**Database System**

*Project Title*: **Event Management System**

*Submitted by* ***:***  *Submitted to :*

Tayyaba Abbas BSEF21M011 Sir. Asif Sohail

Laiba Shakoor BSEF21M021

Aalia Siddique BSEF21M034



**University of Punjab**

Faculty of Computing and Information Technology, Allama Iqbal Campus

***Introduction of Event Management System:***

The **Event Management System** (EMS) orchestrates seamless event execution by integrating a **network** of interrelated tables. The core "Event" table serves as a hub, linking to diverse subtypes like exhibitions, conferences, and workshops, each with specific attributes. Organizers, attendees, staff, and resources are efficiently managed through dedicated tables. Marketing strategies, feedback, and budgeting are streamlined, ensuring comprehensive event **coordination**. Attendee information is captured, and venue and ticket details are synchronized. The system's modular design allows for **flexibility** and **adaptability**, making it a robust solution for the **end-to-end management** of various events with precision and ease.

***Entity Relation Diagram of System:***

The following Entity-Relationship Diagram (ERD) represents the relationships of different entities in this **Event Management System** with respect to their attributes and primary as well as foreign keys. It also describes this system’s 1:1, 1:M, M:N relationships.

1. ***ERD Model:***

The entities for **Event Management System** are as follows:

1. Event
2. Workshop
3. Conference
4. Concert
5. Exhibition
6. Sport
7. Organizer
8. Venue
9. Attendee
10. Customer
11. Staff
12. Catering
13. Security
14. Technical crew
15. Décor
16. Resource
17. Budget
18. Ticket
19. Marketing
20. Digital
21. Traditional
22. Feedback
23. ***Attributes of Entities:***

|  |  |
| --- | --- |
| Entities | Attributes |
| Event | eventID, E\_type, E\_time, E\_date, E\_status |
| Workshop | eventID, ws\_instructor, ws\_duration, ws\_category |
| Exhibition | eventID, ex\_exhibitor, ex\_booth, ex\_booth\_des |
| Conference | eventID, conf\_speakerID, conf\_speakerName |
| Concert | eventID, con\_artistID, con\_artist |
| Sport | eventID,teamID, spo\_name, spo\_result, spo\_bestPlayer,spo\_refree |
| Organizer | orgID, org\_name, org\_DOB, org\_cnic, org\_email, org\_phone#, org\_address, org\_sal |
| Venue | venueID, V\_capacity, V\_name, V\_address |
| Attendee | attendeeID, att\_name, att\_cnic, att\_phone#, att\_email |
| Customer | customerID, c\_name, c\_address, c\_email, c\_phone#,c\_cnic |
| Ticket | ticketID, T\_type, T\_price, T\_availibility |
| Staff | staffID, S\_name, S\_address, S\_email, S\_phone#, S\_DOB, S\_type, S\_salary, S\_cnic |
| Catering | staffID, cat\_category |
| Technical\_Crew | staffID, TC\_role |
| Security | staffID, sec\_weapons1,sec\_weapons2 |
| Décor | staffID, D\_theme,D\_floralArrangements, D\_  sittingArrangements ,D\_sittingArrangements |
| Resource | resourceID, R\_name, R\_price, R\_qty |
| Budget | budgetID, B\_totalAmount, B\_amountPaid, B\_amountDue, B\_estimatedAmount |
| Marketing | marketingID, M\_type, M\_cost |
| Traditional | marketingID ,TM\_material, TM\_mediaOutlet,  TM\_printProduction |
| Digital | marketingID, DM\_socialMedia, DM\_email, DM\_SEO,  DM\_strategy |
| Feedback | feedbackID, fb\_rating, fb\_comments |

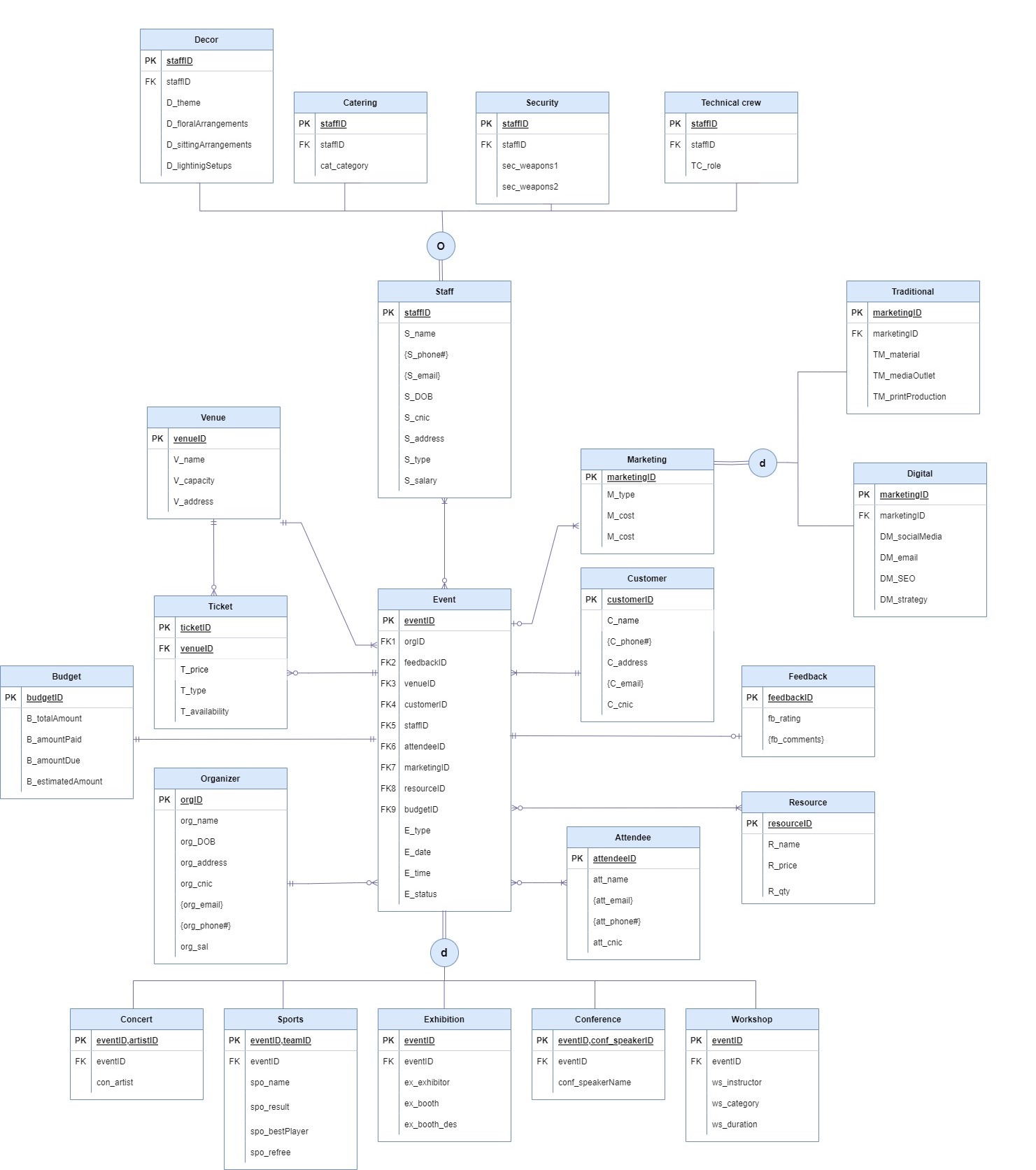
1. ***Connectivity Table:***

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity** | **Relationship** | **Connectivity** | **Entity** |
| Event | Is attended by | M:M | Attendee |

|  |  |  |  |
| --- | --- | --- | --- |
| Staff | Works in | M:M | Event |

|  |  |  |  |
| --- | --- | --- | --- |
| Event | Has used | M:M | Resource |

1. ***ERD Diagram:***



**Relational Schema by using both bottom-up approach**

**and top-down approach**

***Top Down Approach***

**1.Event**(eventID,E\_type,E\_date,E\_time,E\_status,orgID,feedbackID,customerID,staffID,attendeeID,ticketID,maketingID,resourceID,budgetID)

* **1NF (remove multivalued artibute)**

This table is already in 1NF because there is no multivalued artibute in event table.

* **2NF(remove partial dependency)**

This table is already in 2NF because there is no partial dependency in event table.All non-key artibutes are fully functional dependent on primary key**(eventID).**

* **3NF(remove transitive dependency)**

The table is already in 3NF because there is no transitive dependency exist in this table.

**The final normalized form is**

**Event**(eventID,E\_type,E\_date,E\_time,E\_status,orgID,feedbackID,customerID,staffID, ticketID ,attendeeID,maketingID,resourceID,budgetID)

2.**Organizer(**orgID,org\_name,org\_DOB,org\_address,org\_cnic,org\_email,org\_phone#,org\_sal)

* **1NF (remove multivalued artibute)**

There are two multivalued artibutes org\_email,org\_phone# so we decompose in separate table

**Organizer(**orgID,org\_name,org\_DOB,org\_address,org\_cnic, org\_sal)

**OrgEmail(**orgID(fk), org\_email)

**orgPhone**(orgID(fk),org\_phone#)

* **2NF(remove partial dependency)**

These tables are already in 2NF because there is no partial dependency .All non-key artibutes are fully functional dependent on primary keys.

* **3NF(remove transitive dependency)**

These tables are already in 3NF because there is no transitive dependency exist in these tables.

**The final normalized form is:**

**Organizer(**orgID,org\_name,org\_DOB,org\_address,org\_cnic, org\_sal)

**OrgEmail(**orgID(fk), org\_email)

**orgPhone**(orgID(fk),org\_phone#)

**3.Venue**(venueID,V\_name,V\_capacity,V\_address)

* **1NF (remove multivalued artibutes)**

This table is already in 1NF because there is no multivalued artibute in this table.

* **2NF(remove partial dependency)**

This table is already in 2NF because there is no partial dependency in this table.All non-key artibutes are fully functional dependent on primary key**(orgID).**

* **3NF(remove transitive dependency)**

The table is already in 3NF because there is no transitive dependency exist in this table.

**The final normalized form is:**

**Venue**(venueID,V\_name,V\_capacity,V\_address)

**4.Ticket**(ticketID, venueID ,T\_price,T\_type,T\_availibility)

* **1NF (remove multivalued artibutes)**

This table is already in 1NF because there is no multivalued artibute in this table.

* **2NF(remove partial dependency)**

This table is already in 2NF because there is no partial dependency in this table.All non-key artibutes are fully functional dependent on primary key**(ticketID).**

* **3NF(remove transitive dependency)**

There is transitive dependency exist in this table as

**T\_type->T\_price**

Both T\_type and T\_price are non-prime artibutes

**Ticket**(ticketID,T\_type,T\_availibility, venueID)

**TicketPrice**(T\_type, T\_price)

**The final normalized form is:**

**Ticket**(ticketID,T\_type,T\_availibility, venueID)

**TicketPrice**(T\_type, T\_price)

**5.Attendee**(attendeeID,Att\_name,Att\_email,Att\_phone#,Att\_cnic)

* **1NF (remove multivalued artibute)**

There are two multivalued artibutes Att\_email,Att\_phone# so we decompose in separate table

**Attendee**(attendeeID,Att\_name, Att\_cnic)

**Atten\_Email**(attendeeID(fk), Att\_email)

**Atten\_Phone**(attendeeID(fk), Att\_phone#)

* **2NF(remove partial dependency)**

These tables are already in 2NF because there is no partial dependency .All non-key artibutes are fully functional dependent on primary keys.

* **3NF(remove transitive dependency)**

These tables are already in 3NF because there is no transitive dependency exist in these tables.

**The final normalized form is:**

**Attendee**(attendeeID,Att\_name, Att\_cnic)

**Atten\_Email**(attendeeID(fk), Att\_email)

**Atten\_Phone**(attendeeID(fk), Att\_phone#)

**6.Customer**(customerID,C\_name,C\_phone#,C\_address,C\_email,C\_cnic)

* **1NF (remove multivalued artibute)**

There are two multivalued artibutes C\_email,C\_phone# so we decompose in separate table

**Customer(**customerID,C\_name, C\_address,C\_cnic)

**Cust\_Email(**customerID(fk), C\_email)

**Cust\_Phone(**customer(fk), C\_phone#)

* **2NF(remove partial dependency)**

These tables are already in 2NF because there is no partial dependency .All non-key artibutes are fully functional dependent on primary keys.

* **3NF(remove transitive dependency)**

These tables are already in 3NF because there is no transitive dependency exist in these tables.

**The final normalized form is:**

**Customer(**customerID,C\_name, C\_address,C\_cnic)

**Cust\_Email(**customerID(fk), C\_email)

**Cust\_Phone(**customerID(fk), C\_phone#)

**7. Staff**(staffID,S\_name,S\_phone#,S\_email,S\_DOB,S\_address,S\_type,S\_salary)

* **1NF (remove multivalued artibute)**

There are two multivalued artibutes S\_email,S\_phone# so we decompose in separate table

**Staff**(staffID,S\_name, S\_DOB,S\_address,S\_type,S\_salary)

**Staff\_Email**(staffID(fk), S\_email)

**Staff\_Phone**(staffID(fk), ,S\_phone#)

* **2NF(remove partial dependency)**

These tables are already in 2NF because there is no partial dependency .All non-key artibutes are fully functional dependent on primary keys.

* **3NF(remove transitive dependency)**

Transitive dependency exist in staff table as

**S\_type->S\_salary**

The S->type and S\_salary both are non-prime artibutes.

**Staff**(staffID,S\_name, S\_DOB,S\_address,S\_type)

**Staff\_sal(**S\_type, S\_salary)

**Staff\_Email**(staffID(fk), S\_email)

**Staff\_Phone**(staffID(fk),S\_phone#)

**The final normalized form is:**

**Staff**(staffID,S\_name, S\_DOB,S\_address,S\_type)

**Staff\_sal(**S\_type, S\_salary)

**Staff\_Email**(staffID(fk), S\_email)

**Staff\_Phone**(staffID(fk), ,S\_phone#)

**8.Feedback**(feedbackID,fb\_ratings,fb\_comments)

* **1NF (remove multivalued artibutes)**

This table exist multivalued artibute e.g fb\_comments

**Feedback**(feedbackID,fb\_comments)

**Feedback\_comm**(feedbackID(fk), fb\_comments)

* **2NF(remove partial dependency)**

These tables are already in 2NF because there is no partial dependency .All non-key artibutes are fully functional dependent on primary keys.

* **3NF(remove transitive dependency)**

These tables are already in 3NF because there is no transitive dependency exist in these tables.

**The final normalized form is:**

**Feedback**(feedbackID,fb\_ratings)

**Feedback\_comm**(feedbackID(fk), fb\_comments)

**9.Resources**(resourceID,R\_name,R\_price,R\_qty)

* **1NF (remove multivalued artibutes)**

This table is already in 1NF because there is no multivalued artibute in this table.

* **2NF(remove partial dependency)**

These tables are already in 2NF because there is no partial dependency .All non-key artibutes are fully functional dependent on primary keys.

* **3NF(remove transitive dependency)**

These tables are already in 3NF because there is no transitive dependency exist in these tables.

**Final normalized form is:**

**Resources(**(resourceID,R\_name,R\_price,R\_qty)

10**.Budget**(budgetID,B\_totalAmount,B\_amountPaid,B\_amountDue,B\_estimatedAmount)

* **1NF (remove multivalued artibutes)**

This table is already in 1NF because there is no multivalued artibute in this table.

* **2NF(remove partial dependency)**

These tables are already in 2NF because there is no partial dependency .All non-key artibutes are fully functional dependent on primary keys.

* **3NF(remove transitive dependency)**

These tables are already in 3NF because there is no transitive dependency exist in these tables.

**Final normalized form is:**

**Budget**(budgetID,B\_totalAmount,B\_amountPaid,B\_amountDue,B\_estimatedAmount)

**11.Marketing**(marketingID,M\_type,M\_cost)

**This table is aleardy in 1NF,2NF and 3NF**

# ***Bottom Up Approach***

**Relation**(eventID,E\_type,E\_date,E\_time,E\_status,orgID,org\_name,org\_DOB,org\_address,org\_cnic,org\_email,org\_phone#,org\_sal,feedbackID,fb\_ratings,fb\_comments,venueID,V\_name,V\_capacity,V\_address,customerID,C\_name,C\_phone#,C\_address,C\_email,C\_cnic,staffID,S\_name,S\_phone#,S\_email,S\_DOB,S\_address,S\_type,S\_salary,attendeeID,Att\_name,Att\_email,Att\_phone#,Att\_cnic,maketingID,M\_type,M\_cost,resourceID,R\_name,R\_price,budgetID,B\_totalAmount,B\_amountPaid,B\_amountDue,B\_estimatedAmount,ticketID,T\_price,T\_type,T\_availibility,qty)

* **1NF(remove multivalued artibutes)**

**Event(**eventID,E\_type,E\_date,E\_time,E\_status,orgID,org\_name,org\_DOB,org\_address,org\_cnic,org\_sal,feedbackID,fb\_ratings,venueID,V\_name,V\_capacity,V\_address,customerID,C\_name,C\_address,C\_cnic,staffID,S\_name,S\_DOB,S\_address,S\_type,S\_salary,attendeeID,Att\_name,Att\_cnic,marketingID,M\_type,M\_cost,resourceID,R\_name,R\_price,budgetID,B\_totalAmount,B\_amountPaid,B\_amountDue,B\_estimatedAmount,ticketID,T\_price,T\_type,T\_availibility,qty)

**OrgEmail(**orgID(fk), org\_email)

**orgPhone**(orgID(fk),org\_phone#)

**Atten\_Email**(attendeeID(fk), Att\_email)

**Atten\_Phone**(attendeeID(fk), Att\_phone#)

**Cust\_Email(**customerID(fk), C\_email)

**Cust\_Phone(**customer(fk), C\_phone#)

**Staff\_Email**(staffID(fk), S\_email)

**Staff\_Phone**(staffID(fk), ,S\_phone#)

**Feedback\_comm**(feedbackID(fk), fb\_comments)

* **2NF(remove partial dependency)**

All these relations are already in 2NF because there in no partial dependency exitst in these table.

* **3NF(remove transitive dependency)**

**1.Event(**eventID,E\_type,E\_date,E\_time,E\_status,orgID,feedbackID,,venueID,customerID,staffID,attendeeID, marketingID,resourceID, ticketID)

**organizer(**orgID,org\_name,org\_DOB,org\_address,org\_cnic,org\_sal)

**Venue(**venueID**,**V\_name,V\_capacity,V\_address,ticketID,T\_price,

T\_type,T\_availibility)

This table is further normalized in 3NF because there is transitive dependency

**tickeID-> T\_price,T\_type,T\_availibility**

**Ticket**(tickeID,T\_price,T\_type,T\_availibility,venueID)

* There is transitive dependency in this event\_ticket table

T\_type->T\_price

**ticket\_price(**T\_type,T\_price)

**Venue(**venueID**,**V\_name,V\_capacity,V\_address)

**Feedback(**feedbackID,fb\_ratings)

**Attendee**(attendeeID,Att\_name, Att\_cnic)

**Customer(**customerID,C\_name, C\_address,C\_cnic)

**Staff**(staffID,S\_name, S\_DOB,S\_address,S\_type,S\_salary)

This table is more normalized because there is transitive dependency between

**S\_type->S\_salary**

**Staff**(staffID,S\_name, S\_DOB,S\_address,S\_type)

**Staff\_sal(**S\_type, S\_salary)

**Budget**(budgetID,B\_totalAmount,B\_amountPaid,B\_amountDue,B\_estimatedAmount)

**Marketing**(marketingID,M\_type,M\_cost)

**All other tables other than event are already in 3NF**

**Final normalized form is:**

1. **1.Event(**eventID,E\_type,E\_date,E\_time,E\_status,orgID,feedbackID,customerID,staffID,attendeeID, marketingID,resourceID, ticketID)
2. **Organizer(**orgID,org\_name,org\_DOB,org\_address,org\_cnic,org\_sal)
3. **Ticket**(tickeID,T\_price,T\_type,T\_availibility,venueID)
4. **ticket\_price(**T\_type,T\_price)
5. **Venue(**venueID**,**V\_name,V\_capacity,V\_address)
6. **Feedback(**feedbackID,fb\_ratings)
7. **Attendee**(attendeeID,Att\_name, Att\_cnic)
8. **Customer(**customerID,C\_name, C\_address,C\_cnic)
9. **Staff**(staffID,S\_name, S\_DOB,S\_address,S\_type)
10. **Staff\_sal(**S\_type, S\_salary)
11. **Budget**(budgetID,B\_totalAmount,B\_amountPaid,B\_amountDue,B\_estimatedAmount)
12. **Marketing**(marketingID,M\_type,M\_cost)
13. **OrgEmail(**orgID(fk), org\_email)
14. **orgPhone**(orgID(fk),org\_phone#)
15. **Atten\_Email**(attendeeID(fk), Att\_email)
16. **Atten\_Phone**(attendeeID(fk), Att\_phone#)
17. **Cust\_Email(**customerID(fk), C\_email)
18. **Cust\_Phone(**customer(fk), C\_phone#)
19. **Staff\_Email**(staffID(fk), S\_email)
20. **Staff\_Phone**(staffID(fk), ,S\_phone#)
21. **Resources**(resourceID,R\_name,R\_price,qty)
22. **Feedback\_comm**(feedbackID(fk), fb\_comments)

***Description of relations***

1. ***Table Name: Event***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| eventID | Number | 4 | Primary Key |
| E\_type | Varchar2 | 15 | Can be ‘CONCERT’, ‘WORKSHOP’, ‘CONFERENCE’, ‘EXHIBITION’,‘SPORT’ |
| E\_date | Date |  |  |
| E\_time | Varchar2 | 8 |  |
| E\_status | Varchar2 | 10 | Can be ‘COMPLETED’, ‘PENDING’ |
| budgetID | Number | 4 | Foreign Key |
| orgID | Number | 4 | Foreign Key |
| venueID | Number | 4 | Foreign Key |
| StaffID | Number | 4 | Foreign Key |
| FeedbackID | Number | 4 | Foreign Key |
| CustomerID | Number | 4 | Foreign Key |
| AttendeeID | Number | 4 | Foreign Key |
| MarketingID | Number | 4 | Foreign Key |
| ResourceID | Number | 4 | Foreign Key |

1. ***Table Name: Exhibition***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| eventID | Number | 4 | Primary Key, Foreign Key |
| ex\_exhibitor | Varchar2 | 15 | Not null |
| ex\_booth | Number | 5 |  |
| ex\_booth\_des | Varchar2 | 25 |  |

1. ***Table Name: Conference***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| eventID | Number | 4 | Primary Key, Foreign Key |
| conf\_speakerID | Number | 4 | Primary Key |
| conf\_speakerName | Varchar2 | 25 |  |

1. ***Table Name: Workshop***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| eventID | Number | 4 | Primary Key, Foreign Key |
| ws\_duration | Varchar2 | 10 |  |
| ws\_instructor | Varchar2 | 25 | Not null |
| ws\_category | Varchar2 | 25 | Can be ‘EDUCATIONAL’, ‘SKILL BUILDING','TRAINING','INTREPRENEURSHIP','HEALTH' |

1. ***Table Name: Concert***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| eventID | Number | 4 | Primary Key, Foreign Key |
| artistID | Number | 4 | Primary Key |
| Con\_artist | Varchar2 | 25 | Not null |

1. ***Table Name: Sport***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| eventID | Number | 4 | Primary Key, Foreign Key |
| teamID | Number | 4 | Primary Key |
| spo\_name | Varchar2 | 25 | Not null |
| spo\_result | Varchar2 | 10 | Can be ‘WON','LOST','WITHDRAW' |
| spo\_bestPlayer | Varchar2 | 25 |  |
| spo\_refree | Varchar2 | 25 |  |

1. ***Table Name: Organizer***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| orgID | Number | 4 | Primary Key |
| org\_name | Varchar2 | 25 | Not null |
| org\_DOB | Date |  |  |
| org\_cnic | Varchar2 | 15 |  |
| org\_address | Varchar2 | 50 |  |
| org\_sal | Number | 8 |  |

1. ***Table Name: Org\_Email***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| orgID | Number | 4 | Primary Key, Foreign Key |
| org\_Email | Varchar2 | 20 | Primary Key |

1. ***Table Name: Org\_Phone***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| orgID | Number | 4 | Primary Key, Foreign Key |
| org\_Phone# | Varchar2 | 20 | Primary Key |

1. ***Table Name: Resource***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| resourceID | Number | 4 | Primary Key |
| R\_name | Varchar2 | 25 | Not null |
| R\_price | Number | 10 | Not null |

1. ***Table Name: Venue***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| venueID | Number | 4 | Primary Key |
| V\_name | Varchar2 | 25 | Not null |
| V\_capacity | Number | 4 |  |
| V\_address | Varchar2 | 50 |  |

1. ***Table Name: Staff***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| staffID | Number | 4 | Primary Key |
| S\_name | Varchar2 | 25 | Not Null |
| S\_DOB | Date |  |  |
| S\_cnic | Varchar2 | 15 |  |
| S\_adddress | Varchar2 | 50 |  |
| S\_type | Char | 1 | Can be ‘C’,’D’,’S’,’T’ and Foreign Key |

1. ***Table Name: Staff\_sal***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| S\_type | Char | 1 | Can be ‘C’,’D’,’S’,’T’ and Primary Key |
| S\_salary | Number | 8 | Not Null |

1. ***Table Name: Staff\_Phone***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| staffID | Number | 4 | Primary Key and Foreign Key |
| S\_phone# | Varchar2 | 12 | Primary Key |

1. ***Table Name: Staff\_Email***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| staffID | Number | 4 | Primary Key and Foreign Key |
| S\_email | Varchar2 | 20 | Primary Key |

1. ***Table Name: Security***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| staffID | Number | 4 | Primary Key and Foreign Key |
| sec\_weapons1 | Varchar2 | 15 |  |
| sec\_weapons2 | Varchar2 | 15 |  |

1. ***Table Name: Catering***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| staffID | Number | 4 | Primary Key and Foreign Key |
| Cat\_category | Varchar2 | 15 | Can be ‘BUFFET’,’CORPORATE’,’DESERT’,’BREAKFAST’ |

1. ***Table Name: Technical\_crew***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| staffID | Number | 4 | Primary Key and Foreign Key |
| TC\_role | Varchar2 | 20 | Can be 'CINEMATOGRAPHER','GAFFE','LIGHTINING',  'PROJECTIONIST','RIGGERS |

1. ***Table Name: Decor***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| staffID | Number | 4 | Primary Key and Foreign Key |
| D\_theme | Varchar2 | 20 | Can be 'NEON','BLACK AND WIGHT','VINTAGE |
| D\_floralArrangements | Varchar2 | 20 | Can be 'BOUQUETS','GARLAND',  'CENTERPIECES |
| D\_sittingArrangements | Varchar2 | 20 | Can be 'OPEN','FORMAL','CIRCULAR' |
| D\_lightiningSetups | Varchar2 | 20 | Can be 'NATURAL','PHOTOGRAPHY','THREE POINTS' |

1. ***Table Name: Feedback***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| feedbackID | Number | 4 | Primary Key |
| fb\_rating | Number | 1 |  |

1. ***Table Name: Feedback\_comm***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| feedbackID | Number | 4 | Primary Key and Foreign Key |
| Fb\_comments | Varchar2 | 20 | Primary Key |

1. ***Table Name: Budget***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| budgetID | Number | 4 | Primary Key |
| B\_totalAmount | Number(10,2) | 25 |  |
| B\_amountPaid | Number(10,2) |  |  |
| B\_amountDue | Number(10,2) | 15 |  |
| B\_estimatedAmount | Number(10,2) | 50 |  |

1. ***Table Name: Marketing***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| marketingID | Number | 4 | Primary Key |
| M\_type | Char | 1 | Can be ‘D’, ’T’ |
| M\_cost | Number | 8 | Not Null |

1. ***Table Name:Traditional***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| marketingID | Number | 4 | Primary Key,Foreign Key |
| TM\_material | Varchar2 | 40 | Can be 'BROCHURES', 'FLYERS','BUSINESS CARDS' |
| TM\_mediaOutlet | Varchar2 | 10 | Can be 'RADIO', 'NEWSPAPER','MAGZINE','TV' |
| TM\_printProduction | Varchar2 | 40 | Can be 'RADIO', 'NEWSPAPER','MAGZINE','ARTICLE’ |

1. ***Table Name:Digital***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| marketingID | Number | 4 | Primary Key,Foreign Key |
| DM\_socialMedia | Varchar2 | 40 | Can be ‘INSTRAGRAM’,’FACEBOOK’,’TWITTER’,  ‘TIKTOK’,,'LINKEDIN','YOUTUBE','SNAPCHAT',  'TUMBLR' |
| DM\_email | Varchar2 | 20 | Not Null |
| DM\_SEO | Varchar2 | 25 | Can be 'TECHNICAL SEO','UX','OFF PAGE','ON PAGE','KEYWORD RESEARCH' |
| DM\_strategy | Varchar2 | 30 | CONTENT CREATION', 'SEO INTEGRATION', 'SOCIAL MEDIA MARKETING', 'EMAIL MARKETING', 'STORYTELLING' |

1. ***Table Name: Attendee***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| attendeeID | Number | 4 | Primary Key |
| att\_name | Varchar2 | 25 | Not Null |
| att\_cnic | Varchar2 | 15 |  |

1. ***Table Name: Atten\_Email***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| attendeeID | Number | 4 | Primary Key,Foreign key |
| att\_email | Varchar2 | 20 | Primary key |

1. ***Table Name: Atten\_Phone***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| attendeeID | Number | 4 | Primary Key,Foreign key |
| att\_phone# | Varchar2 | 12 | Primary key |

1. ***Table Name: Customer***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| customerID | Number | 4 | Primary Key |
| C\_name | Varchar2 | 25 | Not Null |
| C\_addres | Varchar2 | 50 |  |
| C\_cnic | Varchar2 | 15 |  |

1. ***Table Name: Cust\_Email***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| customerID | Number | 4 | Primary Key,Foreign key |
| C\_email | Varchar2 | 20 | Primary key |

1. ***Table Name: Atten\_Phone***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| customerID | Number | 4 | Primary Key,Foreign key |
| C\_phone# | Varchar2 | 12 | Primary key |

1. ***Table Name: Ticket***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| ticketID | Number | 4 | Primary Key |
| venueID | Number | 4 | Foreign Key |
| T\_type | Varchar2 | 25 | Can be 'VIP','REGULAR',Foreign key |
| T\_availability | Char | 1 | Can be 'Y','N' |
| C\_cnic | Varchar2 | 15 |  |

1. ***Table Name: TicketPrice***

|  |  |  |  |
| --- | --- | --- | --- |
| Attributes | Data Type | Size | Constraints |
| T\_type | Varchar2 | 25 | Can be 'VIP','REGULAR',Primary key |
| T\_PRICE | Number | 4 |  |

***Table Creation***

1. ***Event table:***

create table Event

(

eventID NUMBER(4) CONSTRAINT eventID\_PK PRIMARY KEY,

E\_type VARCHAR2(15) CONSTRAINT ETYPE\_CHK CHECK(E\_type IN ('CONCERT','CONFERENCE', 'WORKSHOP', 'EXHIBITION', 'SPORT')),

E\_date DATE,

E\_time VARCHAR2(8),

E\_status VARCHAR2(10) CONSTRAINT ESTATUS\_CHK CHECK(E\_status IN ('COMPLETED', 'PENDING')),

budgetID NUMBER(4),

orgID NUMBER(4) ,

staffID NUMBER(4),

venueID NUMBER(4),

feedbackID NUMBER(4),

customerID NUMBER(4),

attendeeID NUMBER(4),

marketingID NUMBER(4),

resourceID NUMBER(4),

CONSTRAINT EVENT\_ORGID\_FK FOREIGN KEY(orgID) REFERENCES Organizer(orgID),

CONSTRAINT EVENT\_FEEDBACKID\_FK FOREIGN KEY(feedbackID) REFERENCES Feedback(feedbackID),

CONSTRAINT EVENT\_VENUEID\_FK FOREIGN KEY(venueID) REFERENCES Venue(venueID),

CONSTRAINT EVENT\_CUSTOMERID\_FK FOREIGN KEY(customerID) REFERENCES Customer(customerID),

CONSTRAINT EVENT\_STAFF\_FK FOREIGN KEY(staffID) REFERENCES Staff(staffID),

CONSTRAINT EVENT\_ATTENDEE\_FK FOREIGN KEY(attendeeID) REFERENCES Attendee(attendeeID),

CONSTRAINT EVENT\_MARKETING\_FK FOREIGN KEY(marketingID) REFERENCES Marketing(marketingID),

CONSTRAINT EVENT\_BUDGET\_FK FOREIGN KEY(budgetID) REFERENCES Budget(budgetID),

CONSTRAINT EVENT\_RES\_FK FOREIGN KEY (resourceID) REFERENCES Res(resourceID) );

**Event Subtypes:**

1. ***Exhibition Event:***

CREATE TABLE Exhibition

(

eventID NUMBER(4),

ex\_exhibitor VARCHAR2(15) CONSTRAINT EXHIBITION\_NN NOT NULL,

ex\_booth NUMBER(5),

ex\_booth\_des VARCHAR2(25),

CONSTRAINT EXHIBITION\_FK FOREIGN KEY(eventID) REFERENCES Event(eventID),

CONSTRAINT EXHIBITION\_PK PRIMARY KEY(eventID)

);

1. ***Conference Event:***

CREATE TABLE Conference

(

eventID NUMBER(4),

conf\_speakerID NUMBER(4),

conf\_speakerName VARCHAR2(25),

CONSTRAINT CONFERENCE\_PK PRIMARY KEY (eventID, conf\_speakerID ),

CONSTRAINT CONFERENCE\_FK FOREIGN KEY (eventID) REFERENCES Event(eventID)

);

1. ***Workshop Event:***

CREATE TABLE Workshop

(

eventID NUMBER(4),

ws\_instructor VARCHAR2(25) CONSTRAINT WORKSHOP\_NN NOT NULL,

ws\_duration VARCHAR2(10),

ws\_category VARCHAR2(25) CONSTRAINT WORKSHOP\_CATEGORY\_CHK CHECK(ws\_category IN ('EDUCATIONAL','SKILL BUILDING','TRAINING','INTREPRENEURSHIP','HEALTH')),

CONSTRAINT WORKSHOP\_PK PRIMARY KEY (eventID),

CONSTRAINT WORKSHOP\_FK FOREIGN KEY (eventID) REFERENCES Event(eventID)

);

1. ***Concert Event:***

CREATE TABLE Concert

(

eventID NUMBER(4),

artistID NUMBER(4),

con\_artist VARCHAR2(25) CONSTRAINT CONCERT\_ARTIST\_NN NOT NULL,

CONSTRAINT CONCERT\_PK PRIMARY KEY (eventID, artistID),

CONSTRAINT CONCERT\_FK FOREIGN KEY (eventID) REFERENCES Event(eventID)

);

1. ***Sports Event:***

CREATE TABLE Sport

(

eventID NUMBER(4),

teamID NUMBER(4),

spo\_name VARCHAR2(25) CONSTRAINT SPORT\_NAME\_NN NOT NULL,

spo\_result VARCHAR2(10) CONSTRAINT SPORT\_RESULT\_CHK CHECK(spo\_result IN('WON','LOST','WITHDRAW')),

spo\_bestPlayer VARCHAR2(25),

spo\_refree VARCHAR2(25),

CONSTRAINT SPORT\_FK FOREIGN KEY(eventID) REFERENCES Event(eventID),

CONSTRAINT SPORT\_PK PRIMARY KEY(eventID,teamID)

);

1. ***Organizer table:***

CREATE TABLE Organizer

(

orgID NUMBER(4) CONSTRAINT ORGID\_PK PRIMARY KEY,

org\_name VARCHAR(25) CONSTRAINT ORGNAME\_NN NOT NULL,

org\_DOB DATE,

org\_cnic VARCHAR2(15),

org\_address VARCHAR2(50) ,

org\_salary NUMBER(8)

)

1. ***Org\_Email table:***

create table Org\_Email

(

orgID number(4),

org\_email varchar2(20),

CONSTRAINT ORGANIZER\_ID\_Email\_PK PRIMARY KEY(orgID, org\_email),

CONSTRAINT ORGANIZER\_FK FOREIGN KEY(orgID) REFERENCES Organizer(orgID) )

1. ***Org\_Phone table:***

create table Org\_Phone

(

orgID number(4),

org\_phone# varchar2(20),

CONSTRAINT ORGANIZER\_ID\_PHONE\_PK PRIMARY KEY(orgID, org\_phone#),

CONSTRAINT ORGANIZER\_PH\_FK FOREIGN KEY(orgID) REFERENCES Organizer(orgID)

)

1. ***Resource table:***

CREATE TABLE Res

(

resourceID NUMBER(4) CONSTRAINT RESOURCE\_ID\_PK PRIMARY KEY,

R\_name VARCHAR2(25) CONSTRAINT R\_NAME\_NN NOT NULL,

R\_price NUMBER(10) CONSTRAINT R\_PRICE\_NN NOT NULL,

R\_qty NUMBER(3)

)

1. ***Marketing table:***

create table Marketing

(

marketingID NUMBER(4) CONSTRAINT MARK\_ID\_PK PRIMARY KEY,

M\_type CHAR(1) CONSTRAINT M\_TYPE\_CHK CHECK(M\_type in('D','T')),

M\_cost number(8) CONSTRAINT M\_COST\_NN NOT NULL

)

**Marketing Subtypes:**

1. ***Traditional Marketing****:*

create table Traditional

(

marketingID NUMBER(4) CONSTRAINT TRAD\_MARK\_ID\_PK PRIMARY KEY,

TM\_material VARCHAR2(40) CONSTRAINT TRAD\_MATERIAL\_CHK CHECK(TM\_material IN('BROCHURES', 'FLYERS','BUSINESS CARDS')),

TM\_mediaOutlet VARCHAR2(10) CONSTRAINT TRAD\_MEDIA\_CHK CHECK(TM\_mediaOutlet IN('RADIO', 'NEWSPAPER','MAGZINE','TV')),

TM\_printProduction varchar2(40) CONSTRAINT TRAD\_MARK\_PROD\_CHK CHECK(TM\_printProduction IN ('NEWSPAPER','MAGZINE','ARTICLE')),

CONSTRAINT TRADITIONAL\_FK FOREIGN KEY(marketingID) references Marketing(marketingID)

)

1. ***Digital Marketing:***

create table Digital

(

marketingID NUMBER(4) CONSTRAINT DIG\_MARK\_ID\_PK PRIMARY KEY,

DM\_socialMedia varchar2(40) CONSTRAINT DIG\_SOCIAL\_CHK CHECK(DM\_socialMedia IN('INSTAGRAM','FACEBOOK','TWITTER','TIKTOK','LINKEDIN','YOUTUBE','SNAPCHAT','TUMBLR')),

DM\_email varchar2(20) CONSTRAINT DIG\_EMAIL\_NN NOT NULL,

DM\_SEO varchar2(25) CONSTRAINT DIG\_SEO\_CHK CHECK(DM\_SEO IN('TECHNICAL SEO','UX','OFF PAGE','ON PAGE','KEYWORD RESEARCH')),

DM\_strategy varchar2(30) CONSTRAINT DIG\_STRATEGY\_CHK CHECK(DM\_strategy IN ('CONTENT CREATION', 'SEO INTEGRATION', 'SOCIAL MEDIA MARKETING', 'EMAIL MARKETING', 'STORYTELLING')),

CONSTRAINT DIG\_FK FOREIGN KEY(marketingID) references Marketing(marketingID)

)

1. ***Feedback table:***

CREATE TABLE Feedback

(

feedbackID NUMBER(4) CONSTRAINT FEEDBACKID\_PK PRIMARY KEY,

fb\_rating NUMBER(1)

)

1. ***Feedback\_comm table:***

create table Feedback\_comm

(

feedbackID NUMBER(4) ,

fb\_comments VARCHAR2(50),

CONSTRAINT FEEDBACK\_ID\_COMM\_PK PRIMARY KEY(feedbackID , fb\_comments),

CONSTRAINT FEEDBACK\_COMM\_FK FOREIGN KEY(feedbackID) REFERENCES Feedback(feedbackID) )

1. ***Staff table:***

CREATE TABLE Staff

(

staffID NUMBER(4) CONSTRAINT staffID\_PK PRIMARY KEY,

S\_name VARCHAR(25) CONSTRAINT SNAME\_NN NOT NULL,

S\_DOB DATE,

S\_cnic VARCHAR2(15),

S\_address VARCHAR2(50),

S\_type CHAR(1) CONSTRAINT STYPE#\_CHK CHECK(S\_type IN ('C','D','S','T')),

CONSTRAINT Staff\_FK FOREIGN KEY(S\_type) REFERENCES Staff\_sal(S\_type)

)

1. ***Staff\_Phone table:***

create table Staff\_Phone

(

staffID number(4),

S\_phone# varchar2(12),

CONSTRAINT STAFF\_ID\_phone\_PK PRIMARY KEY(staffID , S\_phone#),

CONSTRAINT STAFF\_PH\_FK FOREIGN KEY(staffID) REFERENCES Staff(staffID)

)

1. ***Staff\_Email table:***

create table Staff\_Email

(

staffID number(4),

S\_email varchar2(20),

CONSTRAINT STAFF\_ID\_EMAIL\_PK PRIMARY KEY(staffID , S\_email),

CONSTRAINT STAFF\_EM\_FK FOREIGN KEY(staffID) REFERENCES Staff(staffID)

)

1. ***Staff\_sal table***

create table Staff\_sal

(

S\_type CHAR(1) CONSTRAINT STAFF\_SAL\_CHK CHECK(S\_type IN ('C','D','S','T')),

S\_salary NUMBER(8) CONSTRAINT STAFF\_SAL\_NN NOT NULL ,

CONSTRAINT STAFF\_SAL\_PK PRIMARY KEY(S\_type)

)

**Staff Subtypes:**

1. ***Décor staff:***

create table Decor

(

staffID NUMBER(4) CONSTRAINT DECOR\_PK PRIMARY KEY,

D\_theme VARCHAR(20) CONSTRAINT D\_THEME\_CHK CHECK(D\_theme IN('NEON','BLACK AND WIGHT','VINTAGE')),

D\_floralArrangements VARCHAR2(20) CONSTRAINT D\_FLORAL\_ARR\_CHK CHECK(D\_floralArrangements IN('BOUQUETS','GARLAND','CENTERPIECES')),

D\_sittingArrangements VARCHAR2(20) CONSTRAINT D\_SIT\_ARR\_CHK CHECK(D\_sittingArrangements IN('OPEN','FORMAL','CIRCULAR')),

D\_lightinigSetups VARCHAR2(20) CONSTRAINT D\_LIGHT\_SETUP\_CHK CHECK(D\_lightinigSetups IN('NATURAL','PHOTOGRAPHY','THREE POINTS')),

CONSTRAINT DECOR\_FK FOREIGN KEY(staffID) REFERENCES Staff(staffID)

);

1. ***Catering staff:***

create table Catering

(

staffID NUMBER(4) CONSTRAINT CATERING\_PK PRIMARY KEY,

cat\_category VARCHAR(15) CONSTRAINT CAT\_CATEGORY\_CHK CHECK(cat\_category IN('BUFFET','CORPORATE','DESERT','BREAKFAST')),

CONSTRAINT CAT\_FK FOREIGN KEY(staffID) REFERENCES Staff(staffID)

);

1. ***Security staff:***

create table Security

(

staffID NUMBER(4) CONSTRAINT SECURITY\_PK PRIMARY KEY,

sec\_weapons1 VARCHAR(15),

sec\_weapons2 VARCHAR(15),

CONSTRAINT SECURITY\_FK FOREIGN KEY(staffID) REFERENCES Staff(staffID)

);

1. ***Technical\_crew staff:***

create table Technical\_crew

(

staffID NUMBER(4) CONSTRAINT TC\_PK PRIMARY KEY,

TC\_role VARCHAR(20) CONSTRAINT TC\_ROLE\_CHK CHECK(TC\_role IN('CINEMATOGRAPHER','GAFFE','LIGHTINING','PROJECTIONIST','RIGGERS')),

CONSTRAINT TC\_FK FOREIGN KEY(staffID) REFERENCES Staff(staffID)

);

1. ***Attendee table***

create table Attendee

(

attendeeID number(4) CONSTRAINT ATTENDEE\_ID\_PK PRIMARY KEY,

att\_name VARCHAR2(25) CONSTRAINT ATT\_NAME\_NN NOT NULL

att\_cnic varchar2(15) CONSTRAINT ATT\_CNIC\_UNQ UNIQUE

)

1. ***Atten\_Email table:***

create table Atten\_Email

(

attendeeID number(4),

att\_email varchar2(20),

CONSTRAINT ATTENDEE\_ID\_EMAIL\_PK PRIMARY KEY(attendeeID , att\_email),

CONSTRAINT ATTENDEE\_EM\_FK FOREIGN KEY(attendeeID ) REFERENCES Attendee(attendeeID)

)

1. ***Atten\_Phone table:***

create table Atten\_Phone

(

attendeeID number(4),

att\_phone# varchar2(12),

CONSTRAINT ATTENDEE\_ID\_PHONE\_PK PRIMARY KEY(attendeeID , att\_phone#),

CONSTRAINT ATTENDEE\_PHONE\_FK FOREIGN KEY(attendeeID ) REFERENCES Attendee(attendeeID )

)

1. ***Customer table:***

create table Customer

(

customerID number(4) CONSTRAINT CUSTOMER\_ID\_PK PRIMARY KEY,

C\_name varchar2(25) CONSTRAINT CNAME\_NN NOT NULL,

C\_address varchar2(50),

C\_cnic varchar2(15) CONSTRAINT CCNIC\_UNQ UNIQUE

)

1. ***Cust\_Phone table:***

create table Cust\_Phone

(

customerID number(4),

C\_phone# varchar2(12),

CONSTRAINT CUSTOMER\_ID\_phone\_PK PRIMARY KEY(customerID, C\_phone#),

CONSTRAINT CUSTOMER\_ID\_FK FOREIGN KEY(customerID) REFERENCES Customer(customerID)

)

1. ***Cust\_Email table:***

create table Cust\_Email

(

customerID number(4),

venueID number(4),

C\_email varchar2(20),

CONSTRAINT CUSTOMER\_ID\_Email\_PK PRIMARY KEY(customerID, C\_email),

CONSTRAINT CUSTOMERID\_FK FOREIGN KEY(customerID) REFERENCES Customer(customerID)

)

1. ***Ticket table:***

create table Ticket

(

ticketID NUMBER(4) CONSTRAINT TICKET\_PK PRIMARY KEY,

venueID NUMBER(4),

T\_type VARCHAR2(25) CONSTRAINT Ticket\_TYPE\_CHK CHECK (T\_type IN ('VIP', 'REGULAR')),

T\_availability CHAR(1) CONSTRAINT AVAILABILITY\_CHK CHECK (T\_availability IN ('Y', 'N')),

CONSTRAINT TTYPE\_FK FOREIGN KEY (T\_type) REFERENCES TicketPrice(T\_type),

CONSTRAINT Ticket\_FK FOREIGN KEY (venueID) REFERENCES Venue(venueID))

1. ***TicketPrice table:***

CREATE TABLE TicketPrice

(

T\_type varchar2(25) CONSTRAINT T\_TYPE\_CHK CHECK(T\_type in('VIP','REGULAR')),

T\_PRICE NUMBER(4),

CONSTRAINT TTYPE\_PK PRIMARY KEY(T\_type)

)

1. ***Venue table:***

create table Venue

(

venueID number(4) CONSTRAINT VENUEID\_PK PRIMARY KEY,

V\_name varchar2(25) CONSTRAINT VNAME\_NN NOT NULL,

V\_capacity number(4),

v\_address varchar2(50)

)

1. ***Budget table:***

CREATE TABLE Budget

(

budgetID NUMBER(4) CONSTRAINT BUDGETID\_PK PRIMARY KEY,

B\_totalAmount NUMBER(10,2),

B\_amountPaid NUMBER(10,2),

B\_amountDue NUMBER(10,2),

B\_estimatedAmount NUMBER(10,2) )

***Views***

1. **CREATE A VIEW TO RETRIEVE STAFF AND THEIR SUBTYPES DATA FOR A SPECIFIC EVENT**

CREATE OR REPLACE VIEW StaffEventView AS

SELECT

S.staffID AS StaffID,

S.S\_name AS StaffName,

S.S\_DOB AS StaffDOB,

S.S\_cnic AS StaffCNIC,

S.S\_address AS StaffAddress,

S.S\_type AS StaffType,

E.eventID AS EventID,

E.E\_type AS EventType,

E.E\_date AS EventDate,

E.E\_time AS EventTime,

E.E\_status AS EventStatus,

C.CAT\_CATEGORY AS C\_CATEGORY,

D.D\_theme AS D\_THEME,

D.D\_floralArrangements AS D\_FLORALS,

D.D\_sittingArrangements AS D\_SITTINGS,

D.D\_lightinigSetups AS D\_LIGHNING,

SE.sec\_weapons1 AS W1,

SE.sec\_weapons2 AS W2,

T.TC\_ROLE AS T\_ROLE

FROM

Staff S

JOIN Event E ON S.EVENTID = E.EVENTID

LEFT JOIN Catering C ON S.STAFFID = C.STAFFID

AND S.S\_TYPE='C'

LEFT JOIN Decor D ON S.STAFFID = D.STAFFID

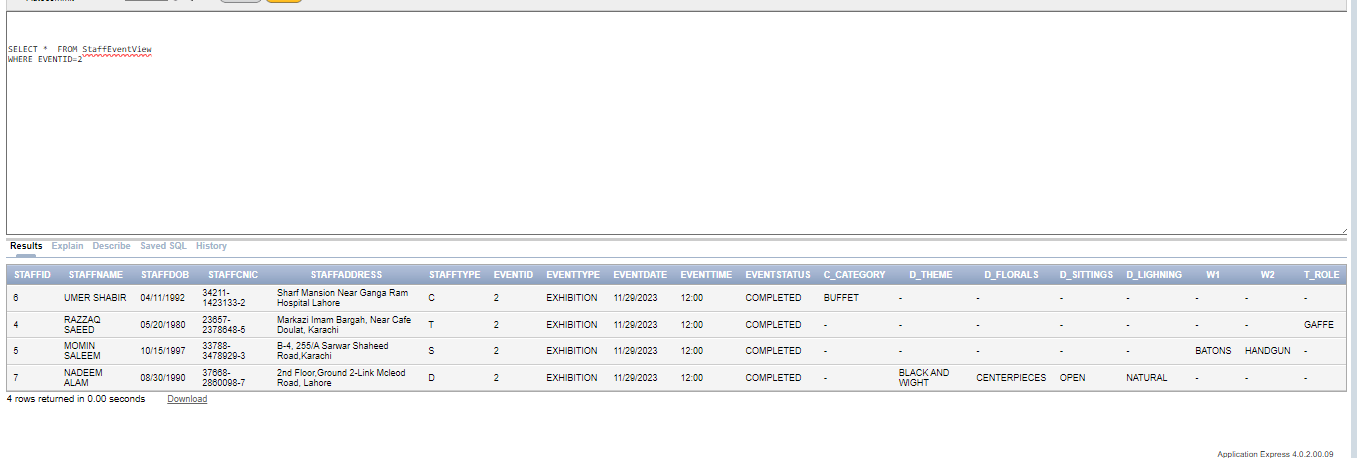
AND S.S\_TYPE='D'

LEFT JOIN Technical\_crew T ON S.STAFFID = T.STAFFID

AND S.S\_TYPE='T'

LEFT JOIN Security SE ON S.STAFFID = SE.STAFFID

AND S.S\_TYPE='S'



1. **CREATE A VIEW FOR RETRIEVING UPCOMING EVNETS WITH SOME DETAIL**

CREATE OR REPLACE VIEW UpcomingEventsView AS

SELECT

E.eventID AS EventID,

E.E\_type AS EventType,

E.E\_date AS EventDate,

E.E\_time AS EventTime,

E.E\_status AS EventStatus,

O.ORGID AS "ORGANIZER ID",

O.ORG\_NAME AS "ORGANIZER NAME",

V.venueID AS VenueID,

V.V\_NAME AS "VENUE NAME",

C.customerID AS CustomerID,

C.C\_NAME AS "CUSTOMER NAME"

FROM

Event E

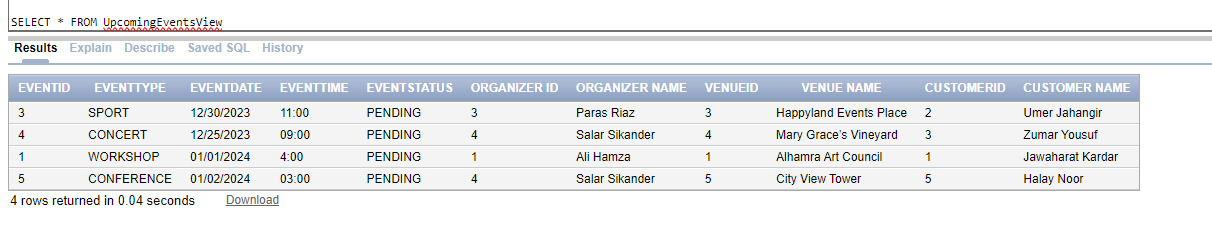
LEFT JOIN ORGANIZER O ON O.ORGID=E.ORGID

LEFT JOIN CUSTOMER C ON C.CUSTOMERID=E.CUSTOMERID

LEFT JOIN VENUE V ON V.VENUEID=E.EVENTID

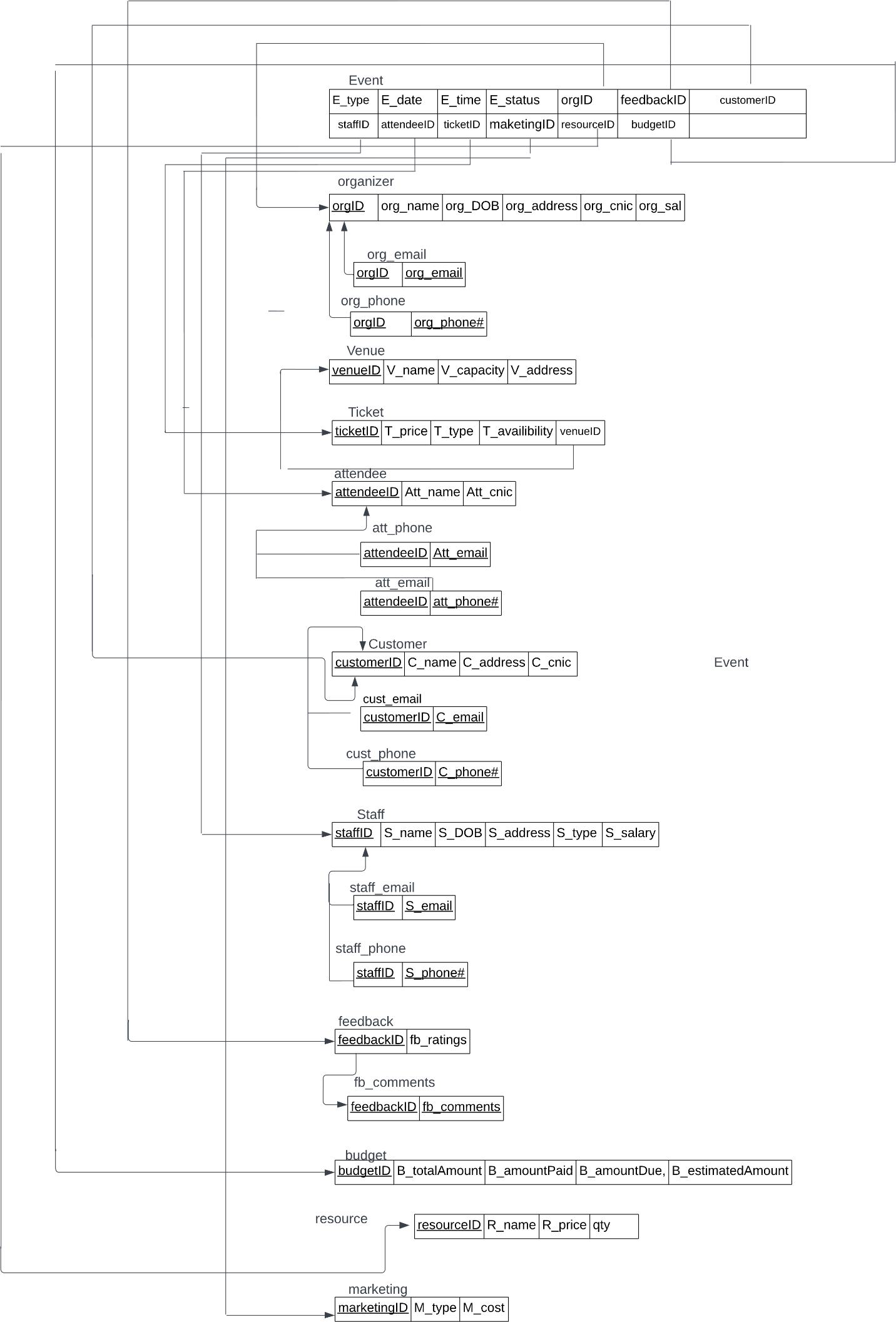
WHERE

E\_date > SYSDATE;



---------------------------------------------------------------------------------------

***Relational Data Model showing association***



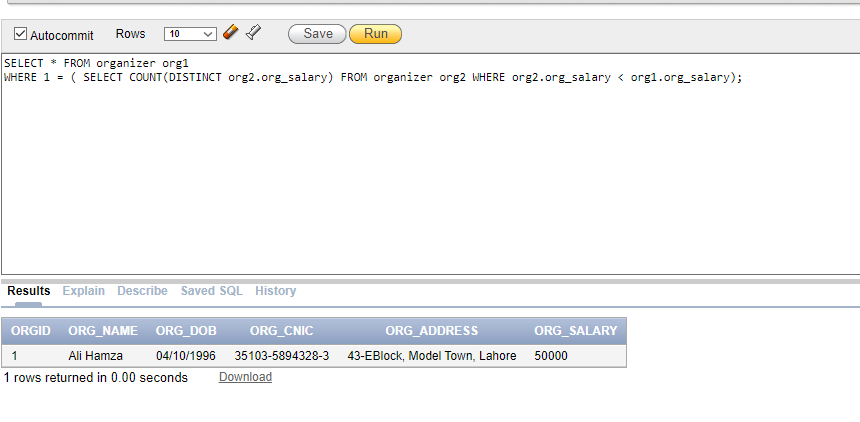
***Reports***

1. **2nd minimum salary of an organizer**

SELECT \* FROM organizer org1

WHERE 1 = ( SELECT COUNT(DISTINCT org2.org\_salary) FROM organizer org2

WHERE org2.org\_salary < org1.org\_salary);



1. **Retrieve marketing approach used in event**

SELECT E.eventID, E.E\_type,E.E\_Date, E.E\_time, E.E\_status, M.M\_type, M.M\_cost,

CASE

WHEN M.M\_type = 'D' THEN

'Social Media: ' || D.DM\_socialMedia || ', Email: ' || D.DM\_email || ', SEO: ‘|| D.DM\_SEO || ', Strategy: ' || D.DM\_strategy

WHEN M.M\_type = 'T' THEN

'Material: ' || T.TM\_material || ', Media Outlet: ' || T.TM\_mediaOutlet || ', Print Production: ' || T.TM\_printProduction

END AS MarketingDetail

FROM

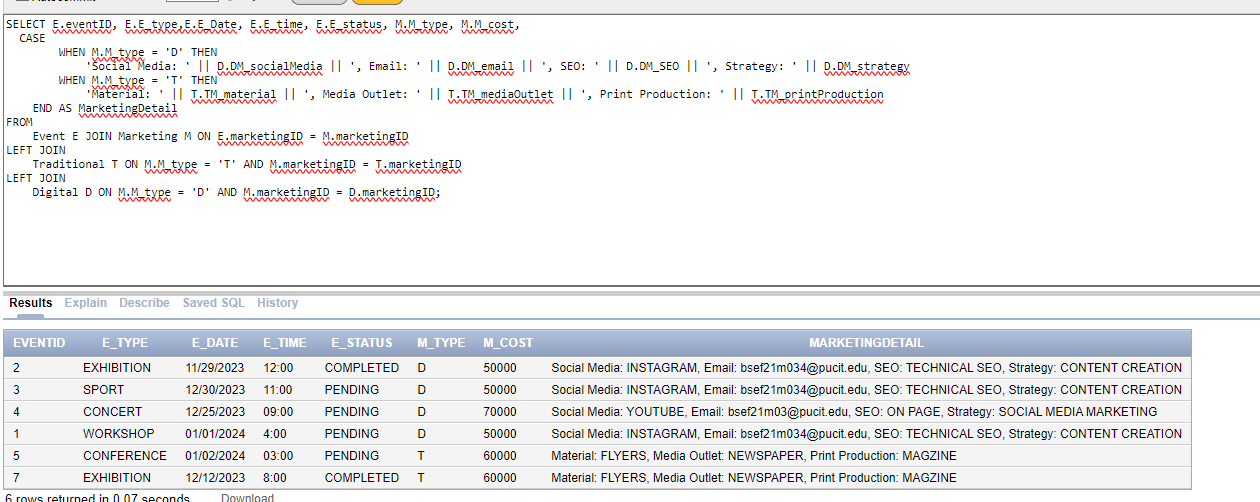
Event E JOIN Marketing M ON E.marketingID = M.marketingID

LEFT JOIN

Traditional T ON M.M\_type = 'T' AND M.marketingID = T.marketingID

LEFT JOIN

Digital D ON M.M\_type = 'D' AND M.marketingID = D.marketingID;

****

1. **Tickets used / tickets left for a venue**

SELECT e.eventID, e.E\_type, v.V\_name, v.V\_capacity AS TotalTickets,

COUNT(t.ticketID) AS TicketsUsed,

v.V\_capacity - COUNT(t.ticketID) AS TicketsLeft

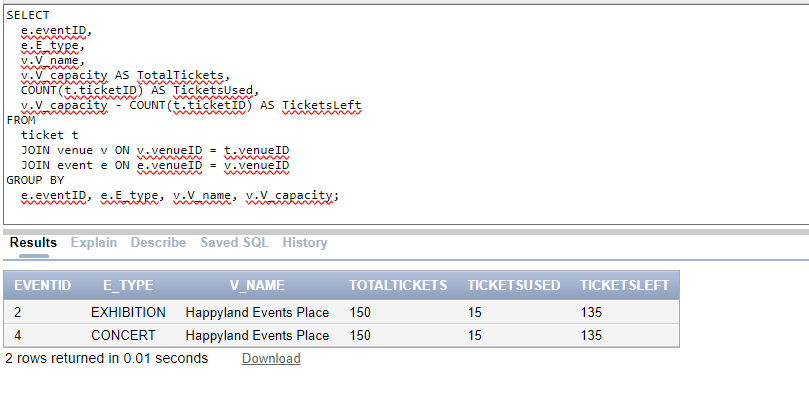
FROM ticket t

JOIN venue v ON v.venueID = t.venueID

JOIN event e ON e.venueID = v.venueID

GROUP BY

e.eventID, e.E\_type, v.V\_name, v.V\_capacity;



1. **Find in which event there are Maximum number of attendees**

SELECT e.eventID, e.E\_type,

e.E\_date, e.E\_time, e.E\_status, (SELECT MAX(attendee\_count) FROM (SELECT eventID, COUNT(attendeeID) AS attendee\_count FROM

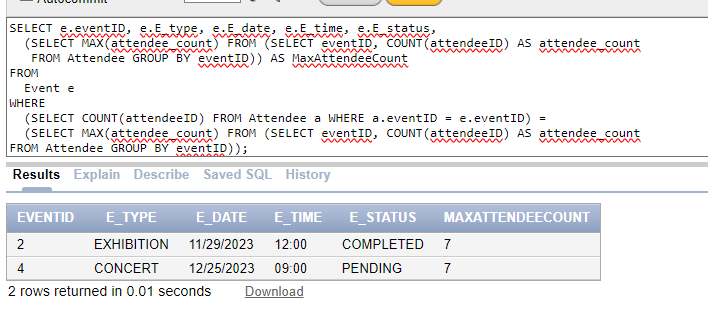
Attendee GROUP BY eventID)) AS MaxAttendeeCount

FROM Event e

WHERE

(SELECT COUNT(attendeeID) FROM Attendee a WHERE a.eventID = e.eventID) =

(SELECT MAX(attendee\_count) FROM (SELECT eventID, COUNT(attendeeID) AS attendee\_count FROM Attendee GROUP BY eventID));



1. **Sort the events with respect to their feedback ratings in decreasing order**

SELECT e.eventID, e.E\_type, nvl(AVG(f.fb\_rating),0) AS Ratings

FROM

Event e

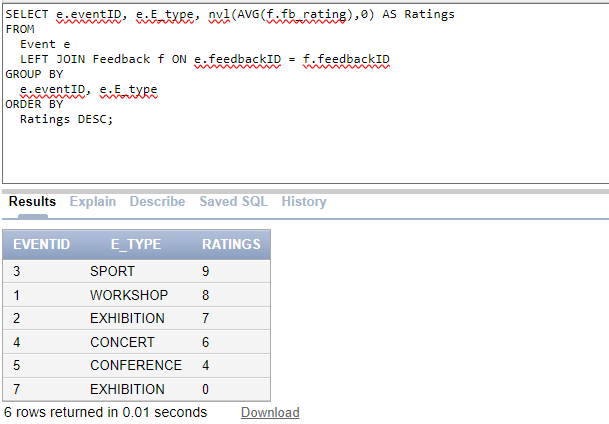
LEFT JOIN Feedback f ON e.feedbackID = f.feedbackID

GROUP BY

e.eventID, e.E\_type

ORDER BY

Ratings DESC;

****

***Triggers, Procedures, Functions***

***TRIGGERS***

1. **TRIGGER BEFORE INSERTION ON EVENT TO VALIDATE THAT IF DATE AND TIME AND STATUS ARE VALID WITH EACH OTHER T**

CREATE OR REPLACE TRIGGER EventStatusTrigger

BEFORE INSERT OR UPDATE ON Event

FOR EACH ROW

BEGIN

IF ((:NEW.E\_date >= SYSDATE) AND (TO\_TIMESTAMP(:NEW.E\_TIME,'HH12:MI') < CURRENT\_TIMESTAMP)) AND (:NEW.E\_STATUS = 'COMPLETED') THEN

RAISE\_APPLICATION\_ERROR(-20001, 'EVENT STATUS IS INVALID! Event can not be completed before the held date.');

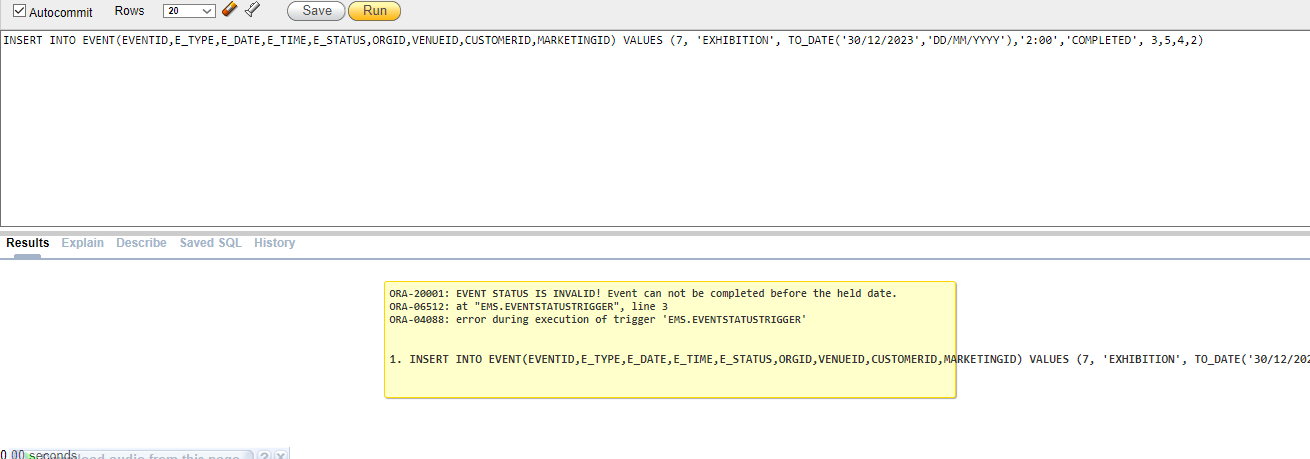
ELSIF((:NEW.E\_date <= SYSDATE) AND (TO\_TIMESTAMP(:NEW.E\_TIME,'HH12:MI') <= CURRENT\_TIMESTAMP)) AND (:NEW.E\_STATUS = 'PENDING')

THEN

RAISE\_APPLICATION\_ERROR(-20002, 'EVENT STATUS IS INVALID! Event can not be PENDING AFTER the held date.');

END IF;

END;



1. **Trigger before insertion and updation on RES for calculating and setting total amount and amount due.**

CREATE OR REPLACE TRIGGER UpdateBudgetTotalTrigger

BEFORE INSERT OR UPDATE ON Res

FOR EACH ROW

DECLARE

v\_totalAmount NUMBER(10, 2);

M\_SUM NUMBER(10, 2);

v\_Due NUMBER(10, 2);

BEGIN

v\_totalAmount := :NEW.R\_qty \* :NEW.R\_price;

IF B\_totalAmount IS NULL THEN

SELECT v\_totalAmount + M.M\_cost

INTO M\_SUM

FROM Marketing M

WHERE M.marketingID = (SELECT marketingID FROM Event WHERE eventID = :NEW.eventID);

ELSE

M\_SUM := B\_totalAmount;

END IF;

UPDATE Budget

SET B\_totalAmount = NVL(B\_totalAmount, 0) + M\_SUM

WHERE eventID = :NEW.eventID;

-- Calculate amount due

SELECT (B\_TOTALAMOUNT - B\_amountPaid) INTO v\_Due

FROM Budget

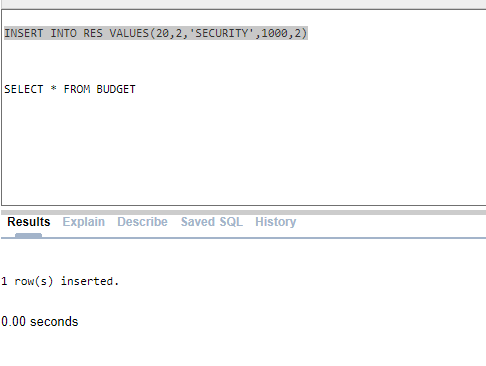
WHERE eventID = :NEW.eventID;

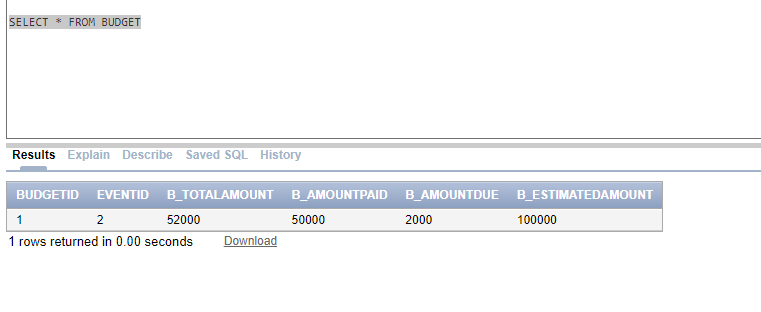
UPDATE Budget

SET B\_amountDue = v\_Due

WHERE eventID = :NEW.eventID;

END;





***Procedures:***

1. **Find staff and display details from its CNIC**

CREATE OR REPLACE PROCEDURE FIND\_STAFF(input VARCHAR2)

IS

CURSOR CURS\_STAFF IS SELECT \* FROM ST WHERE S\_CNIC = INPUT;

DETAIL ST%ROWTYPE;

v\_StaffCount NUMBER := 0;

BEGIN

OPEN CURS\_STAFF;

LOOP

FETCH CURS\_STAFF INTO DETAIL;

EXIT WHEN CURS\_STAFF%NOTFOUND;

END LOOP;

v\_StaffCount := CURS\_STAFF%ROWCOUNT;

IF v\_StaffCount > 0 THEN

DBMS\_OUTPUT.PUT\_LINE('STAFF ID IS: ' || DETAIL.STAFFID);

DBMS\_OUTPUT.PUT\_LINE('EVENT ID IS: ' || DETAIL.EVENTID);

DBMS\_OUTPUT.PUT\_LINE('STAFF NAME IS: ' || DETAIL.S\_NAME);

DBMS\_OUTPUT.PUT\_LINE('STAFF DOB IS: ' || DETAIL.S\_DOB);

DBMS\_OUTPUT.PUT\_LINE('STAFF CNIC IS: ' || DETAIL.S\_CNIC);

DBMS\_OUTPUT.PUT\_LINE('STAFF ADDRESS IS: ' || DETAIL.S\_ADDRESS);

DBMS\_OUTPUT.PUT\_LINE('STAFF TYPE IS: ' || DETAIL.S\_TYPE);

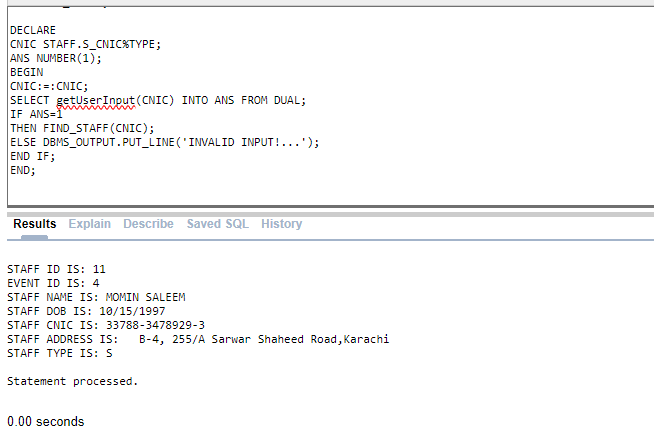
ELSE

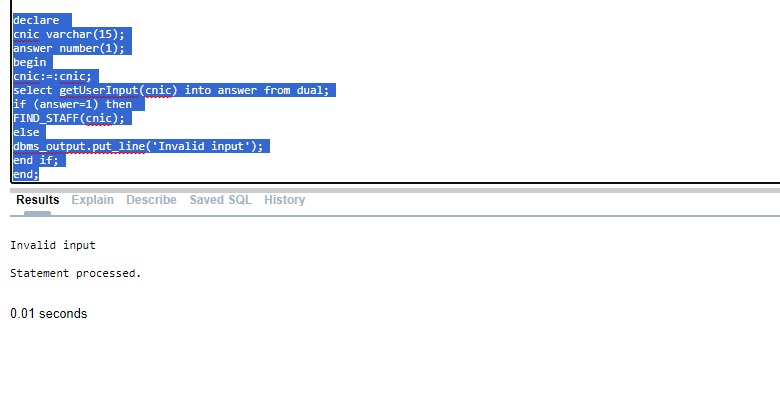
DBMS\_OUTPUT.PUT\_LINE('NO STAFF FOUND FOR GIVEN CNIC');

END IF;

CLOSE CURS\_STAFF;

END FIND\_STAFF;





1. **Procedure for increase in organizer’s salary on basis of range**

CREATE OR REPLACE PROCEDURE sal\_raise(sal IN out NUMBER)

AS

s NUMBER := 0;

BEGIN

IF sal between 20000 and 50000 THEN

s := sal \* 0.05;

IF s > 15000 THEN

s := 15000;

END IF;

ELSIF sal between 50001 and 80000 THEN

s := sal \* 0.10;

IF s > 15000 THEN

s := 15000;

END IF;

ELSIF sal between 80001 and 120000 then

s := sal \* 0.15;

IF s > 15000 THEN

s := 15000;

END IF;

END IF;

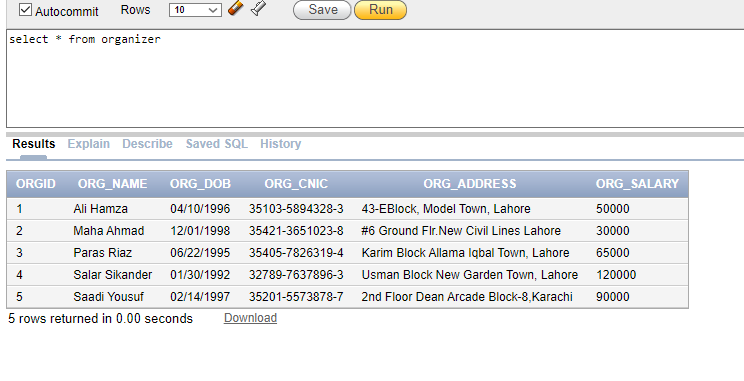
sal:=sal+s;

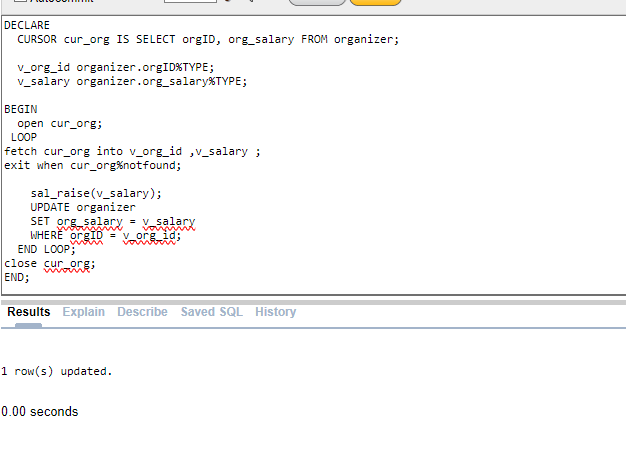
EXCEPTION

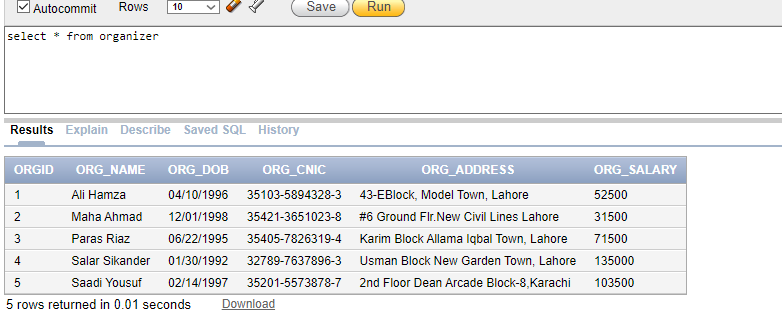
WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('NO DATA FOUND');

END sal\_raise;







***Functions:***

1. **Validate CNIC of either customer,staff or organizer**

create or replace function getUserInput(cnic varchar2)

return number is

c char(1);

begin

if (length(cnic) != 15) then

return 0;

end if;

for i in 1..15 loop

c := substr(cnic, i, 1);

if (c >= lower('a') and c <= lower('z')) then

return 0;

elsif (i = 6 OR i = 14) then

if (c != '-') then

return 0;

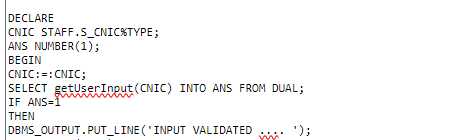
end if;

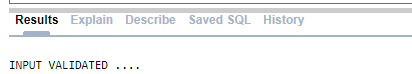
end if;

end loop;

return 1;

end;





1. **CHECKS WHETHER A CERTAIN EVENT HAS REGISTERED ATTENDEES OR NOT**

CREATE OR REPLACE FUNCTION HasRegisteredAttendees(ID IN NUMBER)

RETURN BOOLEAN

AS

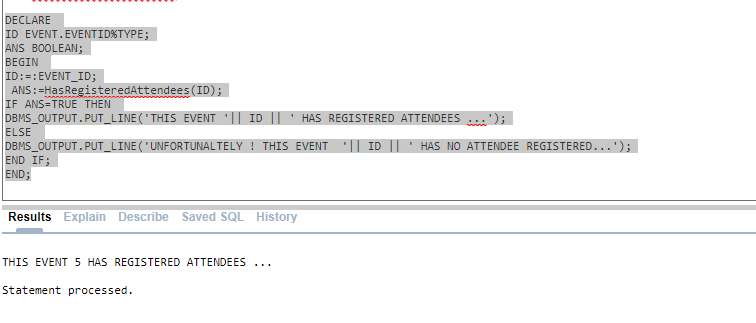
attendeeCount NUMBER;

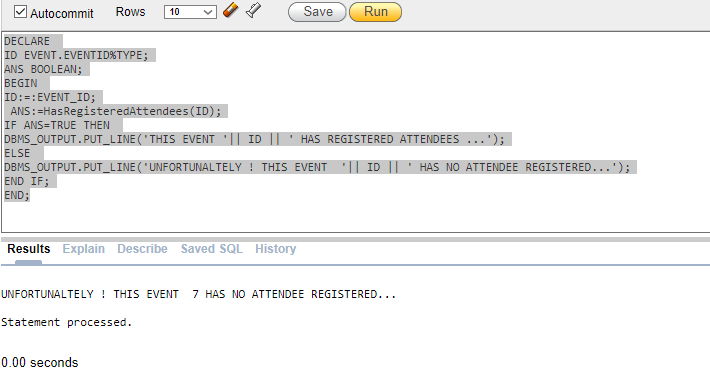
BEGIN

SELECT COUNT(\*) INTO attendeeCount FROM AttendeE WHERE EventID = ID;

return attendeeCount > 0;

END HasRegisteredAttendees;

****

****

-----------------------------------------------------------------------------------------------------------------------------