



Dhirubhai Ambani Institute of Information and Communication Technology

Final Submission

Project Title: College Fest Database

Submitted to:
Teaching Assistant:
Subject:

Dr. Minal Bhise
Vedant Dave
DBMS LAB

Submitted by:

- Aditya Kumar (202212046)
- Saheb Singh (202212071)

Course: MSc IT (2022-24)

Index

1. SRS.....	3
2. Problem Description.....	3
a. Requirement collection documentation.....	4
b. List of requirements.....	4
c. Identification of constraints.....	5
d. Identification of various categories of users and their privileges.....	5
3. ER Diagram.....	16
4. ER to Relational Mapping.....	17
5. Final List of relations with attributes and constraints.....	17
6. DDL Statements for creating tables.....	24
7. Details of populating data into the tables.....	26
8. List of queries for the database (English).....	37
9. List of queries for the database (SQL).....	39

Software Requirement Specifications:

College Fest Database Problem Statement:

- This database includes the detail about college fest of any college.
- It maps student details to his/her participation in any fest activities.
- It has separate details about winners and prizes.
- This system also includes detail about teams formed during the fest.

Description:

The college fest database contains the information about events, event co-ordinators, event participants, results. It also contains the information for the schedule of events over the span of the event.

Problem Scenario:

The handling of event registration data and teams is tough if done using pen and paper. Here comes the database management system to the rescue as it makes the data storage and accessing really simple and convenient.

Definition:

Normally the registration process for college fest is done on papers and it consumes a lot of time and effort for completing and managing these documents. And to eliminate this extra effort we have designed a database management system.

Features of the database:

- View the events hosted during the span of the college fest
- Registration for the events
 - Team based registration
 - Solo registration
- Event registration committee details
- Event discipline committee details
- Generating a access code for the students who register for any event

Objective:

The objective of this approach is to simplify the registration process for the college events and make it easier to access, manage, and view the required data.

This database helps check if an event committee or event discipline committee is busy on an event during the same time slot.

Research plan:

- Reading about how events are managed at a large scale
- Reading about the team to be created for different management.
- Ground reality about the official documents for the event location.
- Researching about the permission to be taken from higher authorities.
- To understand the concept of event management
- To study the different types of events
- To analyse the role of creativity in our event process. Questionnaire:
 - Choose the proffered themes for your college fest
 - Select 16 events of your choice.
 - Choose between rock band and DJ.
 - Choose your DJ;
 - Choose your band
 - What would you like to improve from the previous fest?
 - How would like to rate our previous fest.

Requirements Collection Phase:

- Functional Requirements
- Non-functional requirements

Functional requirements:

- Student registration
- Student login
- Viewing the events and time slots for each event
- Registration for an event
- Issuing a registration number to the students

Non-functional requirements:

- Allow event registrations 1 day before the event.
- Forbid event registrations if the event date and time has passed.
- Send a confirmation email to the students on registration with the registration id.
- Show the list of registered events to the students in a dashboard

Reliability of the system:

The system should be reliable and a backup should be maintained at regular intervals and the database should be rolled back to the previous state in case of any failure.

Maintainability of the system: The database code and queries should be maintainable and any complex functionality should be divided into smaller modules.

Accessibility of the system: The application will be web based and be accessible through an internet connected device at all times.

Security of the system: The application must be secure against common SQL injection and XSS attacks. The user data should be encrypted and stored in a secure way.

Portability of the system: The application must be portable as it might get accessed through different devices and browsers.

Privileges:

1. User satisfaction: The main priority of a database is to provide its users with the ease of operating on the database.
2. Analytics: The database should provide its users with the basic analytics.
3. Ease of Payments: The users should have multiple options for payments.

Assumptions:

1. User has access to internet and is apt enough to use UI.
2. User is able to do online transaction through one of the many options available.
3. User has a modern web browser that supports features of the ticket booking platform.
4. User has a modern device which can smoothly function during the ticket booking process.

Constraints:

1. Equipment constraint: The database system must be optimized to process the queries in less time.
2. People constraint: The database system should be easy to operate and the tables should be stored with easily identifiable names.
3. Time constraint: The database system should be responsive enough to process the queries without any delays and it should be easily maintainable.

Fact finding techniques:

- Background reading
- Interviews
- Observations
- Questionnaires

Background reading:

The college fest database system is designed to solve the general requirements for registering, maintaining and accessing the student data who have registered for an event. The objective is to design an application that will facilitate the student registration process and allow the students to register for the events of their liking. The system will also show if there are any overlapping events and notify the students about those events. On successful registration a registration number will be generated which will be emailed to the student and the event management committee. The system will allow the students to register for any event only till 1 day before the event.

The event is categorised in local event, engendering pride in the community, strengthening a feeling of belonging and creating a sense of place.

The event will be a destination event because; people from different colleges will be there for enjoying the fest.

The permissions and the documentation process are to be done from a verified police officer for the DJ and band authorities. An accommodation committee will be doing all the work related to accommodation of the guest and the notable artist which are going to perform.

We need to create a database table for the arrival time of the artist and assign a dedicated team to a separate artist for the accommodation and services.

A total number of seats to be kept in the front of the stage are to be calculated and kept with the help of staff. The seats are calculated by the amount of registration and the google form which we circulated.

The security committee will decide how much security is needed in every event for the security of students and welfare of the event. The students' personal data may be collected at the time of registering for the event like:

- Course
- Student id
- Gender

The event is organised in such a planned structure that a person can take part in all the events at once.

Interview Plan:

DA-IICT College Fest: Interview Plan

System: DA-IICT College Fest

Project Reference:

Participants: Aditya Kumar (202212046)

Saheb Singh (202212071)

Date: 27-August-2022 Time: 15:00

Duration: 1 Hour Place: DA-IICT General Lab

Purpose of Interview: Initial meeting to identify the requirements for creating a database for college fest.

Agenda:

- Number of events and types of events per day.
- Approx. number of participants in each event.
- Time slots and day for the events.
- Media team to cover the photography, videography, and social media promotions.
- Initial ideas to discuss during the meeting.
- Follow up actions on the meeting discussions.

Documents to be brought to the interview:

- Rough plan of the event and types of events.
- Description of the events.
- Approx. number of events and participants.

Interview Summary:

DA-IICT College Fest: Interview Summary

System: DA-IICT College Fest

Project Reference:

Participants: Aditya Kumar (202212046)

Saheb Singh (202212071)

Date: 27-August-2022 Time: 15:00

Duration: 1 Hour Place: DA-IICT General Lab Purpose of Interview:

Initial meeting to identify the problems and requirements regarding security at the college fest.

- Fest duration [number of days].
- Number of events per day.
- Types of events.
- Number of approx. participants.
- Time slot for the events.
- Number of event co-ordinator.
- Creating an advertisement committee.
- Opening ceremony/ closing ceremony of the event.
- Various prize distributions for the winners in the event. • Initializing of the security committee and event committee.
- Further discussion needed when more information is available. (arrange follow up meeting with Saheb and Aditya)

Observations:

Objective	Technique	Subject	Time Commitment
To get background of college festival and the events.	Background reading	College fest documents	0.5 day.
To gain understanding of the events, events committee, events coordinator, media team, event plan.	Interview	Gain understanding / event management committee.	0,5 hour per day
To discuss out how the events will be managed, event coordinator, event participants, results.	Interview	Event management committee	2 X1 each day
To follow up on development of fest database.	Observation	Organising committee	2 X 1 Hour
To determine the role and relation of events committee, discipline committee, media team.	Interview	Human resource committee	3 X 1 Hour
To decide what event related documents are kept.	Interview/document sampling.	Resource manager	1.5 Hour each
To decide additional requirements for the new database.	Interview	Core committee/organising committee	0.5 each day

Questionnaire:

DA-IICT – Survey for DA-IICT college fest. Please tick our answers for the following questions.

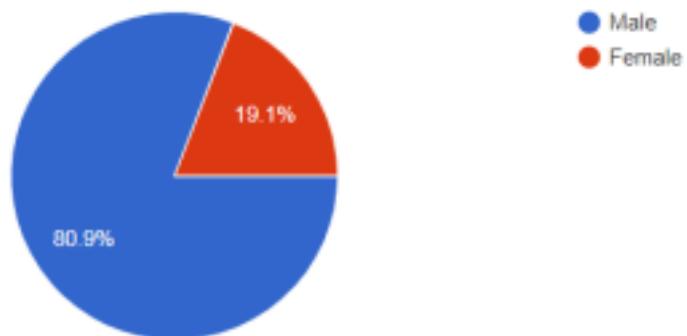
- Your Name
- Gender
 - Male
 - Female
- Age Group
 - 15 to 18 Years
 - 18 to 25 Years
 - 25+ Year
- Are you planning to attend the fest?
 - Yes
 - No
- Email ID
- ID Number
- Choose your preferable theme.
 - Cultural
 - Back to the 90s
 - Halloween
 - Hawaii Magic
- Select 16 events of your choice
 - LAN Gaming
 - Robo Game
 - Poetry writing and recitation
 - Face painting
 - Debugging
 - Marketing and Selling
 - Story writing competition
 - Short film making
 - Young scientist and innovation
 - Handcrafts
 - Cultural competition

- Fashion show
- Culinary contest
- Debate
- Murder investigation
- Laser tag
- Rubix cube competition
- Treasure hunt
- Mimicry competition
- Photography
- Choose between Rock band and DJ
 - Rock Band
 - DJ
- Choose your DJ
 - David Guetta
 - Martin Garrix
 - Nucleya
 - DJ Man-Dee
- Choose your band
 - The Local Train
 - Indian Ocean
 - The Byrds
 - Testo Sisters
- What would you like to improve from the previous fest?
- How would you rate our previous fest?

Questionnaire Results:

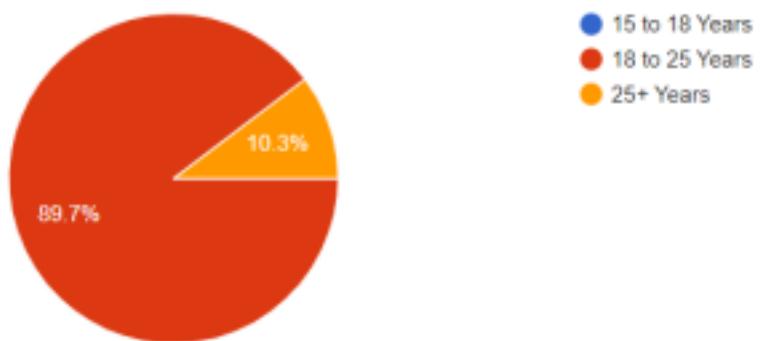
Gender

68 responses



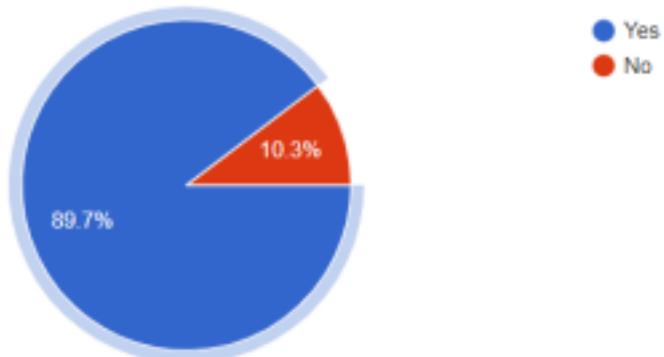
Age Group

68 responses



Are you planning to attend the college fest?

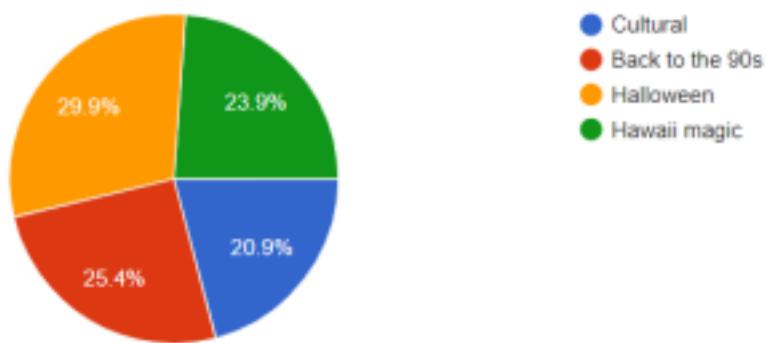
68 responses



Survey

Choose your preferable theme for your college fest

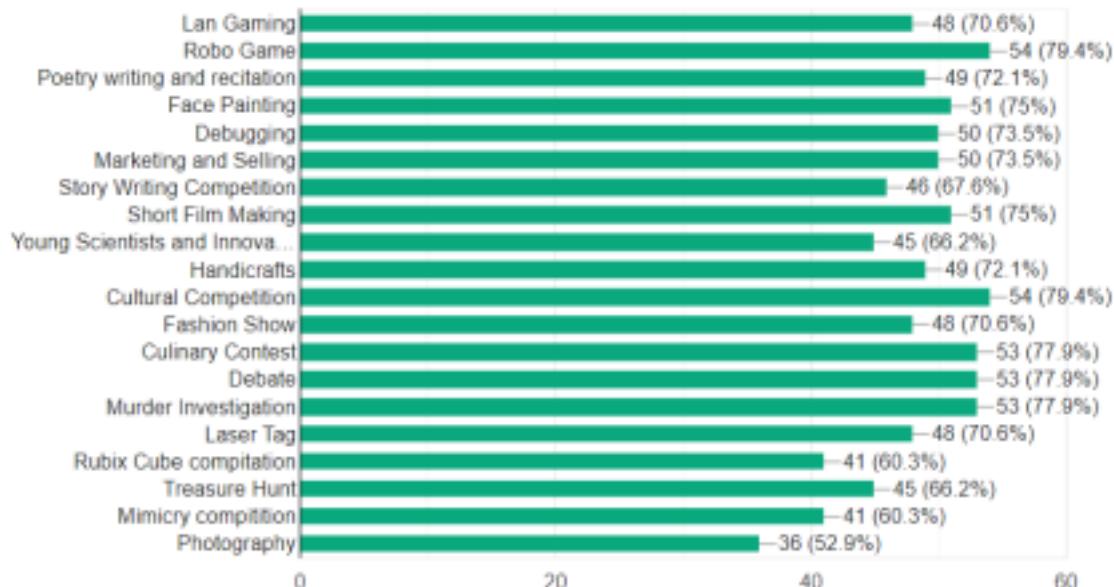
67 responses



Select 16 events of your choice out of 20.

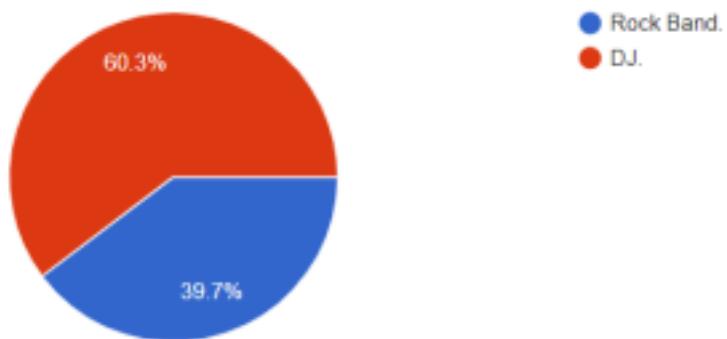
Copy

68 responses



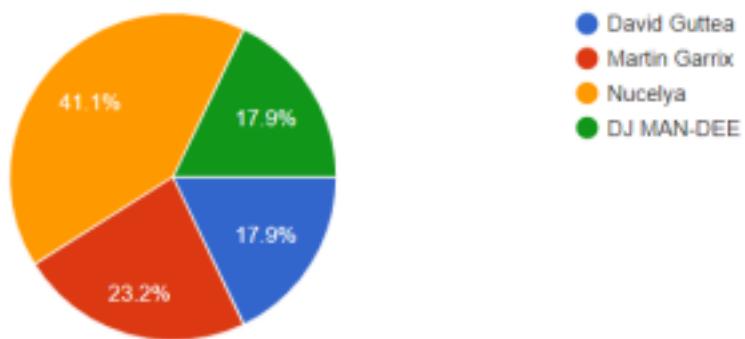
Choose between RockBand or DJ:

68 responses



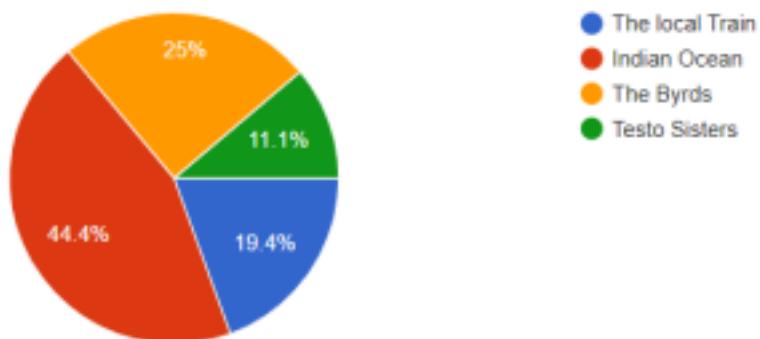
Choose your DJ

56 responses



Choose Your Band.

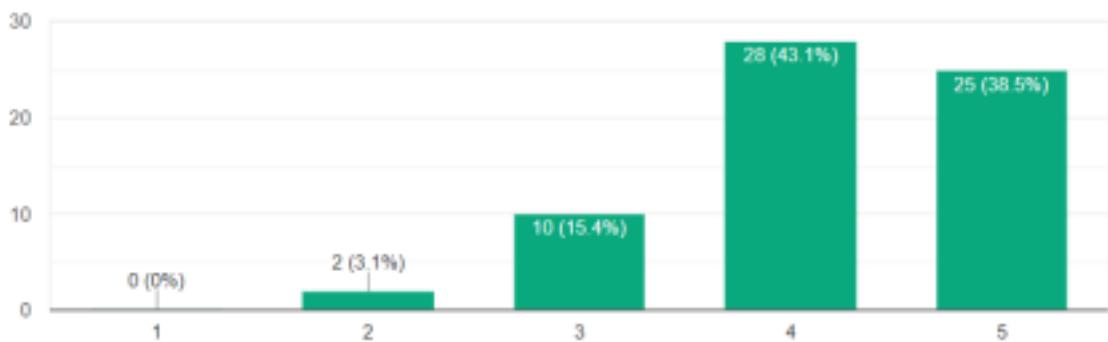
36 responses



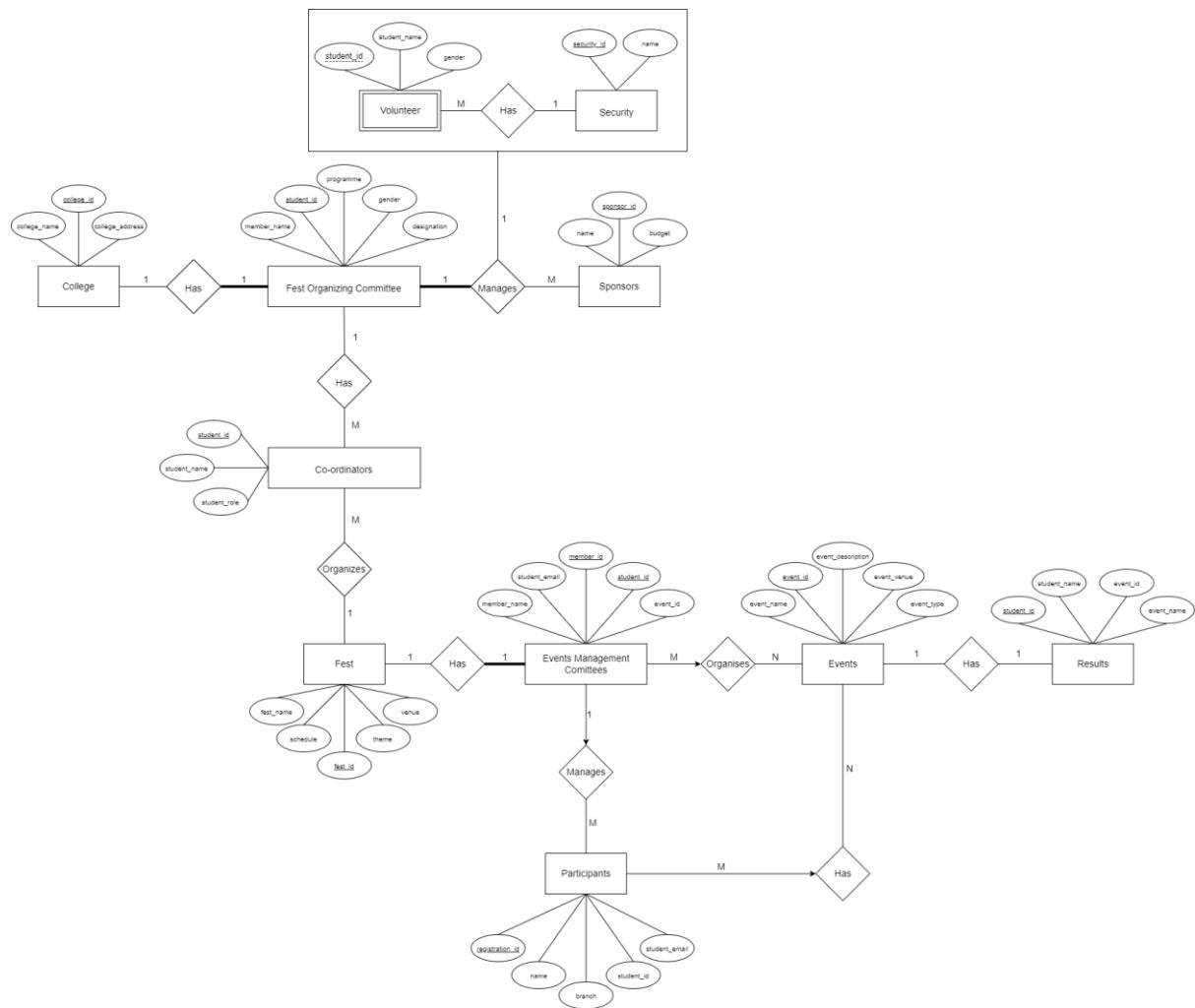
How would like to rate our previous fest?

 Copy

65 responses



ER Diagram:



Final List of Attributes and Constraints:

1. Relation: College

College (college_name : string, college_id : string, college_address : string)

Attribute Name	Constraints
College_name	NOT NULL
College_id	PRIMARY KEY
College_address	NOT NULL

2. Relation: Fest Organising Committee

Fest_organising_committee (member_name : string, student_id : long integer, programme : string, gender : string, designation : string)

Attribute Name	Constraints
Member_name	NOT NULL
Student_id	PRIMARY KEY
Programme	NOT NULL
Gender	NOT NULL
Designation	NOT NULL

3. Relation: Sponsors

Sponsors (name : string, sponsor_id : long integer, Budget : integer)

Attribute Name	Constraints
Name	NOT NULL
Sponsor_id	PRIMARY KEY
Budget	NOT NULL

4. Relation: Co-ordinators

Coordinators (student_id : integer, student_name : string, student_role : string)

Attribute Name	Constraints
Student_id	PRIMARY KEY
Student_name	NOT NULL
Student_role	NOT NULL

5. Relation: Fest

Fest (fest_name : string, schedule : date_time, fest_id : string, theme : string, venue : string)

Attribute Name	Constraints
Fest_name	NOT NULL
Schedule	NOT NULL
Fest_id	PRIMARY KEY
Theme	NOT NULL
Venue	NOT NULL

6. Relation: Event Management Committee

Event_management_committee (member_name : string, student_email : string, member_id : string, student_id : integer, event_id : string)

Attribute Name	Constraints
Member_name	NOT NULL
Student_email	NOT NULL
Member_id	NOT NULL
Student_id	PRIMARY KEY
Event_id	NOT NULL

7. Relation: Events

Events (event_name : String, event_id : string, event_description : String, event_venue : String, event_type : String)

Attribute Name	Constraints
Event_name	NOT NULL
Event_id	PRIMARY KEY
Event_description	NOT NULL
Event_venue	NOT NULL
Event_type	NOT NULL

8. Relation: Results

Results (student_id : Integer, student_name : String, event_id : String, event_name : String)

Attribute Name	Constraints
Student_id	PRIMARY KEY
Student_name	NOT NULL
Event_id	NOT NULL
Event_name	NOT NULL

9. Relation: Participants

Participants (registration_id : Integer, Name : String, Branch : String, Student_id : Integer, Student_email : String)

Attribute Name	Constraints
Registration_id	PRIMARY KEY
Name	NOT NULL
Branch	NOT NULL
Student_id	NOT NULL
Student_email	NOT NULL

10. Relation: Volunteer

Volunteer (student_id : Integer, student_name : String, gender : Character)

Attribute Name	Constraints
Student_id	PARTIAL KEY
Student_name	NOT NULL
Gender	NOT NULL

11. Relation: Security

Security (security_id : Integer, name : String)

Attribute Name	Constraints
Security_id	PRIMARY KEY
Name	NOT NULL

ER Diagram to Relational Mapping:

Relation: College

College (college_name : string, college_id : string, college_address : string)

Attribute Name	Data Type	Constraints	Description
College_name	String	NOT NULL	Name of the college.
College_id	String	PRIMARY KEY	ID number of the college.
College_address	String	NOT NULL	Address of the college.

Relation: Fest Organising Committee

Fest_organising_committee (member_name : string, student_id : long integer, programme : string, gender : string, designation : string)

Attribute Name	Data Type	Constraints	Description
Member_name	String	NOT NULL	Name of fest organising committee member.
Student_id	Long integer	PRIMARY KEY	ID number of the member.
Programme	String	NOT NULL	Enrolled programme of the fest organising committee member.
Gender	String	NOT NULL	Gender of the member.
Designation	String	NOT NULL	Designation of the Fest organising committee member.

Relation: Sponsors

Sponsors (name : string, sponsor_id : long integer, Budget : integer)

Attribute Name	Data Type	Constraints	Description
Name	String	NOT NULL	Name of the sponsors.
Sponsor_id	Long integer	PRIMARY KEY	ID number of the sponsor.
Budget	Integer	NOT NULL	Budget of the sponsors.

Relation: Co-ordinators

Coordinators (student_id : integer, student_name : string, student_role : string)

Attribute Name	Data Type	Constraints	Description
Student_id	Integer	PRIMARY KEY	ID number of the student.
Student_name	String	NOT NULL	Name of the student.
Student_role	String	NOT NULL	Role of the student as co-ordinator.

Relation: Fest

Fest (fest_name : string, schedule : date_time, fest_id : string, theme : string, venue : string)

Attribute Name	Data Type	Constraints	Description
Fest_name	String	NOT NULL	Name of the college fest.
Schedule	Date Time	NOT NULL	Schedule of the fest.
Fest_id	String	PRIMARY KEY	ID number of the college fest.
Theme	String	NOT NULL	Theme of the college fest.
Venue	String	NOT NULL	Venue details of the college fest.

Relation: Event Management Committee

Event_management_committee (member_name : string, student_email : string, member_id : string, student_id : integer, event_id : string)

Attribute Name	Data Type	Constraints	Description
Member_name	String	NOT NULL	Name of the member.
Student_email	String	NOT NULL	Email address of the student member.
Member_id	String	NOT NULL	ID number of the student member.
Student_id	Integer	PRIMARY_KEY	Student ID of the member.
Event_id	String	NOT NULL	Event ID of the event managed by the student member.

Relation: Events

Events (event_name : String, event_id : string, event_description : String, event_venue : String, event_type : String)

Attribute Name	Data Type	Constraints	Description
Event_name	String	NOT NULL	Name of the event being organised.
Event_id	String	PRIMARY KEY	ID of the event being organised.
Event_description	String	NOT NULL	Description of the event being organised.
Event_venue	String	NOT NULL	Venue of the event being organised.
Event_type	String	NOT NULL	The category of the event being organised.

Relation: Results

Results (student_id : Integer, student_name : String, event_id : String, event_name : String)

Attribute Name	Data Type	Constraints	Description
Student_id	Integer	PRIMARY KEY	ID number of the student.
Student_name	String	NOT NULL	Name of the student.
Event_id	String	NOT NULL	ID of the event.
Event_name	String	NOT NULL	Name of the event.

Relation: Participants

Participants (registration_id : Integer, Name : String, Branch : String, Student_id : Integer, Student_email : String)

Attribute Name	Data Type	Constraints	Description
Registration_id	Integer	PRIMARY KEY	Registration ID of the participant.
Name	String	NOT NULL	Name of the participant.
Branch	String	NOT NULL	Course branch of the participants.
Student_id	Integer	NOT NULL	ID number of the students.
Student_email	String	NOT NULL	Email address of the students.

Relation: Volunteer

Volunteer (student_id : Integer, student_name : String, gender : Character)

Attribute Name	Data Type	Constraints	Description
Student_id	Integer	PARTIAL KEY	ID number of the student volunteer.
Student_name	String	NOT NULL	Name of the student volunteer.
Gender	Character	NOT NULL	Gender of the student volunteer.

Relation: Security

Security (security_id : Integer, name : String)

Attribute Name	Data Type	Constraints	Description
Security_id	Integer	PRIMARY KEY	ID number of the security members.
Name	String	NOT NULL	Name of the security members.

DDL Statements:

1. DDL for creating Schema:

```
CREATE SCHEMA College-Fest-DB
```

2. DDL for creating tables:

College:

```
CREATE TABLE College (
    College_name CHAR(20) NOT NULL,
    College_id CHAR(10) NOT NULL,
    College_address CHAR(50) NOT NULL,
    PRIMARY KEY (College_id)
);
```

Fest Organising Committee:

```
CREATE TABLE Fest_organising_committee (
    Member_name CHAR(20) NOT NULL,
    Student_id CHAR(10) NOT NULL,
    Programme CHAR(10) NOT NULL,
    Gender CHAR(6) NOT NULL,
    Designation CHAR(10) NOT NULL,
    PRIMARY KEY (Student_id)
);
```

Sponsors:

```
CREATE TABLE Sponsors (
    Name CHAR(20) NOT NULL,
    Sponsor_id CHAR(10) NOT NULL,
    Budget INT
    PRIMARY KEY (Sponsor_id)
);
```

Coordinators:

```
CREATE TABLE Coordinators (
    Student_id CHAR(10) NOT NULL,
    Student_name CHAR(50) NOT NULL,
    Student_role CHAR(10) NOT NULL
    PRIMARY KEY (Student_id)
);
```

Fest:

```
CREATE TABLE Fest (
    Fest_name CHAR(15) NOT NULL,
    Schedule DATE NOT NULL,
    Fest_id CHAR(10) NOT NULL,
    Theme CHAR(15) NOT NULL,
    Venue CHAR(10) NOT NULL
    PRIMARY KEY (Fest_id)
);
```

Event Management Committee:

```
CREATE TABLE Event_management_committee (
    Member_name CHAR(15) NOT NULL,
    Student_email CHAR(30) NOT NULL,
    Member_id CHAR(10) NOT NULL,
    Student_id CHAR(10) NOT NULL,
    Event_id CHAR(10) NOT NULL,
    PRIMARY KEY (Member_id)
);
```

Event:

```
CREATE TABLE Event (
    Event_name CHAR(15) NOT NULL,
    Event_id CHAR(10) NOT NULL,
    Event_description CHAR(50) NOT NULL,
    Event_venue CHAR(20) NOT NULL,
    Event_type CHAR(15) NOT NULL,
    PRIMARY KEY (Event_id)
);
```

Result:

```
CREATE TABLE Result (
    Student_id CHAR(15) NOT NULL,
    Student_name CHAR(20) NOT NULL,
    Event_id CHAR(10) NOT NULL,
    Event_name CHAR(15) NOT NULL
    PRIMARY KEY (Student_id)
);
```

Participants:

```
CREATE TABLE Participants (
    Registration_id CHAR(10) NOT NULL,
    Name CHAR(15) NOT NULL,
    Branch CHAR(15) NOT NULL,
    Student_id CHAR(10) NOT NULL,
    Student_email CHAR(30) NOT NULL
    PRIMARY KEY (Registration_id)
);
```

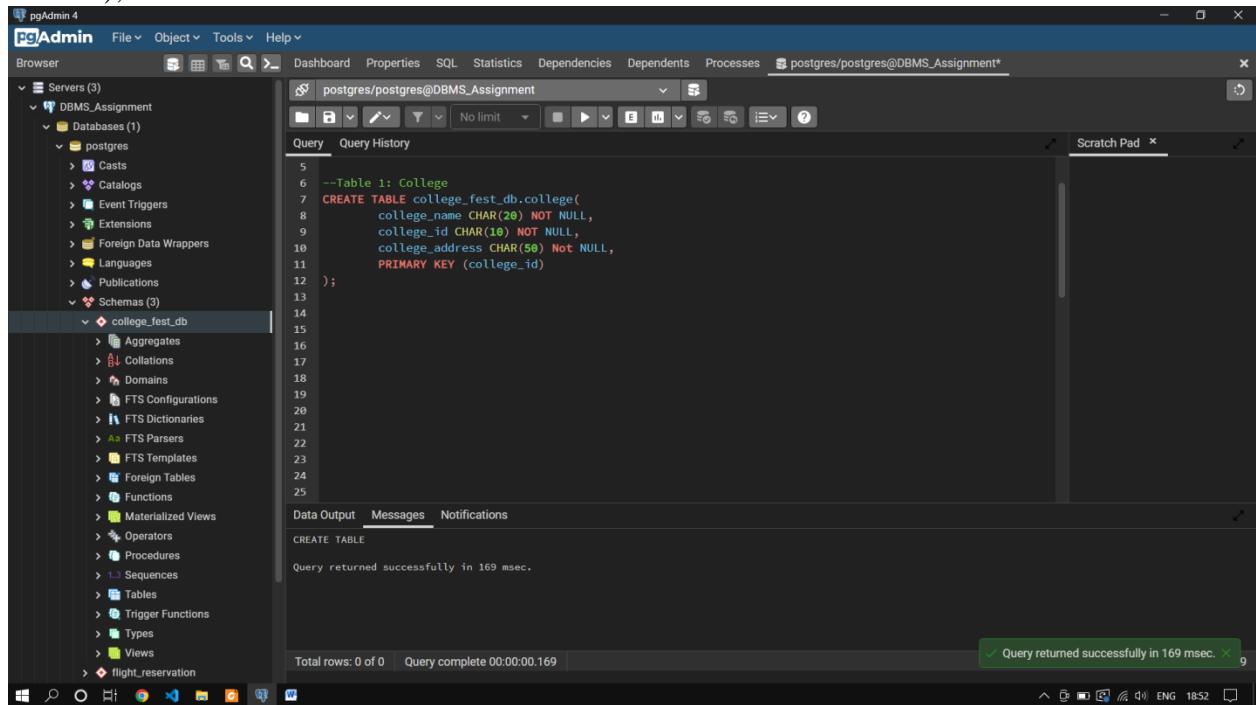
Details of creating tables and populating the data into table:

1. Create a table for College
2. Create a table for Fest Organising Committee
3. Create a table for Sponsors
4. Create a table for Coordinators
5. Create a table for Fest
6. Create a table for Event Management Committee
7. Create a table for Event
8. Create a table for Result
9. Create a table for Participants
10. Create a table for Volunteers
11. Create a table for Security

1. --Table 1: College

Query:

```
CREATE TABLE college_fest_db.college(  
    college_name CHAR(20) NOT NULL,  
    college_id CHAR(10) NOT NULL,  
    college_address CHAR(50) Not NULL,  
    PRIMARY KEY (college_id)  
);
```

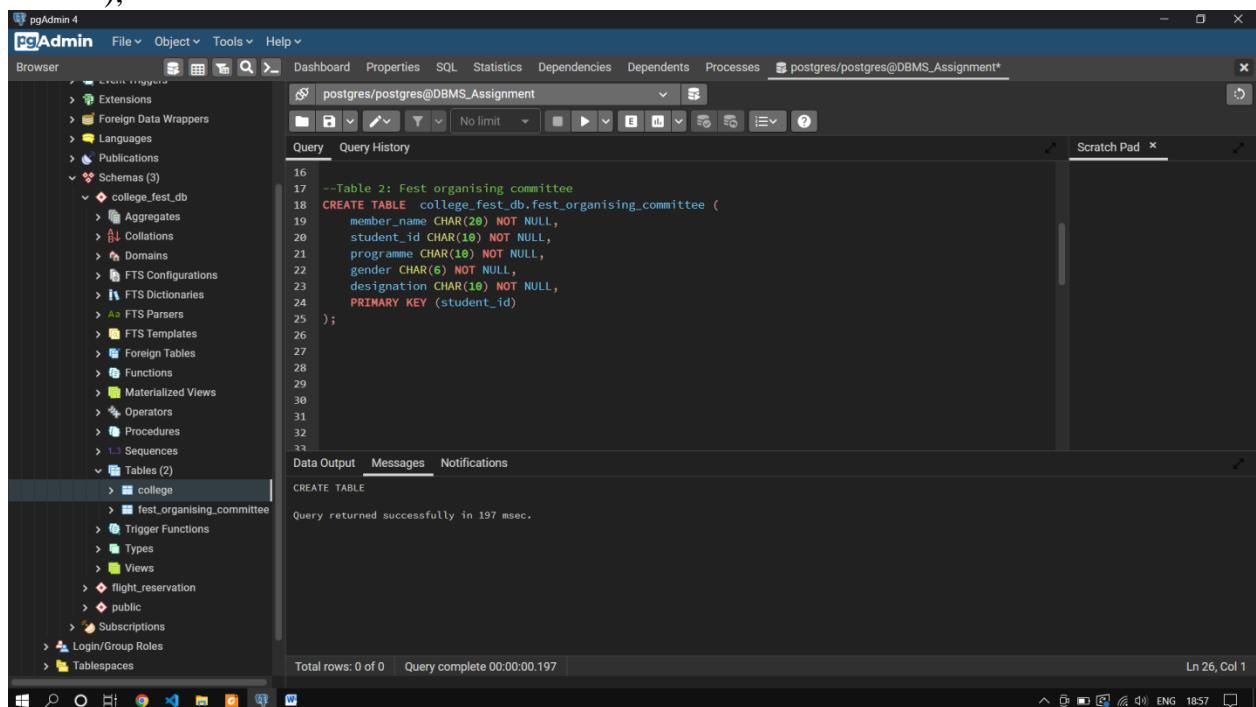


The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure under 'Servers (3)'. The 'DBMS_Assignment' server is selected, showing 'Databases (1)' containing 'postgres'. The 'Schemas (3)' section shows 'college_fest_db' selected, listing various objects like Aggregates, Collations, Domains, FTS Configurations, FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, Tables, Trigger Functions, Types, Views, and flight_reservation. The main query editor window shows the SQL code for creating the 'college' table. The status bar at the bottom indicates 'Query returned successfully in 169 msec.' and 'Total rows: 0 of 0 | Query complete 00:00:00.169'.

2. Table 2: Fest organising committee

Query:

```
CREATE TABLE college_fest_db.fest_organising_committee (
    member_name CHAR(20) NOT NULL,
    student_id CHAR(10) NOT NULL,
    programme CHAR(10) NOT NULL,
    gender CHAR(6) NOT NULL,
    designation CHAR(10) NOT NULL,
    PRIMARY KEY (student_id)
);
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure under the 'college_fest_db' schema, including tables like 'college', 'fest_organising_committee', and 'flight_reservation'. The main window shows the SQL query for creating the table, which is then executed successfully. The status bar at the bottom indicates 'Query returned successfully in 197 msec.'

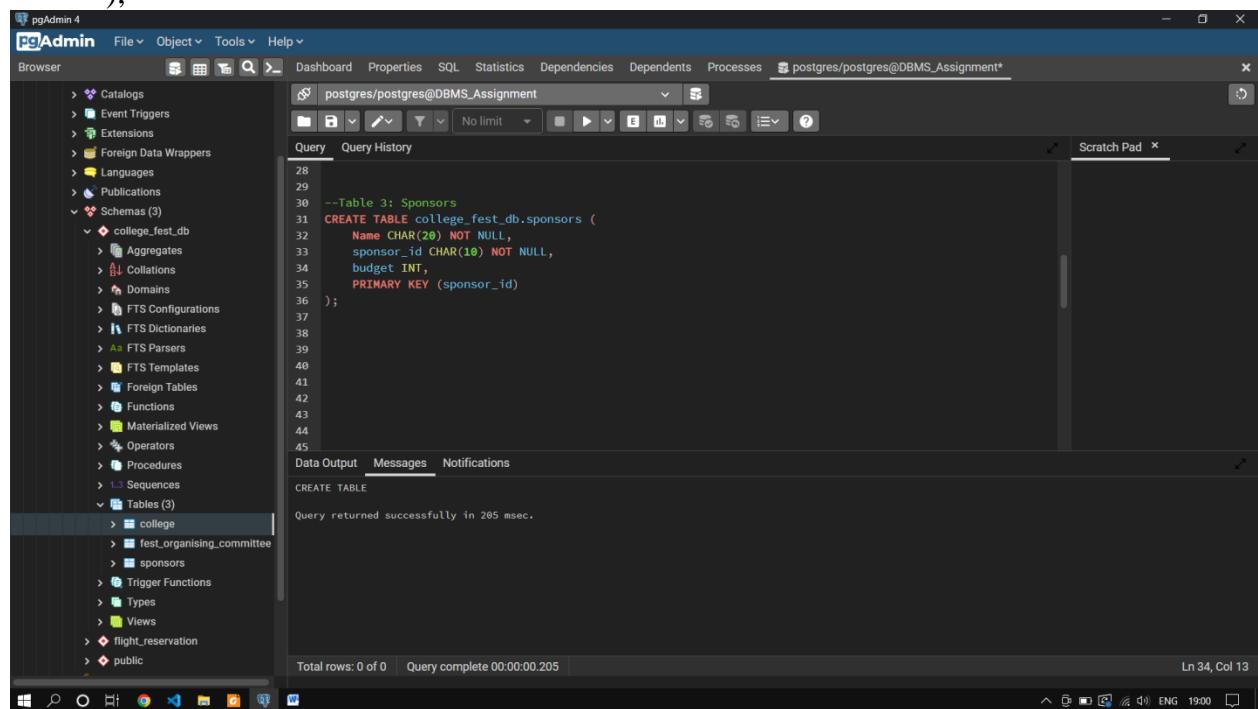
```
--Table 2: Fest organising committee
CREATE TABLE college_fest_db.fest_organising_committee (
    member_name CHAR(20) NOT NULL,
    student_id CHAR(10) NOT NULL,
    programme CHAR(10) NOT NULL,
    gender CHAR(6) NOT NULL,
    designation CHAR(10) NOT NULL,
    PRIMARY KEY (student_id)
);
```

Total rows: 0 of 0 | Query complete 00:00:00.197 | Ln 26, Col 1

3. Table 3: Sponsors

Query:

```
CREATE TABLE college_fest_db.sponsors (
    Name CHAR(20) NOT NULL,
    sponsor_id CHAR(10) NOT NULL,
    budget INT,
    PRIMARY KEY (sponsor_id)
);
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure, including the 'college_fest_db' schema which contains tables like 'college', 'fest_organising_committee', 'sponsors', and 'flight_reservation'. The main window (Query) shows the SQL code for creating the 'sponsors' table. The code is as follows:

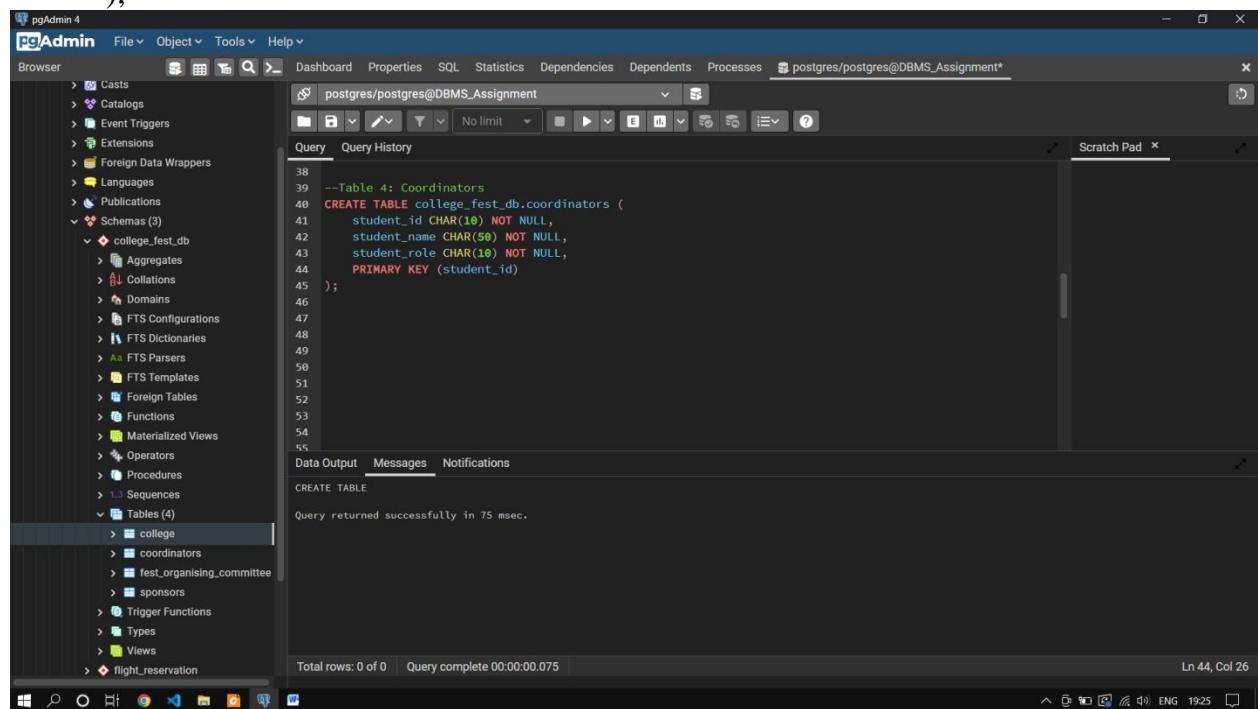
```
28
29
30 --Table 3: Sponsors
31 CREATE TABLE college_fest_db.sponsors (
32     Name CHAR(20) NOT NULL,
33     sponsor_id CHAR(10) NOT NULL,
34     budget INT,
35     PRIMARY KEY (sponsor_id)
36 );
37
38
39
40
41
42
43
44
45
```

Below the code, the 'Data Output' tab shows the message: 'Query returned successfully in 285 msec.' and 'Total rows: 0 of 0'. The 'Messages' tab shows 'Query complete 00:00:00.205' and 'Ln 34, Col 13'. The bottom status bar shows 'Windows' and 'Eng 19:00'.

4. Table: Coordinators

Query:

```
CREATE TABLE college_fest_db.coordinators (
    student_id CHAR(10) NOT NULL,
    student_name CHAR(50) NOT NULL,
    student_role CHAR(10) NOT NULL,
    PRIMARY KEY (student_id)
);
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure, including Schemas (college_fest_db), Tables (coordinators), and other objects like college, fest_organising_committee, sponsors, etc. The main window (Query) shows the SQL code for creating the coordinators table. The code is as follows:

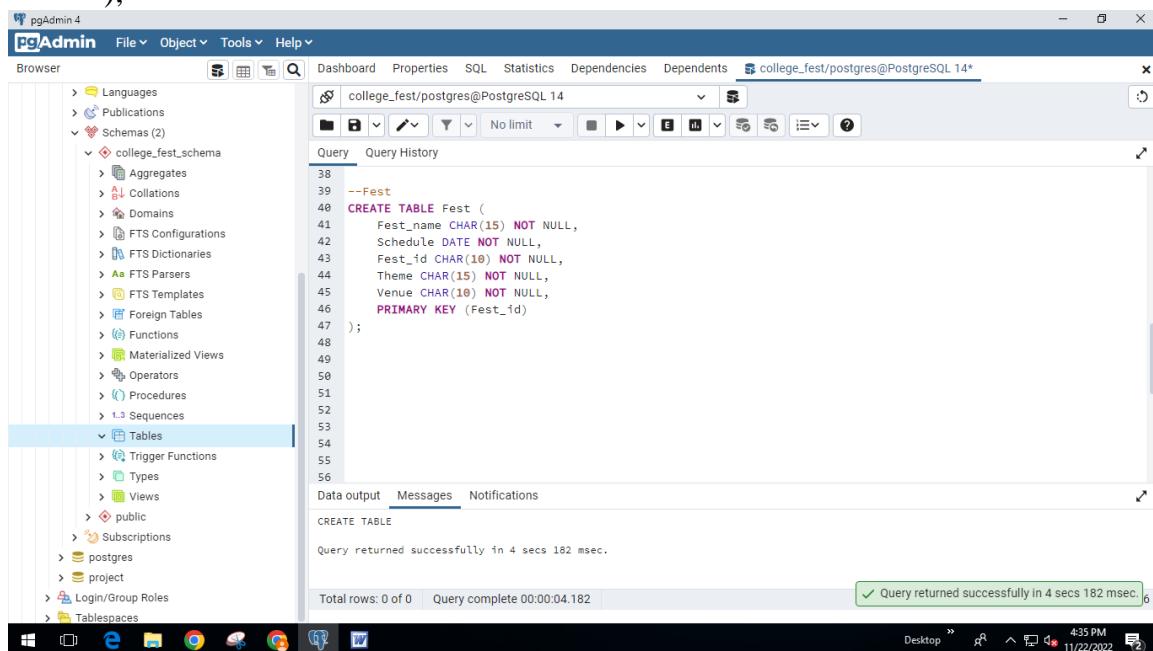
```
--Table 4: Coordinators
CREATE TABLE college_fest_db.coordinators (
    student_id CHAR(10) NOT NULL,
    student_name CHAR(50) NOT NULL,
    student_role CHAR(10) NOT NULL,
    PRIMARY KEY (student_id)
);
```

The 'Messages' tab in the bottom panel shows the message: "Query returned successfully in 75 msec." The status bar at the bottom right indicates "Ln 44, Col 26".

5. Table: Fest

Query:

```
CREATE TABLE college_fest_db.fest (
    fest_name CHAR(15) NOT NULL,
    schedule DATE NOT NULL,
    fest_id CHAR(10) NOT NULL,
    theme CHAR(15) NOT NULL,
    venue CHAR(10) NOT NULL,
    PRIMARY KEY (fest_id)
);
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure under the 'college_fest_schema' node, with the 'Tables' node selected. The main pane shows the SQL query for creating the 'Fest' table, and the status bar at the bottom indicates the query was successful.

```
38
39 --Fest
40 CREATE TABLE Fest (
41     Fest_name CHAR(15) NOT NULL,
42     Schedule DATE NOT NULL,
43     Fest_id CHAR(10) NOT NULL,
44     Theme CHAR(15) NOT NULL,
45     Venue CHAR(10) NOT NULL,
46     PRIMARY KEY (Fest_id)
47 );
48
49
50
51
52
53
54
55
56
```

CREATE TABLE

Query returned successfully in 4 secs 182 msec.

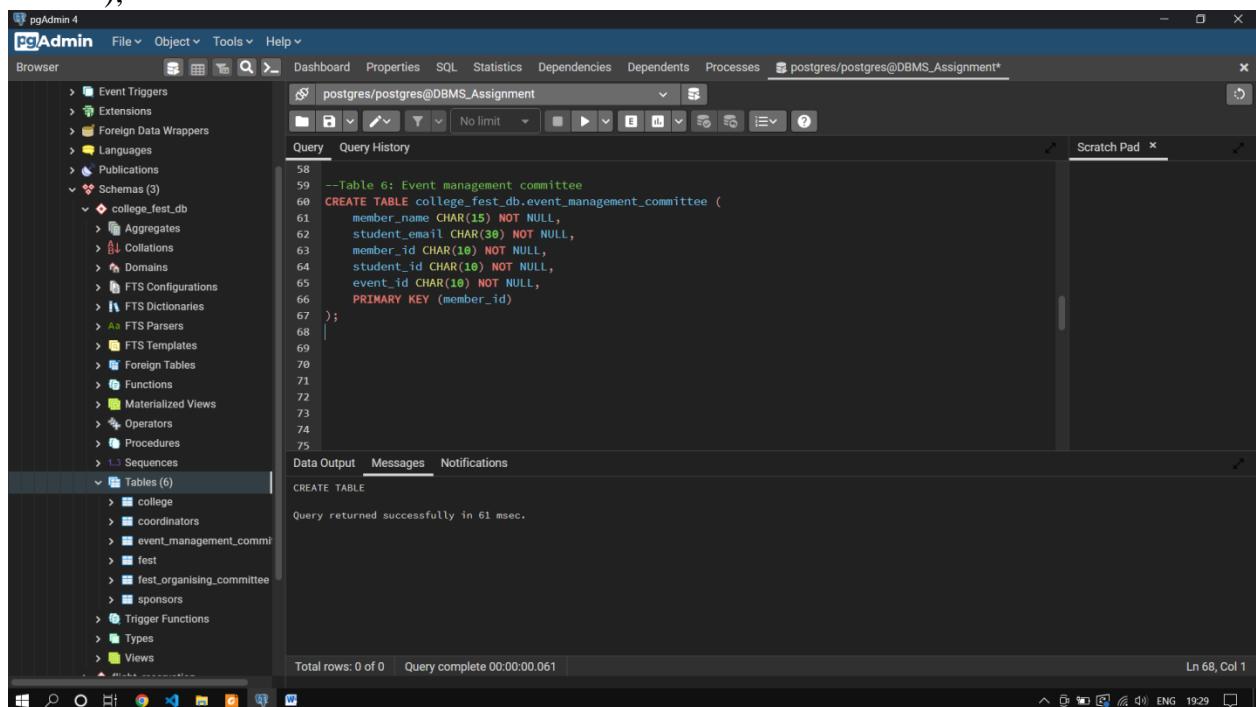
Total rows: 0 of 0 Query complete 00:00:04.182

✓ Query returned successfully in 4 secs 182 msec. 6

6. Table: Event management committee

Query:

```
CREATE TABLE college_fest_db.event_management_committee (
    member_name CHAR(15) NOT NULL,
    student_email CHAR(30) NOT NULL,
    member_id CHAR(10) NOT NULL,
    student_id CHAR(10) NOT NULL,
    event_id CHAR(10) NOT NULL,
    PRIMARY KEY (member_id)
);
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including Schemas, Tables (6), and various system objects. The main pane shows the SQL query being run:

```
58 --Table 6: Event management committee
59 CREATE TABLE college_fest_db.event_management_committee (
60     member_name CHAR(15) NOT NULL,
61     student_email CHAR(30) NOT NULL,
62     member_id CHAR(10) NOT NULL,
63     student_id CHAR(10) NOT NULL,
64     event_id CHAR(10) NOT NULL,
65     PRIMARY KEY (member_id)
66 );
67
68
69
70
71
72
73
74
75
```

The 'Messages' tab shows the successful creation of the table:

CREATE TABLE

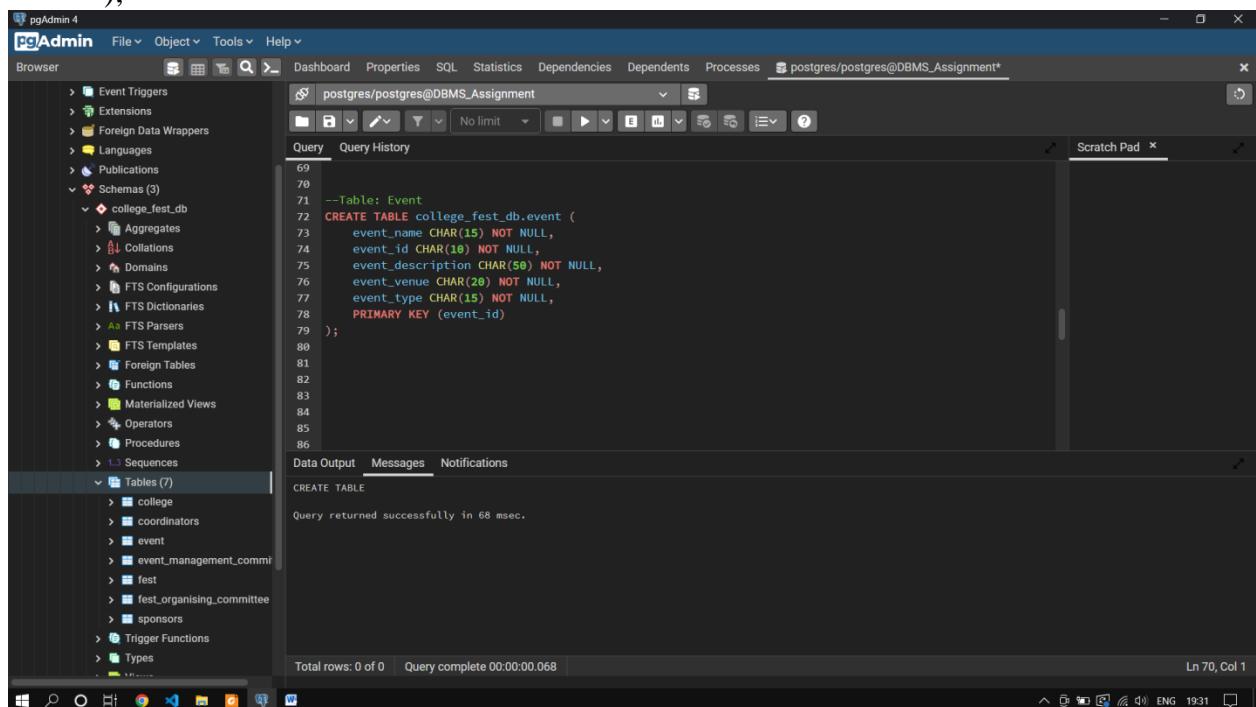
Query returned successfully in 61 msec.

Total rows: 0 of 0 | Query complete 00:00:00.061 | Ln 68, Col 1

7. Table: Event

Query:

```
CREATE TABLE college_fest_db.event (
    event_name CHAR(15) NOT NULL,
    event_id CHAR(10) NOT NULL,
    event_description CHAR(50) NOT NULL,
    event_venue CHAR(20) NOT NULL,
    event_type CHAR(15) NOT NULL,
    PRIMARY KEY (event_id)
);
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure, including Schemas (college_fest_db), Tables (7), and other objects like college, coordinators, event, etc. The main window (Query) shows the SQL code for creating the 'event' table. The code is as follows:

```
69
70  --Table: Event
71  CREATE TABLE college_fest_db.event (
72      event_name CHAR(15) NOT NULL,
73      event_id CHAR(10) NOT NULL,
74      event_description CHAR(50) NOT NULL,
75      event_venue CHAR(20) NOT NULL,
76      event_type CHAR(15) NOT NULL,
77      PRIMARY KEY (event_id)
78  );
79
80
81
82
83
84
85
86
```

The 'Messages' tab shows the successful creation of the table:

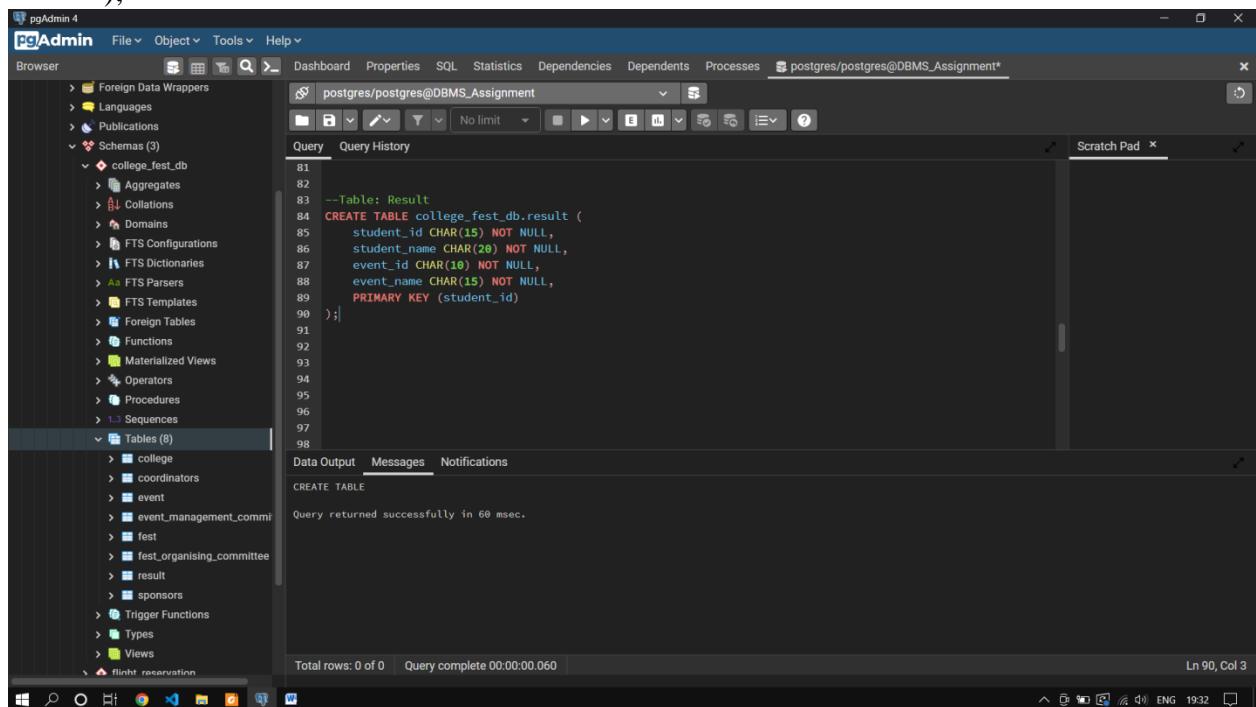
```
CREATE TABLE
Query returned successfully in 68 msec.
```

At the bottom, it shows 'Total rows: 0 of 0' and 'Query complete 00:00:00.068'.

8. Table: Result

Query:

```
CREATE TABLE college_fest_db.result (
    student_id CHAR(15) NOT NULL,
    student_name CHAR(20) NOT NULL,
    event_id CHAR(10) NOT NULL,
    event_name CHAR(15) NOT NULL,
    PRIMARY KEY (student_id)
);
```



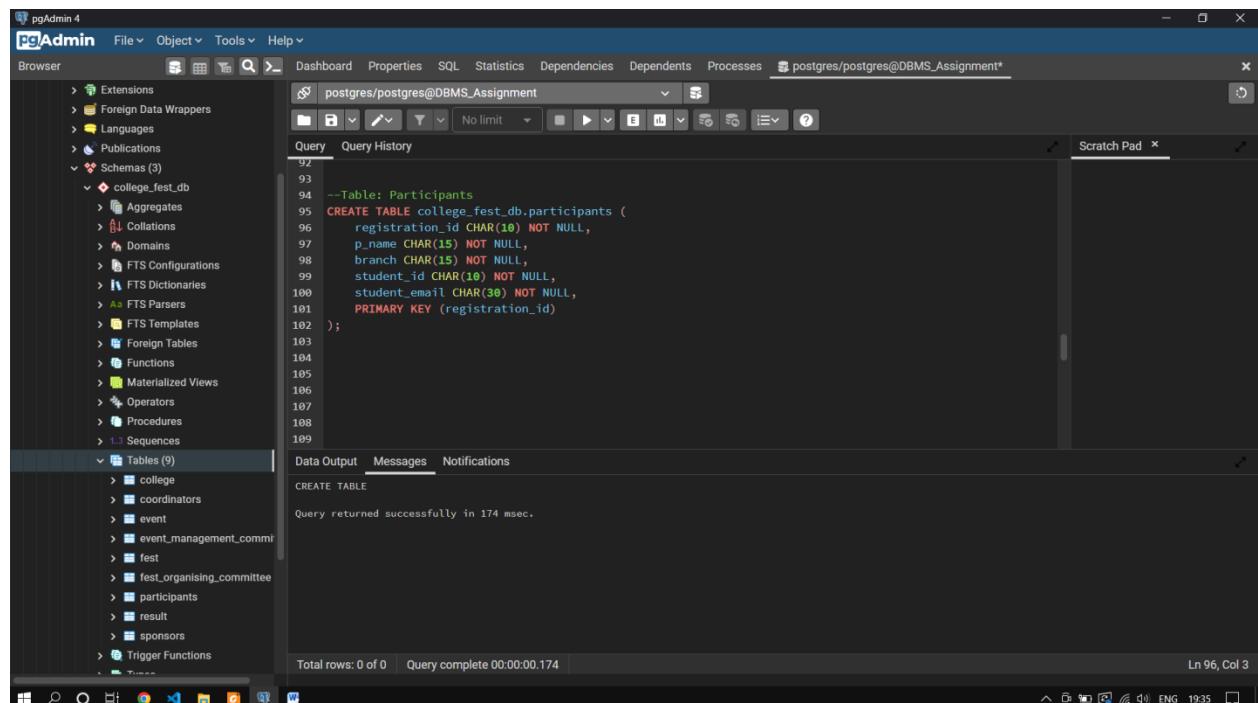
The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure, including the 'college_fest_db' schema which contains tables like 'college', 'coordinators', 'event', etc. The main area (Query) shows the SQL code for creating the 'result' table. The 'Messages' tab indicates that the query was executed successfully in 60 msec. The bottom status bar shows 'Query complete 00:00:00.060' and 'Ln 90, Col 3'.

```
--Table: Result
CREATE TABLE college_fest_db.result (
    student_id CHAR(15) NOT NULL,
    student_name CHAR(20) NOT NULL,
    event_id CHAR(10) NOT NULL,
    event_name CHAR(15) NOT NULL,
    PRIMARY KEY (student_id)
);
```

9. Table: Participants

Query:

```
CREATE TABLE college_fest_db.participants (
    registration_id CHAR(10) NOT NULL,
    p_name CHAR(15) NOT NULL,
    branch CHAR(15) NOT NULL,
    student_id CHAR(10) NOT NULL,
    student_email CHAR(30) NOT NULL,
    PRIMARY KEY (registration_id)
);
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure, including the 'college_fest_db' schema which contains 9 tables: college, coordinators, event, event_management_committee, fest, fest_organising_committee, participants, result, and sponsors. The main area (Query) shows the SQL code for creating the 'participants' table. The code is as follows:

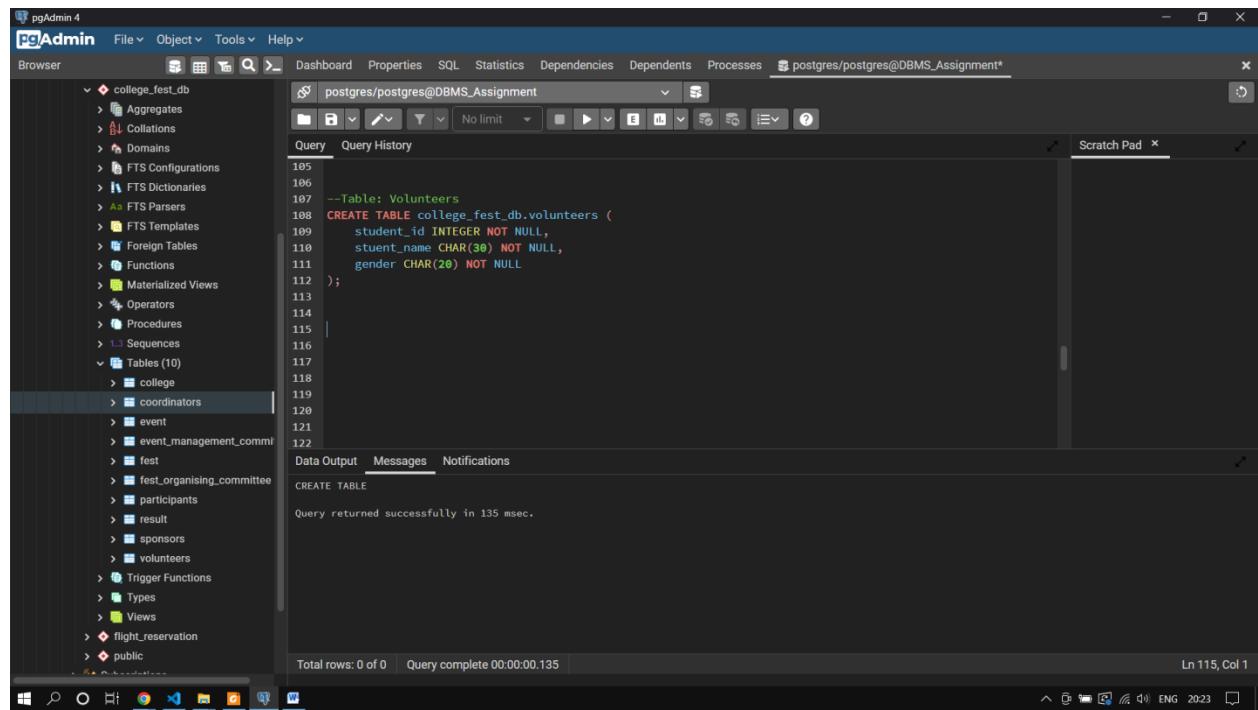
```
92
93 --Table: Participants
94 CREATE TABLE college_fest_db.participants (
95     registration_id CHAR(10) NOT NULL,
96     p_name CHAR(15) NOT NULL,
97     branch CHAR(15) NOT NULL,
98     student_id CHAR(10) NOT NULL,
99     student_email CHAR(30) NOT NULL,
100    PRIMARY KEY (registration_id)
101 );
102
103
104
105
106
107
108
109
```

The 'Messages' tab shows the message: "Query returned successfully in 174 msec." and "Total rows: 0 of 0". The status bar at the bottom right indicates "Ln 96, Col 3".

10. Table: Volunteers

Query:

```
CREATE TABLE college_fest_db.volunteers (
    student_id INTEGER NOT NULL,
    stucent_name CHAR(30) NOT NULL,
    gender CHAR(20) NOT NULL
);
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure under 'college_fest_db', including 'Tables (10)' which contains 'college', 'coordinators', 'event', 'event_management_committee', 'fest', 'fest_organising_committee', 'participants', 'result', 'sponsors', and 'volunteers'. The main window shows a query editor with the following SQL code:

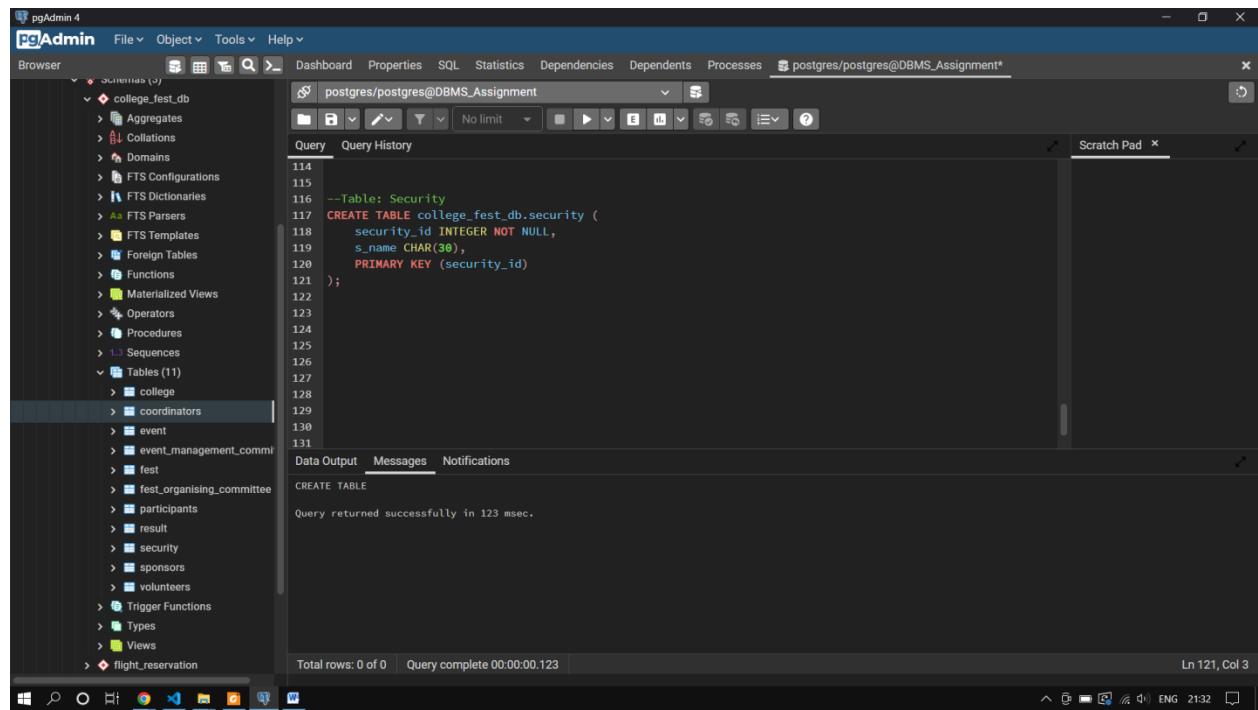
```
105
106
107 --Table: Volunteers
108 CREATE TABLE college_fest_db.volunteers (
109     student_id INTEGER NOT NULL,
110     stucent_name CHAR(30) NOT NULL,
111     gender CHAR(20) NOT NULL
112 );
113
114
115
116
117
118
119
120
121
122
```

Below the code, the 'Messages' tab is selected, showing the message: 'Query returned successfully in 135 msec.' The status bar at the bottom indicates 'Total rows: 0 of 0' and 'Query complete 00:00:00.135'.

11. Security

Query:

```
CREATE TABLE college_fest_db.security (
    security_id INTEGER NOT NULL,
    s_name CHAR(30),
    PRIMARY KEY (security_id)
);
```



pgAdmin 4

File Object Tools Help

Browser Schemas (1) college_fest_db Aggregates Collations Domains FTS Configurations FTS Dictionaries FTS Parsers FTS Templates Foreign Tables Functions Materialized Views Operators Procedures Sequences Tables (11) college coordinators event event_management_committee fest fest_organising_committee participants result security sponsors volunteers Trigger Functions Types Views flight_reservation

Query History

```
114
115
116 --Table: Security
117 CREATE TABLE college_fest_db.security (
118     security_id INTEGER NOT NULL,
119     s_name CHAR(30),
120     PRIMARY KEY (security_id)
121 );
122
123
124
125
126
127
128
129
130
131
```

Data Output Messages Notifications

CREATE TABLE

Query returned successfully in 123 msec.

Total rows: 0 of 0 Query complete 00:00:00.123 Ln 121, Col 3

List of English queries:

1. Display the President from the Co-ordinators.
2. Display the Vice President from the Co-ordinators.
3. List all the coordinators whose names start with 'A'.
4. Select all the designers.
5. Select all the Web Developers who are from MSc IT course.
6. Select all the designers who are female.
7. Select personal information of students who have the role of Tresurer and are from MSc IT programme.
8. Count the total number of participating colleges.
9. Count the total number of participants from each branch.
10. Display the following details
 - a. Student id
 - b. Student name
 - c. Branch
 - d. Event name
 - e. PositionOf participants who secured first position.
11. Display name and venue of all the fests that are between dates 11-11-2022 and 13-11-2022.
12. Display the count of Male and Female volunteers.
13. Display all the sponsors.
14. Display sponsors starting with S.
15. Select the branch with total number of participations in ascending order of count.
16. Select the participant names and their email address.
17. Select participants' names who secured first position from the top 3 events in the Event table.
18. Select volunteers with names starting with B.
19. Display the following details of volunteers from MSc IT programme
 - a. Student ID
 - b. Student name
 - c. Programme
 - d. Designation
20. Select all the Security members.
21. Display total count of distinct Student roles.
22. Count the total number of Second positions.
23. Display all the Event Name and Event Type which are hosted in CEP Block.
24. Select all the Student IDs of Designers.
25. Display the count of participating colleges whose name starts with A.
26. Display the following participants details
 - a. Registration ID
 - b. Name
 - c. Branch
 - d. Student ID
 - e. Email Address
27. Display the Member Names from the Management Committee of Literature event.

28. Select the following details of Fest Organising Committee members

- a. Student ID
- b. Student Name
- c. Designation

Where the Programme is Btech.

29. Display the following details for Student IDs: 20221220, 20221224, 20221228

- a. Student Name
- b. Event Name
- c. Position

30. Select names of Male volunteers.

31. Select the following details from the Fest Organising Committee

- a. Member name
- b. Programme
- c. Designation

And sort them in ascending order w.r.t member name.

32. Display all the participating Medical Sciences colleges.

33. Display the names of all the students who secured first position.

34. Display the name and email address of participants from Btech.

35. Select the following details

- a. Student ID
- b. Student name
- c. Event name

Who are from MSc branch and secured first position

36. Display the total number of events hosted on date 12-11-2022.

37. Display the number of each event type.

38. Select the following details

- a. Fest name
- b. Venue
- c. Schedule

Where fest theme is Retro.

39. Select the following details

- a. Event name
- b. Member name

For events that were hosted on Open Air Theatre.

40. Display the following details

- a. Event venue
- b. Event name
- c. Event type

Of event that are of type

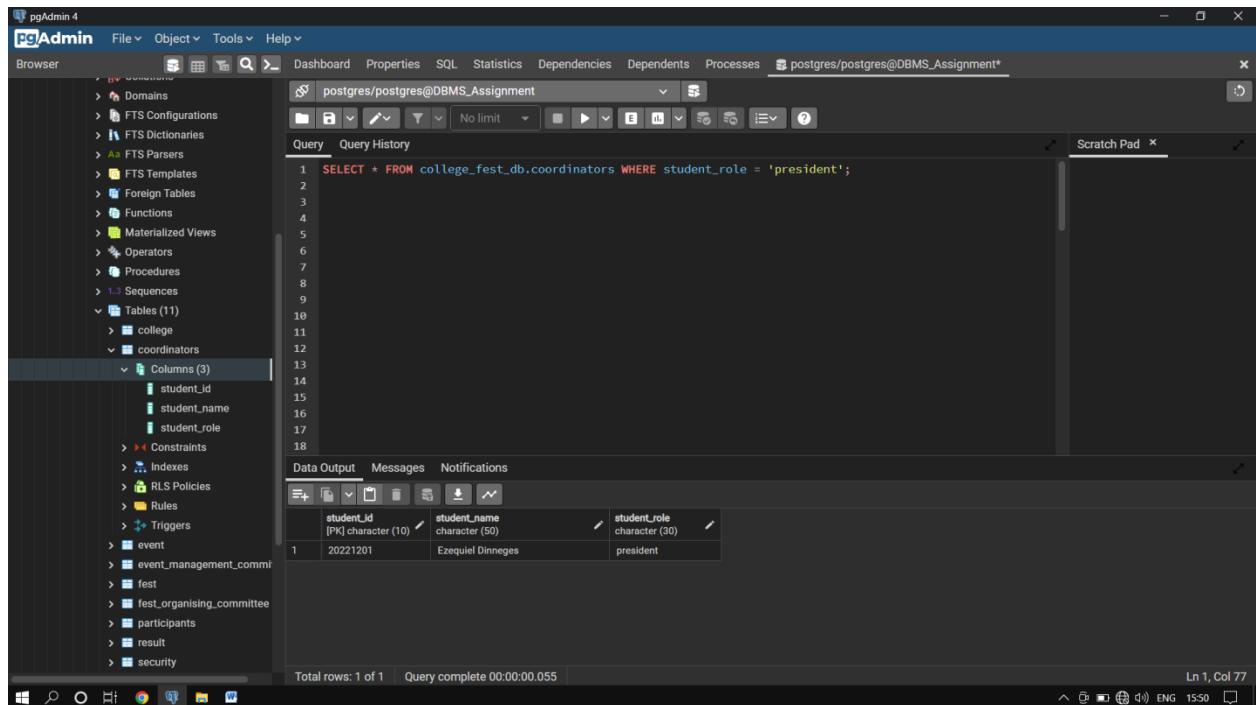
- a. Fun
- b. OAT Event
- c. Outdoor games

List of SQL queries:

1. English query: To display the President from the Co-ordinators.

Query:

```
SELECT * FROM college_fest_db.coordinators WHERE student_role = 'president';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays a tree view of database objects, including 'Domains', 'FTS Configurations', 'FTS Dictionaries', 'FTS Parsers', 'FTS Templates', 'Foreign Tables', 'Functions', 'Materialized Views', 'Operators', 'Procedures', 'Sequences', and 'Tables (11)'. Under 'Tables (11)', 'coordinators' is selected, showing 'Columns (3)': student_id, student_name, and student_role. The main query editor window contains the following SQL code:

```
1  SELECT * FROM college_fest_db.coordinators WHERE student_role = 'president';
```

The results pane shows a table with the following data:

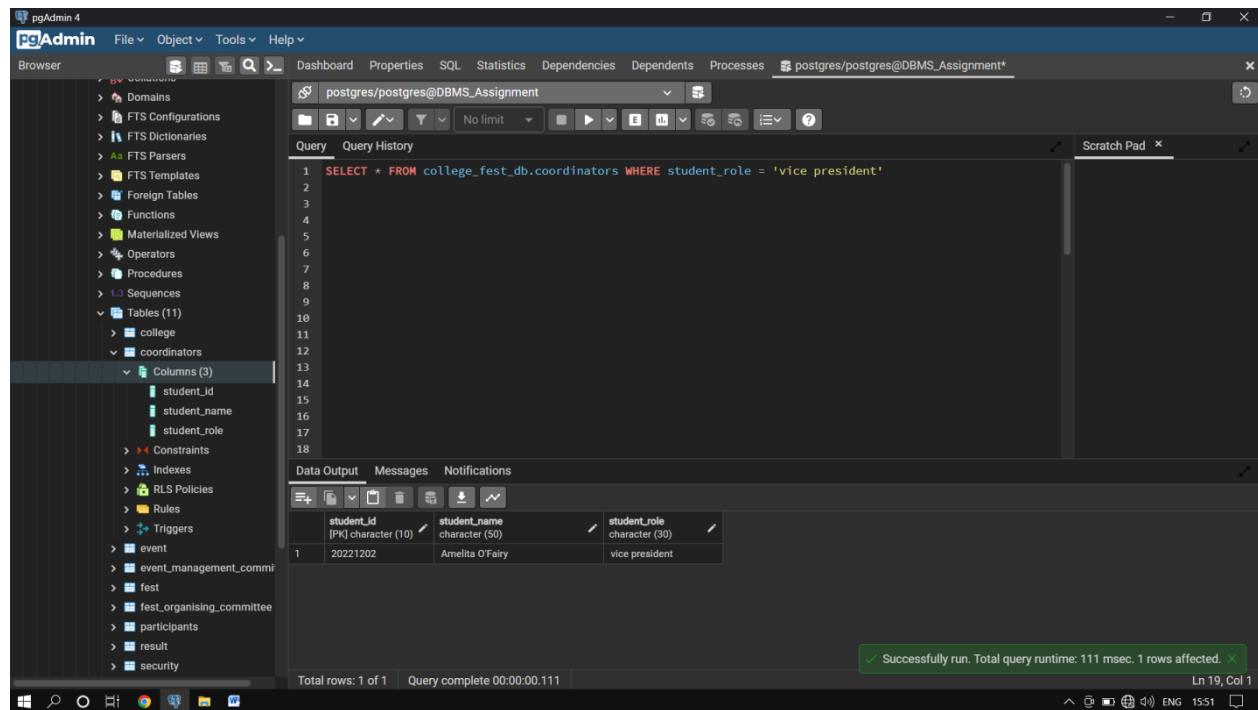
	student_id	student_name	student_role
1	20221201	Ezequiel Dinnege	president

Below the table, the status bar indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.055'.

2. English query: To display the Vice President from the Co-ordinators.

Query:

```
SELECT * FROM college_fest_db.coordinators WHERE student_role = 'vice
president';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) shows the database structure with 'Tables (11)' expanded, showing 'college' and 'coordinators'. The 'coordinators' table has 3 columns: 'student_id', 'student_name', and 'student_role'. The main query editor window contains the following SQL code:

```
1 SELECT * FROM college_fest_db.coordinators WHERE student_role = 'vice president'
```

The results pane shows a single row of data:

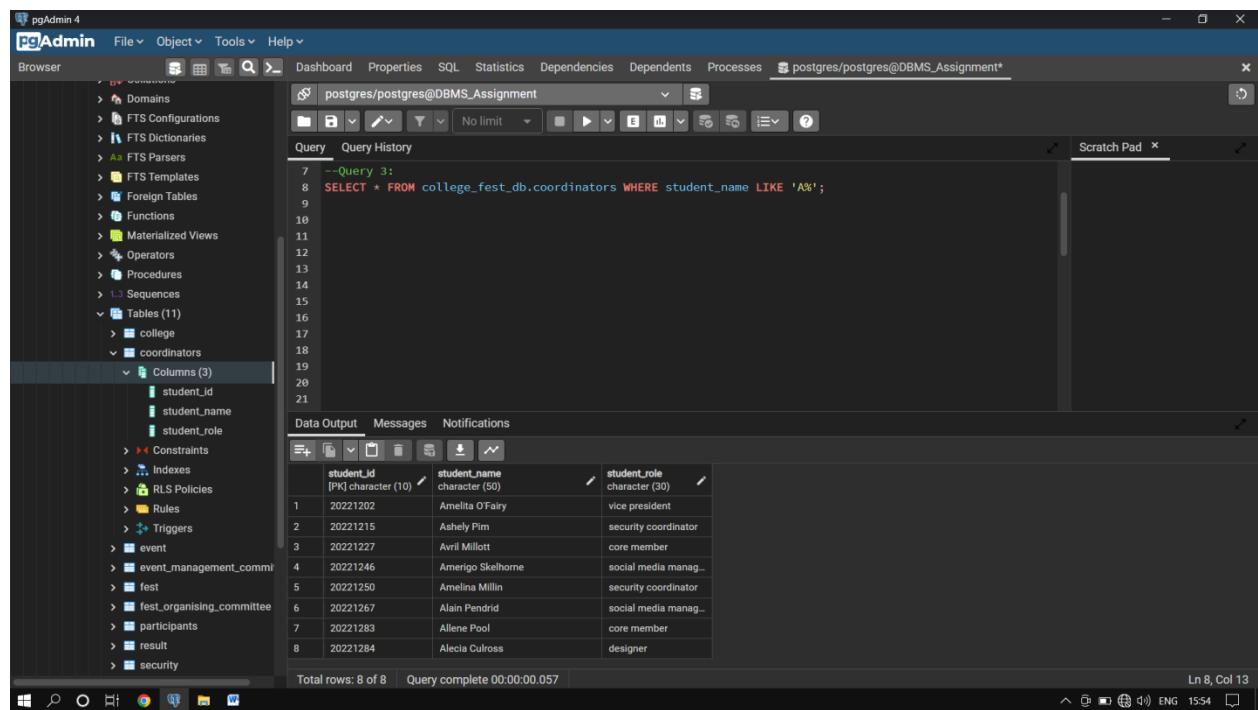
student_id	student_name	student_role
20221202	Amelita O'Fairy	vice president

Below the results, a message box indicates: 'Successfully run. Total query runtime: 111 msec. 1 rows affected.' The status bar at the bottom right shows 'Ln 19, Col 1'.

3. English query: List all the o-ordinators whose names start with 'A'.

Query:

```
SELECT * FROM college_fest_db.coordinators WHERE student_name LIKE 'A%';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) shows the database structure with 'Tables (11)' expanded, showing 'college' and 'coordinators'. The 'coordinators' table has 3 columns: 'student_id', 'student_name', and 'student_role'. The main query editor window contains the following SQL code:

```
7 --Query 3:
8 SELECT * FROM college_fest_db.coordinators WHERE student_name LIKE 'A%';
```

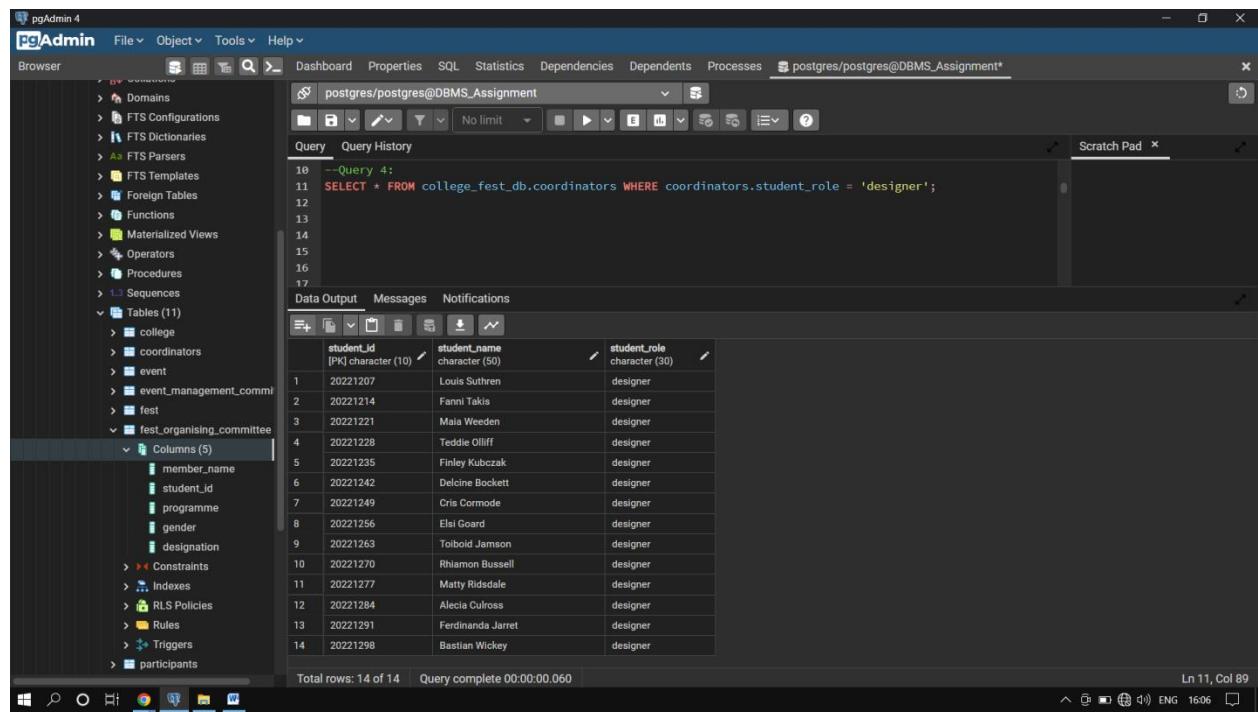
The results pane shows 8 rows of data:

student_id	student_name	student_role
20221202	Amelita O'Fairy	vice president
20221215	Ashely Prim	security coordinator
20221227	Avril Millott	core member
20221246	Amerigo Stelhorne	social media manag...
20221250	Amelia Millin	security coordinator
20221267	Alain Pendrid	social media manag...
20221283	Allene Pool	core member
20221284	Alecia Culross	designer

Below the results, a message box indicates: 'Total rows: 8 of 8 Query complete 00:00:00.057'. The status bar at the bottom right shows 'Ln 8, Col 13'.

4. English query: Select all the designers.

Query: `SELECT * FROM college_fest_db.coordinators WHERE coordinators.student_role = 'designer';`

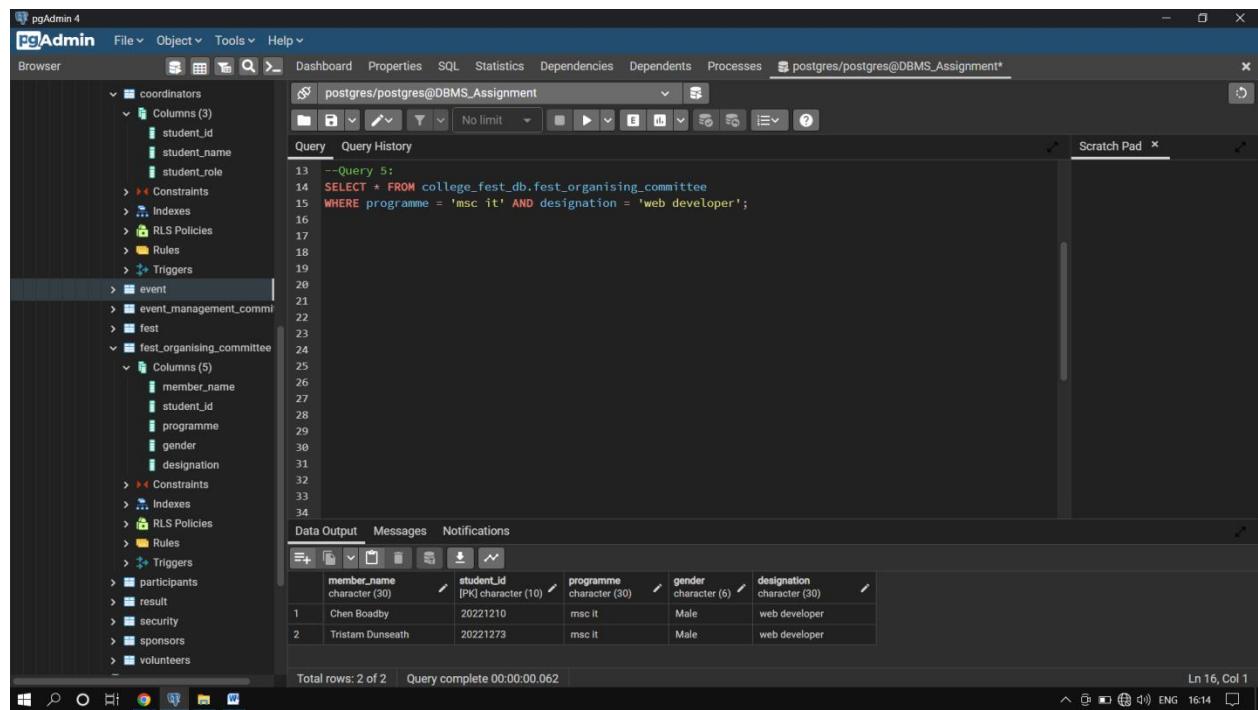


```
--Query 4:
SELECT * FROM college_fest_db.coordinators WHERE coordinators.student_role = 'designer';

student_id      student_name      student_role
20221207      Louis Suthren      designer
20221214      Fanni Takis      designer
20221221      Maia Weeden      designer
20221228      Teddie Olliff      designer
20221235      Finley Kubczak      designer
20221242      Delcine Bockett      designer
20221249      Cris Cormode      designer
20221256      Elsa Goard      designer
20221263      Tolbold Jamson      designer
20221270      Rhiamon Bussell      designer
20221277      Matty Ridsdale      designer
20221284      Alecia Culross      designer
20221291      Ferdinand Jarret      designer
20221298      Bastian Wickey      designer
```

5. English query: Select all the Web Developers who are from MSc IT course.

Query: `SELECT * FROM college_fest_db.fest_organising_committee WHERE programme = 'msc it' AND designation = 'web developer';`



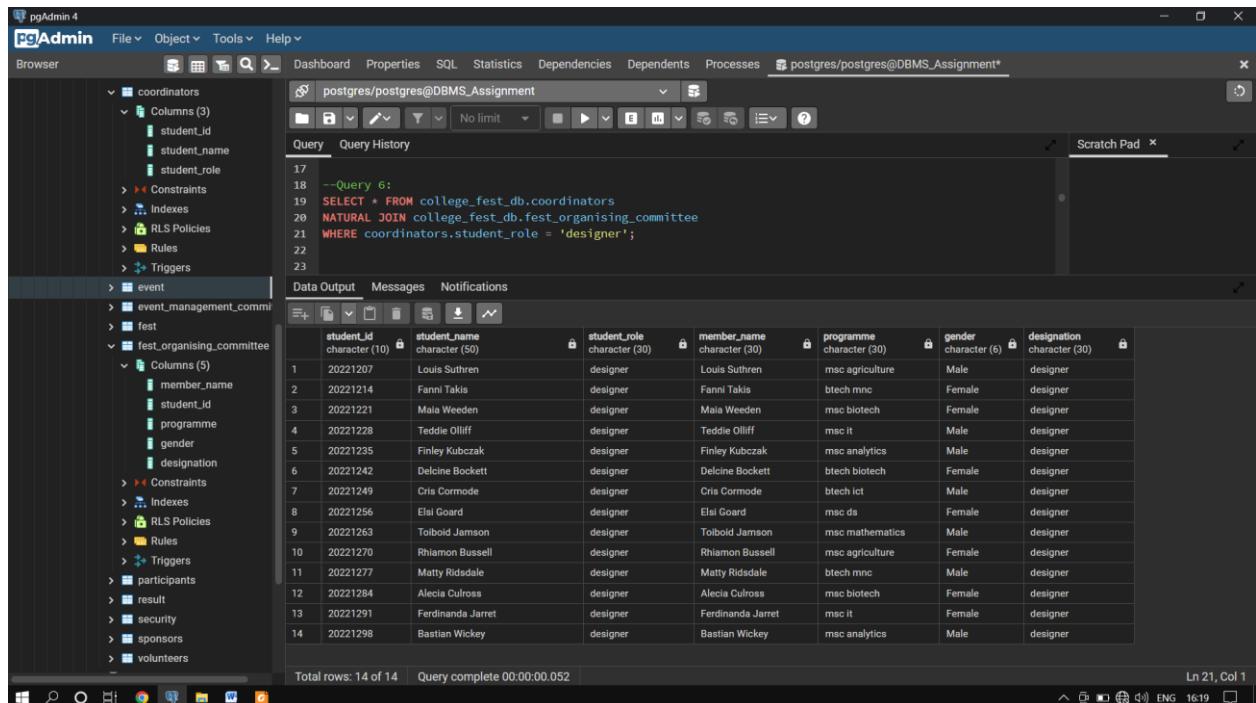
```
--Query 5:
SELECT * FROM college_fest_db.fest_organising_committee
WHERE programme = 'msc it' AND designation = 'web developer';

member_name      student_id      programme      gender      designation
Chen Boatby      20221210      msc it      Male      web developer
Tristan Dunseath 20221273      msc it      Male      web developer
```

6. English query: Select all the designers who are female.

Query:

```
SELECT * FROM college_fest_db.coordinators NATURAL JOIN
college_fest_db.fest_organising_committee WHERE coordinators.student_role =
'designer';
```



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

- Browser:** Shows the database schema with tables: coordinators, event, event_management_committee, fest, fest_organising_committee, result, security, sponsors, and volunteers.
- Query Editor:** The query is displayed in the Query tab:

```
--Query 6:
SELECT * FROM college_fest_db.coordinators
NATURAL JOIN college_fest_db.fest_organising_committee
WHERE coordinators.student_role = 'designer';
```
- Data Output:** The results of the query are shown in a table with 14 rows. The columns are: student_id, student_name, student_role, member_name, programme, gender, and designation.
- Table Data:**

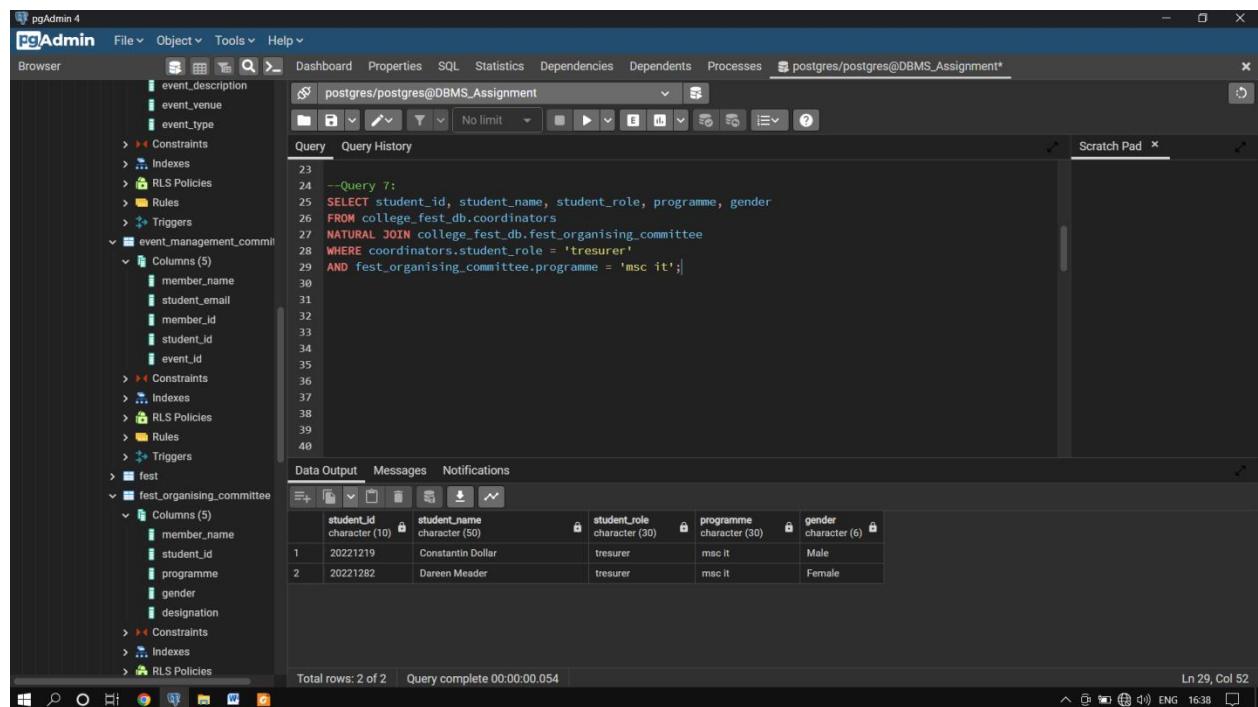
student_id	student_name	student_role	member_name	programme	gender	designation
20221207	Louis Suthren	designer	Louis Suthren	msc agriculture	Male	designer
20221214	Fanni Takis	designer	Fanni Takis	bttech mnc	Female	designer
20221221	Mala Weeden	designer	Mala Weeden	msc biotech	Female	designer
20221228	Teddie Olliff	designer	Teddie Olliff	msc it	Male	designer
20221235	Finley Kubczak	designer	Finley Kubczak	msc analytics	Male	designer
20221242	Delcine Bockett	designer	Delcine Bockett	bttech biotech	Female	designer
20221249	Cris Cormode	designer	Cris Cormode	bttech iot	Male	designer
20221256	Eli Goard	designer	Eli Goard	msc ds	Female	designer
20221263	Toiboid Jamson	designer	Toiboid Jamson	msc mathematics	Male	designer
20221270	Rhamon Bussell	designer	Rhamon Bussell	msc agriculture	Female	designer
20221277	Matty Riddsdale	designer	Matty Riddsdale	bttech mnc	Male	designer
20221284	Alecia Culross	designer	Alecia Culross	msc biotech	Female	designer
20221291	Ferdinanda Jarret	designer	Ferdinanda Jarret	msc it	Female	designer
20221298	Bastian Wickey	designer	Bastian Wickey	msc analytics	Male	designer

- Message Bar:** Shows "Total rows: 14 of 14" and "Query complete 00:00:00.052".
- System Bar:** Shows the taskbar with various icons and the system clock.

7. English query: Select personal information of students who have the role of Treasurer and are from MSc IT programme.

Query:

```
SELECT student_id, student_name, student_role, programme, gender
FROM college_fest_db.coordinators
NATURAL JOIN college_fest_db.fest_organising_committee
WHERE coordinators.student_role = 'treasurer'
AND fest_organising_committee.programme = 'msc it';
```



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

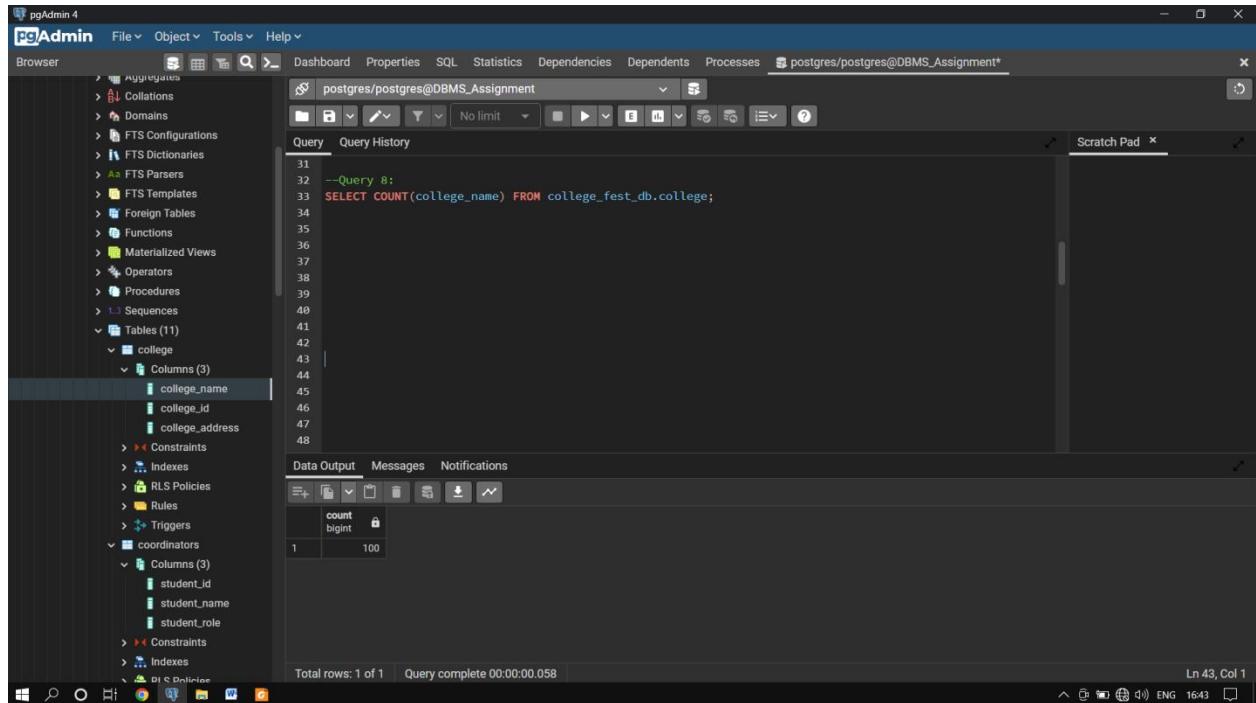
- Toolbar:** File, Object, Tools, Help.
- Browser:** Shows the database structure for the 'DBMS_Assignment' database, including 'event_description', 'event_venue', 'event_type', 'Constraints', 'Indexes', 'RLS Policies', 'Rules', 'Triggers', 'event_management_committee' (with 'Columns' and 'Triggers' listed), 'fest', and 'fest_organising_committee' (with 'Columns' and 'Triggers' listed).
- Query Editor:** The query is pasted into the 'Query' tab:

```
23 --Query 7:
24
25 SELECT student_id, student_name, student_role, programme, gender
26 FROM college_fest_db.coordinators
27 NATURAL JOIN college_fest_db.fest_organising_committee
28 WHERE coordinators.student_role = 'treasurer'
29 AND fest_organising_committee.programme = 'msc it';
```
- Data Output:** The results are displayed in a table:

	student_id	student_name	student_role	programme	gender
1	20221219	Constantin Dollar	treasurer	msc it	Male
2	20221282	Dareen Meader	treasurer	msc it	Female

Total rows: 2 of 2 | Query complete 00:00:00.054 | Ln 29, Col 52

8. English query: Count the total number of participating colleges.
Query: `SELECT COUNT(college_name) FROM college_fest_db.college;`



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure, including a table named 'college' with three columns: 'college_name', 'college_id', and 'college_address'. The main area (Query Editor) shows the following SQL query:

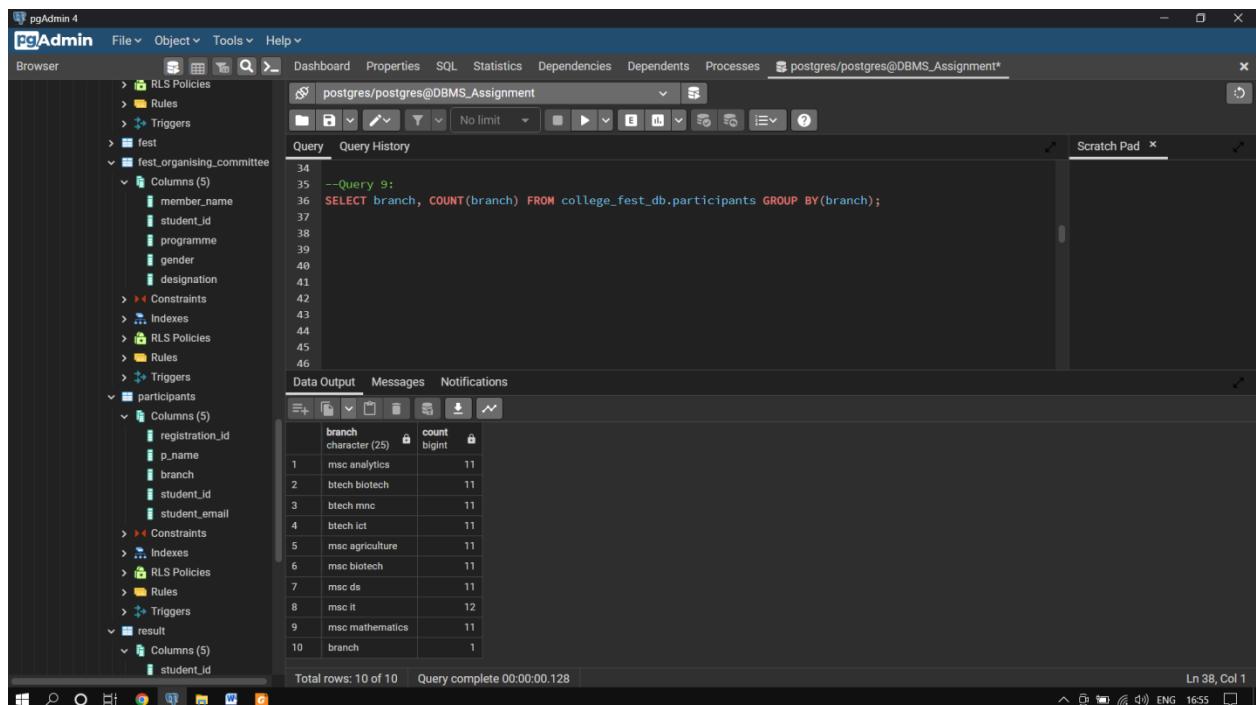
```
--Query 8:  
SELECT COUNT(college_name) FROM college_fest_db.college;
```

The Data Output tab shows the result of the query:

count	bigint
1	100

Total rows: 1 of 1 | Query complete 00:00:00.058 | Ln 43, Col 1

9. English query: Count the total number of participants from each branch.
Query:
`SELECT branch, COUNT(branch) FROM college_fest_db.participants GROUP BY(branch);`



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure, including a table named 'participants' with five columns: 'registration_id', 'p_name', 'branch', 'student_id', and 'student_email'. The main area (Query Editor) shows the following SQL query:

```
--Query 9:  
SELECT branch, COUNT(branch) FROM college_fest_db.participants GROUP BY(branch);
```

The Data Output tab shows the result of the query:

branch	count
msc analytics	11
btech biotech	11
btech mnc	11
btech ict	11
msc agriculture	11
msc biotech	11
msc ds	11
mac it	12
msc mathematics	11
branch	1

Total rows: 10 of 10 | Query complete 00:00:00.128 | Ln 38, Col 1

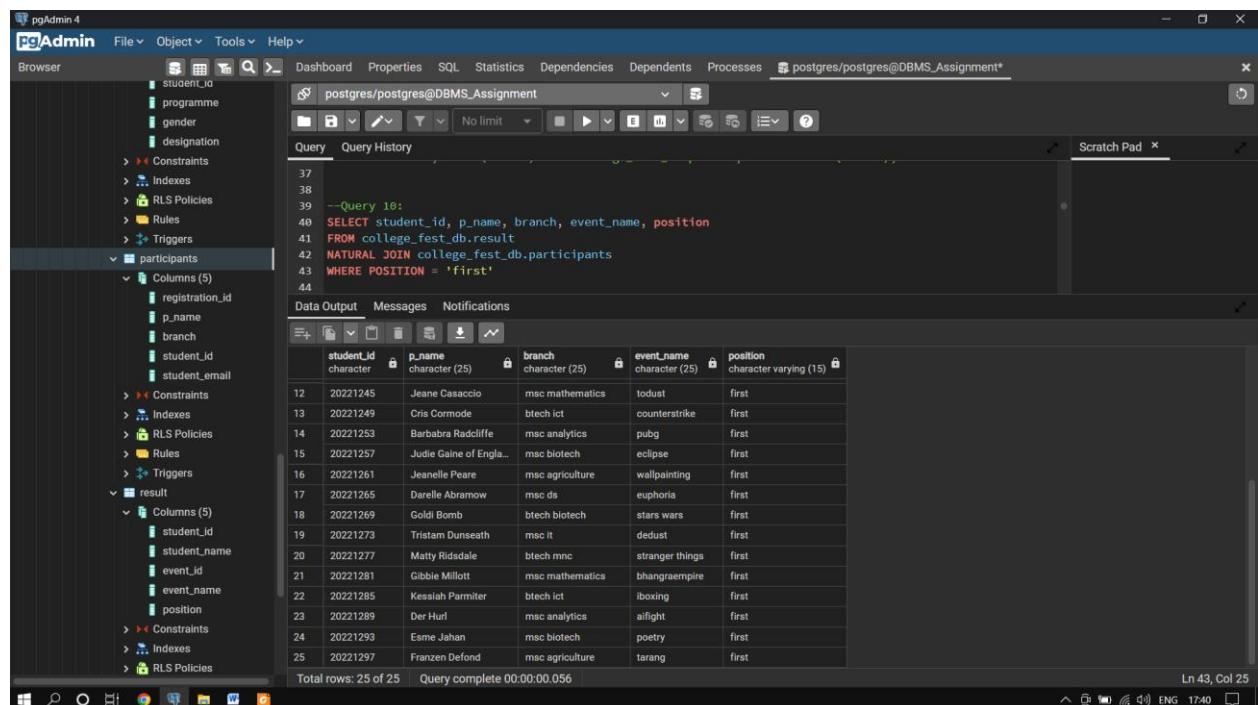
10. English query: Display the following details

- a. Student id
- b. Student name
- c. Branch
- d. Event name
- e. Position

Of participants who secured first position.

Query: `SELECT student_id, p_name, branch, event_name, position
FROM college_fest_db.result`

`NATURAL JOIN college_fest_db.participants
WHERE POSITION = 'first';`

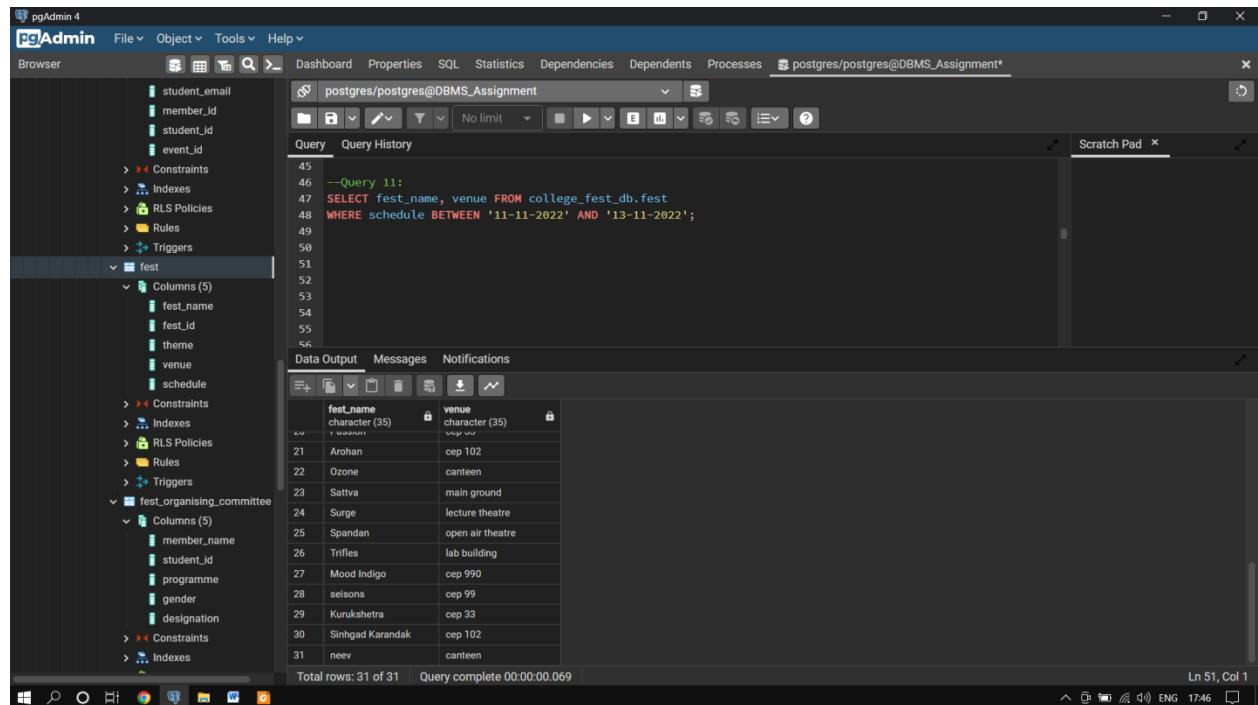


student_id	p_name	branch	event_name	position
12	20221245	Jeane Casaccio	msc mathematics	todust
13	20221249	Cris Cormode	btech ict	counterstrike
14	20221253	Barbara Radcliffe	msc analytics	pubg
15	20221257	Judie Gaine of Engla...	msc biotech	eclipse
16	20221261	Jeanelle Peare	msc agriculture	wallpainting
17	20221265	Darelle Abramow	msc ds	euphoria
18	20221269	Gold Bomb	btech biotech	stars wars
19	20221273	Tristam Dunseath	msc it	dedust
20	20221277	Matty Riddale	btech mnc	stranger things
21	20221281	Gibbie Millott	msc mathematics	bhangraempire
22	20221285	Kessiah Parmiter	btech ict	iboxing
23	20221289	Der Hurl	msc analytics	afight
24	20221293	Esme Jahan	msc biotech	poetry
25	20221297	Franzen Defond	msc agriculture	tarang

11. English query: Display name and venue of all the fests that are between dates 11-11-2022 and 13-11-2022

Query:

```
SELECT * FROM college_fest_db.fest
WHERE schedule BETWEEN '11-11-2022' AND '13-11-2022';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database schema with tables like student_email, member_id, student_id, event_id, fest, fest_organising_committee, and their respective columns and constraints. The central area shows the SQL query being run:

```
45
46 --Query 11:
47 SELECT fest_name, venue FROM college_fest_db.fest
48 WHERE schedule BETWEEN '11-11-2022' AND '13-11-2022';
49
50
51
52
53
54
55
56
```

The results are displayed in a table:

fest_name	venue
Arohan	cep 102
Ozone	canteen
Sattva	main ground
Surge	lecture theatre
Spandan	open air theatre
Trifles	lab building
Mood Indigo	cep 990
seisons	cep 99
Kurukshetra	cep 33
Sinhgad Karandak	cep 102
neev	canteen

Total rows: 31 of 31 | Query complete 00:00:00.069 | Ln 51, Col 1

12. English query: Display the count of Male and Female volunteers

Query:

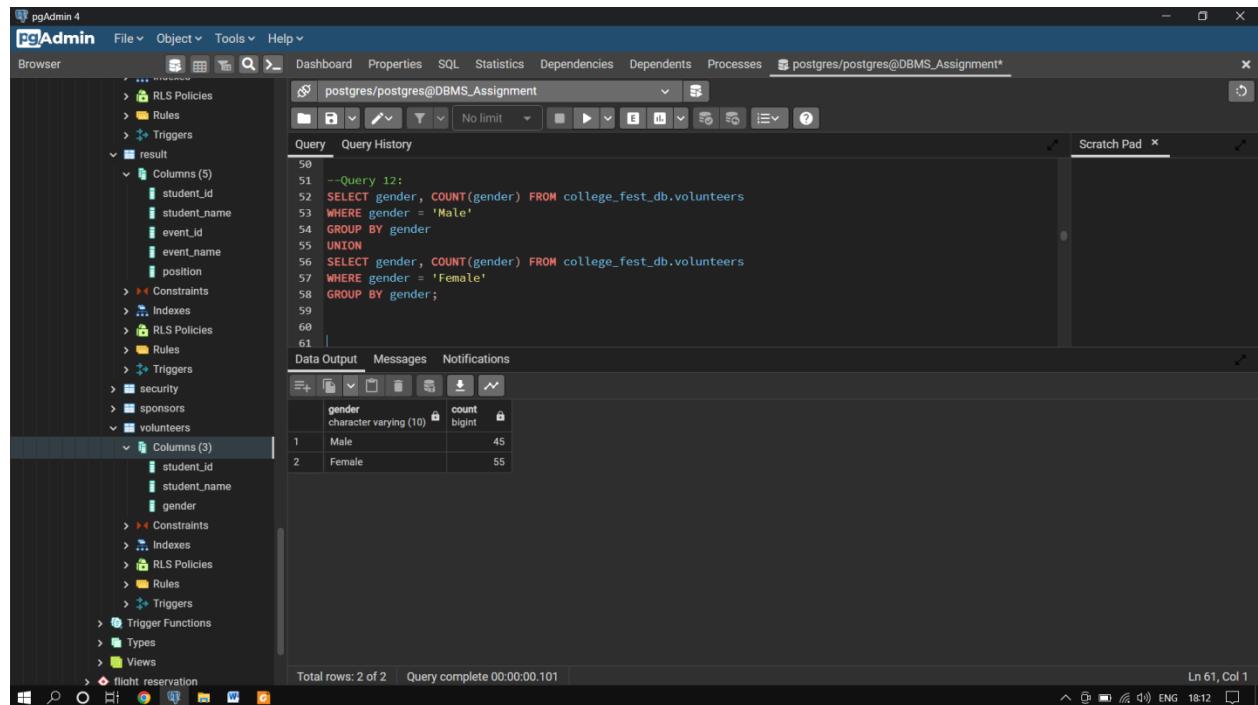
```
SELECT COUNT(gender) FROM college_fest_db.volunteers
```

```
WHERE gender = 'Male'
```

```
UNION
```

```
SELECT COUNT(gender) FROM college_fest_db.volunteers
```

```
WHERE gender = 'Female';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with tables like RLS Policies, Rules, Triggers, and volunteers. The volunteers table is selected, showing columns student_id, student_name, event_id, event_name, and position. The main window shows a query editor with the following SQL code:

```
50
51 --Query 12:
52 SELECT gender, COUNT(gender) FROM college_fest_db.volunteers
53 WHERE gender = 'Male'
54 GROUP BY gender
55 UNION
56 SELECT gender, COUNT(gender) FROM college_fest_db.volunteers
57 WHERE gender = 'Female'
58 GROUP BY gender;
59
60
61
```

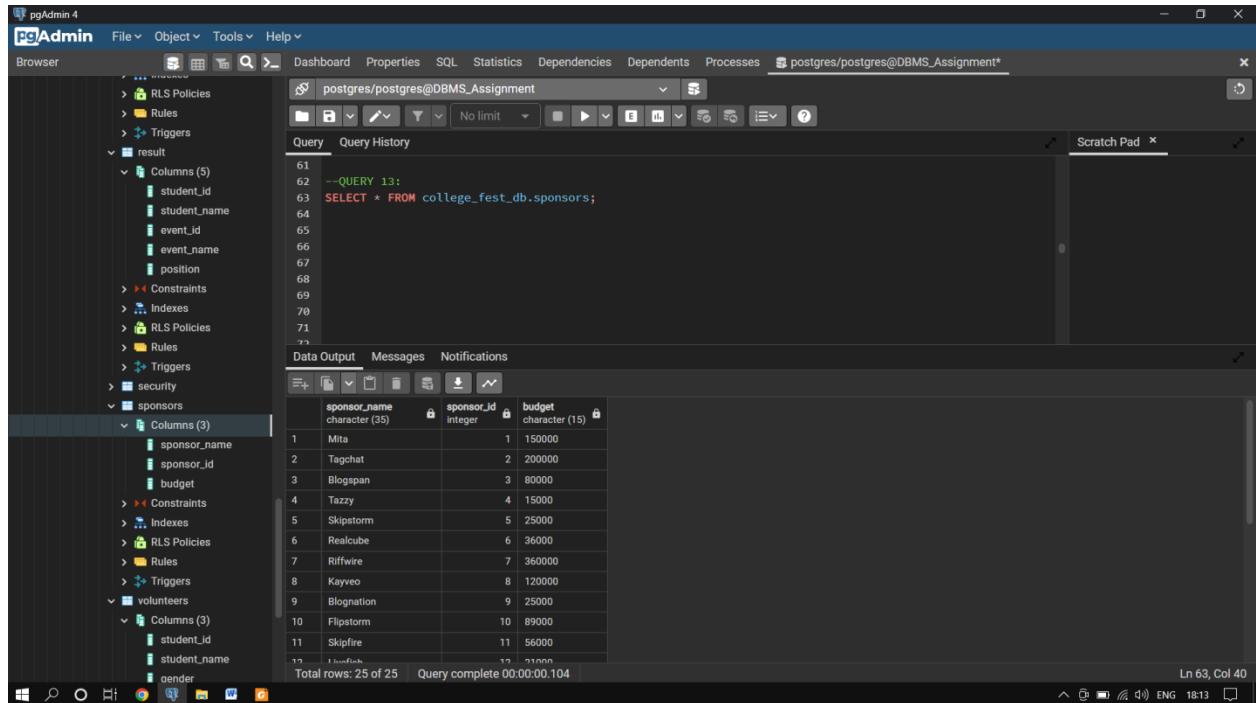
The Data Output tab shows the results of the query:

gender	count
Male	45
Female	55

At the bottom, it says "Total rows: 2 of 2" and "Query complete 00:00:00.101".

13. English query: Display all the sponsors.

Query: `SELECT * FROM college_fest_db.sponsors;`



The screenshot shows the pgAdmin 4 interface with the 'sponsors' table selected in the left sidebar. The 'Data Output' tab is active, displaying the results of the query. The results are as follows:

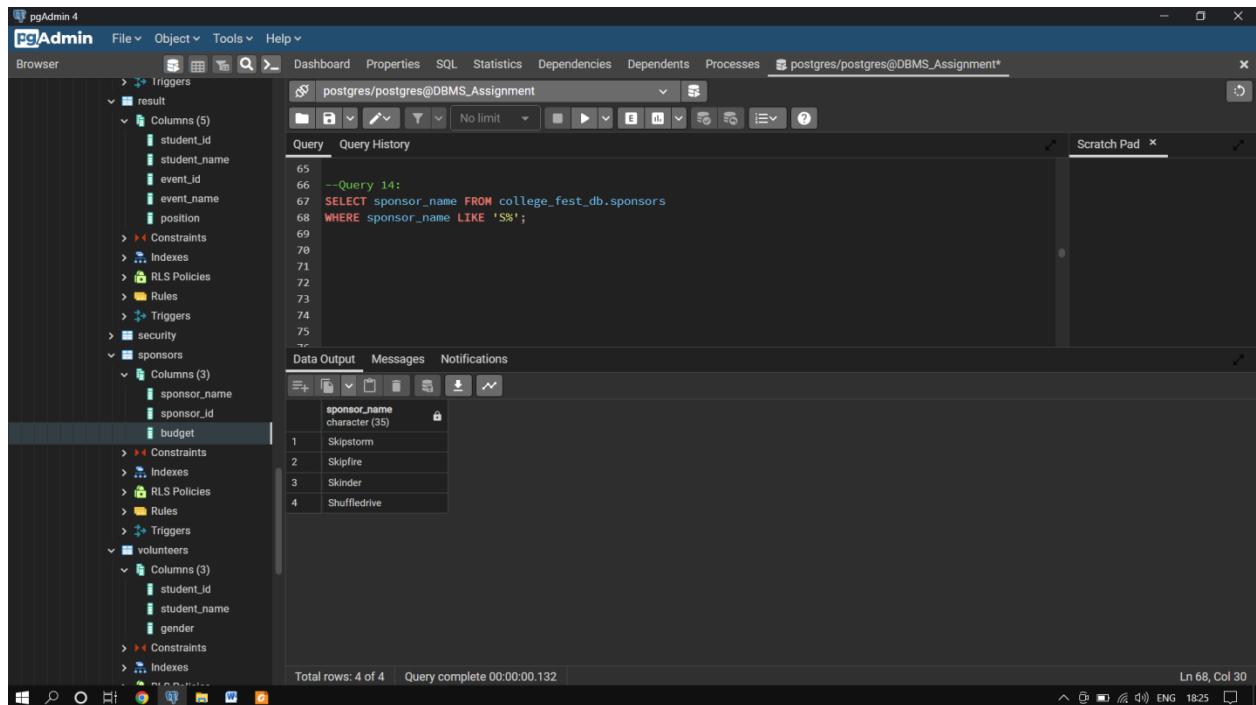
	sponsor_name	sponsor_id	budget
1	Mita	1	150000
2	Tagchat	2	200000
3	Blogspan	3	80000
4	Tazzy	4	15000
5	Skipstorm	5	25000
6	Realcube	6	36000
7	Riffwire	7	360000
8	Kayeo	8	120000
9	Blognation	9	25000
10	Flipstorm	10	89000
11	Skipfire	11	56000
12	Uniflink	12	31000

Total rows: 25 of 25 | Query complete 00:00:00.104 | Ln 63, Col 40

14. English query: Display sponsors starting with S.

Query:

`SELECT sponsor_name FROM college_fest_db.sponsors
WHERE sponsor_name LIKE 'S%';`



The screenshot shows the pgAdmin 4 interface with the 'sponsors' table selected in the left sidebar. The 'Data Output' tab is active, displaying the results of the query. The results are as follows:

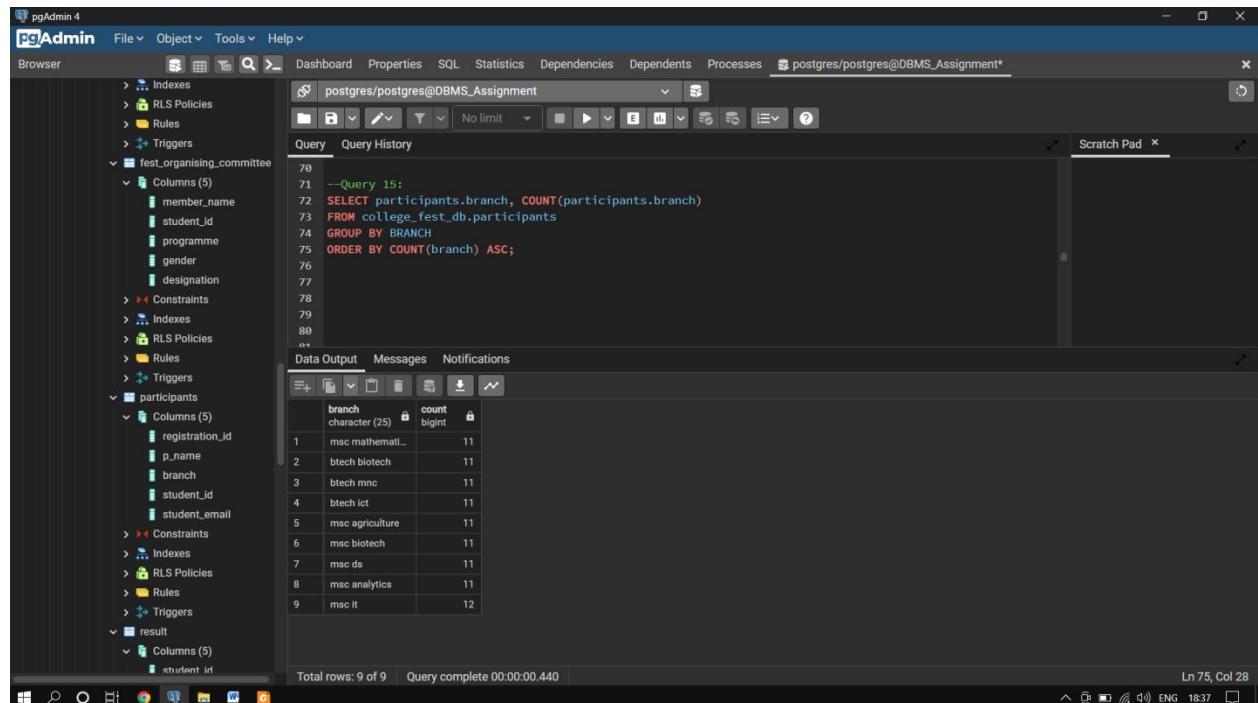
	sponsor_name
1	Skipstorm
2	Skipfire
3	Skinder
4	Shuffledrive

Total rows: 4 of 4 | Query complete 00:00:00.132 | Ln 68, Col 30

15. English query: Select the branch with total number of participations in ascending order of count.

Query:

```
SELECT participants.branch, COUNT(participants.branch)
FROM college_fest_db.participants
GROUP BY BRANCH
ORDER BY COUNT(branch) ASC;
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure for the 'college_fest_db' schema, including the 'fest_organising_committee' and 'participants' tables. The 'participants' table has 5 columns: registration_id, p_name, branch, student_id, and student_email. The 'fest_organising_committee' table has 5 columns: member_name, student_id, programme, gender, and designation. The central pane shows the SQL query being run:

```
--Query 15:
SELECT participants.branch, COUNT(participants.branch)
FROM college_fest_db.participants
GROUP BY BRANCH
ORDER BY COUNT(branch) ASC;
```

The results pane displays the data output, which is a table with two columns: 'branch' and 'count'. The data is as follows:

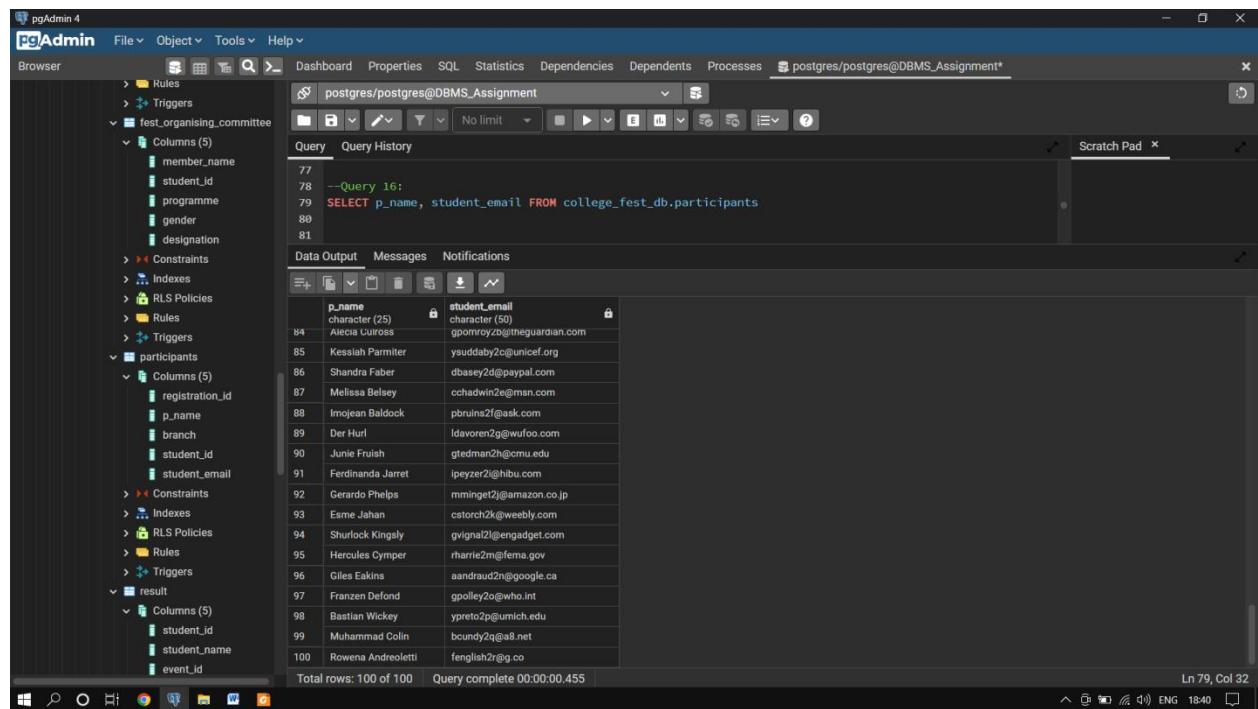
branch	count
msc mathematics	11
btech biotech	11
btech mnc	11
btech ict	11
msc agriculture	11
msc biotech	11
msc ds	11
msc analytics	11
msc it	12

Below the table, the status bar indicates 'Total rows: 9 of 9' and 'Query complete 00:00:00.440'. The bottom right corner shows the system status with 'Ln 75, Col 28' and other system icons.

16. English query: Select the participant names and their email address.

Query:

```
SELECT p_name, student_email FROM college_fest_db.participants;
```



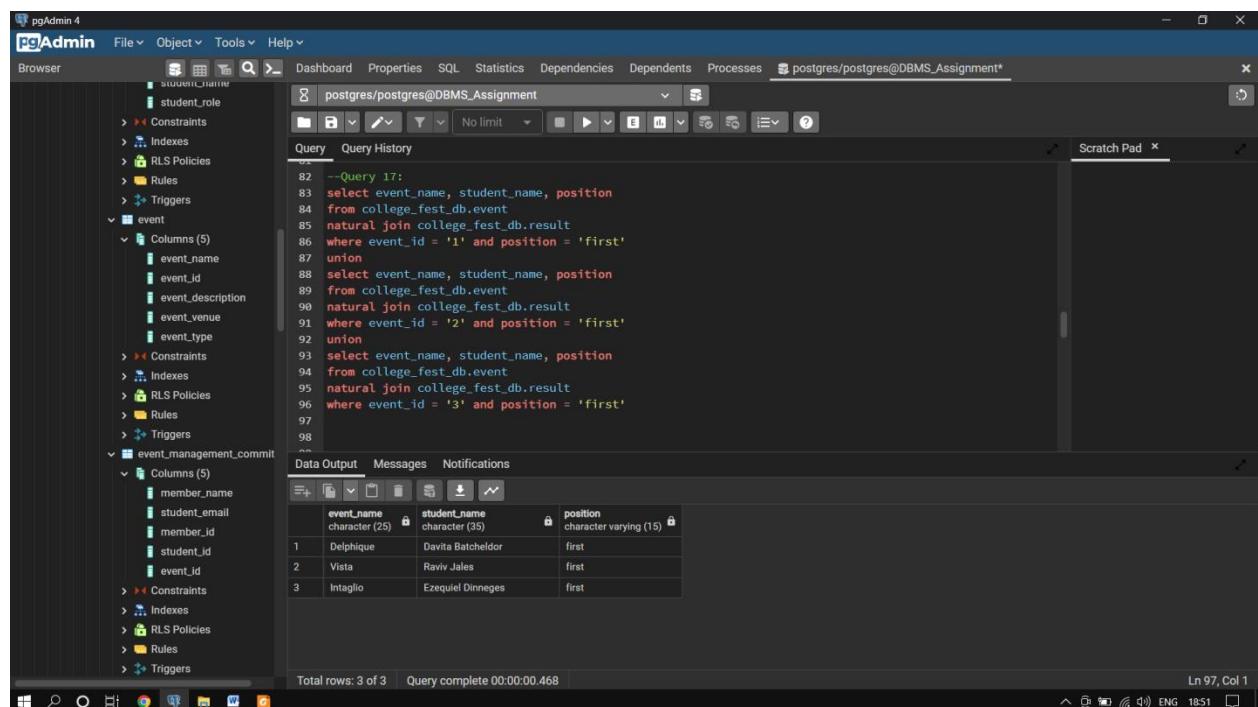
The screenshot shows the pgAdmin 4 interface with the following details:

- File Menu:** File, Object, Tools, Help
- Toolbar:** Browser, Dashboard, Properties, SQL, Statistics, Dependencies, Dependents, Processes, Database (postgres/postgres@DBMS_Assignment*)
- Query Editor:** Shows the query: `--Query 16:
SELECT p_name, student_email FROM college_fest_db.participants`
- Data Output:** A table with 100 rows showing the results of the query. The columns are `p_name` and `student_email`. The data includes rows for various participants like Alicia Cuross, Shandra Faber, and Junie Fruish, along with their respective email addresses.
- Messages:** No messages are present.
- Notifications:** No notifications are present.
- Bottom Status:** Total rows: 100 of 100, Query complete 00:00:00.455, Ln 79, Col 32

17. English query: Select participants' names who secured first position from the top 3 events in the Event table.

Query:

```
select event_name, student_name, position
from college_fest_db.event
natural join college_fest_db.result
where event_id = '1' and position = 'first'
union
select event_name, student_name, position
from college_fest_db.event
natural join college_fest_db.result
where event_id = '2' and position = 'first'
union
select event_name, student_name, position
from college_fest_db.event
natural join college_fest_db.result
where event_id = '3' and position = 'first';
```



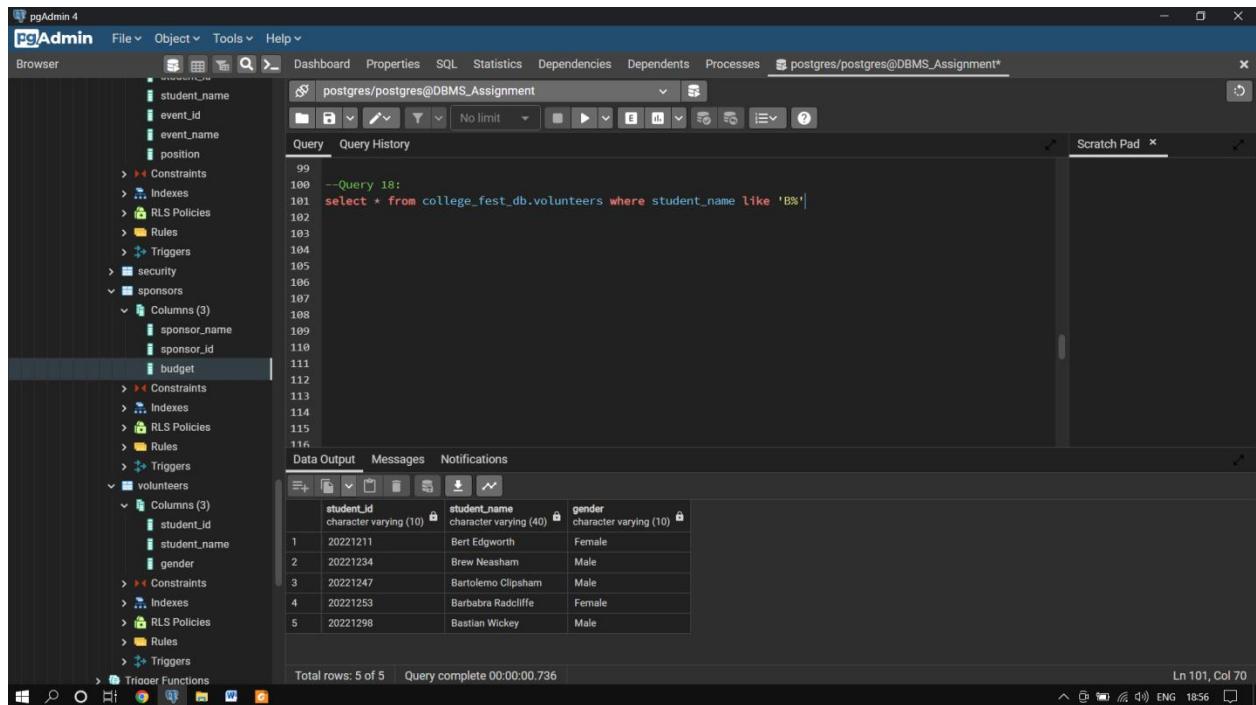
The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, displaying the database schema with tables like student_role, event, and event_management_commit. The main pane is the 'Query' pane, showing the SQL query and its execution results. The results table shows three rows of data:

	event_name	student_name	position
1	Delphique	Davita Batchelder	first
2	Vista	Raviv Jales	first
3	Intaglio	Ezequiel Dinnege	first

Total rows: 3 of 3 | Query complete 00:00:00.468 | Ln 97, Col 1

18. English query: Select volunteers with names starting with B

Query: select * from college_fest_db.volunteers where student_name like 'B%';



The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, which lists database objects: student_name, event_id, event_name, position, Constraints, Indexes, RLS Policies, Rules, Triggers, security, sponsors, and volunteers. The 'sponsors' and 'volunteers' nodes are expanded, showing their respective columns. The 'sponsors' node has three columns: sponsor_name, sponsor_id, and budget. The 'volunteers' node has three columns: student_id, student_name, and gender. The right pane is the 'Query' pane, showing the following SQL code:

```
99
100 --Query 18:
101 select * from college_fest_db.volunteers where student_name like 'B%';
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
```

Below the code, the 'Data Output' tab is selected, displaying the results of the query:

	student_id	student_name	gender
1	20221211	Bert Edgworth	Female
2	20221234	Brew Neasham	Male
3	20221247	Bartolomeo Clipsham	Male
4	20221253	Barbabra Radcliffe	Female
5	20221298	Bastian Wickey	Male

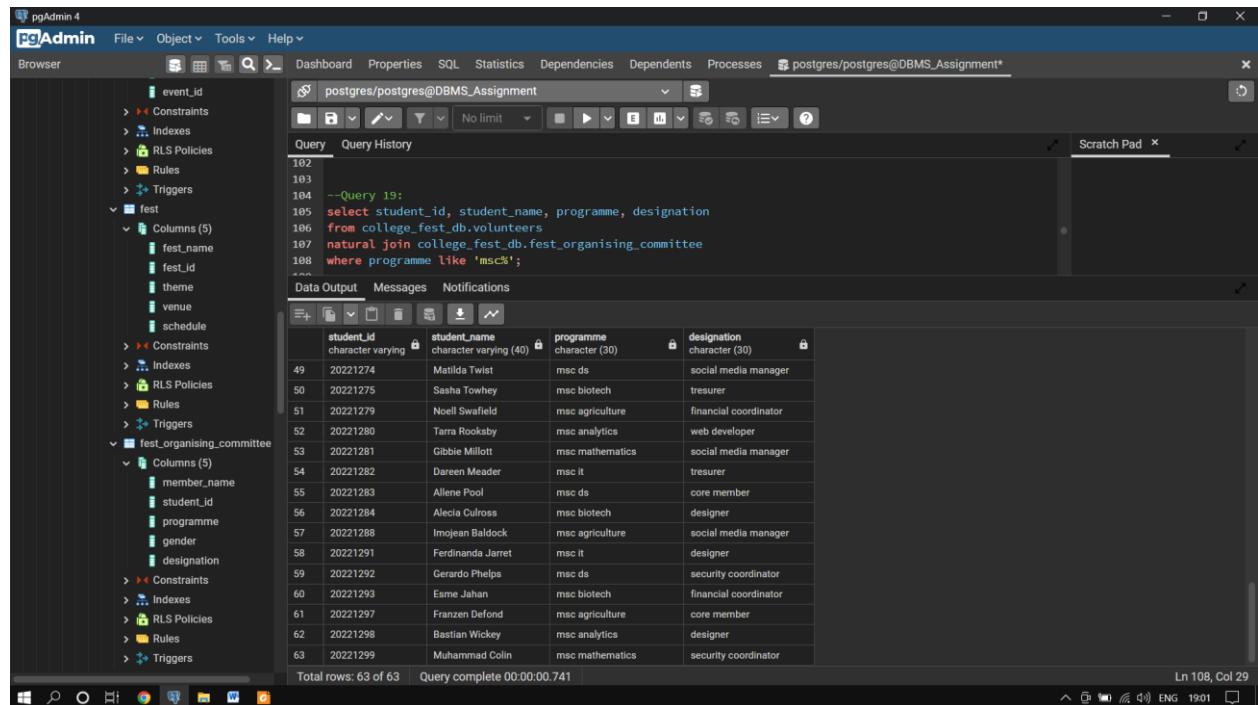
The status bar at the bottom indicates 'Total rows: 5 of 5' and 'Query complete 00:00:00.736'.

19. English query: Display the following details of volunteers from MSc IT programme

- a. Student ID
- b. Student name
- c. Programme
- d. Designation

Query:

```
select student_id, student_name, programme, designation
from college_fest_db.volunteers
natural join college_fest_db.fest_organising_committee
where programme like 'msc%';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with tables like 'fest', 'fest_organising_committee', and their columns. The main area shows the query editor with the following SQL code:

```
--Query 19:
select student_id, student_name, programme, designation
from college_fest_db.volunteers
natural join college_fest_db.fest_organising_committee
where programme like 'msc%';
```

Below the query, the results are displayed in a table:

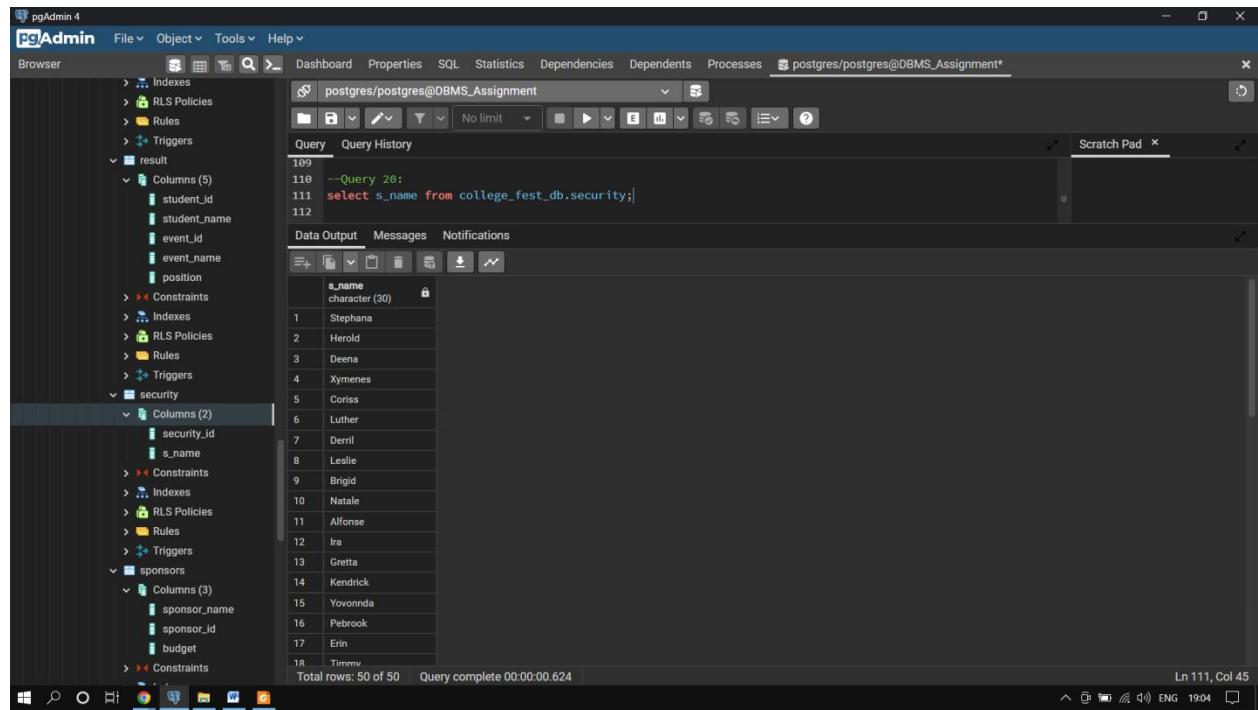
	student_id	student_name	programme	designation
49	20221274	Matilda Twist	msc ds	social media manager
50	20221275	Sasha Towhey	msc biotech	treasurer
51	20221279	Noell Swafeld	msc agriculture	financial coordinator
52	20221280	Terra Rooksby	msc analytics	web developer
53	20221281	Gibbie Millott	msc mathematics	social media manager
54	20221282	Dareen Meader	msc it	treasurer
55	20221283	Allene Pool	msc ds	core member
56	20221284	Alecia Culross	msc biotech	designer
57	20221288	Imogen Baldoek	msc agriculture	social media manager
58	20221291	Ferdinanda Jarret	msc it	designer
59	20221292	Gerardo Phelps	msc ds	security coordinator
60	20221293	Esme Jahan	msc biotech	financial coordinator
61	20221297	Franzen Defond	msc agriculture	core member
62	20221298	Bastian Wickey	msc analytics	designer
63	20221299	Muhammad Colin	msc mathematics	security coordinator

Total rows: 63 of 63 | Query complete 00:00:00.741 | Ln 108, Col 29

20. English query: Select all the Security members.

Query:

```
select s_name from college_fest_db.security;
```



The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, which is expanded to show the 'result' node. Under 'result', there are two tables: 'student' (5 columns) and 'security' (2 columns). The 'security' table is currently selected. The main pane is the 'Query' tab, showing the following SQL query:

```
--Query 20:  
select s_name from college_fest_db.security;
```

The 'Data Output' tab shows the results of the query, listing 18 rows of security members:

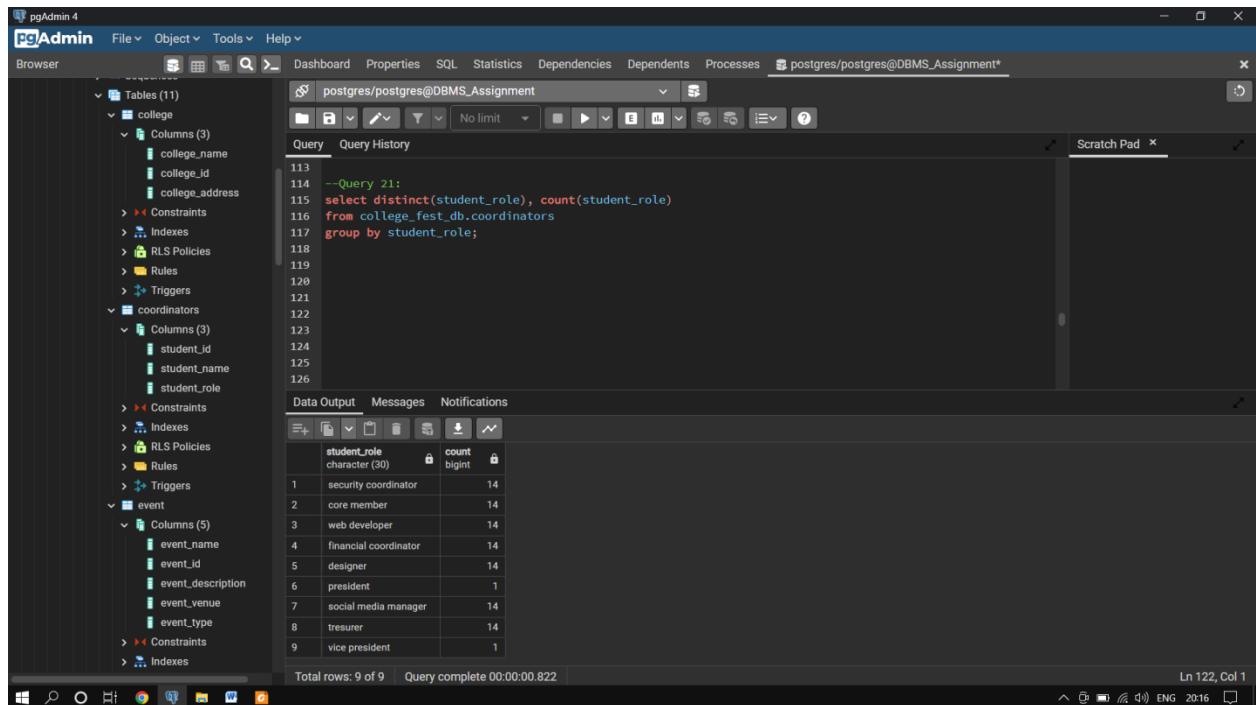
s_name
Stephana
Herold
Deena
Xymenes
Coriss
Luther
Derril
Leslie
Brigid
Natale
Alfonse
Ira
Gretta
Kendrick
Yovonna
Pebrook
Erin
Timmu

At the bottom of the main pane, it says 'Total rows: 50 of 50' and 'Query complete 00:00:00.624'. The status bar at the bottom right shows 'Ln 111, Col 45'.

21. English query: Display total count of distinct Student roles.

Query:

```
select distinct(student_role), count(student_role)
from college_fest_db.coordinators
group by student_role;
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database structure with tables: college, coordinators, and event. The college table has columns college_name, college_id, and college_address. The coordinators table has columns student_id, student_name, and student_role. The event table has columns event_name, event_id, event_description, event_venue, and event_type. The main window (Query Editor) shows the SQL query:

```
--Query 21:
select distinct(student_role), count(student_role)
from college_fest_db.coordinators
group by student_role;
```

The Data Output tab shows the results of the query:

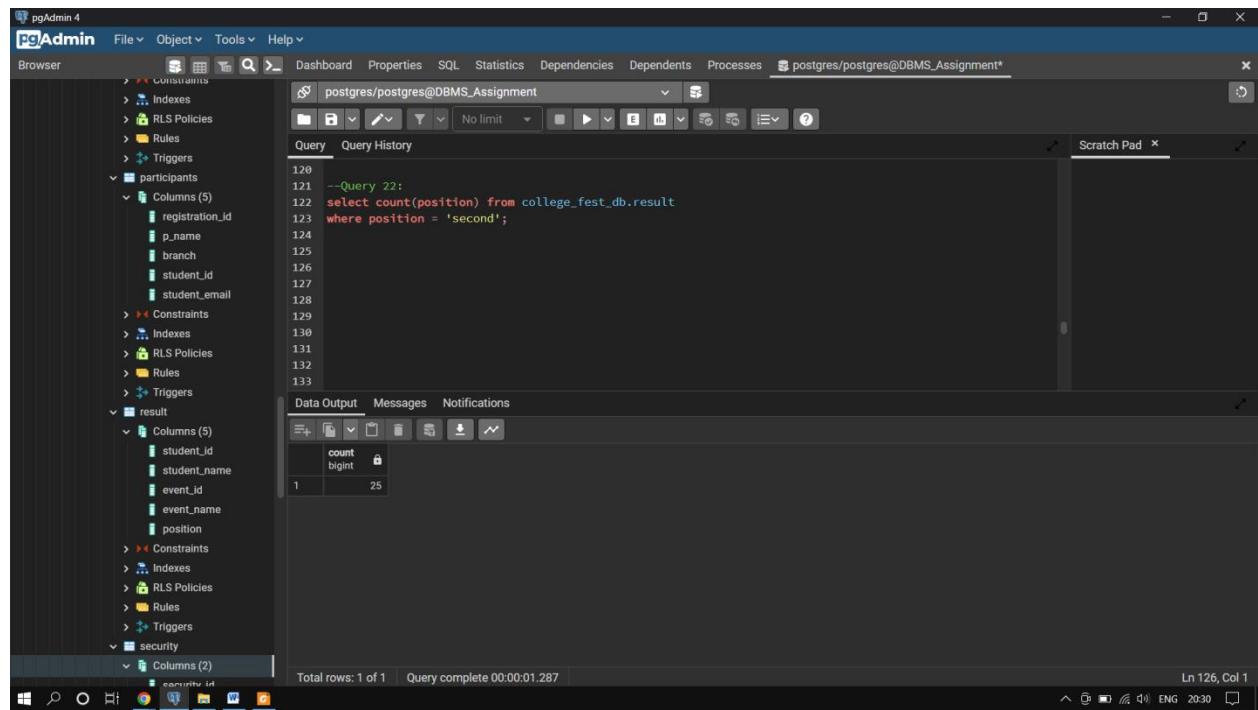
student_role	count
security coordinator	14
core member	14
web developer	14
financial coordinator	14
designer	14
president	1
social media manager	14
treasurer	14
vice president	1

Total rows: 9 of 9 Query complete 00:00:00.822 Ln 122, Col 1

22. English query: Count the total number of Second positions

Query:

```
select count(position) from college_fest_db.result  
where position = 'second';
```



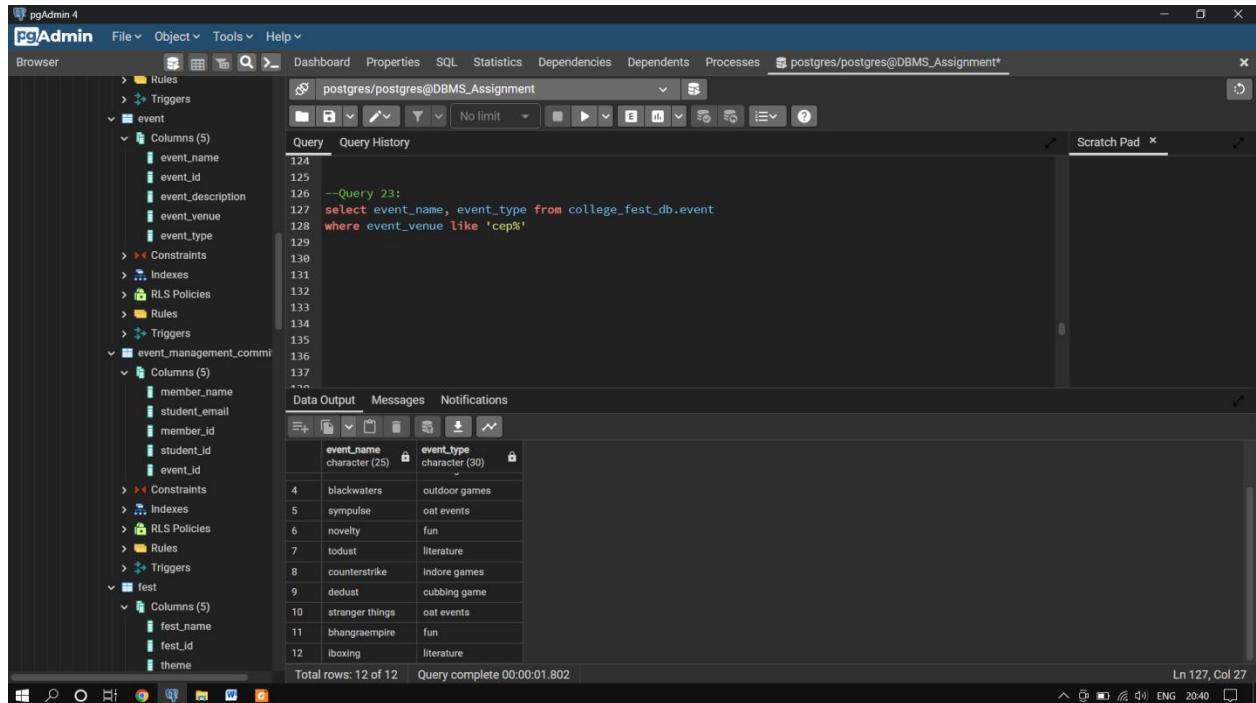
The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, which lists the database schema. It includes the 'participants' table with columns: registration_id, p_name, branch, student_id, and student_email. It also lists the 'result' table with columns: student_id, student_name, event_id, event_name, and position. The main area is the 'Query' pane, showing the SQL query and its execution results. The results table has one row with 'count' as 1 and 'bigint' as 25. The status bar at the bottom indicates 'Query complete 00:00:01.287'.

count	bigint
1	25

23. English query: Display all the Event Name and Event Type which are hosted in CEP Block.

Query:

```
select event_name, event_type from college_fest_db.event
where event_venue like 'cep%';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure for the 'college_fest_db' schema, including tables like 'event', 'event_management_commi', and 'fest'. The main window shows a query editor with the following SQL code:

```
--Query 23:
select event_name, event_type from college_fest_db.event
where event_venue like 'cep%';
```

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

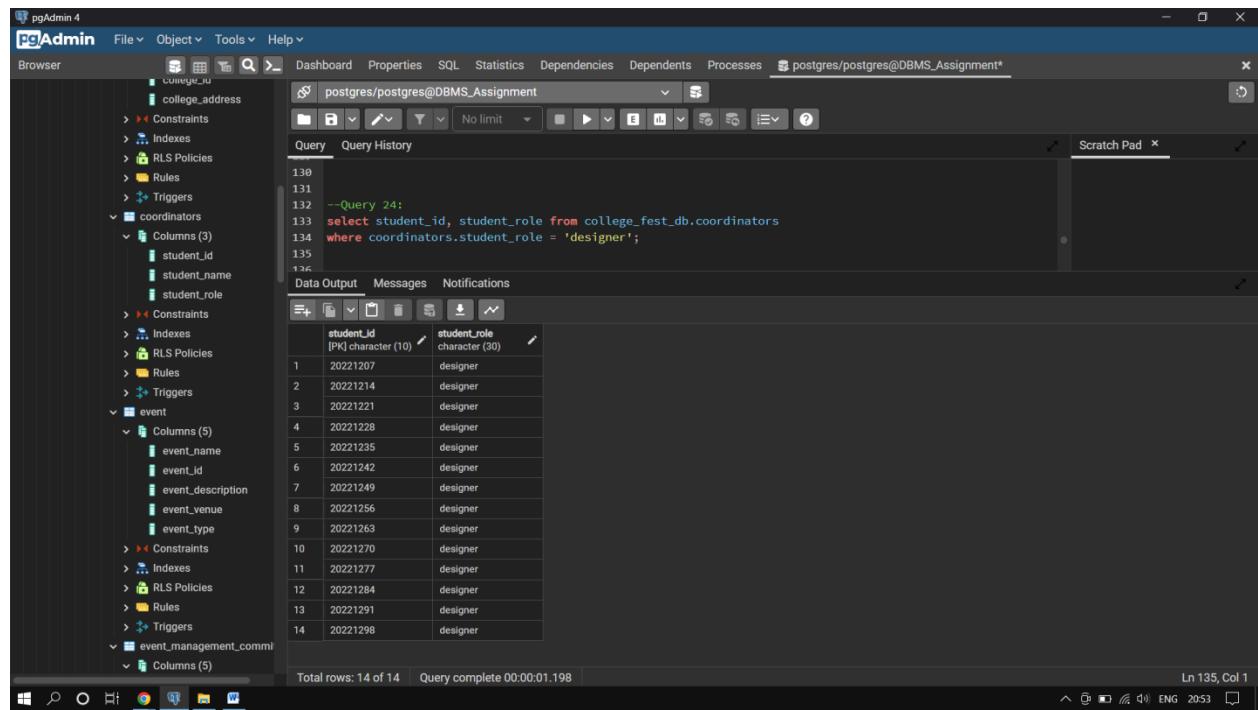
	event_name	event_type
4	blackwaters	outdoor games
5	sympulse	oat events
6	novelty	fun
7	todust	literature
8	counterstrike	Indore games
9	dedust	cubbing game
10	stranger things	oat events
11	bhangraempire	fun
12	lboxing	literature

The status bar at the bottom indicates 'Total rows: 12 of 12' and 'Query complete 00:00:01.802'.

24. English query: Select all the Student IDs of Designers

Query:

```
select student_id, student_role from college_fest_db.coordinators  
where coordinators.student_role = 'designer';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database schema with tables like 'college_fest_db' and 'coordinators'. The main area (Query) contains the SQL query:

```
--Query 24:  
select student_id, student_role from college_fest_db.coordinators  
where coordinators.student_role = 'designer';
```

The results are displayed in a table:

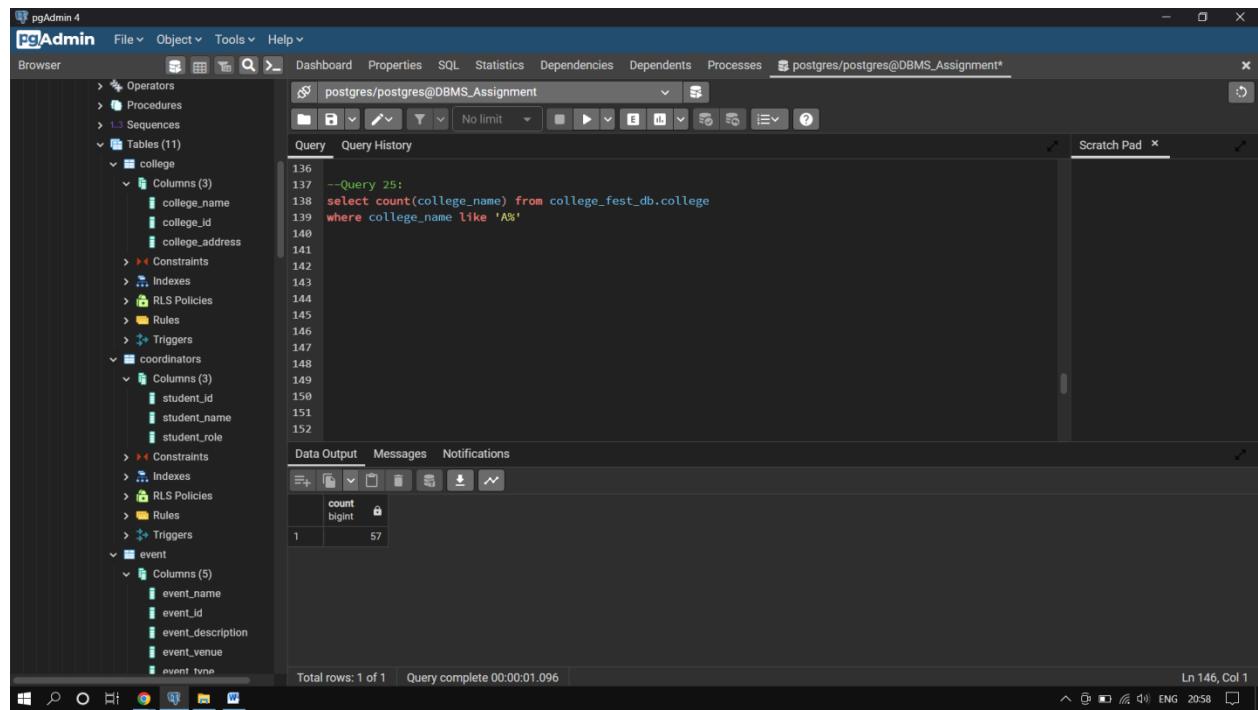
student_id	student_role	
1	20221207	designer
2	20221214	designer
3	20221221	designer
4	20221228	designer
5	20221235	designer
6	20221242	designer
7	20221249	designer
8	20221256	designer
9	20221263	designer
10	20221270	designer
11	20221277	designer
12	20221284	designer
13	20221291	designer
14	20221298	designer

At the bottom, the status bar shows 'Total rows: 14 of 14' and 'Query complete 00:00:01.198'.

25. English query: Display the count of participating colleges whose name start with A.

Query:

```
select count(college_name) from college_fest_db.college
where college_name like 'A%';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, which lists database objects: Operators, Procedures, Sequences, Tables (11), Constraints, Indexes, RLS Policies, Rules, Triggers, coordinators, event, and event. The 'Tables (11)' section is expanded, showing the 'college' table with columns college_name, college_id, and college_address, and the 'event' table with columns event_name, event_id, event_description, event_venue, and event_type. The right pane is the 'Query' editor, showing the following SQL code:

```
136 --Query 25:
137 select count(college_name) from college_fest_db.college
138 where college_name like 'A%'
139
140
141
142
143
144
145
146
147
148
149
150
151
152
```

Below the code, the 'Data Output' tab is selected, showing the result of the query:

count	bigint
1	57

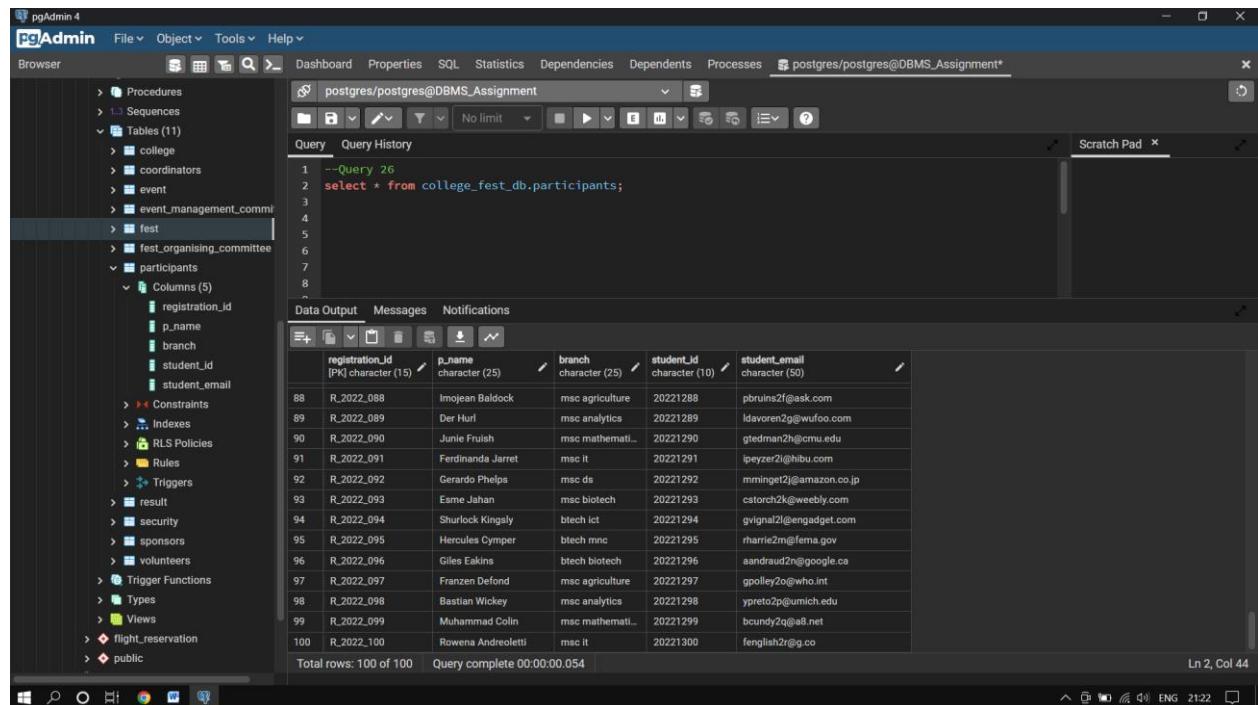
At the bottom of the query editor, it says 'Total rows: 1 of 1' and 'Query complete 00:00:01.096'. The status bar at the bottom right shows 'Ln 146, Col 1'.

26. English query: Display the following participants details

- Registration ID
- Name
- Branch
- Student ID
- Email Address

Query:

```
select * from college_fest_db.participants;
```



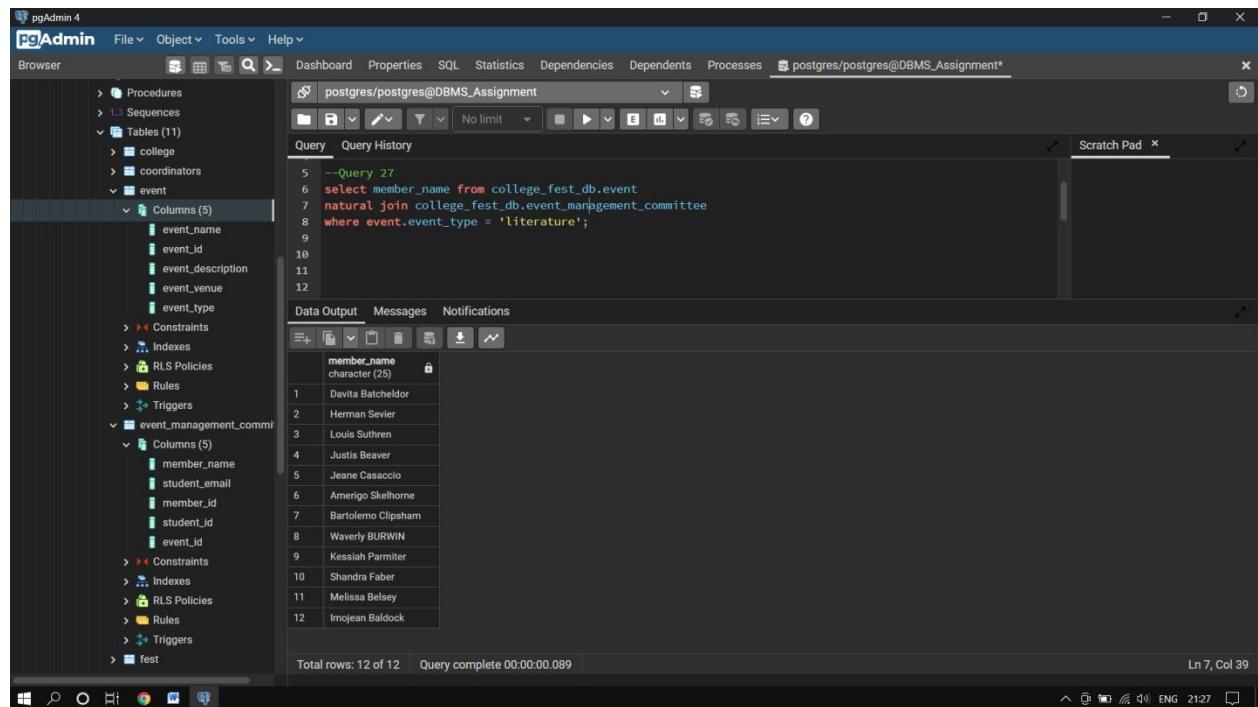
The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure, including tables like college, coordinators, event, event_management_committi, fest, fest_organising_committee, and participants. The participants table is selected, showing 5 columns: registration_id, p_name, branch, student_id, and student_email. The main window shows the query `select * from college_fest_db.participants;` and the resulting data output, which is a table with 100 rows of participant details. The table has columns: registration_id, p_name, branch, student_id, and student_email. The data includes various names, branches (like agriculture, analytics, math, IT, ds, biotech, etc.), student IDs, and email addresses.

registration_id	p_name	branch	student_id	student_email
R_2022_088	Imojean Baldock	msc agriculture	20221288	pbruins2f@ask.com
R_2022_089	Der Hurl	msc analytics	20221289	ldavoren2g@wufuu.com
R_2022_090	Junie Fruish	msc mathemati...	20221290	gtedman2h@cmu.edu
R_2022_091	Ferdinanda Jarret	msc it	20221291	ipeyzer2i@hibu.com
R_2022_092	Gerardo Phelps	msc ds	20221292	mminge12j@amazon.co.jp
R_2022_093	Esme Jahan	msc biotech	20221293	cstorch2k@weebly.com
R_2022_094	Shurlock Kingsly	btech ict	20221294	gvignal2l@engadget.com
R_2022_095	Hercules Cymper	btech mnc	20221295	rharrie2m@fema.gov
R_2022_096	Giles Eakins	btech biotech	20221296	aandrau2n@google.ca
R_2022_097	Franzen Defond	msc agriculture	20221297	gpolley2o@who.int
R_2022_098	Bastian Wickey	msc analytics	20221298	yprete2p@umich.edu
R_2022_099	Muhammad Colin	msc mathemati...	20221299	bcundy2q@a8.net
R_2022_100	Rowena Androletti	msc it	20221300	fenglish2r@g.co

27. English query: Display the Member Names from the Management Committee of Literature event.

Query:

```
select member_name from college_fest_db.event
natural join college_fest_db.event_management_committee
where event.event_type = 'literature';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure with tables like 'college', 'coordinators', 'event', and 'event_management_committee'. The 'event' table has 5 columns: event_name, event_id, event_description, event_venue, and event_type. The 'event_management_committee' table has 5 columns: member_name, student_email, member_id, student_id, and event_id. The central pane shows the SQL query being run:

```
5 --Query 27
6 select member_name from college_fest_db.event
7 natural join college_fest_db.event_management_committee
8 where event.event_type = 'literature';
9
10
11
12
```

The 'Data Output' pane displays the results:

member_name
1 Davita Batcheldor
2 Herman Sevier
3 Louis Suthren
4 Justis Beaver
5 Jeane Casaclo
6 Amerigo Skelhome
7 Bartolome Clipsham
8 Waverly BURWIN
9 Kessiah Parmiter
10 Shandra Faber
11 Melissa Belsey
12 Imojean Baldock

Total rows: 12 of 12 | Query complete 00:00:00.089 | Ln 7, Col 39

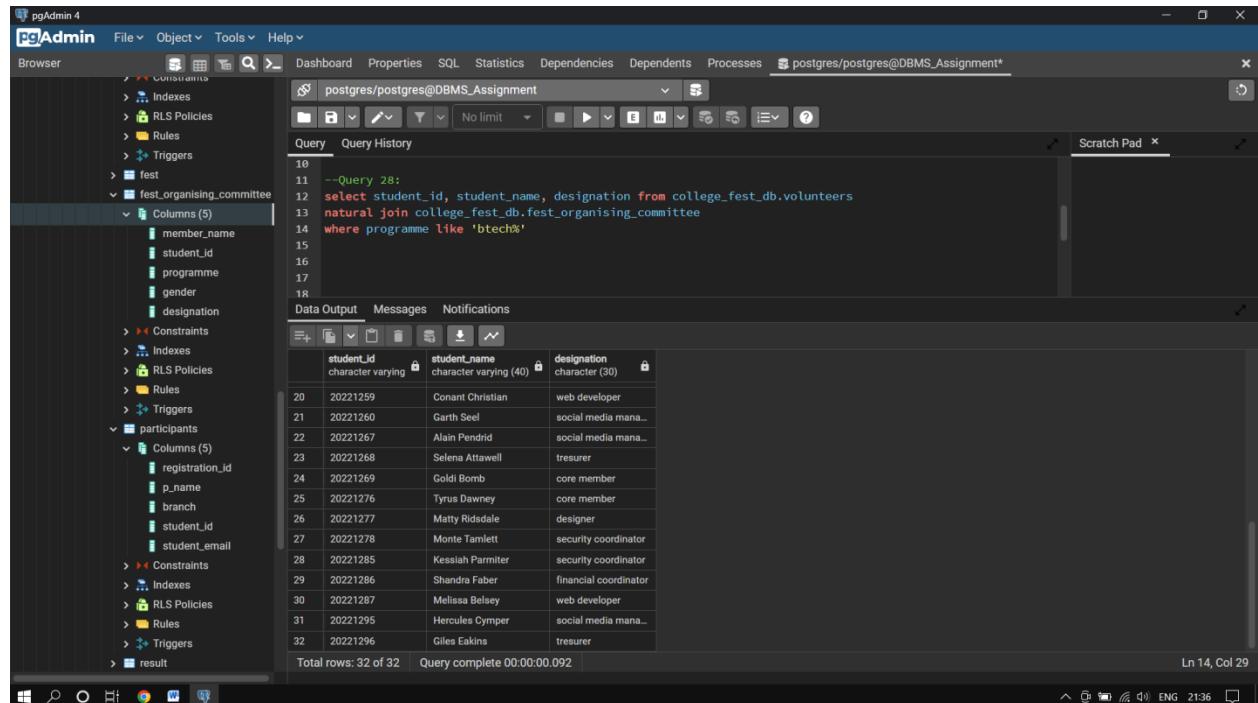
28. English query: Select the following details of Fest Organising Committee members

- Student ID
- Student Name
- Designation

Where the Programme is Btech.

Query:

```
select student_id, student_name, designation from college_fest_db.volunteers
natural join college_fest_db.fest_organising_committee
where programme like 'btech%';
```



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

- Browser:** The left sidebar shows the database structure for the 'college_fest_db' schema, including 'fest' and 'fest_organising_committee' tables.
- Query Editor:** The main area displays the SQL query for selecting data from the 'fest_organising_committee' table based on the 'student_id' and 'programme' columns.
- Data Output:** The results of the query are shown in a table format, listing 32 rows of data.
- Table Headers:** The columns are labeled 'student_id', 'student_name', and 'designation'.
- Table Data:** The data includes various student IDs and names, along with their designations such as 'web developer', 'social media mana...', 'treasurer', 'core member', etc.
- Message Bar:** The bottom bar indicates 'Query complete 00:00:00.092'.
- System Bar:** The bottom right shows system status including 'Ln 14, Col 29'.

29. English query: Display the following details for Student IDs: 20221220, 20221224, 20221228

- a. Student Name
- b. Event Name
- c. Position

Query:

```
select student_name, event_name, position from college_fest_db.result
```

```
where student_id = '20221220'
```

```
union
```

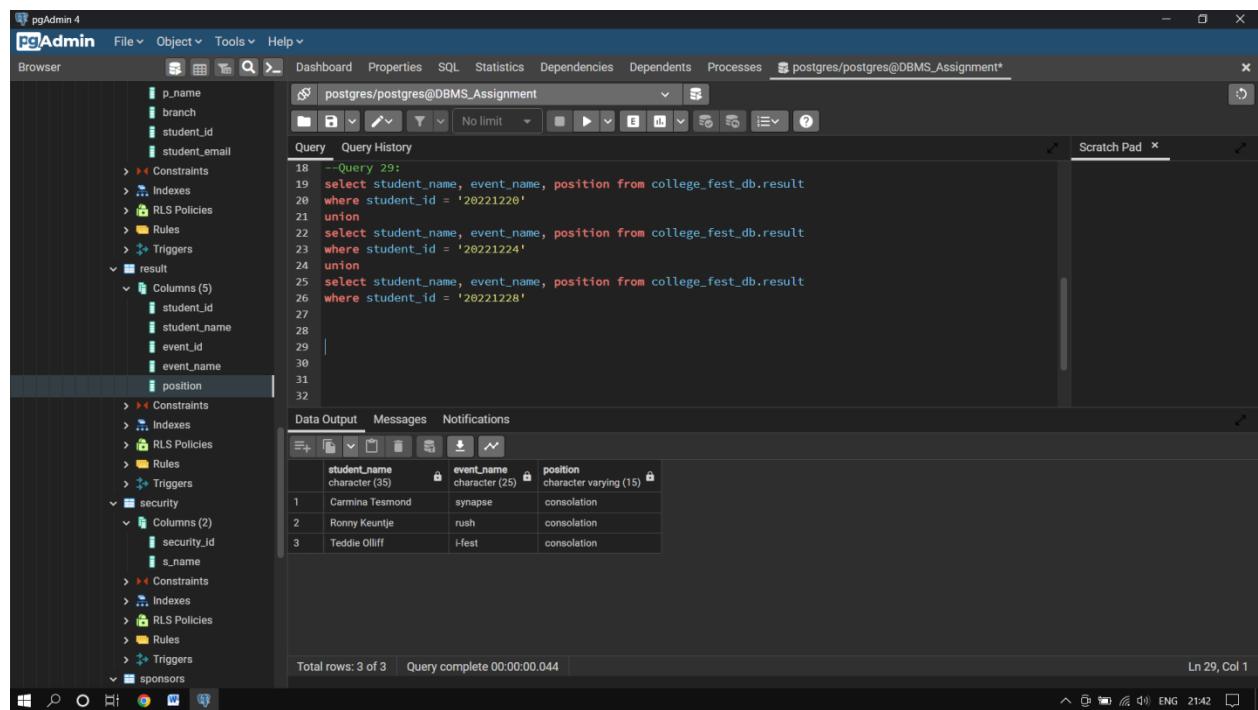
```
select student_name, event_name, position from college_fest_db.result
```

```
where student_id = '20221224'
```

```
union
```

```
select student_name, event_name, position from college_fest_db.result
```

```
where student_id = '20221228';
```



The screenshot shows the pgAdmin 4 interface with a query editor and a results table.

Query Editor:

```
--Query 29:
18
19 select student_name, event_name, position from college_fest_db.result
20 where student_id = '20221220'
21 union
22 select student_name, event_name, position from college_fest_db.result
23 where student_id = '20221224'
24 union
25 select student_name, event_name, position from college_fest_db.result
26 where student_id = '20221228'
```

Results Table:

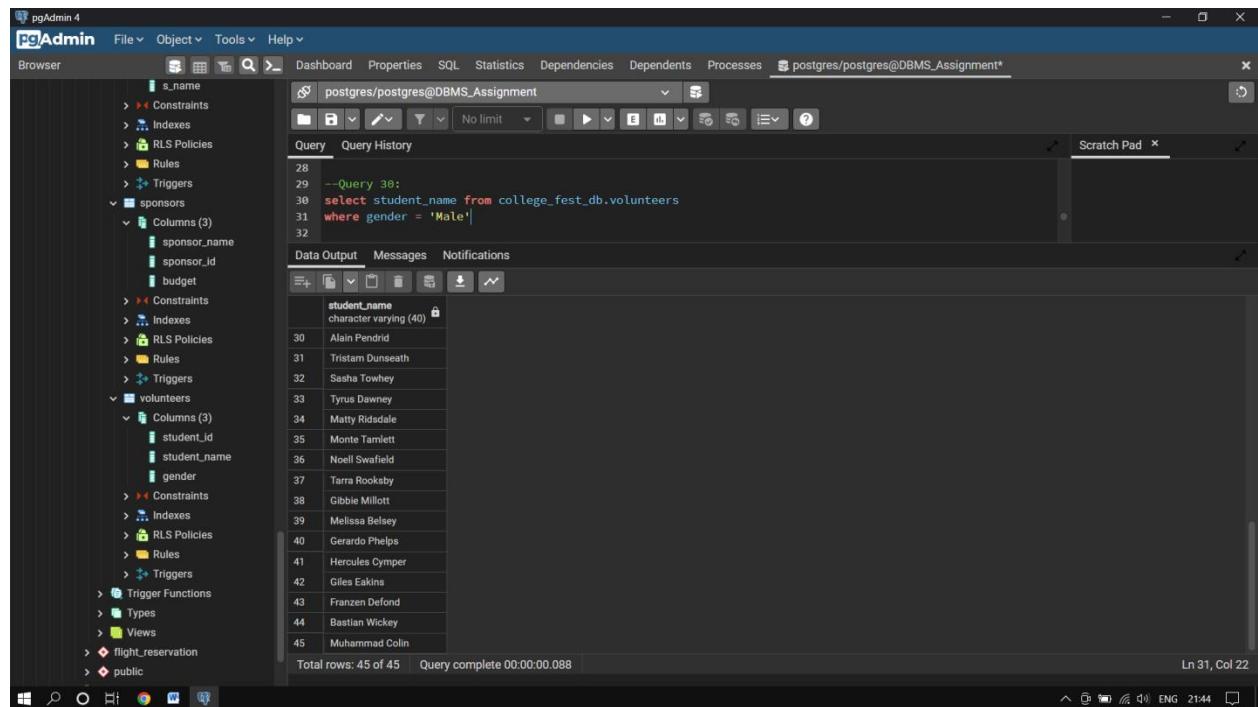
	student_name	event_name	position
1	Carmina Tesmond	synapse	consolation
2	Ronny Keuntje	rush	consolation
3	Teddie Olliff	i-fest	consolation

Total rows: 3 of 3 Query complete 00:00:00.044 Ln 29, Col 1

30. English query: Select names of Male volunteers.

Query:

```
select student_name from college_fest_db.volunteers
where gender = 'Male';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, which lists various database objects: 'sponsors' (with 'Columns (3)' including 'sponsor_name', 'sponsor_id', and 'budget'), and 'volunteers' (with 'Columns (3)' including 'student_id', 'student_name', and 'gender'). The right pane is the 'Query' editor, showing the following SQL code:

```
28
29  --Query 30:
30  select student_name from college_fest_db.volunteers
31  where gender = 'Male';
32
```

Below the code, the 'Data Output' tab is selected, displaying a table with the results:

student_name
character varying (40)
30 Alain Pendrid
31 Tristam Dunseath
32 Sasha Towhey
33 Tyrus Dawney
34 Maty Riddale
35 Monte Tamlett
36 Noell Swafield
37 Tarr Rooksby
38 Gibbie Millott
39 Melissa Belsey
40 Gerardo Phelps
41 Hercules Cymper
42 Giles Eakins
43 Franzen Defond
44 Bastian Wickey
45 Muhammed Colin

At the bottom of the query editor, it says 'Total rows: 45 of 45' and 'Query complete 00:00:00.088'. The status bar at the bottom right shows 'Ln 31, Col 22'.

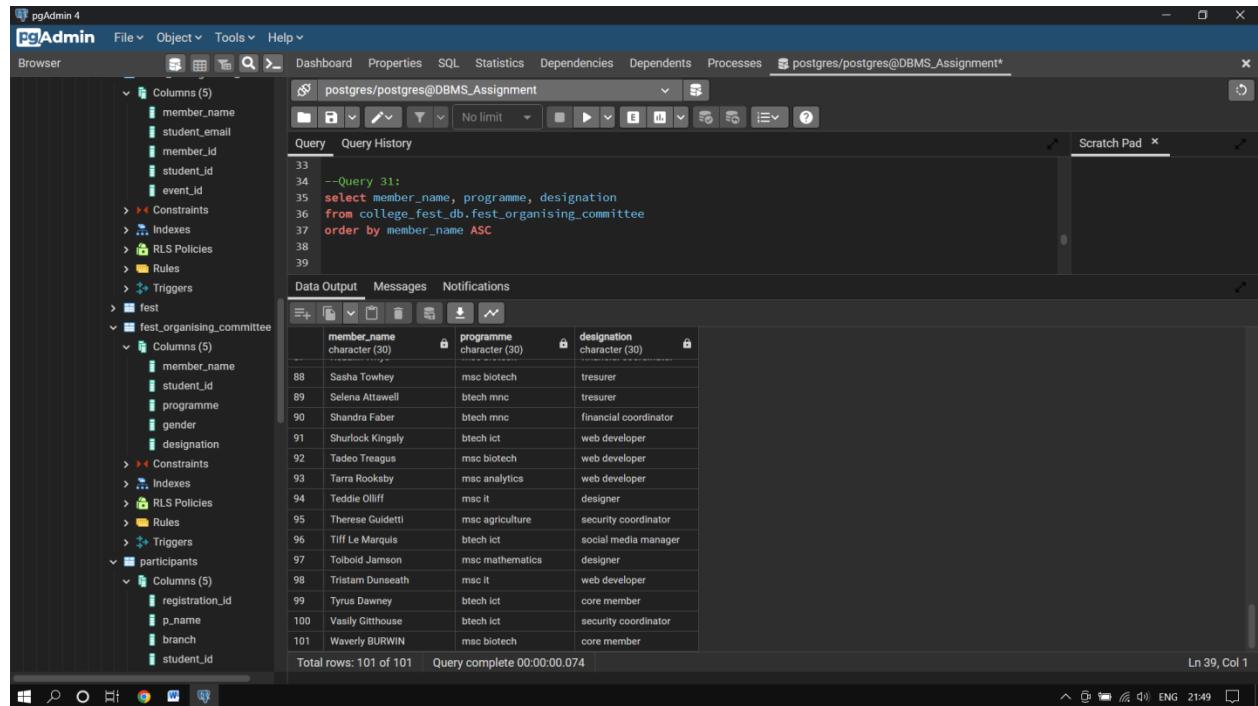
31. English query: Select the following details from the Fest Organising Committee

- a. Member name
- b. Programme
- c. Designation

And sort them in ascending order w.r.t member name.

Query:

```
select member_name, programme, designation
from college_fest_db.fest_organising_committee
order by member_name ASC;
```



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

- Browser:** The left sidebar shows the database structure. Under the 'fest' schema, there are three tables: 'fest_organising_committee' (with columns: member_name, student_id, programme, gender, designation), 'participants' (with columns: registration_id, p_name, branch, student_id), and 'fest' (with columns: student_email, student_id, event_id).
- Query Editor:** The main area contains the SQL query:

```
33
34 --Query 31:
35 select member_name, programme, designation
36 from college_fest_db.fest_organising_committee
37 order by member_name ASC
38
39
```

- Data Output:** The results are displayed in a table:

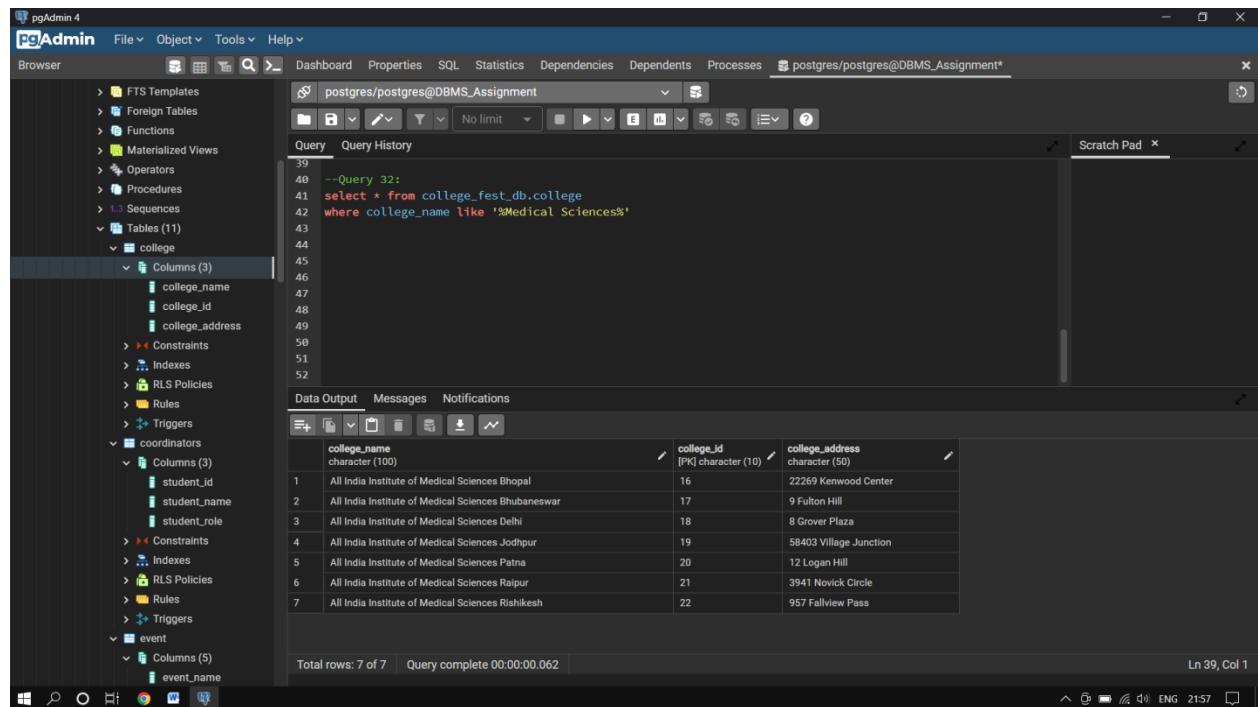
member_name	programme	designation
Sasha Towhey	msc biotech	treasurer
Selena Attawell	btech mnc	treasurer
Shandra Faber	btech mnc	financial coordinator
Shurlock Kingsly	btech ict	web developer
Tadeo Tregus	msc biotech	web developer
Tarna Rooksby	msc analytics	web developer
Teddie Olliff	msc it	designer
Therese Guidetti	msc agriculture	security coordinator
Tiff Le Marquis	btech ict	social media manager
Tolbold Jamson	msc mathematics	designer
Tristam Dunseath	msc it	web developer
Tyrus Dawney	btech ict	core member
Vasily Githhouse	btech ict	security coordinator
Waverly BURWIN	msc biotech	core member

- Messages:** The message bar at the bottom right shows: Total rows: 101 of 101 | Query complete 00:00:00.074 | Ln 39, Col 1.

32. English query: Display all the participating Medical Sciences colleges.

Query:

```
select * from college_fest_db.college
where college_name like '%Medical Sciences%';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, which lists various database objects: FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, Tables (11), and college (with sub-items: Columns (3), Constraints, Indexes, RLS Policies, Rules, Triggers). The right pane is the 'Query' pane, showing the SQL query and its execution results.

Query pane content:

```
39
40 --Query 32:
41 select * from college_fest_db.college
42 where college_name like '%Medical Sciences%'
43
44
45
46
47
48
49
50
51
52
```

Data Output pane content:

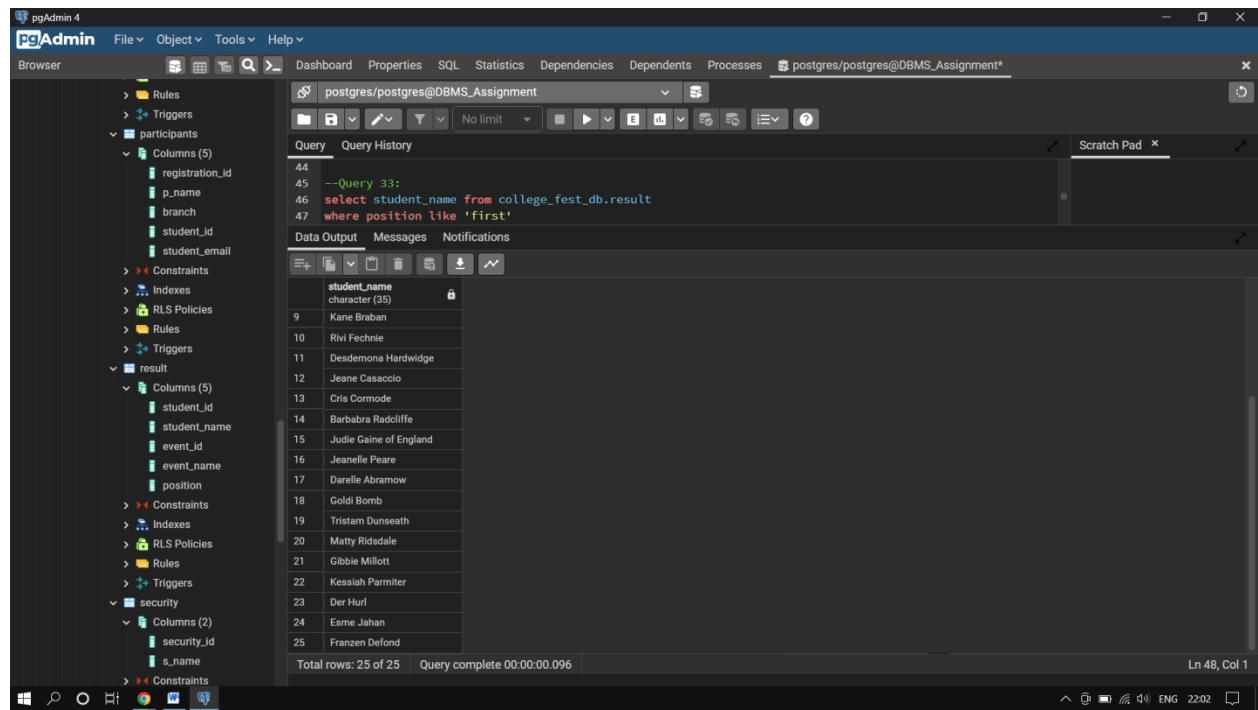
	college_name	college_id	college_address
1	All India Institute of Medical Sciences Bhopal	16	22269 Kenwood Center
2	All India Institute of Medical Sciences Bhubaneswar	17	9 Fulton Hill
3	All India Institute of Medical Sciences Delhi	18	8 Grover Plaza
4	All India Institute of Medical Sciences Jodhpur	19	58403 Village Junction
5	All India Institute of Medical Sciences Patna	20	12 Logan Hill
6	All India Institute of Medical Sciences Raipur	21	3941 Novick Circle
7	All India Institute of Medical Sciences Rishikesh	22	957 Fallview Pass

Total rows: 7 of 7 | Query complete 00:00:00.062 | Ln 39, Col 1

33. English query: Display the names of all the students who secured first position

Query:

```
select student_name from college_fest_db.result  
where position like 'first';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar (Browser) displays the database schema with tables like 'participants', 'result', and 'security'. The main area shows a query window with the following SQL code:

```
--Query 33:  
select student_name from college_fest_db.result  
where position like 'first'
```

The results pane displays a list of 25 student names, each preceded by a number (9 to 25) and a lock icon. The names are:

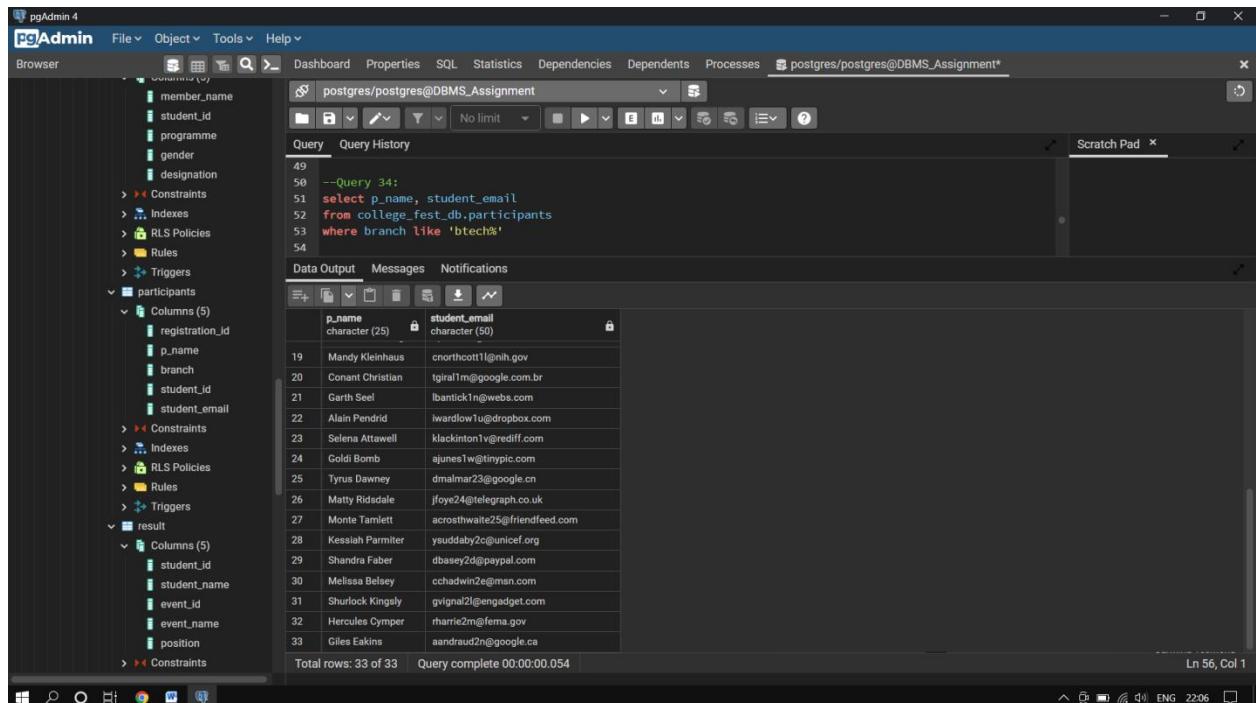
student_name
Kane Braban
Rivi Fehnle
Desdemona Hardwidge
Jeane Casaccio
Cris Cormode
Barbara Radcliffe
Judie Gaine of England
Jeanelle Peare
Darelle Abramow
Goldi Bomb
Tristam Dunseath
Matty Riddale
Gibble Millott
Kessiah Parmiter
Der Hurl
Erne Jahan
Franzen Defond

At the bottom of the results pane, it says 'Total rows: 25 of 25' and 'Query complete 00:00:00.096'.

34. English query: Display the name and email address of participants from Btech

Query:

```
select p_name, student_email
from college_fest_db.participants
where branch like 'btech%';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database structure for the 'participants' table, including columns: member_name, student_id, programme, gender, designation, p_name, branch, student_id, and student_email. The main query editor window contains the following SQL code:

```
--Query 34:
select p_name, student_email
from college_fest_db.participants
where branch like 'btech%'
```

The results table shows 33 rows of data, each with a row number (19 to 33) and columns for p_name and student_email. The data is as follows:

	p_name	student_email
19	Mandy Kleinhaus	cnorthcott1@nih.gov
20	Conant Christian	tgral1m@google.com.br
21	Garth Seel	lbantick1@webs.com
22	Alain Pendrid	lwardlow1@dropbox.com
23	Selena Attawell	klackinton1v@rediff.com
24	Goldi Bomb	ajunes1w@tinyPic.com
25	Tyrus Dawney	dmalmar23@google.cn
26	Matty Riddale	jfoye24@telegraph.co.uk
27	Monte Tamlett	acrosthwaite25@friendfeed.com
28	Kessiah Parmiter	ysuddaby2c@unicef.org
29	Shandra Faber	dbasey2d@paypal.com
30	Melissa Belsey	cchadwin2e@msn.com
31	Shurlock Kingsly	gvignau2f@engadget.com
32	Hercules Cymper	rharrie2m@fema.gov
33	Giles Eakins	aandraud2n@google.ca

At the bottom of the results window, it says "Total rows: 33 of 33" and "Query complete 00:00:00.054".

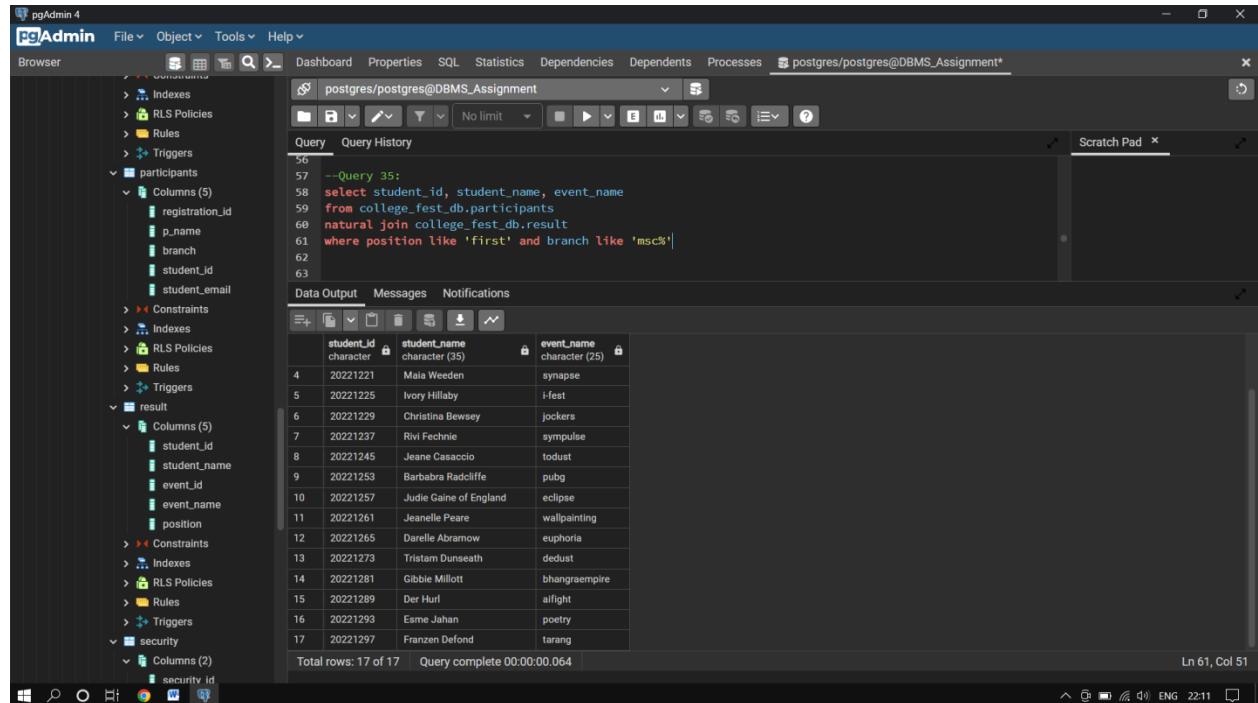
35. English query: Select the following details

- Student ID
- Student name
- Event name

Who are from MSc branch and secured first position

Query:

```
select student_id, student_name, event_name
from college_fest_db.participants
natural join college_fest_db.result
where position like 'first' and branch like 'msc%';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with tables 'participants' and 'result'. The 'participants' table has columns: registration_id, p_name, branch, student_id, and student_email. The 'result' table has columns: student_id, student_name, and event_name. The central pane shows the SQL query being run:

```
--Query 35:
select student_id, student_name, event_name
from college_fest_db.participants
natural join college_fest_db.result
where position like 'first' and branch like 'msc%';
```

The results pane displays the following data:

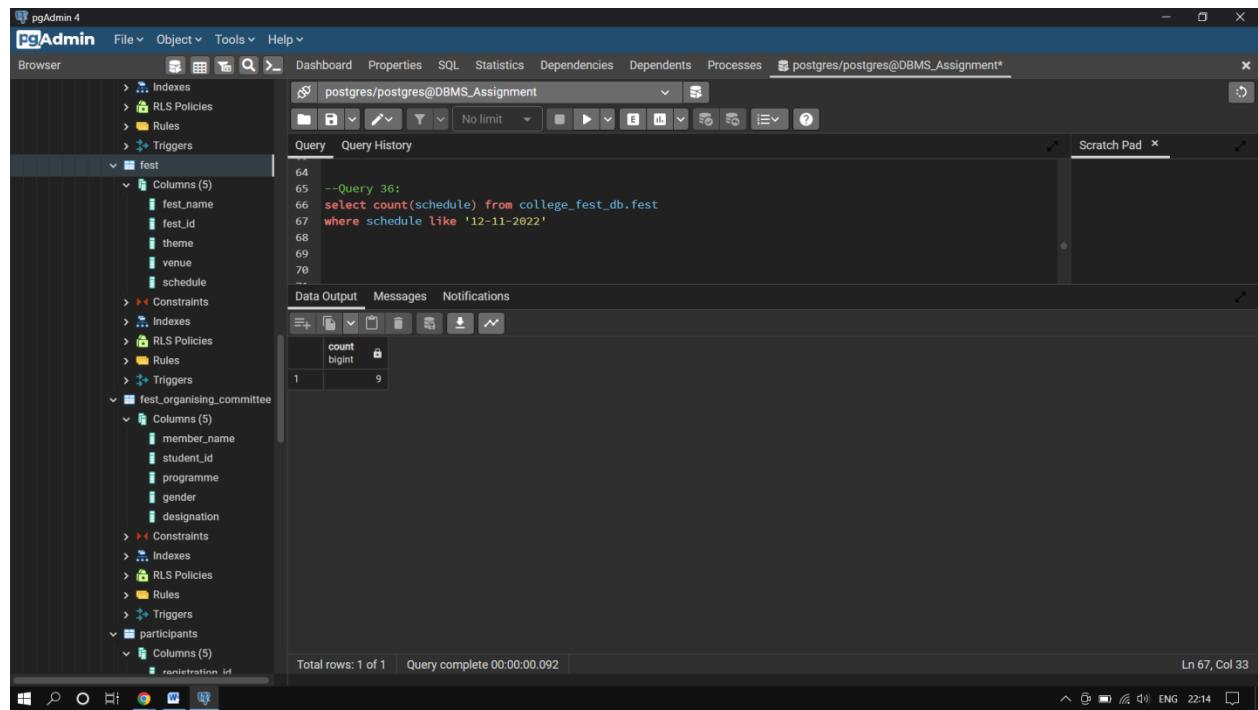
	student_id	student_name	event_name
4	20221221	Maia Weeden	synapse
5	20221225	Ivory Hillaby	i-fest
6	20221229	Christina Bewsey	jockers
7	20221237	Rivi Fechnie	sympulse
8	20221245	Jeane Casaccio	todust
9	20221253	Barbara Radcliffe	pubg
10	20221257	Judie Gaine of England	eclipse
11	20221261	Jeanelle Pearse	wallpainting
12	20221265	Darelle Abramow	euphoria
13	20221273	Tristam Dunseath	dedust
14	20221281	Gibbie Millott	bhangraempire
15	20221289	Der Hurl	aiflight
16	20221293	Esme Jahan	poetry
17	20221297	Franzen Defond	tarang

Total rows: 17 of 17 | Query complete 00:00:00.064 | Ln 61, Col 51

36. English query: Display the total number of events hosted on date 12-11-2022

Query:

```
select count(schedule) from college_fest_db.fest
where schedule like '12-11-2022';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with a tree view of tables and their columns. The main area shows a query editor with the following SQL code:

```
--Query 36:
select count(schedule) from college_fest_db.fest
where schedule like '12-11-2022'
```

The results pane shows a single row of data:

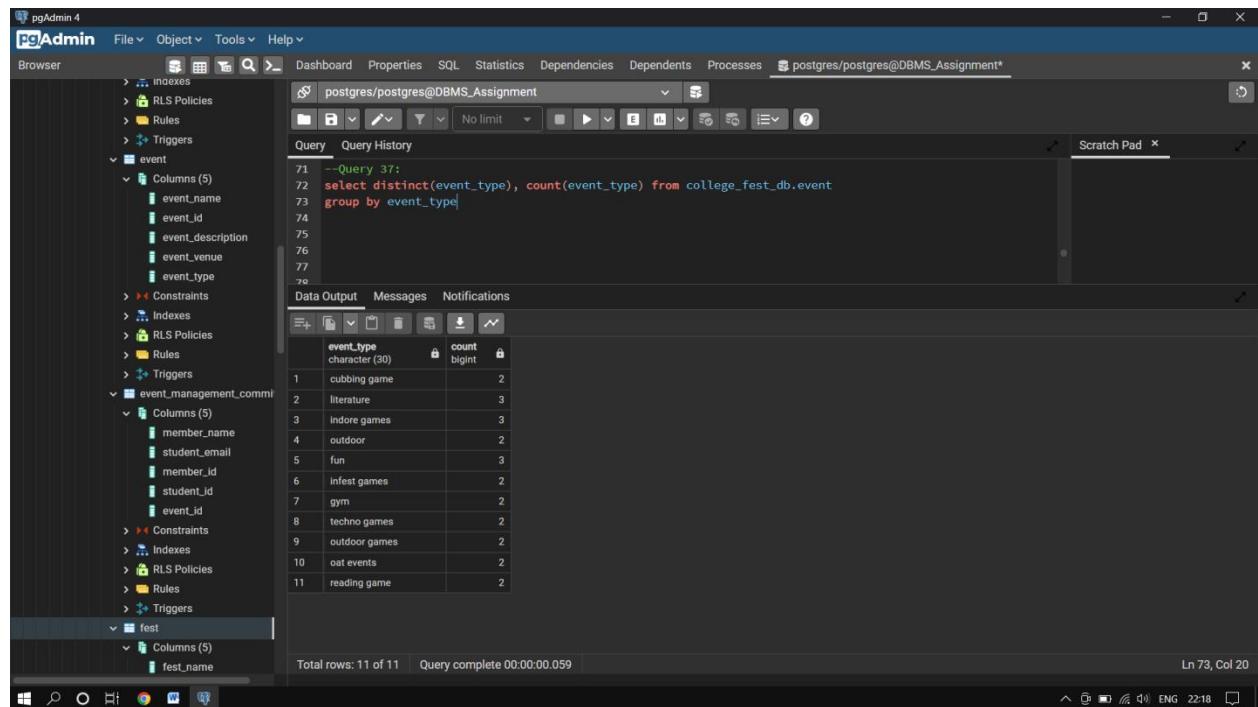
count	bigint
1	9

Below the results, the status bar indicates "Total rows: 1 of 1" and "Query complete 00:00:00.092". The bottom right corner shows the system tray with the text "Ln 67, Col 33".

37. English query: Display the number of each event type.

Query:

```
select distinct(event_type), count(event_type) from college_fest_db.event  
group by event_type;
```



The screenshot shows the pgAdmin 4 interface. The left sidebar is the 'Browser' pane, which contains a tree view of database objects. The 'event' table is selected, and its columns (event_name, event_id, event_description, event_venue, event_type) are listed. The 'fest' table is also visible. The main area is the 'Query' pane, showing the SQL query and its execution results. The results table shows 11 rows of event types and their counts.

event_type	count
cubbing game	2
literature	3
indore games	3
outdoor	2
fun	3
infest games	2
gym	2
techno games	2
outdoor games	2
oat events	2
reading game	2

Total rows: 11 of 11 | Query complete 00:00:00.059 | Ln 73, Col 20

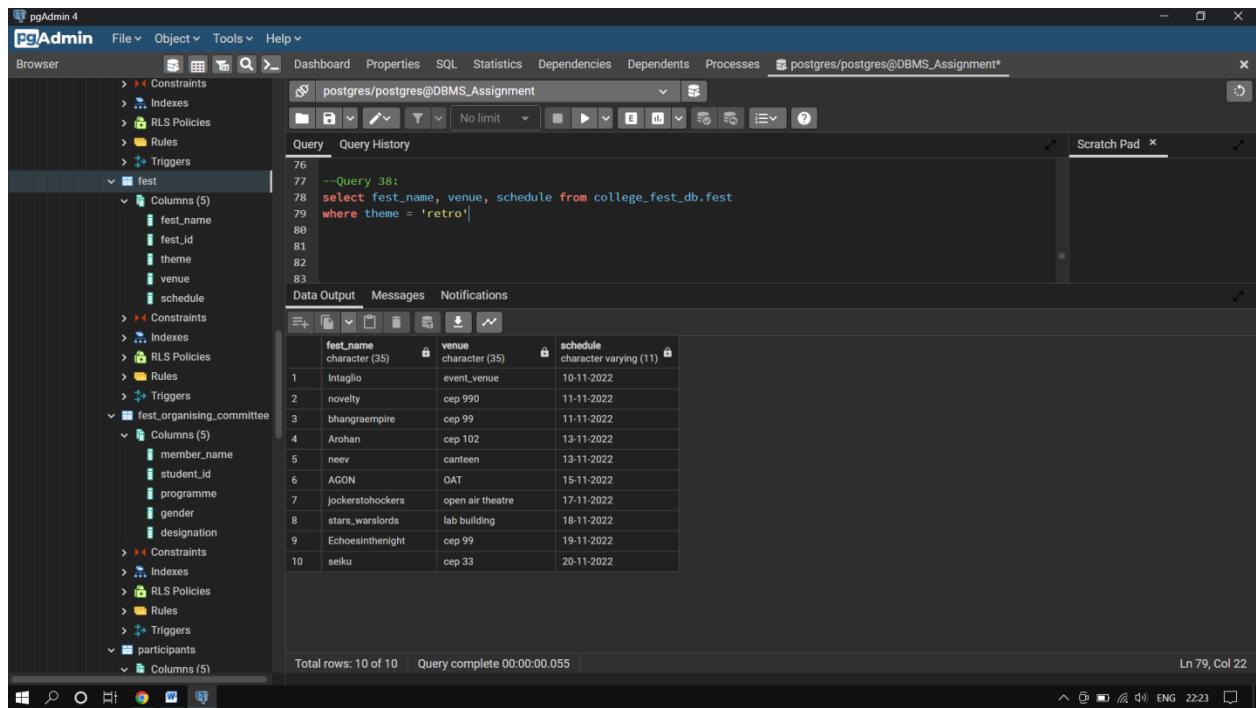
38. English query: Select the following details

- a. Fest name
- b. Venue
- c. Schedule

Where fest theme is Retro.

Query:

```
select fest_name, venue, schedule from college_fest_db.fest
where theme = 'retro';
```



The screenshot shows the pgAdmin 4 interface. The left sidebar displays the database schema with tables like fest, fest_organising_committee, and participants. The central pane shows the SQL query being run:

```
--Query 38:
select fest_name, venue, schedule from college_fest_db.fest
where theme = 'retro'|
```

The results are displayed in a table:

	fest_name	venue	schedule
1	Intaglio	event_venue	10-11-2022
2	novelty	cep 990	11-11-2022
3	bhangraempire	cep 99	11-11-2022
4	Arohan	cep 102	13-11-2022
5	neev	canteen	13-11-2022
6	AGON	OAT	15-11-2022
7	jockersthockers	open air theatre	17-11-2022
8	stars_warslords	lab building	18-11-2022
9	Echoesintheneight	cep 99	19-11-2022
10	seiku	cep 33	20-11-2022

Total rows: 10 of 10 | Query complete 00:00:00.055 | Ln 79, Col 22

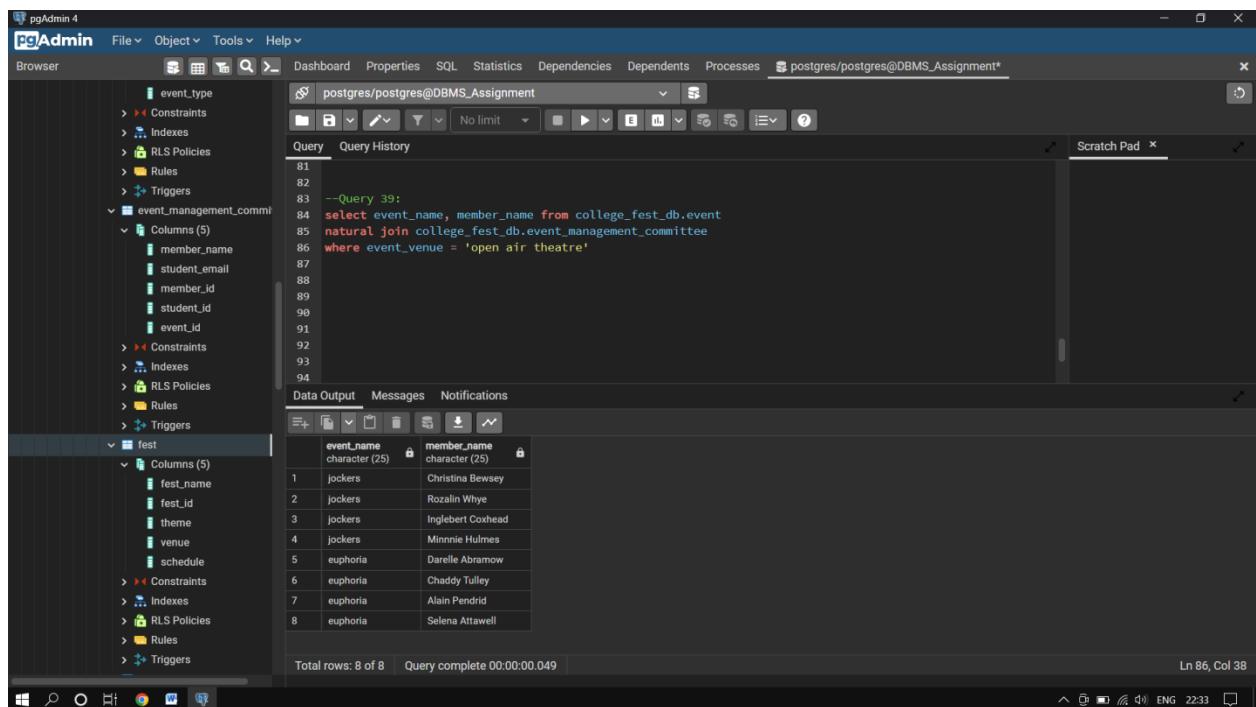
39. English query: Select the following details

- a. Event name
- b. Member name

For events that were hosted on Open Air Theatre

Query:

```
select event_name, member_name from college_fest_db.event
natural join college_fest_db.event_management_committee
where event_venue = 'open air theatre';
```



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

- Browser:** Shows the schema structure of the database, including the `event` and `event_management_committee` tables.
- Query Editor:** Contains the SQL query for question 39.
- Data Output:** Displays the results of the query, showing 8 rows of data.

	event_name	member_name
1	jokers	Christina Bewsey
2	jokers	Rozalin Whye
3	jokers	Inglebert Cothead
4	jokers	Minnie Hulmes
5	euphoria	Darelle Abramow
6	euphoria	Chaddy Tulley
7	euphoria	Alain Pendrid
8	euphoria	Selena Attawell

- Message Bar:** Shows "Total rows: 8 of 8" and "Query complete 00:00:00.049".
- System Bar:** Shows the Windows taskbar with various icons.

40. English query: Display the following details

- a. Event venue
- b. Event name
- c. Event type

Of event that are of type

- a. Fun
- b. OAT Event
- c. Outdoor games

Query:

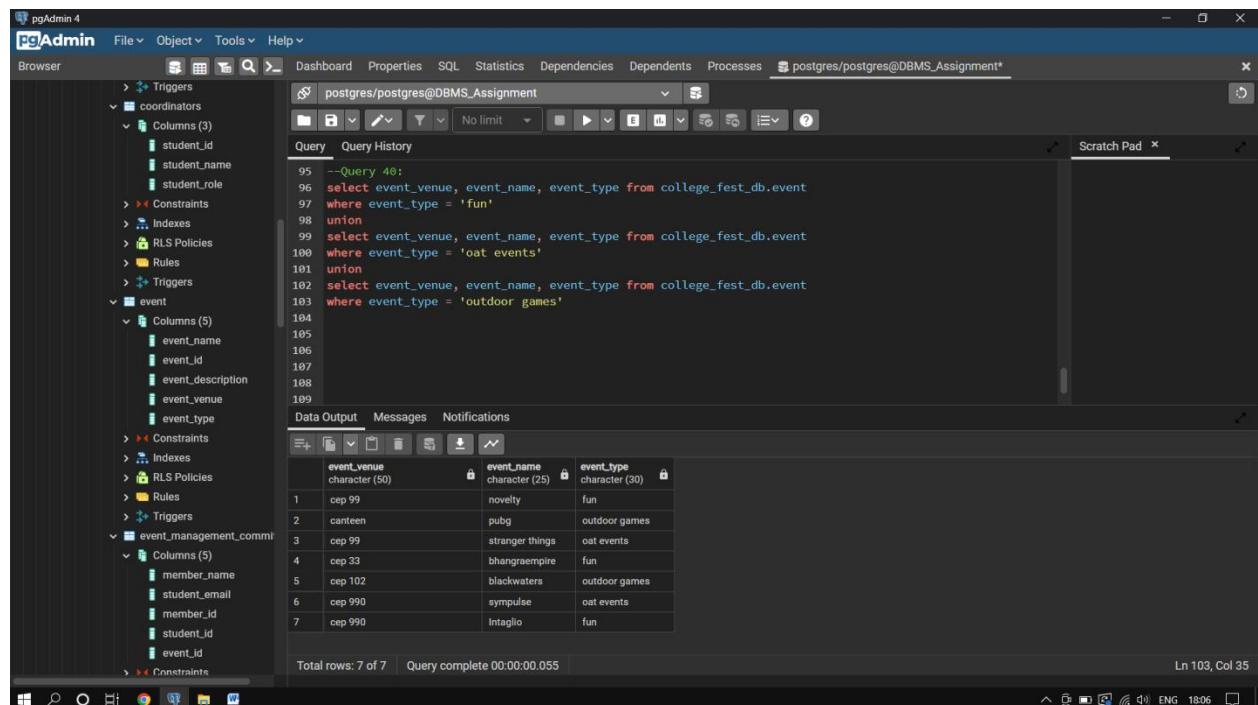
```
select event_venue, event_name, event_type from college_fest_db.event
where event_type = 'fun'
```

union

```
select event_venue, event_name, event_type from college_fest_db.event
where event_type = 'oat events'
```

union

```
select event_venue, event_name, event_type from college_fest_db.event
where event_type = 'outdoor games';
```



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

- Browser:** The left sidebar shows the database structure. Under the 'event' table, there are columns: event_venue, event_name, event_id, event_description, event_venue, and event_type.
- Query Editor:** The main area contains the SQL query for the English query. The results are displayed in a table.
- Table Results:** The table has three columns: event_venue, event_name, and event_type. The data is as follows:

	event_venue	event_name	event_type
1	cep 99	novely	fun
2	canteen	pubg	outdoor games
3	cep 99	stranger things	oat events
4	cep 33	bhengraempire	fun
5	cep 102	blackwaters	outdoor games
6	cep 990	sympulse	oat events
7	cep 990	Intaglio	fun

Total rows: 7 of 7 | Query complete 00:00:00.055 | Ln 103, Col 35