21');
37
```

Below the Query Editor is the 'Messages' pane, which displays the output:

Successfully run. Total query runtime: 107 msec.  
1 rows affected.

At the bottom right of the pgAdmin window, there is a green success message box:

✓ Successfully run. Total query runtime: 107 msec. 1 rows affected.

## SQL queries:

### 1. Determine staff utilization for both the staff allocations.

For first allocation:

```
select employee_id, count(reg_id)
from vaccine_employee natural join registration_details
where "date" < '2021-09-10'
group by employee_id
order by employee_id;
```

```
28 from vaccination_center
29 where c_address='ahmedabad';
30
31
32 select employee_id, count(reg_id)
33 from vaccine_employee natural join registration_details
34 where "date" < '2021-09-10'
35 group by employee_id
36 order by employee_id;
37
```

Messages

Successfully run. Total query runtime: 112 msec.  
57 rows affected.

employee_id	count
1	2
2	1
3	2
4	2
5	4
6	3
7	3
8	1
9	2
10	2

No. of tuples = 57.

For second allocation:

```
select employee_id, count(reg_id)
from vaccine_employee natural join registration_details
where "date" >= '2021-09-10'
group by employee_id
order by employee_id;
```

The screenshot shows the pgAdmin 4 interface. The left sidebar is titled 'Browser' and lists various database objects under 'Tables (14)'. The 'vaccine\_employee' table is selected, and its 'Columns (2)' are shown: 'reg\_Id' and 'employee\_Id'. The main area is the 'Query Editor' containing the following SQL code:

```

28 from vaccination_center
29 where c_address='ahmedabad';
30
31
32 select employee_id,count(reg_id)
33 from vaccine_employee natural join registration_details
34 where "date">>='2021-09-10'
35 group by employee_id
36 order by employee_id;
37

```

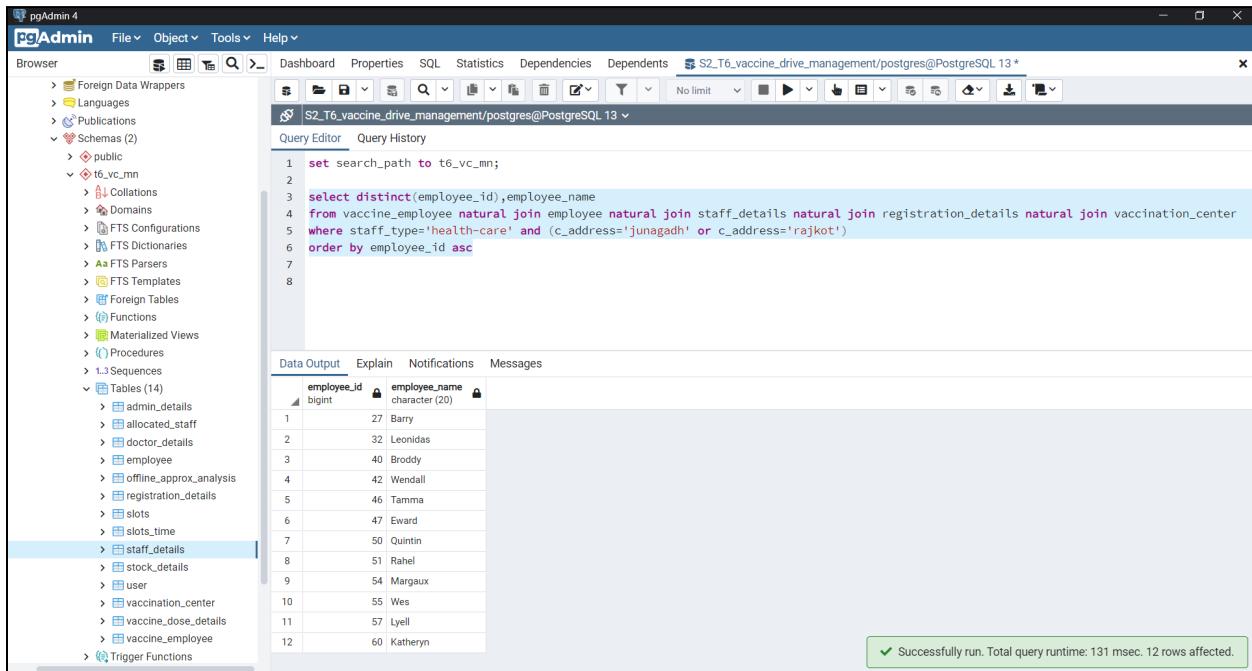
Below the code, the 'Messages' section indicates the query was successfully run with a runtime of 70 msec and 53 rows affected.

employee_id	count
1	1
2	2
3	3
4	1
5	2
6	2
7	2
8	4
9	2
10	2

No. of tuples = 53.

**2. List out ids and names of health-care employees who work in city junagadh or rajkot.**

```
select distinct(employee_id),employee_name  
from vaccine_employee natural join employee natural join staff_details natural join registration_details natural join  
vaccination_center  
where staff_type='health-care' and (c_address='junagadh' or c_address='rajkot')  
order by employee_id asc
```



The screenshot shows the pgAdmin 4 interface with a query editor window. The query is:

```
set search_path to t6_vc_mn;  
select distinct(employee_id),employee_name  
from vaccine_employee natural join employee natural join staff_details natural join registration_details natural join vaccination_center  
where staff_type='health-care' and (c_address='junagadh' or c_address='rajkot')  
order by employee_id asc
```

The results table has two columns: employee\_id (bigint) and employee\_name (character(20)). The data is:

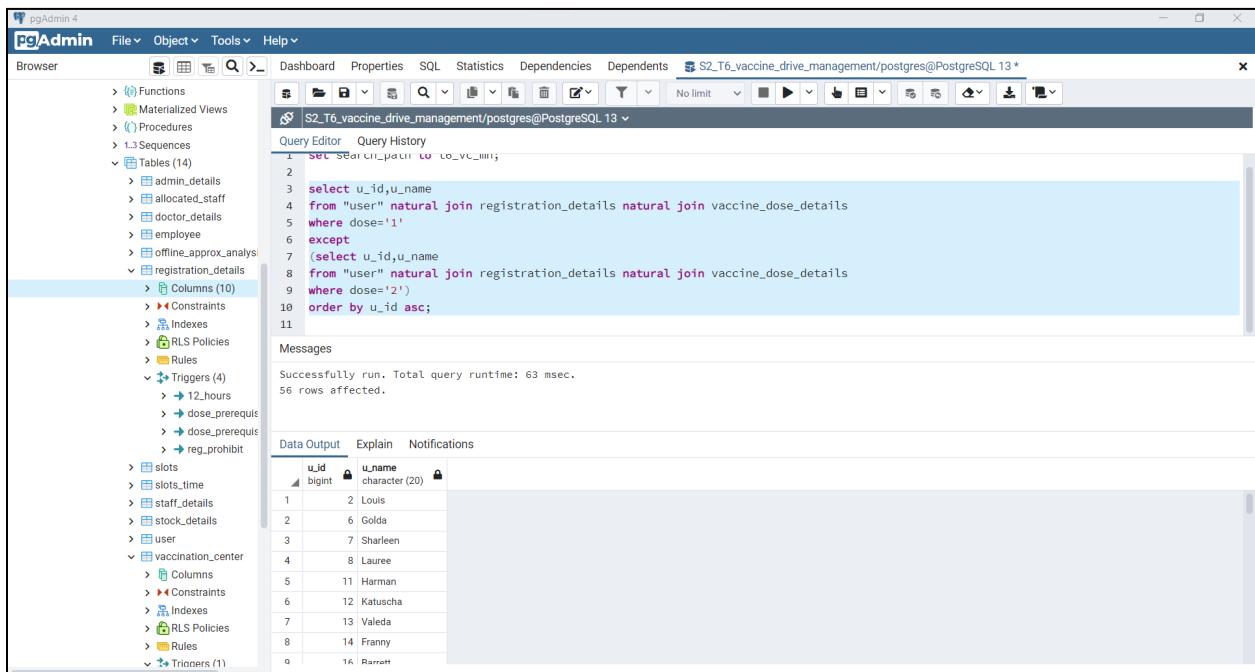
employee_id	employee_name
1	27 Barry
2	32 Leonidas
3	40 Broddy
4	42 Wendall
5	46 Tamina
6	47 Eward
7	50 Quintin
8	51 Rahel
9	54 Margeaux
10	55 Wes
11	57 Lyell
12	60 Katheryn

A message at the bottom right indicates: "Successfully run. Total query runtime: 131 msec. 12 rows affected."

No of tuples : 12

### 3. List out all the users who are partially vaccinated.

```
select u_id,u_name
from "user" natural join registration_details natural join vaccine_dose_details
where dose='1'
except
(select u_id,u_name
from "user" natural join registration_details natural join vaccine_dose_details
where dose='2')
order by u_id asc;
```



The screenshot shows the PgAdmin 4 interface with the following details:

- Browser:** Shows the database schema structure, including tables like `admin_details`, `allocated_staff`, `doctor_details`, `employee`, `offline_approx_analysis`, and `registration_details`.
- Query Editor:** Displays the SQL query entered by the user.
- Messages:** Shows the execution results, indicating 56 rows affected.
- Data Output:** Shows the results of the query in a tabular format.

u_id	u_name
1	Louis
2	Golda
3	Sharleen
4	Lauree
5	Harman
6	Katuscha
7	Valeda
8	Franny
9	Ronett

No. of tuples = 56.

#### 4. Find the no. of remaining vaccines at every center for both the stocks.

create view tmp2 as

```
select vaccination_center."c_id",stock_details."date",count(reg_id) as vaccines
from (vaccination_center natural join registration_details natural join vaccine_dose_details) inner join stock_details
on vaccination_center."c_id"=stock_details."c_id"
where "registration_details"."date" < '2021-09-10'
group by vaccination_center."c_id",stock_details."date"
having stock_details."date" = '2021-07-21'
order by c_id;
```

```
select c_id,proquad_vaccine + varivax_vaccine - wasted_vaccines - vaccines as remaining_vaccines
from stock_details natural join tmp2;
```

c_id	remaining_vaccines
1	8
2	8
3	6
4	6
5	7
6	6
7	7
8	8
9	7

No. of tuples = 19.

create view tmp3 as

```
select vaccination_center."c_id",stock_details."date",count(reg_id) as vaccines
from (vaccination_center natural join registration_details natural join vaccine_dose_details) inner join stock_details
on vaccination_center."c_id"=stock_details."c_id"
where "registration_details"."date" >= '2021-09-10'
group by vaccination_center."c_id",stock_details."date"
having stock_details."date" = '2021-09-10'
order by c_id;
```

```
select c_id,proquad_vaccine + varivax_vaccine - wasted_vaccines - vaccines as remaining_vaccines
from stock_details natural join tmp3;
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' panel displays a tree view of database objects under the schema 'S2\_T6\_vaccine\_dr'. The 'Tables (14)' section is expanded, showing tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', 'employee', etc. The 'Allocated\_Staff' table is currently selected. In the center, the 'Query Editor' window contains the following SQL query:

```

4 select vaccination_center."c_id",stock_details."date",count(reg_id) as vaccines
5 from (vaccination_center natural join registration_details natural join vaccine_dose_details) inner join stock_details
6 on vaccination_center."c_id"=stock_details."c_id"
7 where "registration_details"."date"='2021-09-10'
8 group by vaccination_center."c_id",stock_details."date"
9 having stock_details."date"='2021-09-10'
10 order by c_id;
11
12 select c_id,proquad_vaccine + varivax_vaccine - wasted_vaccines - vaccines as remaining_vaccines
13 from stock_details natural join tmp3;

```

Below the query, the 'Messages' section shows the output: 'Successfully run. Total query runtime: 69 msec.' and '18 rows affected.'

The 'Data Output' tab is selected in the bottom navigation bar, and the results are displayed in a table:

c_id	remaining_vaccines
1	7
2	11
3	8
4	8
5	7
6	5
7	8
8	5
9	5
10	5
..	..

No. of tuples = 18.

**5. Give details of the vaccination center that belongs to vadodara.**

```
select *  
from vaccination_center  
where c_address='vadodara'
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with various objects like Catalogs, Event Triggers, Extensions, Foreign Data Wrappers, Languages, Publications, Schemas (2), and Tables (14). The Tables section is currently selected. The main pane shows a query editor with the following SQL code:

```
1 set search_path to t6_vc_mn;  
2  
3 select *  
4 from vaccination_center  
5 where c_address='vadodara'
```

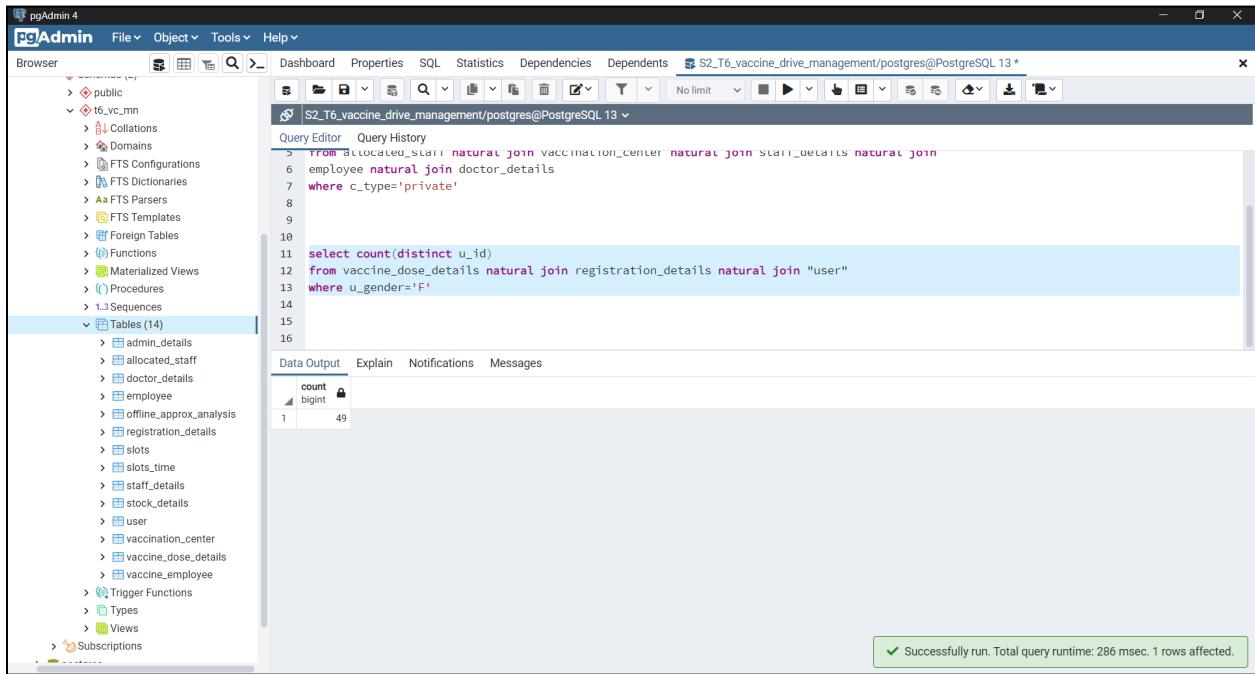
Below the query editor is a data output table with the following columns: c\_id, c\_name, c\_address, c\_type, vaccine\_price, and max\_staff\_capacity. The data shows four rows of vaccination centers, all located in Vadodara:

c_id	c_name	c_address	c_type	vaccine_price	max_staff_capacity
1	13 Mandvi UHC	vadodara	government	0.00	3
2	14 Hari Nagar UHC	vadodara	private	1500.00	3
3	15 Savli UHC	vadodara	government	0.00	3
4	16 Akota UHC	vadodara	government	0.00	3

No of tuples : 4

## 6. Count female users who have taken at least one dose of vaccine.

```
select count(distinct u_id)
from vaccine_dose_details natural join registration_details natural join "user"
where u_gender='F'
```



The screenshot shows the pgAdmin 4 interface. On the left, the 'Tables (14)' section is selected in the browser pane, listing tables like admin\_details, allocated\_staff, doctor\_details, employee, offline\_approx\_analysis, registration\_details, slots, slots\_time, staff\_details, stock\_details, user, vaccination\_center, vaccine\_dose\_details, and vaccine\_employee. The main pane displays the SQL query:

```
FROM allocated_staff natural join vaccination_center natural join staff_details natural join
employee natural join doctor_details
WHERE c_type='private'

select count(distinct u_id)
FROM vaccine_dose_details natural join registration_details natural join "user"
WHERE u_gender='F'
```

The 'Data Output' tab shows the result of the query:

count	bigint
1	49

A green success message at the bottom right of the results pane states: "Successfully run. Total query runtime: 286 msec. 1 rows affected."

No of tuples : 1

Number of female users : 49

**7. Give name of the vaccination center which has minimum staff capacity.**

```
select c_name
from vaccination_center
where max_staff_capacity in (select min(max_staff_capacity) from vaccination_center)
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with various objects like Catalogs, Event Triggers, Extensions, Foreign Data Wrappers, Languages, Publications, Schemas (2), and Tables (14). The Tables section is currently selected. The main pane contains a Query Editor window with the following SQL code:

```
1 set search_path to t6_vc_mn;
2
3 select c_name
4 from vaccination_center
5 where max_staff_capacity in (select min(max_staff_capacity) from vaccination_center)
6
7
8
9
```

Below the Query Editor, the Data Output tab is active, showing the results of the query:

c.name
Gimar UHC
Anjar UHC

No of tuples : 2

## 8. List user details whose guardian is Lennie.

```
select u_id,u_name,u_age,u_gender,u_address,u_phone_number  
from "user"  
where u_guardian='Lennie'
```

The screenshot shows the pgAdmin 4 interface. On the left is a tree view of database objects under 'S2\_T6\_vaccine\_drive\_management/postgres@PostgreSQL 13'. The 'Tables (14)' node is selected. In the center is the 'Query Editor' window containing the SQL query:

```
8  
9  
10  
11 select u_id,u_name,u_age,u_gender,u_address,u_phone_number  
12 from "user"  
13 where u_guardian='Lennie'  
14  
15  
16  
17  
18  
19
```

Below the query editor is the 'Data Output' tab, which displays the results of the query:

	u_id	u_name	u_age	u_gender	u_address	u_phone_number
1	[PK] bigint	character(20)	numeric(4,2)	'char'(1)	character varying(100)	character(15)
	10	Arther	2.00	M	rajkot	5785394151

A green success message at the bottom right of the data output area says: "Successfully run. Total query runtime: 111 msec. 1 rows affected."

No of tuples : 1

**9. List the user names who are male and at least 7 years old.**

```
select u_name  
from "user"  
where u_age>=7 and u_gender='M'
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays a tree view of database objects, including 'Tables (14)' which is expanded to show 'user' and its columns: u\_id, u\_name, u\_age, u\_gender, u\_address, u\_phone\_number, and u\_guardian. The 'Query Editor' tab is active, containing the following SQL code:

```
1 set search_path to t6_vc_mn;  
2  
3 select u_name  
4 from "user"  
5 where u_age>=7 and u_gender='M'  
6
```

The 'Data Output' tab is selected, showing the results of the query:

u.name
Allyn
Abrahan
Tadio
Laurence
Merrill
Brand
Nealson
Jasen
Knox
Dag
Francesco
Pavel
Skipp
Sherlocke
Lucian
Brendan
Dominique

No of tuples : 17

**10. Give user details such as name, gender and age who has taken at least 1st dose and belongs to city vadodara.**

```
select u_name,u_gender,u_age  
from vaccine_dose_details natural join registration_details natural join "user" natural join vaccination_center  
where c_address='vadodara' and dose='1'
```

The screenshot shows the pgAdmin 4 interface with the following details:

- Browser:** Shows the schema tree for the database "S2\_T6\_vaccine\_drive\_management".
- Query Editor:** Contains the SQL query:

```
1 set search_path to t6_vc_mn;  
2  
3 select u_name,u_gender,u_age  
4 from vaccine_dose_details natural join registration_details natural join "user" natural join vaccination_center  
5 where c_address='vadodara' and dose='1'
```
- Data Output:** Displays the results of the query in a tabular format. The columns are u\_name, u\_gender, and u\_age. The data consists of 17 rows:

	u_name	u_gender	u_age
1	Golda	F	9.50
2	Valeda	F	12.00
3	Diannne	F	6.25
4	Theodora	F	8.00
5	Giralda	F	7.00
6	Hillyer	M	2.00
7	Merrill	M	11.00
8	Cathryn	F	6.00
9	Calley	F	4.00
10	Ninnette	F	8.75
11	Novelia	F	15.00
12	Marquita	F	14.50
13	Ilene	F	14.00

- Message Bar:** Shows a green checkmark and the message "Successfully run. Total query runtime: 159 msec. 17 rows affected."

No of tuples : 17

**11. Print all the user ids which have registered more than one time.**

```
select u_id  
from vaccine_dose_details natural join registration_details  
group by u_id  
having count(reg_id)>1  
order by u_id
```

The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, displaying the database schema with two schemas: 'public' and 't6\_vc\_mn'. Under 't6\_vc\_mn', there are various objects like Collations, Domains, FTS Configurations, etc. The main area is the 'Query Editor' where the following SQL query is written:

```
1 set search_path to t6_vc_mn;  
2  
3 select u_id  
4 from vaccine_dose_details natural join registration_details  
5 group by u_id  
6 having count(reg_id)>1  
7 order by u_id  
8
```

Below the query editor is the 'Data Output' tab, which displays the results of the query. The results are a table with a single column 'u\_id' containing 24 rows of data:

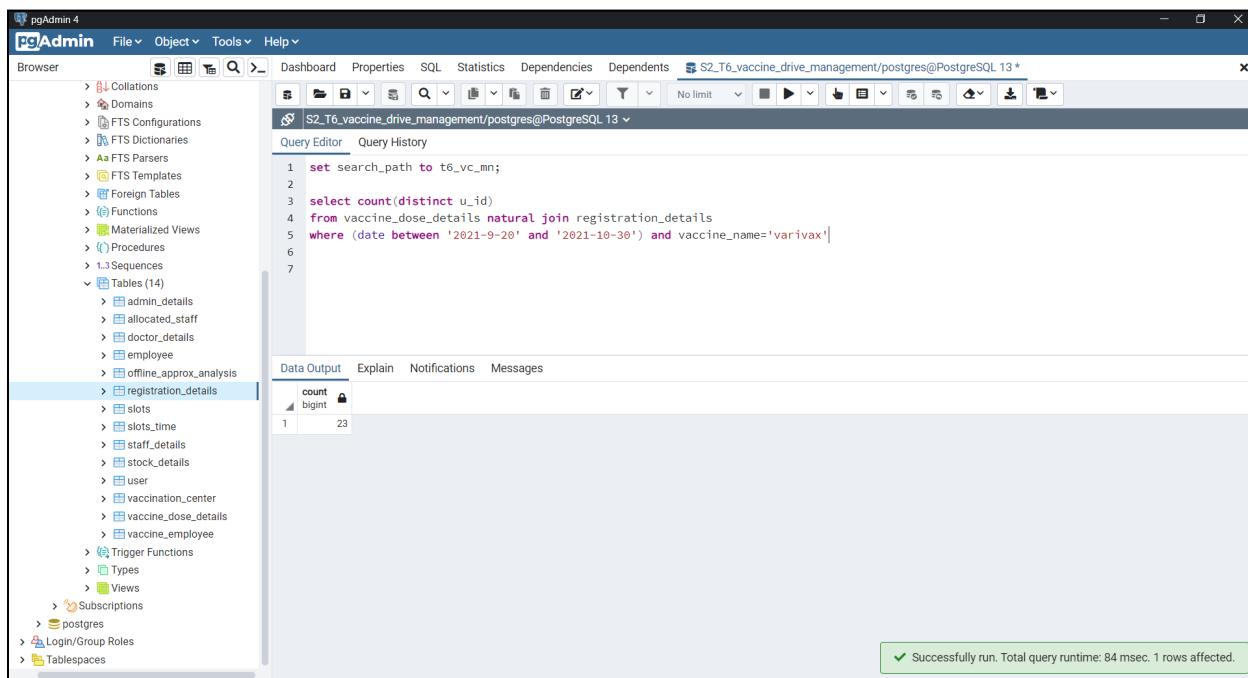
u_id
1
3
4
5
9
10
15
18
20
21
22
24
27

At the bottom right of the data output area, there is a green success message: "Successfully run. Total query runtime: 101 msec. 24 rows affected."

No of tuples : 24

**12. Find the number of users who have taken varivax vaccine's at least one dose between 20 sep to 30 oct.**

```
select count(distinct u_id)
from vaccine_dose_details natural join registration_details
where (date between '2021-9-20' and '2021-10-30') and vaccine_name='varivax'
```



```
set search_path to t6_vc_mn;
select count(distinct u_id)
from vaccine_dose_details natural join registration_details
where (date between '2021-9-20' and '2021-10-30') and vaccine_name='varivax'
```

count	bigint
1	23

Successfully run. Total query runtime: 84 msec. 1 rows affected.

No of tuples : 1

Number of such users : 23

**13. Find details of the non profit vaccination center along with the doctor associated with that center that are located in surat.**

```
select *
from vaccination_center natural join doctor_details
where c_address='surat' and c_type!='private'
```

The screenshot shows the PgAdmin 4 interface. The left sidebar displays the database schema, including tables like 'vaccination\_center', 'doctor\_details', and 'employee'. The main window has a 'Query Editor' tab where the following SQL query is written:

```
1 set search_path to t6_vc_mn;
2
3 select *
4 from vaccination_center natural join doctor_details
5 where c_address='surat' and c_type!='private'
6
7
```

The 'Data Output' tab shows the results of the query:

c_id	c_name	c_address	c_type	vaccine_price	max_staff_capacity	employee_id	doctor_email
1	17 Adajan UHC	surat	government	0.00	3	77	eoetuohyg@hotmail.com
2	18 Kamrej UHC	...	gov-O2D	0.00	4	78	rmurlli@yahoo.com
3	20 Kataragam UHC	surat	government	0.00	3	80	jfawthorpe@gmail.com

A green status bar at the bottom right indicates: "Successfully run. Total query runtime: 238 msec. 3 rows affected."

No of tuples : 3

#### 14. Print all the distinct cities where it is possible to take vaccine

```
select distinct c_address  
from vaccination_center
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with tables like admin\_details, allocated\_staff, doctor\_details, employee, offline\_approx\_analysis, registration\_details, slots, slots\_time, staff\_details, stock\_details, and user. The main area shows a query editor with the following SQL code:

```
1 select distinct c_address  
2 from vaccination_center
```

The results pane shows a table with one column, c\_address, containing six distinct values:

c_address
junagadh
vadodara
surat
rajkot
ahmedabad
kutch

A message at the bottom right indicates the query was successfully run.

No. of tuples : 6

## 15. Calculate the revenue for each private vaccination center

```
select c_name,c_address,count(reg_id)*vaccine_price as revenue  
from registration_details natural join vaccine_dose_details natural join vaccination_center  
group by c_id,c_name  
having c_type='private'  
order by revenue desc;
```

The screenshot shows the pgAdmin 4 interface with the following details:

- File Bar:** File, Object, Tools, Help.
- Toolbar:** Includes icons for New, Open, Save, Print, Copy, Paste, Find, Replace, Undo, Redo, and various database management tools.
- Connections:** S2\_T6\_vaccine\_drive\_management/postgres@PostgreSQL 13\*
- Query Editor:** Contains the SQL query provided above.
- Data Output:** A table showing the results of the query. The columns are c\_name, c\_address, and revenue. The data is as follows:

c_name	c_address	revenue
Ram Krishna Nagar UHC	rajkot	10500.00
Satellite UHC	ahmedabad	9000.00
Joshipura UHC	jungadh	9000.00
Hari Nagar UHC	vadodara	7500.00
Puna UHC	surat	4500.00

- Messages:** Shows a green success message: "Successfully run. Total query runtime: 46 msec. 5 rows affected."

No. of tuples : 5

## 16. List the vaccination centers where door-to-door service is available

```
select *  
from vaccination_center  
where c_type='gov-D2D'
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with tables like vaccination\_center, user, and others. The central area contains the Query Editor with the following SQL code:

```
1 select *  
2 from vaccination_center  
3 where c_type='gov-D2D'  
4  
5
```

The Data Output tab shows the results of the query:

c_id	c_name	c_address	c_type	vaccine_price	max_staff_capacity
1	2 Maninagar UHC	ahmedabad	gov-D2D	0.00	4
2	18 Kamrej UHC	surat	gov-D2D	0.00	4

A green success message at the bottom right indicates: "Successfully run. Total query runtime: 83 msec. 2 rows affected."

No. of tuples : 2

**17. Give the registration id of all users who have registered online for proquad vaccine.**

```
select reg_id  
from registration_details  
where reg_mode='online' and vaccine_name='proquad'
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays the database schema, including tables like 'registration\_details' and 'user'. The 'Query Editor' pane contains the SQL query:

```
1 select reg_id  
2 from registration_details  
3 where reg_mode='online' and vaccine_name='proquad'  
4  
5
```

The 'Data Output' pane shows the results of the query, which consists of a single column 'reg\_id' containing 59 rows of data. A message at the bottom right of the output pane indicates: "Successfully run. Total query runtime: 43 msec. 59 rows affected."

reg_id
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

No. of tuples : 59

**18. List out the centers and dates when some vaccines were wasted.**

```
select c_id,date  
from stock_details  
where wasted_vaccines>0
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects, including tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', 'employee', 'offline\_approx\_analys', 'registration\_details', 'slots', 'slots\_time', 'staff\_details', and 'stock\_details'. The 'stock\_details' table is currently selected. The main area contains a 'Query Editor' tab with the following SQL code:

```
1 select c_id,date  
2 from stock_details  
3 where wasted_vaccines>0  
4
```

Below the query editor is a 'Data Output' tab showing the results of the query:

c_id	date
1	2 2021-07-21
2	4 2021-07-21
3	7 2021-07-21
4	11 2021-09-10
5	18 2021-09-10

A green success message at the bottom right of the data output pane states: "Successfully run. Total query runtime: 43 msec. 5 rows affected."

No. of tuples : 5

**19. Find user\_id of all those patients who were attended to more than once by employee id 9.**

```
select u_id  
from vaccine_employee natural join registration_details  
where employee_id='9'  
group by u_id  
having count(u_id)>1
```

The screenshot shows the pgAdmin 4 interface. On the left is the 'Browser' pane, which contains a tree view of database objects. The 'vaccine\_employee' table is selected, showing its columns: u\_id, reg\_id, and employee\_id. The 'Query Editor' pane on the right contains the following SQL code:

```
1 select u_id  
2 from vaccine_employee natural join registration_details  
3 where employee_id='9'  
4 group by u_id  
5 having count(u_id)>1
```

The 'Data Output' pane below the query editor shows the results of the query:

u_id
1 35
2 30

A green success message at the bottom right of the Data Output pane states: "Successfully run. Total query runtime: 60 msec. 2 rows affected."

No. of Tuples : 2

**20. List the certificate ids of all users that took the vaccine on or after 25th August 2021.**

```
select e_id,date  
from vaccine_dose_details natural join registration_details natural join slots  
where date >= '2021-08-25'
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects, including tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', 'employee', 'offline\_approx\_analysis', 'registration\_details', and 'slots'. The 'slots' table is currently selected, showing its columns: slot\_id, date, and e\_id. The central area contains a 'Query Editor' tab with the following SQL code:

```
1 select e_id,date  
2 from vaccine_dose_details natural join registration_details natural join slots  
3 where date >= '2021-08-25'  
4
```

The 'Data Output' tab shows the results of the query, which are 70 rows of data. The columns are e\_id, slot\_id, and date. The data looks like this:

	e_id	slot_id	date
1	85568		2021-08-25
2	48954		2021-08-25
3	22570		2021-08-26
4	78843		2021-08-27
5	64280		2021-08-28
6	39628		2021-08-29
7	79216		2021-08-30
8	47401		2021-08-31
9	44392		2021-09-01
10	79449		2021-09-02
11	61557		2021-09-02
12	89870		2021-09-03
13	83404		2021-09-04
14	74943		2021-09-05
15	12939		2021-09-06

A green success message at the bottom right of the results pane states: "Successfully run. Total query runtime: 43 msec. 70 rows affected."

No. of Tuples : 70

**21. Give the top 3 cities that have the maximum total staff capacity.**

```
select c_address,sum(max_staff_capacity) as maxCap  
from vaccination_center  
group by c_address  
order by maxCap desc limit 3
```

The screenshot shows the PgAdmin 4 interface. On the left, the 'Browser' pane displays a tree view of database objects, including tables like 'vaccination\_center', 'admin\_details', 'allocated\_staff', etc. The 'Query Editor' pane contains the following SQL code:

```
1 select c_address,sum(max_staff_capacity) as maxCap  
2 from vaccination_center  
3 group by c_address  
4 order by maxCap desc limit 3
```

The 'Data Output' pane shows the results of the query:

c.address	maxcap
surat	13
ahmedabad	13
vadodara	12

A green message bar at the bottom right indicates: "Successfully run. Total query runtime: 49 msec. 3 rows affected."

No. of Tuples : 3

## 22. List the admin landline numbers of government vaccine centers in kutch or vadodara

```
select c_name,landline_number  
from vaccination_center natural join admin_details  
where c_type='government' and (c_address='kutch' or c_address='vadodara')
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays the database schema with tables like 'admin\_details', 'employee', 'allocated\_staff', 'doctor\_details', 'offline\_approx\_analysis', and 'registration\_details'. The 'registration\_details' table is currently selected, showing its columns: reg\_id, u\_id, slot\_id, date, vaccine\_name, reg\_mode, c\_id, dose, reg\_date, and reg\_time. In the center, the 'Query Editor' pane contains the SQL query:

```
1 select c_name,landline_number  
2 from vaccination_center natural join admin_details  
3 where c_type='government' and (c_address='kutch' or c_address='vadodara')  
4
```

The 'Data Output' pane shows the results of the query:

c.name	landline_number
Gandhidham UHC	7099026660
Anjar UHC	2764463195
Mandvi UHC	8032922385
Savil UHC	3275700163
Akota UHC	5555551499

At the bottom right of the Data Output pane, there is a green success message: "Successfully run. Total query runtime: 40 msec. 5 rows affected."

No. of Tuples : 5

**23. Print the vaccine ids of all users whose name starts with B.**

```
select v_id,u_name  
from "user" natural join registration_details natural join vaccine_dose_details  
where u_name like 'B%';
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' panel displays the database schema with tables like 'offline\_approx\_analysis', 'registration\_details', and 'user'. The 'user' table is expanded to show columns such as 'v\_id', 'u\_id', 'slot\_id', 'date', 'vaccine\_name', 'reg\_mode', 'c\_id', 'dose', 'reg\_date', and 'reg\_time'. The 'Query Editor' tab contains the SQL query:

```
1 select v_id,u_name  
2 from "user" natural join registration_details natural join vaccine_dose_details  
3 where u_name like 'B%';
```

The 'Data Output' tab shows the results of the query:

v_id	u_name
1	462 Beatrisa
2	764 Barrett
3	424 Beverie
4	873 Baudoin
5	528 Bibby
6	753 Beatrisa
7	609 Brand
8	944 Brittni
9	504 Bilby
10	944 Brendan
11	334 Baudoin
12	836 Beverie

A green success message at the bottom right of the Data Output pane states: "Successfully run. Total query runtime: 39 msec. 12 rows affected."

Number of tuples : 12

**24. List out all the details of private vaccination centers available in every city.**

```
select *\nfrom vaccination_center\nwhere c_type='private';
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects under the schema 'S2\_T6\_vaccine\_drive\_management'. The 'Tables' node is expanded, showing 14 tables, with 'vaccination\_center' selected. The central pane contains a 'Query Editor' window with the following SQL code:

```
1 set search_path to t6_vc_mn;\n2 select *\n3 from vaccination_center\n4 where c_type='private';
```

Below the query editor, the 'Messages' section shows the output: "Successfully run. Total query runtime: 82 msec. 5 rows affected." The 'Data Output' tab is selected, displaying a table with the results of the query:

c_id	c_name	c_address	c_type	vaccine_price	max_staff_capacity
1	3 Satellite UHC	ahmedabad	private	1500.00	3
2	5 Joshipura UHC	junagadh	private	1500.00	3
3	12 Ram Krishna Nagar UHC	rajkot	private	1500.00	3
4	14 Hari Nager UHC	vadodara	private	1500.00	3
5	19 Puna UHC	surat	private	1500.00	3

No. of tuples : 5

**25. Find the names of the youngest male users.**

```
select u_name
from "user"
where u_gender='M' and u_age=(select min(u_age)
                           from "user"
                           where u_gender='M');
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects, including schemas like 'S2\_T6\_vaccine...' and tables like 'user'. The main area is the Query Editor, which contains the following SQL code:

```
1 set search_path to t6_vc_mn;
2
3 select u_name
4   from "user"
5  where u_gender='M' and u_age=(select min(u_age)
6                                from "user"
7                                where u_gender='M');
```

Below the Query Editor, the Messages pane shows the output of the query: "Successfully run. Total query runtime: 82 msec. 1 rows affected." The Data Output pane shows the result of the query:

u_name
Evered

No. of tuples : 1

**26. Print the vaccination centers' names visited by 'Indira'.**

```
select c_name
from registration_details natural join "user" natural join vaccination_center
where u_name='Indira';
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects, including 'Tables (14)' which contains 'registration\_details'. The main area is the 'Query Editor' tab, showing the following SQL code:

```
1 set search_path to t6_vc_mn;
2 select c_name
3 from registration_details natural join "user" natural join vaccination_center
4 where u_name='Indira';
5
6
7
```

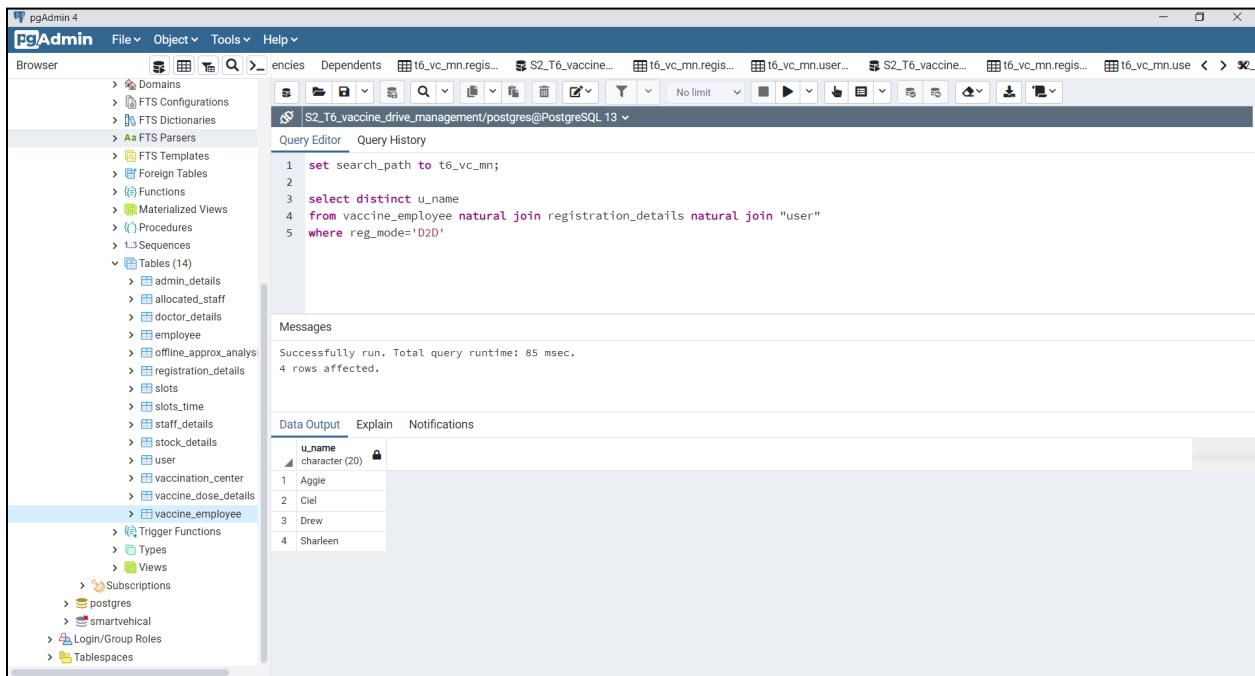
The 'Messages' panel below the editor shows the output of the query: "Successfully run. Total query runtime: 64 msec. 1 rows affected." The 'Data Output' panel shows the result of the query:

name
Gimar UHC

No. of tuples : 1

**27. Find the name of users who take vaccines by door to door service.**

```
select distinct u_name  
from vaccine_employee natural join registration_details natural join "user"  
where reg_mode='D2D'
```



The screenshot shows the pgAdmin 4 interface with the following details:

- Browser Panel:** Shows the database schema structure under the "S2\_T6\_vaccine\_drive\_management" database, including tables like "vaccine\_employee", "registration\_details", and "user".
- Query Editor:** Displays the SQL query:

```
1 set search_path to t6_vc_mn;  
2  
3 select distinct u_name  
4 from vaccine_employee natural join registration_details natural join "user"  
5 where reg_mode='D2D'
```
- Messages Panel:** Shows the execution results:

```
Successfully run. Total query runtime: 85 msec.  
4 rows affected.
```
- Data Output Panel:** Shows the resulting data in a table:

u_name
Aggie
Ciel
Drew
Sharleen

No. of tuples : 4

**28. Give the details of users whose name starts with 'A'.**

```
select *  
from "user"  
where u_name like 'A%'
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects, including 'Tables (14)' which is currently selected. The main area has three tabs: 'Query Editor', 'Query History', and 'Messages'. The 'Query Editor' tab contains the following SQL code:

```
1 set search_path to t6_vc_mn;  
2  
3 select *  
4 from "user"  
5 where u_name like 'A%'  
6
```

The 'Messages' tab shows the output of the query: "Successfully run. Total query runtime: 235 msec. 8 rows affected."

The 'Data Output' tab displays the results of the query as a table:

	u_id	u_name	u_age	u_gender	u_address	u_phone_number	u_guardian
1	9	Allyn	13.00	M	rajkot	2227476868	Noach
2	10	Arther	2.00	M	rajkot	5785394151	Lennie
3	23	Abrahan	10.00	M	ahmedabad	4911991138	Levin
4	31	Ava	4.00	F	surat	7322358773	Babs
5	35	Adriena	3.25	F	junagadh	8914618663	Damita
6	48	Aggie	3.00	F	ahmedabad	5188662631	Janelle
7	53	Adena	17.50	F	surat	6584523703	Avis
8	64	April	15.00	F	rajkot	6728447541	Glynda

No. of tuples : 8

## 29. Find out how many doctors are available in every city.

```
select c_address, count(employee_id) as no_of_doctors  
from doctor_details natural join vaccination_center  
group by c_address  
order by no_of_doctors desc;
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects, including tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', 'employee', 'offline\_approx\_analyis', 'registration\_details', 'slots', 'slots\_time', 'staff\_details', 'stock\_details', 'user', 'vaccination\_center', 'vaccine\_dose\_details', 'vaccine\_employee', and 'trigger\_functions'. The 'registration\_details' table is currently selected. The main area contains a 'Query Editor' tab with the following SQL code:

```
1 set search_path to t6_vc_mn;  
2  
3 select c_address, count(employee_id) as no_of_doctors  
4 from doctor_details natural join vaccination_center  
5 group by c_address  
6 order by no_of_doctors desc;  
7
```

Below the code, the 'Messages' section shows the output: "Successfully run. Total query runtime: 53 msec. 6 rows affected."

The 'Data Output' tab is selected, displaying the results of the query in a table:

c_address	no_of_doctors
vadodara	4
surat	4
ahmedabad	4
junagadh	3
rajkot	3
kutch	2

No. of tuples : 6

**30. Give all details of users who visited the vaccination center located at ahmedabad and take a second dose of vaccine.**

```
select *
from vaccine_dose_details natural join registration_details natural join "user" natural join vaccination_center
where c_address='ahmedabad' and dose='2'
```

c_id	w_id	reg_id	e_id	v_id	slot_id	date	vaccine_name	reg_mode	dose	reg_date	reg_time	u_name	u_sq
1	1	5	95	86718	209	2	2021-10-17	proquad	online	2	2021-10-16	16:05:00	Drew
2	2	48	97	46950	606	1	2021-10-18	proquad	online	2	2021-10-17	15:23:00	Aggie
3	1	20	98	57984	993	1	2021-10-19	proquad	online	2	2021-10-18	19:28:00	Vilma
4	4	27	99	43879	800	1	2021-10-20	proquad	online	2	2021-10-19	04:52:00	Rebeca
5	3	62	107	80258	876	2	2021-10-28	varivax	online	2	2021-10-27	09:49:00	Meggi

No. of tuples : 5

**31. Give the details of the doctor available at the vaccination center located at surat.**

```
select employee_id,doctor_email  
from doctor_details natural join vaccination_center  
where c_address='surat'
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects under the 'Tables (14)' section, with 'doctor\_details' selected. The main area contains a 'Query Editor' tab with the following SQL code:

```
1 set search_path to t6_vc_mn;  
2  
3 select employee_id,doctor_email  
4 from doctor_details natural join vaccination_center  
5 where c_address='surat'
```

The 'Messages' panel below the editor shows the output: "Successfully run. Total query runtime: 62 msec. 4 rows affected."

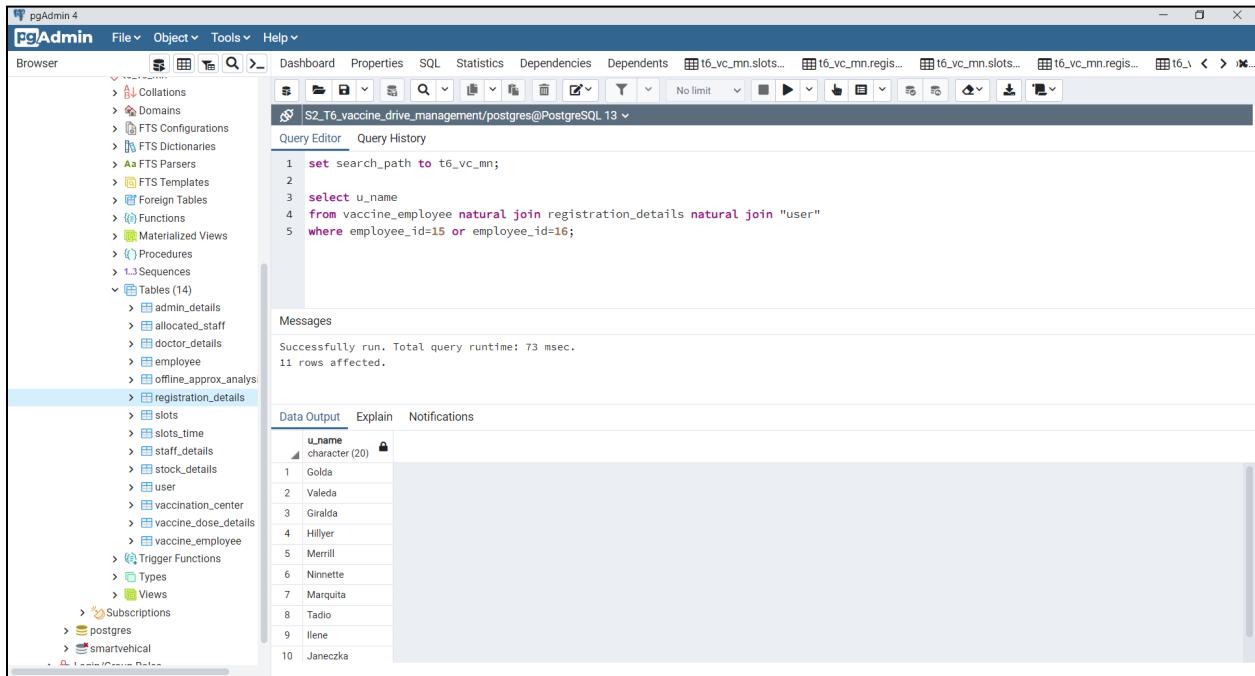
The 'Data Output' tab is active, displaying the results of the query:

employee_id	doctor_email
77	eotuhyg@hotmail.com
78	mrurllh@yahoo.com
79	epurcer@gmail.com
80	jfawthorpej@gmail.com

No. of tuples : 4

**32. Give the name of users who were attended by employee id 15 or 16.**

```
select u_name  
from vaccine_employee natural join registration_details natural join "user"  
where employee_id=15 or employee_id=16;
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects, including tables like 'admin\_details', 'allocated\_staff', 'doctor\_details', 'employee', 'offline\_approx\_analysis', 'registration\_details', 'slots', 'slots\_time', 'staff\_details', 'stock\_details', 'user', 'vaccination\_center', 'vaccine\_dose\_details', and 'vaccine\_employee'. The main area shows the SQL Query Editor with the following code:

```
1 set search_path to t6_vc_mn;  
2  
3 select u_name  
4 from vaccine_employee natural join registration_details natural join "user"  
5 where employee_id=15 or employee_id=16;
```

The Messages panel indicates the query was successfully run with a total runtime of 73 msec and 11 rows affected. The Data Output panel shows the results:

u.name
Golda
Valeda
Giralda
Hillyer
Merrill
Ninnette
Marquita
Tadio
Ilene
Janeczka

No. of tuples : 11

**33. Find top 3 vaccination centers which have maximum offline approximate registration in slot 1.**

```
select c_name,sumoffline_approx_reg as total  
from offline_approx_analysis natural join vaccination_center  
where slot_id=1  
group by c_id,c_name  
order by total desc limit 3
```

The screenshot shows the pgAdmin 4 interface with the following details:

- Browser:** Shows the database schema structure, including tables like 'offline\_approx\_analysis' which is highlighted.
- Query Editor:** Contains the SQL query:

```
1 set search_path to t6_vc_mn;  
2  
3 select c_name,sumoffline_approx_reg as total  
4 from offline_approx_analysis natural join vaccination_center  
5 where slot_id=1  
6 group by c_id,c_name  
7 order by total desc limit 3
```
- Messages:** Displays the execution message: "Successfully run. Total query runtime: 89 msec. 3 rows affected."
- Data Output:** Shows the results of the query in a table:

c_name	total
Kamnej UHC	14
Maninagar UHC	13
Hari Nagar UHC	12

No. of tuples : 3

### 34. Find out how many vaccination centers are there in every city.

```
select c_address, count(c_id) as no_of_centers  
from vaccination_center  
group by c_address  
order by no_of_centers desc;
```

The screenshot shows the PgAdmin 4 interface. On the left is the 'Browser' pane, which lists servers, databases, schemas, and tables. The 't6\_vc\_mn' schema is selected. In the center is the 'Query Editor' pane, containing the SQL query. Below it is the 'Messages' pane, which displays the execution results: 'Successfully run. Total query runtime: 68 msec.' and '6 rows affected.'. At the bottom is the 'Data Output' pane, which shows a table with the results:

c_address	no_of_centers
vaddoda	4
surat	4
ahmedabad	4
junagadh	3
rajkot	3
kutch	2

No. of tuples = 6.

### 35. Which slot is more preferable by the users.

```
select slot_id,count(reg_id)
from registration_details
group by slot_id;
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects under the schema 't6\_vc\_mn'. The 'Tables' node is expanded, showing 14 tables: admin\_details, allocated\_staff, doctor\_details, employee, offline\_approx\_analysis, registration\_details, slots, slots\_time, staff\_details, stock\_details, user, vaccination\_center, vaccine\_dose\_details, and vaccine\_employee. The 'Query Editor' tab is active, containing the following SQL code:

```
1 set search_path to t6_vc_mn;
2
3 select slot_id, count(reg_id)
4 from registration_details
5 group by slot_id;
```

The 'Messages' panel below the query editor shows the output of the query execution:

Successfully run. Total query runtime: 57 msec.  
2 rows affected.

The 'Data Output' panel displays the results of the query:

slot_id	count
1	51
2	60

According to past data, slot 2 is chosen more often by the users.

No. of tuples = 2.

**36. Give the name of the city which has provided maximum vaccination.**

```
select c_address,count(reg_id)
from registration_details natural join vaccine_dose_details natural join vaccination_center
group by c_address
order by count(reg_id) desc limit 1;
```

The screenshot shows the PgAdmin 4 interface. The left sidebar displays the database schema with tables like employee, offline\_approx\_analysis, registration\_details, slots, and stock\_details. The central area contains the Query Editor with the following SQL code:

```
1 set search_path to t6_vc_mn;
2
3
4 select c_address, count(reg_id)
5 from registration_details natural join vaccine_dose_details natural join vaccination_center
6 group by c_address
7 order by count(reg_id) desc limit 1;
```

The Messages panel shows the output: "Successfully run. Total query runtime: 73 msec. 1 rows affected." The Data Output panel shows the result of the query:

c_address	count
rajkot	24

No. of tuples = 1.

**37. List out vaccination centers and users attended by the center in decreasing order.**

```
select c_name,c_address,count(reg_id)
from registration_details natural join vaccine_dose_details natural join vaccination_center
group by c_id,c_name
order by count(reg_id) desc;
```

The screenshot shows the PgAdmin 4 interface with a query editor window. The query is:

```
1 set search_path to t6_vc_mn;
2
3 select c_name,c_address,count(reg_id)
4 from registration_details natural join vaccine_dose_details natural join vaccination_center
5 group by c_id,c_name
6 order by count(reg_id) desc ;
7
```

The results table has three columns: c\_name, c\_address, and count. The data is:

c_name	c_address	count
Nana Mava UHC	rajkot	9
GIDC UHC	rajkot	8
Adajan UHC	surat	8
Kataragam UHC	surat	7
Ram Krishna Nagar UHC	rajkot	7
Mandvi UHC	vadodara	6
Joshihpura UHC	junagadh	6
Satellite UHC	ahmedabad	6
Maninagar UHC	ahmedabad	6

No. of tuples = 20.

**38. Give the details of staff who have worked at more than one vaccination centers.**

```
select employee_id,employee_name,staff_type,count(distinct c_id)
from allocated_staff natural join staff_details natural join employee
group by employee_id,employee_name,staff_type
having count(distinct c_id)>1;
```

The screenshot shows the PgAdmin 4 interface. The left sidebar displays the database schema with various objects like tables, functions, and triggers. The main area has a 'Query Editor' tab where the SQL query is typed. Below it is a 'Messages' panel showing the execution results. The 'Data Output' panel shows the resulting table.

```
set search_path to t6_vc_mn;
select employee_id,employee_name,staff_type,count(distinct c_id)
from allocated_staff natural join staff_details natural join employee
group by employee_id,employee_name,staff_type
having count(distinct c_id)>1;
```

Messages

Successfully run. Total query runtime: 97 msec.  
4 rows affected.

employee_id	employee_name	staff_type	count
1	7 Florry	management	2
2	9 Melicent	management	2
3	40 Broddy	health-care	2
4	50 Quintin	health-care	2

No. of tuples = 4.

**39. Give the details of employees who have attended user\_id 6.**

```
select employee_id,employee_name,employee_number,staff_type  
from registration_details natural join vaccine_employee natural join employee natural join staff_details  
where u_id=6;
```

The screenshot shows the pgAdmin 4 interface. On the left is a tree view of database objects under 'S2\_T6\_vaccine\_drive\_management/postgres@PostgreSQL 13'. The 'employee' table is selected, and its columns (employee\_id, employee\_name, employee\_type, employee\_number) are visible. The main area contains the SQL query:

```
35 select employee_id,employee_name,employee_number,staff_type  
36 from registration_details natural join vaccine_employee natural join employee natural join staff_details  
37 where u_id=6;  
38  
39  
40  
41  
42  
43  
44
```

The 'Messages' section below the query editor shows: "Successfully run. Total query runtime: 81 msec. 2 rows affected."

The 'Data Output' tab displays the results of the query:

employee_id	employee_name	employee_number	staff_type
1	15 Monica	4125015529	management
2	38 Cherrita	2731826553	health-care

A green success message at the bottom right of the data output area says: "Successfully run. Total query runtime: 81 msec. 2 rows affected."

No. of tuples = 2.

**40. Give details of the users who have not registered for the vaccine in their own city.**

```
select u_id,u_name  
from "user" natural join registration_details natural join vaccination_center  
where u_address!=c_address;
```

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database browser with several schemas and objects listed. The main area contains a query editor window with the following SQL code:

```
1 set search_path to t6_vc_mn;  
2  
3 select u_id,u_name  
4 from "user" natural join registration_details natural join vaccination_center  
5 where u_address!=c_address;
```

Below the query editor, the 'Messages' section shows the output: "Successfully run. Total query runtime: 80 msec. 3 rows affected." The 'Data Output' section displays the results in a table:

u_id	u_name
36	Hillyer
39	Cathryn
24	Tadio

No. of tuples = 3.

## Section7:

Project Code with output screenshots

## Front-end Code

### Connect Nodejs interface with PostgreSQL Database:

**Code:**

```
const { Client } = require('pg');
const scanf = require('scanf');

const client = new Client({
  host: "localhost",
  user: "postgres",
  password: "admin",
  database: "S2_T6_vaccine_drive_management"
})
client.connect();

console.log("\n1: New Registration");
console.log("2: View Registration details");
console.log("3: Update Registration Details");
console.log("4: Delete Registration");

console.log("\nSelect your choice to proceed: ")
var c = scanf("%d");
console.log("Your query %d is being processed...", c);

switch (c) {
  case 1:
  {
    client.query('set search_path to t6_vc_mn')
    client.query("INSERT INTO t6_vc_mn.user(u_id, u_name, u_age, u_gender, u_address, u_phone_number, u_guardian) VALUES('87','Kavya','20','F','ahmedabad','9585849844','Kavya')", (err, result) =>
    {
      if(!err)
      {
        console.log("\nNew row is successfully Inserted in User's table\n")
      }
      else
        console.log("Error in Inserting\n")
    })
  }
}
```

```

client.query('set search_path to t6_vc_mn')
client.query("INSERT INTO t6_vc_mn.registration_details( reg_id, u_id, slot_id, date, vaccine_name,
reg_mode, c_id, dose, reg_date, reg_time)
VALUES('116','87','1','2021-07-29','varivax','online','1','1','2021-07-28','20:15:00')", (err, result) =>
{
  if(!err)
  {
    console.log("New row is successfully Inserted in Registration details table\n")
  }
  else
    console.log("Error in Inserting\n")
})
break;
}

case 2:
{
  console.log("Enter Registration ID: ");
  var regis=scanf("%d");

  client.query('set search_path to t6_vc_mn');
  client.query("select * from registration_details where reg_id=$1", [regis], (err, result) =>
{
  if(!err)
  {
    console.log("\nGiven below are your registration details\n")
    console.log(result.rows);
  }
  else
    console.log("Error in displaying\n")
})
break;
}

case 3:
{
  client.query('set search_path to t6_vc_mn');
  client.query("UPDATE t6_vc_mn.registration_details SET slot_id='2', c_id='3' WHERE
reg_id='116';", (err, result) =>
{
}

```

```

        if(!err)
        {
            console.log("Data updated successfully\n")
        }
        else
            console.log("Error in updating\n")
            client.end();
    })
    break;
}

case 4:
{
    client.query('set search_path to t6_vc_mn');
    client.query("Delete from registration_details where reg_id='116'",(err,result) =>
    {
        if(!err)
        {
            console.log("The registration is successfully deleted\n")
        }
        else
            console.log("Error in Deleting\n")
            client.end();
    })
    break;
}

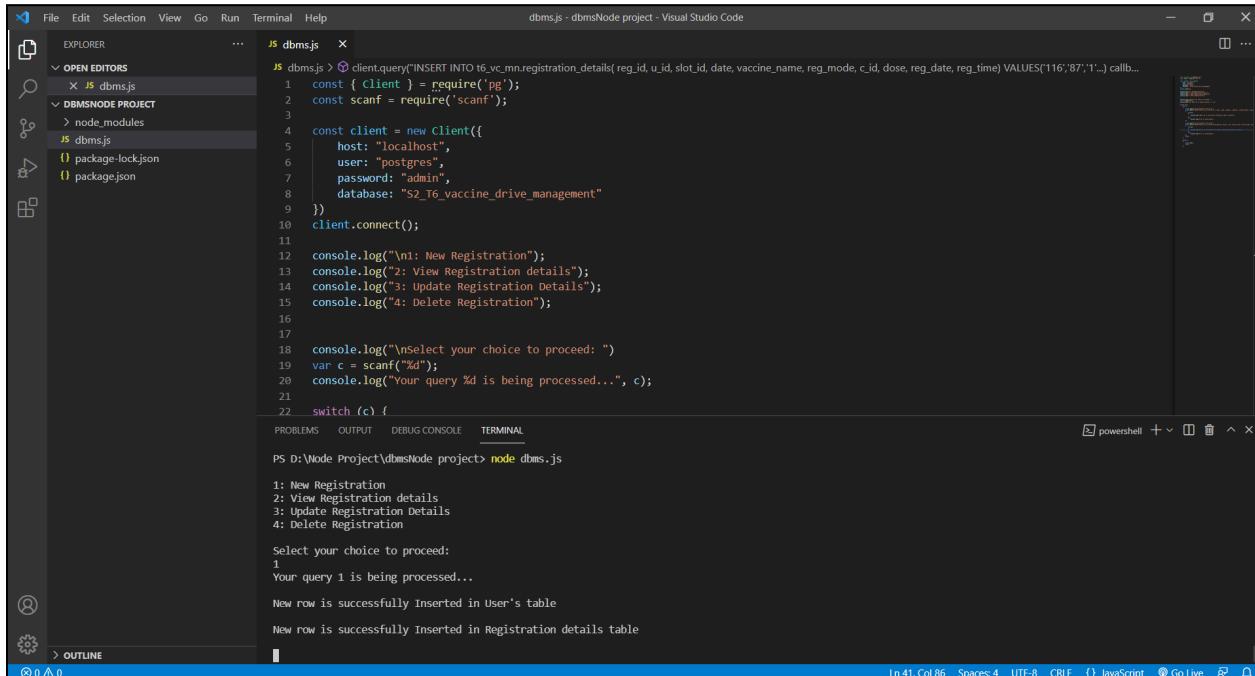
default:
{
    client.query('set search_path to t6_vc_mn')
    console.log("Enter the city: ");
    var city=scanf("%s");
    client.query("select c_id,c_name,c_type,vaccine_price from vaccination_center where
c_address=$1",[city],(err,result) =>
    {
        if(!err)
        {
            console.log("\nThe centres in the given city are: \n")
            console.log(result.rows);
        }
        else

```

```
    console.log("Error in displaying\n")
  })
  break;
}
client.end();
}
```

# Queries and Output Screenshots:

## 1. Insertion in user and registration\_details tables:



```

File Edit Selection View Go Run Terminal Help
dbmsjs - dbmsNode project - Visual Studio Code

EXPLORER
OPEN EDITORS
JS dbmsjs ×
DBMSNODE PROJECT
node_modules
JS dbmsjs
package-lock.json
package.json

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\Node Project\dbmsNode project> node dbms.js

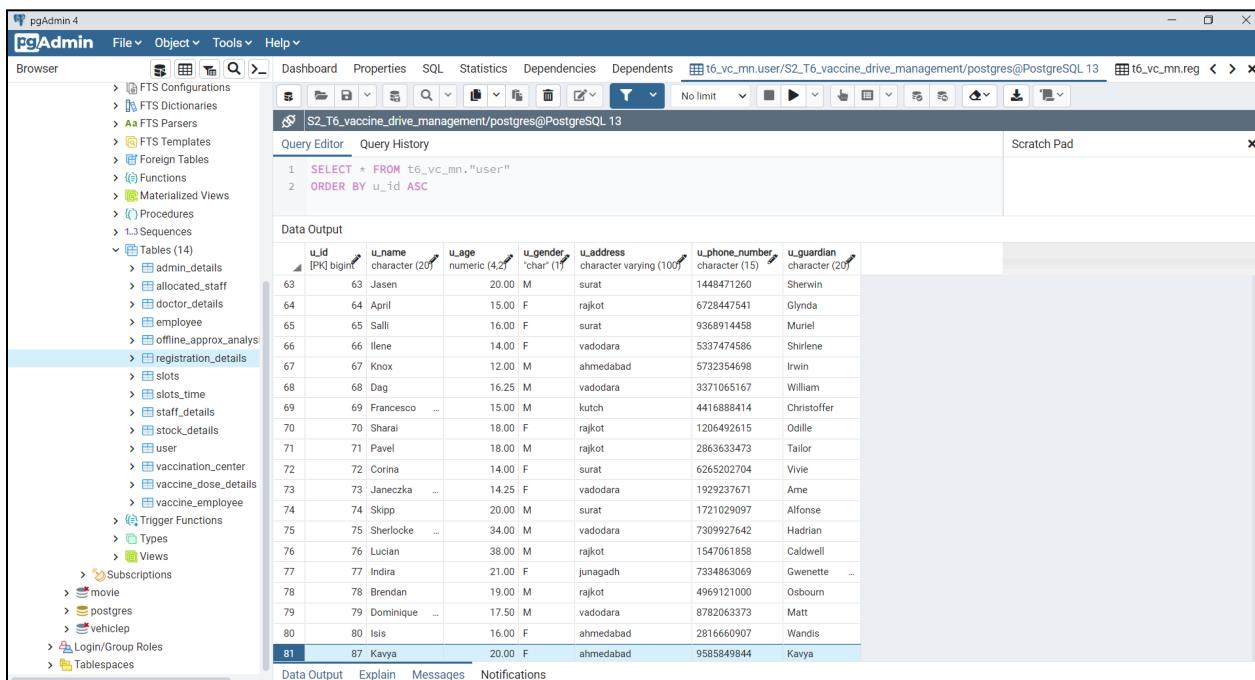
1: New Registration
2: View Registration details
3: Update Registration Details
4: Delete Registration

Select your choice to proceed:
1
Your query 1 is being processed...

New row is successfully Inserted in User's table
New row is successfully Inserted in Registration details table

powerShell + ×
In 41 Col 66 Spaces: 4 UTF-8 CRLF { JavaScript Go Live

```



	u_id	u_name	u_age	u_gender	u_address	u_phone_number	u_guardian
63	63	Jasen	20.00	M	surat	1448471260	Sherwin
64	64	April	15.00	F	rajkot	6728447541	Glynda
65	65	Salli	16.00	F	surat	9368914458	Muriel
66	66	Irene	14.00	F	vadodara	5337474586	Shirlene
67	67	Knox	12.00	M	ahmedabad	5723254698	Irwin
68	68	Dag	16.25	M	vadodara	3371065167	William
69	69	Francesco	...	M	kutch	4416888414	Christoffer
70	70	Sharai	18.00	F	rajkot	1206492615	Odille
71	71	Pavel	18.00	M	rajkot	2863633473	Talor
72	72	Corina	14.00	F	surat	6265202704	Vivie
73	73	Janecka	...	F	vadodara	1929237671	Arne
74	74	Skipp	20.00	M	surat	1721029097	Alfonse
75	75	Sherlocke	...	M	vadodara	7309927642	Hadrian
76	76	Lucian	38.00	M	rajkot	1547061858	Caldwell
77	77	Indira	21.00	F	junganadh	7334863069	Gwenette
78	78	Brendan	19.00	M	rajkot	4969121000	Osbourne
79	79	Dominique	...	M	vadodara	8782063373	Matt
80	80	Isis	16.00	F	ahmedabad	2816660907	Wandis
81	87	Kavya	20.00	F	ahmedabad	9585849844	Kavya

Screenshot of pgAdmin 4 showing the registration\_details table in the S2\_T6\_vaccine\_drive\_management database.

**Query Editor:**

```
1 SELECT * FROM t6_vc_mn.registration_details
2 ORDER BY reg_id ASC
```

**Data Output:**

reg_id	u_id	slot_id	date	vaccine_name	reg_mode	c_id	dose	reg_date	reg_time
94	94	18	2 2021-10-16	varivax	online	18	2	2021-10-15	05:55:00
95	95	5	2 2021-10-17	proquad	online	1	2	2021-10-16	16:05:00
96	96	4	2 2021-10-18	proquad	online	10	2	2021-10-17	04:43:00
97	97	48	1 2021-10-18	proquad	online	2	2	2021-10-17	15:23:00
98	98	20	1 2021-10-19	proquad	online	1	2	2021-10-18	19:28:00
99	99	27	1 2021-10-20	proquad	online	4	2	2021-10-19	04:52:00
100	100	30	2 2021-10-21	proquad	online	5	2	2021-10-20	15:03:00
101	101	52	1 2021-10-22	proquad	online	20	2	2021-10-21	15:05:00
102	102	55	1 2021-10-23	varivax	offline	5	2	2021-10-22	02:12:00
103	103	45	1 2021-10-24	proquad	online	12	2	2021-10-23	12:19:00
104	104	9	1 2021-10-25	varivax	online	11	2	2021-10-24	17:41:00
105	105	77	1 2021-10-26	varivax	online	6	1	2021-10-25	20:24:00
106	106	61	2 2021-10-27	varivax	online	17	2	2021-10-26	08:30:00
107	107	62	2 2021-10-28	varivax	online	3	2	2021-10-27	09:49:00
108	108	65	1 2021-10-29	varivax	online	20	2	2021-10-28	04:39:00
109	109	31	1 2021-10-30	proquad	online	17	2	2021-10-29	10:14:00
110	110	35	2 2021-10-31	proquad	offline	5	2	2021-10-30	17:22:00
111	116	87	1 2021-07-29	varivax	online	1	1	2021-07-28	20:15:00

## 2. View registration details of given reg\_id :

Screenshot of Visual Studio Code showing a Node.js application (dbmsjs) interacting with a PostgreSQL database.

**Code (dbmsjs.js):**

```
case 2:
{
    console.log("Enter Registration ID: ");
    var regis=scnf("%d");
    client.query('set search_path to t6_vc_mn');
    client.query("select * from registration_details where reg_id=$1",[regis],(err,result) =>
    {
        if(err)
        {
            Given below are your registration details
            [
                {
                    reg_id: '116',
                    u_id: '87',
                    slot_id: 1,
                    date: 2021-07-28T18:30:00.000Z,
                    vaccine_name: 'varivax',
                    reg_mode: 'online',
                    c_id: 1,
                    dose: '1',
                    reg_date: 2021-07-27T18:30:00.000Z,
                    reg_time: '20:15:00'
                }
            ]
        }
    })
}
```

**Terminal Output:**

```
PS D:\Node Project\dbmsNode project> node dbmsjs.js
1: New Registration
2: View Registration details
3: Update Registration Details
4: Delete Registration

Select your choice to proceed:
2
Your query 2 is being processed...
Enter Registration ID:
116

Given below are your registration details
[
  {
    reg_id: '116',
    u_id: '87',
    slot_id: 1,
    date: 2021-07-28T18:30:00.000Z,
    vaccine_name: 'varivax',
    reg_mode: 'online',
    c_id: 1,
    dose: '1',
    reg_date: 2021-07-27T18:30:00.000Z,
    reg_time: '20:15:00'
  }
]
```

### 3. Updation in registration\_details table:

```

JS dbms.js
  ...
  case 3:
    {
      client.query('set search_path to t6_vc_mn');
      client.query("UPDATE t6_vc_mn.registration_details SET slot_id='2', c_id='3' WHERE reg_id='116';", (err,result) =>
      {
        if(!err)
        {
          console.log("Data updated successfully\n")
        }
        else
          console.log("Error in updating\n")
        client.end();
      })
      break;
    }
  }

  case 4:
    {
      client.query('set search_path to t6_vc_mn');
      client.query("DELETE FROM t6_vc_mn.registration_details WHERE reg_id='116';", (err,result) =>
      {
        if(!err)
        {
          console.log("Data deleted successfully\n")
        }
        else
          console.log("Error in deleting\n")
        client.end();
      })
      break;
    }
  }

  default:
    {
      console.log("Invalid choice\n")
    }
  }
}

client.end();
}
  
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS D:\Node Project\dbmsNode project> node dbms.js

1: New Registration  
2: View Registration details  
3: Update Registration Details  
4: Delete Registration

Select your choice to proceed:  
PS D:\Node Project\dbmsNode project> node dbms.js

1: New Registration  
2: View Registration details  
3: Update Registration Details  
4: Delete Registration

Select your choice to proceed:  
3  
Your query 3 is being processed...  
Data updated successfully

PS D:\Node Project\dbmsNode project>

Ln 106, Col 15 Spaces: 4 UTF-8 CRLF {} JavaScript Go Live

reg_id	u_id	slot_id	date	vaccine_name	reg_mode	c_id	dose	reg_date	reg_time
94	94	18	2	2021-10-16	varivax	online	18 2	2021-10-15	05:55:00
95	95	5	2	2021-10-17	proquad	online	1 2	2021-10-16	16:05:00
96	96	4	2	2021-10-18	proquad	online	10 2	2021-10-17	04:43:00
97	97	48	1	2021-10-18	proquad	online	2 2	2021-10-17	15:23:00
98	98	20	1	2021-10-19	proquad	online	1 2	2021-10-18	19:28:00
99	99	27	1	2021-10-20	proquad	online	4 2	2021-10-19	04:52:00
100	100	30	2	2021-10-21	proquad	online	5 2	2021-10-20	15:03:00
101	101	52	1	2021-10-22	proquad	online	20 2	2021-10-21	15:05:00
102	102	55	1	2021-10-23	varivax	offline	5 2	2021-10-22	02:12:00
103	103	45	1	2021-10-24	proquad	online	12 2	2021-10-23	12:19:00
104	104	9	1	2021-10-25	varivax	online	11 2	2021-10-24	17:41:00
105	105	77	1	2021-10-26	varivax	online	6 1	2021-10-25	20:24:00
106	106	61	2	2021-10-27	varivax	online	17 2	2021-10-26	08:30:00
107	107	62	2	2021-10-28	varivax	online	3 2	2021-10-27	09:49:00
108	108	65	1	2021-10-29	varivax	online	20 2	2021-10-28	04:39:00
109	109	31	1	2021-10-30	proquad	online	17 2	2021-10-29	10:14:00
110	110	35	2	2021-10-31	proquad	offline	5 2	2021-10-30	17:22:00
111	116	87	2	2021-07-29	varivax	online	3 1	2021-07-28	20:15:00

#### 4. Deletion in registration\_details table:

```

File Edit Selection View Go Run Terminal Help
dbms.js x
dbms.js > client.query("Delete from registration_details where reg_id='116'" ) callback
case 4:
{
  client.query('set search_path to t6_vc_mn');
  client.query("Delete from registration_details where reg_id='116'",(err,result) =>
  {
    if(!err)
    {
      console.log("the registration is successfully deleted\n")
    }
    else
      console.log("Error in Deleting\n")
      client.end();
    })
  break;
}
default:
{
  client.end();
  break;
}
}

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\Node Project\dbmsNode project> node dbms.js
1: New Registration
2: View Registration details
3: Update Registration Details
4: Delete Registration
Select your choice to proceed:
4
Your query 4 is being processed...
The registration is successfully deleted
PS D:\Node Project\dbmsNode project>

```

Ln 92, Col 42 Spaces: 4 UTF-8 CR/LF { } JavaScript Go Live

reg_id	u_id	slot_id	date	vaccine_name	reg_mode	c_id	dose	reg_date	reg_time	
95	95	5	2021-10-17	proquad	online	1	2	2021-10-16	16:05:00	
96	96	4	2021-10-18	proquad	online	10	2	2021-10-17	04:43:00	
97	97	48	1	2021-10-18	proquad	online	2	2	2021-10-17	15:23:00
98	98	20	1	2021-10-19	proquad	online	1	2	2021-10-18	19:28:00
99	99	27	1	2021-10-20	proquad	online	4	2	2021-10-19	04:52:00
100	100	30	2	2021-10-21	proquad	online	5	2	2021-10-20	15:03:00
101	101	52	1	2021-10-22	proquad	online	20	2	2021-10-21	15:05:00
102	102	55	1	2021-10-23	varivax	offline	5	2	2021-10-22	02:12:00
103	103	45	1	2021-10-24	proquad	online	12	2	2021-10-23	12:19:00
104	104	9	1	2021-10-25	varivax	online	11	2	2021-10-24	17:41:00
105	105	77	1	2021-10-26	varivax	online	6	1	2021-10-25	20:24:00
106	106	61	2	2021-10-27	varivax	online	17	2	2021-10-26	08:30:00
107	107	62	2	2021-10-28	varivax	online	3	2	2021-10-27	09:49:00
108	108	65	1	2021-10-29	varivax	online	20	2	2021-10-28	04:39:00
109	109	31	1	2021-10-30	proquad	online	17	2	2021-10-29	10:14:00
110	110	35	2	2021-10-31	proquad	offline	5	2	2021-10-30	17:22:00

## 5. Query to view centers in a given city:

The screenshot shows a Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help
- Terminal Tab:** dbms.js
- Terminal Content:**

```
dbms.js - dbmsNode project - Visual Studio Code

SELECT your choice to proceed:
10
Your query 10 is being processed...
Enter the city:
ahmedabad

The centres in the given city are:
[{"c_id": "1", "c_name": "Ashram road UHC", "c_type": "government", "vaccine_price": "0.00"}, {"c_id": "3", "c_name": "Satellite UHC", "c_type": "private", "vaccine_price": "1500.00"}, {"c_id": "4", "c_name": "Godhpur UHC", "c_type": "government", "vaccine_price": "0.00"}, {"c_id": "2", "c_name": "Maninagar UHC", "c_type": "gov-D2D", "vaccine_price": "0.00"}]
```
- Status Bar:** In 121, Col 5, Spaces: 4, UTF-8, CRLF, {} JavaScript, Go Live, etc.