Dhirubhai Ambani
Institute of Information and Communication Technology

S6_TEAM-4

HOCKEY TRAINING MANAGEMENT SYSTEM

GROUP MEMBERS NAME &ID-

LAKSHYA SINGH - 201901248
MARTIN KUMAR GAMIT- 201901141
KEYUR CHAMPAWAT- 201901458
SHUBH JHANWAR- 201901204

MENTOR TA NAME-

KAVAN SIR - 202111007

INDEX

Section 1: Final version of SRS

Index of SRS

Section1: Final version of SRS

(A)Introduction:

1.1 Purpose

The Hockey training Management System objective is to provide a system which manages the activity of hockey at a time. The system users will consume less amount of time when compared to manual paper work through the automated system and also take care of all the training activity in an easier manner. This document aims to cover an in-depth overview of the functioning and features of the database. Trainers should be able to manage timetables, participants, and pieces of training. It will be able to check any report at any time. This system will provide the serving activity in a quick and easy manner. It will consume less amount of time as it is based on an automatic system. The purpose of our database would be to identify, clarify and organize the relationship between various entities of the "Sports Training Management System"like Super Admin, System User, Trainer, and Players who perform different activities according to their roles.

1.2 Intended Audience and Reading Suggestions

Hockey is a national sport of India. Unfortunately, the popularity of hockey is less as compared to other sports like Cricket and football. This application is mainly intended for the interested players and the companies. The intended companies can play a huge role in increasing the popularity of hockey by making games, apps. This data is made for the companies so they can use it to make games or apps.

1.3 Product Scope

India national sport is Hockey. India achieved a huge success in hockey as we go through India history in hockey. Recently the Indian hockey team, both men and women, represented our country in the Olympics. The team shows excellent performance. But the popularity of this sport is less as compared to other sports. Many people want to take it as a profession but they don't get sufficient facilities and opportunities. So this application targets to increase the popularity of hockey and to help people to make hockey as a profession.

1.4 Description

This project is for the online implementation of the sports management system. The database will have data of players, team coach, assistants,and trainers. This application is designed to allow Super-admin to manage Hockey, trainers, players, fees, schedules, etc. Super admin can easily access players enrolled for hockey and also inform them about upcoming training times. This application is designed to organize the data of coaches, assistants, and Users. Also, this will be able to simplify the complexity of data of a vast number of users. Players will have some specific actions to add and modify their data. The Super admin communicates with players, coaches, and admin through email services. The System user's role would be there in addition to the Super admin. The System Users role is designed to help players in their training and The System users role can track records of Players also maintain results. This application will provide integration with some live

stream platforms so that players can get better facilities and training through this application. The Hockey training Management System objective is to provide a system which manages the activity of hockey at a time. The system users will consume less amount of time when compared to manual paper work through the automated system and also take care of all the training activity in an easier manner. This document aims to cover an in-depth overview of the functioning and features of the database. Trainers should be able to manage timetables, participants, and pieces of training. It will be able to check any report at any time. This system will provide the serving activity in a quick and easy manner. It will consume less amount of time as it is based on an automatic system. The purpose of our database would be to identify, clarify and organize the relationship between various entities of the "Sports Training Management System" like Super Admin, System User, Trainer, and Players who perform different activities according to their roles.

(B)Document the Requirements Collection/ Fact-Finding

(1) Background Reading

Our product is based upon the Sports management System for hockey.Some key features of our product are –
1. Provide great facilities for players.
2. Opportunities.
3. About Hockey.
4. Increasing the popularity of Hockey.

(1.1)Reference

1. https://www.sportlomo.com/sports/field-hockey/

2.http://www.fih.ch/inside-fih/hockey-resource-centre/eventsmanagement/fih-event-delivery/tournament-management-system/

3. https://www.uplifterinc.com/hockey-academy-software

4. https://www.arlo.co/training-management-system

(2) Interview Plan and Summary

System – Hockey Training Management System
Project Reference: SF/SJ/2003/1

Interviewee: 1. Martin Kumar Gamit
Designation – CEO
Contact - +91 9742567813
Email - martingamit142@gmail.com

Interviewer: 1. Graham Reid
Designation – CEO (Sports management system)
2. Tej Prakash
Designation - Co-Founder at Netrin Sports technology

Date: 10/08/2021
Time: 9:00
Duration: 45 minutes
Place: Sports Authority of India Ahmedabad (SAI)

Purpose of Interview:

Preliminary meeting to identify problems and new requirements regarding the system management for hockey training.

Agenda:

1. Improve the management system.
2. Better performance of the system.
3. New ideas for system upgrading.
4. Security of system.
5. How we can make the platform more suitable for users.

Documents to be brought to the interview:

1. Rough plan of structure of hockey training system.
2. Documents related to how to increase the security of the management system.
3. Documents related to new features that are required on the hockey training management system.

Questions

1.What are the dimensions of a Hockey field?

Ans 1. The dimensions of a hockey field are 91.4m X 55m.

2. Does your software have an online registration feature for players? Explain briefly?

Ans 2. Yes, our software has an online registration system. Our software consists of an easy registration process for players and teams, Advanced player registration and payment reports and collect payments online securely.

3. What is the data consumption rate of your software? Is it high or low?

Ans 3. The platform is specifically designed to consume less internet data.

4. How does your software perform scheduling of training?

Ans 4. Our software consists of scheduling of training in which we can create multi-team recurring practices, events.

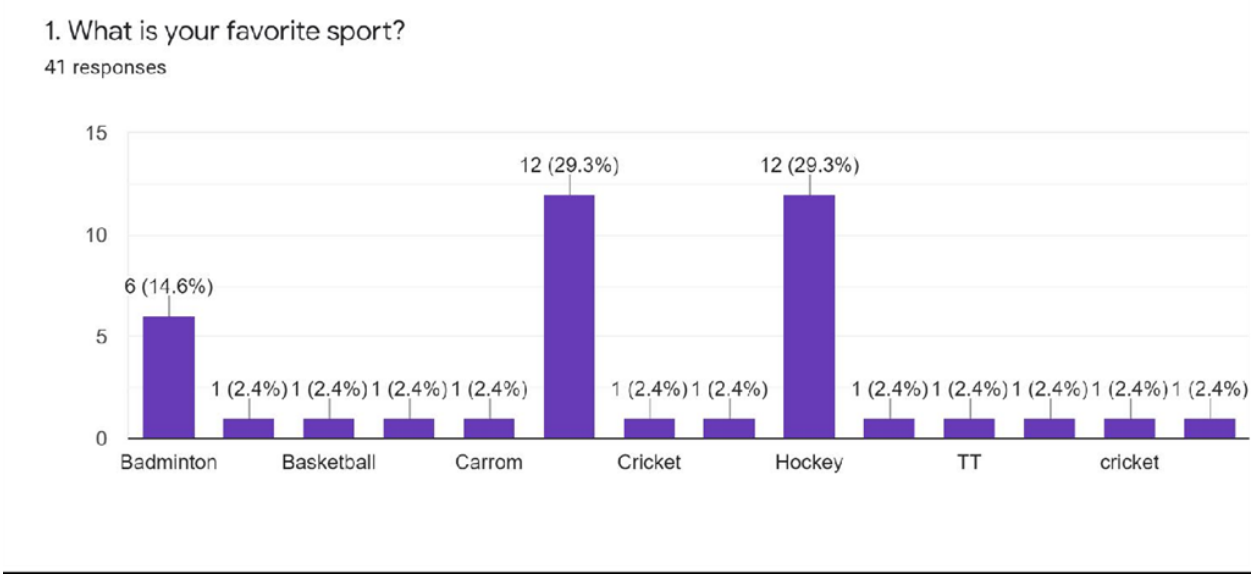
5. Do you have a website for your software? If yes then tell me some features of your website?

Ans 5. Yes, we have made a website for our software in which we included secure team pages, customized with few clicks, custom domain and SSL, updated in real Time, and cool widgets for all web pages.

(3) Questionnaire

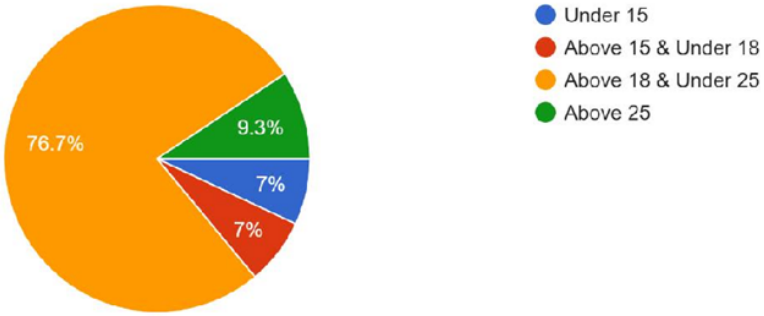
Google form link :

https://docs.google.com/forms/d/1QiROAIva62Ndqbqwj7H711SYc0XGBNCqzrvmN508oO0/prefill.



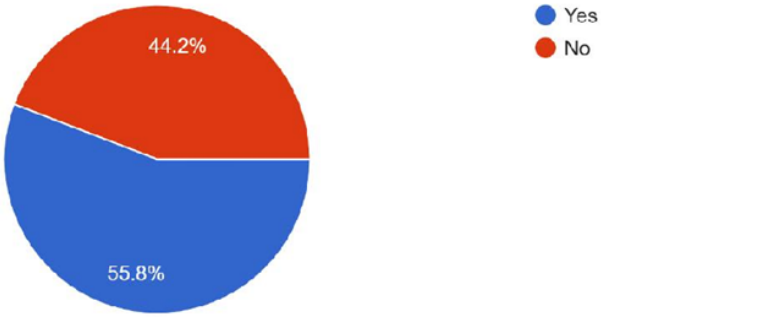
2. Which age group do you belong to ?

43 responses



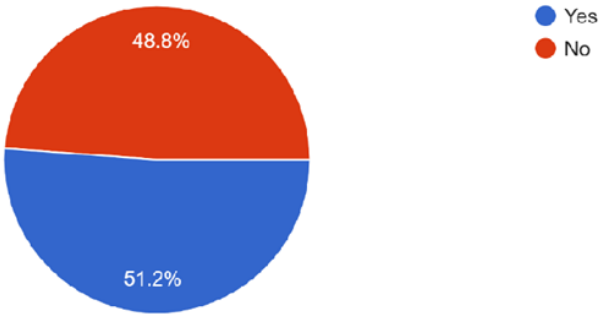
3. Do you watch Hockey ?

43 responses



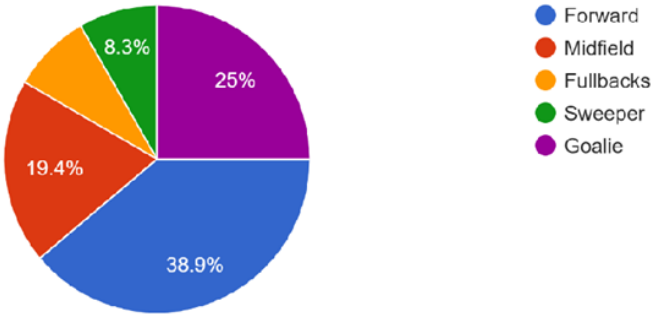
4. Do you have any experience in hockey?

43 responses



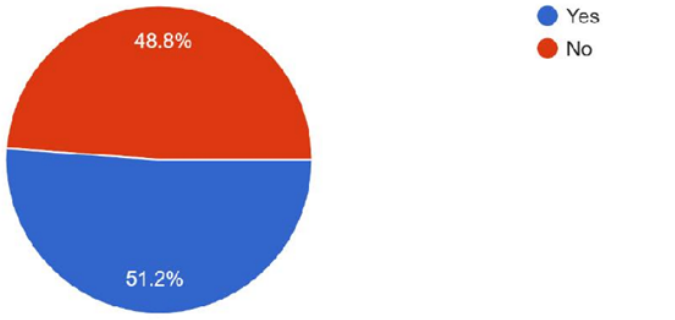
5. Which position do you play ?

36 responses



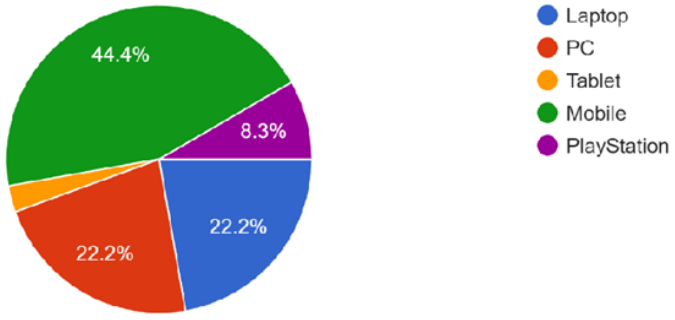
6. Do you play hockey video games?

43 responses



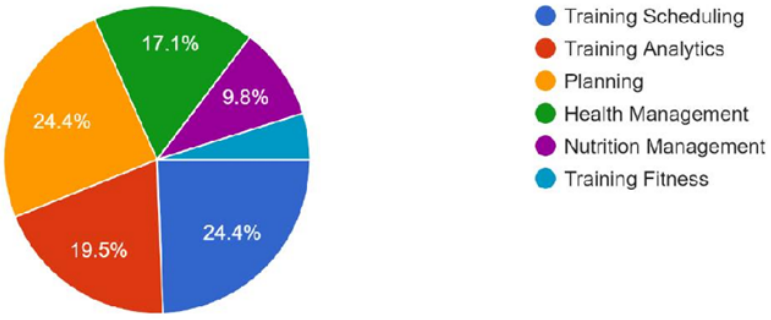
7. In which device do you play video games?

36 responses



8. What are the benefits that you think is important for this system?

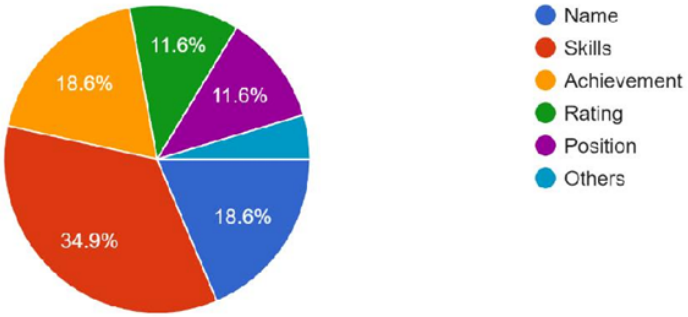
41 responses



---

9. Which information is shown on software for player info ?

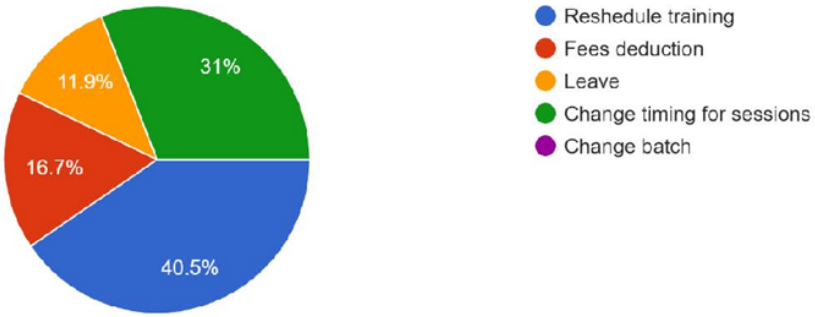
43 responses



---

10. Which request are allowed to ask the admin?

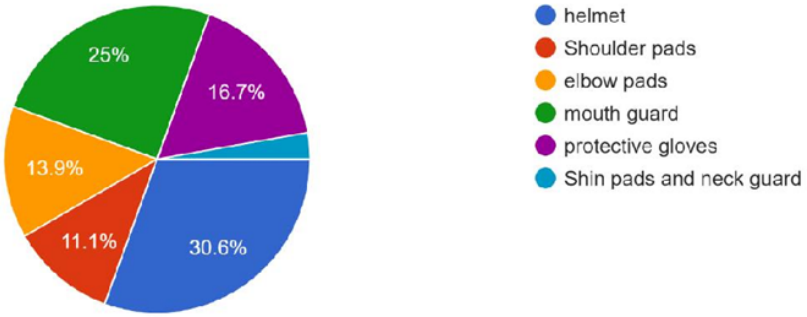
42 responses



---

11. Which equipment did you get from hockey training ?

36 responses



---

(4) Observation

System: Hockey Training Management System
Project Reference: SF/SJ/2003/12
Observations by: Lakshya Singh
Designation- Admin of hockey management system

Date: 10/08/2021
Time: 9:00 Duration: 45 minutes
Place: Administration (SAI Ahmedabad)

Observation

1. The software has an easy-to-use interface.
2. The software has a free mobile app to access team schedule, live chats, stats and much more.
3. This software also provides a player portal, sets goals for players, tracks player stats, and evaluates player performance.
4. The software has a low data consumption rate.
5. The software has a website that works for all which need no technical skill to access it.
6.Health of the players is very important in this sport. So, this software has injury tracking and discipline reporting.

(C) Fact -Finding -Chart

OBJECTIVE	TECHNIQUE	SUBJECTS	TIME COMMITMENT
To get the background of hockey practice ground and atmospheric conditions suitable for training.	**Background reading**	**Ground report (soil sample), Weather reports**	**1 DAY**
To manage the working of the system.	**Interview**	**One Manager**	**1 x 1 hour**
To restore proper functioning to the body.	**Interview**	**2 Physio**	**1 x 1 hour**

To facilitate the inclusion of healthy eating behaviors and empower their clients to take responsibility for their own health.	**Interview**	**Nutrition coach**	**1 x 1 hour**

To teach student athletes the basics of a particular sport.	**Interview**	**Head coach Assistant coach**	**2 x 1 hour**
To determine the role of support/admin.	**Interview**	**Admin staff**	**1.5 hour**
To manage the security of the system.	**Interview**	**Technical staff**	**1.5 hour**

(D)List Requirement

1.Service-provider and some upper management are the only who can modify or delete the data.
2.Good and Powerful security systems.
3. Training center should run on low network usage.
4. Performance should be fast.
5. Require large storage infrastructure.
6. Add new services seeing the requirements.
7. Discussion forums.
8.Take required action against any data privacy violation.
9.Improve current available functionality.

(E)User classes and Characteristics

1. Admin
The admin role can perform almost every action on every role. But some of them require the permission of other roles as well. They can access the whole database.

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

3. Players
They are the people who will use the services. They will need the database for viewing the various player details.

4. Super user
They are the only type of user who can update, add or delete existing services.

5. Trainer
They train the players and keep updates of the players about their performance.

[F]Operating Environment

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

Hardware and Software requirements

For user:

- Computer:
- Browser Support:
• Google Chrome (recommended)
• Firefox
• Safari
• Internet Explorer 11
• Microsoft Edge
- Reliable internet with minimum 500kbps internet speed
- Minimum 1 GB RAM

- Mobile:
• Android 5.0+
• iOS and iPad 9.0+

Server side:
• Linux/ windows hosting database
• data storage – 50TB • Sufficient RAM
• Apache MySQL database • Connectivity: bandwidth
• Cloud based services like AWS (Amazon Web Services), Azure can be used.
• Language: Java, C, C++, Python.

(G) Product Functions

1. Hockey training solutions - Hockey training is especially focused on optimal performance in hockey. Its main aim is to develop the performance capacity of players, so that they achieve the highest possible performance. Record the performance of each player for analysis and work on the basis of that to each player.

2. Fitness assessment. - Identifies the fitness level of each player and serves as a baseline, or starting point of the body's fitness. Figure out the players training needs and goals. Then compare the progress over time to the initial fitness assessment.

3. Health management service- Plays a role of leadership and gives direction to organizations that deliver personal health services.Analysis the health of the players and taking care of the players.

4. Nutrition management function - Role in healthcare delivery, working closely with many other healthcare professionals to ensure players receive the best nutritional care possible, and have the best healthcare experience and outcome. Taking care of proper nutrition should be given to the players for their better performance.

5. Proper training - It enhances players competence in hockey and familiarizes them with the role required of others on their team.

6. Training VR & AR solution - It provides a better way of practicing skills. It analyzes the movement of the players during the practice sessions.

7. Result analysis - It analyzes all the results of the players related to health report, players performance, match performance and many more.

Functional Requirement

The functional requirements of a hockey training management system are those requirements that are necessary to the eye of the players. Here we try to make the module possible to accomplish the need of the desired function. A few of its functional requirements are as follows

1. Interested players visit our website or system and registration will be done with the details of him and the admin will collect all the details of the players and tell all the details about the system.

2. Before that admin collects all the details in a very classified manner according to the criteria and assigns the players.

3. Now players are allotted trainers and, in our system, players can change the trainers.

4. System also provides functions which let the user view their history, information and time schedule, and reserve the equipment / arena/time.

5. Our project qualifies all the factors of helpful and not helpful consequently and the system is up to mark performance devices. Here we'd prefer to need the care of a few lots of things before heading towards the system. many sensible intuitive interfaces are usually created. that ultimately builds an interface easy to use for a lengthy time.

(H)Privileges

1. Update services in database Service Provider
2. Access to users' data Some upper management and users can access their own data but not others' data.
3. System Update (adding new features) IT Department employees after successful testing (System Administrators).

(I)Assumptions

1.Users have an id and a password to login into our system.
2.Users have a PC or mobile which is able to connect with the network.
3.Users have needed stable internet connection (Approx.1 MB data rate required)
4.Users' computer or mobile has the required Hardware which are mentioned earlier.

(J)Business Constraints

1.There is no copyright violation by our product.
2.User's data will not be shared with third parties.
3.Premium users will get additional benefits and services from our company.
4.The approximate budget is INR 2 lakh. These prices may vary after the complete development of the application

Section2: Noun Analysis

Problem Description

Hockey training management system is designed to simplify the tracking and development of individual athletes. Easily track skill evaluations on the ground using tablet or smartphone devices. When training is complete, you can quickly create digital report cards to empower players with a sense of achievement. If one child in the family plays hockey, the other is likely to play too. That's why the hockey training management system is built with hockey families in mind. The system allows parents to register multiple kids on one invoice, and track weekly activities with family calendars and individual athlete profiles. This project is for the online implementation of the sports management system. The database will have data of players, team coach, assistants, and trainers. This application is designed to allow Superadmin to manage Hockey, trainers, players, fees, schedules, etc. Super admin can easily access players enrolled for hockey and also inform them about upcoming training times. This application is designed to organize the data of coaches, assistants, and Users. Also, this will be able to simplify the complexity of data of a vast number of users. Players will have some specific actions to add and modify their data. The Super admin communicates with players, coaches, and admin through email services. The System user's role would be there in addition to the Super admin. The System Users role is designed to help players in their training and The System user's role can track records of Players also maintain results. This application will provide integration with some live stream platforms so that players can get better facilities and training through this application. The Hockey training Management System objective is to provide system which manages the activity of hockey at a time. The system users will consume less amount of time when compared to manual paper work through the automated system and also take care of all thetraining activity in a easier manner. This document aims to cover an in-depth overview of the functioning and features of the database. Trainers should be able to manage timetables, participants, and piecesof training. It will be able to check any report at any time. This system will provide the serving activity in quick and easy manner. It will consume less amount of time as it is based on automatic system. The purpose of our database would be to identify, clarify and organize therelationship between various entities of the "Sports Training Management System" like Super Admin, System User, Trainer, and Players who perform different activities according to their roles. The functional requirements of a hockey training management system are those requirements that are necessary to the eye of the players. Here we try to make the module possible to accomplish the need of the desired function. A few of its functional requirements are as follows 1. Interested players visit our website or system and registration will be done with the details of him and admin will collect all the details of the players and tell all the details about the system. 2. Before that admin collect all the details in a very classified manner in according to the criteria and assigns the players. 3. Now players are allotted trainers and, in our system, players can change the trainers. 4. System also provide function which let the user view their history, information and time schedule, reserve the equipment / arena/time. 5. Our project qualifies all the factors of helpful and not helpful consequently and the system is up to mark performance device. Here we'd prefer to need the care of few lots of things before heading towards the system. the many sensible intuitive interfaces are usually created. that ultimately build interface easy to use for a lengthy time. The admin role can perform almost every action on every role. But some of them require the permission of other roles as well. They can access the whole database. Consists of the CTO and the database administrator. These people manage the database and maintain it. They are the people who will use the services. They will need the database for viewing the various player details. They are the only type of user who can update, add or delete existing services. They train the players and keep updates of the players about their performance. 1.Hockey training solutions Hockey training is especially focused on optimal

performance in hockey. Its main aim is to develop the performance capacity of players, so that they achieve the highest possible performance. Record the performance of each players for analysis and work on the basis of that to each players. Identifies the fitness level of each players and serves as a baseline, or starting point of body's fitness. Figure out the players training needs and goals. Then compare the progress over time to the initial fitness assessment.service Plays a role to leadership and give direction to organizations that deliver personal health services.Analysis the health of the players and taking care of the players. function Role in healthcare delivery, working closely with many other healthcare professionals to ensure players receive the best nutritional care possible, and have the best healthcare experience and outcome. Taking care of proper nutrition should be given to the players for their better performance. It enhance players competence in hockey and familiarize them with the role required of others on their team. It provides the better way of practicing skills. Its analyst the movement of the players during the practice sessions.It analysis all the result of the players related to health report, players performance, match performance and many more.

1.Noun and verb analysis.

- **Find the are nouns (entities) or verbs (relationships) in sentences of the problem description using Noun Analysis Method.**
- **List all the extracted Nouns & Verbs in the below-given table format.**

NOUNS	VERBS
Project	is
Implementation	Will have
Sports	Is designed
Management	allow
System	manage

Database	**can**
Data	**enrolled**
Players	**inform**
Team	**Is designed**
Coach	**organize**
Assistance	**Will be**
Players	**simplify**
Trainer	**Will have**
Application	**add**
Hockey	**modify**
Fees	**communicate**
Schedules	**email**

Super admin	**Would be**
Access	**Is designed**
Times	**help**

Users	**can**
complexity	**maintain**
number	**Will provide**
Action	**live**
Service	**Can get**
Role	**Is designed**
System	**simplify**
Track	**tracking**

Record	**using**
Result	**create**
integration	**empower**
platform	**plays**
Stream	**built**
facilities	**register**
sentence	**allows**
development	**Trains**
athletes	**Analyze**
skill	**Issue**
evaluation	**Register**
ground	

Tablet	
device	
achievement	
Report cards	
sense	
Hockey families	
mind	
parents	

kids	
track	
activities	
family calendars	
athlete profiles	
Equipment	
Serial No.	
Name	
Quality	
Description	
Weight	
User ID	

Password	
Login	
Website	
Personal	
DOB	
Location	
Data of joining	
Password	
Equipment Renew	
Equipment Return	
Registration	
Distribution	

Time	
CO	

1. Criteria for Truncating Initial Noun List

Candidate Entity Set	**Candidate Attribute Set**	**Candidate Relationship Set**
1.Admin	**Schedules, Data , Database , Management ,Facilities ,Fees**	**Organize ,Register, Communication, Help, Maintain , Issue**
2. Equipment	**Serial No. , Name, Quantity , Description**	**Return, add**
3. Ground	**Area, Dimension, Equipment**	**Designed, Manufacture**
4. Trainer	**Name , Experience , Trainer_ID, Skill**	**Manage, Plays, Tracking , Maintain**

5. Personal	**Name, DOB , Date of joining , Age**	
6. Website	**ID , Password**	**Log in**
7. Users	**User_ID, Username , Password**	**Role**
8. Result	**Data, Player_ID , Performance**	**Analyze**

Reject Noun

NOUN	**REJECTED REASONS**
Record	**Duplicates**
Athletes	**Duplicates**
Application	**Duplicates**

Location	**Duplicates**
Complexity	**Irrelevant**
Number	**General**
Action	**Duplicates**
Track	**Duplicates**
Integration	**Irrelevant**
Stream	**Vague**
Platform	**General**
Sentence	**Vague**
Development	**General**
Weight	**Vague**
Project	**Vague**

Implementation	**Duplicates**
Times	**Irrelevant**
Assistance	**Duplicates**
Distribution	**Duplicates**
Date	**General**
CO	**Vague**
Type	**Duplicates**
Equipment Return	**Irrelevant**
Equipment Renew	**Irrelevant**

Reject Verb

VERBS	REJECTED REASONS
Assign	Duplicates
Will be	Irrelevant
Modify	Duplicates
See	Irrelevant
Contain	Duplicates
Own	Vague
Will have	Irrelevant
Simplify	Duplicates
Handle	Vague
Scan	Irrelevant

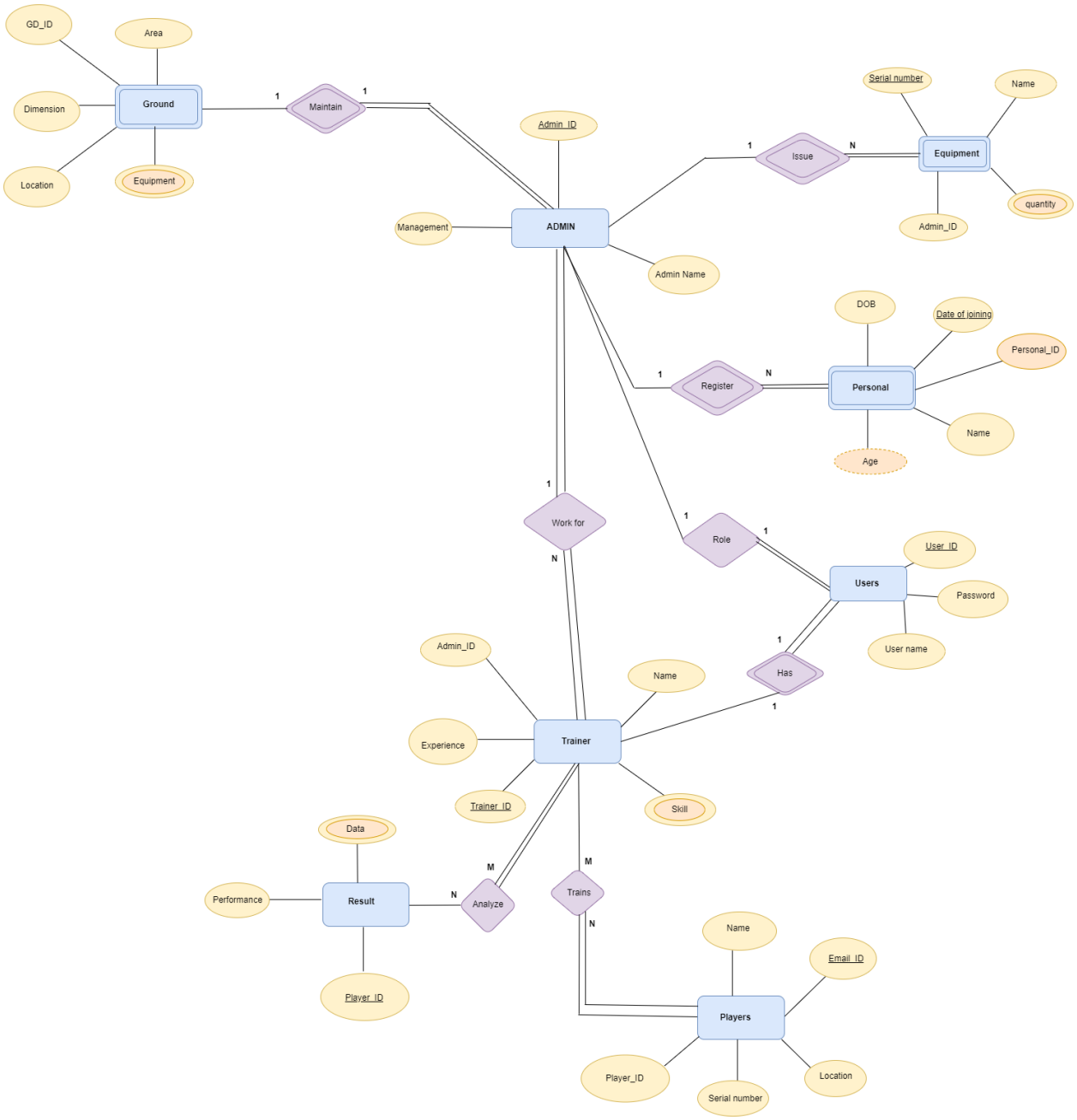
Receive	**General**
Contain	**Duplicates**
Will provide	**Irrelevant**
Damage	**Vague**
Require	**Duplicates**
Access	**Duplicates**
Authorize	**Vague**
Check	**General**

View	**Vague**
Relate	**Duplicates**
Break	**Vague**

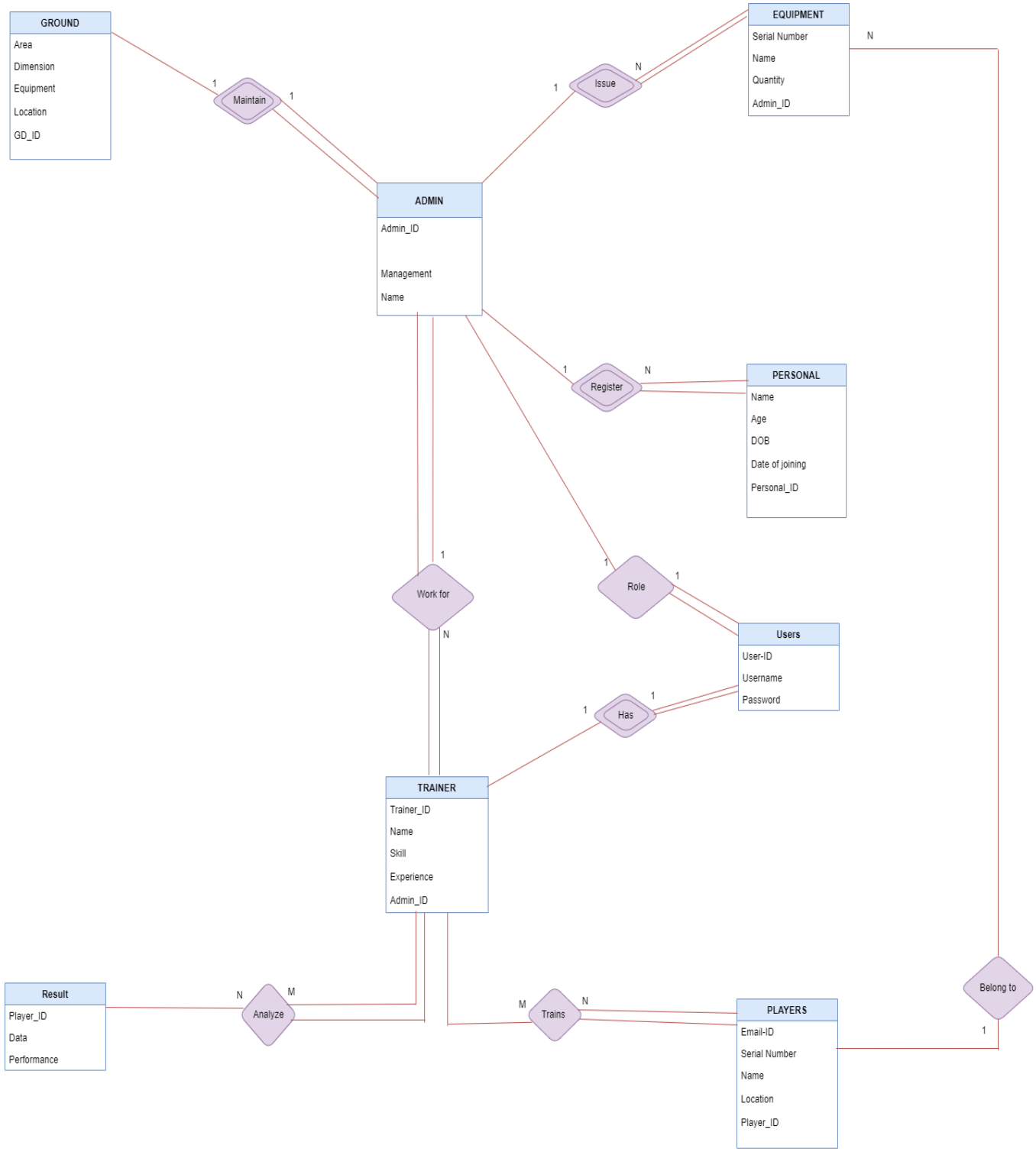
Categorize	**General**
deploy	**Duplicates**
Incur	**Vague**

Section2: E-R Diagrams all versions

ER DIAGRAMS V-2



E-R Diagram Final



Section4: Conversion of Final ER-Diagram to Relational Model

Relational Model

1.Admin (Admin_ID, Management, Name)

2.Equipment (Serial number, Name , Quantity , Admin _ID)

3.Personal(Name , Age , DOB , Date of_ joining , Personal_ID)

4.Ground(Area , Dimension , Equipment,GD_ID)

5.Users (User_ID , Password , Username)

6.Trainer(Name , Experience , Trainer_ID , Skill , Admin_ID)

7.Players(Name, Email_ID, Serial Number, Location, Player_ID)

8.Result (Data , Performance , Player _ID)

Section5: Normalization and Schema Refinement

Normalization and Schema Refinement

1.ADMIN(Admin_ID, Management, Admin_Name)

Pk Dependencies - Admin_ID→Admin_Name, Admin_ID→Management.

Functional Dependencies

Admin_ID→Admin_Name
Admin_ID→Management

Redundancy - None

Anomalies -

Insert - Suppose a new Admin joins the , who is under training and currently not assigned to any management then we would not be able to insert the data into the table if the management field doesn't allow nulls.

Delete - Suppose, if at a point of time the system closes the one management department like ground then deleting the rows that are having management as ground would also delete the information of Admin_Name since it is assigned only to this department.

Update - In the above if we have two rows for Admin_Name as he belongs to two management departments of the system . If we want to update the department of that Admin_Name then we have to update the same in two rows or the data will become inconsistent.

2.Equipment (Serial number, Name , Quantity,Admin_ID)

Pk Dependencies - Serial number→ Name, Serial number→ Quantity, Serial number→Admin_ID

Functional Dependencies -

Serial number→ Name
Serial number→ Quantity
Serial number→Admin_ID

Redundancy

For distributing the equipment , we need only the Serial number as the primary key and not the whole candidate key {Name, Quantity}. Admin-ID is redundant in this case.
Anomalies -

Insert - If a new player joins the system, So the player is currently not assigned any equipment then we would not be able to insert the data into the table if the Quantity doesn't allow nulls.

Delete - Suppose, if at a point of time the system closes the one player name as Ravi does not join the system then deleting the rows of the quantity of equipment would also delete the information of that player since the quantity of equipment was assigned to that player.

Update - In the above if we have two rows of the same Name as they get the quantity of equipment . If we want to update the quantity of that particular name then we have to update the same in two rows or the data will become inconsistent.

3.Personal(Name , Age , DOB , Date of joining, Personal_ID)

Pk Dependencies - Personal_ID→ Name, Personal_ID→ Age, Personal_ID→DOB, Personal_ID→Date of joining

Functional Dependencies

Personal_ID→ Name
Personal_ID→ Age
Personal_ID→DOB
Personal_ID→Date of joining

Redundancy - None.

Anomalies -

Insert - If a new player joins the system, So the player has not submitted the personal details. We would not be able to insert the data into the table.

Delete - Suppose, if at a point of time the system closes the one player name as Ravi does not join the system then deleting the rows of the specific player would also delete the information of that player.

Update - In the above if we have two rows of the same Name or same address . Then we want to update the address of that particular name then we have to update the same in two rows or the data will become inconsistent.

4.Ground(Location , Dimension , Equipment,GD_ID, Area)

Pk Dependencies - GD_ID→Location, GD_ID→Dimension, GD_ID→Equipment, GD_ID→Area.

Functional Dependencies -

GD_ID→Location
GD_ID→Dimension
GD_ID→Equipment
GD_ID→Area

Redundancy- None.

Anomalies -

Insert - If a new ground joins the system, If the system does not have ground details.We would not be able to insert the data into the table.

Delete - If we try to delete a tuple from Officials with some GD_ID that does not exist any details, then it will not allow the deletion.

Update - In the above if we have two rows of the same Dimension or same location . Then we want to update the location of that Dimension then we have to update the same in two rows or the data will become inconsistent.

5.Users (User_ID , Password , Username)

Pk Dependencies - User_ID→Password, User_ID→ Username.

Functional Dependencies -

User_ID→Password
User_ID→ Username

Redundancy - None.

Anomalies -

Insert - If a User joins the system, So the User is currently not assigned any Username and password then we would not be able to insert the data into the table if the Quantity doesn't allow nulls.

Delete - Suppose, if at a point of time the system closes the one user does not join the system then deleting the rows of the username would also delete the information of that user since the Username was assigned to that user .

Update - In the above if we have two rows of the same password or same username . Then we want to update the password of that username then we have to update the same in two rows or the data will become inconsistent.

6.Trainer(Name , Experience , Trainer_ID , Skill, Admin_ID)

Pk Dependencies- Trainer_ID→ Name, Trainer_ID→Experience, Trainer_ID→Skill, Trainer_ID→Admin_ID.

Functional Dependencies -

Trainer_ID→ Name
Trainer_ID→Experience
Trainer_ID→Skill
Trainer_ID→Admin_ID

Redundancy - For allotting the trainers , we need only the Trainer_ID as the primary key and not the whole candidate key {Experience , Trainer_ID , Skill}. Admin-ID is redundant in this case.

Anomalies -

Insert - If a New trainer joins the system, If the Trainer is currently not assigned any Trainer-ID then we would not be able to insert the data into the table if the Trainer_ID doesn't allow nulls.

Delete - Suppose, if at a point of time the system closes the one trainer does not join the system then deleting the rows of the trainer would also delete the information of that trainer since the trainer_ID was assigned to that trainer .

Update - In the above if we have two rows of the same trainer name or same Skill or experience . Then we want to update the skill of that trainer then we have to update the same in two rows or the data will become inconsistent.

7.Players(Player_ID,Name, Email_ID, Serial Number, Location)

Pk Dependencies - Player_ID→ Name, Player_ID→ Email_ID, Player_ID→Serial Number, Player_ID→Location.

Functional Dependencies

Player_ID→ Name
Player_ID→ Email_ID
Player_ID→Serial Number
Player_ID→Location

Redundancy - None

Anomalies -

Insert - If a New player joins the system, If the player is currently not assigned any player_ID then we would not be able to insert the data into the table if the player_ID doesn't allow nulls.

Delete - Suppose, if at a point of time the system closes the one player does not join the system then deleting the rows of the player would also delete the information of that player since the player_ID was assigned to that player.

Update - In the above if we have two rows of the same player name or location . Then we want to update the location of that player then we have to update the same in two rows or the data will become inconsistent.

8.Result (Data , Performance , Player _ID)

Pk Dependencies - Player_ID→ Performance, Player_ID→Data.

Functional Dependencies -
Player_ID→ Performance
Player_ID→Data

Redundancy - None

Anomalies -

Insert - If a New player joins the system, If the player is currently not analysed by the trainers then we would not be able to insert the data into the table if the data doesn't allow nulls.

Delete - Suppose, if at a point of time the system closes the one player does not join the system then deleting the rows of the player would also delete the information of that player since the data was assigned to that player.

Update - In the above if we have two rows of the same player_ID or data . Then we want to update the data of that player_ID then we have to update the same in two rows or the data will become inconsistent

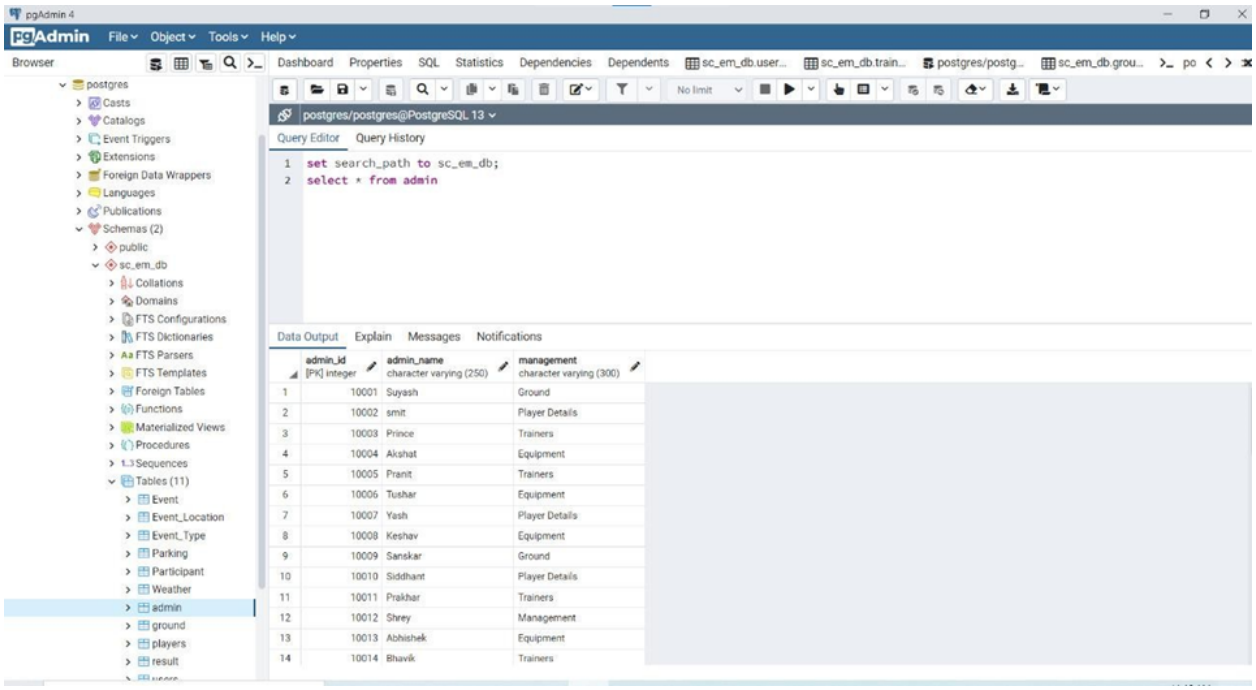
Section6: SQL: Final DDL Scripts, Insert statements, 40 SQL Queries with Snapshots of output of each query

Final DDL Scripts

1.ADMIN(Admin_ID, Management, Admin_Name)

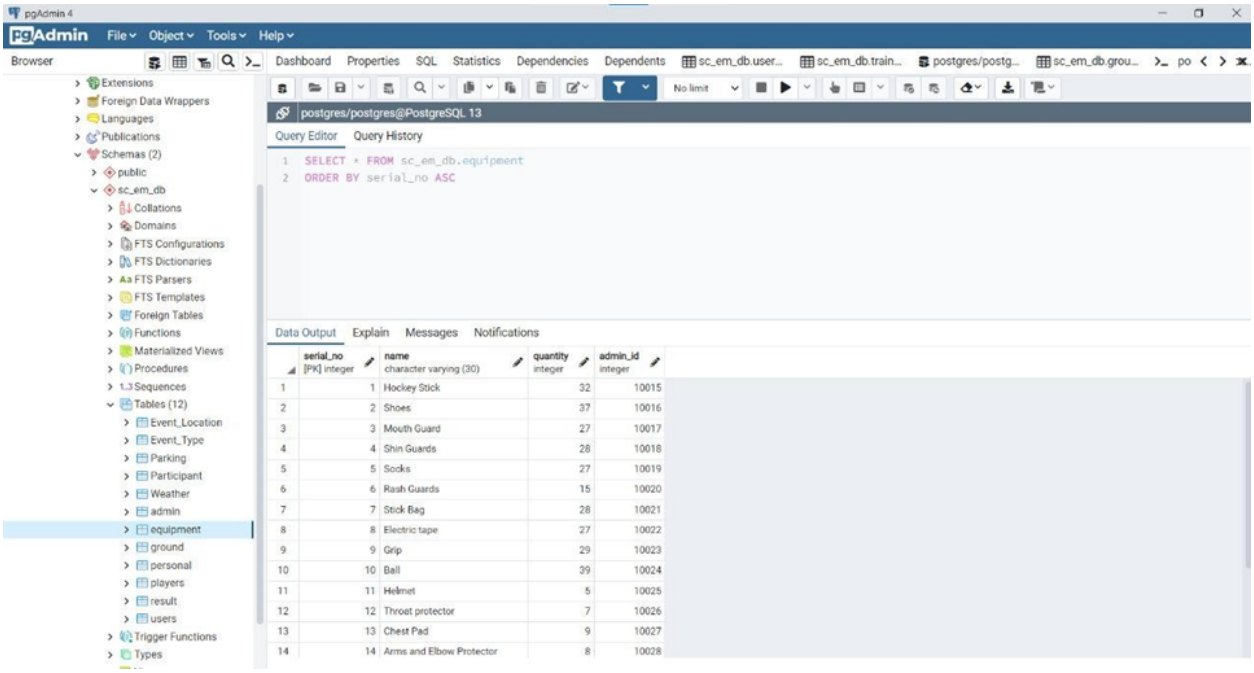
CREATE TABLE ADMIN (Admin_ID INT

PRIMARY KEY , Admin_name varchar(250),Management varchar(300));



2.Equipment (Serial number, Name , Quantity,Admin_ID)

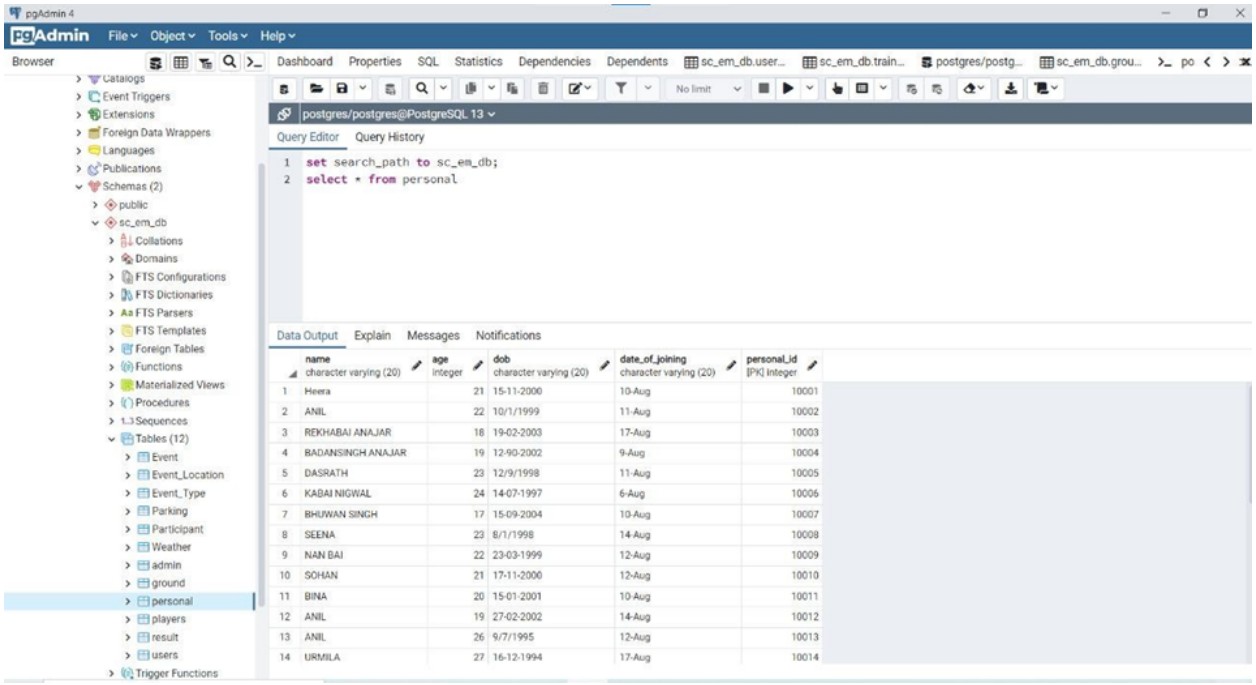
CREATE TABLE Personal(Name character varying(20), Age INT ,DOB character varying(20),
Date _of_ joining character varying(20),
Personal_ID INT PRIMARY KEY);



3.Personal(Name , Age , DOB , Date of joining, Personal_ID)

_CREATE TABLE Equipment(serial_no INT PRIMARY KEY, name VARCHAR(30),

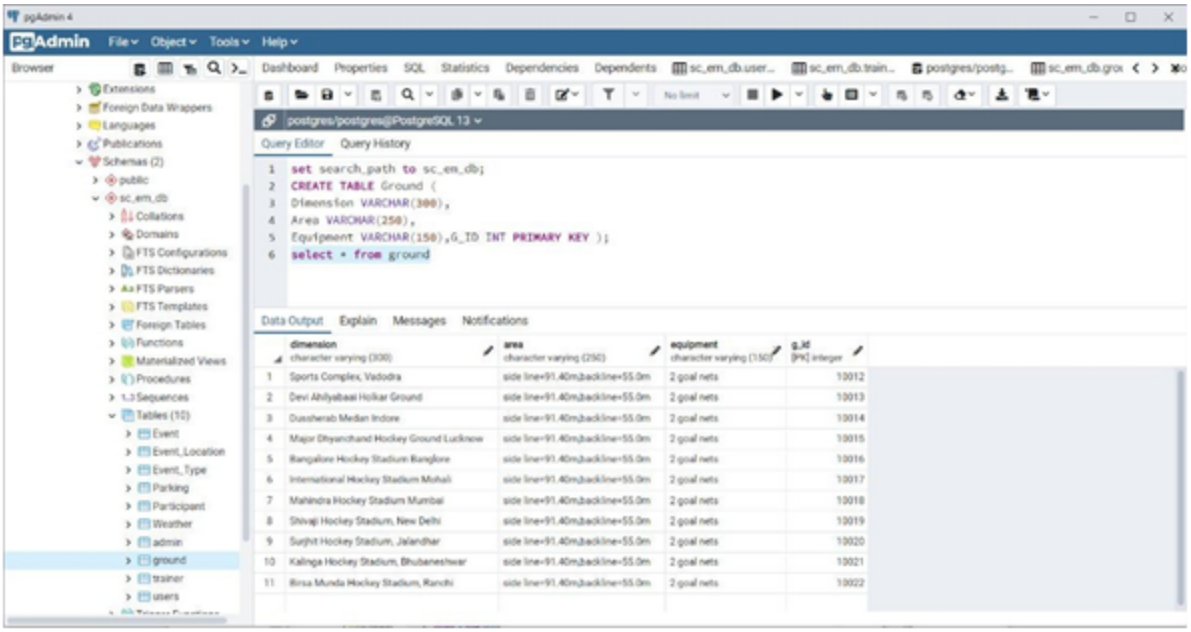
Quantity INT,admin_id INT, FOREIGN KEY (admin_id)
references sc_em_db.admin ON UPDATE CASCADE ON DELETE
CASCADE
);



	name character varying (20)	age integer	dob character varying (20)	date_of_joining character varying (20)	personal_id [PK] integer
1	Heera	21	15-11-2000	10-Aug	10001
2	ANIL	22	10/1/1999	11-Aug	10002
3	REKHABAI ANAJAR	18	19-02-2003	17-Aug	10003
4	BADANSINGH ANAJAR	19	12-90-2002	9-Aug	10004
5	DASRATH	23	12/9/1998	11-Aug	10005
6	KABAI NIGWAL	24	14-07-1997	6-Aug	10006
7	BHUWAN SINGH	17	15-09-2004	10-Aug	10007
8	SEENA	23	8/1/1998	14-Aug	10008
9	NAN BAI	22	23-03-1999	12-Aug	10009
10	SOHAN	21	17-11-2000	12-Aug	10010
11	BINA	20	15-01-2001	10-Aug	10011
12	ANIL	19	27-02-2002	14-Aug	10012
13	ANIL	26	9/7/1995	12-Aug	10013
14	URMILA	27	16-12-1994	17-Aug	10014

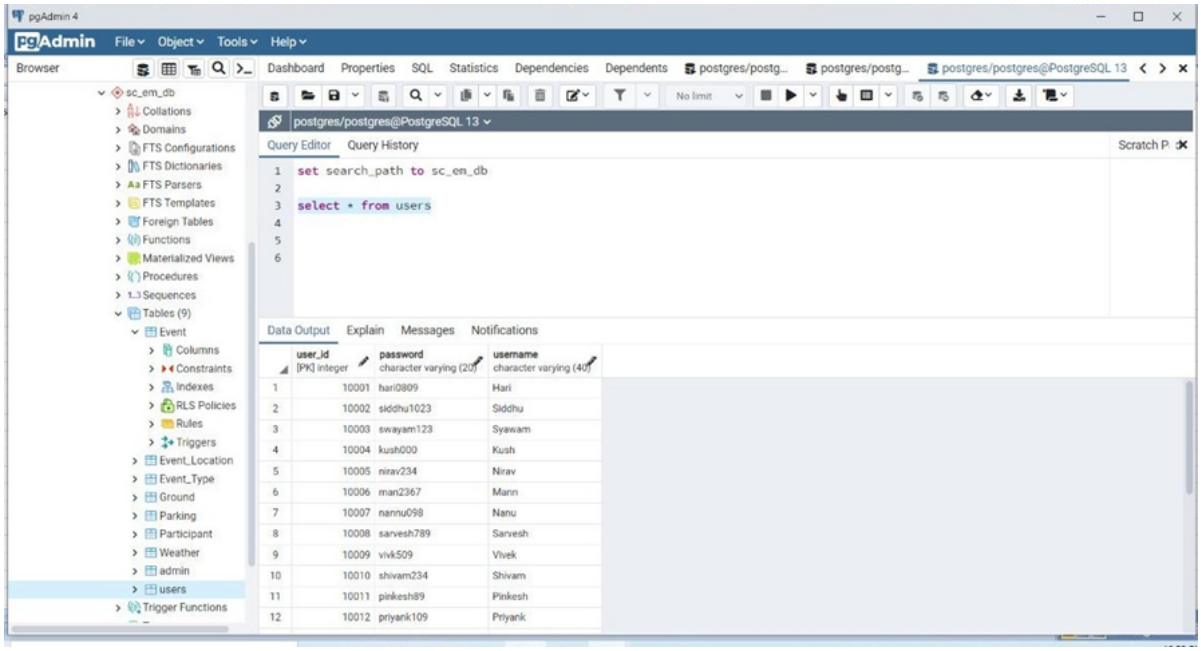
4.Ground(Location , Dimension , Equipment,GD_ID, Area)

CREATE TABLE Event (Team VARCHAR(150) , Data INT ,Result numeric ,Schedules time PRIMARY KEY, FOREIGN KEY(Admin_ID) references ADMIN ON DELETE CASCADE);


pgAdmin 4
pgAdmin
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (2)
public
sc_em_db
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Procedures
Sequences
Tables (10)
Event
Event_Location
Event_Type
Parking
Participant
Weather
admin
ground
trainer
users
postgres/postgres@PostgreSQL 13
Query Editor
Query History
1 set search_path to sc_em_db;
2 CREATE TABLE Ground (
3 Dimension VARCHAR(300),
4 Area VAROMAR(250),
5 Equipment VARCHAR(150),G_ID INT PRIMARY KEY);
6 select * from ground
Data Output
Explain
Messages
Notifications
dimension character varying (300)
area character varying (250)
equipment character varying (150)
g_id [PK] integer
1 Sports Complex, Vadodra side line=91.40m,backline=55.0m 2 goal nets 10012
2 Devi Ahilyabaai Holkar Ground side line=91.40m,backline=55.0m 2 goal nets 10013
3 Dussherab Medan Indore side line=91.40m,backline=55.0m 2 goal nets 10014
4 Major Dhyanchand Hockey Ground Lucknow side line=91.40m,backline=55.0m 2 goal nets 10015
5 Bangalore Hockey Stadium Banglore side line=91.40m,backline=55.0m 2 goal nets 10016
6 International Hockey Stadium Mohali side line=91.40m,backline=55.0m 2 goal nets 10017
7 Mahindra Hockey Stadium Mumbai side line=91.40m,backline=55.0m 2 goal nets 10018
8 Shivaji Hockey Stadium, New Delhi side line=91.40m,backline=55.0m 2 goal nets 10019
9 Surjhit Hockey Stadium, Jalandhar side line=91.40m,backline=55.0m 2 goal nets 10020
10 Kalinga Hockey Stadium, Bhubaneshwar side line=91.40m,backline=55.0m 2 goal nets 10021
11 Birsa Munda Hockey Stadium, Ranchi side line=91.40m,backline=55.0m 2 goal nets 10022

5.Users (User_ID , Password , Username)

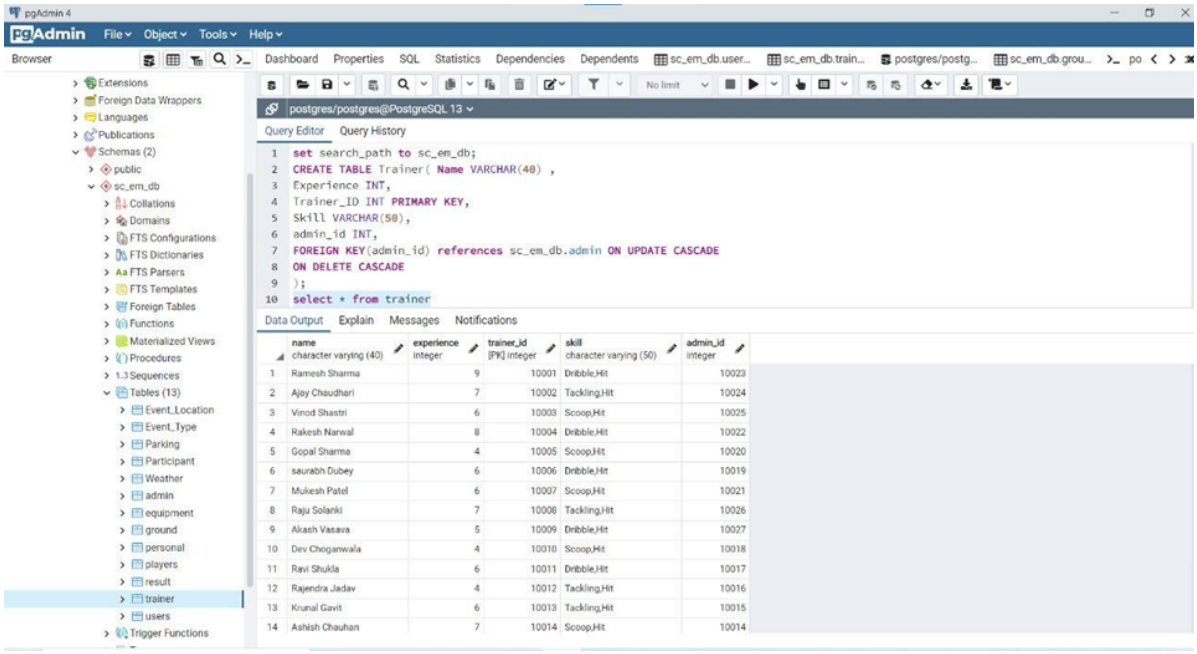
CREATE TABLE Users(User_ID INT PRIMARY KEY,Password VARCHAR(20) , Username VARCHAR(40));



6.Trainer(Name , Experience , Trainer_ID , Skill, Admin_ID)

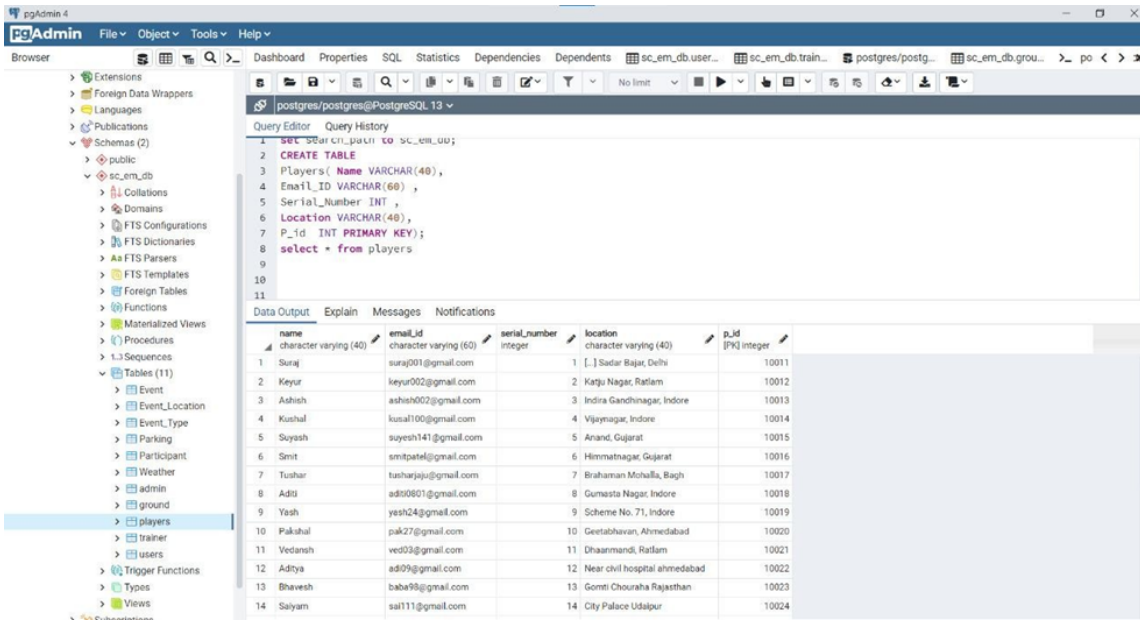
CREATE TABLE Trainer(Name VARCHAR(40) ,Experience INT, Trainer_ID INT PRIMARY KEY,Skill VARCHAR(50),admin_id INT,

FOREIGN KEY(admin_id) references sc_em_db.admin ON UPDATE CASCADE ON DELETE CASCADE);



7.Players(Player_ID,Name, Email_ID, Serial Number, Location)

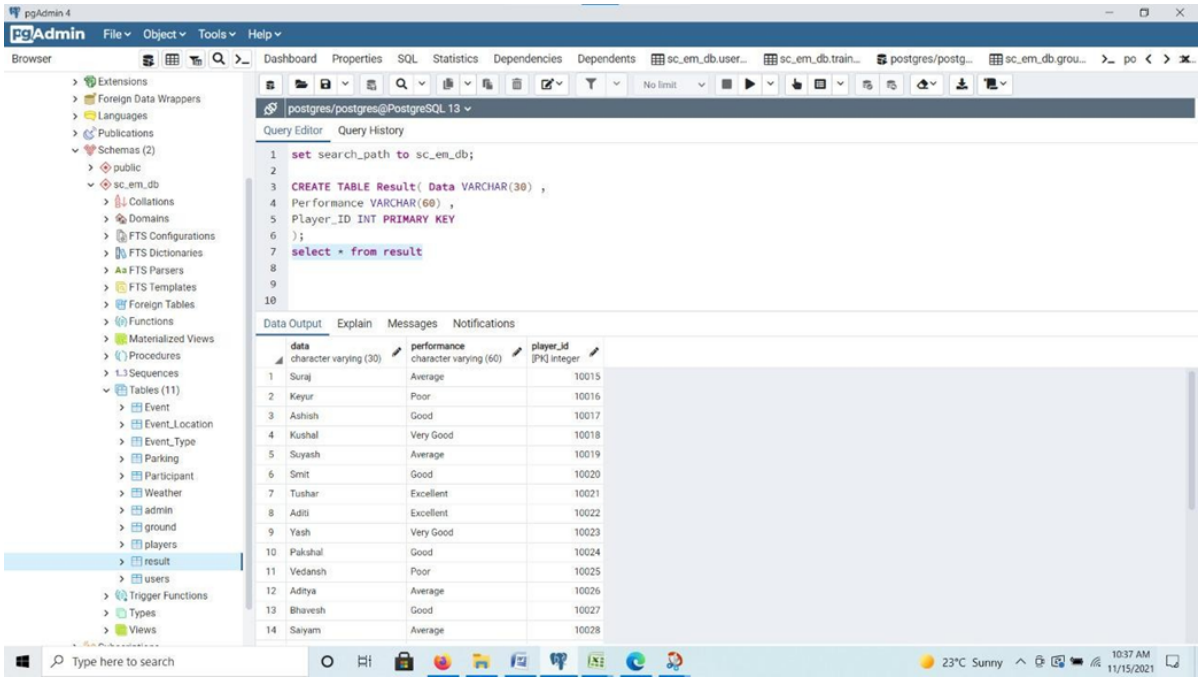
CREATE TABLEPlayers(Name VARCHAR(40), Email_ID VARCHAR(60) ,Serial_Number INT ,Location VARCHAR(40),p_id INT PRIMARY KEY,);



8.Result (Data , Performance , Player _ID)

_CREATE TABLE Result(Data VARCHAR(30) ,

Performance VARCHAR(60) , Player _ID INT PRIMARY KEY);



40 SQLQueries

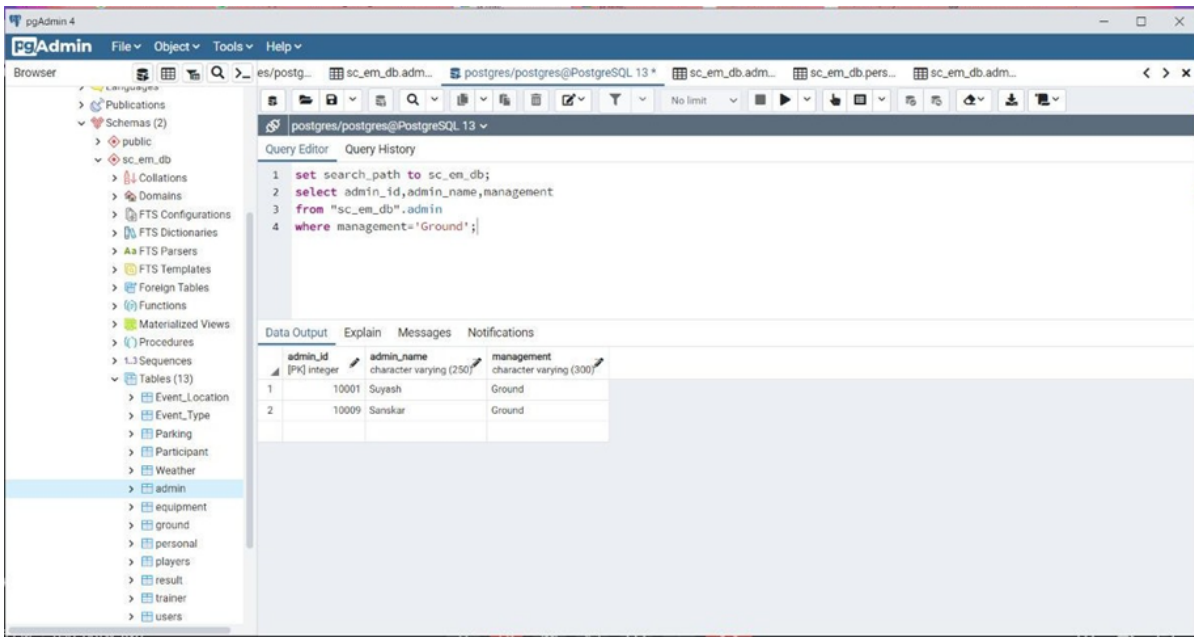
1. Retrieve the data of the Admin with their ID who have allotted management on ground.

set search_path to sc_em_db;

select admin_id,admin_name,management from "sc_em_db".admin

where management='Ground';

SNAPSHOT:



2.Display the name with age 21 from the table personal.

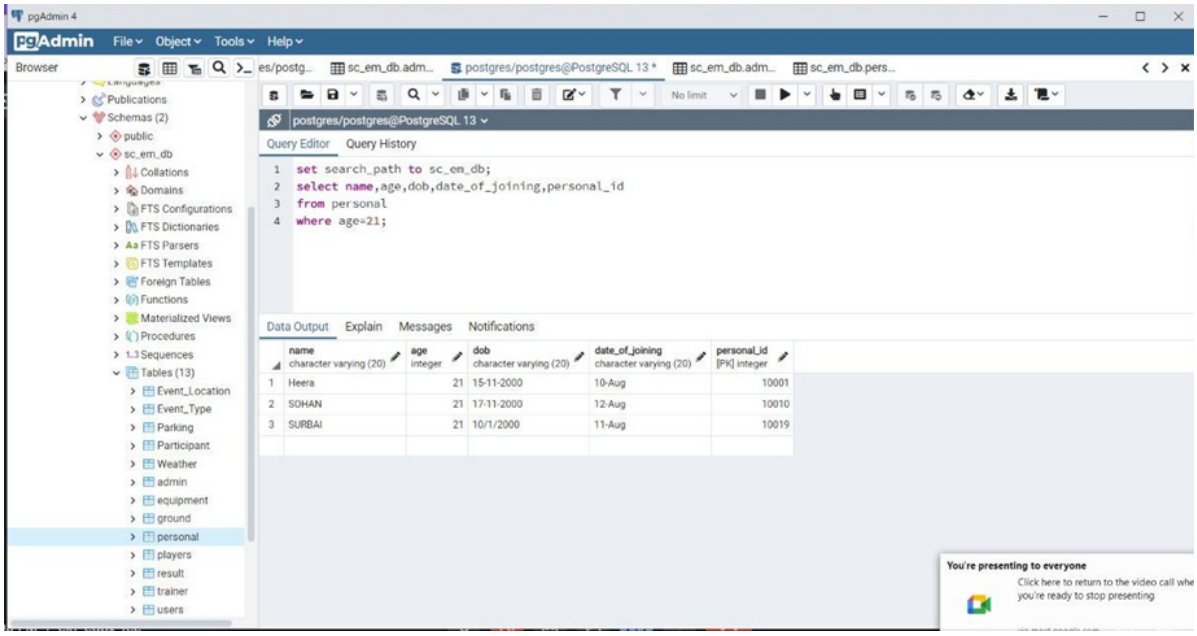
SQL query:

set search_path to sc_em_db;

Select name, age,name, data_of_joining , personal_id From personal

Where age = 21;

SNAPSHOT:



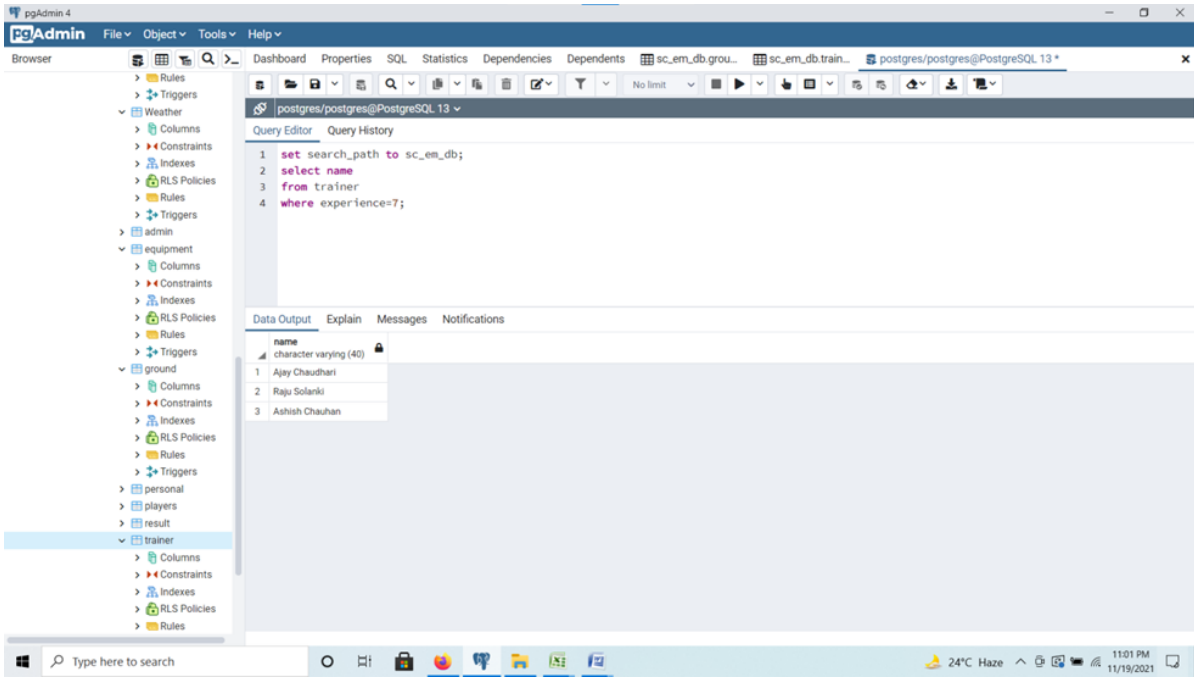
3.List the name of trainers having experience of 7 years in training.

set search_path to sc_em_db;

select name

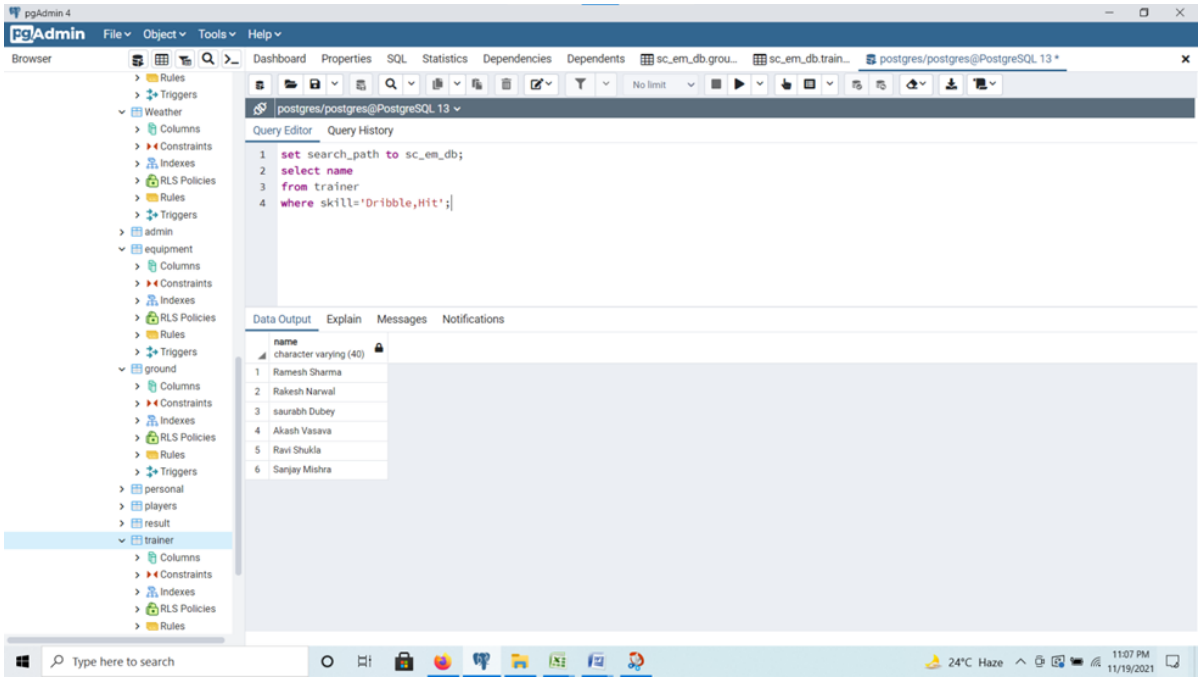
from trainer

where experience=7;



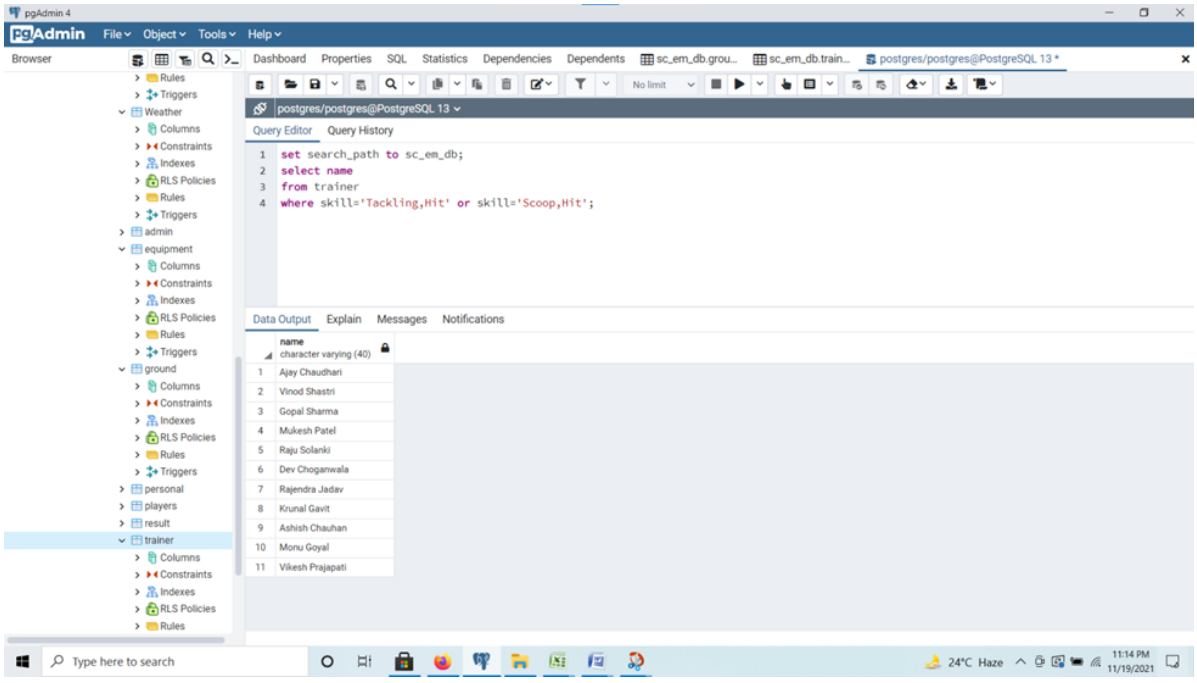
4.List the name of trainers having skill Dribble and Hit in hockey .

```
set search_path to sc_em_db;

select name

from trainer

where skill='Dribble,Hit';
```



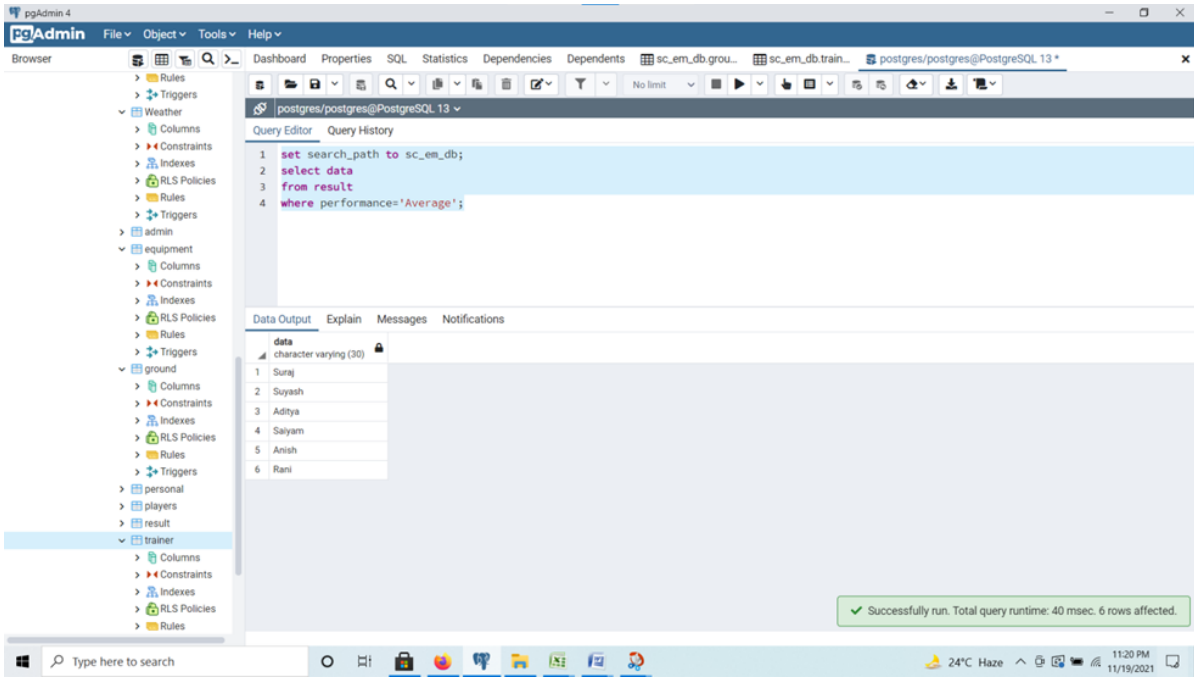
5.List the name of trainers having skill of Tackling,Hit or Scoop,Hit .

```
set search_path to sc_em_db;

 select name

 from trainer

 where skill='Tackling,Hit' or skill='Scoop,Hit';
```



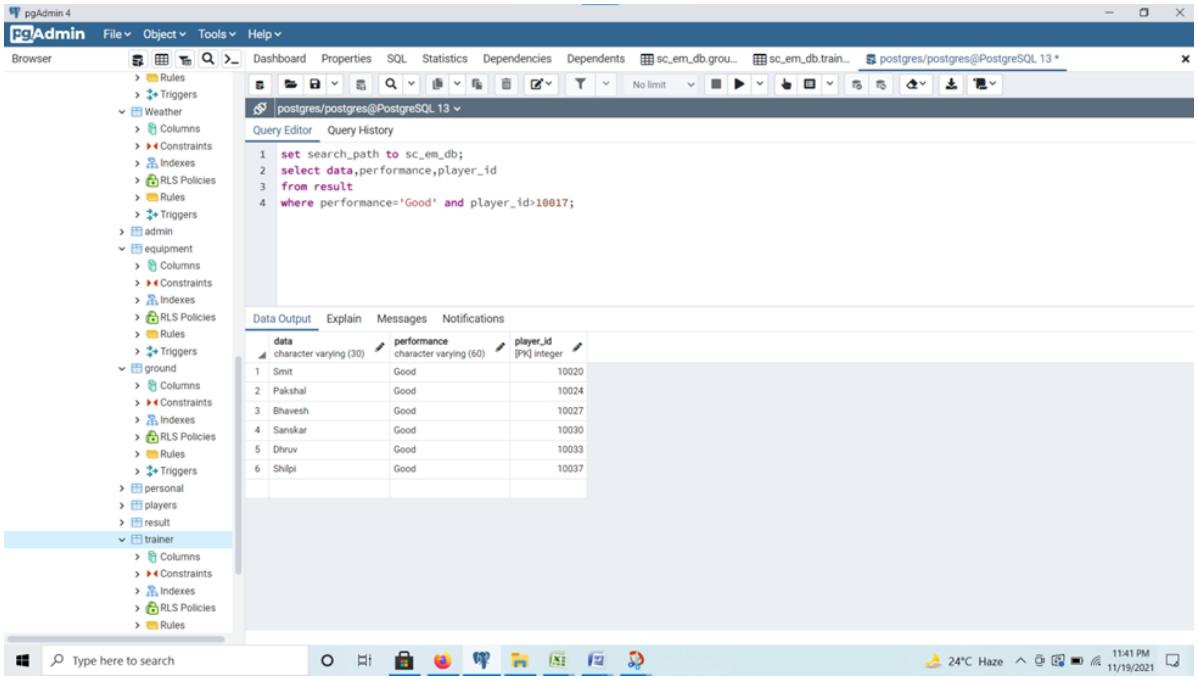
6.List the names of players having Average performance in hockey training.

```
set search_path to sc_em_db;

 select data

 from result

 where performance='Average';
```



7.Retrieve the data of players having good performance and player id is greater than 10017 in hockey training.

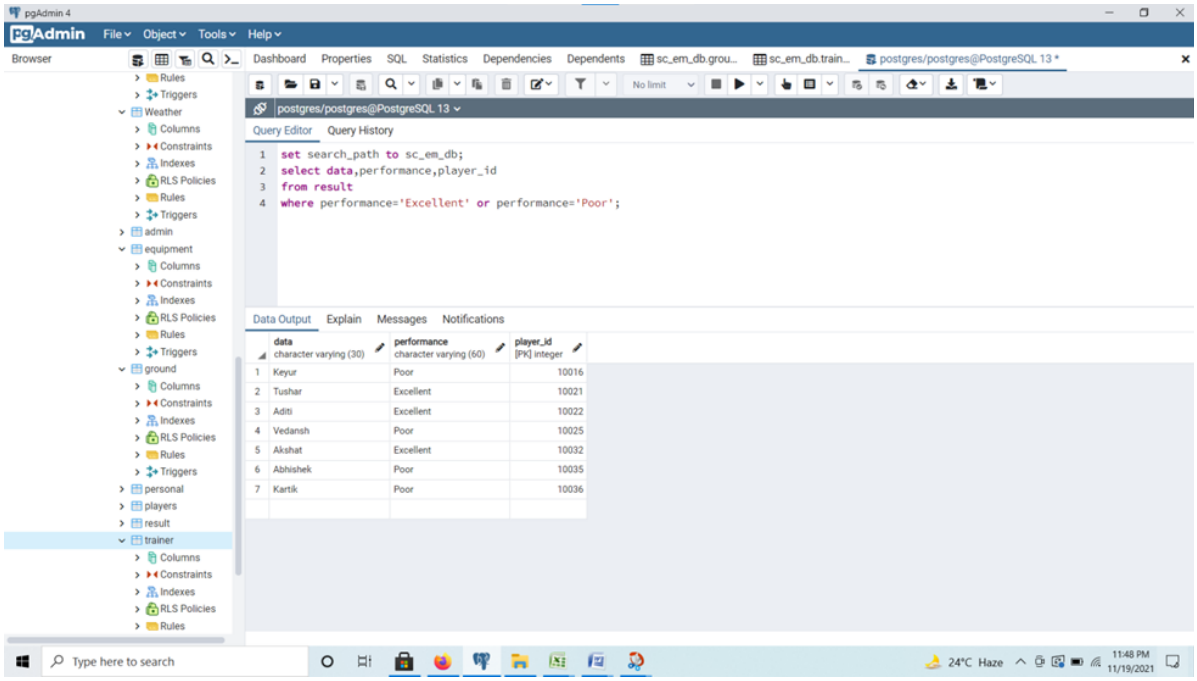
```
set search_path to sc_em_db;

select data,performance,player_id

from result

where performance='Good' and player_id>10017;
```



8.Retrieve the data of players having Excellent or Poor performance in hockey training.

```
set search_path to sc_em_db;

 select data,performance,player_id

 from result
```

where performance='Excellent' or performance='Poor';



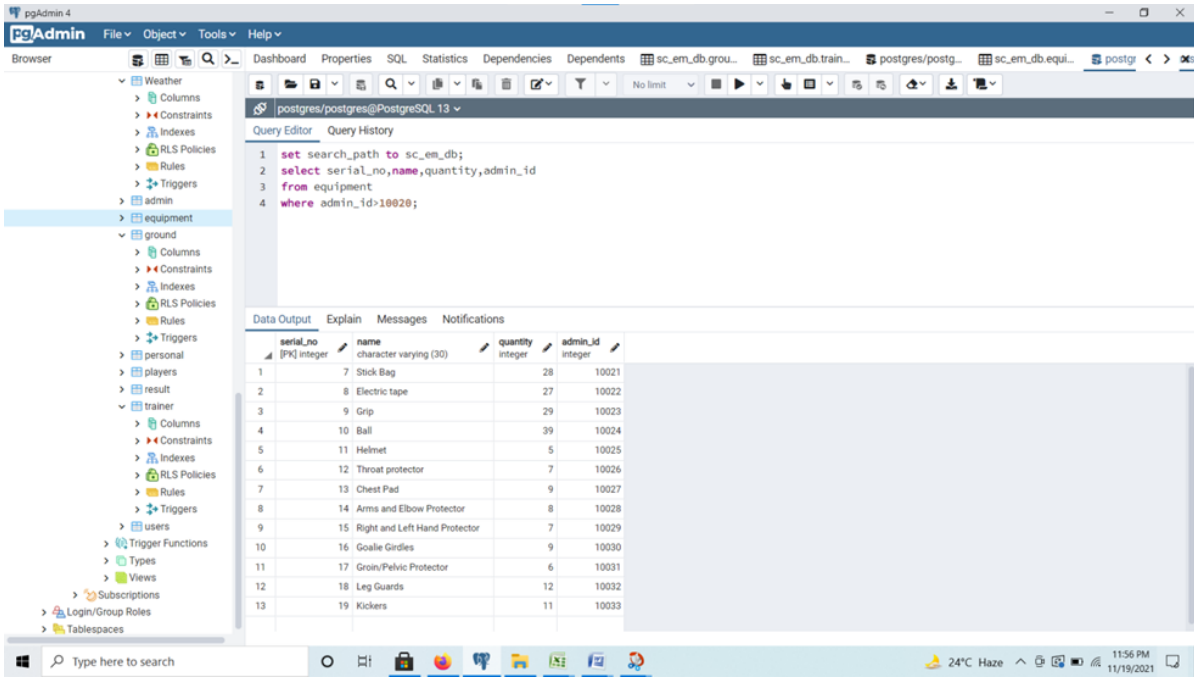
9.Retrieve the data of equipments with adminid greater than 10019.

set search_path to sc_em_db;

select serial_no,name,quantity,admin_id

from equipment

where admin_id>10020;



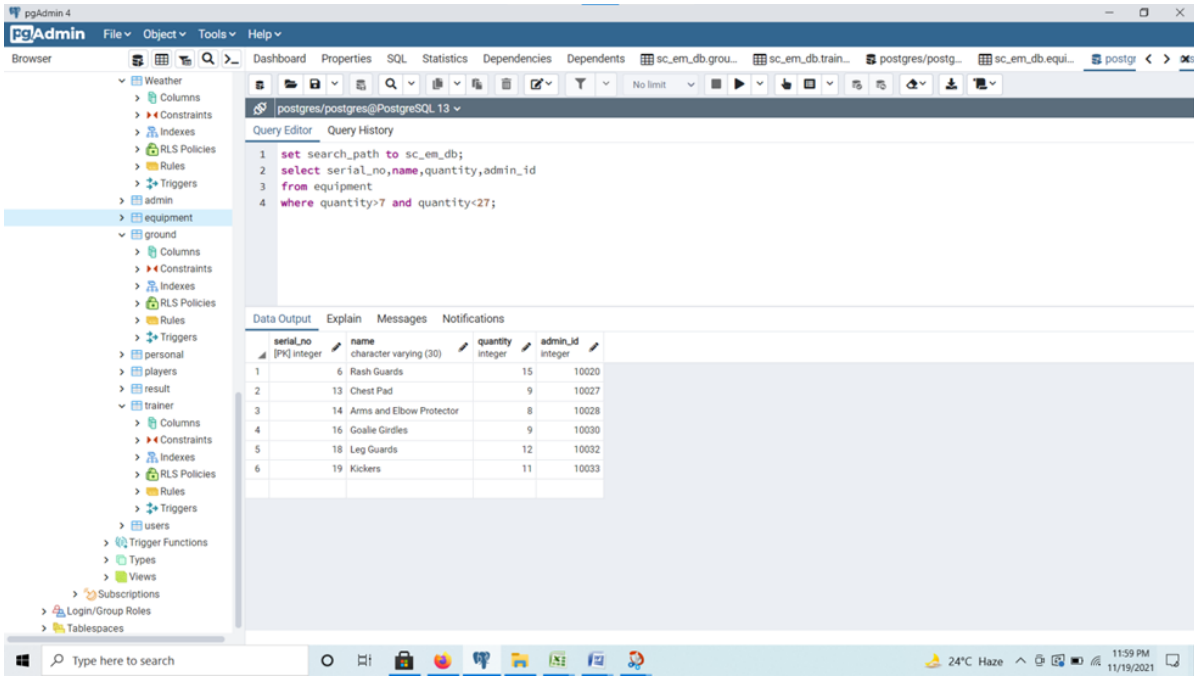
10.Retrieve the data of equipment whose quantity is greater than seven or less than twenty-seven.

set search_path to sc_em_db;

select serial_no,name,quantity,admin_id

from equipment

where quantity>7 and quantity

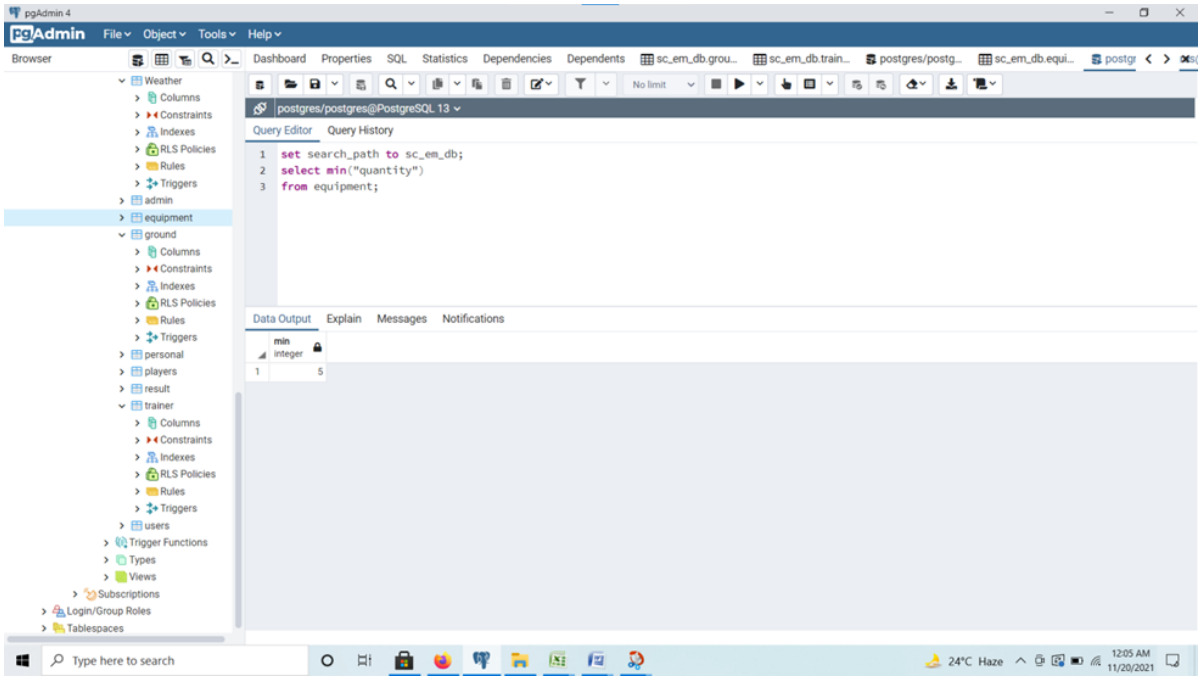


11.Find minimum quantity of equipment .

set search_path to sc_em_db;

select min("quantity")

from equipment;

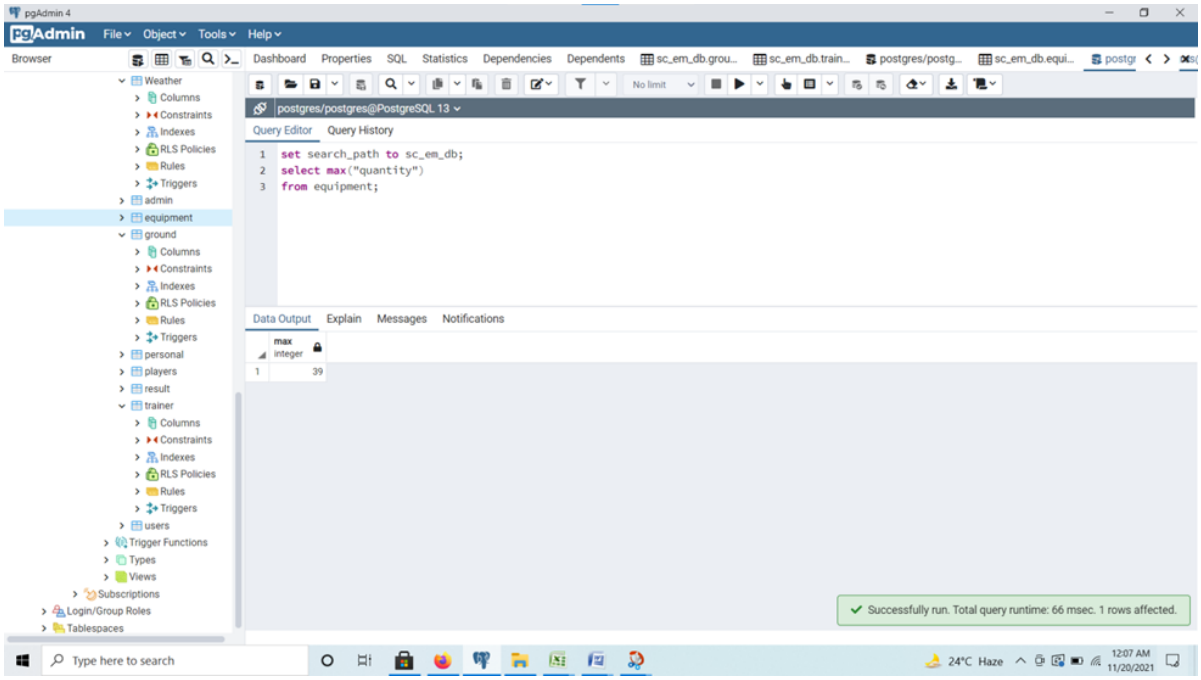


12.Find maximum quantity of equipment.

set search_path to sc_em_db;

select max("quantity")

from equipment;



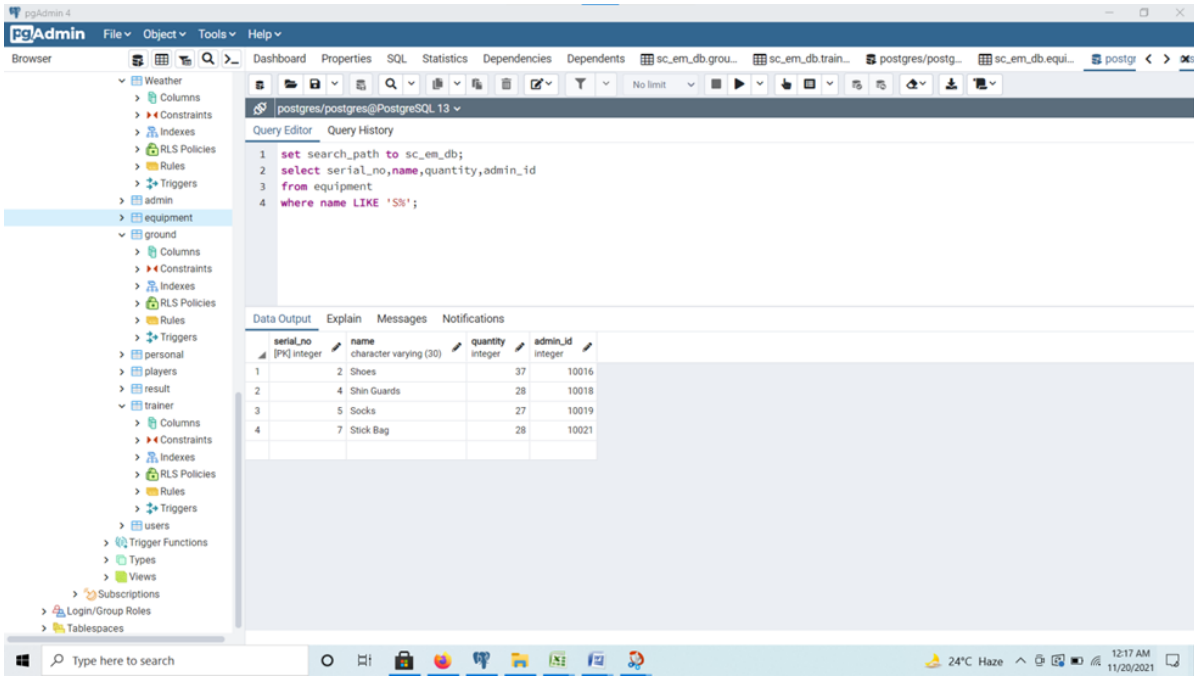
13.Retrieve the data of equipment where the name of quantity begins with ‘S’.

set search_path to sc_em_db;

select serial_no,name,quantity,admin_id

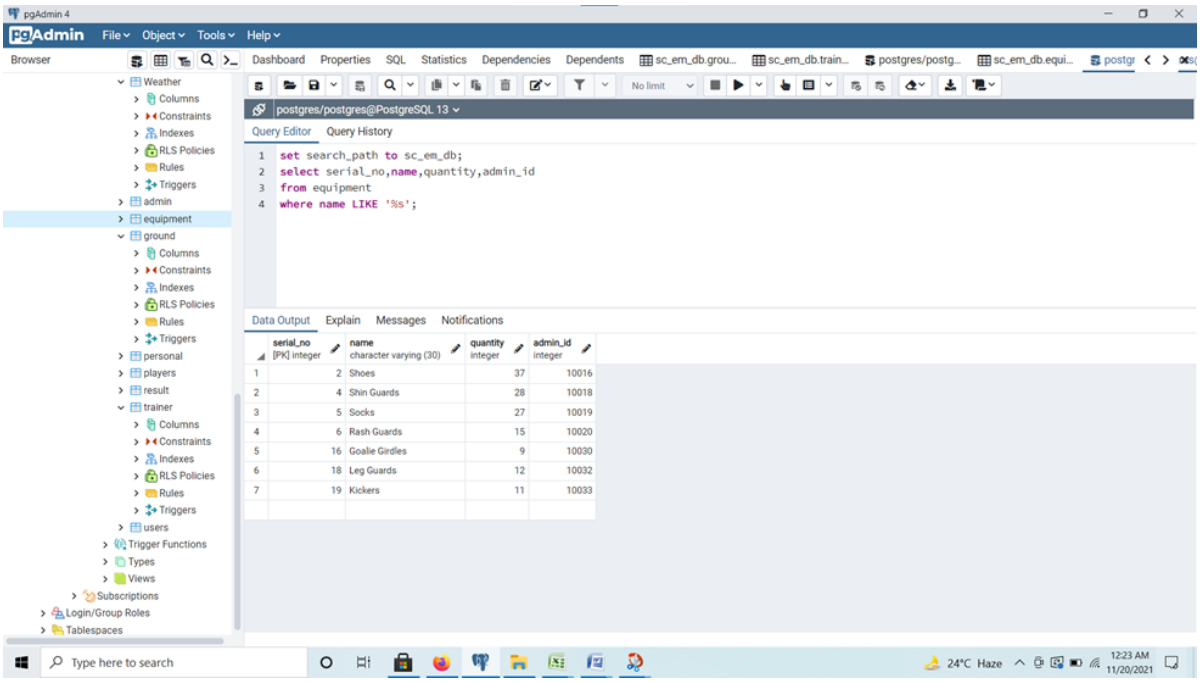
from equipment

where name LIKE 'S%';


pgAdmin 4
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
postgres/postgres@PostgreSQL 13
Query Editor
Query History
set search_path to sc_em_db;
select serial_no,name,quantity,admin_id
from equipment
where name LIKE 'S%';
Data Output
Explain
Messages
Notifications
serial_no [PK] integer
name character varying (30)
quantity integer
admin_id integer
1 2 Shoes 37 10016
2 4 Shin Guards 28 10018
3 5 Socks 27 10019
4 7 Stick Bag 28 10021
Weather
Columns
Constraints
Indexes
RLS Policies
Rules
Triggers
admin
equipment
ground
personal
players
result
trainer
users
Trigger Functions
Types
Views
Subscriptions
Login/Group Roles
Tablespaces
Type here to search
24°C Haze
12:17 AM
11/20/2021

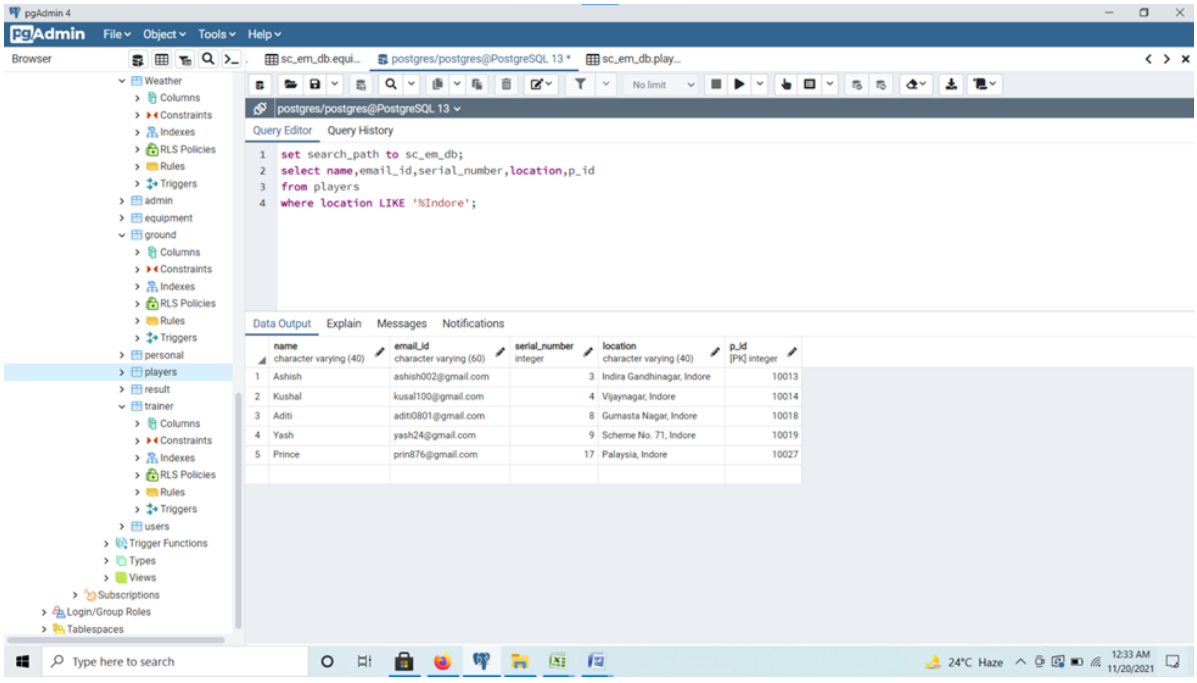
14. Retrieve the data of equipment where the name of quantity ends with ‘s’.

```
set search_path to sc_em_db;

select serial_no,name,quantity,admin_id

from equipment

where name LIKE '%s';
```



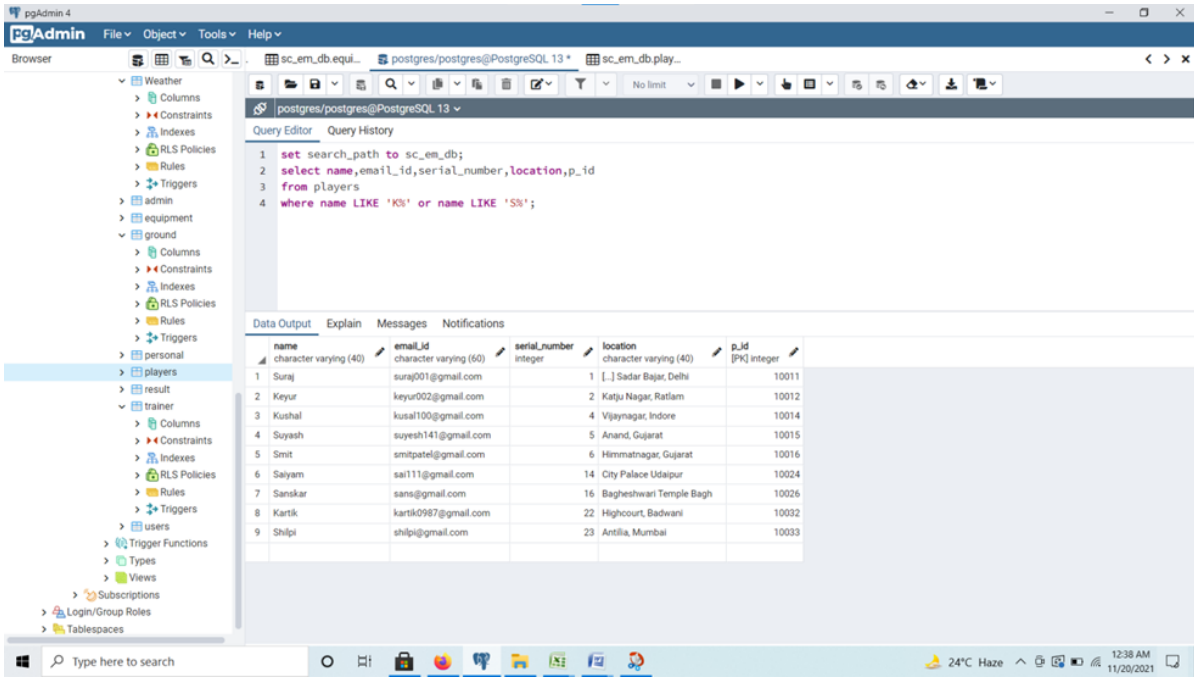
15.Retrieve the data of players undergoing training in Indore.

```
set search_path to sc_em_db;

select name,email_id,serial_number,location,p_id

from players

where location LIKE '%Indore';
```



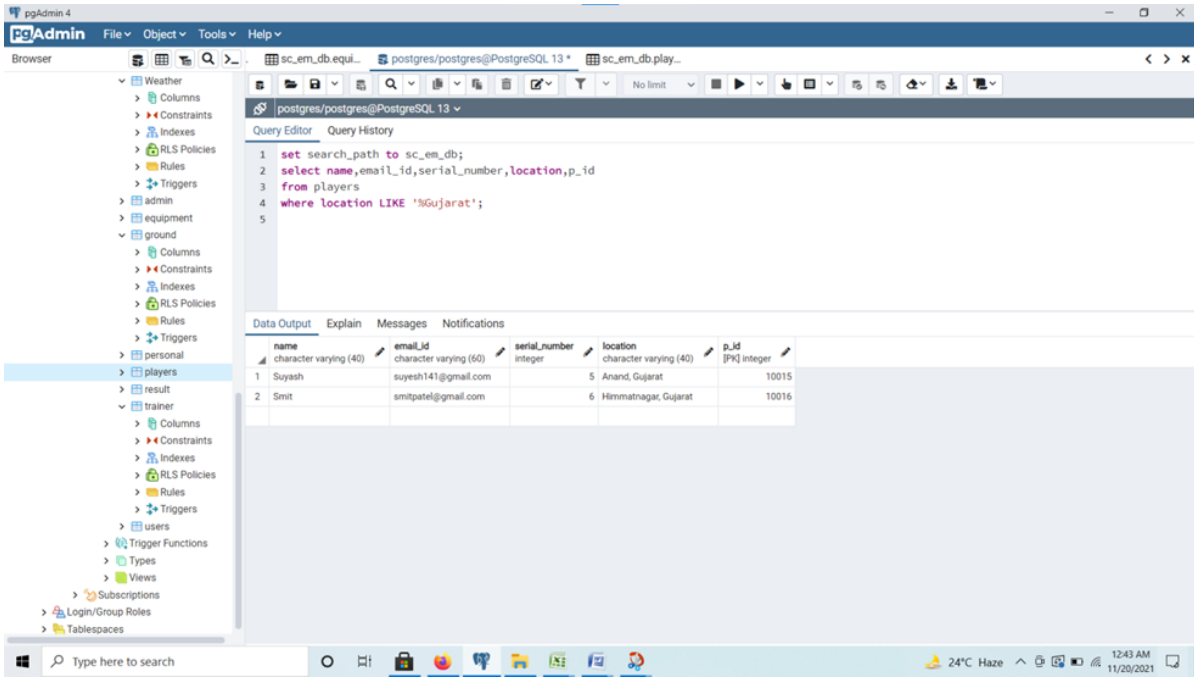
16.Retrieve the data of players whose name begins with ‘k’ or ‘S’.

```
set search_path to sc_em_db;

select name,email_id,serial_number,location,p_id

from players

where name LIKE 'K%' or name LIKE 'S%';
```



17. Retrieve the data of players undergoing training in Gujarat.

```
set search_path to sc_em_db;

select name,email_id,serial_number,location,p_id

from players

where location LIKE '%Gujarat
```



18. Retrieve the data of players with p_id>10013 and p_id<10027.

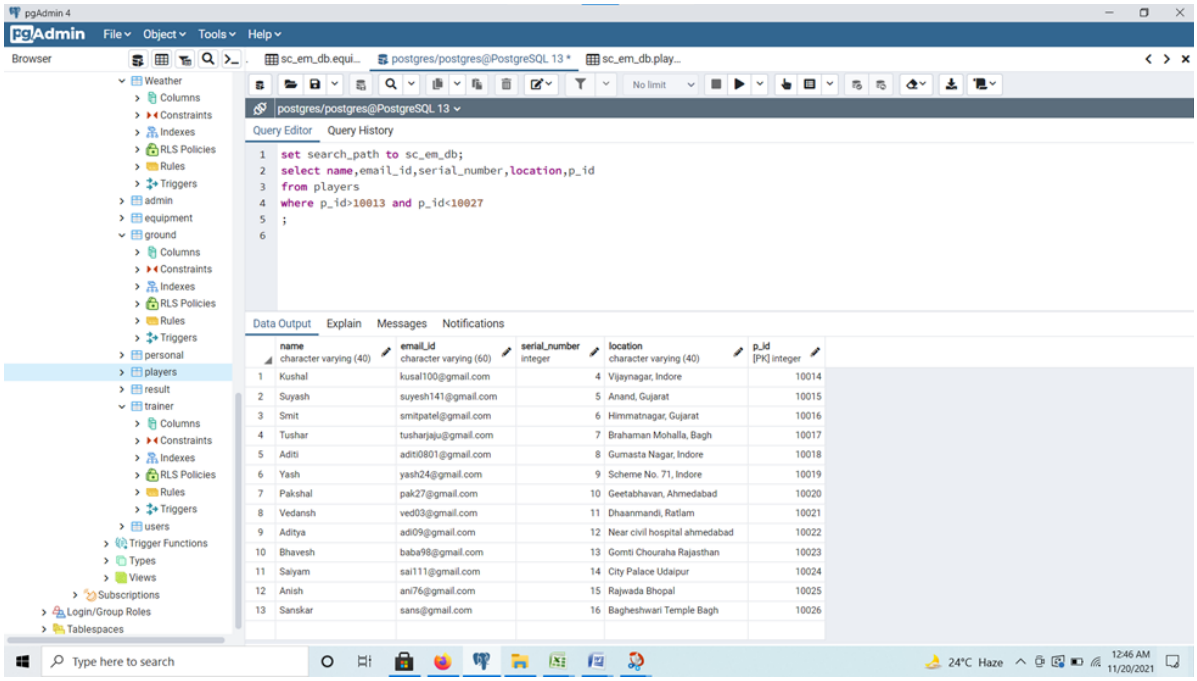
set search_path to sc_em_db;

select name,email_id,serial_number,location,p_id

from players

where p_id>10013 and p_id<10027

;



19.Update name Suyash to Shreyansh in player table.

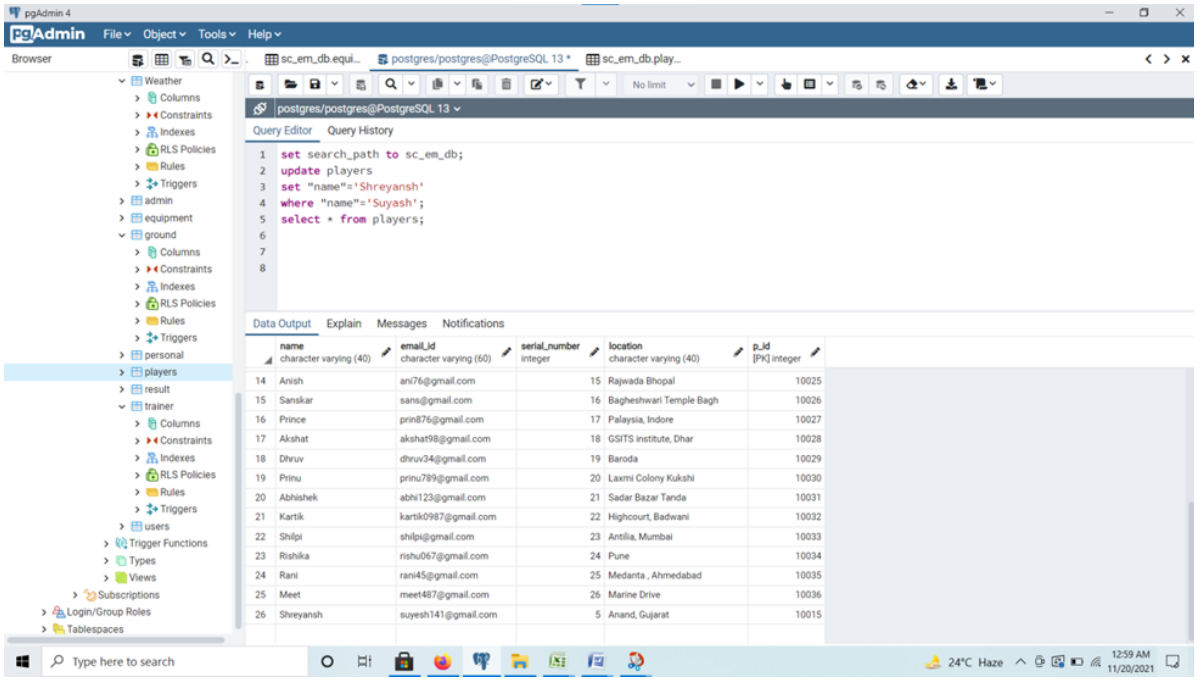
set search_path to sc_em_db;

update players

set "name"='Shreyansh'

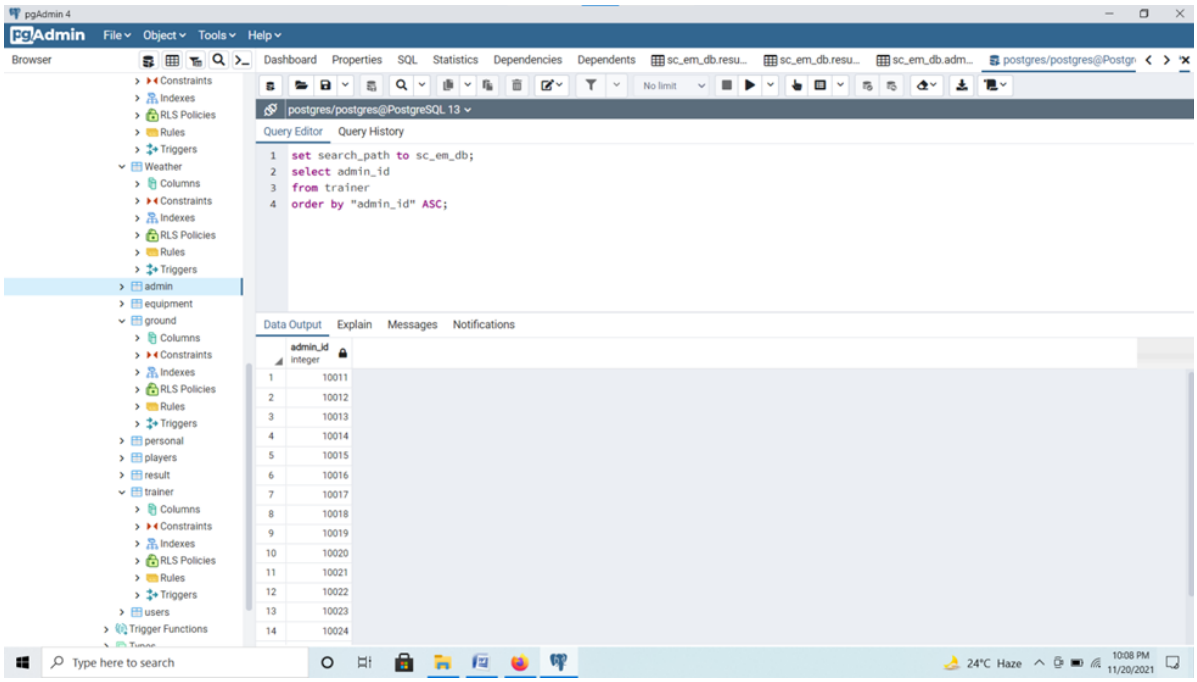
where "name"='Suyash';

select * from players;



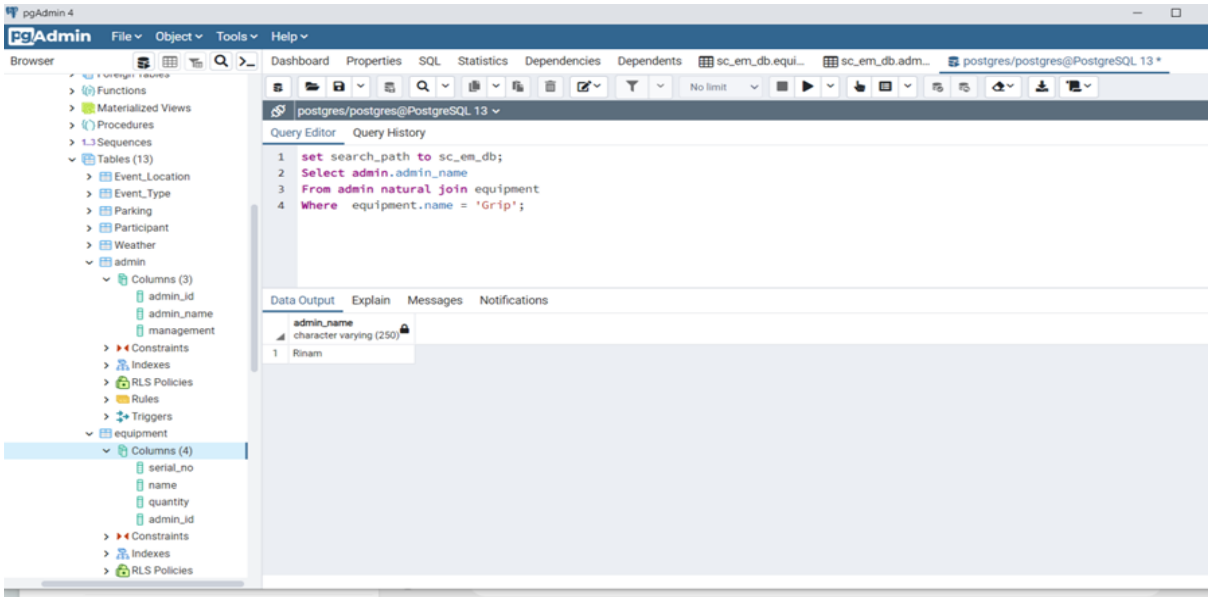
20.Arrange the admin_id of the trainer table in ascending order.

```
set search_path to sc_em_db;

select admin_id

from trainer

order by "admin_id" ASC;
```



24. write query to display admin name where equipment name is Grip

```
set search_path to sc_em_db;

Select admin.admin_name

From admin natural join equipment

Where  equipment.name = 'Grip';
```

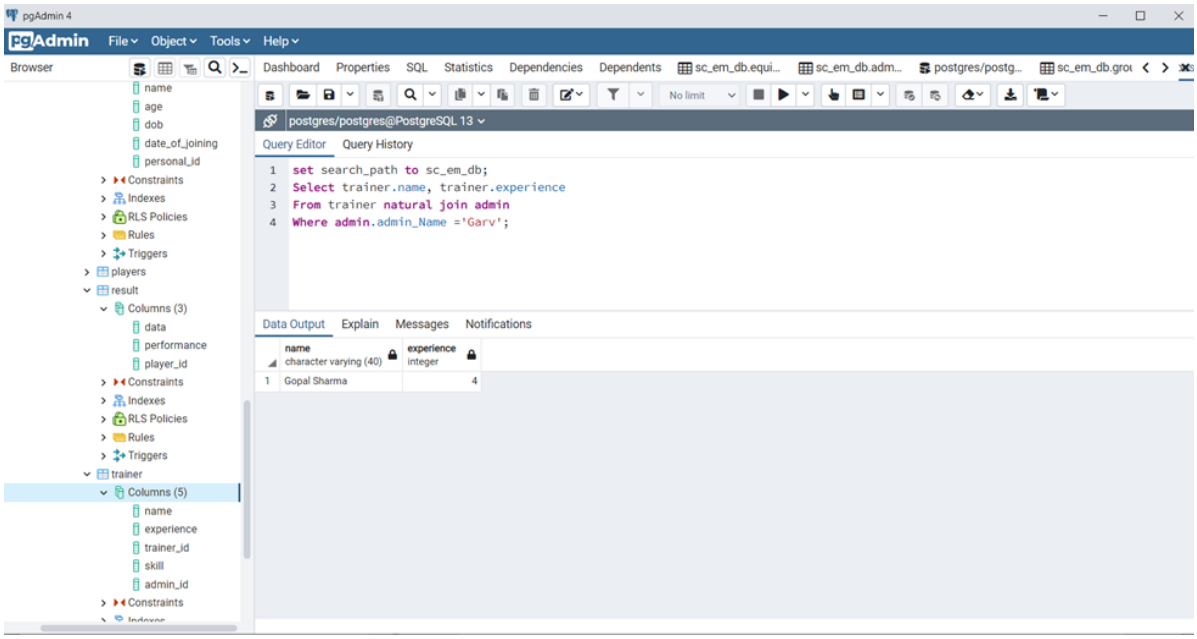


25. Write query to find trainer name and experience whose admin name is Garv.

```
Select Trainer.name, trainer.experience
```

From trainer natural join “sc_em_db”.admin

Where admin.admin_Name = “Garv’;

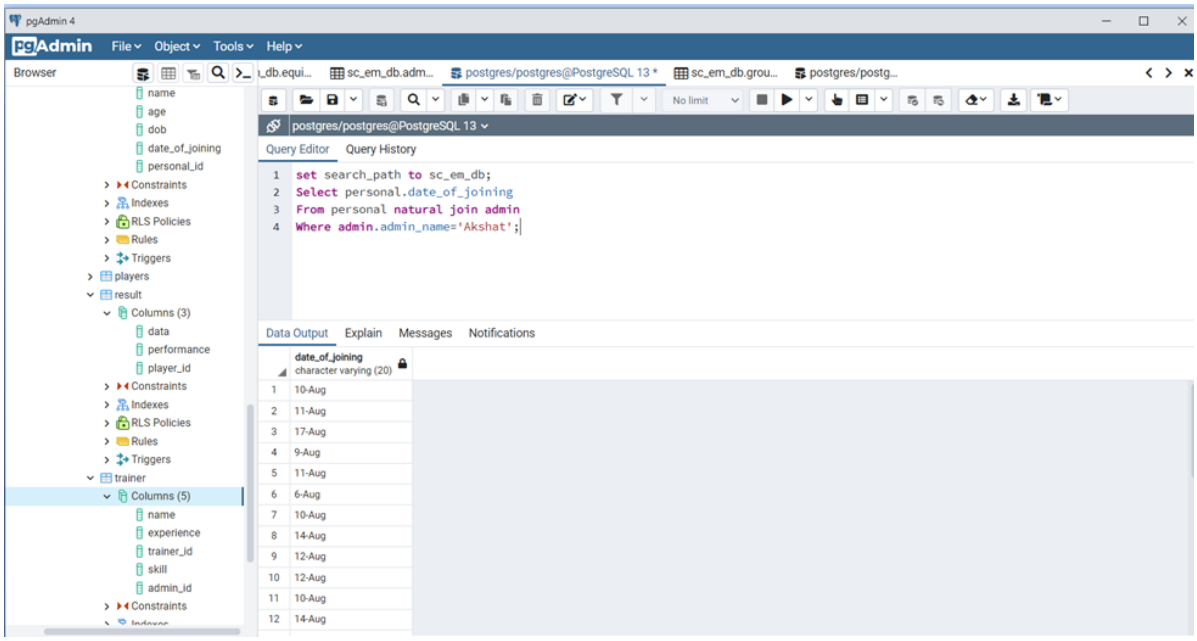


26. write query to display date of joining of player whose name is Akshat.

Select personal.date_of_joining

From personal natural join admin

Where admin.admin_name=’Akshat’;



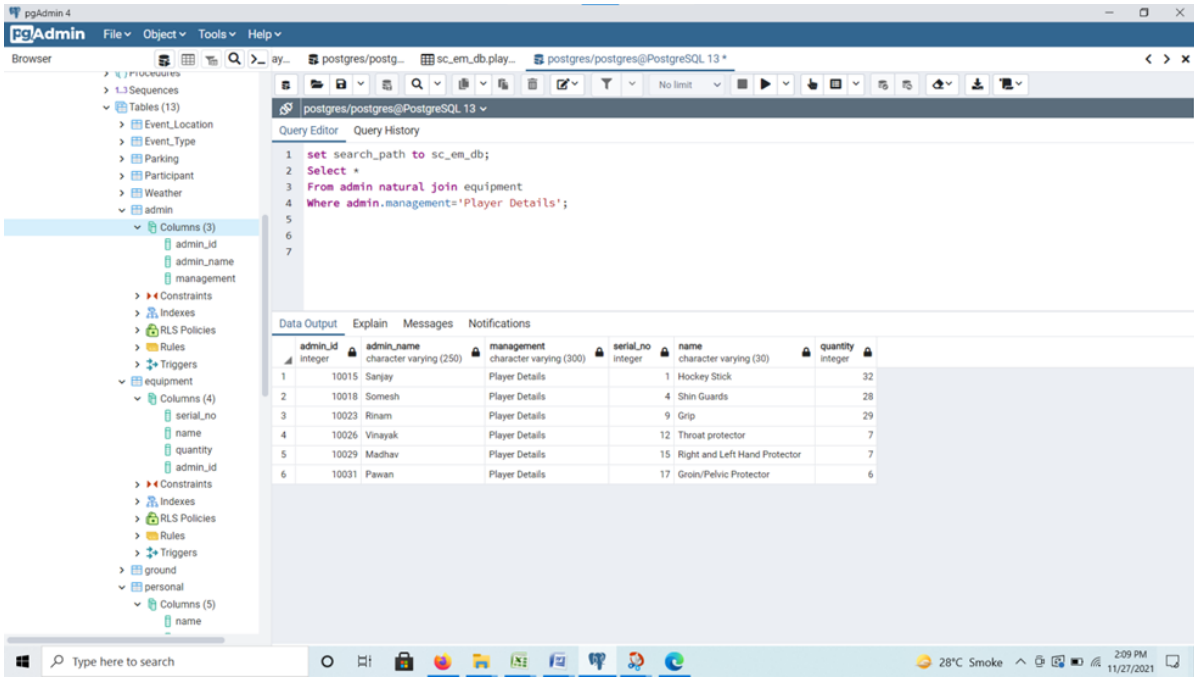
27. write a query to display all details where admin’s management is Player Details.

Set search_path to sc_em_db;

Select *

From admin natural join equipment.

Where management =’Player Details’;

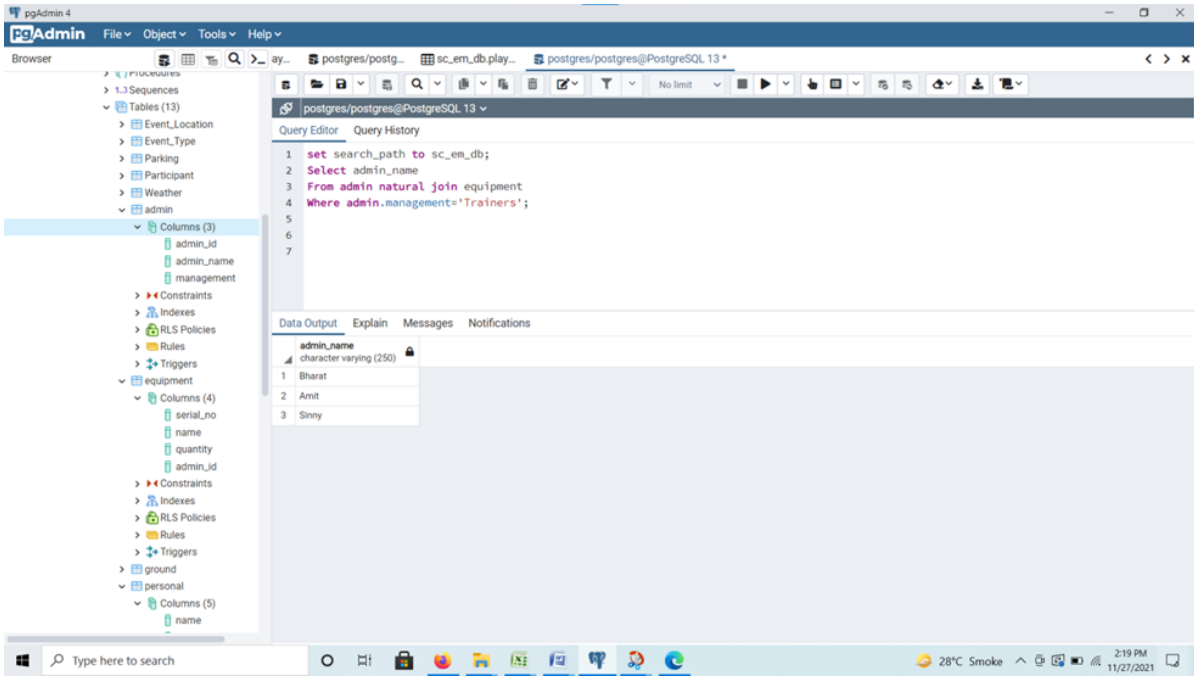


26. write a query to display admin_name whose management is Trainers.

Select admin_name

From admin natural join equipment

Where management=’Trainers’;

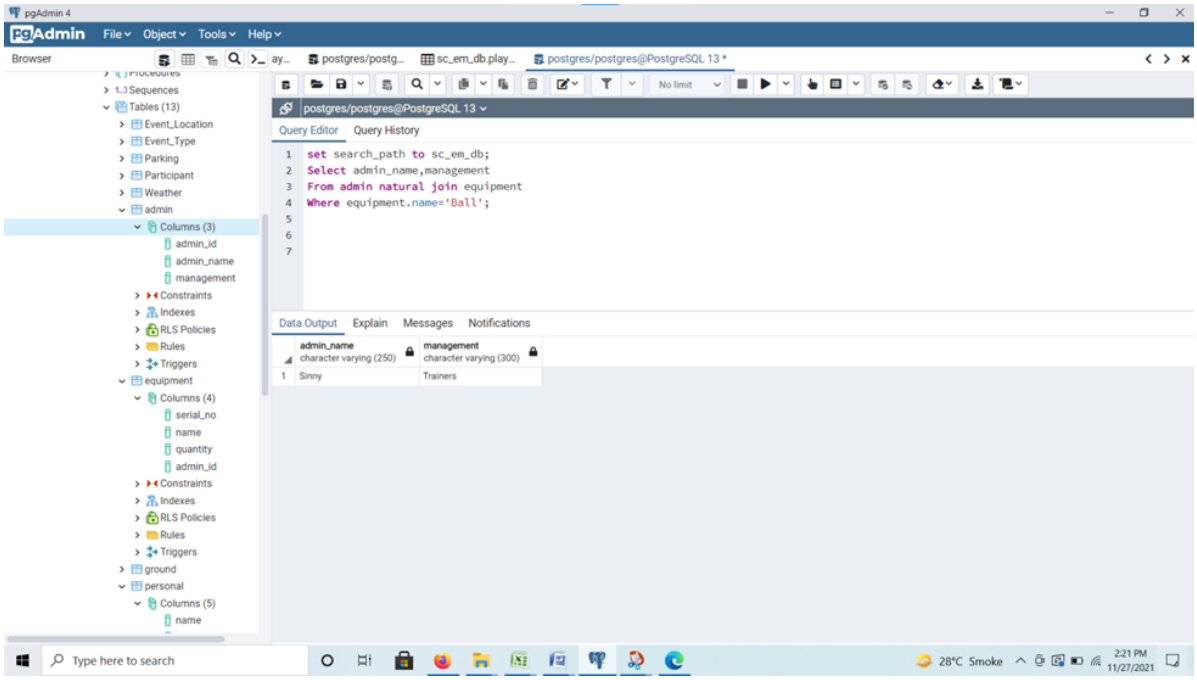


27. write query to display to display admin_name,management where name of equipment is Ball

Select admin_name,management

From admin natural join equipment

Where equipment.name=’Ball’;

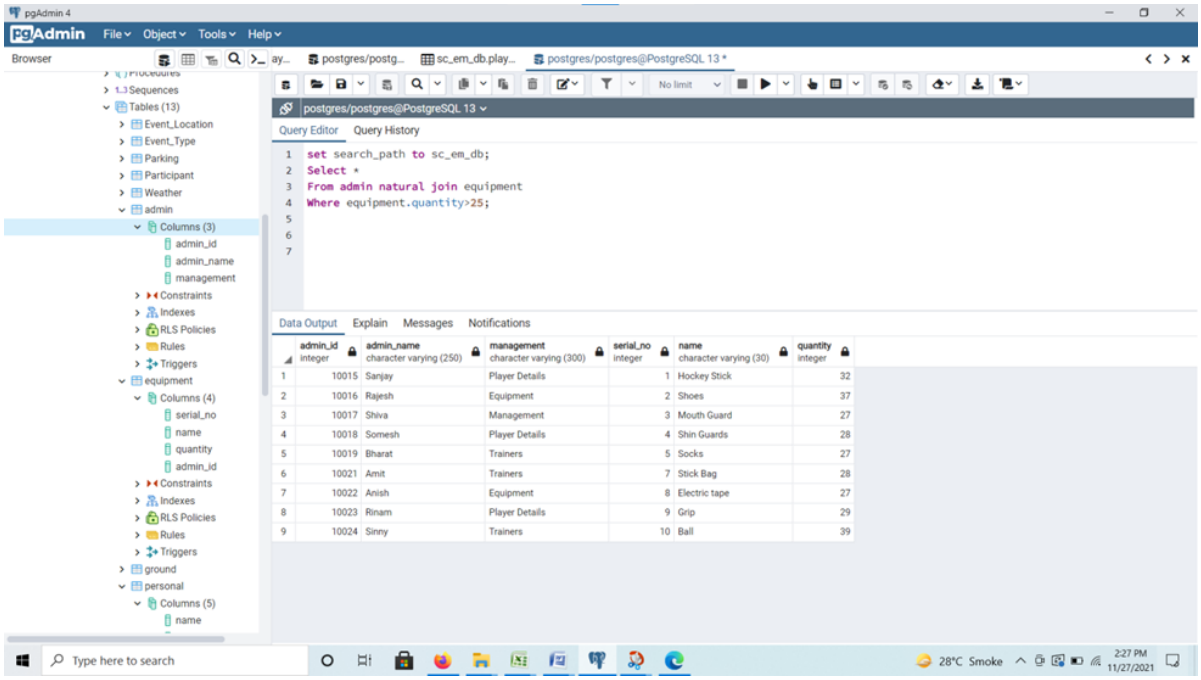


28. write query to display all details where quantity of equipment is greater than 25.

Select *

From admin natural join equipment

Where equipment.quantity>25;

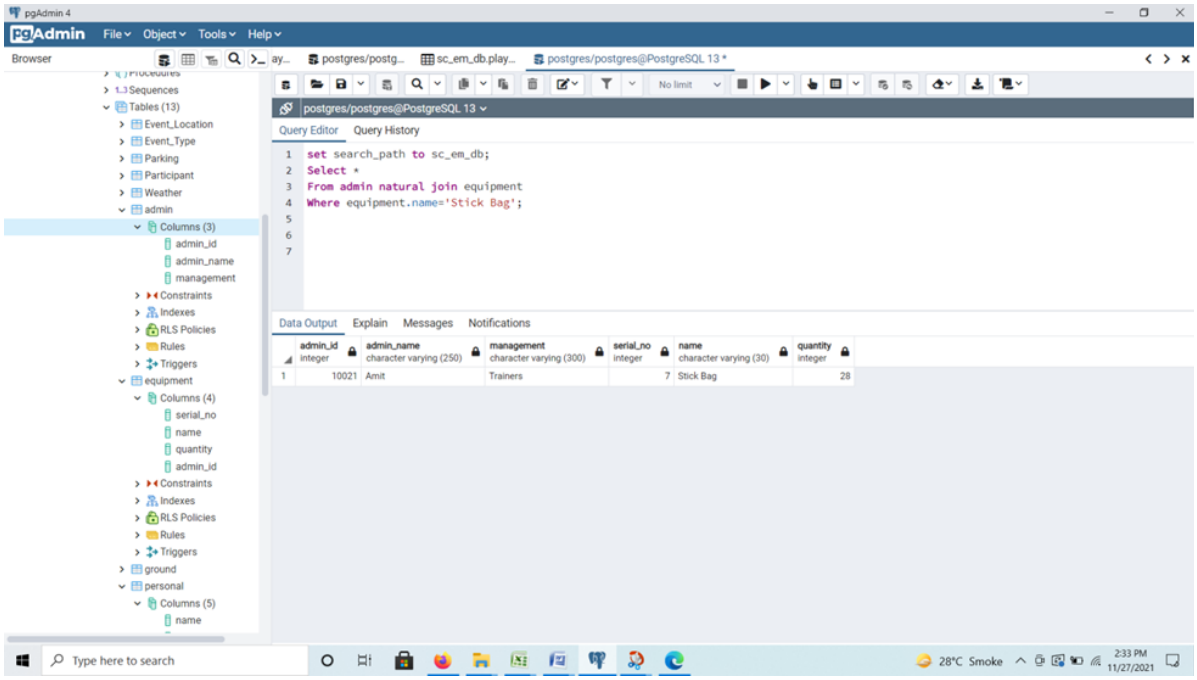


29. write query to display all details where name of equipment is Stick Bag.

Select *

From admin natural join equipment

Where equipment.name='Stick Bag';

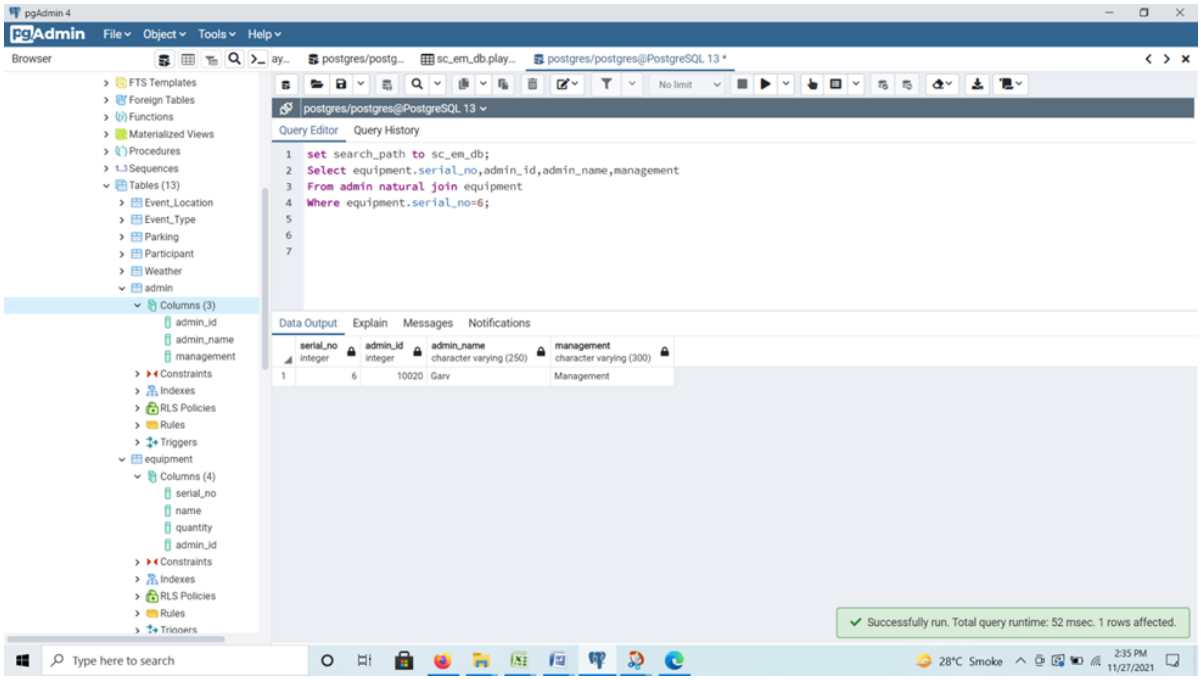


30.Write query to display serial no,admin_id,admin_name ,management where serial no is 6.

Select equipment.serial_no,admin_id,admin_name,management

From admin natural join equipment

Where equipment.serial_no=6.

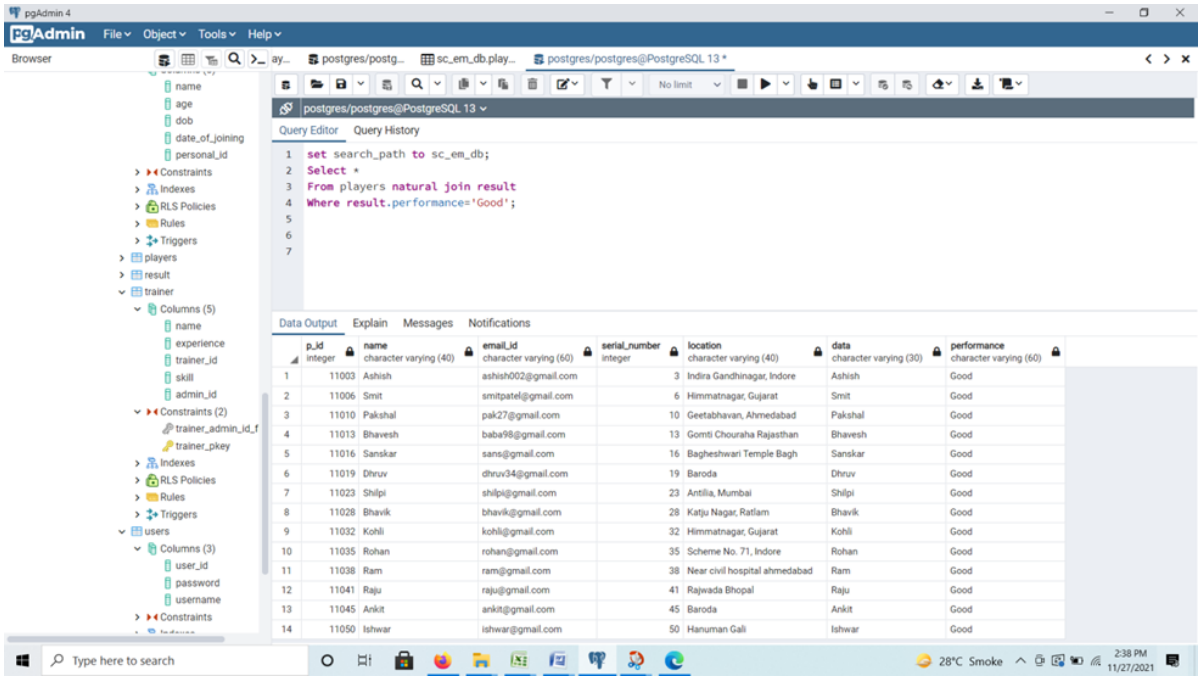


31. write query to display all details where performance of player in result is Good.

Select *

From players natural join result

Where result.performance='Good';

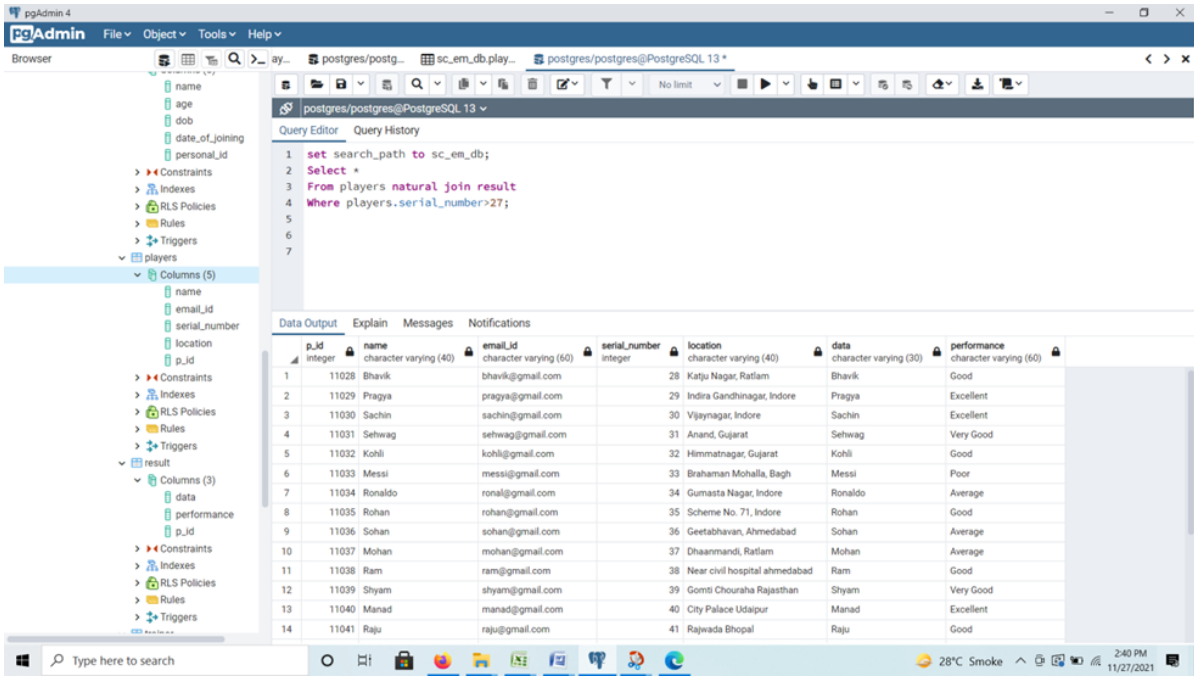


32. write query to display all details where serial number of player is greater than 27.

Select *

From players natural join result

Where players.serial_number>27;

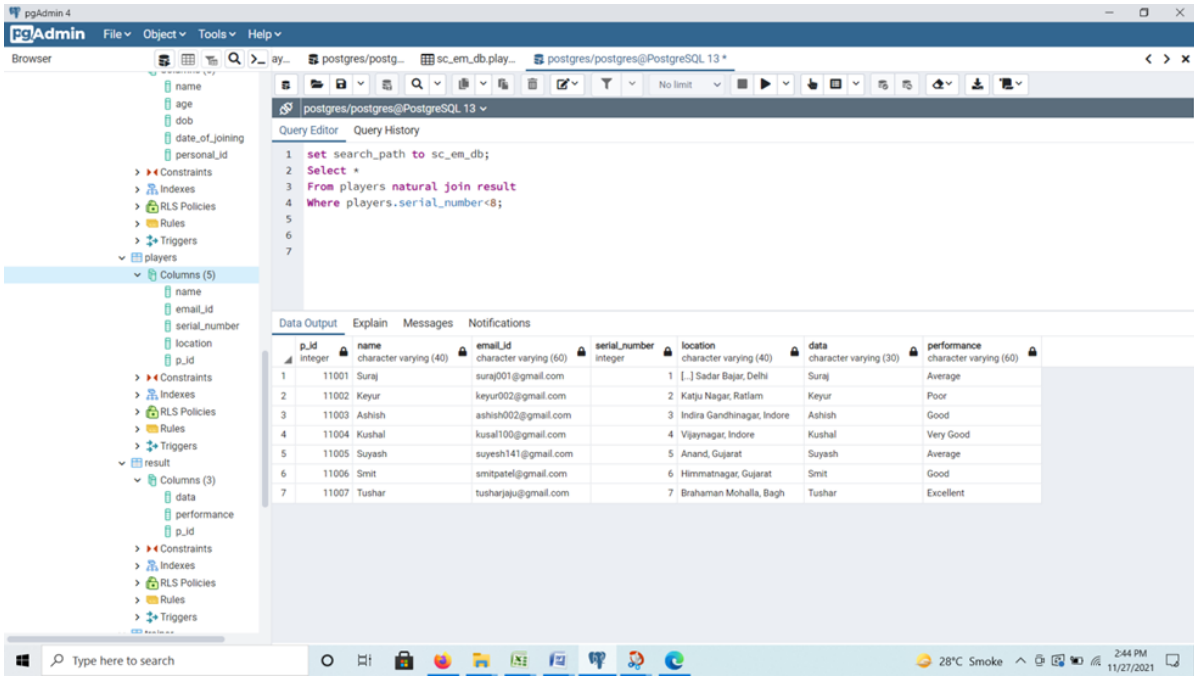


33. write query to display all details where serial number of player is less than 8.

Select *

From players natural join result

Where players.serial_number<8;

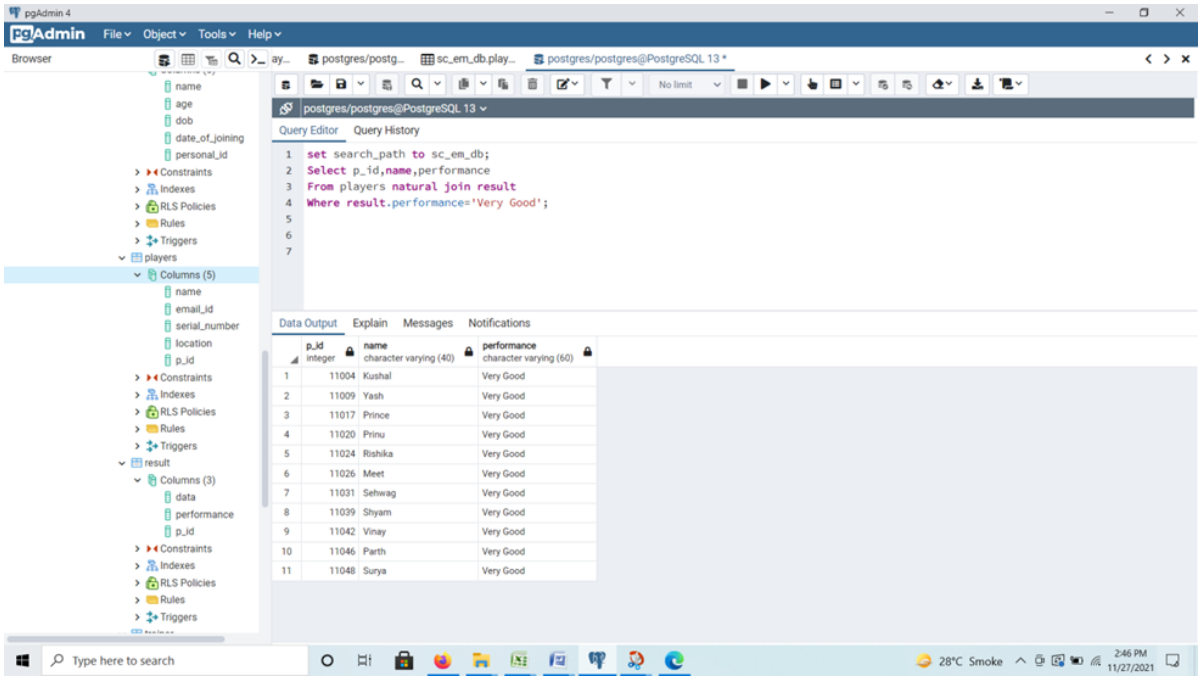


34. write query to display player_id,performance and name where performance of player is 'Very Good'.

Select p_id,name,performance

From players natural join result

Where result.performance='Very Good';

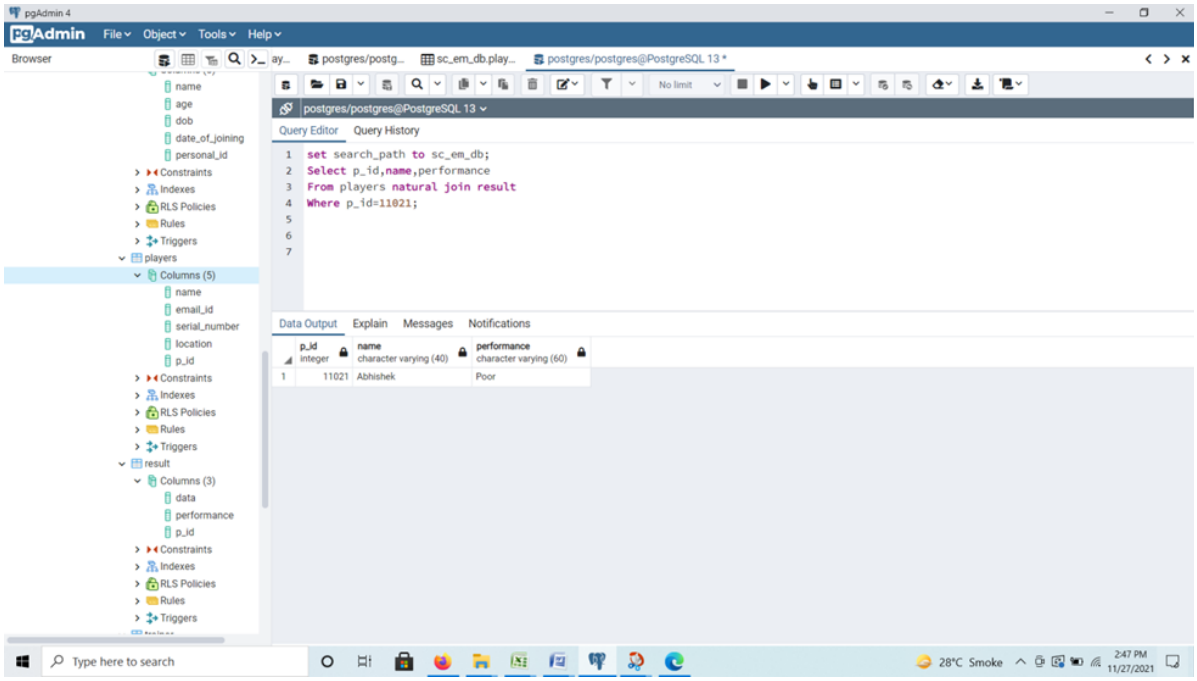


35. write query to display player id,name ,performance where player id =11021.

Select p_id,name,performance

From players natural join result

Where p_id=11021;

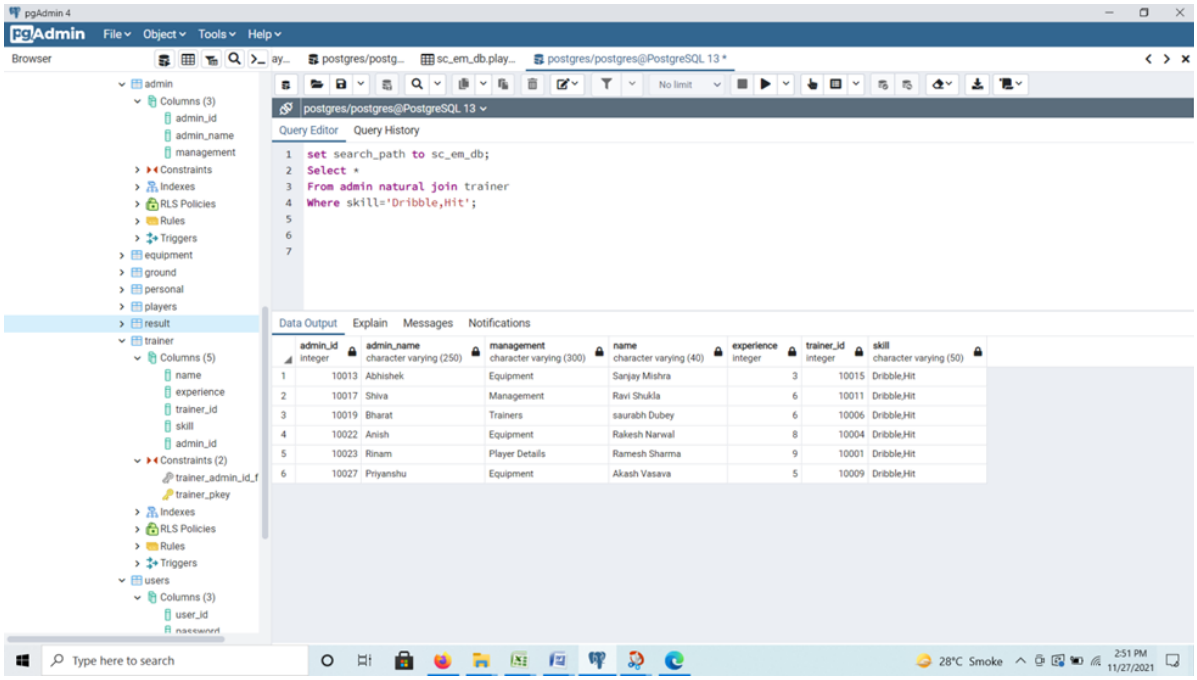


36. write query to display all details where skill of trainer is ‘Dribble,Hit’.

Select *

From admin natural join trainer

Where skill=’Dribble,Hit’;

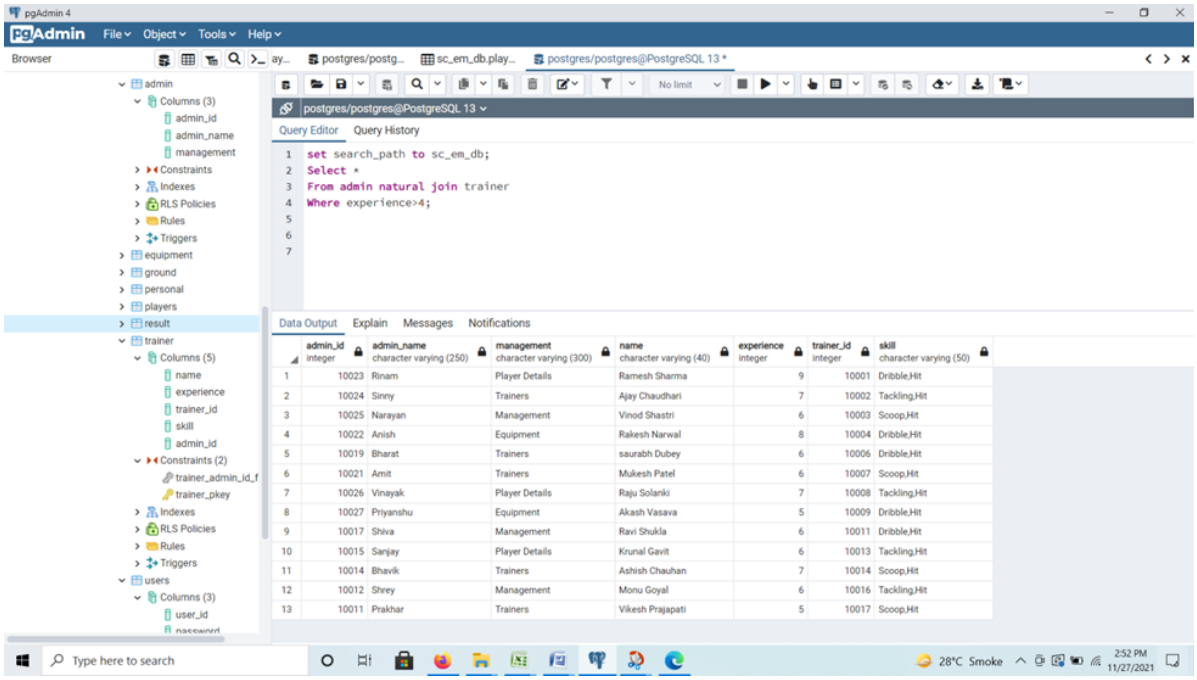


37. write query to display all details where experience of trainer is greater than 4.

Select *

From admin natural join trainer

Where experience>4;

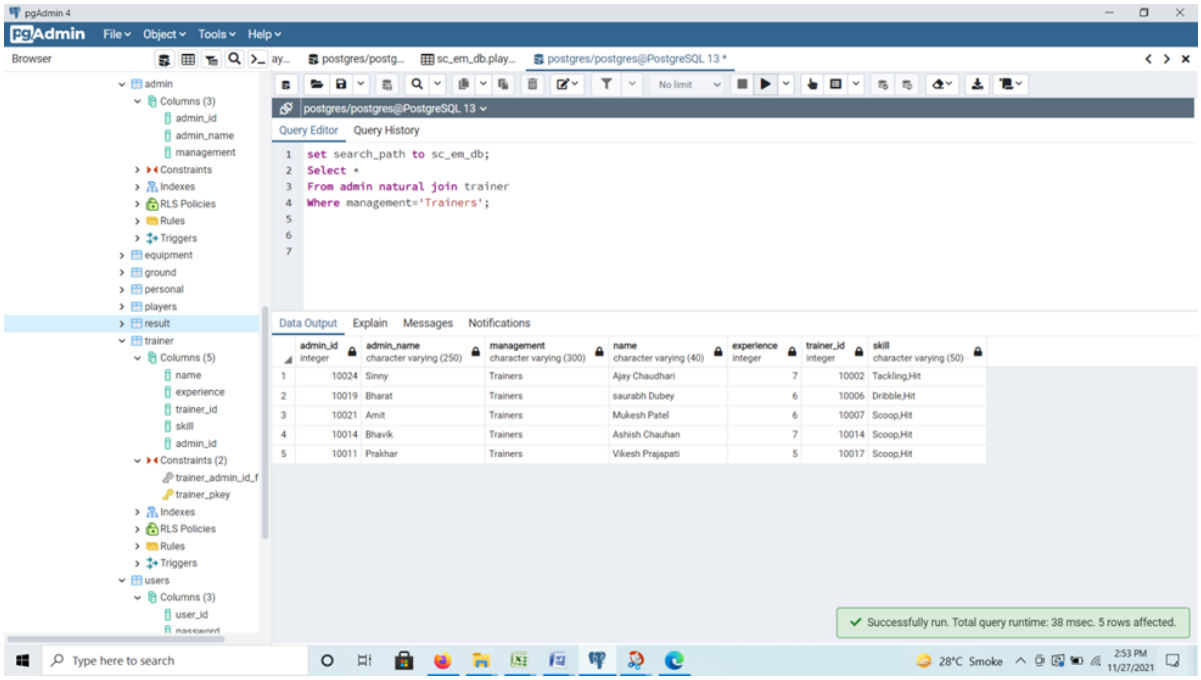


38. write query to display all details where management is Trainers.

Select *

From admin natural join trainer

Where management=’Trainers’;

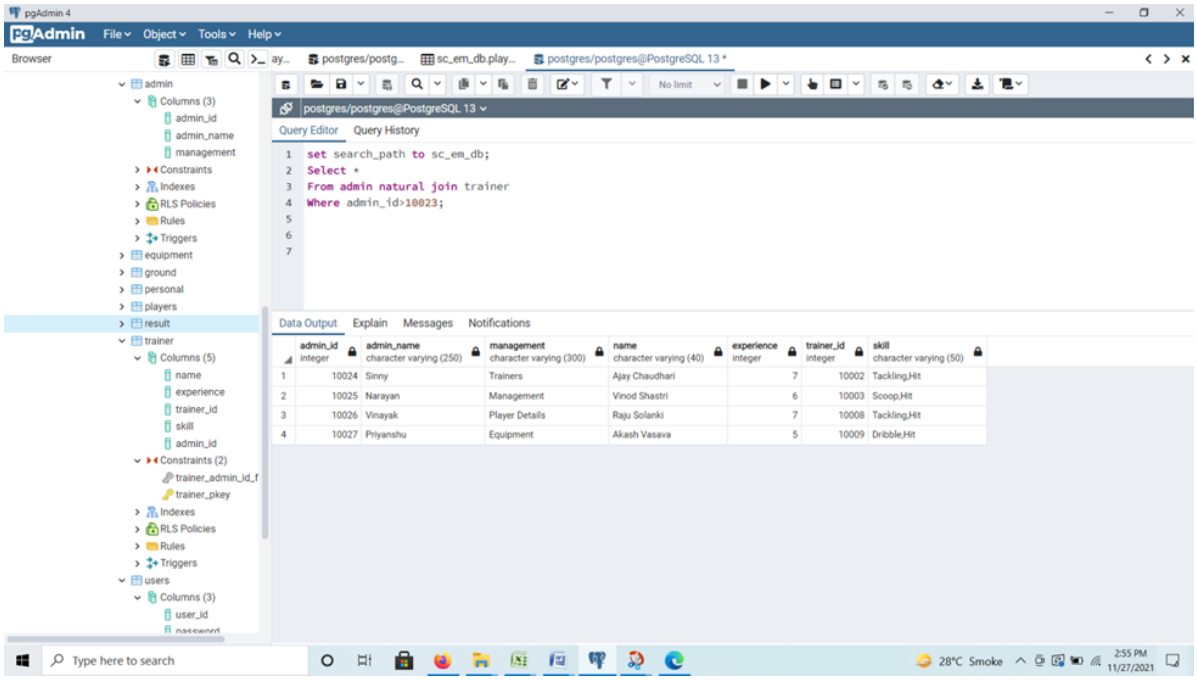


39. write query to display all details where admin_id is greater than 10023.

Select *

From admin natural join trainer

Where admin_id>10023.

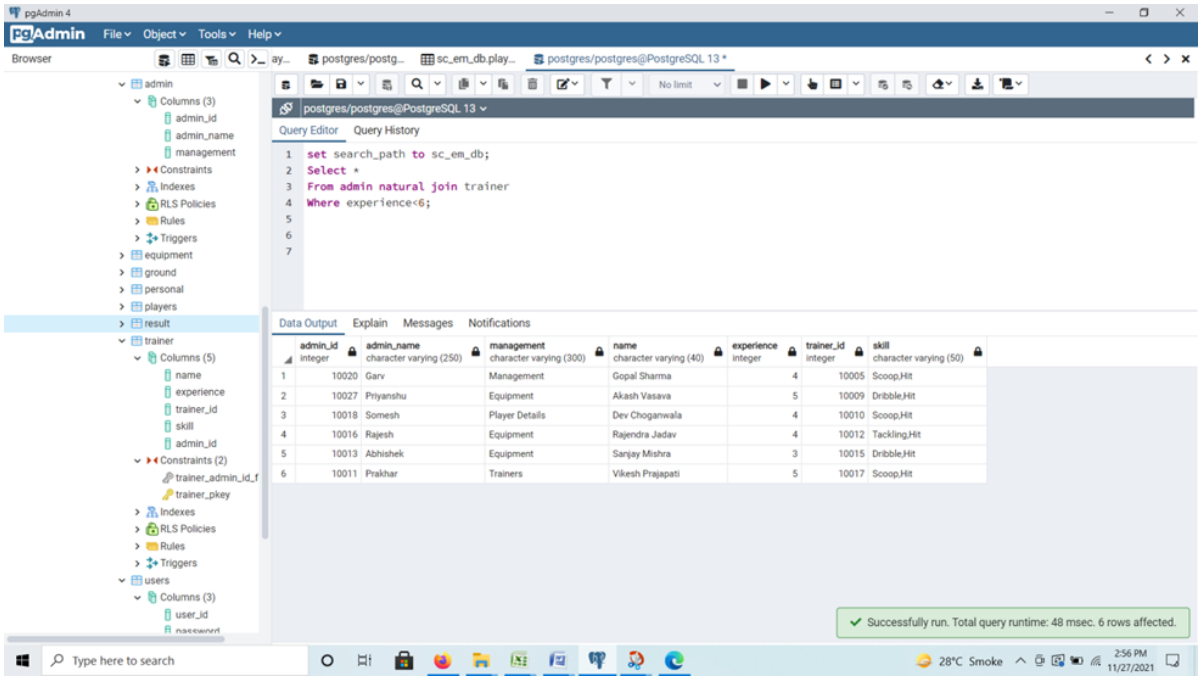


40. write query to display all details where experience of trainer is less than 6.

Select *

From admin natural join trainer

Where experience<6;


pgAdmin 4
File
Object
Tools
Help
Browser
postgres/postgres@PostgreSQL 13 *
admin
Columns (3)
admin_id
admin_name
management
Constraints
Indexes
RLS Policies
Rules
Triggers
equipment
ground
personal
players
result
trainer
Columns (5)
name
experience
trainer_id
skill
admin_id
Constraints (2)
trainer_admin_id_f
trainer_pkey
Indexes
RLS Policies
Rules
Triggers
users
Columns (3)
user_id
Query Editor
Query History
1 set search_path to sc_em_db;
2 Select *
3 From admin natural join trainer
4 Where experience<6;
5
6
7
Data Output
Explain
Messages
Notifications
admin_id integer
admin_name character varying (250)
management character varying (300)
name character varying (40)
experience integer
trainer_id integer
skill character varying (50)
1 10020 Garv Management Gopal Sharma 4 10005 Scoop,Hit
2 10027 Priyanshu Equipment Akash Vasava 5 10009 Dribble,Hit
3 10018 Somesh Player Details Dev Choganwala 4 10010 Scoop,Hit
4 10016 Rajesh Equipment Rajendra Jadav 4 10012 Tackling,Hit
5 10013 Abhishek Equipment Sanjay Mishra 3 10015 Dribble,Hit
6 10011 Prakhar Trainers Vikesh Prajapati 5 10017 Scoop,Hit
Successfully run. Total query runtime: 48 msec. 6 rows affected.
Type here to search
28°C Smoke
2:56 PM
11/27/2021

Section7: Project Code with output screenshots

1.CODE FOR CONNECTING DATABASE(PLAYERS AND TRAINERS)

(A)

```
<?php include("db.php"); ?>


<?php include('includes/header.php'); ?>


<main class="container p-4">
  <div class="row">
    <div class="col-md-4">
      <!-- MESSAGES -->


      <?php if (isset($_SESSION['message'])) { ?>
      <div class="alert alert-<?= $_SESSION['message_type']?> alert-dismissible fade show" role="alert">
        <?= $_SESSION['message']?>
        <button type="button" class="close" data-dismiss="alert" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <?php session_unset(); } ?>
```

```
    <!-- ADD TASK FORM -->

    <div class="card card-body">

      <form action="save_player.php" method="POST">

                            <div class="form-group">

         <input type="text" name="player_id" class="form-control" placeholder="Enter player ID" required
autofocus>

        </div>

        <div class="form-group">

         <input type="text" name="name" class="form-control" placeholder="Enter Name" required
autofocus>

        </div>

        <div class="form-group">

         <input type="text" name="age" class="form-control" placeholder="Enter Age" required autofocus>

        </div>

                            <div class="form-group">

         <input type="text" name="dob" class="form-control" placeholder="Enter DOB" required
autofocus>

        </div>

                            <div class="form-group">

         <input type="text" name="doj" class="form-control" placeholder="Enter DOJ" required autofocus>

        </div>

        <input type="submit" name="save_player" class="btn btn-success btn-block" value="Add Player">

      </form>

    </div>

  </div>

  <div class="col-md-8">

    <table class="table table-bordered">
```

```
<thead>
 <tr>
  <th>Player ID</th>
  <th>Name</th>
  <th>Age</th>
                        <th>Date of Birth</th>
                        <th>Date of Joining</th>
  <th>Action</th>
 </tr>
</thead>
<tbody>

 <?php
 $query = "SELECT * FROM players";
 $result_tasks = mysqli_query($conn, $query);

 while($row = mysqli_fetch_assoc($result_tasks)) { ?>
 <tr>
  <td><?php echo $row['player_id']; ?></td>
  <td><?php echo $row['name']; ?></td>
  <td><?php echo $row['age']; ?></td>
                        <td><?php echo $row['dob']; ?></td>
                        <td><?php echo $row['doj']; ?></td>
  <td>
   <a href="edit_player.php?id=<?php echo $row['id']?>" class="btn btn-secondary">
    <i class="fas fa-marker"></i>
```

```
            </a>
            <a href="delete_player.php?id=<?php echo $row['id']?>" class="btn btn-danger">
              <i class="far fa-trash-alt"></i>
            </a>
          </td>
        </tr>
        <?php } ?>
      </tbody>
    </table>
  </div>
 </div>
</main>

<?php include('includes/footer.php'); ?>
```

(B)

```
<?php include("db.php"); ?>

<?php include('includes/header.php'); ?>

<main class="container p-4">
 <div class="row">
  <div class="col-md-4">
   <!-- MESSAGES -->

   <?php if (isset($_SESSION['message'])) { ?>
```

```
    <div class="alert alert-<?= $_SESSION['message_type']?> alert-dismissible fade show" role="alert">
      <?= $_SESSION['message']?>
      <button type="button" class="close" data-dismiss="alert" aria-label="Close">
        <span aria-hidden="true">&times;</span>
      </button>
    </div>
    <?php session_unset(); } ?>

    <!-- ADD TASK FORM -->
    <div class="card card-body">
      <form action="save_trainer.php" method="POST">
                          <div class="form-group">
          <input type="text" name="trainer_id" class="form-control" placeholder="Enter trainer ID"
required autofocus>
        </div>
        <div class="form-group">
          <input type="text" name="name" class="form-control" placeholder="Enter Name" required
autofocus>
        </div>
        <div class="form-group">
          <input type="text" name="experience" class="form-control" placeholder="Enter experience"
required autofocus>
        </div>
                          <div class="form-group">
          <input type="text" name="skill" class="form-control" placeholder="Enter Skill" required
autofocus>
        </div>
                          <div class="form-group">
```

```
          <input type="text" name="admin_id" class="form-control" placeholder="Enter Admin ID" required autofocus>
        </div>
        <input type="submit" name="save_trainer" class="btn btn-success btn-block" value="Add Trainer">
      </form>
    </div>
  </div>
  <div class="col-md-8">
    <table class="table table-bordered">
      <thead>
        <tr>
          <th>Trainer_id ID</th>
          <th>Name</th>
          <th>Experience</th>
                                <th>Skill</th>
                                <th>Admin ID</th>
          <th>Action</th>
        </tr>
      </thead>
      <tbody>

        <?php
        $query = "SELECT * FROM trainers";
        $result_tasks = mysqli_query($conn, $query);

        while($row = mysqli_fetch_assoc($result_tasks)) { ?>
```

```
        <tr>
          <td><?php echo $row['trainer_id']; ?></td>
          <td><?php echo $row['name']; ?></td>
          <td><?php echo $row['experience']; ?></td>
                                <td><?php echo $row['skill']; ?></td>
                                <td><?php echo $row['admin_id']; ?></td>
          <td>
            <a href="edit_trainer.php?id=<?php echo $row['id']?>" class="btn btn-secondary">
              <i class="fas fa-marker"></i>
            </a>
            <a href="delete_trainer.php?id=<?php echo $row['id']?>" class="btn btn-danger">
              <i class="far fa-trash-alt"></i>
            </a>
          </td>
        </tr>
        <?php } ?>
      </tbody>
    </table>
  </div>
 </div>
</main>

<?php include('includes/footer.php'); ?>
```

2.CODE FOR INSERT PLAYERS AND TRAINER DATA

(A)

```
<?php

include('db.php');

if (isset($_POST['save_player'])) {
  $player_id = $_POST['player_id'];
  $name = $_POST['name'];
  $age = $_POST['age'];
  $dob = $_POST['dob'];
  $doj = $_POST['doj'];
  $query = "INSERT INTO players(player_id, name, age, dob, doj ) VALUES ('$player_id', '$name','$age', '$dob','$doj')";
  $result = mysqli_query($conn, $query);
  if(!$result) {
    die("Query Failed.");
  }

  $_SESSION['message'] = 'Player Saved Successfully';
  $_SESSION['message_type'] = 'success';
  header('Location: index.php');
```

```
}

?>
```

(B)

```
<?php

include('db.php');

if (isset($_POST['save_trainer'])) {
  $trainer_id = $_POST['trainer_id'];
  $name = $_POST['name'];
  $experience = $_POST['experience'];
  $skill = $_POST['skill'];
  $admin_id = $_POST['admin_id'];
  $query = "INSERT INTO trainers(trainer_id, name, experience, skill, admin_id ) VALUES ('$trainer_id', '$name','$experience', '$skill','$admin_id')";
  $result = mysqli_query($conn, $query);
  if(!$result) {
    die("Query Failed.");
  }

  $_SESSION['message'] = 'Trainer Saved Successfully';
  $_SESSION['message_type'] = 'success';
  header('Location: trainer.php');
```

```
}

?>
```

3. CODE FOR DELETE PLAYERS AND TRAINERS DATA

(A)

```
<?php

include("db.php");

if(isset($_GET['id'])) {
  $id = $_GET['id'];
  $query = "DELETE FROM players WHERE id = $id";
  $result = mysqli_query($conn, $query);
  if(!$result) {
    die("Query Failed.");
  }

  $_SESSION['message'] = 'Player Removed Successfully';
  $_SESSION['message_type'] = 'danger';
  header('Location: index.php');
}

?>
```

(B)

```
<?php
```

```
include("db.php");


if(isset($_GET['id'])) {
  $id = $_GET['id'];
  $query = "DELETE FROM trainers WHERE id = $id";
  $result = mysqli_query($conn, $query);
  if(!$result) {
    die("Query Failed.");
  }

  $_SESSION['message'] = 'Trainer Removed Successfully';
  $_SESSION['message_type'] = 'danger';
  header('Location: trainer.php');
}


?>
```

4.CODE FOR EDIT PLAYERS AND TRAINER DATA

(A)

```
<?php
include("db.php");
$title = '';
$description= '';
```

```
if  (isset($_GET['id'])) {
  $id = $_GET['id'];
  $query = "SELECT * FROM players WHERE id=$id";
  $result = mysqli_query($conn, $query);
  if (mysqli_num_rows($result) == 1) {
    $row = mysqli_fetch_array($result);
    $player_id = $row['player_id'];
    $name = $row['name'];
            $age = $row['age'];
            $dob = $row['dob'];
            $doj = $row['doj'];
  }
}

if (isset($_POST['update'])) {
  $id = $_GET['id'];
  $player_id = $_POST['player_id'];
  $name = $_POST['name'];
  $age = $_POST['age'];
  $dob = $_POST['dob'];
  $doj = $_POST['doj'];

  $query = "UPDATE players SET player_id='$player_id',name='$name',age='$age',dob='$dob',doj='$doj'
WHERE id=$id";
  mysqli_query($conn, $query);
  $_SESSION['message'] = 'Player Updated Successfully';
```

```
 $_SESSION['message_type'] = 'warning';

 header('Location: index.php');

}



?>

<?php include('includes/header.php'); ?>

<div class="container p-4">

 <div class="row">

   <div class="col-md-4 mx-auto">

    <div class="card card-body">

    <form action="edit_player.php?id=<?php echo $_GET['id']; ?>" method="POST">

                            <div class="form-group">

        <input type="text" name="player_id" class="form-control" placeholder="Enter player ID"
value="<?php echo $player_id; ?>" required autofocus>

        </div>

        <div class="form-group">

        <input type="text" name="name" class="form-control" placeholder="Enter Name" value="<?php
echo $name; ?>" required autofocus>

        </div>

        <div class="form-group">

        <input type="text" name="age" class="form-control" placeholder="Enter Age" value="<?php echo
$age; ?>" required autofocus>

        </div>

                            <div class="form-group">

        <input type="text" name="dob" class="form-control" placeholder="Enter DOB" value="<?php
echo $dob; ?>" required autofocus>

        </div>

                            <div class="form-group">
```

```
          <input type="text" name="doj" class="form-control" placeholder="Enter DOJ" value="<?php echo
$doj; ?>" required autofocus>

        </div>

       <button class="btn-success" name="update">

        Update

</button>

     </form>

     </div>

   </div>

  </div>

</div>

<?php include('includes/footer.php'); ?>

(B)

<?php

include("db.php");

$title = '';

$description= '';


if  (isset($_GET['id'])) {

  $id = $_GET['id'];

  $query = "SELECT * FROM trainers WHERE id=$id";

  $result = mysqli_query($conn, $query);

  if (mysqli_num_rows($result) == 1) {

    $row = mysqli_fetch_array($result);

    $trainer_id = $row['trainer_id'];

    $name = $row['name'];
```

```
        $experience = $row['experience'];

        $skill = $row['skill'];

        $admin_id = $row['admin_id'];

  }

}


if (isset($_POST['update'])) {

  $id = $_GET['id'];

  $trainer_id = $_POST['trainer_id'];

  $name = $_POST['name'];

  $experience = $_POST['experience'];

  $skill = $_POST['skill'];

  $admin_id = $_POST['admin_id'];


  $query = "UPDATE trainers SET
trainer_id='$trainer_id',name='$name',experience='$experience',skill='$skill',admin_id='$admin_id'
WHERE id=$id";

  mysqli_query($conn, $query);

  $_SESSION['message'] = 'trainer Updated Successfully';

  $_SESSION['message_type'] = 'warning';

  header('Location: trainer.php');

}


?>

<?php include('includes/header.php'); ?>

<div class="container p-4">

  <div class="row">
```

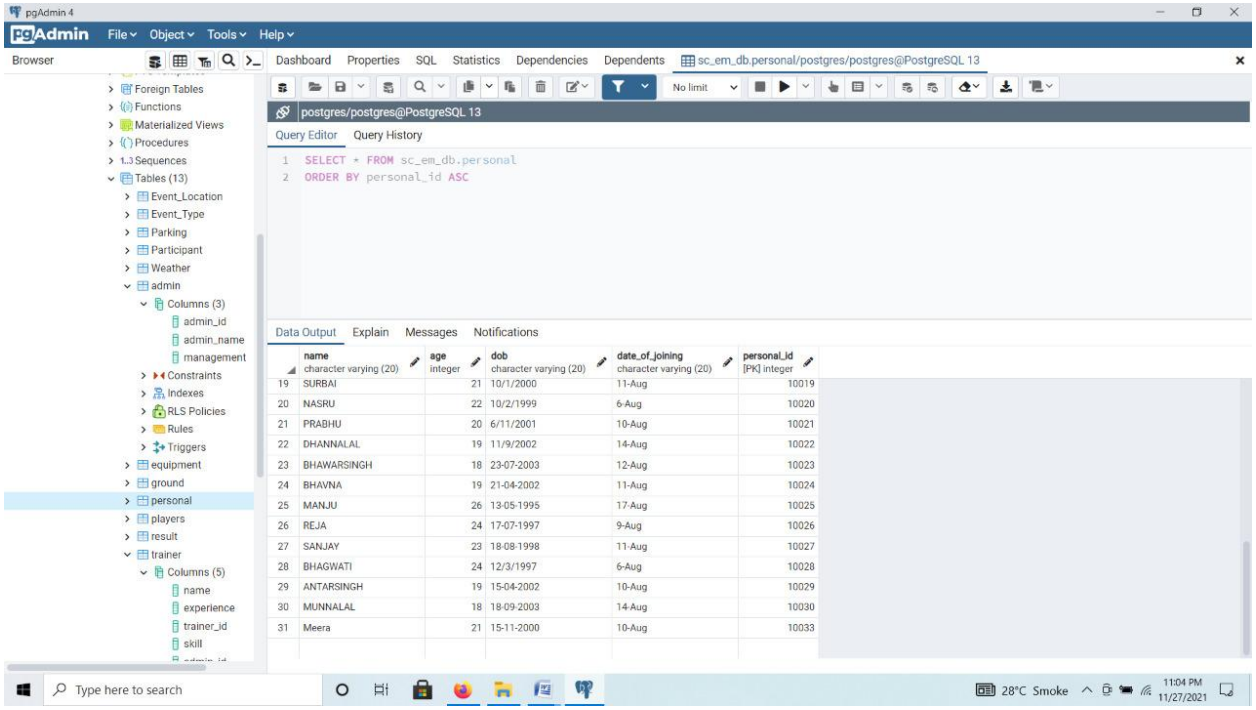
```
   <div class="col-md-4 mx-auto">

    <div class="card card-body">

    <form action="edit_trainer.php?id=<?php echo $_GET['id']; ?>" method="POST">

                            <div class="form-group">

         <input type="text" name="trainer_id" class="form-control" placeholder="Enter trainer ID"
value="<?php echo $trainer_id; ?>" required autofocus>

        </div>

        <div class="form-group">

         <input type="text" name="name" class="form-control" placeholder="Enter Name" value="<?php
echo $name; ?>" required autofocus>

        </div>

        <div class="form-group">

         <input type="text" name="experience" class="form-control" placeholder="Enter experience"
value="<?php echo $experience; ?>" required autofocus>

        </div>

                            <div class="form-group">

         <input type="text" name="skill" class="form-control" placeholder="Enter skill" value="<?php echo
$skill; ?>" required autofocus>

        </div>

                            <div class="form-group">

         <input type="text" name="admin_id" class="form-control" placeholder="Enter admin_id"
value="<?php echo $admin_id; ?>" required autofocus>

        </div>

      <button class="btn-success" name="update">

        Update

</button>

    </form>

    </div>
```

```
  </div>
 </div>
</div>
<?php include('includes/footer.php'); ?>
```

Screenshot of output or web/app pages with data

(A)Players

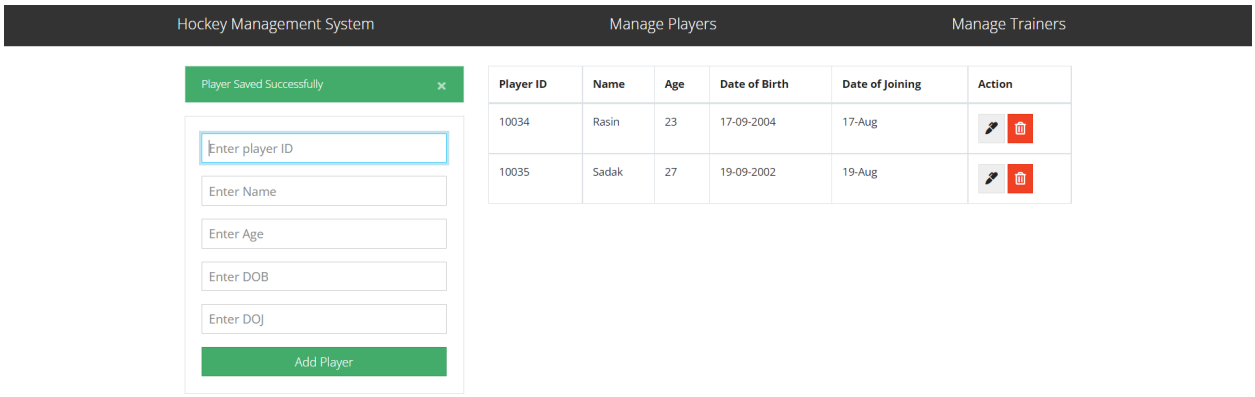
Before insertion

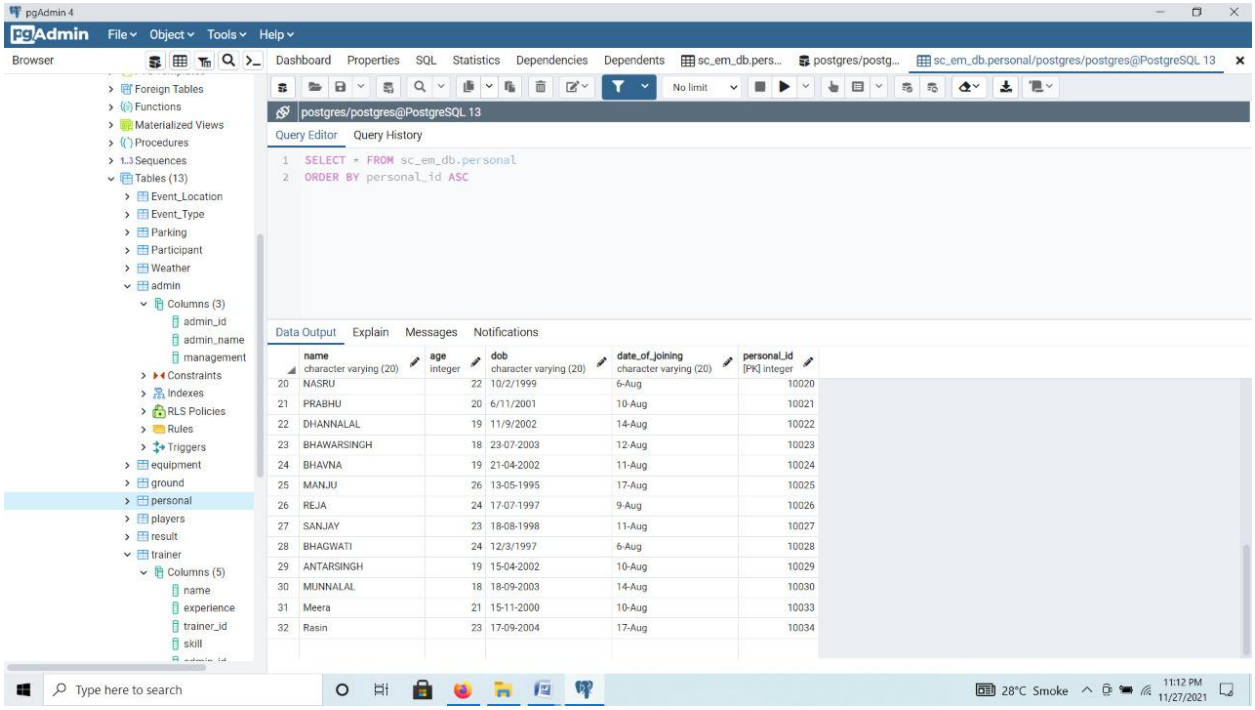


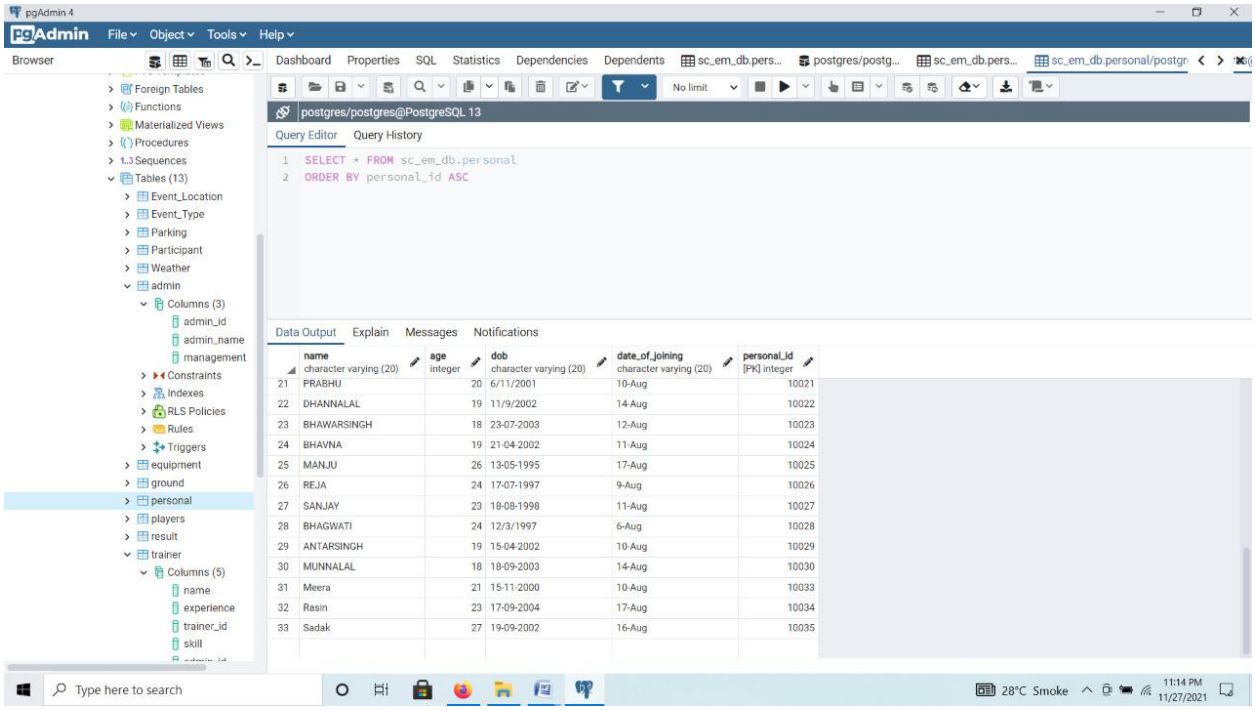
	name character varying (20)	age integer	dob character varying (20)	date_of_joining character varying (20)	personal_id [PK] integer
19	SURBAI	21	10/1/2000	11-Aug	10019
20	NASRU	22	10/2/1999	6-Aug	10020
21	PRABHU	20	6/11/2001	10-Aug	10021
22	DHANNALAL	19	11/9/2002	14-Aug	10022
23	BHAWARSINGH	18	23-07-2003	12-Aug	10023
24	BHAVNA	19	21-04-2002	11-Aug	10024
25	MANJU	26	13-05-1995	17-Aug	10025
26	REJA	24	17-07-1997	9-Aug	10026
27	SANJAY	23	18-08-1998	11-Aug	10027
28	BHAGWATI	24	12/3/1997	6-Aug	10028
29	ANTARSINGH	19	15-04-2002	10-Aug	10029
30	MUNNALAL	18	18-09-2003	14-Aug	10030
31	Meera	21	15-11-2000	10-Aug	10033

After Insertion

Web Page



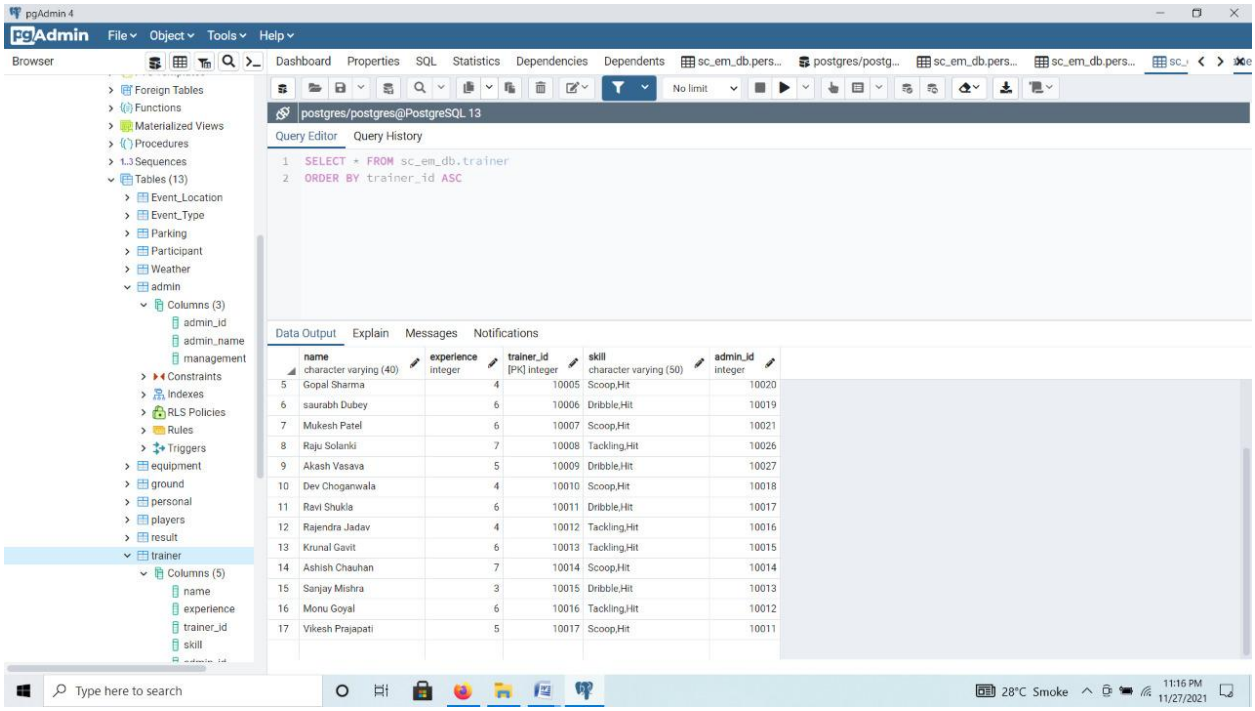




	name character varying (20)	age integer	dob character varying (20)	date_of_joining character varying (20)	personal_id [PK] integer
21	PRABHU	20	6/11/2001	10-Aug	10021
22	DHANNALAL	19	11/9/2002	14-Aug	10022
23	BHAWARSINGH	18	23-07-2003	12-Aug	10023
24	BHAVNA	19	21-04-2002	11-Aug	10024
25	MANJU	26	13-05-1995	17-Aug	10025
26	REJA	24	17-07-1997	9-Aug	10026
27	SANJAY	23	18-08-1998	11-Aug	10027
28	BHAGWATI	24	12/3/1997	6-Aug	10028
29	ANTARSINGH	19	15-04-2002	10-Aug	10029
30	MUNNALAL	18	18-09-2003	14-Aug	10030
31	Meera	21	15-11-2000	10-Aug	10033
32	Rasin	23	17-09-2004	17-Aug	10034
33	Sadak	27	19-09-2002	16-Aug	10035

(B) Trainers

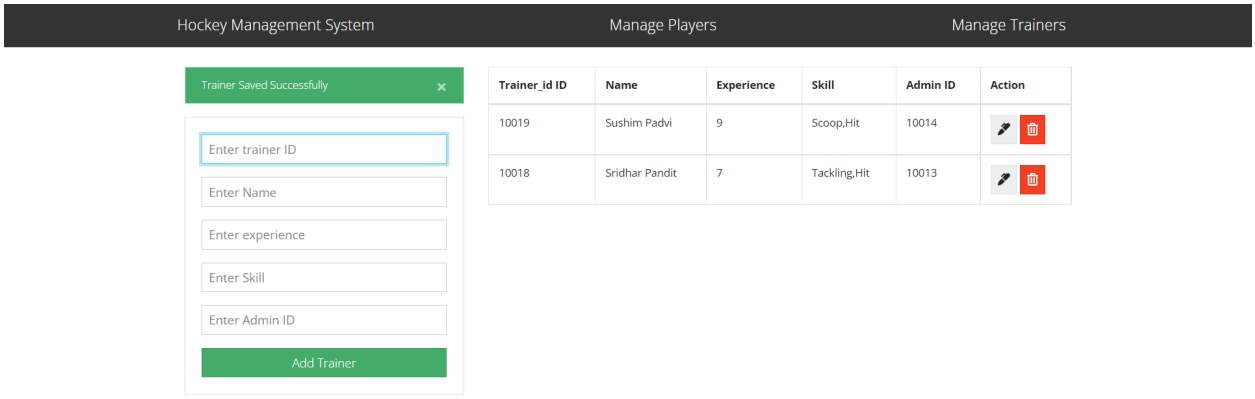
Before insertion

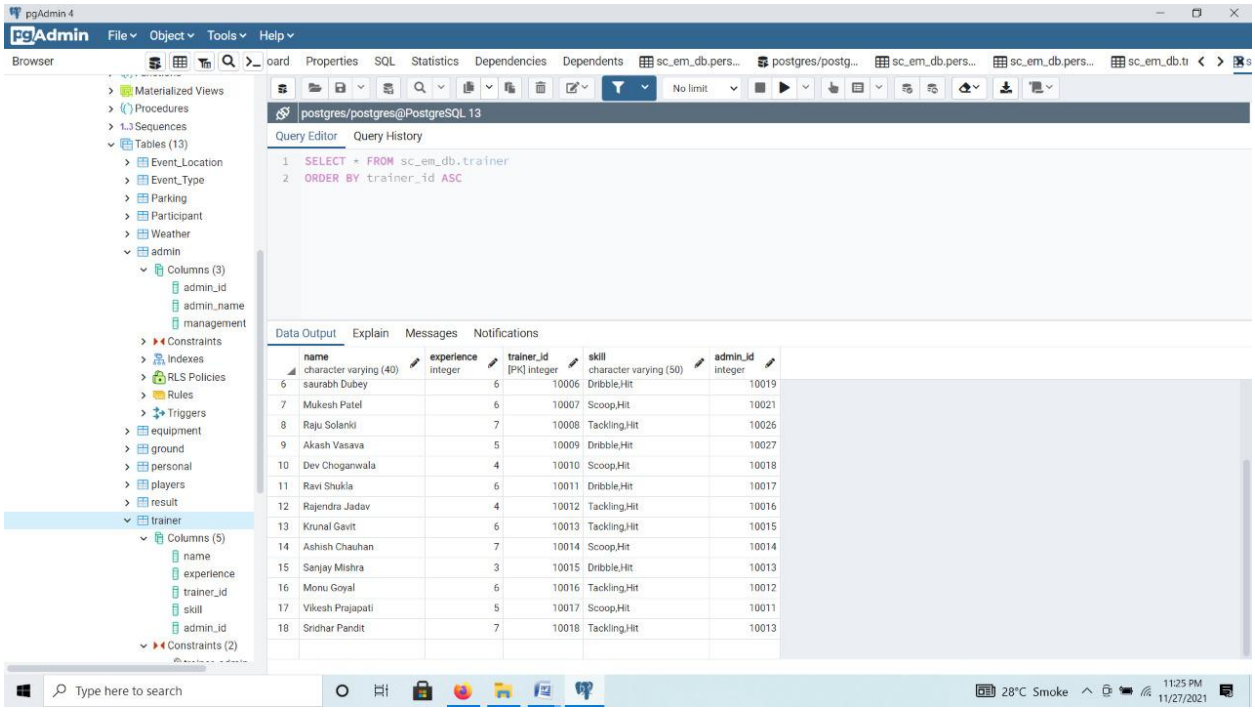


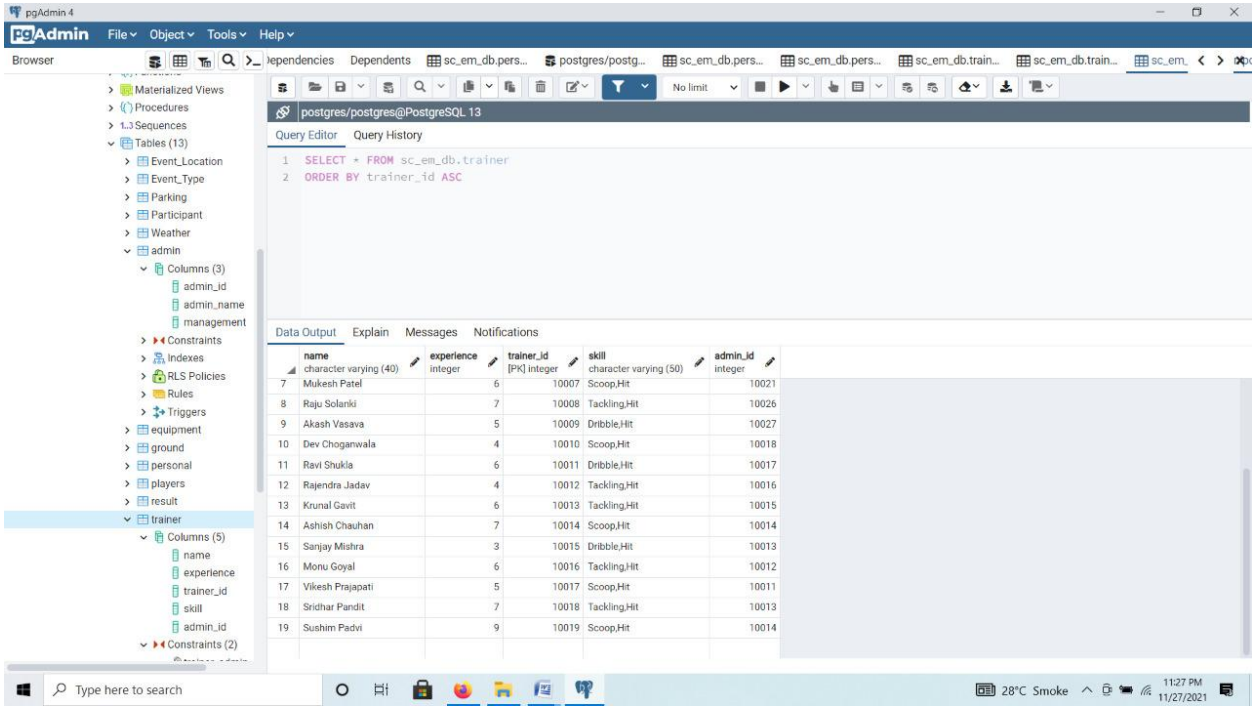
	name character varying (40)	experience integer	trainer_id [PK] integer	skill character varying (50)	admin_id integer
5	Gopal Sharma	4	10005	Scoop,Hit	10020
6	saurabh Dubey	6	10006	Dribble,Hit	10019
7	Mukesh Patel	6	10007	Scoop,Hit	10021
8	Raju Solanki	7	10008	Tackling,Hit	10026
9	Akash Vasava	5	10009	Dribble,Hit	10027
10	Dev Choganwala	4	10010	Scoop,Hit	10018
11	Ravi Shukla	6	10011	Dribble,Hit	10017
12	Rajendra Jadav	4	10012	Tackling,Hit	10016
13	Krunal Gavit	6	10013	Tackling,Hit	10015
14	Ashish Chauhan	7	10014	Scoop,Hit	10014
15	Sanjay Mishra	3	10015	Dribble,Hit	10013
16	Monu Goyal	6	10016	Tackling,Hit	10012
17	Vikesh Prajapati	5	10017	Scoop,Hit	10011

After insertion

Web Page







	name character varying (40)	experience integer	trainer_id [PK] integer	skill character varying (50)	admin_id integer
7	Mukesh Patel	6	10007	Scoop,Hit	10021
8	Raju Solanki	7	10008	Tackling,Hit	10026
9	Akash Vasava	5	10009	Dribble,Hit	10027
10	Dev Choganwala	4	10010	Scoop,Hit	10018
11	Ravi Shukla	6	10011	Dribble,Hit	10017
12	Rajendra Jadav	4	10012	Tackling,Hit	10016
13	Krunal Gavit	6	10013	Tackling,Hit	10015
14	Ashish Chauhan	7	10014	Scoop,Hit	10014
15	Sanjay Mishra	3	10015	Dribble,Hit	10013
16	Monu Goyal	6	10016	Tackling,Hit	10012
17	Vikesh Prajapati	5	10017	Scoop,Hit	10011
18	Sridhar Pandit	7	10018	Tackling,Hit	10013
19	Sushim Padvi	9	10019	Scoop,Hit	10014