University of British Columbia, Vancouver

Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: October 20, 2023

Group Number: 118

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Caitlin Wu	93509883	o0j3b	caitlinwu13@gmail.com
Linh Trinh	57843690	s2y7q	tklinh202@gmail.com
Shahrukh Islam	47610068	l5l9f	s.prithibi15@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

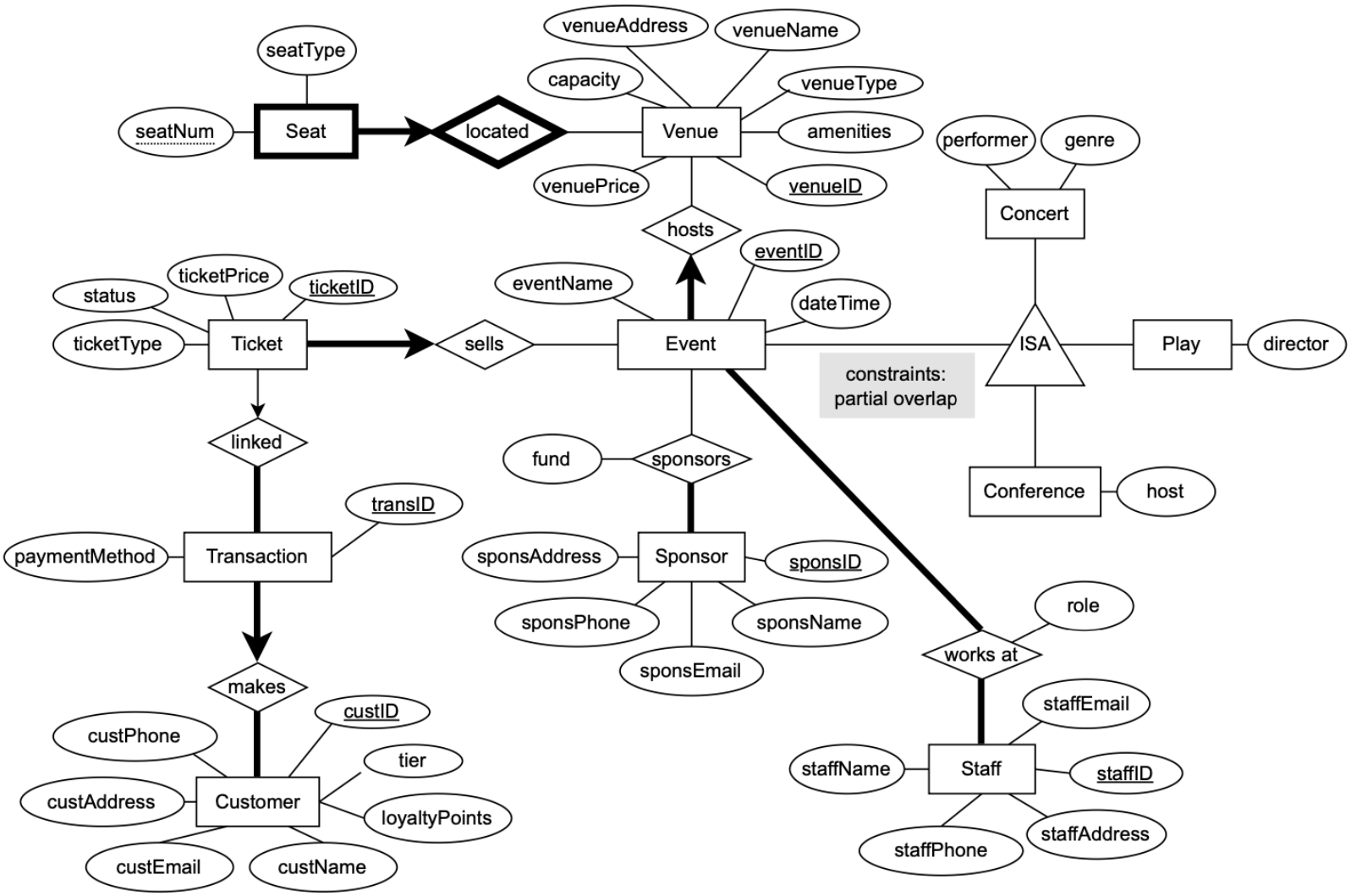
In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

Event Company Organization

Project summary:

The Event Management database supports event planning, sponsorship management, attendee engagement, and financial transactions. All elements of the database collaborate to create a holistic ecosystem that caters to the diverse needs of event organizers, staff, and attendees across various event types and scales. The system is designed to streamline event operations, enhance external partnerships, and provide attendees with an enjoyable event experience.

ER Diagram:



Compared to the diagram in milestone 1, the duplicate field names of each entity have been changed to ‘entity+field’ based on the feedback. The ‘role’ attribute of **Staff** entity has been moved to ‘works at’ relationship as the roles of staff members can change depending on which event they work at.

For additional functional dependencies, the following attributes have been added:

- Attributes ‘venueType’, ‘venuePrice’, and ‘amenities’ in **Venue** entity.
- Attribute ‘status’ (i.e., available/sold) in **Ticket** entity.
- Attribute ‘loyaltyPoints’ and ‘tier’ in **Customer** entity.

Relational Schema:

1. **Venue**(venueID: integer, venueName: varchar, venueAddress: varchar, capacity: integer, venueType: varchar, venuePrice: integer, amenities: varchar)
    a. PK: venueID
    b. CK: venueAddress
    c. Constraints:
        i. venueAddress must be unique and cannot be null
        ii. Capacity cannot be null
2. **Seat**(venueID: integer, seatNum: varchar, seatType: varchar)
    a. PK: {venueID, seatNum}
    b. FK: venueID references Venue
3. **Event**(eventID: integer, eventName: varchar, dateTime: datetime, venueID: integer)
    a. PK: eventID
    b. FK: venueID references Venue
    c. Constraints:
        i. venueID cannot be null
        ii. dateTime cannot be null
4. **Concert**(eventID: integer, performer: varchar, genre: varchar)
    a. PK: eventID
    b. FK: eventID references Event
5. **Play**(eventID: integer, director: varchar)
    a. PK: eventID
    b. FK: eventID references Event
6. **Conference**(eventID: integer, host: varchar)
    a. PK: eventID
    b. FK: eventID references Event
7. **Ticket**(ticketID: integer, ticketType: varchar, ticketPrice: integer, eventID: integer, transID: integer, status: varchar)
    a. PK: ticketID
    b. FK:
        i. eventID references Event
        ii. transID references Transaction
    c. Constraints: eventID cannot be null
8. **Transaction**(transID: integer, paymentMethod: varchar, custID: integer)
    a. PK: transID
    b. FK: custID references Customer
    c. Constraints: custID cannot be null
9. **Customer**(custID: integer, custName: varchar, custAddress: varchar, custPhone: varchar, custEmail: varchar, loyaltyPoints: integer, tier: varchar)

    a. PK: custID
    b. CK: custEmail
    c. Constraints:
        i. custEmail must be unique and cannot be null
10. **Sponsor**(sponsID: integer, sponsName: varchar, sponsAddress: varchar, sponsPhone: varchar, sponsEmail: varchar)
    a. PK: sponsID
    b. CK: sponsEmail
    c. Constraints:
        i. sponsEmail must be unique and cannot be null
11. **Sponsorship**(eventID: integer, sponsID: integer, fund: integer)
    a. PK: {eventID, sponsID}
    b. FK:
        i. eventID references Event
        ii. sponsID references Sponsor
12. **Staff**(staffID: integer, staffName: varchar, staffAddress: varchar, staffPhone: varchar, staffEmail: varchar)
    a. PK: staffID
    b. CK: staffEmail
    c. Constraints:
        i. staffEmail must be unique and cannot be null
13. **Assignment**(eventID: integer, staffID:integer, role: varchar)
    a. PK: {eventID, staffID}
    b. FK:
        i. eventID references Event
        ii. staffID references Staff

Functional Dependencies:

1. Venue(venueID, venueName, venueAddress, capacity, venueType, venuePrice, amenities)
    a. venueID → venueName, venueAddress, capacity, venueType, venuePrice, amenities
    b. venueAddress → venueID
    c. venueType, capacity → venuePrice
    d. venueType → amenities
2. Seat(**venueID**, seatNum, seatType)
    a. venueID, seatNum → seatType
3. Event(eventID, eventName, dateTime, **venueID**)
    a. eventID → eventName, dateTime, venueID
    b. venueID, dateTime → eventName
4. Concert(**eventID**, performer, genre)
    a. eventID → performer, genre
    b. performer → genre
5. Play(**eventID**, director)

    a. eventID → director
6. Conference(**eventID**, host)
    a. eventID → host
7. Ticket(ticketID, ticketType, ticketPrice, **eventID**, **transID**, status)
    a. ticketID → ticketType, ticketPrice, eventID, transID, status
    b. ticketType, eventID → ticketPrice
    c. transID → status
8. Transaction(transID, paymentMethod, **custID**)
    a. transID → paymentMethod, custID
9. Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints, tier)
    a. custID → custName, custAddress, custPhone, custEmail, loyaltyPoints, tier
    b. custEmail → custID
    c. loyaltyPoints → tier
10. Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail)
    a. sponsID → sponsName, sponsAddress, sponsPhone, sponsEmail
    b. sponsEmail → sponsID
11. Sponsorship(**eventID**, **sponsorID**, fund)
    a. eventID, sponsID → fund
12. Staff(staffID, staffName, staffAddress, staffPhone, staffEmail)
    a. staffID → staffName, staffAddress, staffPhone, staffEmail
    b. staffEmail → staffID
13. Assignment(**eventID**, **staffID**, role)
    a. eventID, staffID → role

Normalization (BCNF):

1. **Venue(venueID, venueName, venueAddress, capacity, venueType, venuePrice, amenities)**
    a. FDs:
        i. venueID → venueName, venueAddress, capacity, venueType, venuePrice, amenities
        ii. venueAddress → venueID
        iii. venueType, capacity → venuePrice
        iv. venueType → amenities
    b. Decomposition:
        i. Venue(venueID, venueName, venueAddress, capacity, venueType, venuePrice, amenities)
        ii. Decompose Venue into:
            1. Rent(venueType, capacity, venuePrice)
            2. Venue(venueID, venueName, venueAddress, capacity, venueType, amenities)
        iii. Decompose Venue into:
            1. Facility(venueType, amenities)
            2. Venue(venueID, venueName, venueAddress, capacity, venueType)

c. Normalized Tables:
    i. Rent(**venueType**, capacity, venuePrice)
    ii. Facility(venueType, amenities)
    iii. Venue(venueID, venueName, venueAddress, **capacity**, **venueType**)
    (venueAddress must be unique and cannot be null)

2. Seat(venueID, seatNum, seatType)

a. FDs:
    i. venueID, seatNum → seatType
b. Decomposition:
    i. Seat(venueID, number, seatType) is already in BCNF.
c. Normalized Tables:
    i. Seat(**venueID**, seatNum, seatType)

3. Event(eventID, eventName, dateTime, venueID)

a. FDs:
    i. eventID → eventName, dateTime, venueID
    ii. venueID, dateTime → eventName
b. Decomposition:
    i. Event(eventID, eventName, dateTime, venueID)
    ii. Decompose Event into:
        1. EventName(venueID, dateTime, eventName)
        2. Event(eventID, venueID, dateTime)
c. Normalized Tables:
    i. EventName(**venueID**, dateTime, eventName)
    ii. Event(eventID, **venueID**, **dateTime**)

4. Concert(eventID, performer, genre)

a. FDs:
    i. eventID → performer, genre
    ii. performer → genre
b. Decomposition:
    i. Concert(eventID, performer, genre)
    ii. Decompose Concert into:
        1. Genre(performer, genre)
        2. Concert(eventID, performer)
c. Normalized Tables:
    i. Genre(performer, genre)
    ii. Concert(**eventID**, **performer**)

5. Play(eventID, director)

a. FDs:
    i. eventID → director
b. Decomposition:

i. Play(eventID, director) is already in BCNF.

c. Normalized Tables:

i. Play(**<u>eventID</u>**, director)

6. **Conference(eventID, host)**
    a. FDs:
        i. eventID → host
    b. Decomposition:
        i. Conference(eventID, host) is already in BCNF.
    c. Normalized Tables:
        i. Conference(**<u>eventID</u>**, host)

7. **Ticket(ticketID, ticketType, ticketPrice, eventID, transID, status)**
    a. FDs:
        i. ticketID → ticketType, ticketPrice, eventID, transID
        ii. ticketType, eventID → ticketPrice
        iii. transID → status
    b. Decomposition:
        i. Ticket(ticketID, ticketType, ticketPrice, eventID, transID, status)
        ii. Decompose Ticket into:
            1. TicketPrice(ticketType, eventID, ticketPrice)
            2. Ticket(ticketID, ticketType, eventID, transID, status)
        iii. Decompose Ticket into:
            1. TicketStatus(transID, status)
            2. Ticket(ticketID, ticketType, eventID, transID)
    c. Normalized Tables:
        i. TicketPrice(<u>ticketType</u>, **<u>eventID</u>**, ticketPrice)
        ii. TicketStatus(**<u>transID</u>**, status)
        iii. Ticket(<u>ticketID</u>, **<u>ticketType</u>**, **<u>eventID</u>**, **<u>transID</u>**)

8. **Transaction(transID, paymentMethod, custID)**
    a. FDs:
        i. transID → paymentMethod, custID
    b. Decomposition:
        i. Transaction(transID, paymentMethod, custID) is already in BCNF.
    c. Normalized Tables:
        i. Transaction(<u>transID</u>, paymentMethod, **custID**) (custID cannot be null)

9. **Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints, tier)**
    a. FDs:
        i. custID → custName, custAddress, custPhone, custEmail
        ii. custEmail → custID
        iii. loyaltyPoints → tier

b. Decomposition:
    i. Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints, tier)
    ii. Decompose Customer into:
        1. CustomerTier(loyaltyPoints, tier)
        2. Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints)
c. Normalized Tables:
    i. CustomerTier(<u>loyaltyPoints</u>, tier)
    ii. Customer(<u>custID</u>, custName, custAddress, custPhone, custEmail, **<u>loyaltyPoints</u>**) (custEmail must be unique and cannot be null)

10. Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail)

a. FDs:
    i. sponsID → sponsName, sponsAddress, sponsPhone, sponsEmail
    ii. sponsEmail → sponsID
b. Decomposition:
    i. Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail) is already in BCNF.
c. Normalized Tables:
    i. Sponsor(<u>sponsID</u>, sponsName, sponsAddress, sponsPhone, sponsEmail) (sponsEmail must be unique and cannot be null)

11. Sponsorship(eventID, sponsorID, fund)

a. FDs:
    i. eventID, sponsID → fund
b. Decomposition:
    i. Sponsorship(eventID, sponsorID, fund) is already in BCNF.
c. Normalized Tables:
    i. Sponsorship(**<u>eventID</u>**, **<u>sponsorID</u>**, fund)

12. Staff(staffID, staffName, staffAddress, staffPhone, staffEmail)

a. FDs:
    i. staffID → staffName, staffAddress, staffPhone, staffEmail
    ii. staffEmail → staffID
b. Decomposition:
    i. Staff(staffID, staffName, staffAddress, staffPhone, staffEmail) is already in BCNF.
c. Normalized Tables:
    i. Staff(<u>staffID</u>, staffName, staffAddress, staffPhone, staffEmail) (staffEmail must be unique and cannot be null)

13. Assignment(eventID, staffID, role)

a. FDs:

i. eventID, staffID → role

b. Decomposition:
   i. Assignment(eventID, staffID, role) is already in BCNF.

c. Normalized Tables:
   i. Assignment(**eventID**, **staffID**, role)

SQL DDL Statements:

1. Rent(**venueType**, capacity, venuePrice):

```
CREATE TABLE Rent(
        venueType VARCHAR,
        capacity INTEGER NOT NULL,
        venuePrice INTEGER,
        PRIMARY KEY (venueType, capacity),
        FOREIGN KEY (venueType) REFERENCES Facility ON DELETE CASCADE
);
```

2. Facility(venueType, amenities):

```
CREATE TABLE Facility (
        venueType VARCHAR PRIMARY KEY,
        amenities VARCHAR
);
```

3. Venue(venueID, venueName, venueAddress, **capacity**, **venueType**) (venueAddress must be unique and cannot be null):

```
CREATE TABLE Venue (
        venueID INTEGER,
        venueName VARCHAR,
        venueAddress VARCHAR NOT NULL UNIQUE,
        capacity INTEGER,
        venueType VARCHAR,
        PRIMARY KEY (venueID, capacity, venueType),
        FOREIGN KEY (venueType, capacity) REFERENCES Rent(venueType,
capacity) ON DELETE CASCADE
);
```

4. Seat(**venueID**, seatNum, seatType):

```
CREATE TABLE Seat (
        venueID INTEGER,
        seatNum VARCHAR,
        seatType VARCHAR,
        PRIMARY KEY (venueID, seatNum),
        FOREIGN KEY (venueID) REFERENCES Venue ON DELETE CASCADE
);
```

5. EventName(**venueID**, dateTime, eventName):
   CREATE TABLE EventName (
   venueID INTEGER,
   dateTime DATETIME,
   eventName VARCHAR,
   PRIMARY KEY (venueID, dateTime),
   FOREIGN KEY (venueID) REFERENCES Venue ON DELETE CASCADE
   );

6. Event(eventID, **venueID**, **dateTime**):
   CREATE TABLE Event (
   eventID INTEGER,
   venueID INTEGER,
   dateTime DATETIME NOT NULL,
   PRIMARY KEY (eventID, venueID, dateTime),
   FOREIGN KEY (venueID, dateTime) REFERENCES EventName(venueID, dateTime) ON DELETE CASCADE
   );

7. Genre(performer, genre):
   CREATE TABLE Genre (
   performer VARCHAR PRIMARY KEY,
   genre VARCHAR
   );

8. Concert(**eventID**, **performer**):
   CREATE TABLE Concert (
   eventID INTEGER,
   performer VARCHAR,
   PRIMARY KEY (eventID, performer),
   FOREIGN KEY (eventID) REFERENCES Event ON DELETE CASCADE,
   FOREIGN KEY (performer) REFERENCES Genre ON DELETE CASCADE
   );

9. Play(**eventID**, director):
   CREATE TABLE Play (
   eventID INTEGER PRIMARY KEY,
   director VARCHAR,
   FOREIGN KEY (eventID) REFERENCES Event ON DELETE CASCADE
   );

10. Conference(**eventID**, host):
    CREATE TABLE Conference (
    eventID INTEGER PRIMARY KEY,

host VARCHAR,
FOREIGN KEY (eventID) REFERENCES Event ON DELETE CASCADE
);

11. TicketPrice(ticketType, **eventID**, ticketPrice):
CREATE TABLE TicketPrice (
ticketType VARCHAR,
eventID INTEGER,
ticketPrice INTEGER,
PRIMARY KEY (ticketType, eventID),
FOREIGN KEY (eventID) REFERENCES Event ON DELETE CASCADE
);

12. TicketStatus(**transID**, status):
CREATE TABLE TicketStatus (
transID INTEGER PRIMARY KEY,
status VARCHAR,
FOREIGN KEY (transID) REFERENCES Transaction ON DELETE CASCADE
);

13. Ticket(ticketID, **ticketType**, **eventID**, **transID**):
CREATE TABLE Ticket (
ticketID INTEGER,
ticketType VARCHAR,
eventID INTEGER,
transID INTEGER,
PRIMARY KEY (ticketID, ticketType, eventID, transID),
FOREIGN KEY (ticketType, eventID) REFERENCES TicketPrice(ticketType, eventID) ON DELETE CASCADE,
FOREIGN KEY (transID) REFERENCES Transaction ON DELETE CASCADE
);

14. Transaction(transID, paymentMethod, **custID**) (custID cannot be null):
CREATE TABLE Transaction (
transID INTEGER PRIMARY KEY,
paymentMethod VARCHAR,
custID INTEGER NOT NULL,
FOREIGN KEY (custID) REFERENCES Customer ON DELETE CASCADE
);

15. CustomerTier(loyaltyPoints, tier):
CREATE TABLE CustomerTier (
loyaltyPoints INTEGER PRIMARY KEY,
tier VARCHAR

);

16. Customer(<u>custID</u>, custName, custAddress, custPhone, custEmail, **<u>loyaltyPoints</u>**) (custEmail must be unique and cannot be null):
    CREATE TABLE Customer (
        custID INTEGER,
        custName VARCHAR,
        custAddress VARCHAR,
        custPhone VARCHAR,
        custEmail VARCHAR NOT NULL UNIQUE,
        loyaltyPoints INTEGER,
        PRIMARY KEY (custID, loyaltyPoints),
        FOREIGN KEY (loyaltyPoints) REFERENCES (CustomerTier) ON DELETE CASCADE
    );

17. Sponsor(<u>sponsID</u>, sponsName, sponsAddress, sponsPhone, sponsEmail) (sponsEmail must be unique and cannot be null):
    CREATE TABLE Sponsor (
        sponsID INTEGER PRIMARY KEY,
        sponsName VARCHAR,
        sponsAddress VARCHAR,
        sponsPhone VARCHAR,
        sponsEmail VARCHAR NOT NULL UNIQUE,
    );

18. Sponsorship(**<u>eventID</u>**, **<u>sponsorID</u>**, fund):
    CREATE TABLE Sponsorship (
        eventID INTEGER,
        sponsorID INTEGER,
        fund INTEGER,
        PRIMARY KEY (eventID, sponsorID),
        FOREIGN KEY (eventID) REFERENCES Event ON DELETE CASCADE,
        FOREIGN KEY (sponsorID) REFERENCES Sponsor ON DELETE CASCADE
    );

19. Staff(<u>staffID</u>, staffName, staffAddress, staffPhone, staffEmail) (staffEmail must be unique and cannot be null):
    CREATE TABLE Staff (
        staffID INTEGER PRIMARY KEY,
        staffName VARCHAR,
        staffAddress VARCHAR,
        staffPhone VARCHAR,
        staffEmail VARCHAR NOT NULL UNIQUE,

);

20. Assignment(**eventID**, **staffID**, role):
    CREATE TABLE Assignment (
        eventID INTEGER,
        staffID INTEGER,
        role VARCHAR,
        PRIMARY KEY (eventID, staffID),
        FOREIGN KEY (eventID) REFERENCES Event ON DELETE CASCADE,
        FOREIGN KEY (staffID) REFERENCES Staff ON DELETE CASCADE
    );

INSERT Statements:

1. Rent(**venueType**, capacity, venuePrice):
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Conference Room’, 50, 200);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Conference Room’, 200, 500);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Banquet Hall’, 100, 500);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Banquet Hall’, 500, 1000);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Theatre’, 500, 2000);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Theatre’, 1000, 5000);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Arena’, 10000, 5000);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Arena’, 30000, 20000);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Stadium’, 30000, 30000);
   INSERT INTO Rent(venueType, capacity, venuePrice) VALUES
   (‘Stadium’, 50000, 50000;

2. Facility(venueType, amenities):
   INSERT INTO Facility(venueType, amenities) VALUES
   (‘Conference Room’, ’Audiovisual, Wi-Fi, Seating, Whiteboard, Air-conditioning’);
   INSERT INTO Facility(venueType, amenities) VALUES
   (‘Banquet Hall’, ’Audiovisual, Wi-Fi, Seating, Stage, Air-conditioning’);
   INSERT INTO Facility(venueType, amenities) VALUES
   (‘Theatre’, ’Seating, Sound, Screen, Concessions, Accessibility');
   INSERT INTO Facility(venueType, amenities) VALUES
   (‘Arena’, ’Seating, Sound, Lighting, Concessions, Lockers, First Aid Station’);

INSERT INTO Facility(venueType, amenities) VALUES
(‘Stadium’, ’Seating, Sound, Lighting, Concessions, VIP Suites, Sports Bars’);

3. Venue(venueID, venueName, venueAddress, **capacity**, **venueType**):
   INSERT INTO Venue(venueID, venueName, venueAddress, capacity, venueType)
   VALUES (1, 'City Arena', '123 Main Street', 10000, 'Arena');
   INSERT INTO Venue(venueID, venueName, venueAddress, capacity, venueType)
   VALUES (2, 'Hillside Stadium', '456 Elm Street', 30000, 'Stadium');
   INSERT INTO Venue(venueID, venueName, venueAddress, capacity, venueType)
   VALUES (3, 'Grand Theater', '789 Oak Avenue', 1000, 'Theatre');
   INSERT INTO Venue(venueID, venueName, venueAddress, capacity, venueType)
   VALUES (4, 'Elegant Banquet Hall', '222 Rose Lane', 100, 'Banquet Hall');
   INSERT INTO Venue(venueID, venueName, venueAddress, capacity, venueType)
   VALUES (5, 'Downtown Convention Center', '101 Maple Drive', 50, 'Conference Room');

4. Seat(**venueID**, seatNum, seatType):
   INSERT INTO Seat(venueID, seatNum, seatType) VALUES (1, 'A101', 'VIP');
   INSERT INTO Seat(venueID, seatNum, seatType) VALUES (1, 'A102', 'VIP');
   INSERT INTO Seat(venueID, seatNum, seatType) VALUES
   (2, 'Section 1, Row A, Seat 1', 'Standard');
   INSERT INTO Seat(venueID, seatNum, seatType) VALUES
   (3, 'Section D, Seat 1', 'Standard');
   INSERT INTO Seat(venueID, seatNum, seatType) VALUES
   (3, 'Section B, Seat 10', 'VIP');

5. EventName(**venueID**, dateTime, eventName):
   INSERT INTO EventName(venueID, dateTime, eventName) VALUES
   (1, '2023-10-20 18:00:00', 'Grand Opening Gala');
   INSERT INTO EventName(venueID, dateTime, eventName) VALUES
   (2, '2023-11-15 14:30:00', 'Music Festival');
   INSERT INTO EventName(venueID, dateTime, eventName) VALUES
   (3, '2023-12-05 19:00:00', 'Shakespearean Play');
   INSERT INTO EventName(venueID, dateTime, eventName) VALUES
   (4, '2023-11-30 20:00:00', 'Annual Awards Ceremony');
   INSERT INTO EventName(venueID, dateTime, eventName) VALUES
   (5, '2023-10-25 10:00:00', 'Tech Conference');

6. Event(eventID, **venueID**, **dateTime**):
   INSERT INTO Event(eventID, venueID, dateTime) VALUES
   (1, 1, '2023-10-20 18:00:00');
   INSERT INTO Event(eventID, venueID, dateTime) VALUES
   (2, 2, '2023-11-15 14:30:00');
   INSERT INTO Event(eventID, venueID, dateTime) VALUES
   (3, 3, '2023-12-05 19:00:00');

INSERT INTO Event(eventID, venueID, dateTime) VALUES
(4, 4, '2023-11-30 20:00:00');
INSERT INTO Event(eventID, venueID, dateTime) VALUES
(5, 5, '2023-10-25 10:00:00');

7. Genre(performer, genre):
   INSERT INTO Genre(performer, genre) VALUES ('Taylor Swift', 'Pop');
   INSERT INTO Genre(performer, genre) VALUES ('John Legend', 'R&B');
   INSERT INTO Genre(performer, genre) VALUES ('Coldplay', 'Alternative');
   INSERT INTO Genre(performer, genre) VALUES ('Adele', 'Soul');
   INSERT INTO Genre(performer, genre) VALUES ('The Beatles', 'Rock');

8. Concert(**eventID**, **performer**):
   INSERT INTO Concert(eventID, performer) VALUES (6, 'Taylor Swift');
   INSERT INTO Concert(eventID, performer) VALUES (7, 'John Legend');
   INSERT INTO Concert(eventID, performer) VALUES (8, 'Coldplay');
   INSERT INTO Concert(eventID, performer) VALUES (9, 'Adele');
   INSERT INTO Concert(eventID, performer) VALUES (10, 'The Beatles');

9. Play(**eventID**, director):
   INSERT INTO Play(eventID, director) VALUES (11, 'John Smith');
   INSERT INTO Play(eventID, director) VALUES (12, 'Sarah Johnson');
   INSERT INTO Play(eventID, director) VALUES (13, 'Michael Brown');
   INSERT INTO Play(eventID, director) VALUES (14, 'Emily Davis');
   INSERT INTO Play(eventID, director) VALUES (15, 'David Wilson');

10. Conference(**eventID**, host):
    INSERT INTO Conference(eventID, host) VALUES (16, 'TechCorp'),
    INSERT INTO Conference(eventID, host) VALUES (17, 'BusinessExpo');
    INSERT INTO Conference(eventID, host) VALUES (18, 'Medical Association');
    INSERT INTO Conference(eventID, host) VALUES (19, 'Education Summit');
    INSERT INTO Conference(eventID, host) VALUES (20, 'Finance Forum');

11. TicketPrice(ticketType, **eventID**, ticketPrice):
    INSERT INTO TicketPrice(ticketType, eventID, ticketPrice) VALUES ('GA', 1, 50);
    INSERT INTO TicketPrice(ticketType, eventID, ticketPrice) VALUES ('VIP', 1, 150);
    INSERT INTO TicketPrice(ticketType, eventID, ticketPrice) VALUES ('Standard', 2, 40);
    INSERT INTO TicketPrice(ticketType, eventID, ticketPrice) VALUES ('Student', 2, 25);
    INSERT INTO TicketPrice(ticketType, eventID, ticketPrice) VALUES ('Regular', 3, 30);

12. TicketStatus(**transID**, status):
    INSERT INTO TicketStatus(transID, status) VALUES (0, ‘Available’);
    INSERT INTO TicketStatus(transID, status) VALUES (1849354, 'Sold');
    INSERT INTO TicketStatus(transID, status) VALUES (2128973, 'Sold');

INSERT INTO TicketStatus(transID, status) VALUES (3127543, 'Sold'');
INSERT INTO TicketStatus(transID, status) VALUES (4904532, 'Sold');
INSERT INTO TicketStatus(transID, status) VALUES (5012394, 'Sold'');

13. Ticket(ticketID, **ticketType**, **eventID**, **transID**):
INSERT INTO Ticket(ticketID, ticketType, eventID, transID) VALUES
(1578, 'General Admission', 1, 1849354);
INSERT INTO Ticket(ticketID, ticketType, eventID, transID) VALUES
(1580, 'General Admission', 1, 1849354);
INSERT INTO Ticket(ticketID, ticketType, eventID, transID) VALUES
(2908, 'VIP', 1, 2128973);
INSERT INTO Ticket(ticketID, ticketType, eventID, transID) VALUES
(2909, 'VIP', 1, 0);
INSERT INTO Ticket(ticketID, ticketType, eventID, transID) VALUES
(28097, 'Standard', 2, 3127543);
INSERT INTO Ticket(ticketID, ticketType, eventID, transID) VALUES
(14934, 'Student', 2, 4904532);
INSERT INTO Ticket(ticketID, ticketType, eventID, transID) VALUES
(389, 'Regular', 3, 5012394);

14. Transaction(transID, paymentMethod, **custID**):
INSERT INTO Transaction(transID, paymentMethod, custID) VALUES
(0, NULL, 0);
INSERT INTO Transaction(transID, paymentMethod, custID) VALUES
(1849354, 'Credit Card', 23235);
INSERT INTO Transaction(transID, paymentMethod, custID) VALUES
(2128973, 'Cash', 6989);
INSERT INTO Transaction(transID, paymentMethod, custID) VALUES
(3127543, 'PayPal', 102);
INSERT INTO Transaction(transID, paymentMethod, custID) VALUES
(4904532, 'PayPal', 1798);
INSERT INTO Transaction(transID, paymentMethod, custID) VALUES
(5012394, 'Credit Card', 35);

15. CustomerTier(loyaltyPoints, tier):
INSERT INTO CustomerTier(loyaltyPoints, tier) VALUES (0, 'Bronze');
INSERT INTO CustomerTier(loyaltyPoints, tier) VALUES (500, 'Silver');
INSERT INTO CustomerTier(loyaltyPoints, tier) VALUES (1000, 'Gold');
INSERT INTO CustomerTier(loyaltyPoints, tier) VALUES (2000, 'Platinum');
INSERT INTO CustomerTier(loyaltyPoints, tier) VALUES (5000, 'Diamond');

16. Customer(custID, custName, custAddress, custPhone, custEmail, **loyaltyPoints**):
INSERT INTO Customer(custID, custName, custAddress, custPhone, custEmail,
loyaltyPoints) VALUES

(0, NULL, NULL, NULL, ‘Default’, 0);
INSERT INTO Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints) VALUES
(23235, 'John Doe', '123 Main Street', '555-123-4567', 'johndoe@example.com', 500);
INSERT INTO Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints) VALUES
(6989, 'Alice Smith', '456 Elm Road', '555-987-6543', 'alice@example.com', 1000);
INSERT INTO Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints) VALUES
(102, 'Bob Johnson', '789 Oak Avenue', '555-456-7890', 'bob@example.com', 1500);
INSERT INTO Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints) VALUES
(1789, 'Eva Williams', '321 Pine Lane', '555-234-5678', 'eva@example.com', 2000);
INSERT INTO Customer(custID, custName, custAddress, custPhone, custEmail, loyaltyPoints) VALUES
(35, 'David Brown', '567 Cedar Drive', '555-876-5432', 'david@example.com', 2500);

17. Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail):
    INSERT INTO Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail) VALUES (1, 'ABC Corporation', '123 Main Street', '555-123-4567', 'abc@example.com');
    INSERT INTO Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail) VALUES (2, 'XYZ Ltd', '456 Elm Road', '555-987-6543', 'xyz@example.com');
    INSERT INTO Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail) VALUES (3, 'Sample Corp', '789 Oak Avenue', '555-456-7890', 'sample@example.com');
    INSERT INTO Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail) VALUES (4, 'Global Solutions', '321 Pine Lane', '555-234-5678', 'global@example.com');
    INSERT INTO Sponsor(sponsID, sponsName, sponsAddress, sponsPhone, sponsEmail) VALUES (5, 'Mega Corp', '567 Cedar Drive', '555-876-5432', 'mega@example.com');

18. Sponsorship(**eventID**, **sponsorID**, fund):
    INSERT INTO Sponsorship(eventID, sponsorID, fund) VALUES (1, 1, 5000);
    INSERT INTO Sponsorship(eventID, sponsorID, fund) VALUES (1, 2, 3000);
    INSERT INTO Sponsorship(eventID, sponsorID, fund) VALUES (2, 3, 8000);
    INSERT INTO Sponsorship(eventID, sponsorID, fund) VALUES (3, 4, 6000);
    INSERT INTO Sponsorship(eventID, sponsorID, fund) VALUES (4, 5, 7500);

19. Staff(staffID, staffName, staffAddress, staffPhone, staffEmail):
    INSERT INTO Staff(staffID, staffName, staffAddress, staffPhone, staffEmail) VALUES
    (1, 'John Smith', '123 Main Street', '555-123-4567', 'john@example.com');
    INSERT INTO Staff(staffID, staffName, staffAddress, staffPhone, staffEmail) VALUES
    (2, 'Jane Doe', '456 Elm Road', '555-987-6543', 'jane@example.com');
    INSERT INTO Staff(staffID, staffName, staffAddress, staffPhone, staffEmail) VALUES
    (3, 'Robert Johnson', '789 Oak Avenue', '555-456-7890', 'robert@example.com');
    INSERT INTO Staff(staffID, staffName, staffAddress, staffPhone, staffEmail) VALUES

(4, 'Sara Wilson', '321 Pine Lane', '555-234-5678', 'sara@example.com');
INSERT INTO Staff(staffID, staffName, staffAddress, staffPhone, staffEmail) VALUES
(5, 'David Brown', '567 Cedar Drive', '555-876-5432', 'david@example.com');

20. Assignment(**eventID**, **staffID**, role):
    INSERT INTO Assignment(eventID, staffID, role) VALUES (1, 1, 'Coordinator');
    INSERT INTO Assignment(eventID, staffID, role) VALUES (1, 2, 'Assistant');
    INSERT INTO Assignment(eventID, staffID, role) VALUES (2, 3, 'Coordinator');
    INSERT INTO Assignment(eventID, staffID, role) VALUES (3, 4, 'Manager');
    INSERT INTO Assignment(eventID, staffID, role) VALUES (4, 5, 'Technician');