Event Planning System

Group 2:

Hak Monghout

Sopha Nara

Hour Chansopheak

Lecturer: Leng Rathana

TABLE OF CONTENT

Chapter One:

Introduction

A. Introduction

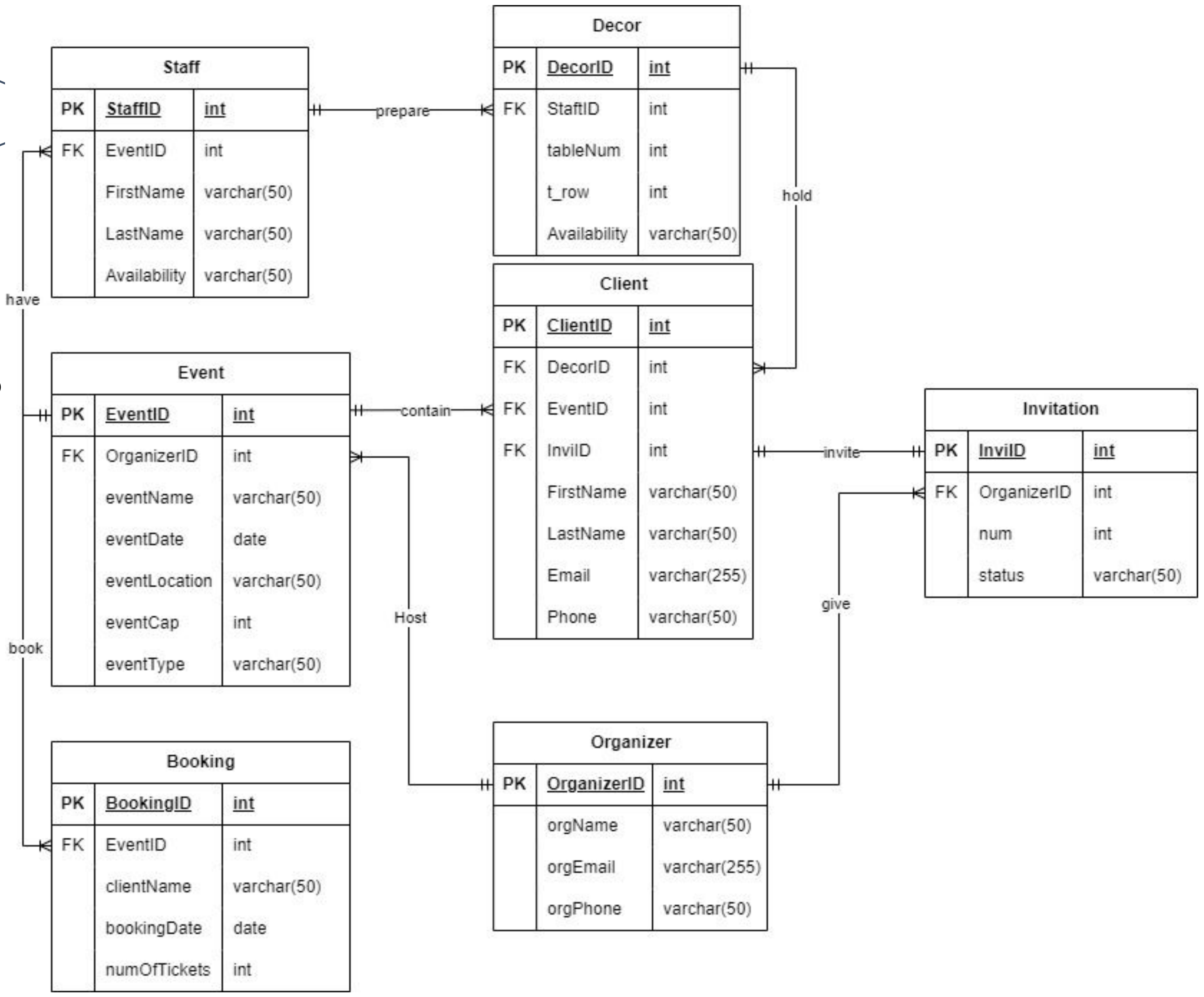
The Event Planning System is a software solution designed to Help with setting up the planning and organizing events. It offers a range of features and functionalities to assist users in managing various aspects of event management, from client and staff coordination to date and time scheduling. It includes managing various events, including concerts, conferences, dinners, and sports tournaments. EPS's primary goal is to help clients plan and execute successful events with precision and efficiency. The services cater to the needs of event organizers, clients, and attendees, ensuring smooth event management.

B. Overview of the Proposed Database System

EPS aims to develop a comprehensive and robust database system to streamline its operations and improve event management. This system will serve as the foundation for all event-related activities, offering an organized approach to store, retrieve, and analyze data.The proposed database system will play a vital role in:Managing Event Information: Storing event details like names, dates, locations, capacities, and types (e.g., concerts, conferences, festivals) in a centralized repository for event tracking and planning.Handling Client and Organizer Information: Efficiently managing client and event organizer details, facilitating effective communication and coordination during event planning.

Chapter Two:
Database Design

A: Entity Relationship Diagram (ERD)



Staff
Pk : staffid
Fk :eventid

Decor
Pk : decorID
Fk : staffID

Event
Pk : EventID
Fk : OrganizerID

Booking
Pk : BookingID
Fk : eventID

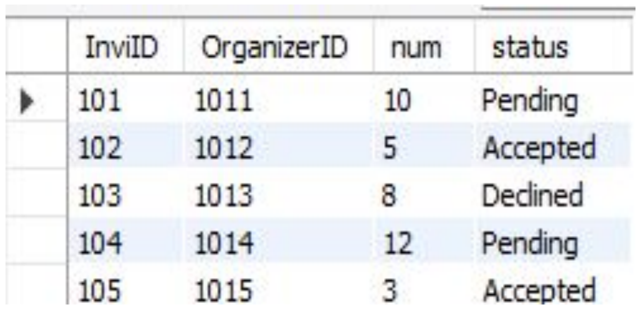
Organizer
PK : OrganizerID

Client
PK: ClientID
Fk: DecorID
Fk: EventID
Fk: InviID

Invitation
Pk: InviID
Fk: OrganizerID

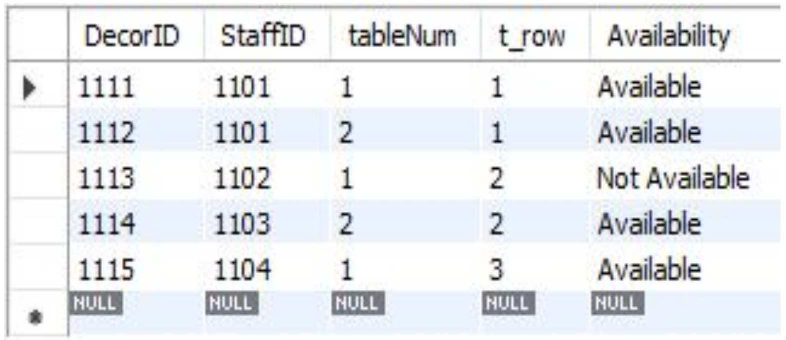
B: Normalization table

Invitation Table



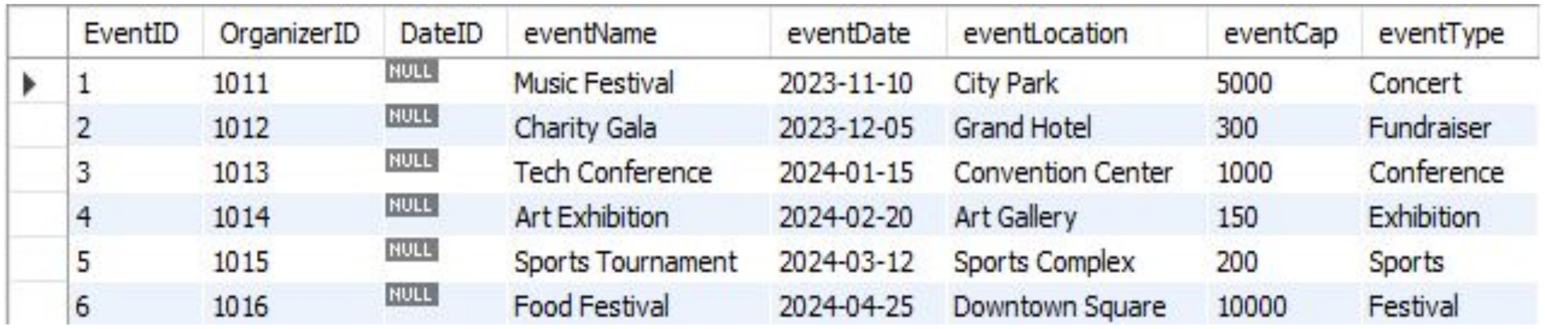
InviID	OrganizerID	num	status
101	1011	10	Pending
102	1012	5	Accepted
103	1013	8	Declined
104	1014	12	Pending
105	1015	3	Accepted

Decor Table



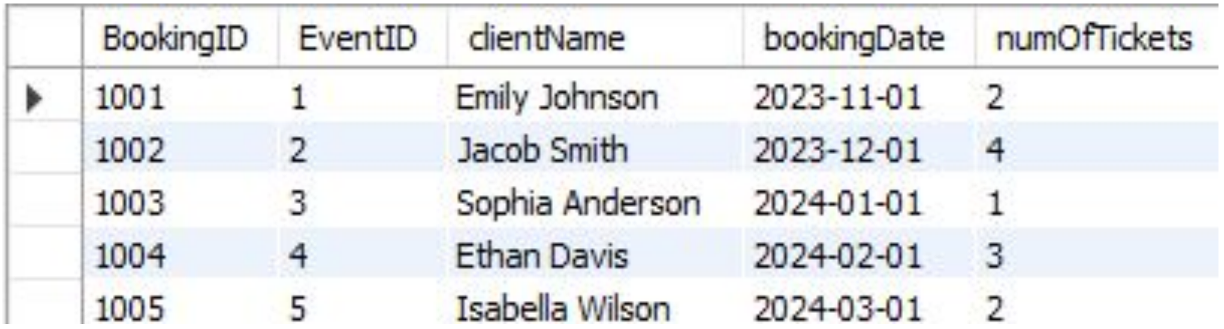
DecorID	StaffID	tableNum	t_row	Availability
1111	1101	1	1	Available
1112	1101	2	1	Available
1113	1102	1	2	Not Available
1114	1103	2	2	Available
1115	1104	1	3	Available
NULL	NULL	NULL	NULL	NULL

Event Table



EventID	OrganizerID	DateID	eventName	eventDate	eventLocation	eventCap	eventType
1	1011	NULL	Music Festival	2023-11-10	City Park	5000	Concert
2	1012	NULL	Charity Gala	2023-12-05	Grand Hotel	300	Fundraiser
3	1013	NULL	Tech Conference	2024-01-15	Convention Center	1000	Conference
4	1014	NULL	Art Exhibition	2024-02-20	Art Gallery	150	Exhibition
5	1015	NULL	Sports Tournament	2024-03-12	Sports Complex	200	Sports
6	1016	NULL	Food Festival	2024-04-25	Downtown Square	10000	Festival

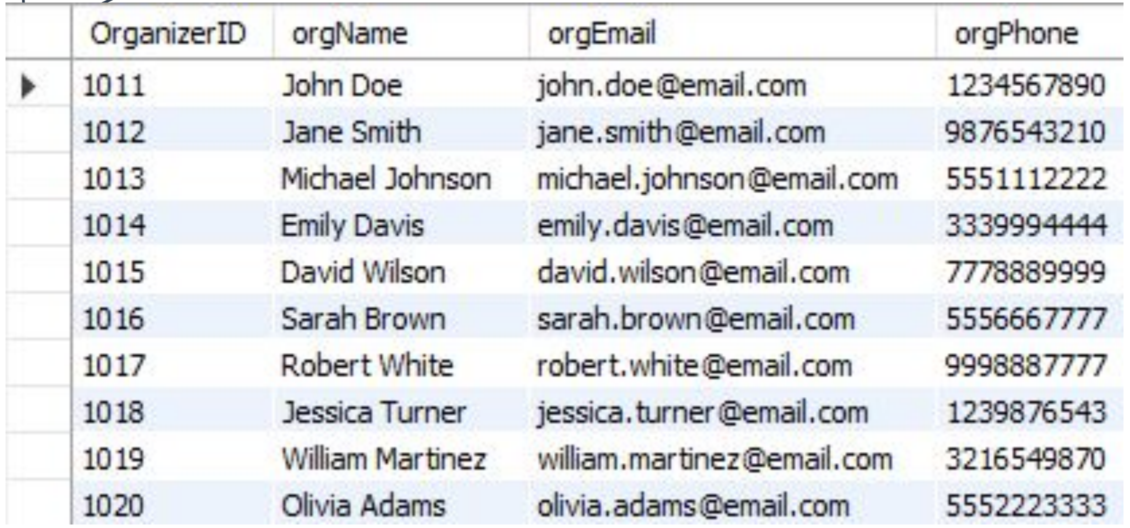
Booking Table



BookingID	EventID	clientName	bookingDate	numOfTickets
1001	1	Emily Johnson	2023-11-01	2
1002	2	Jacob Smith	2023-12-01	4
1003	3	Sophia Anderson	2024-01-01	1
1004	4	Ethan Davis	2024-02-01	3
1005	5	Isabella Wilson	2024-03-01	2

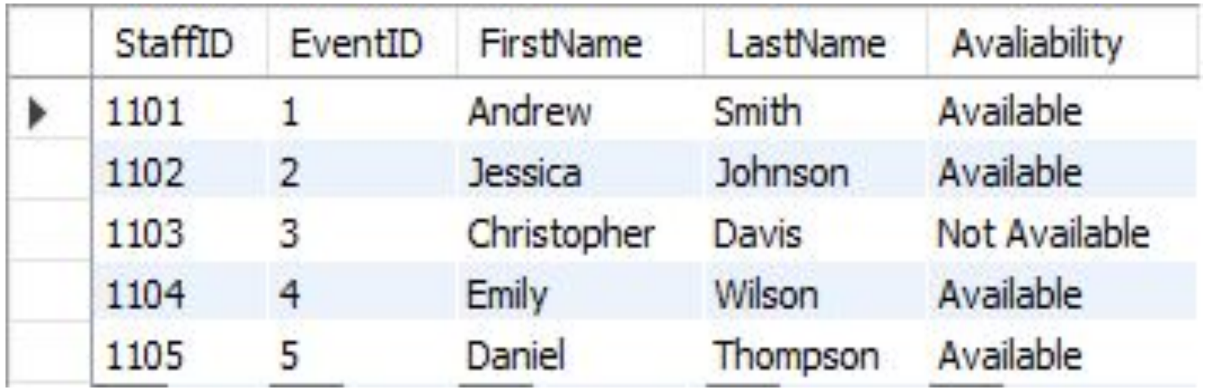
B: Normalization table

Organizer Table



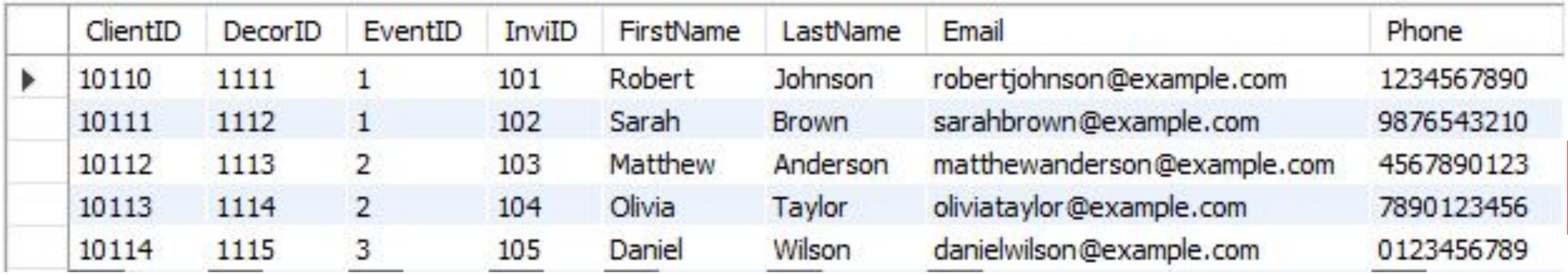
OrganizerID	orgName	orgEmail	orgPhone
1011	John Doe	john.doe@email.com	1234567890
1012	Jane Smith	jane.smith@email.com	9876543210
1013	Michael Johnson	michael.johnson@email.com	5551112222
1014	Emily Davis	emily.davis@email.com	3339994444
1015	David Wilson	david.wilson@email.com	7778889999
1016	Sarah Brown	sarah.brown@email.com	5556667777
1017	Robert White	robert.white@email.com	9998887777
1018	Jessica Turner	jessica.turner@email.com	1239876543
1019	William Martinez	william.martinez@email.com	3216549870
1020	Olivia Adams	olivia.adams@email.com	5552223333

Staff Table



StaffID	EventID	FirstName	LastName	Avaliability
1101	1	Andrew	Smith	Available
1102	2	Jessica	Johnson	Available
1103	3	Christopher	Davis	Not Available
1104	4	Emily	Wilson	Available
1105	5	Daniel	Thompson	Available

Client Table



ClientID	DecorID	EventID	InviID	FirstName	LastName	Email	Phone
10110	1111	1	101	Robert	Johnson	robertjohnson@example.com	1234567890
10111	1112	1	102	Sarah	Brown	sarahbrown@example.com	9876543210
10112	1113	2	103	Matthew	Anderson	matthewanderson@example.com	4567890123
10113	1114	2	104	Olivia	Taylor	oliviataylor@example.com	7890123456
10114	1115	3	105	Daniel	Wilson	danielwilson@example.com	0123456789

B: Data Dictionary

Booking Table

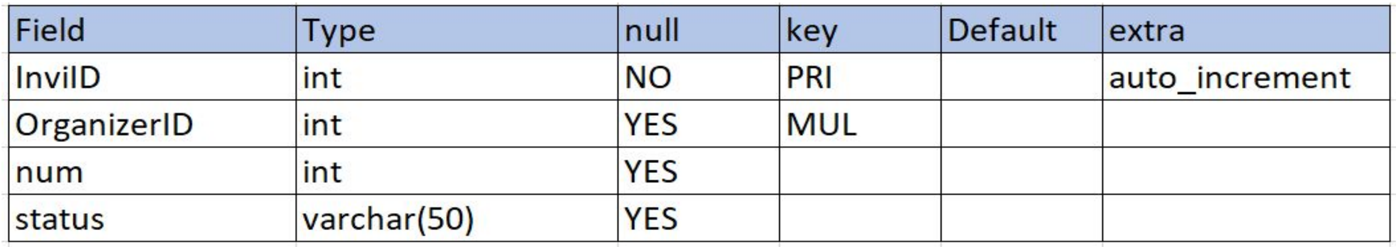
Field	Type	null	key	Default	extra
BookingID	int	NO	PRI	null	auto_increment
EventID	int	NO	MUL	null	
clientName	varchar(50)	NO		null	
bookingDate	date	NO		null	
numOfTickets	int	NO		null	
bookingcol	varchar(45)	NO		null	

Client Table

Field	Type	null	key	Default	extra
ClientID	int	NO	PRI		auto_increment
TableID	int	YES	MUL		
EventID	int	YES	MUL		
InvilD	int	YES	MUL		
FirstName	varchar(50)	YES			
LastName	varchar(50)	YES			
Email	varchar(255)	YES			
Phone	varchar(50)	YES			

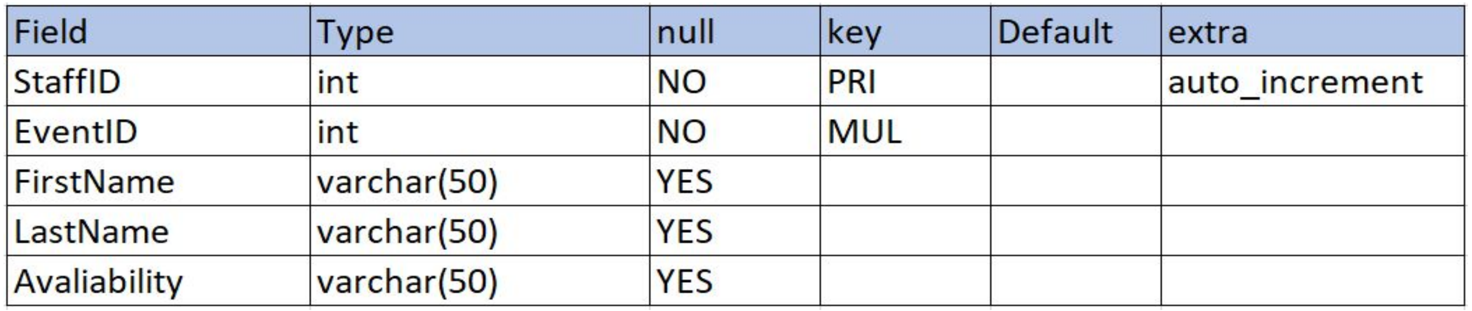
B: Data Dictionary

Invitation Table



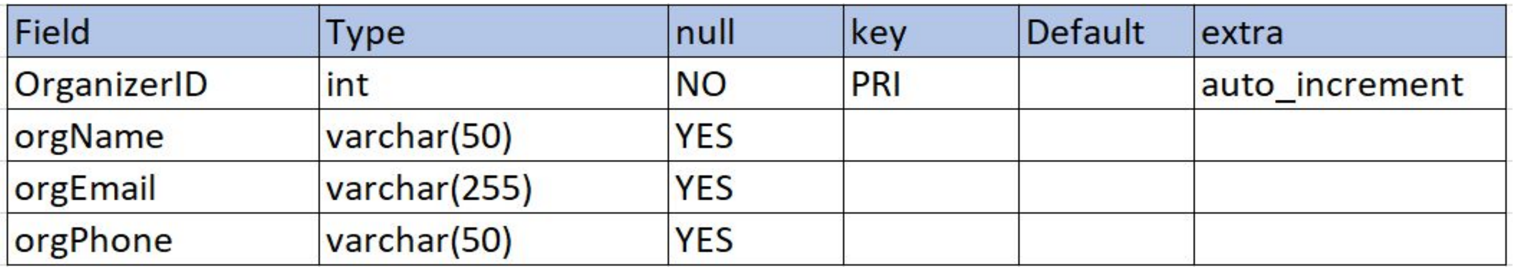
Field	Type	null	key	Default	extra
InvilD	int	NO	PRI		auto_increment
OrganizerID	int	YES	MUL		
num	int	YES			
status	varchar(50)	YES			

Staff Table



Field	Type	null	key	Default	extra
StaffID	int	NO	PRI		auto_increment
EventID	int	NO	MUL		
FirstName	varchar(50)	YES			
LastName	varchar(50)	YES			
Avaliability	varchar(50)	YES			

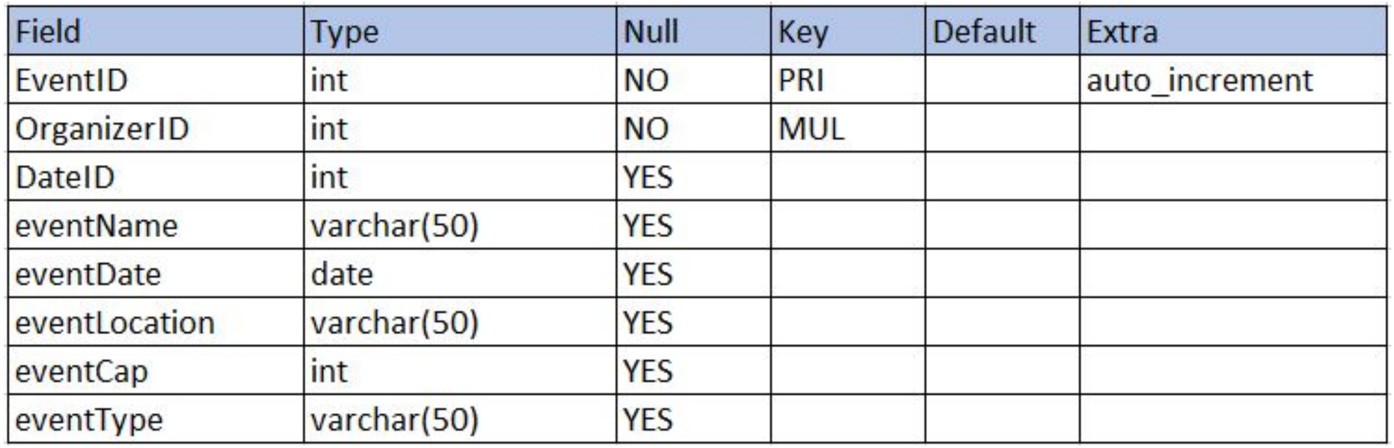
Organizer Table



Field	Type	null	key	Default	extra
OrganizerID	int	NO	PRI		auto_increment
orgName	varchar(50)	YES			
orgEmail	varchar(255)	YES			
orgPhone	varchar(50)	YES			

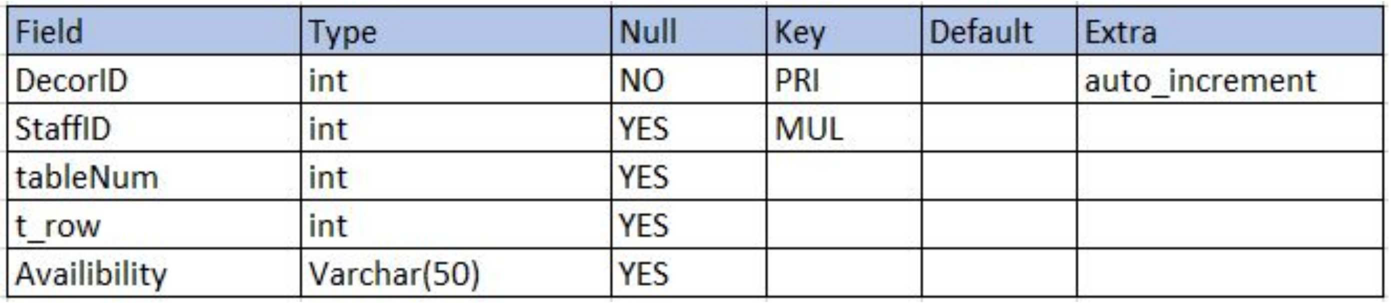
B: Data Dictionary

Event Table



Field	Type	Null	Key	Default	Extra
EventID	int	NO	PRI		auto_increment
OrganizerID	int	NO	MUL		
DateID	int	YES			
eventName	varchar(50)	YES			
eventDate	date	YES			
eventLocation	varchar(50)	YES			
eventCap	int	YES			
eventType	varchar(50)	YES			

Decor Table



Field	Type	Null	Key	Default	Extra
DecorID	int	NO	PRI		auto_increment
StaffID	int	YES	MUL		
tableNum	int	YES			
t_row	int	YES			
Availibility	Varchar(50)	YES			

Chapter 3: Implementation and Loading

CREATE

```
create database EPS;
use EPS;
```

Date Table

```
create table Date(
DateID int primary key auto_increment,
date date not null,
hour time not null
);
```

Stafft Table

```
create table Staff(
StaffID int primary key auto_increment,
EventID int not null,
FirstName varchar(50),
LastName varchar(50),
Avaliability varchar(50)
);
```

Event Table

```
create table Event(
EventID int primary key auto_increment,
Organizer int not null,
DateID int,
eventName varchar(50),
eventDate date,
eventLocation varchar(50),
eventCap int,
eventType varchar(50)
);
```

Booking Table

```
create table Booking(
BookingID int primary key auto_increment,
clientName varchar(50),
bookingDate date,
numOfTickets int
);
```

Organizer Table

```
create table Organizer(
OrganizerID int primary key auto_increment,
orgName varchar(50),
orgEmail varchar(255),
orgPhone varchar(50)
);
```

Database

```
create database EPS;

use EPS;
```

Invitation Table

```
create table Invitation(
InviID int primary key auto_increment,
OrganizerID int,
num int,
status varchar(50)
);
```

Client Table

```
create table Client(
ClientID int primary key auto_increment,
TableID int,
EventID int,
InviID int,
FirstName varchar(50),
LastName varchar(50),
Email varchar(255),
Phone varchar(50)
);
```

Decor Table

```
create table Decor(
TableID int primary key auto_increment,
StaffID int,
tableNum int,
t_row int,
Availability varchar(50)
);
```

Foreign Key

```
alter table staff add constraint event_have_staff foreign key (EventID) references Event (EventID);

alter table event add constraint Organizer_host_event foreign key (OrganizerID) references Organizer (OrganizerID);

alter table booking add constraint Booking_setup_event foreign key (EventID) references Event (EventID);

alter table tables add constraint staff_prepare_tables foreign key (StaffID) references staff (StaffID);

alter table client add constraint Decor_hold_client foreign key (DecorID) references Decor (DecorID);

alter table client add constraint event_contain_client foreign key (EventID) references Event (EventID);

alter table client add constraint invitation_invite_client foreign key (InviID) references invitation (InviID);

alter table invitation add constraint organizer_give_invitaion foreign key (OrganizerID) references Organizer (OrganizerID);
```

Insert INTO Organizer

```
INSERT INTO Organizer (OrganizerID, orgName, orgEmail,
orgPhone) VALUES
(1011, 'John Doe', 'john.doe@email.com', 1234567890),
(1012, 'Jane Smith', 'jane.smith@email.com', 9876543210),
(1013, 'Michael Johnson', 'michael.johnson@email.com',
5551112222),
(1014, 'Emily Davis', 'emily.davis@email.com', 3339994444),
(1015, 'David Wilson', 'david.wilson@email.com',
7778889999),
(1016, 'Sarah Brown', 'sarah.brown@email.com',
5556667777),
(1017, 'Robert White', 'robert.white@email.com',
9998887777),
(1018, 'Jessica Turner', 'jessica.turner@email.com',
1239876543),
(1019, 'William Martinez', 'william.martinez@email.com',
3216549870),
(1020, 'Olivia Adams', 'olivia.adams@email.com',
5552223333);
```

Insert INTO Event

```
INSERT INTO Event (EventID, OrganizerID, eventName,
eventDate, eventLocation, eventcap, eventType)
VALUES
  (1, 1011, 'Music Festival', '2023-11-10', 'City Park', 5000,
'Concert'),
  (2, 1012, 'Charity Gala', '2023-12-05', 'Grand Hotel', 300,
'Fundraiser'),
  (3, 1013, 'Tech Conference', '2024-01-15', 'Convention
Center', 1000, 'Conference'),
  (4, 1014, 'Art Exhibition', '2024-02-20', 'Art Gallery', 150,
'Exhibition'),
  (5, 1015, 'Sports Tournament', '2024-03-12', 'Sports
Complex', 200, 'Sports'),
  (6, 1016, 'Food Festival', '2024-04-25', 'Downtown Square',
10000, 'Festival');
```

Insert INTO Invitation

```
INSERT INTO Invitation (InviID, OrganizerID, num, status)
VALUES
  (101, 1011, 10, 'Pending'),
  (102, 1012, 5, 'Accepted'),
  (103, 1013, 8, 'Declined'),
  (104, 1014, 12, 'Pending'),
  (105, 1015, 3, 'Accepted');
```

Insert INTO Booking

```
INSERT INTO Booking (BookingID, EventID, clientName,
bookingdate, numofTickets)
VALUES
  (1001, 1, 'Emily Johnson', '2023-11-01', 2),
  (1002, 2, 'Jacob Smith', '2023-12-01', 4),
  (1003, 3, 'Sophia Anderson', '2024-01-01', 1),
  (1004, 4, 'Ethan Davis', '2024-02-01', 3),
  (1005, 5, 'Isabella Wilson', '2024-03-01', 2);
```

Insert INTO Staff

```
INSERT INTO Staff (staffID, EventID, FirstName, LastName,
Avaliability)
VALUES
  (1101, 1, 'Andrew', 'Smith', 'Available'),
  (1102, 2, 'Jessica', 'Johnson', 'Available'),
  (1103, 3, 'Christopher', 'Davis', 'Not Available'),
  (1104, 4, 'Emily', 'Wilson', 'Available'),
  (1105, 5, 'Daniel', 'Thompson', 'Available');
```

Insert INTO Client

```
INSERT INTO Client (ClientID, TableID, EventID, InviID,
FirstName, LastName, Email, Phone)
VALUES
  (1, 1111, 1, 101, 'Robert', 'Johnson',
'robertjohnson@example.com', '1234567890'),
  (2, 1112, 1, 102, 'Sarah', 'Brown',
'sarahbrown@example.com', '9876543210'),
  (3, 1113, 2, 103, 'Matthew', 'Anderson',
'matthewanderson@example.com', '4567890123'),
  (4, 1114, 2, 104, 'Olivia', 'Taylor', 'oliviataylor@example.com',
'7890123456'),
  (5, 1115, 3, 105, 'Daniel', 'Wilson',
'danielwilson@example.com', '0123456789');
```

Insert INTO Decor (tables)

```
INSERT INTO Decor (DecorID, StaffID, tableNum, t_row,
Availability)
VALUES
  (1111, 1101, 1, 1, 'Available'),
  (1112, 1101, 2, 1, 'Available'),
  (1113, 1102, 1, 2, 'Not Available'),
  (1114, 1103, 2, 2, 'Available'),
  (1115, 1104, 1, 3, 'Available');
```

Chapter 4: Testing and Evaluation

TWO (2) queries involving relation from two tables.

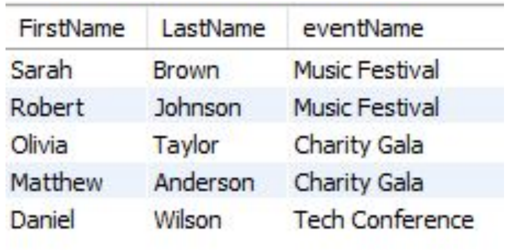
1. Retrieve the event name and the client name.

SELECT Event.eventName, Client.FirstName,

Client.LastName

FROM Event

JOIN Client ON Client.EventID = Event.EventID;



FirstName	LastName	eventName
Sarah	Brown	Music Festival
Robert	Johnson	Music Festival
Olivia	Taylor	Charity Gala
Matthew	Anderson	Charity Gala
Daniel	Wilson	Tech Conference

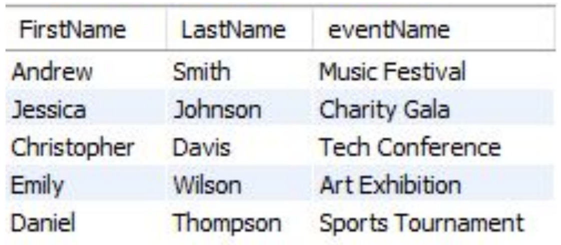
2. List staff members and their corresponding event types.

SELECT Staff.FirstName, Staff.LastName,

Event.eventType

FROM Staff

JOIN Event ON Staff.EventID = Event.EventID;



FirstName	LastName	eventName
Andrew	Smith	Music Festival
Jessica	Johnson	Charity Gala
Christopher	Davis	Tech Conference
Emily	Wilson	Art Exhibition
Daniel	Thompson	Sports Tournament

Queries Involving Aggregate Functions (SUM, COUNT, AVG, MAX, MIN):

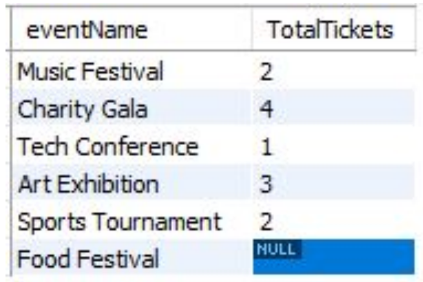
Sum : total number of tickets booked for each event

SELECT Event.eventName, SUM(Booking.numOfTickets) AS TotalTickets

FROM Event

LEFT JOIN Booking ON Event.EventID = Booking.EventID

GROUP BY Event.eventName;



eventName	TotalTickets
Music Festival	2
Charity Gala	4
Tech Conference	1
Art Exhibition	3
Sports Tournament	2
Food Festival	NULL

Count : total number of clients who have booked tickets for a specific event

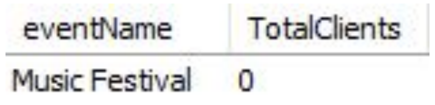
SELECT Event.eventName, COUNT(Client.ClientID) AS TotalClients

FROM Event

LEFT JOIN Booking ON Event.EventID = Booking.EventID

LEFT JOIN Client ON Booking.EventID = Client.ClientID

WHERE Event.EventID = 1 ;

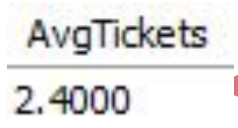


eventName	TotalClients
Music Festival	0

AVG: average number of tickets

SELECT AVG(Booking.numOfTickets) AS AvgTickets

FROM Booking;



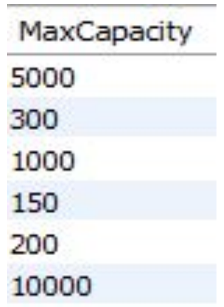
AvgTickets
2.4000

MAX : maximum of max capacity for each event

SELECT MAX(Event.eventCap) AS MaxCapacity

FROM Event

GROUP BY Event.eventName;

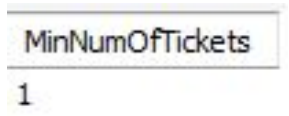


MaxCapacity
5000
300
1000
150
200
10000

MIN : minimum number of tickets

SELECT MIN(numOfTickets) AS MinNumOfTickets

FROM Booking;



MinNumOfTickets
1

Two (2) Queries Involving Complicated SELECTs and JOIN from Three or More Tables

1. Find clients who have attended events organized by a specific organizer.

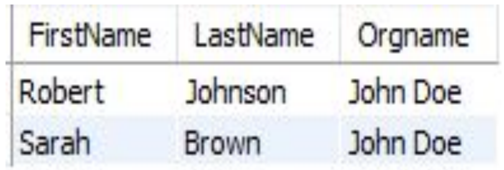
SELECT Client.FirstName, Client.LastName,Orgname

FROM Client

JOIN Event ON Client.EventID = Event.EventID

JOIN Organizer ON Event.OrganizerID = Organizer.OrganizerID

WHERE Organizer.orgName = 'John Doe';



FirstName	LastName	Orgname
Robert	Johnson	John Doe
Sarah	Brown	John Doe

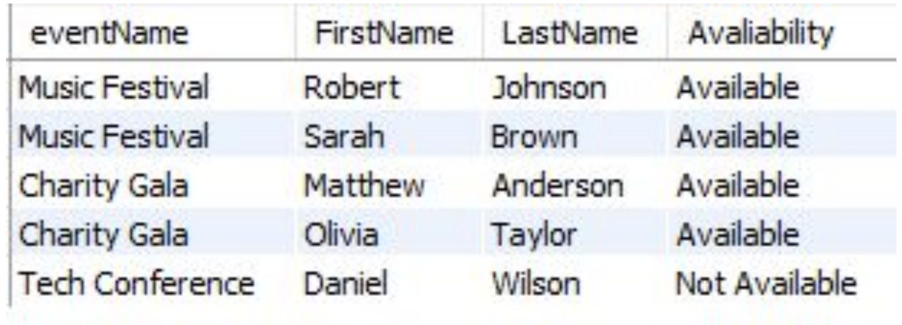
2. Retrieve event details, client names, and staff availability for clients attending specific events.

SELECT Event.eventName, Client.FirstName, Client.LastName, Staff.Avaliability

FROM Event

JOIN Client ON Event.EventID = Client.EventID

LEFT JOIN Staff ON Event.EventID = Staff.EventID;

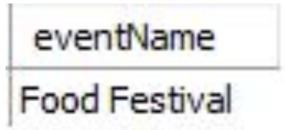


eventName	FirstName	LastName	Avaliability
Music Festival	Robert	Johnson	Available
Music Festival	Sarah	Brown	Available
Charity Gala	Matthew	Anderson	Available
Charity Gala	Olivia	Taylor	Available
Tech Conference	Daniel	Wilson	Not Available

One (1) Query Involving Joins with the NOT Keyword in the Relations:

1. Find events that have no bookings yet.

```
SELECT Event.eventName
FROM Event
LEFT JOIN Booking ON Event.EventID =
Booking.EventID
WHERE Booking.BookingID IS NULL;
```

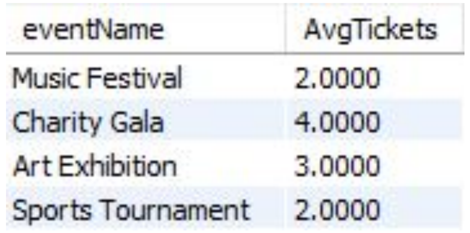


eventName
Food Festival

Two (2) Queries Involving GROUP BY and HAVING Functions:

1. List events and their average ticket bookings where the average bookings are greater than 1

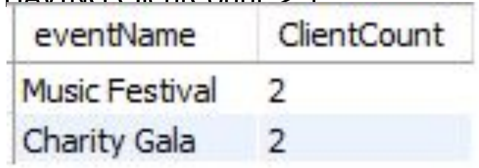
```
SELECT Event.eventName, AVG(Booking.numOfTickets) AS AvgTickets
FROM Event
LEFT JOIN Booking ON Event.EventID = Booking.EventID
GROUP BY Event.eventName
HAVING AvgTickets > 1;
```



eventName	AvgTickets
Music Festival	2.0000
Charity Gala	4.0000
Art Exhibition	3.0000
Sports Tournament	2.0000

2. Count the number of clients for each event and retrieve events with more than 1 clients.

```
SELECT Event.eventName, COUNT(Client.ClientID) AS ClientCount
FROM Event
LEFT JOIN Client ON Event.EventID = Client.EventID
GROUP BY Event.eventName
HAVING ClientCount > 1;
```

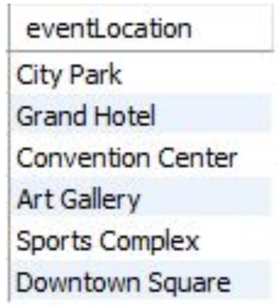


eventName	ClientCount
Music Festival	2
Charity Gala	2

Two (2) Queries Requiring the Use of the DISTINCT and ALL Keywords:

1. Query to retrieve a list of distinct event locations:

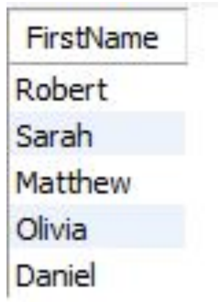
```
SELECT DISTINCT Event.eventLocation
FROM Event;
```



eventLocation
City Park
Grand Hotel
Convention Center
Art Gallery
Sports Complex
Downtown Square

2. Query to get a list of all client first names without duplicates:

```
SELECT ALL Client.FirstName
FROM Client;
```



FirstName
Robert
Sarah
Matthew
Olivia
Daniel

In Conclusion

In Conclusion

The proposed database system for Event Planning Services (EPS) plays a important role in streamlining the company's event management processes, enhancing communication, and providing valuable insights for data-driven decision-making.

Future Enhancements:

1. Visual Enchantment
2. Inventory Management
3. Security Management

Thank you