



جامعة جدة  
University of Jeddah

COMING SOON !

# General Entertainment Authority

INTRODUCTION TO DATABASE  
CCCS 215



INSTRUCTOR:  
DR. MASHAEL M.KHAYYAT

SECTION CO

MAY 2024



# Table Of Content

**1.1 Project Team**

**1.2 GEA Database Description**

**1.3 E-R DIAGRAM**

**1.4 Relational Schema**

**1.5 Normalization**

**1.6 Functional Dependencies**

**1.7 Logical Model**

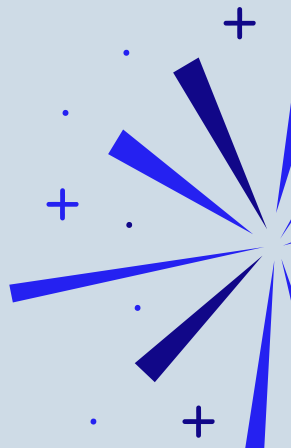
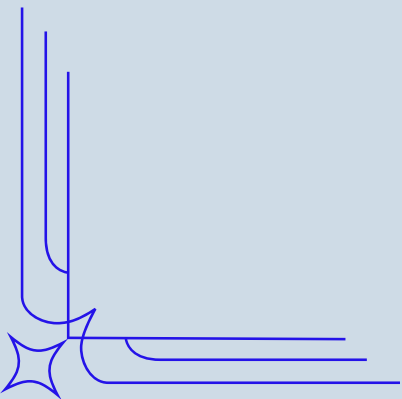
**1.8 Coding**

**1.9 Project Team Tasks**



## 1.1 Team Members

- Hadeel Alharthi 2210794
- Shifa Albadri 2115406
- Aya Alhazmi 2115144
- Kholod Althbeny 2110471

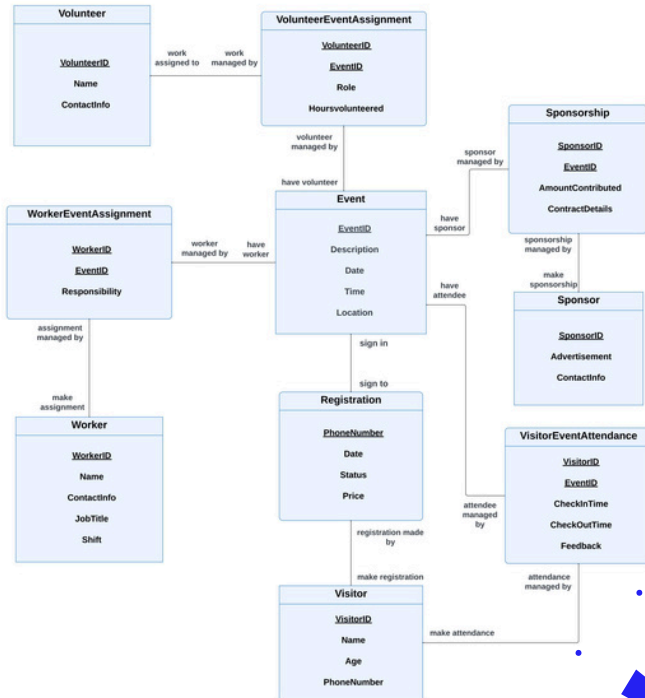


## 1.2 GEA Database Description

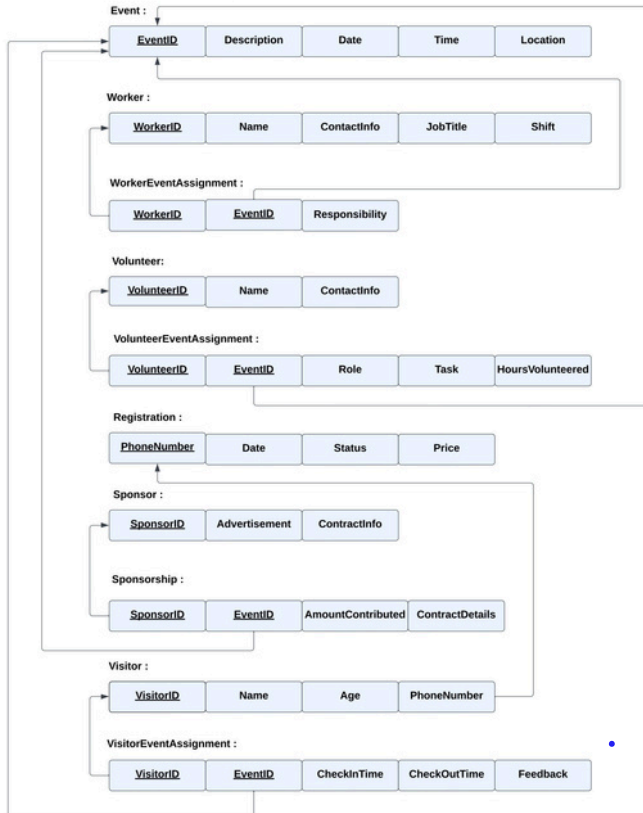
The GEA is struggling with disorganized entertainment data dispersed across platforms, risking loss of insights and compromising security. Prioritizing an extensive database system is essential for organizing, protecting, and accessing valuable data assets to improve the entertainment scene.

we have 10 entities: visitor, volunteer, event, worker, registration, sponsor, visitorEventAttendance, sponsorship, volunteerEventAssignment, and WorkerEventAssignment. Volunteers engage in multiple events, and events entail the involvement of numerous volunteers. To streamline this relationship, we introduced the volunteerEventAssignment, establishing a one-to-one connection. Similarly, workers are involved in multiple events, and events require the contribution of various workers. This complex interaction is simplified through the addition of the workerEventAssignment, resolving it into a one-to-one relationship. Visitors engage in a one-to-one registration and event attendance process, ensuring seamless participation. Moreover, events maintain a reciprocal one-to-one relationship with visitor registrations. Sponsors, engaging in multiple event sponsorships, and events attracting multiple sponsors, have their dynamic bond solidified through the sponsorship entity, establishing a coherent one-to-one association. These meticulous adjustments enhance data integrity and accessibility, essential for the GEA's mission to elevate the entertainment landscape.

# 1.3 E-R DIAGRAM



# 1.4 Relational Schema



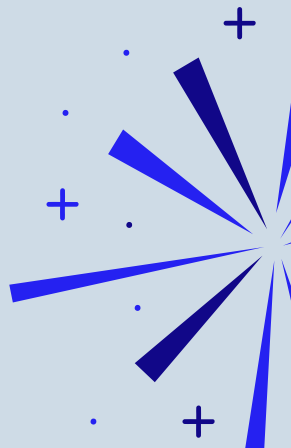
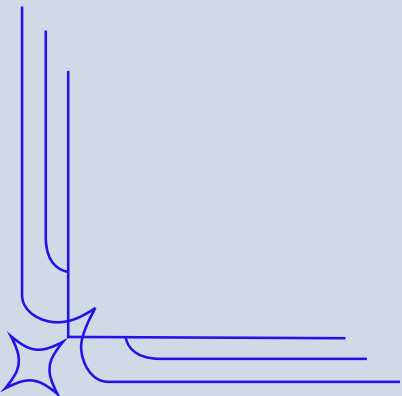


## 1.5 Normalization

**1NF:** - The attribute Shift has been removed from the table WorkerEventAssignment because of duplication.

**2NF:** - has no partial dependency. That is, all non-key attributes are fully dependent The schema is already in 2NF

**3NF:** - have no transitive dependency. The schema is already in 3NF.



## 1.6 Functional Dependencies

FD EventID -> Description, Date, Time, Location

FD WorkerID -> Name, ContactInfo, JobTitle, Shift, responsibility

FD VolunteerID -> Name, ContactInfo, Role, Task, HoursVolunteered

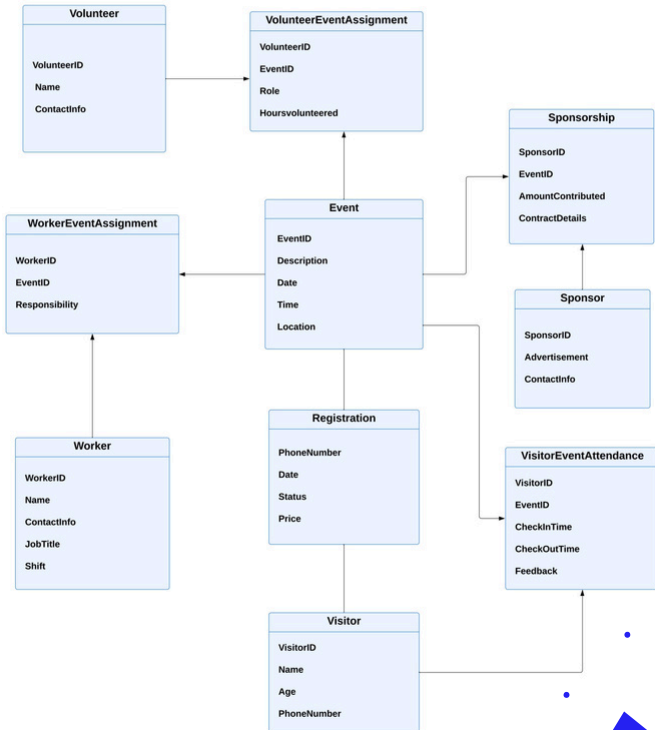
FD PhoneNumber -> Date, Status, Price

FD SponsorID -> Advertisement, ContactInfo, AmountContributed, ContractDetails

FD VisitorID -> Name, Age, CheckInTime, CheckOutTime, Feedback



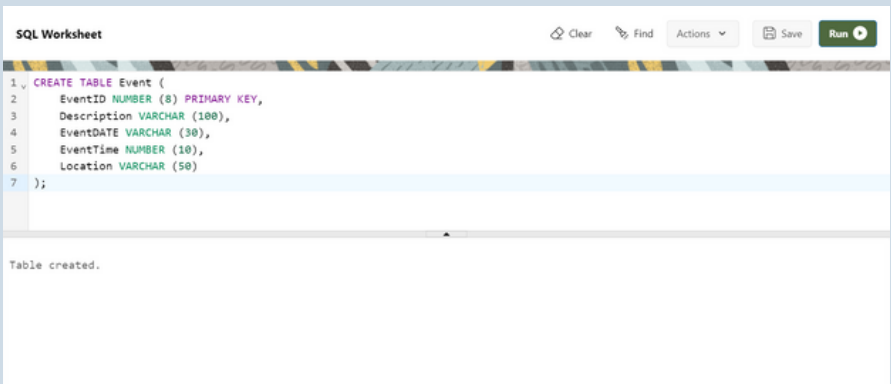
## 1.7 Logical Model



# 1.8 Coding

## 1- Event entity

- table creation



The screenshot shows an SQL Worksheet interface. At the top, there's a title bar with 'SQL Worksheet' on the left and buttons for 'Clear', 'Find', 'Actions', 'Save', and 'Run' on the right. The main area contains a SQL query to create a table named 'Event'. The query is as follows:

```
1 CREATE TABLE Event (  
2     EventID NUMBER (8) PRIMARY KEY,  
3     Description VARCHAR (100),  
4     EventDATE VARCHAR (30),  
5     EventTime NUMBER (10),  
6     Location VARCHAR (50)  
7 );
```

Below the query, a status message indicates 'Table created.'

# 1.8 Coding

## 1- Event entity

- insert into table

SQL Worksheet

```
1 INSERT INTO Event VALUES (11654667, 'Travis Concert', '12 January', 1, 'City walk');
2 INSERT INTO Event VALUES (11764637, 'Art Space', '5 October', 3, 'City Walk');
3 INSERT INTO Event VALUES (11938738, 'Gaming Event', '31 January', 12, 'Superdome');
4 INSERT INTO Event VALUES (11838838, 'Kids Play Time', '7 April', 7, 'Redsea Mall');
5 INSERT INTO Event VALUES (11726763, 'Korean Festival', '28 May', 5, 'Superdome');
6
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

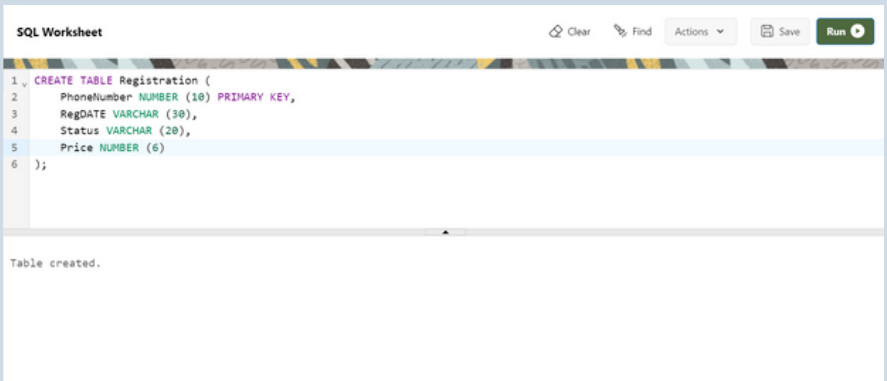
```
1 select * from Event ;
2
3
4
```

EVENTID	DESCRIPTION	EVENTDATE	EVENTTIME	LOCATION
11654667	Travis Concert	12 January	1	City walk
11764637	Art Space	5 October	3	City Walk
11938738	Gaming Event	31 January	12	Superdome
11838838	Kids Play Time	7 April	7	Redsea Mall
11726763	Korean Festival	28 May	5	Superdome

# 1.8 Coding

## 2- Registration entity

- create table



The screenshot shows an SQL Worksheet interface. At the top, there is a toolbar with buttons for 'Clear', 'Find', 'Actions', 'Save', and a 'Run' button with a play icon. Below the toolbar, the SQL code is displayed in a text area with line numbers 1 through 6. The code is: 

```
1. CREATE TABLE Registration (  
2.     PhoneNumber NUMBER (10) PRIMARY KEY,  
3.     RegDATE VARCHAR (30),  
4.     Status VARCHAR (20),  
5.     Price NUMBER (6)  
6. );
```

 Below the code area, a status message reads 'Table created.' The interface has a light blue header and a white background for the code and output areas.

```
1. CREATE TABLE Registration (  
2.     PhoneNumber NUMBER (10) PRIMARY KEY,  
3.     RegDATE VARCHAR (30),  
4.     Status VARCHAR (20),  
5.     Price NUMBER (6)  
6. );
```

Table created.

# 1.8 Coding

## 2- Registration entity

- insert into table

SQL Worksheet

Clear

Find

Actions ▾

Save

Run

```
1 INSERT INTO Registration VALUES (0503435674,'1 February','Active',100);
2 INSERT INTO Registration VALUES (0554521674,'17 February','Active',200);
3 INSERT INTO Registration VALUES (0508807533,'23 March','Inactive',150);
4 INSERT INTO Registration VALUES (0500805634,'5 April','Active',300);
5 INSERT INTO Registration VALUES (0529398763,'20 April','Inactive',100);
```

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

SQL Worksheet

Clear

Find

Actions ▾

Save

Run

```
1 select * from Registration ;
2
3
4
```

PHONENUMBER	REGDATE	STATUS	PRICE
503435674	20 April	Inactive	500
554521674	17 February	Active	200
508807533	23 March	Inactive	150
500805634	5 April	Active	300
529398763	20 April	Inactive	100

# 1.8 Coding

## 3- Visitor entity

- create table

SQL Worksheet

Clear Find Actions Save Run

```
1 CREATE TABLE Visitor (  
2   VisitorID NUMBER (10) PRIMARY KEY,  
3   Name VARCHAR (30),  
4   Age NUMBER (2),  
5   PhoneNumber NUMBER (10)  
6 );
```

Table created.

# 1.8 Coding

## 3- Visitor entity

- insert into table

SQL Worksheet

```
1 INSERT INTO Visitor VALUES (1120052376,'Hadeel Faisal',21,0500804576);
2 INSERT INTO Visitor VALUES (1124367895,'Shifa Ahmad',22,0544378639);
3 INSERT INTO Visitor VALUES (1287445686,'Layan Jamal',34,0566128345);
4 INSERT INTO Visitor VALUES (1042335678,'Rami Mohammed',19,0556764321);
5 INSERT INTO Visitor VALUES (1023125672,'Abdullah Waseem',27,059456587);
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

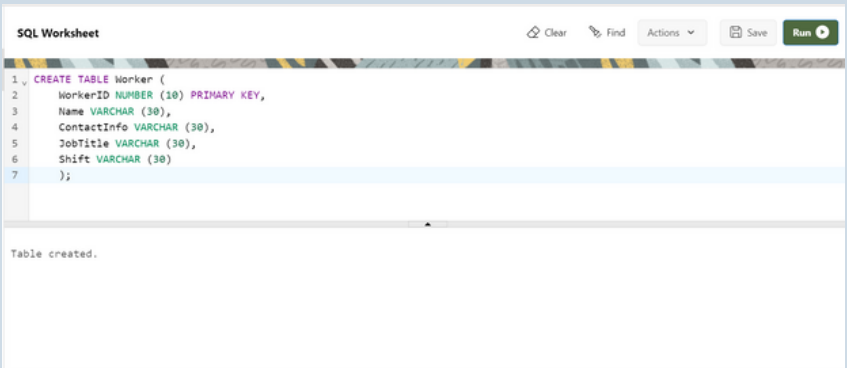
```
1 select * from Visitor ;
```

VISITORID	NAME	AGE	PHONENUMBER
1120052376	Hadeel Faisal	21	0500804576
1124367895	Shifa Ahmad	22	544378639
1287445686	Layan Jamal	34	566128345
1042335678	Rami Mohammed	19	556764321
1023125672	Abdullah Waseem	27	599456587

# 1.8 Coding

## 4- Worker entity

- create table



The screenshot shows an SQL Worksheet interface. At the top, there's a header bar with the title "SQL Worksheet" and several action buttons: "Clear", "Find", "Actions" (with a dropdown arrow), "Save", and "Run" (with a play icon). Below the header, the main area contains a SQL query: 

```
1 CREATE TABLE Worker (  
2   WorkerID NUMBER (10) PRIMARY KEY,  
3   Name VARCHAR (30),  
4   ContactInfo VARCHAR (30),  
5   JobTitle VARCHAR (30),  
6   Shift VARCHAR (30)  
7 );
```

 The query is numbered 1 through 7. Below the query editor, there's a status bar that says "Table created."



# 1.8 Coding

## 4- Worker entity

- insert into table

SQL Worksheet

```
1 INSERT INTO Worker VALUES (1053234562,'Lama Hassan','lanahas9@hotmail.com','Event Manager','Ahmad Badr');
2 INSERT INTO Worker VALUES (1193838209,'Elaf Sami','elaf123@hotmail.com','Medical Coordinator','Sanaa Emad');
3 INSERT INTO Worker VALUES (1172782355,'Mansour Rizk','rizkman@gmail.com','Media Coordinator','Mohammed Asaad');
4 INSERT INTO Worker VALUES (1127673511,'Khalil Rami','kh54ram@gmail.com','Stage Manager','Dona Jamal');
5 INSERT INTO Worker VALUES (1234128974,'Hams Faisal','hamsf77@gmail.com','Social Media Manager','Arwa Ahmad');
6
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

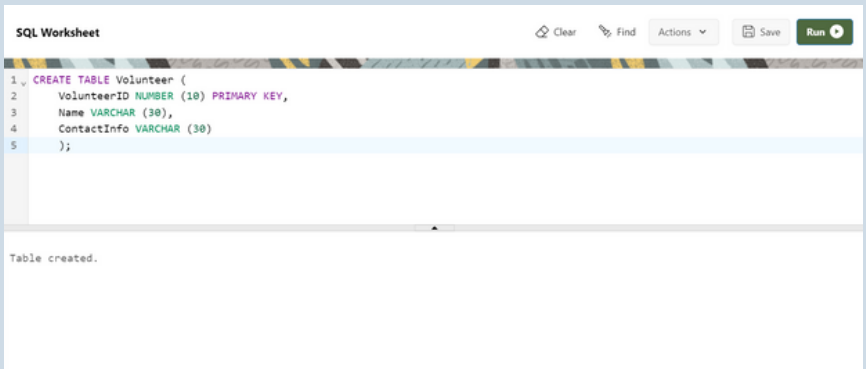
```
1 select * from Worker ;
2
3
4
```

WORKERID	NAME	CONTACTINFO	JOBTITLE	SHIFT
1053234562	Lama Hassan	lanahas9@hotmail.com	Event Manager	Ahmad Badr
1193838209	Elaf Sami	elaf123@hotmail.com	Medical Coordinator	Sanaa Emad
1172782355	Mansour Rizk	rizkman@gmail.com	Media Coordinator	Mohammed Asaad
1127673511	Khalil Rami	kh54ram@gmail.com	Stage Manager	Dona Jamal
1234128974	Hams Faisal	hamsf77@gmail.com	Social Media Manager	Arwa Ahmad

# 1.8 Coding

## 5- Volunteer entity

- create table



The screenshot shows an SQL Worksheet interface. At the top, there's a title bar with 'SQL Worksheet' and several action buttons: 'Clear', 'Find', 'Actions', 'Save', and 'Run'. The main area contains a SQL query that has been executed. The query is as follows:

```
1 CREATE TABLE Volunteer (  
2   VolunteerID NUMBER (10) PRIMARY KEY,  
3   Name VARCHAR (30),  
4   ContactInfo VARCHAR (30)  
5 );
```

Below the query, a status message indicates the result of the execution: 'Table created.'

# 1.8 Coding

## 5- Volunteer entity

- insert into table

SQL Worksheet

Clear Find Actions Save Run

```
1 INSERT INTO Volunteer VALUES (1144644980,'Manal Awad','Mawad@gmail.com');
2 INSERT INTO Volunteer VALUES (1056435670,'Abeer Saad','Asaad@gmail.com');
3 INSERT INTO Volunteer VALUES (1123256743,'Khalid Mohammed','Khalidmoh@gmail.com');
4 INSERT INTO Volunteer VALUES (1023656432,'Mashhoor Sami','Mashsani54@gmail.com');
5 INSERT INTO Volunteer VALUES (1221564389,'Hamad Waleed','hamad123@hotmail.com');
6
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

Clear Find Actions Save Run

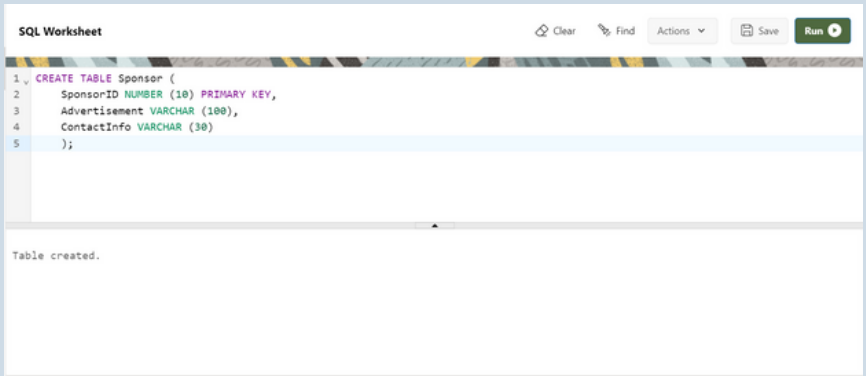
```
1 select * from Volunteer ;
2
3
4
```

VOLUNTEERID	NAME	CONTACTINFO
1144644980	Manal Awad	Mawad@gmail.com
1056435670	Abeer Saad	Asaad@gmail.com
1123256743	Khalid Mohammed	Khalidmoh@gmail.com
1023656432	Mashhoor Sami	Mashsani54@gmail.com
1221564389	Hamad Waleed	hamad123@hotmail.com

# 1.8 Coding

## 6- Sponsor entity

- create table



The screenshot shows an SQL Worksheet interface. At the top, there's a title bar with 'SQL Worksheet' and several action buttons: 'Clear', 'Find', 'Actions', 'Save', and 'Run'. The main area contains a SQL query to create a table named 'Sponsor'. The query is as follows:

```
1 CREATE TABLE Sponsor (  
2   SponsorID NUMBER (10) PRIMARY KEY,  
3   Advertisement VARCHAR (100),  
4   ContactInfo VARCHAR (30)  
5 );
```

Below the query, a status message indicates the successful execution of the command:

Table created.

# 1.8 Coding

## 6- Sponsor entity

- insert into table

SQL Worksheet

```
1 INSERT INTO Sponsor VALUES (1114345577,'Concert Ads','HDLbeast@gmail.com');
2 INSERT INTO Sponsor VALUES (1176888943,'Space Ads','Spaceadvertise@gmail.com');
3 INSERT INTO Sponsor VALUES (1123321567,'Festival Ads','Festivaladvertise@gmail.com');
4 INSERT INTO Sponsor VALUES (1017652345,'Event Ads','Eventmanagement@gmail.com');
5 INSERT INTO Sponsor VALUES (1007624536,'Social Media Ads','Socialmedia@gmail.com');
6
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

```
1 select * from Sponsor ;
2
3
4
```

SPONSORID	ADVERTISEMENT	CONTACTINFO
1114345577	Concert Ads	HDLbeast@gmail.com
1176888943	Space Ads	Spaceadvertise@gmail.com
1123321567	Festival Ads	Festivaladvertise@gmail.com
1017652345	Event Ads	Eventmanagement@gmail.com
1007624536	Social Media Ads	Socialmedia@gmail.com

# 1.8 Coding

## 7- VisitorEventAttendance entity

- create table

SQL Worksheet

Clear Find Actions Save Run

```
1 CREATE TABLE VisitorEventAttendance (  
2   VisitorID NUMBER (10),  
3   EventID NUMBER (8),  
4   CheckInTime NUMBER (30),  
5   CheckOutTime NUMBER (30),  
6   Feedback VARCHAR (100),  
7   PRIMARY KEY (VisitorID, EventID),  
8   FOREIGN KEY (VisitorID) REFERENCES Visitor (VisitorID),  
9   FOREIGN KEY (EventID) REFERENCES Event (EventID)  
10 );
```

Table created.

# 1.8 Coding

## 7- VisitorEventAttendance entity

- insert into table

SQL Worksheet

Clear Find Actions Save Run

```
1 INSERT INTO VisitorEventAttendance VALUES (1120052376,11654667,9,12,'Great!');
2 INSERT INTO VisitorEventAttendance VALUES (1124367895,11764637,1,12,'Great!');
3 INSERT INTO VisitorEventAttendance VALUES (1287445686,11938738,8,10,'Great!');
4 INSERT INTO VisitorEventAttendance VALUES (1042335678,11838838,3,6,'Great!');
5 INSERT INTO VisitorEventAttendance VALUES (1023125672,11726763,7,9,'Great!');
6
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

Clear Find Actions Save Run

```
1 select * from VisitorEventAttendance ;
2
3
4
```

VISITORID	EVENTID	CHECKINTIME	CHECKOUTTIME	FEEDBACK
1120052376	11654667	9	12	Great!
1124367895	11764637	1	12	Great!
1287445686	11938738	8	10	Great!
1042335678	11838838	3	6	Great!
1023125672	11726763	7	9	Great!

# 1.8 Coding

## 8- WowkerEventAssignment entity

- create table

SQL Worksheet

Clear Find Actions Save Run

```
1 CREATE TABLE WorkerEventAssignment (  
2   WorkerID NUMBER (10),  
3   EventID NUMBER (8),  
4   Responsibility VARCHAR (30),  
5   PRIMARY KEY (WorkerID, EventID),  
6   FOREIGN KEY (WorkerID) REFERENCES Worker (WorkerID),  
7   FOREIGN KEY (EventID) REFERENCES Event (EventID)  
8 );
```

Table created.



# 1.8 Coding

## 8- WowkerEventAssignment entity

- insert into table

SQL Worksheet

```
1 INSERT INTO WorkerEventAssignment VALUES (1053234562, 11654667, 'Manage budget');
2 INSERT INTO WorkerEventAssignment VALUES (1193838209, 11764637, 'Account tickets');
3 INSERT INTO WorkerEventAssignment VALUES (1172782355, 11938738, 'Manage tickets');
4 INSERT INTO WorkerEventAssignment VALUES (1127673511, 11838838, 'Manage employees');
5 INSERT INTO WorkerEventAssignment VALUES (1234128974, 11726763, 'Ambulance injured');
6
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

```
1 select * from WorkerEventAssignment ;
2
3
4
```

WORKERID	EVENTID	RESPONSIBILITY
1053234562	11654667	Manage budget
1193838209	11764637	Account tickets
1172782355	11938738	Manage tickets
1127673511	11838838	Manage employees
1234128974	11726763	Ambulance injured

# 1.8 Coding

## 9- VolunteerEventAssignment

- create table

SQL Worksheet

Clear Find Actions Save Run

```
1 CREATE TABLE VolunteerEventAssignment (  
2     VolunteerID NUMBER (10),  
3     EventID NUMBER (8),  
4     VolunteerRole VARCHAR (50),  
5     HoursVolunteered NUMBER (38),  
6     PRIMARY KEY (VolunteerID, EventID),  
7     FOREIGN KEY (VolunteerID) REFERENCES Volunteer (VolunteerID),  
8     FOREIGN KEY (EventID) REFERENCES Event (EventID)  
9 );
```

Table created.

# 1.8 Coding

## 9- VolunteerEventAssignment

- insert into table

SQL Worksheet

```
1 INSERT INTO VolunteerEventAssignment VALUES (1144644900,11654667,'Management',240);
2 INSERT INTO VolunteerEventAssignment VALUES (1056435670,11764637,'Photography',180);
3 INSERT INTO VolunteerEventAssignment VALUES (1123256743,11938738,'Parking management',240);
4 INSERT INTO VolunteerEventAssignment VALUES (1023656432,11838838,'Tickets registration',250);
5 INSERT INTO VolunteerEventAssignment VALUES (1221564389,11726763,'Event management',280);
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

```
1 select * from VolunteerEventAssignment ;
2
3
4
```

VOLUNTEERID	EVENTID	VOLUNTEERROLE	HOURSVOLUNTEERED
1144644900	11654667	Management	240
1056435670	11764637	Photography	180
1123256743	11938738	Parking management	240
1023656432	11838838	Tickets registration	250
1221564389	11726763	Event management	280

# 1.8 Coding

## 10- Sponsorship entity

- create table

SQL Worksheet

Clear Find Actions Save Run

```
1 CREATE TABLE Sponsorship (  
2   SponsorID NUMBER (10),  
3   EventID NUMBER (8),  
4   AmountContributed VARCHAR (100),  
5   ContractDetails VARCHAR (100),  
6   PRIMARY KEY (SponsorID, EventID),  
7   FOREIGN KEY (SponsorID) REFERENCES Sponsor (SponsorID),  
8   FOREIGN KEY (EventID) REFERENCES Event (EventID)  
9 );
```

Table created.

# 1.8 Coding

## 10- Sponsorship entity

- insert into table

SQL Worksheet

Clear Find Actions Save Run

```
1 INSERT INTO Sponsorship VALUES (1114345577, 11654667, '5000$', 'Agreement for promotional');
2 INSERT INTO Sponsorship VALUES (1176888943, 11764637, '3000$', 'Agreement for event sponsorship');
3 INSERT INTO Sponsorship VALUES (1123321567, 11938738, '2500$', 'Agreement for play time');
4 INSERT INTO Sponsorship VALUES (1017652345, 11838838, '9000$', 'Agreement for festival');
5 INSERT INTO Sponsorship VALUES (1007624536, 11726763, '10000$', 'Agreement for banners and logos');
```

1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.  
1 row(s) inserted.

SQL Worksheet

Clear Find Actions Save Run

```
1 select * from Sponsorship ;
2
3
4
```

SPONSORID	EVENTID	AMOUNTCONTRIBUTED	CONTRACTDETAILS
1114345577	11654667	5000\$	Agreement for promotional
1176888943	11764637	3000\$	Agreement for event sponsorship
1123321567	11938738	2500\$	Agreement for play time
1017652345	11838838	9000\$	Agreement for festival
1007624536	11726763	10000\$	Agreement for banners and logos

# 1.8 Coding

## The Queries:

1-

SQL Worksheet

Clear Find Actions Save Run

```
1 SELECT Age, COUNT (Age) FROM Visitor GROUP BY Age;
2
3
```

AGE	COUNT(AGE)
27	1
34	1
21	1
22	1
19	1

2-

SQL Worksheet

Clear Find Actions Save Run

```
1 SELECT MAX (Age) FROM Visitor ;
2
3
```

MAX(AGE)
34

Download CSV

# 1.8 Coding

## The Queries :

3-

SQL Worksheet

Clear Find Actions Save

```
1 SELECT VisitorEventAttendance.VisitorID, Event.EventID, VisitorEventAttendance.Feedback FROM VisitorEventAttendance
2 INNER JOIN Event ON Event.EventID = Event.EventID
3 ORDER BY VisitorEventAttendance.Feedback ;
```

VISITORID	EVENTID	FEEDBACK
1120052376	11654667	Great!
1023125672	11938738	Great!
1120052376	11764637	Great!
1120052376	11838838	Great!
1120052376	11938738	Great!

4-

SQL Worksheet

Clear Find Actions Save Run

```
1 SELECT SponsorID FROM Sponsorship WHERE EventID IN (SELECT EventID FROM Event WHERE EventTime = 12);
2
3
```

SPONSORID

1123321567

Download CSV

# 1.8 Coding

The Queries :

5-

SQL Worksheet

Clear Find Actions Save Run

```
1 SELECT Description FROM Event ORDER BY EventID DESC;  
2  
3
```

DESCRIPTION
Gaming Event
Kids Play Time
Art Space
Korean Festival
Travis Concert

Download CSV



# 1.8 Coding

## The procedures:

```
SQL Worksheet
```

```
1 CREATE OR REPLACE PROCEDURE GetWorkerJobTitle (  
2     p_jobTitle IN VARCHAR  
3 )  
4 AS  
5 BEGIN  
6     FOR worker_event IN (  
7         SELECT  
8             w.workerID,  
9             w.Name ,  
10            w.ContactInfo ,  
11            w.JobTitle,  
12            w.Shift,  
13            e.EventID,  
14            e.Description ,  
15            e.EventDate,  
16            e.Location  
17        FROM  
18            Worker w  
19        INNER JOIN  
20            WorkerEventAssignment wea ON w.workerID = wea.WorkerID  
21        INNER JOIN  
22            Event e ON wea.EventID = e.EventID  
23        WHERE  
24            w.JobTitle = p_jobTitle  
25        )  
26     LOOP  
27         DBMS_OUTPUT.PUT_LINE('Worker Details: ' || worker_event.Name ||  
28             ' Contact Info: ' || worker_event.ContactInfo ||  
29             ' Event Details: ' || worker_event.Description ||  
30             ' Event Location: ' || worker_event.Location);  
31     END LOOP;  
32 END GetWorkerJobTitle;
```

Procedure created.

2024 Oracle - Live SQL 24.1.3, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym

```
SQL Worksheet
```

```
16     e.EventTime,  
17     e.Location  
18 FROM  
19     Worker w  
20 INNER JOIN  
21     WorkerEventAssignment wea ON w.workerID = wea.WorkerID  
22 INNER JOIN  
23     Event e ON wea.EventID = e.EventID  
24 WHERE  
25     w.JobTitle = p_jobTitle  
26 )  
27 LOOP  
28     DBMS_OUTPUT.PUT_LINE('Worker Details: ' || worker_event.Name ||  
29         ' Contact Info: ' || worker_event.ContactInfo ||  
30         ' Event Details: ' || worker_event.Description ||  
31         ' Event Location: ' || worker_event.Location);  
32 END LOOP;  
33 END GetWorkerJobTitle;
```

Procedure created.

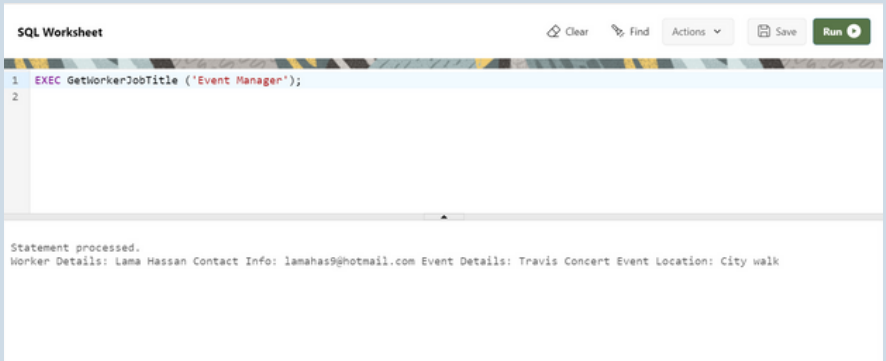
```
SQL Worksheet
```

```
20     INNER JOIN  
21         WorkerEventAssignment wea ON w.workerID = wea.WorkerID  
22     INNER JOIN  
23         Event e ON wea.EventID = e.EventID  
24     WHERE  
25         w.JobTitle = p_jobTitle  
26 )  
27 LOOP  
28     DBMS_OUTPUT.PUT_LINE('Worker Details: ' || worker_event.Name ||  
29         ' Contact Info: ' || worker_event.ContactInfo ||  
30         ' Event Details: ' || worker_event.Description ||  
31         ' Event Location: ' || worker_event.Location);  
32 END LOOP;  
33 END GetWorkerJobTitle;
```

Procedure created.

# 1.8 Coding

Call procedures:



The image shows a screenshot of an SQL Worksheet application. The title bar reads "SQL Worksheet". On the right side of the toolbar, there are buttons for "Clear", "Find", "Actions" (with a dropdown arrow), "Save", and "Run" (with a play icon). The main text area contains two lines of SQL code: `1 EXEC GetWorkerJobTitle ('Event Manager');` and `2` on the next line. Below the code editor, the output area displays the message "Statement processed." followed by the results: "Worker Details: Lama Hassan Contact Info: lamahas9@hotmail.com Event Details: Travis Concert Event Location: City walk".

```
SQL Worksheet
```

Clear Find Actions Save Run

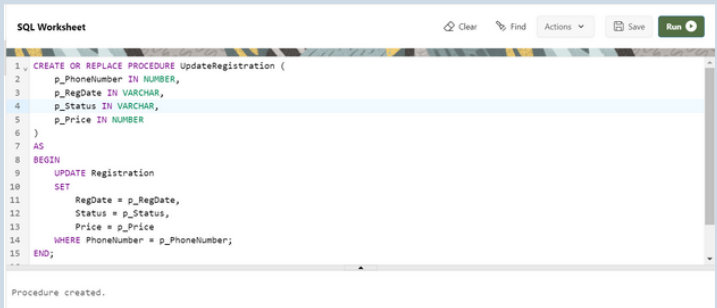
```
1 EXEC GetWorkerJobTitle ('Event Manager');
```

```
2
```

Statement processed.  
Worker Details: Lama Hassan Contact Info: lamahas9@hotmail.com Event Details: Travis Concert Event Location: City walk

# 1.8 Coding

## The Updated procedures:

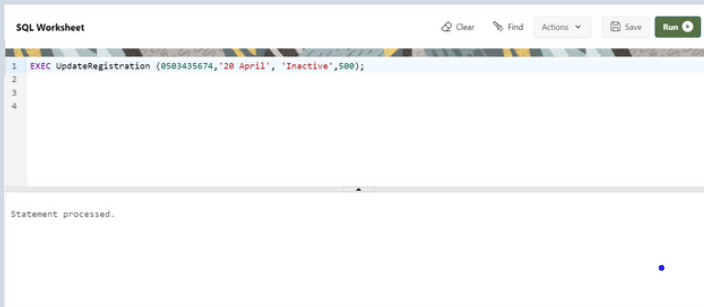


The screenshot shows an SQL Worksheet interface with a toolbar at the top containing 'Clear', 'Find', 'Actions', 'Save', and 'Run' buttons. The main area contains a SQL script for creating and updating a procedure. The script is as follows:

```
1 CREATE OR REPLACE PROCEDURE UpdateRegistration (  
2   p_PhoneNumber IN NUMBER,  
3   p_RegDate IN VARCHAR,  
4   p_Status IN VARCHAR,  
5   p_Price IN NUMBER  
6 )  
7 AS  
8 BEGIN  
9   UPDATE Registration  
10  SET  
11    RegDate = p_RegDate,  
12    Status = p_Status,  
13    Price = p_Price  
14  WHERE PhoneNumber = p_PhoneNumber;  
15 END;
```

Below the script, the status 'Procedure created.' is displayed.

## Call procedures:



The screenshot shows the same SQL Worksheet interface. The script area now contains a single line of code to execute the procedure:

```
1 EXEC UpdateRegistration (0503435674, '20 April', 'Inactive', 500);  
2  
3  
4
```

Below the script, the status 'Statement processed.' is displayed.

## 1.9 Project Team Task

Tasks	Made by
E-R Diagram, Problem description, Relational schema, Logical model, Update procedures, Editing	Shifa Albadri
E-R Diagram, Normalization, Functional dependencies, Select procedures, Update procedures, Editing	Hadeel Alharthi
E-R Diagram, Query (1,2,3), Create & Insert into tables (Sponsor, VisitorEventAttendance, Worker, Sponsorship, Visitor)	Aya Alhazmi
E-R Diagram, Query (4,5), Create & Insert into tables (Volunteer, Event, Registration, VolunteerEventAssignment, WorkerEventAssignment)	Kholod Althbeny

