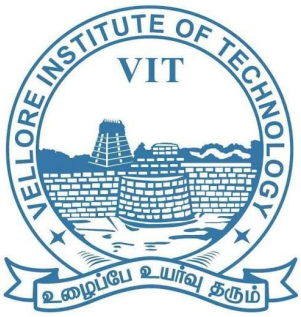


VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

ITE1003: Database Management System

J Component - Review 1

(TITLE - Hackportal)

Faculty:
Bimal Kumar Ray

Submitted By (Team):
Ayush Dwivedi (19BIT0236)
Yash Singhal (19BIT0136)

Introduction

Our DBMS project is a database for a portal for hackathons that makes it easy for organisations to conduct hackathons and individuals to take part in hackathons. In this system, organizations will be able to host hackathons and manage it to some extent and users can register and search for other users and form teams for hackathons. A single user can be a part of multiple teams and can take part in multiple hackathons. Organizations will also be able to find and invite appropriate sponsors, judges, speakers and center for a hackathon. This portal can be used by various engineering, science, tech and commerce sectors to take part in and to conduct hackathons. The system is developed using Oracle as our database management system.

Data Requirements

1. Hackathon

This entity includes details about the hackathon. It contains attributes like hackathon id, hackathon name, hackathon description(rules, eligibility criteria and prizes), important start and end dates, eligible team size (minimum and maximum number of members), number of rounds in hackathon, centre id for centers where hackathon will be held, organisation representative's email who is conducting the hackathon, sponsors(email) of the hackathon and invited judges and speakers' emails.

- Each hackathon has a unique **hack id.**
- We have not taken names as unique so that the same organisation can hold the event next time with the same name.

2. Participant

This entity includes details about the users who are registered in our HackPortal and will be participating in hackathons. It contains their name, email, mobile number, their domain or field of work, where they are from(college, school or company) and the team id of the team in which they are.

- Each user is identified by **their email** which is unique.

3. Team

This entity includes details about the team comprised of users. This contains team id, team name, project details (name, description and its github link), number of members in team and the hackathon id in which the team is participating in.

- **Team ID** will be used to distinguish one team from others.

4. Sponsor

This entity includes details about sponsors of the hackathons and the prizes they will be giving to the top performers. It includes sponsor name, email and details of prizes they will be distributing (prize type and prize name), to whom they later distributed the prize to (team id) and in which hackathon it was distributed (hack id).

- Each sponsor individual or organisation will be distinguished through **their emai**l.

5. Judge

This entity includes the details of the judges who will be grading the teams and their projects. This contains the judge's name, their email, their mobile number, their profession, some description about them (like their work) and the team id of the teams they will be judging plus the hackathon id.

- Each judge is uniquely identified by **their email id**.

6. Organiser

This entity includes the hackathon organiser details like their name, their email and their mobile number and other information.

- Each organizer is uniquely identified by **their email id**.

7. Speaker

This entity includes the invited speakers details which contain their name, their email, mobile number and the organisation

they belong to and some description about them and their work.

- Each speaker is uniquely identified by **their email id**.

8. Center

This entity includes the centre ID , centre name and the address to centre.

- Each center is uniquely identified by their **center id.**

9. Prize

This entity includes the prize ID , prize name and the description about each prize.

- Each prize is uniquely identified by their **prize id.**

10. Each hackathon is conducted at a center.
11. Each hackathon is conducted by an organizer.
12. Each hackathon has a number of sponsors.
13. Each hackathon has a number of judges.
14. Each hackathon has a number of speakers.
15. Participants can be a part of multiple teams.
16. Teams can take part in any number of hackathons.
17. Each prize is given to several teams.
18. Every judge judges at least one team.

19. Organizers allocate speakers for a hackathon.

20. Organizers allocate judges for a hackathon.

21. Organizers allocate sponsors for a hackathon.

22. Organizer finds a center for a hackathon.

Functional Requirements

1. Data Creation

a. **Create Participant Profile** : Participants can create their profile with information like name,email,phone number, domain, college name and their selected team.

b. **Create Organizer Profile:** Organizers can create their organization profile with details like name,email,info and mobile number.

c. **Add Hackathon Details:** Organizers can create hackathon with details like hackathon id,name,description,start and end date, min and max team size, number of rounds to be conducted in hackathon.

d. **Create Team :** Participants can form a team with details like team id,team name, project they are making for a particular hackathon and hackathon ids of the hackathons they are participating in.

e. **Allocate judges for the hackathon :** Organizers can add a judge's information for the hackathon.

f. **Allocate speakers for the hackathon :** Organizers can add speaker’s information for the hackathon.

g. **Allocate sponsors for the hackathon :** Organizers can add sponsor’s information for the hackathon.

h. **Allocate centre for the hackathon :** Organizers can add centre’s information for the hackathon.

i. **Add prize information for the hackathon :** Organizers can add information of various prizes from a sponsor for the hackathon.

j. **Add prize winners for the hackathon :** Organizers can add information of various prizes from a sponsor for the hackathon.

2. Data Modification

a. **Update Participant Profile** : Participants can update their profile information like name,email,phone number, domain, college name and their selected team.

b. **Update Organizer Profile:** Organizers can update their organization profile details like name,email,info and mobile number.

c. **Update Hackathon Details:** Organizers can update hackathon details like hackathon id,name,description,start and end date, min and max team size, number of rounds to be conducted in hackathon.

d. **Update Team :** Participants can update team details like team id,team name, project they are making for a particular hackathon and hackathon ids of the hackathons they are participating in.

e. **Update judges for the hackathon :** Organizers can update the judge's information for the hackathon.

f. **Update speakers for the hackathon :** Organizers can update the speaker’s information for the hackathon.

g. **Update sponsors for the hackathon :** Organizers can update the sponsor’s information for the hackathon.

h. **Update centre for the hackathon :** Organizers can update the centre's information for the hackathon.
i. **Update prize information for the hackathon :** Organizers can update information of various prizes from a sponsor for the hackathon.
j. System can update the description of hackathon to "To be announced" if the description given is NULL.
k. System can update the name of the teams to "No name" if any two teams have the same name.

3. Data Retrieval:

a. **View User Profile:** Participants can view their profile information.

b. **View Organizer Profile:** Organizers can view their profile information.

c. **View Hackathons:** Participants and teams can search a list of hackathons during a particular time period and view information about each hackathon which is conducted by a particular organization.

d. **View Team Details:** All the team members can view team details like the hackathons in which they are participating and the projects they have made for each hack.

e. **View participants of a team:** Everyone can view the members of a particular team.

f. **View winners of a hackathon:** Everyone can see the names of the winning team of a particular hack.

g. **View judges, speakers and sponsors of a hackathon:** Everyone can see the list of judges, speakers and sponsors of a particular hack.

h. **View list of teams for a particular hackathon:** Everyone can see the list of the teams participating in a hackathon.

i. **View centre details of a hackathon:** Everyone can see the details of the centre for a hackathon.

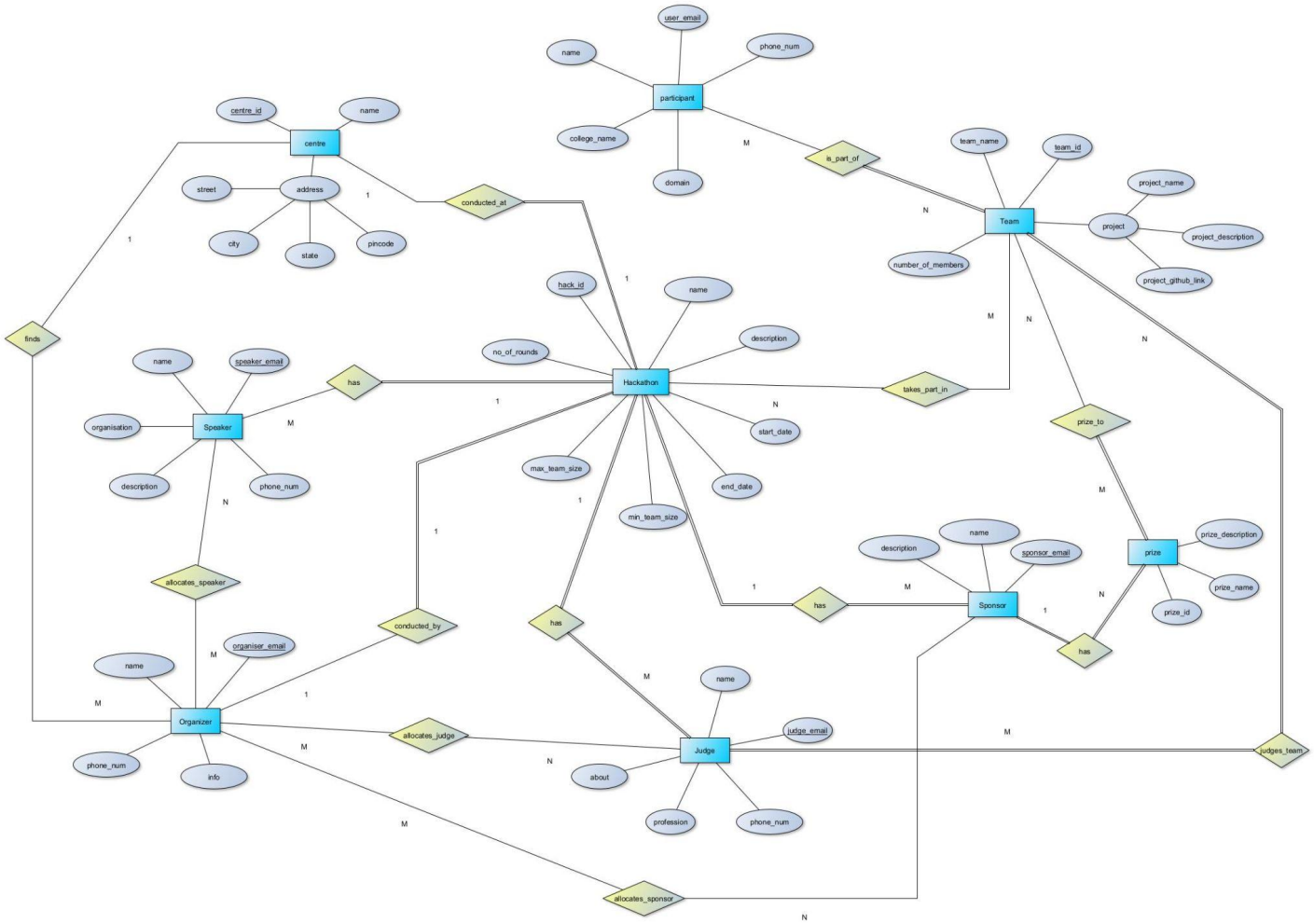
j. Retrieve name of the hackathon with maximum duration,

k. System can show names of the users which are not part of any team.

4. Data Deletion

a. **Delete Participant Profile** : Participants can delete their profile.
b. **Delete Organizer Profile:** Organizers can delete their organization's profile.
c. **Delete Hackathon:** Organizers can delete a hackathon where the number of participating teams is equal to 1.
d. **Delete Team :** Participants can delete their team.
e. System automatically removes teams from a hackathon that violates the number of participants rule for the hackathon.

Entity Relationship Diagram



Review 2

Relationship Database Schema Diagram

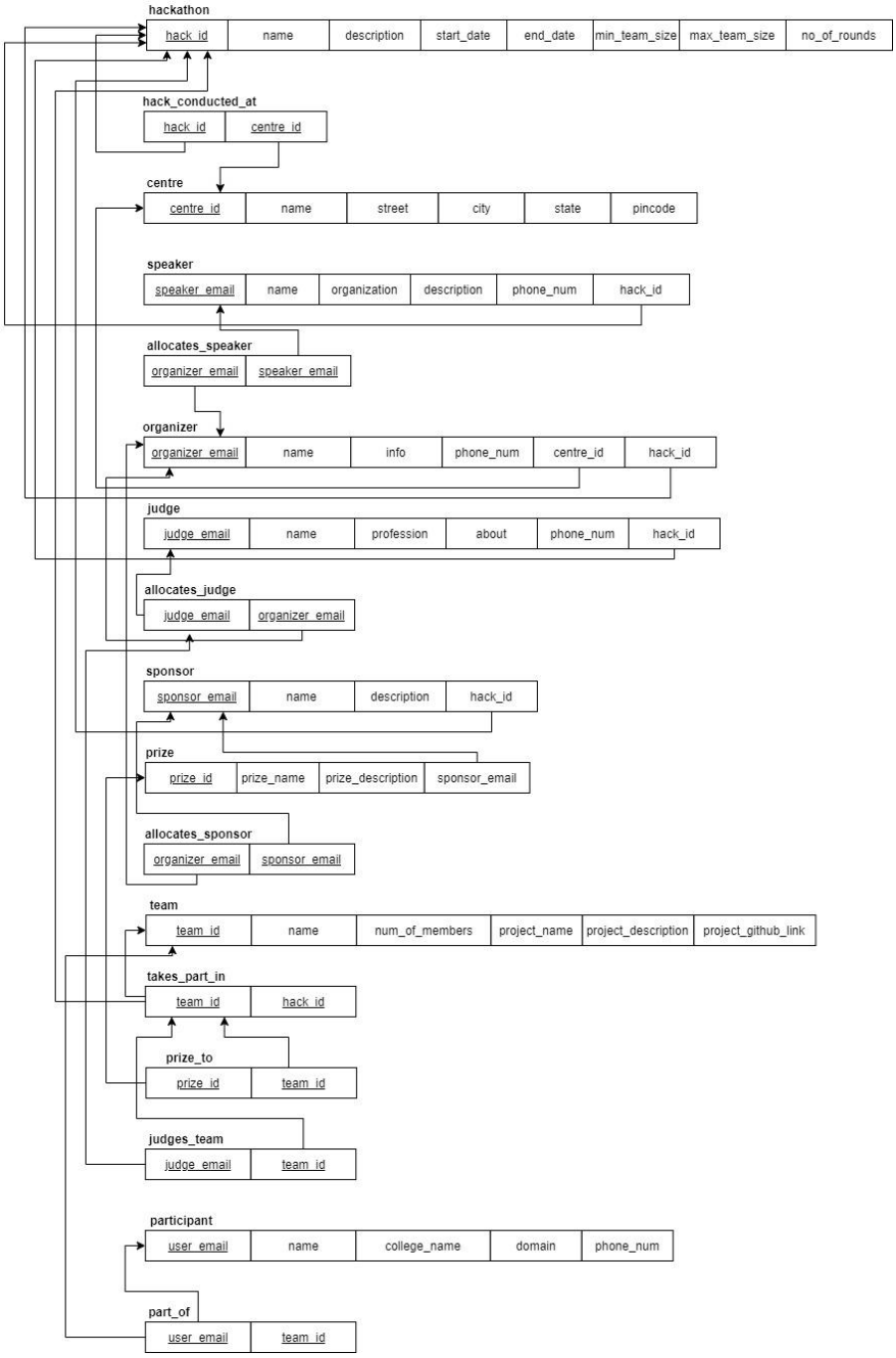


Table creation with Named Integrity constraints (Screenshots)

‘Hackathon’ table creation

```
SQL> CREATE TABLE hackathon(
  2      hack_id VARCHAR(8),
  3      name VARCHAR(20),
  4      description VARCHAR(100),
  5      start_date DATE,
  6      end_date DATE,
  7      min_team_size NUMBER(1),
  8      max_team_size NUMBER(1),
  9      no_of_rounds NUMBER(1),
 10      CONSTRAINT PK_hackathon PRIMARY KEY(hack_id)
 11  );

Table created.

SQL>
```

‘Centre’ table creation

```
SQL> CREATE TABLE CENTRE(
  2      centre_id VARCHAR(4),
  3      name VARCHAR(20),
  4      street VARCHAR(20),
  5      city VARCHAR(20),
  6      state VARCHAR(20),
  7      pincode NUMBER(7),
  8      hack_id VARCHAR(8),
  9      CONSTRAINT PK_centre_id PRIMARY KEY(centre_id),
 10      CONSTRAINT hack_FK FOREIGN KEY(hack_id) REFERENCES hackathon(hack_id) ON DELETE SET NULL
 11  );

Table created.

SQL>
```

‘Speaker’ table creation

```
SQL> CREATE TABLE speaker(
  2      speaker_email VARCHAR(20),
  3      name VARCHAR(20),
  4      organization VARCHAR(20),
  5      description VARCHAR(100),
  6      phone_num NUMBER(12),
  7      hack_id VARCHAR(8),
  8      CONSTRAINT PK_speaker PRIMARY KEY(speaker_email),
  9      CONSTRAINT hack_FK2 FOREIGN KEY(hack_id) REFERENCES hackathon(hack_id) ON DELETE SET NULL,
 10      CONSTRAINT chk_phone CHECK (LENGTH(phone_num)=12)
 11  );

Table created.
```

‘Organizer’ table creation

```
SQL> CREATE TABLE organizer(
  2      organizer_email VARCHAR(20),
  3      name VARCHAR(20),
  4      info VARCHAR(100),
  5      phone_num NUMBER(12),
  6      hack_id VARCHAR(8),
  7      centre_id VARCHAR(8),
  8      CONSTRAINT PK_organizer PRIMARY KEY(organizer_email),
  9      CONSTRAINT hack_FK3 FOREIGN KEY(hack_id) REFERENCES hackathon(hack_id) ON DELETE SET NULL,
 10      CONSTRAINT centre_FK FOREIGN KEY(centre_id) REFERENCES centre(centre_id) ON DELETE SET NULL,
 11      CONSTRAINT chk_phone2 CHECK (LENGTH(phone_num)=12)
 12  );

Table created.

SQL>
```

‘allocates_speaker’ relation table creation

```
SQL> CREATE TABLE allocates_speaker(
  2      organizer_email VARCHAR(20),
  3      speaker_email VARCHAR(20),
  4      CONSTRAINT PK_allocates_speaker PRIMARY KEY(organizer_email,speaker_email),
  5      CONSTRAINT org_FK FOREIGN KEY(organizer_email) REFERENCES organizer(organizer_email) ON DELETE CASCADE,
  6      CONSTRAINT speaker_FK FOREIGN KEY(speaker_email) REFERENCES speaker(speaker_email) ON DELETE CASCADE
  7  );

Table created.

SQL>
```

‘Judge’ table creation

```
SQL> CREATE TABLE judge(
  2      judge_email VARCHAR(20),
  3      name VARCHAR(20),
  4      profession VARCHAR(20),
  5      about VARCHAR(100),
  6      phone_num NUMBER(12),
  7      hack_id VARCHAR(8),
  8      CONSTRAINT PK_judge PRIMARY KEY(judge_email),
  9      CONSTRAINT hack_FK4 FOREIGN KEY(hack_id) REFERENCES hackathon(hack_id) ON DELETE SET NULL
 10  );

Table created.

SQL>
```

‘Allocates_judge’ Relation table creation

```
SQL> CREATE TABLE allocates_judge(
  2      organizer_email VARCHAR(20),
  3      judge_email VARCHAR(20),
  4      CONSTRAINT PK_allocates_judge PRIMARY KEY(organizer_email,judge_email),
  5      CONSTRAINT org_FK2 FOREIGN KEY(organizer_email) REFERENCES organizer(organizer_email) ON DELETE CASCADE,
  6      CONSTRAINT judge_FK FOREIGN KEY(judge_email) REFERENCES judge(judge_email) ON DELETE CASCADE
  7  );

Table created.

SQL>
```

‘Sponsor’ table creation

```
SQL> CREATE TABLE sponsor(
  2      sponsor_email VARCHAR(20),
  3      name VARCHAR(20),
  4      description VARCHAR(100),
  5      hack_id VARCHAR(8),
  6      CONSTRAINT PK_sponsor PRIMARY KEY(sponsor_email),
  7      CONSTRAINT hack_FK5 FOREIGN KEY(hack_id) REFERENCES hackathon(hack_id) ON DELETE SET NULL
  8  );

Table created.

SQL>
```

‘allocates_sponsor’ relation table creation

```
SQL> CREATE TABLE allocates_sponsor(
  2      organizer_email VARCHAR(20),
  3      sponsor_email VARCHAR(20),
  4      CONSTRAINT PK_allocates_sponsor PRIMARY KEY(organizer_email,sponsor_email),
  5      CONSTRAINT org_FK3 FOREIGN KEY(organizer_email) REFERENCES organizer(organizer_email) ON DELETE CASCADE,
  6      CONSTRAINT sponsor_FK2 FOREIGN KEY(sponsor_email) REFERENCES sponsor(sponsor_email) ON DELETE CASCADE
  7  );

Table created.

SQL>
```

‘Prize’ table creation

```
SQL> CREATE TABLE prize(
  2      prize_id VARCHAR(8),
  3      prize_name VARCHAR(20),
  4      prize_description VARCHAR(20),
  5      sponsor_email VARCHAR(20),
  6      CONSTRAINT PK_prize PRIMARY KEY(prize_id),
  7      CONSTRAINT sponsor_FK FOREIGN KEY(sponsor_email) REFERENCES sponsor(sponsor_email) ON DELETE CASCADE
  8  );

Table created.

SQL>
```

‘Team’ table creation

```
SQL> CREATE TABLE team(
  2      team_id VARCHAR(8),
  3      name VARCHAR(20),
  4      num_of_members VARCHAR(20),
  5      project_name VARCHAR(20),
  6      project_description VARCHAR(100),
  7      project_github_link VARCHAR(20),
  8      CONSTRAINT PK_team PRIMARY KEY(team_id)
  9   );

Table created.

SQL>
```

‘Prize_to’ relation table creation

```
SQL> CREATE TABLE prize_to(
  2      prize_id VARCHAR(8),
  3      team_id VARCHAR(8),
  4      CONSTRAINT PK_prize_to PRIMARY KEY(prize_id,team_id),
  5      CONSTRAINT prize_FK FOREIGN KEY(prize_id) REFERENCES prize(prize_id) ON DELETE CASCADE,
  6      CONSTRAINT team_FK2 FOREIGN KEY(team_id) REFERENCES team(team_id) ON DELETE CASCADE
  7   );

Table created.

SQL>
```

‘takes_part_in’ relation table creation

```
SQL> CREATE TABLE takes_part_in(
  2      team_id VARCHAR(8),
  3      hack_id VARCHAR(8),
  4      CONSTRAINT PK_takes_part_in PRIMARY KEY(team_id,hack_id),
  5      CONSTRAINT team_FK FOREIGN KEY(team_id) REFERENCES team(team_id) ON DELETE CASCADE,
  6      CONSTRAINT hack_FK6 FOREIGN KEY(hack_id) REFERENCES hackathon(hack_id) ON DELETE CASCADE
  7  );

Table created.

SQL>
```

‘Judges_team’ relation table creation

```
SQL> CREATE TABLE judges_team(
  2      judge_email VARCHAR(20),
  3      team_id VARCHAR(8),
  4      CONSTRAINT PK_judges_team PRIMARY KEY(judge_email,team_id),
  5      CONSTRAINT judge_FK2 FOREIGN KEY(judge_email) REFERENCES judge(judge_email) ON DELETE CASCADE,
  6      CONSTRAINT team_FK3 FOREIGN KEY(team_id) REFERENCES team(team_id) ON DELETE CASCADE
  7  );

Table created.

SQL>
```

‘Participant’ table creation

```
SQL> CREATE TABLE participant(
  2      user_email VARCHAR(20),
  3      name VARCHAR(20),
  4      college_name VARCHAR(20),
  5      domain VARCHAR(20),
  6      phone_num NUMBER(12),
  7      CONSTRAINT PK_user PRIMARY KEY(user_email)
  8  );

Table created.

SQL>
```

‘part_of’ relation table creation

```
SQL> CREATE TABLE part_of(
  2      user_email VARCHAR(20),
  3      team_id VARCHAR(8),
  4      CONSTRAINT PK_part_of PRIMARY KEY(user_email,team_id),
  5      CONSTRAINT team_FK4 FOREIGN KEY(team_id) REFERENCES team(team_id) ON DELETE CASCADE
  6  );

Table created.

SQL>
```

Data Insertion Screenshots

Insertion Into ‘Hackathon’ Table

```
SQL> INSERT INTO hackathon VALUES (
  2  '1111AAAA',
  3  'Equinox 2021',
  4  'Equinox aims to provide 36 uninterrupted hours of ideation and innovation. Hackers will receive a
platform with the necessary resources to put forth their ideas and skills. In Equinox, imagination is no
t limited by specific problem statements, In addition to it hackers are given the intellectual freedom t
o obliterate the boundaries of their imaginative power and tap into their creativity to come up with uni
que solutions to the problems, they see fit.',
  5  TO_DATE('2021-06-25','YYYY-MM-DD'),
  6  TO_DATE('2021-06-27','YYYY-MM-DD'),
  7  1,
  8  4,
  9  3
 10  );

1 row created.

SQL>
```

```
SQL> INSERT INTO hackathon VALUES (
  2  '2222BBBB',
  3  'Citython Eindhoven',
  4  'This online Citython edition will be held in the city of Eindhoven, The Netherlands, and is focuse
d on mobility optimization, traffic safety areas and a healthy city future. The goal of this Citython is
 that young professionals search for innovative and creative solutions that can be applied in the city o
f Eindhoven to foster innovation.',
  5  TO_DATE('2021-07-2','YYYY-MM-DD'),
  6  TO_DATE('2021-07-3','YYYY-MM-DD'),
  7  1,
  8  3,
  9  2
 10  );

1 row created.
```

```
SQL> INSERT INTO hackathon VALUES (
  2  '3333CCCC',
  3  'AI Fair',
  4  'AI Community is seeking projects across the world, designed and developed by high and middle schoo
l students. These projects should try to solve problems related to community or the environment and use
technology or AI to solve the problems. The fair is open to all high and middle school students who will
 solve a community or environment focused problem using technology. Top projects will be showcased on th
e website and also recognized with awards, scholarships/prizes, and mentorship',
  5  TO_DATE('2021-06-5','YYYY-MM-DD'),
  6  TO_DATE('2021-06-9','YYYY-MM-DD'),
  7  3,
  8  5,
  9  4
 10  );

1 row created.
```

Inserted Data

```
SQL> select * from hackathon;

HACK_ID  NAME                 DESCRIPTION


                                                    START_DAT END_DATE  MIN_TEAM_SIZE MAX_TEA
M_SIZE NO_OF_ROUNDS
-------- -------------------- ------------------------------------------------------------
------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
---------------------------------------------------------- -------- -------- -------------
 ------------ ------------
1111AAAA Equinox 2021         Equinox aims to provide 36 uninterrupted hours of ideation and i
nnovation. Hackers will receive a platform with the necessary resources to put forth their ide
as and skills. In Equinox, imagination is not limited by specific problem statements, In addit
ion to it hackers are given the intellectual freedom to obliterate the boundaries of their ima
ginative power and tap into their creativity to come up with unique solutions to the problems,
 they see fit.                                                  25-JUN-21 27-JUN-21
 1            4            3
2222BBBB Citython Eindhoven   This online Citython edition will be held in the city of Eindhov
en, The Netherlands, and is focused on mobility optimization, traffic safety areas and a healt
hy city future. The goal of this Citython is that young professionals search for innovative an
d creative solutions that can be applied in the city of Eindhoven to foster innovation.

                                           02-JUL-21 03-JUL-21            1            3
          2
3333CCCC AI Fair              AI Community is seeking projects across the world, designed and
developed by high and middle school students. These projects should try to solve problems rela
ted to community or the environment and use technology or AI to solve the problems. The fair i
s open to all high and middle school students who will solve a community or environment focuse
d problem using technology. Top projects will be showcased on the website and also recognized
with awards, scholarships/prizes, and mentorship              05-JUN-21 09-JUN-21
 3            5            4

SQL>
```

Insertion Into 'Centre' Table

```
SQL> INSERT INTO centre VALUES(
  2  '11AA',
  3  'Anna Auditorium',
  4  'VIT',
  5  'Vellore',
  6  'Tamil Nadu',
  7  632014,
  8  '1111AAAA'
  9  );

1 row created.
```

```
SQL> INSERT INTO centre VALUES(
  2  '22BB',
  3  'ST. George College',
  4  'Nehru Enclave',
  5  'Agra',
  6  'Uttar Pradesh',
  7  282001,
  8  '2222BBBB'
  9  );

1 row created.
```

```
SQL> INSERT INTO centre VALUES(
  2  '33CC',
  3  'Computer Centre',
  4  'IIIT Delhi',
  5  'Delhi',
  6  'Delhi',
  7  110020,
  8  '3333CCCC'
  9  );

1 row created.
```

Inserted Data

```
SQL> select * from centre;

CENT NAME                 STREET              CITY                STATE                PINCODE HACK_ID
---- -------------------- ------------------- ------------------- ------------------- ---------- --------
22BB ST. George College   Nehru Enclave       Agra                Uttar Pradesh           282001 2222BBBB
33CC Computer Centre      IIIT Delhi          Delhi               Delhi                   110020 3333CCCC
11AA Anna Auditorium      VIT                 Vellore             Tamil Nadu              632014 1111AAAA
```

Insertion Into 'Speaker' Table

```
SQL> INSERT INTO speaker VALUES(
  2  'umang.s@vit.ac.in',
  3  'Umang Singh',
  4  'VITLUG, VIT',
  5  'Project Head of VIT Linux User Group, begin a startup in 2nd year',
  6  '911234567891',
  7  '1111AAAA'
  8  );

1 row created.
```

```
SQL> INSERT INTO speaker VALUES(
  2  'pravesh@ms.co.in',
  3  'Pravesh Sharma',
  4  'Microsoft India',
  5  'Senior Developer at Microsoft, India. Currently working on WSL. Previous
ly worled on Visual Studio',
  6  '911234567892',
  7  '2222BBBB'
  8  );

1 row created.
```

```
SQL> INSERT INTO speaker VALUES(
  2  'abhi.nav@openAI.us',
  3  'Abhinav Dubey',
  4  'Open AI',
  5  'Top Researcher at OpenAI, the world leading AI Company. Currently Workin
g GPT3',
  6  '911234567893',
  7  '3333CCCC'
  8  );

1 row created.
```

Inserted Data

```
SQL> select * from speaker;

SPEAKER_EMAIL        NAME                ORGANIZATION        DESCRIPTION
                                                                                PHONE
_NUM HACK_ID
-------------------- ------------------- ------------------- --------------------
-------------------------------------------------------------------------------- ----
------ --------
umang.s@vit.ac.in    Umang Singh         VITLUG, VIT         Project Head of VIT L
inux User Group, begin a startup in 2nd year                                    9.1123
E+11 1111AAAA
pravesh@ms.co.in     Pravesh Sharma      Microsoft India     Senior Developer at M
icrosoft, India. Currently working on WSL. Previously worled on Visual Studio   9.11
23E+11 2222BBBB
abhi.nav@openAI.us   Abhinav Dubey       Open AI             Top Researcher at Ope
nAI, the world leading AI Company. Currently Working GPT3                       9.1123
E+11 3333CCCC

SQL>
```

Insertion Into 'Organiser' Table

```
SQL> INSERT INTO organizer VALUES(
  2  'ayush.dwi@vit.ac.in',
  3  'Ayush Dwivedi',
  4  'A member of Management team and a core part of it',
  5  '911234567894',
  6  '1111AAAA',
  7  '11AA'
  8  );

1 row created.
```

```
SQL> INSERT INTO organizer VALUES(
  2  'yash.s@gmail.com',
  3  'Yash Singh',
  4  'A member of Student welfare community at Agra College',
  5  '911234567895',
  6  '2222BBBB',
  7  '33CC'
  8  );

1 row created.
```

```
SQL> INSERT INTO organizer VALUES(
  2  'hvon@outlook.com',
  3  'Harvey Glockner',
  4  'President of DSC Management team at IIIT Delhi',
  5  '911234567896',
  6  '3333CCCC',
  7  '33CC'
  8  );

1 row created.
```

Inserted Data

```
SQL> select * from organizer;

ORGANIZER_EMAIL      NAME                 INFO
                                                                    PHONE_NUM HACK_ID
CENT
-------------------- -------------------- ----------------------------------------
---------------------------------------------------------- ---------- -------- ---
-
ayush.dwi@vit.ac.in  Ayush Dwivedi        A member of Management team and a core pa
rt of it                                                      9.1123E+11 1111AAAA 11AA
yash.s@gmail.com     Yash Singh           A member of Student welfare community at
Agra College                                                  9.1123E+11 2222BBBB 33CC
hvon@outlook.com     Harvey Glockner      President of DSC Management team at IIIT
Delhi                                                         9.1123E+11 3333CCCC 33CC

SQL>
```

Insertion Into ‘allocates_speaker’ Relation Table

```
SQL> INSERT INTO allocates_speaker VALUES(
  2  'ayush.dwi@vit.ac.in',
  3  'umang.s@vit.ac.in'
  4  );

1 row created.
```

```
SQL> INSERT INTO allocates_speaker VALUES(
  2  'yash.s@gmail.com',
  3  'pravesh@ms.co.in'
  4  );

1 row created.
```

```
SQL> INSERT INTO allocates_speaker VALUES(
  2  'hvon@outlook.com',
  3  'abhi.nav@openAI.us'
  4  );

1 row created.
```

Inserted Data

```
SQL> select * from allocates_speaker;

ORGANIZER_EMAIL      SPEAKER_EMAIL
-------------------- --------------------
ayush.dwi@vit.ac.in  umang.s@vit.ac.in
hvon@outlook.com     abhi.nav@openAI.us
yash.s@gmail.com     pravesh@ms.co.in

SQL>
```

Insertion Into 'Judge' Table

```
SQL> INSERT INTO judge VALUES(
  2  'judge1@gamil.com',
  3  'Satyam Pachaoury',
  4  'Senior Developer',
  5  'Performs various development duties, and with specialization as web development',
  6  '911234567897',
  7  '1111AAAA'
  8  );

1 row created.
```

```
SQL> INSERT INTO judge VALUES(
  2  'judge2@gamil.com',
  3  'Pratham Sharma',
  4  'Project Manager',
  5  'Manages various project duties at Microsoft, India',
  6  '911234567898',
  7  '2222BBBB'
  8  );

1 row created.
```

```
SQL> INSERT INTO judge VALUES(
  2  'judge3@gamil.com',
  3  'Pratham Jha',
  4  'AI Developer',
  5  'A very reputed person in AI community',
  6  '911234567899',
  7  '3333CCCC'
  8  );

1 row created.
```

Inserted Data

```
SQL> select * from Judge;

JUDGE_EMAIL          NAME                 PROFESSION           ABOUT

  PHONE_NUM HACK_ID
-------------------- -------------------- -------------------- --------------------
-------------------------------------------------------------------------------- --
-------- --------
judge1@gamil.com     Satyam Pachaoury     Senior Developer     Performs various dev
elopment duties, and with specialization as web development
 9.1123E+11 1111AAAA
judge2@gamil.com     Pratham Sharma       Project Manager      Manages various proj
ect duties at Microsoft, India                                          9.1123E
+11 2222BBBB
judge3@gamil.com     Pratham Jha          AI Developer         A very reputed perso
n in AI community
 9.1123E+11 3333CCCC

SQL>
```

Insertion Into ‘allocates_judge’ Relation Table

```
SQL> INSERT INTO allocates_judge VALUES(
  2  'ayush.dwi@vit.ac.in',
  3  'judge1@gamil.com'
  4  );

1 row created.
```

```
SQL> INSERT INTO allocates_judge VALUES(
  2  'yash.s@gmail.com',
  3  'judge2@gamil.com'
  4  );

1 row created.
```

```
SQL> INSERT INTO allocates_judge VALUES(
  2  'hvon@outlook.com',
  3  'judge3@gamil.com'
  4  );

1 row created.
```

Inserted Data

```
SQL> select * from allocates_judge;

ORGANIZER_EMAIL      JUDGE_EMAIL
-------------------- --------------------
ayush.dwi@vit.ac.in  judge1@gamil.com
hvon@outlook.com     judge3@gamil.com
yash.s@gmail.com     judge2@gamil.com

SQL>
```

Insertion Into ‘Sponsor’ Table

```
SQL> INSERT INTO sponsor VALUES(
  2  'sponsor1@gmail.com',
  3  'Linode',
  4  'Linode is an American cloud hosting company that provides virtual private servers',
  5  '1111AAAA'
  6  );

1 row created.
```

```
SQL> INSERT INTO sponsor VALUES(
  2  'sponsor2@gmail.com',
  3  'Microsoft',
  4  'Microsoft develops, manufactures, supports, and sells software, electronics, PCs
, related services',
  5  '2222BBBB'
  6  );

1 row created.
```

```
SQL> INSERT INTO sponsor VALUES(
  2  'sponsor3@gmail.com',
  3  'OpenAI',
  4  'OpenAI is research and deployment company with mission to ensure that AGI benefi
ts all life',
  5  '3333CCCC'
  6  );

1 row created.
```

Inserted Data

```
SQL> select * from Sponsor;

SPONSOR_EMAIL        NAME                 DESCRIPTION
                                                                        HACK_ID
-------------------- -------------------- ----------------------------------------
---------------------------------------------------------------- --------
sponsor1@gmail.com   Linode               Linode is an American cloud hosting compa
ny that provides virtual private servers                                1111AAAA
sponsor2@gmail.com   Microsoft            Microsoft develops, manufactures, support
s, and sells software, electronics, PCs, related services   2222BBBB
sponsor3@gmail.com   OpenAI               OpenAI is research and deployment company
 with mission to ensure that AGI benefits all life             3333CCCC

SQL>
```

Insertion Into 'allocates_sponsor' Relation Table

```
SQL> INSERT INTO allocates_sponsor VALUES(
  2  'ayush.dwi@vit.ac.in',
  3  'sponsor1@gmail.com'
  4  );

1 row created.

SQL> INSERT INTO allocates_sponsor VALUES(
  2  'yash.s@gmail.com',
  3  'sponsor2@gmail.com'
  4  );

1 row created.

SQL> INSERT INTO allocates_sponsor VALUES(
  2  'hvon@outlook.com',
  3  'sponsor3@gmail.com'
  4  );

1 row created.
```

Inserted Data

```
SQL> select * from allocates_sponsor;

ORGANIZER_EMAIL      SPONSOR_EMAIL
-------------------- --------------------
ayush.dwi@vit.ac.in  sponsor1@gmail.com
hvon@outlook.com     sponsor3@gmail.com
yash.s@gmail.com     sponsor2@gmail.com

SQL>
```

Insertion Into ‘prize’ Table

```
SQL> INSERT INTO prize VALUES(
  2  'P0000001',
  3      '1st Prize',
  4      '$250 and Mentorship',
  5      'sponsor1@gmail.com'
  6  );

1 row created.

SQL> INSERT INTO prize VALUES(
  2  'P0000002',
  3      '2nd Prize',
  4      '$150 and Mentorship',
  5      'sponsor1@gmail.com'
  6  );

1 row created.

SQL> INSERT INTO prize VALUES(
  2  'P0000003',
  3      '3rd Prize',
  4      '$75 and Goodies',
  5      'sponsor1@gmail.com'
  6  );

1 row created.
```

```
SQL> INSERT INTO prize VALUES(
  2  'P0000004',
  3      '1st Prize',
  4      'Microsoft GO Tab',
  5      'sponsor2@gmail.com'
  6  );

1 row created.

SQL> INSERT INTO prize VALUES(
  2  'P0000005',
  3      '2nd Prize',
  4      'Azure 200 credits',
  5      'sponsor2@gmail.com'
  6  );

1 row created.

SQL> INSERT INTO prize VALUES(
  2  'P0000006',
  3      '3rd Prize',
  4      'Azure 100 credits',
  5      'sponsor2@gmail.com'
  6  );

1 row created.
```

```
SQL> INSERT INTO prize VALUES(
  2  'P0000007',
  3      '1st Prize',
  4      '$150 and Free Course',
  5      'sponsor3@gmail.com'
  6  );

1 row created.

SQL> INSERT INTO prize VALUES(
  2  'P0000008',
  3      '2nd Prize',
  4      '$75 and Free Course',
  5      'sponsor3@gmail.com'
  6  );

1 row created.

SQL> INSERT INTO prize VALUES(
  2  'P0000009',
  3      '3rd Prize',
  4      '$50 and Goodies',
  5      'sponsor3@gmail.com'
  6  );

1 row created.
```

Inserted Data

```
SQL> select * from prize;

PRIZE_ID PRIZE_NAME          PRIZE_DESCRIPTION

                                                    SPONSOR_EMAIL
-------- ------------------- --------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
------ --------------------
P0000001 1st Prize           $250 and Mentorship

                                                    sponsor1@gmail.com
P0000002 2nd Prize           $150 and Mentorship

                                                    sponsor1@gmail.com
P0000003 3rd Prize           $75 and Goodies

                                                    sponsor1@gmail.com
P0000004 1st Prize           Microsoft GO Tab

                                                    sponsor2@gmail.com
P0000005 2nd Prize           Azure 200 credits

                                                    sponsor2@gmail.com
P0000006 3rd Prize           Azure 100 credits

                                                    sponsor2@gmail.com
P0000007 1st Prize           $150 and Free Course

                                                    sponsor3@gmail.com
P0000008 2nd Prize           $75 and Free Course

                                                    sponsor3@gmail.com
P0000009 3rd Prize           $50 and Goodies

                                                    sponsor3@gmail.com

9 rows selected.

SQL>
```

Insertion Into ‘team’ Table

```
SQL> INSERT INTO team VALUES(
  2  'T0000001',
  3      'Block Bandits',
  4      3,
  5      'TraceIt',
  6  'TraceIt aims to revolutionize Poultry Market using blockchain providing traceability and trust',
  7  'github.com/team1'
  8  );

1 row created.
```

```
SQL> INSERT INTO team VALUES(
  2  'T0000002',
  3      'Team404',
  4      1,
  5      'Food Site',
  6  'A site where college students can subscribe for food service near college',
  7  'github.com/team2'
  8  );

1 row created.
```

```
SQL> INSERT INTO team VALUES(
  2  'T0000003',
  3      'AI_Dreamers',
  4      4,
  5      'Best Map Route',
  6  'A ML based software that finds the best route on map with least travel time',
  7  'github.com/team3'
  8  );

1 row created.
```

Inserted Data

```
SQL> select * from team;

TEAM_ID  NAME                NUM_OF_MEMBERS PROJECT_NAME         PROJECT_DESCRIPTION
                                                                 PROJECT_GITHUB_LINK
-------- ------------------- -------------- -------------------- ----------------------------------
------------------------------------------------------------ --------------------
T0000001 Block Bandits                    3 TraceIt              TraceIt aims to revolutionize Poultr
y Market using blockchain providing traceability and trust       github.com/team1
T0000002 Team404                          1 Food Site            A site where college students can su
bscribe for food service near college                            github.com/team2
T0000003 AI_Dreamers                      4 Best Map Route       A ML based software that finds the b
est route on map with least travel time                          github.com/team3

SQL> _
```

Insertion Into ‘prize_to’ Relation Table

```
SQL> INSERT INTO prize_to VALUES(
  2  'P0000001',
  3  'T0000001'
  4  );

1 row created.

SQL> INSERT INTO prize_to VALUES(
  2  'P0000005',
  3  'T0000002'
  4  );

1 row created.

SQL> INSERT INTO prize_to VALUES(
  2  'P0000009',
  3  'T0000003'
  4  );

1 row created.
```

Inserted Data

```
SQL> select * from prize_to;

PRIZE_ID TEAM_ID
-------- --------
P0000001 T0000001
P0000005 T0000002
P0000009 T0000003

SQL>
```

Insertion Into ‘takes_part_in’ Relation Table

```
SQL> INSERT INTO takes_part_in VALUES(
  2  'T0000001',
  3  '1111AAAA'
  4  );

1 row created.

SQL> INSERT INTO takes_part_in VALUES(
  2  'T0000002',
  3  '2222BBBB'
  4  );

1 row created.

SQL> INSERT INTO takes_part_in VALUES(
  2  'T0000003',
  3  '3333CCCC'
  4  );

1 row created.
```

Inserted Data

```
SQL> select * from takes_part_in;

TEAM_ID  HACK_ID
-------- --------
T0000001 1111AAAA
T0000002 2222BBBB
T0000003 3333CCCC

SQL>
```

Insertion Into 'judges_team' Relation Table

```
SQL> INSERT INTO judges_team VALUES(
  2  'judge1@gamil.com',
  3  'T0000001'
  4  );

1 row created.

SQL> INSERT INTO judges_team VALUES(
  2  'judge2@gamil.com',
  3  'T0000002'
  4  );

1 row created.

SQL> INSERT INTO judges_team VALUES(
  2  'judge3@gamil.com',
  3  'T0000003'
  4  );

1 row created.
```

Inserted Data

```
SQL> select * from judges_team;

JUDGE_EMAIL          TEAM_ID
-------------------- --------
judge1@gamil.com     T0000001
judge2@gamil.com     T0000002
judge3@gamil.com     T0000003

SQL>
```

Insertion Into ‘participant’ Table

```
SQL> INSERT INTO participant VALUES(
  2  'user1.1@email.com',
  3      'Ayush Dubey',
  4      'VIT, Vellore',
  5      'Frontend Dev',
  6      911234567990
  7  );

1 row created.

SQL> INSERT INTO participant VALUES(
  2  'user1.2@email.com',
  3      'Ayush Dubey',
  4      'VIT, Vellore',
  5      'Frontend Dev',
  6      911234567990
  7  );

1 row created.

SQL> INSERT INTO participant VALUES(
  2  'user1.3@email.com',
  3      'Ayush Dubey',
  4      'VIT, Vellore',
  5      'Frontend Dev',
  6      911234567990
  7  );

1 row created.

SQL>
```

```
SQL> INSERT INTO participant VALUES(
  2  'user2@email.com',
  3      'Ankit Sharma',
  4      'Agra College',
  5      'Blockchain Dev',
  6      911234567991
  7  );

1 row created.
```

```
SQL> INSERT INTO participant VALUES(
  2  'user3.1@email.com',
  3      'Piyush Jain',
  4      'IIIT Delhi',
  5      'Backend Dev',
  6      911234567992
  7  );

1 row created.

SQL> INSERT INTO participant VALUES(
  2  'user3.2@email.com',
  3      'Piyush Jain',
  4      'IIIT Delhi',
  5      'Backend Dev',
  6      911234567992
  7  );

1 row created.
```

```
SQL> INSERT INTO participant VALUES(
  2  'user3.4@email.com',
  3      'Piyush Jain',
  4      'IIIT Delhi',
  5      'Backend Dev',
  6      911234567992
  7  );

1 row created.

SQL> INSERT INTO participant VALUES(
  2  'user3.3@email.com',
  3      'Piyush Jain',
  4      'IIIT Delhi',
  5      'Backend Dev',
  6      911234567992
  7  );

1 row created.

SQL>
```

Inserted Data

```
SQL> select * from participant;

USER_EMAIL           NAME                 COLLEGE_NAME         DOMAIN               PHONE_NUM
-------------------- -------------------- -------------------- -------------------- ----------
user1.1@email.com    Ayush Dubey          VIT, Vellore         Frontend Dev         9.1123E+11
user1.2@email.com    Ayush Dubey          VIT, Vellore         Frontend Dev         9.1123E+11
user1.3@email.com    Ayush Dubey          VIT, Vellore         Frontend Dev         9.1123E+11
user2@email.com      Ankit Sharma         Agra College         Blockchain Dev       9.1123E+11
user3.1@email.com    Piyush Jain          IIIT Delhi           Backend Dev          9.1123E+11
user3.2@email.com    Piyush Jain          IIIT Delhi           Backend Dev          9.1123E+11
user3.4@email.com    Piyush Jain          IIIT Delhi           Backend Dev          9.1123E+11
user3.3@email.com    Piyush Jain          IIIT Delhi           Backend Dev          9.1123E+11

8 rows selected.

SQL>
```

Insertion Into ‘part_of’ Relation Table

```
SQL> INSERT INTO part_of VALUES(
  2  'user1.1@email.com',
  3  'T0000001'
  4  );

1 row created.

SQL> INSERT INTO part_of VALUES(
  2  'user1.2@email.com',
  3  'T0000001'
  4  );

1 row created.

SQL> INSERT INTO part_of VALUES(
  2  'user1.3@email.com',
  3  'T0000001'
  4  );

1 row created.

SQL>
```

```
SQL> INSERT INTO part_of VALUES(
  2  'user2@email.com',
  3  'T0000002'
  4  );

1 row created.

SQL>
```

```
SQL> INSERT INTO part_of VALUES(
  2  'user3.1@email.com',
  3  'T0000003'
  4  );

1 row created.

SQL> INSERT INTO part_of VALUES(
  2  'user3.2@email.com',
  3  'T0000003'
  4  );

1 row created.

SQL> INSERT INTO part_of VALUES(
  2  'user3.3@email.com',
  3  'T0000003'
  4  );

1 row created.

SQL> INSERT INTO part_of VALUES(
  2  'user3.4@email.com',
  3  'T0000003'
  4  );

1 row created.

SQL>
```

Inserted Data

```
SQL> SELECT * FROM part_of;

USER_EMAIL           TEAM_ID
-------------------- --------
user1.1@email.com    T0000001
user1.2@email.com    T0000001
user1.3@email.com    T0000001
user2@email.com      T0000002
user3.1@email.com    T0000003
user3.2@email.com    T0000003
user3.3@email.com    T0000003
user3.4@email.com    T0000003

8 rows selected.
```

Review 3

SQL QUERIES (Screenshots)

Data Retrieval

1. Retrieving list of hackathons during a particular duration (order by clause)

```
SQL> SELECT * FROM hackathon WHERE (start_date >= TO_DATE('2021-06-20','YYYY-MM-DD')) and (end_date <= TO_DATE('2021-06-30','YYYY-MM-DD')) order by name desc;

HACK_ID  NAME                 DESCRIPTION                                                                      START_DAT END_DATE  MIN_TEAM_SIZE MAX_TEAM_SIZE NO_OF_ROUNDS
-------- -------------------- -------------------------------------------------------------------------------- --------- --------- ------------- ------------- ------------
1111AAAA Equinox 2021         Equinox aims to provide 36 uninterrupted hours of ideation and innovation. Hacke 25-JUN-21 27-JUN-21             1             4            3
                              rs will receive a platform with the necessary resources to put forth their ideas
                               and skills. In Equinox, imagination is not limited by specific problem statemen
                              ts, In addition to it hackers are given the intellectual freedom to obliterate t
                              he boundaries of their imaginative power and tap into their creativity to come u
                              p with unique solutions to the problems, they see fit.
```

2. Retrieving a participant profile (use of nvl func)

```
SQL> SELECT user_email,name,nvl(college_name,'Not Provided') as college_name,domain,phone_num FROM participant WHERE user_email = 'user2@email.com';

USER_EMAIL           NAME                 COLLEGE_NAME         DOMAIN               PHONE_NUM
-------------------- -------------------- -------------------- -------------------- ------------
user2@email.com      Ankit Sharma         Agra College         Blockchain Dev       911234567991
```

3. Retrieving hackathon ids for hackathons having no. of registered teams=1 (use of join,group by and having)

```
SQL> select hackathon.hack_id FROM hackathon LEFT JOIN takes_part_in ON hackathon.hack_id = takes_part_in.hack_id GROUP BY hackathon.hack_id HAVING count(takes_part_in.team_id) = 1;

HACK_ID
--------
1111AAAA
3333CCCC
```

4. Retrieving team details (use of nullif func)

```
SQL> select team_id,name,num_of_members,project_name,COALESCE(NULLIF(project_description,to_char(NULL)),'Unknown') as project_description,project_github_link from team where team_id='T0000004';

TEAM_ID  NAME                 NUM_OF_MEMBERS PROJECT_NAME         PROJECT_DESCRIPTION                                                                                  PROJECT_GITHUB_LINK
-------- -------------------- -------------- -------------------- ---------------------------------------------------------------------------------------------------- --------------------
T0000004 Noob squad                       2 Educator.ai          A ML based software that finds the best teachers                                                     github.com/team4
```

5. Retrieving the names of participants who are a part of a team in descending order (correlated nested query)

```
SQL> select * from participant where user_email in (select user_email from part_of where part_of.user_email = participant.user_email) order by name desc;

USER_EMAIL          NAME                COLLEGE_NAME        DOMAIN              PHONE_NUM
------------------- ------------------- ------------------- ------------------- ------------
user1.2@email.com   Yash Singhal        VIT, Vellore        Frontend Dev        911234567990
user1.1@email.com   Vaibhav             VIT, Vellore        Frontend Dev        911234567990
user3.4@email.com   Shyam               IIIT Delhi          Backend Dev         911234567992
user3.3@email.com   Rahul               IIIT Delhi          Backend Dev         911234567992
user3.2@email.com   Om                  IIIT Delhi          Backend Dev         911234567992
user3.1@email.com   Harsh               IIIT Delhi          Backend Dev         911234567992
user1.3@email.com   Ayush Dubey         VIT, Vellore        Frontend Dev        911234567990
user2@email.com     Ankit Sharma        Agra College        Blockchain Dev      911234567991

8 rows selected.
```

6. Retrieving names of judges, speakers and sponsors for a hackathon (use of set operator)

```
SQL> select name as names from sponsor where hack_id='1111AAAA' UNION (select name from judge where hack_id='1111AAAA' UNION select name from speaker where hack_id='1111AAAA');

NAMES
--------------------
Linode
Satyam Pachaoury
Umang Singh
```

7. Retrieving all teams for a particular hackathon (uncorrelated nested query)

```
SQL> select * from team where team_id in (Select team_id from takes_part_in where hack_id = '1111AAAA');

TEAM_ID  NAME                NUM_OF_MEMBERS PROJECT_NAME        PROJECT_DESCRIPTION                                                                                  PROJECT_GITHUB_LINK
-------- ------------------- -------------- ------------------- ---------------------------------------------------------------------------------------------------- -------------------
T0000001 Block Bandits                    3 TraceIt             TraceIt aims to revolutionize Poultry Market using blockchain providing traceability and trust       github.com/team1
T0000004 Noob squad                       2 Educator.ai         A ML based software that finds the best teachers                                                     github.com/team4
```

8. Retrieving centre details for a hackathon (where clause)

```
SQL> SELECT * FROM centre WHERE hack_id = '1111AAAA';

CENT NAME                STREET              CITY                STATE               PINCODE HACK_ID
---- ------------------- ------------------- ------------------- ------------------- ------- --------
11AA Anna Auditorium     VIT                 Vellore             Tamil Nadu          632014  1111AAAA
```

9. Retrieving list of all participants and their corresponding team ids (full outer join)

```
SQL> Select participant.name,NVL(team.team_id,'no team') as team_id from participant FULL OUTER JOIN part_of ON participant.user_email = part_of.user_email FULL OUTER JOIN team ON team.team_id = part_of.team_id;


NAME                 TEAM_ID
-------------------- --------
Ankit Sharma         T0000002
Harsh                T0000003
Harsh                T0000004
Om                   T0000003
Om                   T0000004
Shyam                T0000003
Shyam                T0000005
Ravi                 no team
Yash Singhal         no team
Ayush Dubey          no team
Harsh                no team
Vaibhav              no team

12 rows selected.
```

(Move on to next page)

Data Modification

1. Updating description of every hack to “To be announced ” if the description is null

```
SQL> UPDATE hackathon SET description = NVL(description,'To Be Announced.') where hack_id in (Select hack_id from hackathon);

3 rows updated.
```

2. Updating team names to “no name” of teams having same name

```
SQL> Update team set name='no name 2' where team_id in (Select t1.team_id from team t1 INNER JOIN team t2 ON t1.team_id!=t2.team_id where COALESCE(NULLIF(t1.name,t2.name),'yes')='yes');

5 rows updated.
```

3. Updating participant profile

```
SQL> Update participant set college_name='harvard',domain='AI',phone_num='919645238172' where user_email in (Select user_email from participant where user_email='user3.4@email.com');

1 row updated.
```

4. Updating hackathon details

```
SQL> Update hackathon set name='AI fair 2.0',min_team_size=4 where hack_id in (Select hack_id from hackathon where hack_id='3333CCCC');

1 row updated.
```

Data Deletion

1. Delete hackathons that have number of registered teams equal to 1

```
SQL> delete from hackathon where hack_id in (select
  2      hackathon.hack_id
  3  FROM
  4      hackathon
  5  LEFT JOIN
  6      takes_part_in ON hackathon.hack_id = takes_part_in.hack_id
  7  GROUP BY
  8      hackathon.hack_id
  9  HAVING
 10      count(takes_part_in.team_id) = 1);

1 row deleted.
```

2. Delete participant profile

```
SQL> delete from participant where user_email in (Select user_email from participant where user_email='user3.3@email.com');

1 row deleted.
```

3. Delete team

```
SQL> delete from team where team_id in (Select team_id from team where team_id='T0000001');

1 row deleted.
```

4. Delete organizer profile

```
SQL> delete from organizer where organizer_email in (Select organizer_email from organizer where organizer_email='hvon@outlook.com');

1 row deleted.
```

PL/SQL (Screenshots)

PL/SQL Functions

1. Function to return the name of the hackathon with maximum duration

```
SQL> CREATE OR REPLACE FUNCTION hack_with_max_dur
  2  RETURN VARCHAR
  3  IS
  4  cursor c_hack is SELECT name,start_date,end_date from hackathon;
  5
  6  hack_rec c_hack%ROWTYPE;
  7
  8  max_d NUMBER := 0;
  9  hack_name VARCHAR(20):= 'none';
 10
 11  BEGIN
 12
 13  open c_hack;
 14
 15  fetch c_hack into hack_rec;
 16
 17
 18  while(c_hack%found) loop
 19
 20  if (hack_rec.end_date-hack_rec.start_date)>max_d THEN
 21
 22  max_d := hack_rec.end_date-hack_rec.start_date;
 23  hack_name := hack_rec.name;
 24  END IF;
 25
 26  fetch c_hack into hack_rec;
 27
 28  end loop;
 29
 30  close c_hack;
 31
 32  return hack_name;
 33  END;
 34  /

Function created.
```

```
SQL> BEGIN
  2      dbms_output.put_line(hack_with_max_dur());
  3  END;
  4  /
Equinox 2021

PL/SQL procedure successfully completed.
```

2. Function to return the number of users who are not part of any team

```
SQL> CREATE OR REPLACE FUNCTION teamless_participants
  2  RETURN number
  3  IS
  4  -- Declaring cursor
  5  CURSOR teamless_list IS SELECT
  6  participant.user_email
  7  FROM participant LEFT JOIN part_of
  8  ON participant.user_email = part_of.user_email
  9  GROUP BY
 10  participant.user_email
 11  having
 12  count(team_id) = 0;
 13  -- declaring variables
 14  email participant.user_email%TYPE;
 15  counter number;
 16  BEGIN
 17  counter := 0;
 18  OPEN teamless_list;
 19  LOOP
 20  FETCH teamless_list INTO email;
 21  EXIT WHEN teamless_list%NOTFOUND;
 22
 23  counter := counter + 1;
 24  END LOOP;
 25  CLOSE teamless_list;
 26  -- RETURN TOTAL PARTICIPANTS WITHOUT TEAM
 27  RETURN counter;
 28  END;
 29  /

Function created.
```

```
SQL> begin
  2  dbms_output.put_line(teamless_participants());
  3  end;
  4  /
5

PL/SQL procedure successfully completed.
```

PL/SQL Procedures

1. Retrieving names of winning teams of a particular hack

```
SQL> CREATE OR REPLACE PROCEDURE winners(hid VARCHAR)
  2  IS
  3  -- Declaring cursor
  4  CURSOR c_takes_part IS
  5  SELECT takes_part_in.hack_id, prize_to.prize_id, prize_to.team_id
  6  FROM takes_part_in INNER JOIN prize_to ON takes_part_in.team_id=prize_to.team_id WHERE hack_id=hid;
  7  v_takes_part c_takes_part%ROWTYPE;
  8  v_team_id team.team_id%TYPE;
  9  v_name team.name%TYPE;
 10  BEGIN
 11  OPEN c_takes_part;
 12  WHILE(c_takes_part%FOUND) LOOP
 13  FETCH c_takes_part INTO v_takes_part;
 14  v_team_id := v_takes_part.team_id;
 15  SELECT team.name INTO v_name FROM team WHERE team.team_id=v_team_id;
 16  dbms_output.put_line('Winner of ' || hid || ': '|| v_name);
 17  END LOOP;
 18  CLOSE c_takes_part;
 19  END;
 20  /

Procedure created.
```

```
SQL> begin
  2  winners('2222BBBB');
  3  end;
  4  /
Winner of 2222BBBB: Noob squad
```

2. Disqualifying teams from a hackathon that don't have the specified team size

```
SQL> CREATE OR REPLACE PROCEDURE delete_team(
  2      hackathon_id IN VARCHAR )
  3  IS
  4  cursor c_takes_part is select * from takes_part_in where hack_id=hackathon_id;
  5
  6  takes_part_rec c_takes_part%ROWTYPE;
  7
  8  team_rec team%ROWTYPE;
  9
 10  team_id team.team_id%type;
 11
 12  min_participants NUMBER := 0;
 13  max_participants NUMBER := 0;
 14
 15  BEGIN
 16
 17  open c_takes_part;
 18
 19  fetch c_takes_part into takes_part_rec;
 20
 21  Select min_team_size,max_team_size INTO min_participants, max_participants from hackathon where hack_id=hackathon_id;
 22
 23
 24  while(c_takes_part%found) loop
 25
 26  team_id := takes_part_rec.team_id;
 27
 28  delete from team where team.team_id=team_id and (team.num_of_members<min_participants or team.num_of_members>max_participants);
 29  dbms_output.put_line('team deleted');
 30
 31  fetch c_takes_part into takes_part_rec;
 32
 33  end loop;
 34
 35  close c_takes_part;
 36
 37  END;
 38  /

Procedure created.
```

```
SQL> BEGIN
  2     delete_team('2222BBBB');
  3  END;
  4  /
team deleted
team deleted

PL/SQL procedure successfully completed.
```