SpringSpree Database Management Project

By Jerom Shobi

21EEB0F04

Email : jeromshobi37@gmail.com, js21eeb0f04@student.nitw.ac.in

LinkedIn : Jerom Shobi

Table of Contents

- Introduction
- Tables
- ER Model Assumptions
- Functional Dependencies and Primary Keys
- Normalization
- ER Diagram and Relationship Schema
- Code

Introduction

In this project, I have designed a database management system using **Python (SQLAlchemy)** and **MySQL** to store information about the annual cultural festival of NIT Warangal, ***SpringSpree***. This database was created with the aim of smoothly conducting *SpringSpree* as well as to analyse the data for improving the festival in the coming years.

The database will contain **personal details of attendees** provided by the attendee at the time of purchasing the ticket. Each attendee may provide the name, age, gender, email, phone number, location and the days of the fest they plan to attend. This data can be used to allow entry for the attendees on appropriate days based on ticket purchase, to send updates and promotion material via email or text and to conduct demographic analysis for expanding marketing reach.

The database also contains information about the various **events** and '**proshows'** aka concerts including the **venue** of the event, the day(s) on which they will be conducted, **starting and ending time** and **sponsoring brand(s)**. Information regarding **stalls** set up by the brands will also be documented.

The database stores details of each **organizer** which helps in coordinating the programs. This includes the unique id, name, year of study, course pursued, branch, role and phone number. The database also stores the event(s) managed by the organizer.

The database records information of the **participants** of competitive events. Participants can compete in multiple events. The data consists of distinct participant id, name, institute, phone number, location, days of accommodation and the event(s) in which they are participating.

Tables

1. Attendees Table

Stores details about the attendee

Attribute	Datatype	Constraints
TicketID	CHAR(6)	PRIMARY KEY, NOT NULL, UNIQUE
AttendeeName	VARCHAR(40)	NOT NULL
Age	NUMBER	NOT NULL
Gender	ENUM (‘M’, ‘F’)	None
Email	VARCHAR(50)	DEFAULT NULL
PhoneNumber	VARCHAR(15)	None
PurchaseType	ENUM (‘Offline’, ‘Online’)	None
PurchaseDate	DATE	NOT NULL
City	VARCHAR(40)	DEFAULT NULL
State	VARCHAR(25)	DEFAULT NULL

2. AttendeeDays Table

Stores details about the days on which the attendee will be attending the festival

Attribute	Datatype	Constraints
TicketID	CHAR(6)	PRIMARY KEY, FOREIGN KEY
Day	ENUM(‘D1’, ‘D2’)	PRIMARY KEY

3. Events Table

Stores details about the different events

Attribute	Datatype	Constraints
EventName	VARCHAR(40)	PRIMARY KEY, NOT NULL, UNIQUE
EventCategory	ENUM('Others', 'Music', 'Dance', 'Drama', 'Quiz', 'Interaction')	None
Venue	VARCHAR(25)	FOREIGN KEY
NumParticipants	NUMBER	DEFAULT NULL
RegistrationFee	NUMBER	DEFAULT NULL

4. EventDays Table

Stores details about the timings of events

Attribute	Datatype	Constraints
EventName	VARCHAR(25)	PRIMARY KEY, FOREIGN KEY
Day	ENUM('D1', 'D2')	PRIMARY KEY
StartTime	TIME	NOT NULL
EndTime	TIME	NOT NULL

5. EventsOrganiser Table

Stores details about the organizer(s) supervising the various events

Attribute	Datatype	Constraints
EventName	VARCHAR(25)	PRIMARY KEY
OrganiserID	CHAR(6)	PRIMARY KEY, FOREIGN KEY
ClubInchargeID	CHAR(6)	FOREIGN KEY, DEFAULT NULL

6. EventSponsors Table

Stores details about the events and their corresponding sponsors

Attribute	Datatype	Constraints
EventName	VARCHAR(25)	PRIMARY KEY, FOREIGN KEY
CompanyName	VARCHAR(50)	PRIMARY KEY, FOREIGN KEY

7. FestClubCodes Table

Stores the different codes or suffixes for clubs

Attribute	Datatype	Constraints
ClubName	VARCHAR(25)	PRIMARY KEY, NOT NULL, UNIQUE
ClubCode	VARCHAR(3)	NOT NULL, UNIQUE

8. FestClubs Table

Stores details about the clubs involved in the festival

Attribute	Datatype	Constraints
ClubName	VARCHAR(30)	PRIMARY KEY,

		NOT NULL, UNIQUE
EventInCharge	VARCHAR(50)	NOT NULL
EventInChargeID	CHAR(6)	NOT NULL, UNIQUE
PhoneNumber	VARCHAR(15)	NOT NULL, UNIQUE

9. FestTeam Table

Stores details about the organisers associated with the festival

Attribute	Datatype	Constraints
OrganiserID	VARCHAR(6)	PRIMARY KEY, NOT NULL, UNIQUE
OrganiserName	VARCHAR(50)	NOT NULL
Course	VARCHAR(15)	NOT NULL
Year	ENUM (‘1’, ‘2’, ‘3’, ‘4’, ‘5’)	None
Branch	ENUM('CSE', 'ECE', 'EEE', 'MECH', 'CVL', 'MME', 'CHEM', 'BIOT', 'PHY', 'CHY', 'MATH')	NOT NULL
PhoneNumber	VARCHAR(15)	NOT NULL, UNIQUE
Role	ENUM (‘CORE’, ‘SUBCORE’)	NOT NULL

10. Subteams Table

Stores the different codes or suffixes for different sub-teams in the festival

Attribute	Datatype	Constraints
SubteamCode	VARCHAR(3)	NOT NULL, UNIQUE
SubteamName	VARCHAR(40)	PRIMARY KEY, NOT NULL, UNIQUE

11. Participants Table

Stores personal details about the participants

Attribute	Datatype	Constraints
ParticipantID	CHAR(4)	PRIMARY KEY, NOT NULL, UNIQUE
ParticipantName	VARCHAR(50)	NOT NULL
Age	NUMBER	NOT NULL

Institute	VARCHAR(5)	NOT NULL
Email	VARCHAR(40)	NOT NULL
SoloOrGroup	ENUM('Solo', 'Group')	None
TeamName	VARCHAR(20)	DEFAULT NULL
AccomodationDays	INT	NOT NULL, CHECK IN (0, 1, 2)
RegDate	DATE	NOT NULL

12. ParticipantEvents Table

Stores records about the events which different participants are taking part in

Attribute	Datatype	Constraints
ParticipantID	CHAR(4)	PRIMARY KEY, FOREIGN KEY
EventName	VARCHAR(25)	PRIMARY KEY, FOREIGN KEY

13. Proshows Table

Stores details about the 'proshows' aka concerts

Attribute	**Datatype**	**Constraints**
Performer	VARCHAR(40)	PRIMARY KEY, NOT NULL, UNIQUE
InchargeID	CHAR(6)	FOREIGN KEY
Day	ENUM('D1', 'D2')	None
StartTime	TIME	NOT NULL
EndTime	TIME	NOT NULL

14. Sponsors Table

Stores details about the sponsors

Attribute	**Datatype**	**Constraints**
CompanyName	VARCHAR(50)	PRIMARY KEY, NOT NULL, UNIQUE
Industry	ENUM ('Transportation', 'Pharmaceutical', 'Telecommunications', 'Manufacturing', 'Hospitality', 'Media', 'Agriculture', 'IT', 'Education',	None

	'Finance', 'Healthcare', 'Automobile', 'Dairy', 'Construction', 'Sports', 'Apparels', 'Tourism', 'Beverages', 'Insurance', 'Banking', 'Electronics', 'Food Service')	
AmountSponsored	NUMBER	NOT NULL
Cash_Or_InKind	ENUM (‘Cash’, ‘In Kind’)	None
Category	ENUM ('Diamond', 'Platinum', 'Gold', 'Silver')	None
PrevCollab	ENUM ('Yes', 'No')	None
Contact	VARCHAR(40)	NOT NULL
FestContact	CHAR(6)	NOT NULL, FOREIGN KEY
Email	VARCHAR(40)	DEFAULT NULL
PhoneNumber	VARCHAR(15)	NOT NULL

15. Stalls Table

Stores details about the stalls

Attribute	**Datatype**	**Constraints**
StallID	NUMBER	PRIMARY KEY
CompanyName	VARCHAR(50)	None
Type	ENUM('Food', 'Merch', 'Promotion')	None
NumStalls	NUMBER	None
NumCustomers	NUMBER	None

16. Venue Table

Stores the details of venues across the college

Attribute	**Datatype**	**Constraints**
VenueName	VARCHAR(25)	NOT NULL, UNIQUE, PRIMARY KEY
Capacity	INT	DEFAULT NULL
Location	VARCHAR(25)	NOT NULL

ER Model Assumptions

1. An attendee can buy only one ticket to SpringSpree and is uniquely identified by his/her TicketID.
2. The attendee can attend either the first day (D1), second day (D2) or both days of the festival
3. The festival is comprised of multiple events and 'proshows' conducted in venues across the campus
4. Events are organized by an organizer from the fest team and optionally along with a club member of the appropriate club.
5. Participants can officially register for one more competitive event, either as a solo participant or as a group
6. These events may be sponsored by a company or brand. Sponsors belong to different categories depending on the amount sponsored. Stalls might also be set up by these companies for different purposes.
7. Each day of the festival has two proshows by performers. These are also managed by organizers from the fest team.

Functional Dependencies and Primary Keys

1. **Attendee**

   TicketID → { AttendeeName, Age, Gender, Email, PhoneNumber, PurchaseType, PurchaseDate, City, State }

   Since all fields depend on TicketID, (TicketID)+ → R

   Hence, TicketID is the primary key

2. **AttendeeDays**

   (TicketID, Day) → {TicketID, Day}

   Since all fields depend on TicketID and Day, (TicketID, Day)+ → R

   Hence, (TicketID, Day) is the primary key

3. Events

EventName → { EventCategory, Venue, NumParticipants, RegistrationFees}

Since all fields depend on EventName, (EventName)+ → R

Hence, EventName is the primary key

4. EventDays

(EventName, Day) → { StartTime, EndTime }

Since all fields depend on (EventName, Day), (EventName, Day)+ → R

Hence, (EventName, Day) is the primary key

5. EventsOrganiser

(EventName, OrganiserID) → { ClubInchargeID }

Since all fields depend on (EventName, OrganiserID),

(EventName, OrganiserID)+ → R

Hence, (EventName, OrganiserID) is the primary key

6. EventSponsors

(EventName, CompanyName) → { EventName, CompanyName }

Since all fields depend on (EventName, CompanyName),

(EventName, CompanyName)+ → R

Hence, (EventName, CompanyName) is the primary key

7. FestClubCodes

ClubName → { ClubCode }

Since all fields depend on ClubName, (ClubName)+ → R

Hence, ClubName is the primary key

8. FestTeam

OrganiserID → { OrganiserName, Course, Year, Branch, Role, PhoneNumber }

Since all fields depend on OrganiserID, (OrganiserID)+ → R

Hence, OrganiserID is the primary key

9. Subteams

SubteamName→ { SubteamCode }

Since all fields depend on SubteamName, (SubteamName)+ → R

Hence, SubteamName is the primary key

10. Participants

ParticipantID → { ParticipantName, Age, Email, Institute, SoloOrGroup, TeamName, AccomodationDays, RegDate }

Since all fields depend on ParticipantID , (ParticipantID)+ → R

Hence, ParticipantID is the primary key

11. ParticipantEvents

(ParticipantID, EventName) → { ParticipantID, EventName }

Since all fields depend on (ParticipantID, EventName), (ParticipantID, EventName)+ → R

Hence, (ParticipantID, EventName) is the primary key

12. Proshows

Performer → { InchargeID, Day, StartTime, EndTime}

Since all fields depend on Performer, (Performer)+ → R

Hence, Performer is the primary key

13. Sponsors

CompanyName → { Industry, AmountSponsored, Cash_Or_InKind, Category, PrevCollab, Contact, FestContact, Email, PhoneNumber }

Since all fields depend on CompanyName, (CompanyName)+ → R

Hence, CompanyName is the primary key

14. **Stalls**

StallID → { CompanyName, Type, NumStalls, NumCustomers }

Since all fields depend on StallD, (StallD)+ → R

Hence, StallD is the primary key

15. **Venue**

VenueName → { Capacity, Location }

Since all fields depend on VenueName, (VenueName)+ → R

Hence, VenueName is the primary key

Normalization

1. **Attendee**

   Primary Key: TicketID

   All attributes depend on the TicketID; hence the table is 2NF.

   All attributes depend directly on TicketID; hence the table is in 3NF

   All determinants (TicketID) are candidate keys, hence the table is in BCNF

2. **AttendeeDays**

   Primary Key: (TicketID, Day)

   All attributes depend on (TicketID, Day); hence the table is 2NF.

   All attributes depend directly on (TicketID, Day); hence the table is in 3NF

   All determinants (TicketID, Day) are candidate keys, hence the table is in BCNF

3. **Events**

   Primary Key: EventName

   All attributes depend on the EventName ; hence the table is 2NF.

   All attributes depend directly on EventName; hence the table is in 3NF

   All determinants (EventName) are candidate keys, hence the table is in BCNF

4. **EventDays**

   Primary Key: (EventName, Day)

   All attributes depend on (EventName, Day); hence the table is 2NF.

   All attributes depend directly on (EventName, Day); hence the table is in 3NF

   All determinants (EventName, Day) are candidate keys, hence the table is in BCNF

5. **EventsOrganiser**

   Primary Key: (EventName, OrganiserID)

   All attributes depend on (EventName, OrganiserID); hence the table is 2NF.

All attributes depend directly on (EventName, OrganiserID); hence the table is in 3NF

All determinants (EventName, OrganiserID) are candidate keys, hence the table is in BCNF

6. **EventsSponsors**

Primary Key: (EventName, CompanyName)

All attributes depend on (EventName, CompanyName); hence the table is 2NF.

All attributes depend directly on (EventName, CompanyName); hence the table is in 3NF

All determinants (EventName, CompanyName) are candidate keys, hence the table is in BCNF

7. **FestClubCodes**

Primary Key: ClubName

All attributes depend on the ClubName; hence the table is 2NF.

All attributes depend directly on ClubName; hence the table is in 3NF

All determinants (ClubName) are candidate keys, hence the table is in BCNF

8. **FestClubs**

Primary Key: ClubName

All attributes depend on the ClubName; hence the table is 2NF.

All attributes depend directly on ClubName; hence the table is in 3NF

All determinants (ClubName) are candidate keys, hence the table is in BCNF

9. **FestTeam**

Primary Key: OrganiserID

All attributes depend on the OrganiserID; hence the table is 2NF.

All attributes depend directly on OrganiserID; hence the table is in 3NF

All determinants (OrganiserID) are candidate keys, hence the table is in BCNF

10. **Subteams**

Primary Key: SubteamName

All attributes depend on the SubteamName; hence the table is 2NF.

All attributes depend directly on SubteamName; hence the table is in 3NF

All determinants (SubteamName) are candidate keys, hence the table is in BCNF

11. **Participants**

Primary Key: ParticipantName

All attributes depend on the ParticipantName; hence the table is 2NF.

All attributes depend directly on ParticipantName; hence the table is in 3NF

All determinants (ParticipantName) are candidate keys, hence the table is in BCNF

12. **ParticipantEvents**

Primary Key: (ParticipantID, EventName)

All attributes depend on (ParticipantID, EventName); hence the table is 2NF.

All attributes depend directly on (ParticipantID, EventName); hence the table is in 3NF

All determinants (ParticipantID, EventName) are candidate keys, hence the table is in BCNF

13. **Proshows**

Primary Key: Performer

All attributes depend on the Performer; hence the table is 2NF.

All attributes depend directly on Performer; hence the table is in 3NF

All determinants (Performer) are candidate keys, hence the table is in BCNF

14. Sponsors

Primary Key: CompanyName

All attributes depend on the CompanyName; hence the table is 2NF.

All attributes depend directly on CompanyName; hence the table is in 3NF

All determinants (CompanyName) are candidate keys, hence the table is in BCNF

15. Stalls

Primary Key: StallID

All attributes depend on the StallID; hence the table is 2NF.

All attributes depend directly on StallID; hence the table is in 3NF

All determinants (StallID) are candidate keys, hence the table is in BCNF

16. Venue

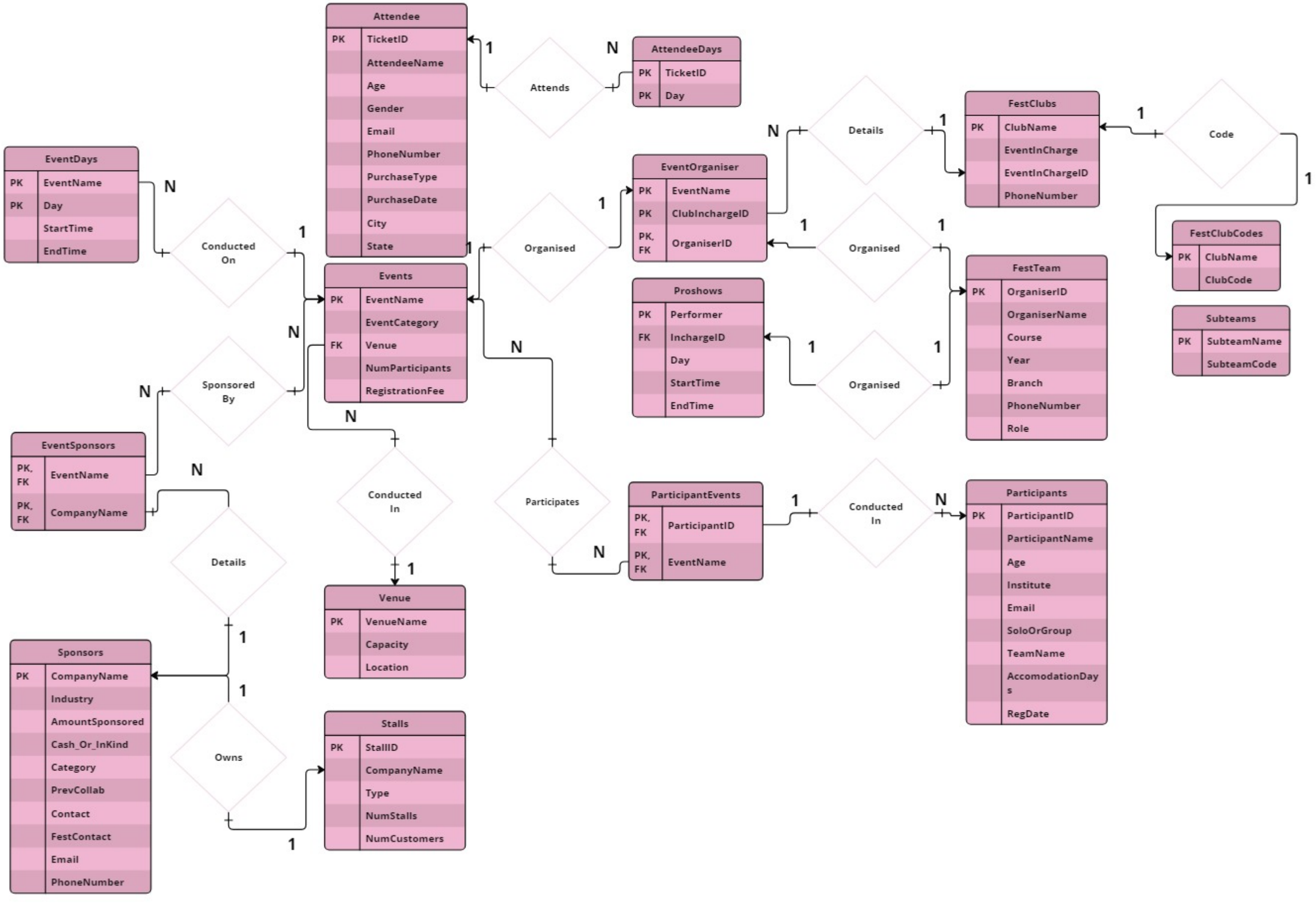
Primary Key: VenueName

All attributes depend on the VenueName; hence the table is 2NF.

All attributes depend directly on VenueName; hence the table is in 3NF

All determinants (VenueName) are candidate keys, hence the table is in BCNF

ER Diagram and Relationship Schema



Code

For Creating 5 Tables

```
create_attendees_query = """
CREATE TABLE IF NOT EXISTS Attendees(
    TicketID CHAR(6) NOT NULL UNIQUE PRIMARY KEY,
    AttendeeName VARCHAR(40) NOT NULL,
    Age INT NOT NULL,
    Gender ENUM ('M', 'F'),
    Email VARCHAR(50) DEFAULT NULL,
    PhoneNumber VARCHAR(15),
    PurchaseType ENUM('Offline', 'Online'),
    PurchaseDate DATE NOT NULL,
    City VARCHAR(40) DEFAULT NULL,
    State VARCHAR(25) DEFAULT NULL
);
"""
connection.execute(text(create_attendees_query))
print("Table 'Attendees' created or already exists")
```

```
create_attendeedays_query = """
CREATE TABLE IF NOT EXISTS AttendeeDays(
    TicketID CHAR(6),
    Day ENUM ('D1', 'D2'),
    PRIMARY KEY (TicketID, Day),
    FOREIGN KEY (TicketID) REFERENCES Attendees(TicketID)
);
"""
connection.execute(text(create_attendeedays_query))
print("Table 'AttendeeDays' created or already exists")
```

```
create_events_query = """
CREATE TABLE IF NOT EXISTS Events(
    EventName VARCHAR(40) NOT NULL UNIQUE PRIMARY KEY,
    EventCategory ENUM ('Others', 'Music', 'Dance', 'Drama', 'Quiz',
'Interaction'),
    Venue VARCHAR(25),
    NumParticipants INT UNSIGNED DEFAULT NULL,
    RegistrationFees INT UNSIGNED DEFAULT NULL,
    FOREIGN KEY (Venue) REFERENCES Venue(VenueName)
    ON DELETE SET NULL ON UPDATE CASCADE
)
"""
connection.execute(text(create_events_query))
print("Table 'Events' created or already exists")
```

```
create_eventdays_query = """
CREATE TABLE IF NOT EXISTS EventDays(
    EventName VARCHAR(25),
    Day ENUM ('D1', 'D2'),
    StartTime TIME NOT NULL,
    EndTime TIME NOT NULL,
    PRIMARY KEY('EventName', 'Day'),
    FOREIGN KEY (EventName) REFERENCES Events(EventName)
);
"""
connection.execute(text(create_eventdays_query))
print("Table 'EventDays' created or already exists")
```

```
create_events_organiser_query = """
CREATE TABLE IF NOT EXISTS EventsOrganiser (
    EventName VARCHAR(25),
    OrganiserID CHAR(6) NOT NULL,
    PRIMARY KEY (EventName, OrganiserID),
    ClubInchargeID CHAR(6) DEFAULT NULL,
    FOREIGN KEY (ClubInchargeID) REFERENCES
FestClubs(EventInchargeID),
    FOREIGN KEY (OrganiserID) REFERENCES FestTeam(OrganiserID)
);
"""
connection.execute(text(create_events_organiser_query))
print('Table "EventsOrganiser" created or already exists')
```

For Inserting Into Above Tables

```
from datetime import datetime, timedelta
from faker import Faker
from datetime import date

fake = Faker('en_IN')
insert_attendees_query = """
INSERT INTO Attendees
    VALUES (:value1, :value2, :value3, :value4, :value5, :value6,
    :value7, :value8, :value9, :value10)
"""
for i in range(100, 600):
    start_date = date(2024, 2, 1)
    end_date = date(2024, 3, 31)
    random_date = fake.date_between(start_date=start_date,
end_date=end_date)
    ticketid = 'RSG' + str(i)
    fakeattend = fake.name()
```

```
    emailchoice = np.random.choice(
        [f"{fakeattend.split()[1].lower()}{np.random.randint(20,
2005)}@gmail.com",
         None])

    if emailchoice == None:
        purtype = 'Offline'
    else:
        purtype = 'Online'

    city = np.random.choice([fake.city(), 'Warangal'])
    if city == 'Warangal':
        state = 'Telangana'
    else:
        state = np.random.choice(['Andhra Pradhesh', 'Maharashtra',
'Telangana'])

    connection.execute(
        text(insert_attendees_query),
        {'value1' : ticketid, 'value2' : fakeattend,
         'value3' : np.random.randint(18, 60),
         'value4' : np.random.choice(['M', 'F']),
         'value5' : emailchoice, 'value7' : purtype,
         'value6' : fake.phone_number(),
         'value8' : random_date,
         'value9' : city,
         'value10' : state
        })
```

```
# Getting TicketIDs of all Attendees
select_tickid_query = """
SELECT TicketID FROM Attendees
```

```
"""
results = connection.execute(text(select_tickid_query))
tickids = list()
for i in results:
    tickids.append(i[0])

insert_attendeedays_query = """
INSERT INTO AttendeeDays VALUES (:value1, :value2)
"""

# Randomly inserting TicketIDs for D1 and D2
for i in tickids:
    connection.execute(
        text(insert_attendeedays_query),
        {'value1' : i, 'value2' : np.random.choice(['D1', 'D2'])})

# Getting list of TicketIDs for Day 1
select_tickid_query = """
SELECT TicketId FROM AttendeeDays WHERE Day = 'D1'
"""
results = connection.execute(text(select_tickid_query))
ticks_d1 = list()
for i in results:
    ticks_d1.append(i[0])

# Getting list of TicketIDs for Day 2
select_tickid_query = """
SELECT TicketId FROM AttendeeDays WHERE Day = 'D2'
"""
results = connection.execute(text(select_tickid_query))
ticks_d2 = list()
for i in results:
```

```
    ticks_d2.append(i[0])

# Eliminating random TicketIDs from Day 1
np.random.shuffle(ticks_d1)
for i in range(np.random.randint(50, 200)):
    ticks_d1.pop()

# Inserting remaining TicketIDs for Day 2
for i in ticks_d1:
    connection.execute(
        text(insert_attendeedays_query),
        {'value1' : i, 'value2' : 'D2'})

# Eliminating random TicketIDs from Day 2
np.random.shuffle(ticks_d2)
for i in range(np.random.randint(50, 200)):
    ticks_d2.pop()

# Inserting remaining TicketIDs for Day 1
for i in ticks_d2:
    connection.execute(
        text(insert_attendeedays_query),
        {'value1' : i, 'value2' : 'D1'})
```

```
events = ['Unsolved', 'Nukkad Natak' , 'Drama Competition', 'Choreo
Night',
          'Anime Museum', 'Photo Hunt', 'Film Treasure Hunt',
          'Solo Idol', 'Film Quiz', 'Interaction with Navdeep',
'Battle Of Bands']
eventcat = ['Others', 'Drama', 'Drama', 'Dance', 'Others', 'Others',
'Others',
```

```
            'Music', 'Quiz', 'Interaction', 'Music']
eventvenue = ['ALC 202', 'ALC 203', 'Homi Bhabha Hall', 'Stadium',
              'Ramanujan Hall', 'ALC 103', 'ALC 102',
              'Bose Hall', 'ALC 201', 'ALC Hall', 'Old Auditorium']
eventpar = [None, None, 20, 23, None, 30, 22, 36, 32, None, 23]
regfees = [None, None, None, 1000, None, None, None, 600, None, None,
1000]

insert_events_query = """
INSERT INTO Events VALUES (:value1, :value2, :value3, :value4,
:value5);
"""

for i in range(len(events)):
    connection.execute(text(insert_events_query),
                       {"value1": events[i], "value2": eventcat[i],
                        "value3": eventvenue[i], "value4":eventpar[i],
                        "value5": regfees[i]})
```

```
events_d1 = {
       'Unsolved': ['D1', time(14, 0), time(18, 0)],
        'Nukkad Natak': ['D1', time(16, 0), time(18, 0)],
        'Drama Competition': ['D1', time(14, 0), time(18, 0)],
        'Choreo Night': ['D1', time(22, 0), time(0, 0)],
        'Anime Museum': ['D1', time(14, 0), time(20, 0)],
        'Photo Hunt': ['D1', time(14, 0), time(17, 0)],
        'Film Treasure Hunt': ['D1', time(16, 0), time(18, 0)],
        'Film Quiz': ['D1', time(16, 0), time(18, 0)]}

events_d2 ={'Anime Museum': ['D2', time(14, 0), time(20, 0)],
        'Film Treasure Hunt' : ['D2', time(11, 0), time(15, 0)],
        'Solo Idol': ['D2', time(12, 0), time(17, 0)],
```

```
        'Film Quiz': ['D2', time(10, 0), time(14, 0)],
        'Interaction with Navdeep': ['D2', time(11, 0), time(14, 0)],
        'Battle Of Bands' : ['D2', time(16, 0), time(19, 0)]
    }

insert_eventdays_query = """
    INSERT INTO EventDays VALUES (:value1, :value2, :value3, :value4);
    """

for i in events_d1:
    connection.execute(text(insert_eventdays_query),
                       {"value1" : i, "value2" : events_d1[i][0],
                        "value3" : events_d1[i][1],
                        "value4" : events_d1[i][2]})

for i in events_d2:
    connection.execute(text(insert_eventdays_query),
                       {"value1" : i, "value2" : events_d2[i][0],
                        "value3" : events_d2[i][1],
                        "value4" : events_d2[i][2]})
```

```
events = ['Unsolved', 'Nukkad Natak' , 'Drama Competition', 'Choreo
Night',
          'Anime Museum', 'Photo Hunt', 'Film Treasure Hunt',
          'Solo Idol', 'Film Quiz', 'Interaction with Navdeep',
         'Battle Of Bands']
orgr_ids = ['ECC101', 'ECC109', 'ECC114', 'ECC100',
           'ECC103', 'ECC116', 'ECC118',
           'ECC119', 'ECC106', 'ECC112', 'ECC102']
club_ids = ['LDC100', 'LDC100', None, 'DND100',
            'ACC100', 'PH100', 'FM100',
```

```
            'MU100', 'FM100', None, 'MU100']

insert_eventsorganiser_query = """
INSERT INTO EventsOrganiser VALUES (:value1, :value2, :value3)
"""
for i in range(len(events)):
    connection.execute(text(insert_eventsorganiser_query),
                       { 'value1' : events[i], 'value2' : orgr_ids[i],
                        'value3' : club_ids[i]})
```