Course Project Work - Big Teeth Reality TV

By Group 5

Jenish Dhaduk(989354198)
Ashlea PIN-JUI Huang (989357938)
Htein Lynn Thar (989357674)
Diem Nguyen (989353845)

December 12, 2019

Distribution Of the project work among group members:

Each group member was responsible for their form. There were four forms and each group member was assigned a Form. First, we started by creating a table that has all the necessary entities for each form. Then we went to performing normalization step by step.

Diem Nguyen (989353845)

Created a list of normalized tables after performing normalization to the main table for APPLICATION FORM. The list contained 8 tables. She turned out to be an excellent co-partner to work on ERD as she provides some detailed insights into creating ERD. Also, provide her expertise on database operations in achieving the team's goal.

Jenish Dhaduk (989354198)

Created a list of normalized tables after performing normalization to the main table for VOTING FORM. The list contained 4 tables.
He turned out to be an excellent co-partner to took charge of the database. When we had all the tables ready. He helped to fill the database with all tables and also populated the data into Database with the help of Htein Lynn Thar.

Ashlea PIN-JUI Huang (989357938)

Created a list of normalized tables after performing normalization to the main table for EVENTS FORM. The list 9 contained tables.
Ashlea did a great job working on ERD and creating list of normalized tables. While creating queries as well she turned out to be an excellent co-partner.

Htein Lynn Thar (989357674)
Created a list of normalized tables after performing normalization to the main table for BACKGROUND CHECK FORM. The list 4 contained tables.
Htein Lynn Thar worked with Jenish Dhaduk on database operations. He actually did really well with inserting relevant data to the database.

Apart from the assigned work all team members also collaborated together whenever it was necessary. For establishing relationships among tables. We all got together as it was required to understand how to connect each table that were created separately.

Preparing for the presentation, and many such instances that required a collaborative approach, All team members gave equal justice to their role in achieving the completion of the project.

Exercises

1.create a normalized list of tables for each of the above forms.

LIST OF TABLES FOR EACH FORM AFTER NORMALIZATION
The following forms are the main forms will be using in the project including application form, background check form, events form, and voting form. After reviewing carefully the forms, we indicate needed tables as well as identify the primary keys and foreign keys for each table.

APPLICATION FORM

1. PRODUCER
2. DIRECTOR
3. CONTESTANT
4. MEDICATIONS
5. JOBS
6. ADDRESS
7. RATINGS
8. CONTESTANT_GENDER

BACKGROUND CHECK FORM

1. EVALUATION
2. EMPLOYER
3. EDUCATION
4. JUDICIAL

EVENTS FORM

1. EPISODE
2. EVENTS
3. DANGER_LEVEL
4. EVENT_ESTIMATED_TIME
5. DIRECTION_SETUP
6. SCENE_TIME
7. TEAMS
8. TASKS
9. POINTS

VOTING FORM

1. LOCATION
2. VOTES
3. METHODS
4. METHODS_ID

SQL QUERIES FOR CREATING TABLES

We then query to create tables so that we can establish our database and get ready for the next steps. The total of tables is 25 including some main tables such as Producers, Directors, Contestants, Ratings. Points and so on. We carefully indicate primary keys and foreign keys to ensure the relationships between tables are accurate which supportively ensure the good data flow between tables so that our next step to dig into the data and normalize them will include no obstacles. We cautiously select the entities for each table so that nothing is missing in our dataset or not so many extra things added to the dataset which might adversely be too much

time consuming for the group and might cause confusion for the end users as the project is implemented.

8 TABLES FOR FORM APPLICANT

```
CREATE TABLE PRODUCERS (
producer_id  int,
name varchar (50),
PRIMARY KEY (producer_id)
);


CREATE TABLE DIRECTORS(
director_id  int,
name varchar (50),
PRIMARY KEY (director_id)
);


CREATE TABLE CONTESTANT(
contestant_id int,
applicant_id int,
name varchar(50),
Email varchar(100),
PRIMARY KEY (contestant_id)
);


CREATE TABLE CONTESTANT_GENDER(
contestant_id int,
gender varchar(50),
PRIMARY KEY (contestant_id)
);


CREATE TABLE MEDICATIONS(
contestant_id int,
medicine varchar(50),
reason varchar(100),
PRIMARY KEY (contestant_id)
);
```

```
CREATE TABLE JOBS(
contestant_id int,
job_title varchar(50),
start_date datetime,
end_date datetime,
description varchar(500),
PRIMARY KEY (contestant_id)
);


CREATE TABLE ADDRESS(
contestant_id int,
apartment_number int,
block varchar(50),
street varchar(100),
city varchar(50),
state varchar(50),
postalcode int,
country varchar(50),
PRIMARY KEY (contestant_id)
);


CREATE TABLE RATINGS(
applicant_id int,
producer_rating int,
director_rating int,
PRIMARY KEY (applicant_id)
);
```

4 TABLES FOR FORM BACKGROUND CHECK

```
CREATE TABLE EVALUATION(
contestant_id int,
national_id varchar(50),
apperance_rating int,
strength_rating int,
religion varchar(50),
PRIMARY KEY (contestant_id)
);
```

```
CREATE TABLE EMPLOYER(
contestant_id int,
employer_name varchar(50),
employer_phone int,
employer_id int,
comments varchar(100),
PRIMARY KEY (employer_id)
);


CREATE TABLE EDUCATION(
contestant_id int,
highest_qualification varchar(50),
comments varchar(100),
institution varchar(50),
institution_contact_number int,
PRIMARY KEY (contestant_id)
);


CREATE TABLE JUDICIAL(
contestant_id int,
judicial_id int,
record_date datetime,
category varchar(100),
outcome varchar(50),
description varchar(500),
PRIMARY KEY (judicial_id)
);
```

9 TABLES FOR FORM EVENTS

```
CREATE TABLE EPISODES(
episode_id int,
date_aired datetime,
name varchar(100),
PRIMARY KEY (episode_id)
);
```

```
CREATE TABLE EVENTS(
event_id int,
episode_id int,
director_id int,
producer_id int,
team_event bit,
name varchar(50),
PRIMARY KEY (event_id)
);


CREATE TABLE TEAMS(
event_id int,
contestant_id int,
team_id int,
points int
PRIMARY KEY (event_id,contestant_id)
);


CREATE TABLE TASKS(
event_id int,
contestant_id int,
task_id int,
completion bit,
time int
PRIMARY KEY (event_id,contestant_id)
);


CREATE TABLE POINTS(
event_id int,
contestant_id int,
points int
PRIMARY KEY (event_id,contestant_id)
);


CREATE TABLE DANGER_LEVEL(
event_id int,
danger_level int,
PRIMARY KEY (event_id)
);
```

```
CREATE TABLE ESTIMATED_TIME(
event_id int,
time_length int,
PRIMARY KEY (event_id)
);


CREATE TABLE DIRECTION_SETUP(
action_scene_id int,
event_id int,
description varchar(100),
camera_setting varchar(100),
PRIMARY KEY (action_scene_id)
);


CREATE TABLE SCENE_TIME(
action_scene_id int,
time int,
PRIMARY KEY (action_scene_id)
);
```

4 TABLES FOR FORM VOTING

```
CREATE TABLE LOCATION(
voting_id int,
region varchar(50),
country varchar(50),
state varchar(100),
city varchar(100),
postal_code int,
PRIMARY KEY (voting_id)
);


CREATE TABLE VOTES(
voting_id int,
event_id int,
episode_id int,
contestant_id int,
```

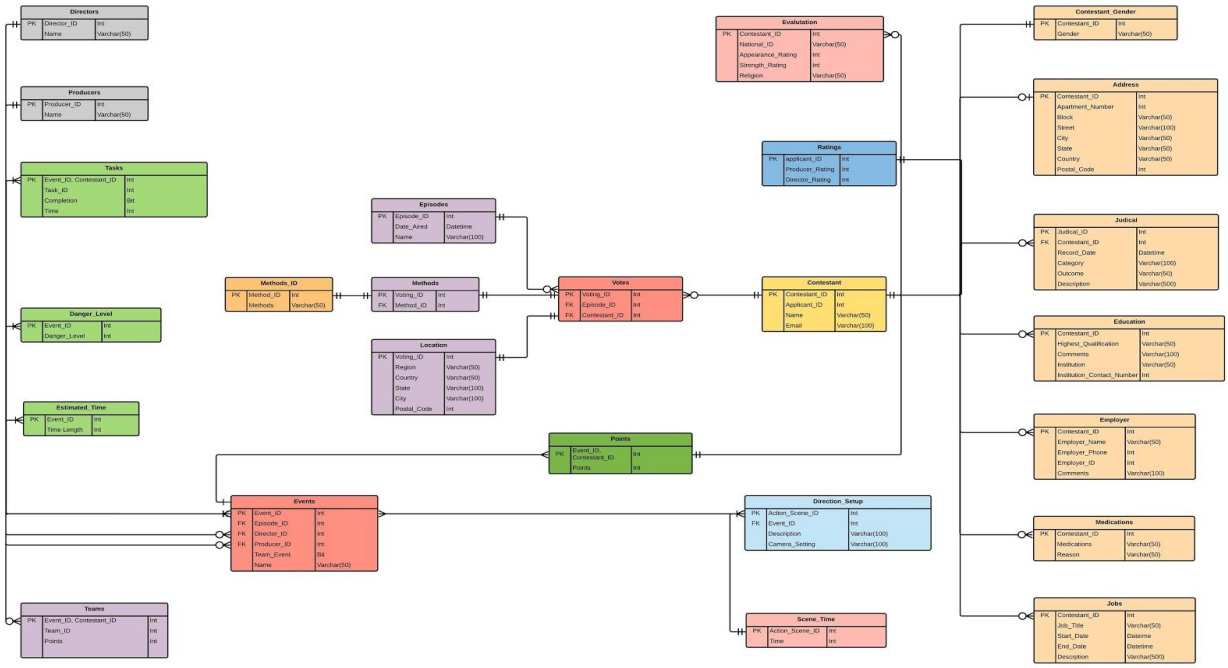
```
PRIMARY KEY (voting_id)
);


CREATE TABLE METHODS(
voting_id int,
method_id int,
PRIMARY KEY (voting_id)
);

CREATE TABLE METHODS_ID(
method_id int,
methods varchar(50),
PRIMARY KEY (method_id)
);
```

2. Create an ERD which shows these tables and fields. You may use PowerPoint or Visio or some other data modeling tool.

The coherent relationship of 25 tables is shown in the entity-relationship diagram (ERD) below. Each of the relationships is built and connected consistently with one or other tables in the dataset. On the right-hand side of the ERD is showing all the information needed from a contestant who will be sending the application to participate in some events, challenges or tasks hosted by the producer and/ or the director so that they can then be one of the promising candidates to join the Big Teeth TV show. Note that the color of the tables is set with the purpose of grouping the related tables together so that the ERD will be easier to follow; especially for the end-users to understand the flow of the project when they are first introduced to it.



3. Enter sample data to populate the tables to test your design. You do not need to create data entry forms

As tables and entity relationships are well prepared in the previous steps, we now will perform the next interesting step which is populating the data into existing tables. Please note the data is populated randomly.

<u>Populating the data</u>

```
INSERT INTO DANGER_LEVEL
(event_id,danger_level)
VALUES
(1,7),
(2,9),
(3,8),
(4,10),
(5,9),
(6,6);


INSERT INTO SCENE_TIME
(action_scene_id,time)
VALUES
(1,15),
(2,25),
```

```
(3,10),
(4,12),
(5,9),
(6,20),
(7,15),
(8,13),
(9,14);


INSERT INTO VOTES
 (voting_id,episode_id,event_id,contestant_id)
VALUES
(1,1,1,2),
(2,1,2,1),
(3,1,2,3),
(4,1,1,4),
(5,1,2,3),
(6,1,2,3),
(7,1,2,3),
(8,1,2,1),
(9,1,2,3),
(10,2,3,4),
(11,2,3,4),
(12,2,4,1),
(13,2,3,4),
(14,2,4,1),
(15,2,4,1),
(16,2,4,1);


INSERT INTO CONTESTANT
(contestant_id,applicant_id,name,Email)
VALUES
(1,2,'ASHLEY','ash456@gmail.com'),
(2,3,'Lynn','lynn666@gmail.com'),
(3,4,'Jenish','jen745@gmail.com'),
(4,5,'Diem','diem444@gmail.com'),
(5,7,'Moxshil','mox4555@gmail.com'),
(6,8,'Jet','jet888@gmail.com'),
(7,11,'Jack','jack7525@gmail.com'),
(8,12,'Mark','Mark8856@gmail.com'),
(9,13,'Steve','Steve7856@gmail.com');
```

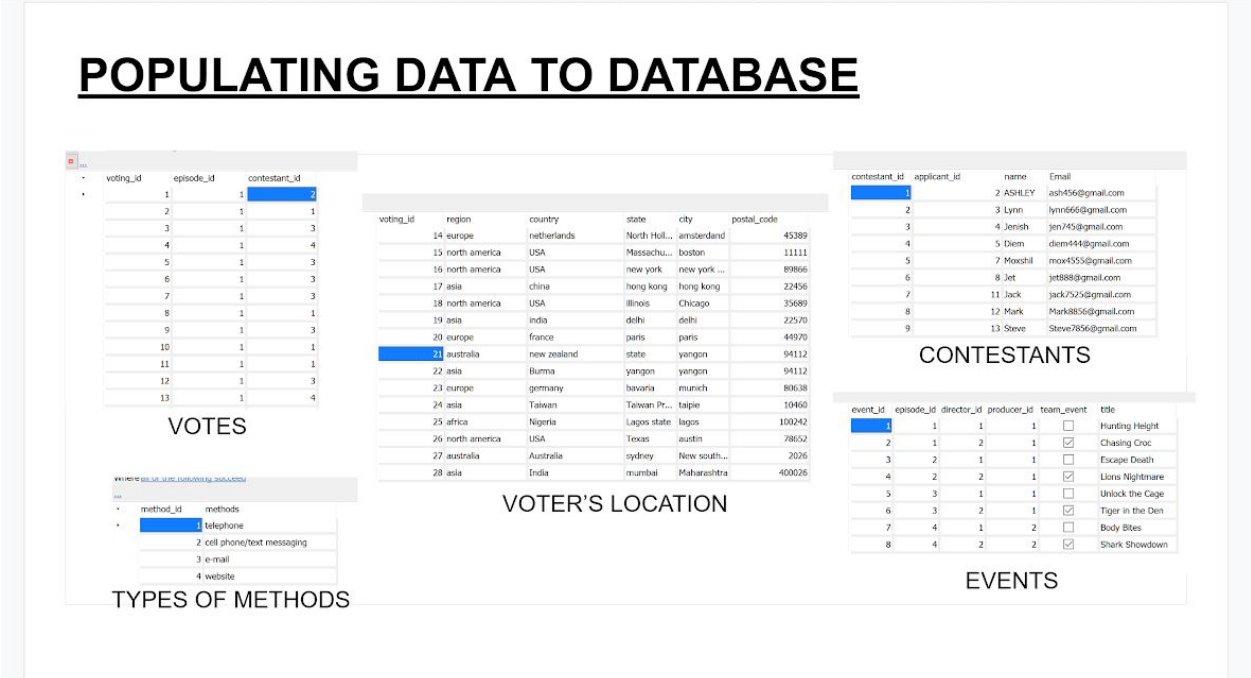
```
INSERT INTO LOCATION
(voting_id,region,country,state,city,postal_code)
VALUES
(11,'australia','new zealand','state','yangon',94112)
(12,'asia','Burma','yangon','yangon',94112),
(13,'europe','germany','bavaria','munich',80638),
(14,'asia','Taiwan','Taiwan Province','taipie',10460),
(15,'africa','Nigeria','Lagos state','lagos',100242),
(16,'north america', 'USA','Texas','austin',78652),
(17,'australia','Australia','sydney','New south wales',2026),
(18,'asia','India','mumbai','Maharashtra',400026),
(19,'asia','China','Jinan','Shandong',250014),
(20,'north america','Canada','torronto','Ontario',91710),
(11,'europe','england','','london',25798),
(12,'Asia','singapore','','singapore',10113),
(13,'europe','spain','','barcelona',87009),
(14,'europe','netherlands','','amsterdand',45389),
(15,'north america','USA','','boston',11111),
(16,'north america','USA','','new york',89866),
(17,'asia','hong kong','','hong kong',22456),
(18,'north america','USA','','Chicago',35689),
(19,'asia','india','','delhi',22570),
(20,'europe','france','','paris',44970);



INSERT INTO TEAM_MEMBER_POINTS (event_id,contestant_id,team_id,points)
VALUES
(2,1,1,8),
(2,2,1,7),
(2,3,1,9),
(2,4,2,-2),
(2,5,2,0),
(2,6,2,3),
(4,4,3,3),
(4,3,3,4),
(4,2,3,6),
(4,1,4,7),
```

(4,5,4,8),
(4,6,4,9);

INSERT INTO TASKS (event_id,contestant_id,task_id,completion,time)
VALUES
(1,1,1,0,45),
(1,2,2,1,30),
(1,3,3,1,25),
(1,4,4,1,35),
(1,5,5,0,50),
(1,6,6,0,55),
(2,1,7,1,130),
(2,2,7,1,130),
(2,3,7,1,130),
(2,4,8,0,150),
(2,5,8,0,150),
(2,6,8,0,150);



Big Teeth TV Show Database

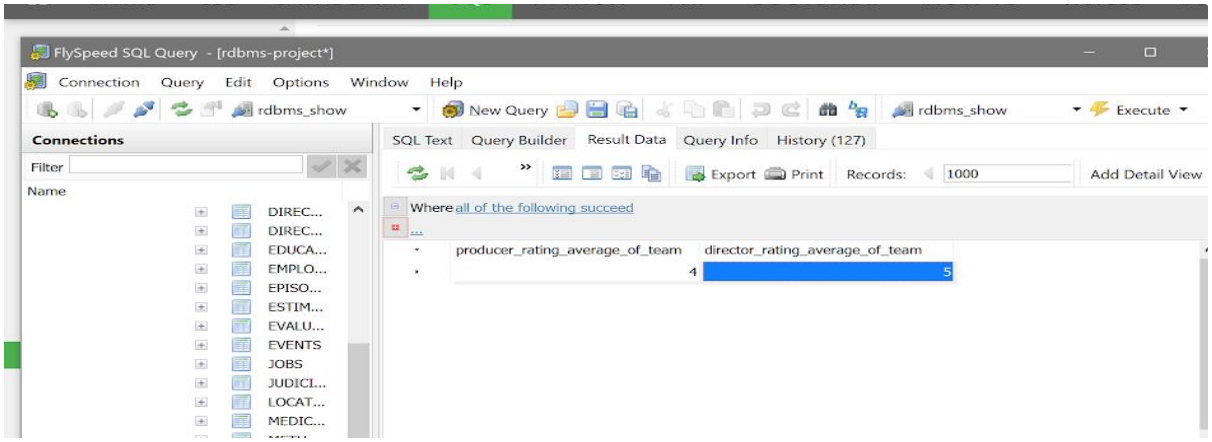
Create queries to answer the following questions:

1. Calculate the average producer and director ratings for a specific event's team members (you pick the event title).

QUERY

```
SELECT  AVG(r.producer_rating)as producer_rating_average_of_team,
      AVG(r.director_rating) as director_rating_average_of_team
FROM RATINGS r
JOIN CONTESTANT c
  ON c.applicant_id = r.applicant_id
WHERE
c.contestant_id IN(SELECT contestant_id
                   FROM TEAMS
                   WHERE team_id=1
                   AND event_id = (SELECT event_id
                                   FROM EVENTS e
                                   WHERE title = 'ChasingCroc'))
```

Answer: the average producer ratings is 4 while it is 5 for the average director ratings.

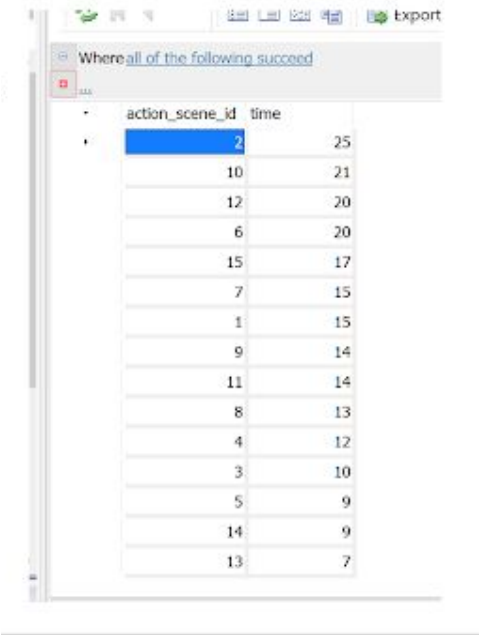


2. Determine which actions will take the longest. Rank all actions from longest to shortest.

QUERY

```
SELECT * FROM SCENE_TIME
ORDER BY
time DESC
```

Answer: action scene with the ID of 2 takes the longest time which is 25.



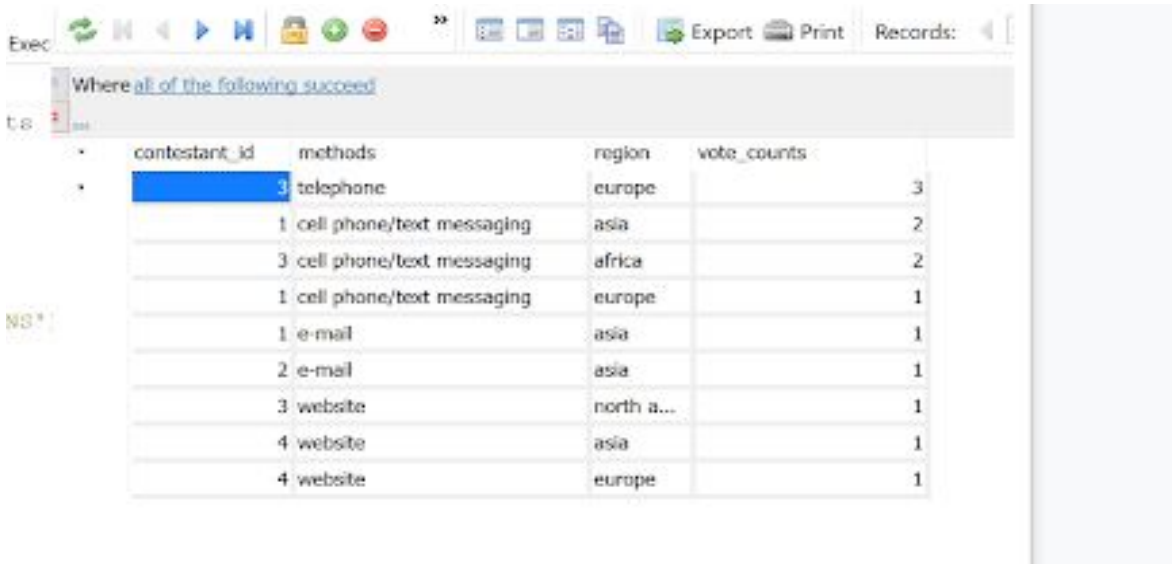
action_scene_id	time
2	25
10	21
12	20
6	20
15	17
7	15
1	15
9	14
11	14
8	13
4	12
3	10
5	9
14	9
13	7

3. List each contestant's total votes by region and method for a specific episode (you pick the episode title). Rank this from highest to lowest.

QUERY

```
SELECT  v.contestant_id,mid.methods,l.region,COUNT(v.voting_id) as vote_counts
FROM VOTES v
JOIN LOCATION l
  ON l.voting_id = v.voting_id
JOIN METHODS m
  ON m.voting_id = l.voting_id
JOIN METHODS_ID mid
  ON mid.method_id = m.method_id
WHERE v.episode_id = (SELECT episode_id FROM EPISODES WHERE name = 'FUN
BEGINS')
GROUP BY
l.region,v.contestant_id,mid.methods
ORDER BY
COUNT(v.voting_id) DESC
```

Answer: Below is the list of 9 contestants with their votes by region and method for the episode names *Fun Begins.*



contestant_id	methods	region	vote_counts
3	telephone	europe	3
1	cell phone/text messaging	asia	2
3	cell phone/text messaging	africa	2
1	cell phone/text messaging	europe	1
1	e-mail	asia	1
2	e-mail	asia	1
3	website	north a...	1
4	website	asia	1
4	website	europe	1

4. Identify which contestants have not participated in any events.

QUERY

```
select contestant_id, name from CONTESTANT
WHERE contestant_id NOT IN (SELECT DISTINCT contestant_id
                            FROM TASKS)
```

Answer: Jack, Mark, and Steve are so far the 3 contestants that have not participated in any events.

contestant_id	name
7	Jack
8	Mark
9	Steve

5. What is the highest estimated danger level for any event?

QUERY

```
SELECT TOP 1 event_id,danger_level FROM DANGER_LEVEL
```

ORDER BY
danger_level DESC

Answer: the event with the Id of 4 has the highest danger level- 10.

6. Show a relational expression for any one query

RELATIONAL EXPRESSION FOR QUERY4

$$\Pi_{\text{contestant\_id,name}}(\sigma_{\text{contestant\_id(CONTESTANT)}} - \sigma_{\text{contestant\_id(TASKS)}})$$