CS 238 Databases Project

An Event Scheduling System

Taught by Professor Dr. Dan Lin

Matthew Harrison
Gary Steelman
Dale Twombly

Table of Contents

1. Problem Statement

- A system needs to be created to allow the organization Well*Life to set up personal health-oriented events. There are two types of events: Classes on how to improve everyday your health in everyday life or Screenings to check physical health, like blood or cholesterol tests. The events are administered by a group of people called staff and attended by a group of people called participants. The following system is proposed for ease of organization, smoothness of execution, and clarity of documentation for these events.

- For more information on Well*Life, see Appendix I.

2. System Requirements

How the system will be used (a typical interaction)

- Well*Life will receive a request from a company or organization to set up an event. Well*Life will be supplied the desired event's information (date, time, location, etc.) by the organization. Well*Life will then locate appropriate staff for the event and have the staff provide their information (name, e-mail, phone number, etc.). Once appropriate staff for an event have been located, the event will be opened to enrollment by participants. The participants will be required to submit their information (name, e-mail, phone number, etc.) to Well*Life. Once a participant submits their information, it will be saved to the database by the database management system (DBMS). Event staff will be able to generate rosters, schedules, etc. for an event using the DBMS. After the event concludes, the system will allow a summary of it to be printed and sent to the sponsoring company/organization. Once the sponsoring company/organization receives the summary of the event, the DBMS deletes all participants' information. The table below lists the information that will be submitted by each party for the event.

Required Information (by Group)			
		Participant	
Company or Organization	**Staff**	**Class**	**Screening**
Name of Company or Organization	First Name	First Name	First Name
Street Address	Last Name	Last Name	Last Name
City	Email*	Street Address	Street Address
State	Street Address	City	City
Zip Code	City	State	State
Contact E-mail*	State/Terriroty	Zip Code	Zip Code
Phone Number 1	Zip	E-Mail*	E-Mail*
Phone Number 2	Country	Phone Number 1	Phone Number 1
Event Date	Phone Number 1	Phone Number 2	Phone Number 2
Event Time	Phone Number 2	BCBS Number	Screening Time
Event Location*	Password	BCBS Suffix	Comments
Number of Expected Participants	Comments	Date of Birth	
Recommended Staff		Comments	
Event Name			
Event Type			
Event Subtype			
Comments			

*Required correct input for proper database operation.

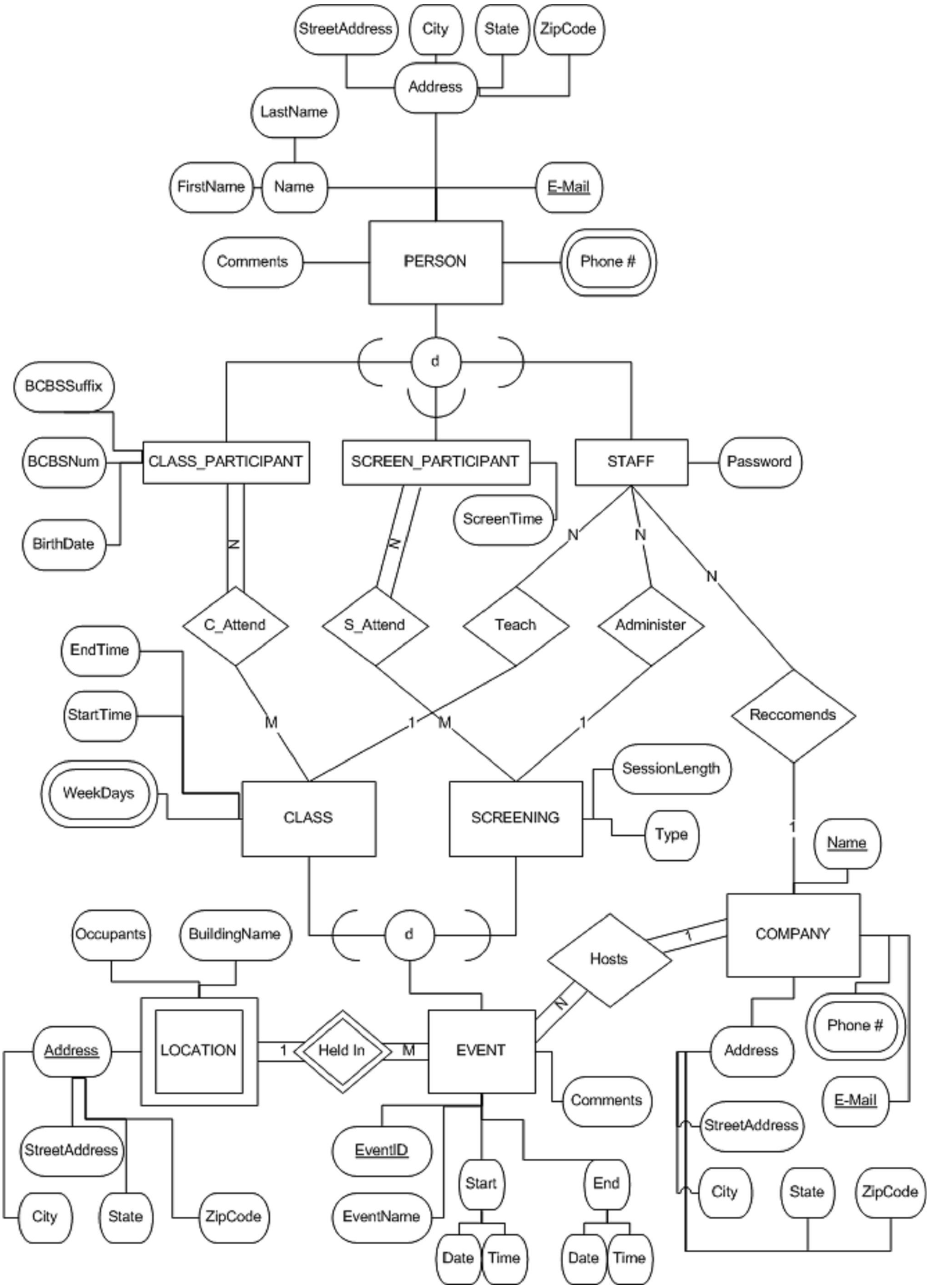
- For Company/Organization
  - Event Subtype is what kind of Event is to be generation: Class or Screening.
  - Recommended Staff is a list of names (first and last) of people the company recommends to act as staff for the Event.

    - For Staff, the system keeps a password so they can return again later to administer more events.
    - For some Classes, a BCBS (BlueCross BlueShield) Number and Suffix can be submitted (and possibly required) for that class, as specified when the Class is generated.
    - For Screening, a screening time must be chosen within the duration of the Event. This is the small window of time it will take to actually screen a person.

- **What the system does or does not do**

System does...	System does not...
Allow Well*Life to set up events with the information supplied by a company or organization.	Allow participants to sign up for multiple sessions in the same event.
Allow participants and staff to sign up for classes or screenings.	Allow staff to commit to staffing multiple events concurrently.
Allow Well*Life to select staff (from those who applied) for an event by Well*Life's criterion.	Have usernames or passwords for participants.
Require participants to have a valid name, phone number, and class or screening selection.	Allow participants to edit their submitted information, cancel appointments, or move appointments within the DBMS.
Keep a record of who staffed which event after the event concludes.	Keep a record of participants for each event past sending a summary to the sponsoring company/organization.
	Store medical records of any kind.

3. **Entity-Relationship Diagram**



4. Functional Requirements

- **The database must be able to retrieve...**

  - all participants' information who are attending a specific screening or class.
    - Input: Event ID.
    - Output: List of all participants attending the screening or class.
    - Database accesses: Person, Event.
    - Database modifies: None.

  - a single participant's information who is attending a specific screening or class.
    - Input:Event ID, Person's e-mail.
    - Output: Information on the participant specified.
    - Database accesses: Person, Event.
    - Database modifies: None.

  - all information on staff administering or teaching a specific event.
    - Input: Event ID.
    - Output: List of all staff administering or teaching the specified event.
    - Database accesses: Person, Event.
    - Database modifies: None.

  - a single staff member's information who is administering or teaching a specific event.
    - Input:Event ID, Staff's e-mail.
    - Output: All information on the staff member specified.
    - Database accesses: Person, Event.
    - Database modifies: None.

  - all times and dates associated with a specific class.
    - Input: Event ID.
    - Output: A schedule for the event including starting and ending time and starting and ending dates.
    - Database accesses: Event.
    - Database modifies: None.

  - all times and dates associated with a specific screening.
    - Input: Event ID.
    - Output: A schedule for the screening including staring and ending time, starting and ending dates, and attendees for each session.
    - Database accesses: Person, Event.
    - Database modifies: None.

  - all information about a specific company.
    - Input: Company e-mail.
    - Output: All information stored on a company including related events and staff recommended by the company.
    - Database accesses: Company, Event.
    - Database modifies: None.

  - all information on a specific event including all participants, staff, dates, times, location, and companies associated with the event.
    - Input: Event ID.
    - Output: All information on an event including starting and ending times, dates, location, related company, staff, and participants.
    - Database accesses: Event, Person, Company.

        - all events associated with a specific company.
            - Input: Company's e-mail.
            - Output: Information on all events associated with the company.
            - Database accesses: Company, Event.
            - Database modifies: None.

    - **The database must be able to add...**

        - a person in the database.
            - Input: All required information for a person (see Section 3).
            - Output: A message confirming successful or unsuccessful addition of the person to the database.
            - Database accesses: Person, Event.
            - Database modifies: Person.

        - an event in the database.
            - Input: All required information for an event (see Section 3).
            - Output: A message confirming successful or unsuccessful addition of the event to the database.
            - Database accesses: Event, Company.
            - Database modifies: Event, Company.

        - a company in the database.
            - Input: All required information for a company (see Section 3).
            - Output: A message confirming successful or unsuccessful addition of the event to the database.
            - Database accesses: Company.
            - Database modifies: Company.

    - **The database must be able to delete...**

        - any person, event, or company in the database.
            - Input: Key information for the person, event, or company.
            - Output: A message confirming successful or unsuccessful deletion of the person, event, or company from the database.
            - Database accesses: Entity associated with the key information provided.
            - Database modifies: Entity associated with the key information provided.

    - **The database must be able to modify...**

        - any staff person, event, or company in the database.
            - Input: All of the required information for an Add operation for the specified Entity.
            - Output: A message confirming successful or unsuccessful modification of the tuple to the database.
            - Database accesses: Entity associated with the Entity type provided.
            - Database modifies: Entity associated with the key information provided.
            - Note: The modify function requires the same inputs as the Add function because the modify function is a delete of the current record followed by an insertion of the new record.

The system requirements can be collectively fulfilled by combining the above operations using relational algebra to produce all required output.

5. Logical Database Design

- The following layout is used:

ENTITY_NAME		
Entity_Attribute	Data_Type, Constraints	Description

Entities:

EVENT		
Event_ID	int, unique, not NULL	Arbitrary for event
Start_Date	int, >= 0, not NULL	The date the event begins
End_Date	int, >= Start_Date, not NULL	The date the event ends
Start_Time	int, not NULL	The time the event begins
End_Time	int, >= Start_Time, not NULL	The time the event ends
Comments	string	Any comments submitted by the user for the event
E_Type	bool, not NULL	true if event is a class false if event is screening
S_Type	string	Type of the screening. NULL for classes.
Session_Length	int	Length of session for screening. NULL for classes.

LOCATION		
Event_ID (FK)	int, unique	Arbitrary for event
Address	string, not NULL	The street address for the event
City	string, not NULL	The event's city
State	char (2), not NULL	The event's state
Zip	int	The event's zip code
Occupants	int, >= 0	Maximum number of occupants allowed
Name	string, not NULL	The event name

CLASS_DATES		
Event_ID (FK)	int, unique	Arbitrary for event
Days	int	The days an event is in session

COMPANY		
Name (CK)	string, unique, not NULL	The company's name
Email (CK)	string, unique, not NULL	The company's e-mail
Address	string	The street address
City	string	The company's city
State	char (2)	The company's state

Zip	int	The company's zip code

COMPANY_PHONE		
Email (FK)	string, unique, not NULL	The company's e-mail
Number	string	The company's phone number

CLASS_PARTICIPANT		
Fname	string	Person's first name
Lname	string	Person's last name
Email	string, unique, not NULL	Person's e-mail
Address	string	The person's street address
City	string	The person's city
State	char (2)	The person's state
Zip	int	The person's zip code
Comments	string	Any comments submitted by user when signing up for a class
BCBSNum	int	BlueCross BlueShield Number (if applicable)
BCBSSuffix	int	BlueCross BlueShield Suffix (if applicable)
Birth_Date	string	The person's birthdate

SCREEN_PARTICIPANT		
Fname	string	Person's first name
Lname	string	Person's last name
Email	string, unique, not NULL	Person's e-mail
Address	string	The person's street address
City	string	The person's city
State	char (2)	The person's state
Zip	int	The person's zip code
Comments	string	Any comments submitted by user when signing up for a class
Time	int	The person's screening time

STAFF		
Fname	string	Person's first name
Lname	string	Person's last name
Email	string, unique, not NULL	Person's e-mail
Address	string	The person's street address
City	string	The person's city
State	char (2)	The person's state

Zip	int	The person's zip code
Comments	string	Any comments submitted by user when signing up for a class
Password	string, not NULL	The staff member's password for login

PERSON_PHONE		
Email (FK)	string, unique, not NULL	Person's e-mail
Number	string	The person's phone number

Relationships:

HOSTS		
Email (FK)	string, unique, not NULL	The company's e-mail
Event_ID (FK)	int, unique, not NULL	Arbitrary for event

RECOMMENDS		
CEmail (FK)	string, unique, not NULL	The company's e-mail
SEmail (FK)	string, unique, not NULL	The staff member's e-mail

HELD_IN		
Event_ID (FK)	int, unique, not NULL	Arbitrary for event
Address (FK)	string, not NULL	The street address for the event

C_ATTEND		
Email (FK)	string, unique, not NULL	The person's e-mail
Event_ID (FK)	int, unique, not NULL	Arbitrary for event

S_ATTEND		
Email (FK)	string, unique, not NULL	The person's e-mail
Event_ID (FK)	int, unique, not NULL	Arbitrary for event

TEACH		
Email (FK)	string, unique, not NULL	The staff's e-mail
Event_ID (FK)	int, unique, not NULL	Arbitrary for event

ADMINISTER		
Email (FK)	string, unique, not NULL	The staff's e-mail
Event_ID (FK)	int, unique, not NULL	Arbitrary for event

6. Design of Application Programs

All function parameters of the same names as attributes in the relational tables from the previous section are of the same data type as the attributes in the relational tables. Function parameters of different names are explained with the function.

- **The following functions perform add, delete, and modify of records in the database as described in the functional requirements section.**

$Recommends is an array of Emails representing any staff members that a company recommends.

```
AddCompany( $Name, $Email, $Address, $City, $State, $Zip, $Phone, $Recommends )
  Insert into COMPANY
  Values( $Name, $Email, $Address, $City, $State, $Zip )

  For each number in $Phone
    Insert into COMPANY_PHONE
    Values( $Email, $Number )

  For each email in $Recommends
    Insert into RECOMMENDS
    Values( $Email, $Recommends[i] )
End Function

DeleteCompany( $Email )
  Remove from COMPANY
  Values( $Email )

  Remove from RECOMMENDS
  Values( $Email )
End Function

ModCompany( $Name, $Email, $Address, $City, $State, $Zip, $Phone )
  DeleteCompany( $Email )
  AddCompany( $Name, $Email, $Address, $City, $State, $Zip, $Phone )
End Function

AddLocation( $Name, $Address, $City, $State, $Zip, $Occupants, $Event_ID )
  Insert into LOCATION
  Values( $Name, $Address, $City, $State, $Zip, $Occupants, $Event_ID )

  Insert into HELD_IN
  Values( $Event_ID, $Address )
End Function

DeleteLocation( $Address, $Event_ID )
  Remove from LOCATION
  Values( $Address, $Event_ID )

  Remove from EVENT
  Values( $Event_ID )
End Function

ModLocation( $Name, $Address, $City, $State, $Zip, $Occupants, $Event_ID )
  DeleteLocation( $Address, $Event_ID )
  AddLocation( $Name, $Address, $City, $State, $Zip, $Occupants, $Event_ID )
End Function
```

$PType is a flag to indicate the type of person being added to the database. It is a string with values 'Class', 'Screen', or 'Staff'.
$SType is a flag to indicate the type of staff a staff member is. If the staff member is to administer a screening, it is 's'. If the staff member is to teach a class, it is 'c'.

```
AddPerson( $Fname, $LName, $Email, $Address, $City, $State, $Zip,
           $Comments, $BCBSNum, $BCBSSuffix, $Birth_Date, $Time,
           $Password, $PType, $Event_ID, $SType )
  If $PType = Class
    Insert into CLASS_PARTICIPANT
    Values( $Fname, $LName, $Email, $Address, $City, $State, $Zip,
            $Comments, $BCBSNum, $BCBSSuffix, $Birth_Date )
    Insert into C_ATTEND
    Values( $Email, $Event_ID )

  Else if $PType = Screen
    Insert into SCREEN_PARTICIPANT
    Values( $Fname, $LName, $Email, $Address, $City, $State, $Zip,
            $Comments, $Time )
    Insert into S_ATTEND
    Values( $Email, $Event_ID )

  Else if $PType = Staff
    Insert into STAFF
    Values( $Fname, $LName, $Email, $Address, $City, $State, $Zip,
            $Comments, $Password )
    If $SType = 'c'
      Insert into TEACH
      Values( $Email, $Event_ID )

    Else
      Insert into ADMINISTER
      Values( $Email, $Event_ID )

    End if

  End if
End Function

DeletePerson( $Email )
  Remove from CLASS_PARTICIPANT
  Values( $Email )

  Remove from SCREEN_PARTICIPANT
  Values( $Email )

  Remove from STAFF
  Values( $Email )
End Function
```

$PType is a flag to indicated the type of person being added to the database. It is a string with values 'Class', 'Screen', or 'Staff'.

```
ModPerson( $Fname, $LName, $Email, $Address, $City, $State, $Zip,
           $Comments, $BCBSNum, $BCBSSuffix, $Birth_Date, $Time,
           $Password, $PType )

  DeletePerson( $Email )
```

```
   AddPerson( $Fname, $LName, $Email, $Address, $City, $State, $Zip,
           $Comments, $BCBSNum, $BCBSSuffix, $Birth_Date, $Time,
           $Password, $PType )
  End Function

  AddEvent( $Event_ID, $SDate, $STime, $EDate, $ETime, $Comments, $ClassOrScreening,
      $Type, $SessionLength, $Weekdays, $Email )
   If $ClassOrSceening = class
    Insert into EVENT
    Values($Event_ID, $SDate, $STime, $EDate, $ETime, $Comments, $ClassOrScreening,
   NULL, NULL, $Location)
    Insert into TEACH
    Values( $Event_ID, $Email )

    For each string in $Weekday
     Insert into CLASS_DAYS
     Values( $Event_ID, $Weekdays[Day] )

   Else
    Insert into EVENT
    Values( $EventID, $SDate, $STime, $EDate, $ETime, $Comments, $ClassOrScreening,
     $Type, $SessionLength, $Weekdays)
    Insert into ADMINISTER
    Values( $Event_ID, $Email )

   End if
  End Function

  DeleteEvent($EventID)
   Remove from EVENT
   Values($EventID)

   Remove from CLASS_DAYS
   Values ($EventID)
 End Function

  ModEvent($EventID, $SDate, $STime, $EDate, $ETime, $Comments, $ClassOrScreening,
      $type, $SessionLenth,$Weekdays, $Location)

   DeleteEvent($EventID)

   AddPerson($EventID, $SDate, $STime, $EDate, $ETime, $Comments, $ClassOrScreening,
     $type, $SessionLenth,$Weekdays, $Location)
  End Function
```

- **The following functions perform simple queries on the database as described in the functional requirements section.**

```
  AllPeople( $Event_ID )
    $Participants=Select from CLASS_PARTICIPANTS, C_ATTEND
    Values( $Event_ID )

    $Participants+=Select from SCREEN_PARTICIPANTS, S_ATTEND
    Values( $Event_ID )
  End Function
```

$EType is the type of event to retrieve a participant for.

```
    SinglePerson( $Event_ID, $Email, $EType)
      If $EType = 'c'
        Select from CLASS_PARTICIPANTS, C_ATTEND
        Values( $Event_ID, $Email )

      Else
        Select from SCREEN_PARTICIPANTS, S_ATTEND
        Values( $Event_ID, $Email )

      End if
    End Function
```

$EType is the type of event to retrieve all staff for.

```
    AllProctor( $Event_ID )
        $Staffers=Select from STAFF, TEACH
        Values( $Event_ID )

        $Staffers+=Select from STAFF, TEACH
        Values( $Event_ID )

    End Function
```

$EType is the type of event to retrieve a staff member for.

```
    SingleProctor( $Event_ID, $Email, $EType )
      If $Etype='c'
        Select from STAFF, TEACH
        Values( $Event_ID, $Email )

      Else
        Select from STAFF, TEACH
        Values( $Event_ID, $Email )

      End if
    End Function

    AllClassTimes( $Event_ID )
      Select from EVENT
      Values( $Event_ID )
    End Function

    CompanyHost( $Email )
      Select from HOSTS
      Values( $Email )
    End Function

    AboutEvent  ( $Event_ID )
      Select form EVENT
      Values( $Event_ID )
    End Function

    AboutCompany( $Email )
      Select form COMPANY
      Values( $Email )
    End Function

    AboutScreening ( $Event_ID )
      Select form Event
```

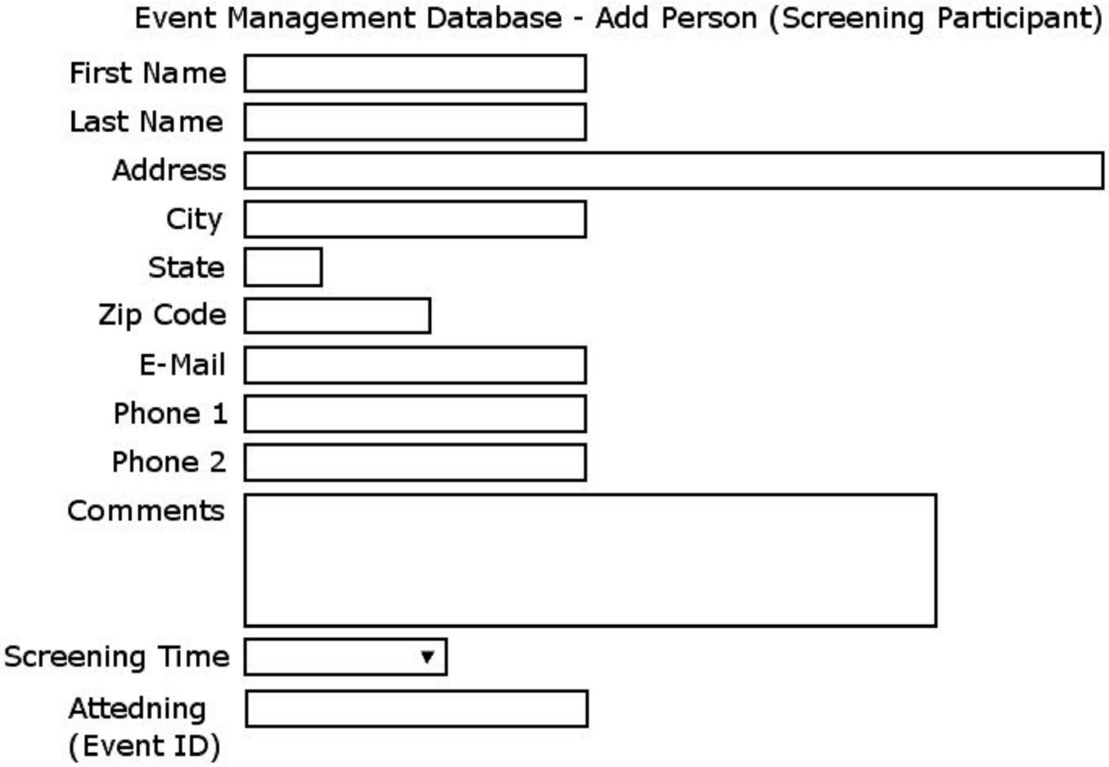
```
  Where $ClassOrScreening = Screening
  Values( $Event_ID )
End Function

AboutClass( $Event_ID )
  Select form Event
  Where $ClassOrScreening = Class
  Values( $Event_ID )
End Function
```

- **The following functions generate statistical reports for the database. Their exact function is described preceding each definition.**

```
CountStaff()
  $totalStaff = Select count( $Email ) from STAFF
 return $totalStaff
End Function

 CountNonStaff()
  $totalNonStaff = Select count( $Email ) from CLASS_PARTICIPANT
    + Select count( $Email ) from SCREENING_PARTICIPANT
  return $totalNonStaff
End Function

CountEvent()
  $totalEvent = Select count( $Event_ID ) from EVENT
  return $totalEvent
End Function

CountCompany()
  $totalCompany = Select count( $Email ) from COMPANY
  return $totalCompany
End Function

MaxPeoplePerEvent()
  $MaxPeople = Select max(list) from CLASS_PARTICIPANT, SCREEN_PARTICIPANT, STAFF
  Group by $Event_ID
  return $MaxPeople
End Function

MinPeoplePerEvent()
  $MinPeople = Select min(list) from CLASS_PARTICIPANT, SCREEN_PARTICIPANT, STAFF
  Group by $Event_ID
  return $MinPeople
End Function

AvgPeoplePerEvent()
  $AvgPeople = Select avg(list) from CLASS_PARTICIPANT, SCREEN_PARTICIPANT, STAFF
  Group by $Event_ID
  return $AvgPeople
End Function
```

7. Interface Design

Screens that add to database:



Event Management Database - Add Person (Screening Participant)

First Name

Last Name

Address

City

State

Zip Code

E-Mail

Phone 1

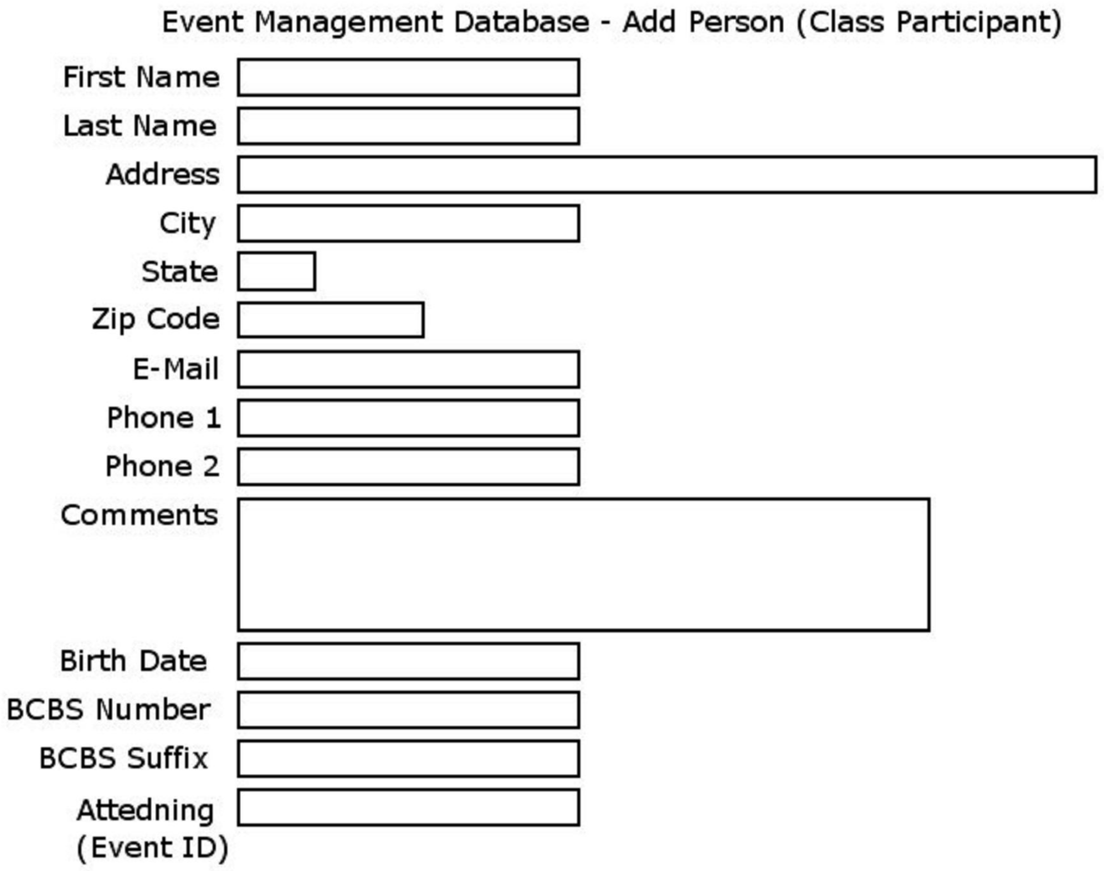
Phone 2

Comments

Screening Time ▼

Attedning (Event ID)

The above screen is seen when adding a screening particpant to the database. All attributes for a screening participant will be added to the SCREEN_PARTICIPANT and S_ATTEND tables.



Event Management Database - Add Person (Class Participant)

First Name

Last Name

Address

City

State

Zip Code

E-Mail

Phone 1

Phone 2

Comments

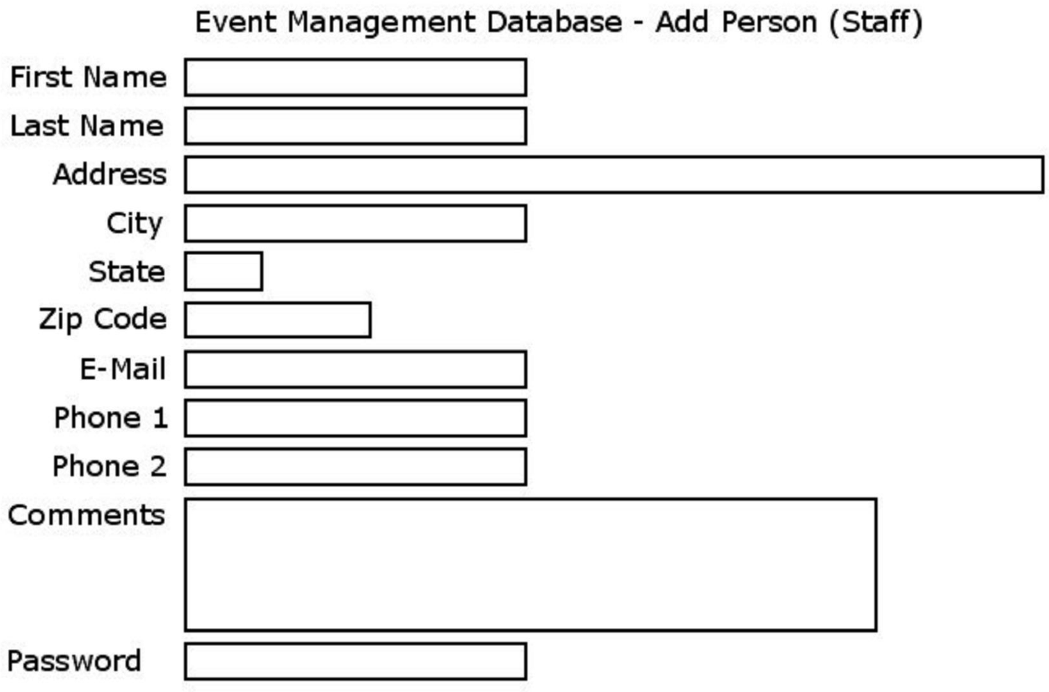
Birth Date

BCBS Number

BCBS Suffix

Attedning (Event ID)

The above screen is seen when adding a class particpant to the database. All attributes for a class participant will be added to the CLASS_PARTICIPANT and C_ATTEND tables.



Event Management Database - Add Person (Staff)

First Name

Last Name

Address

City

State

Zip Code

E-Mail

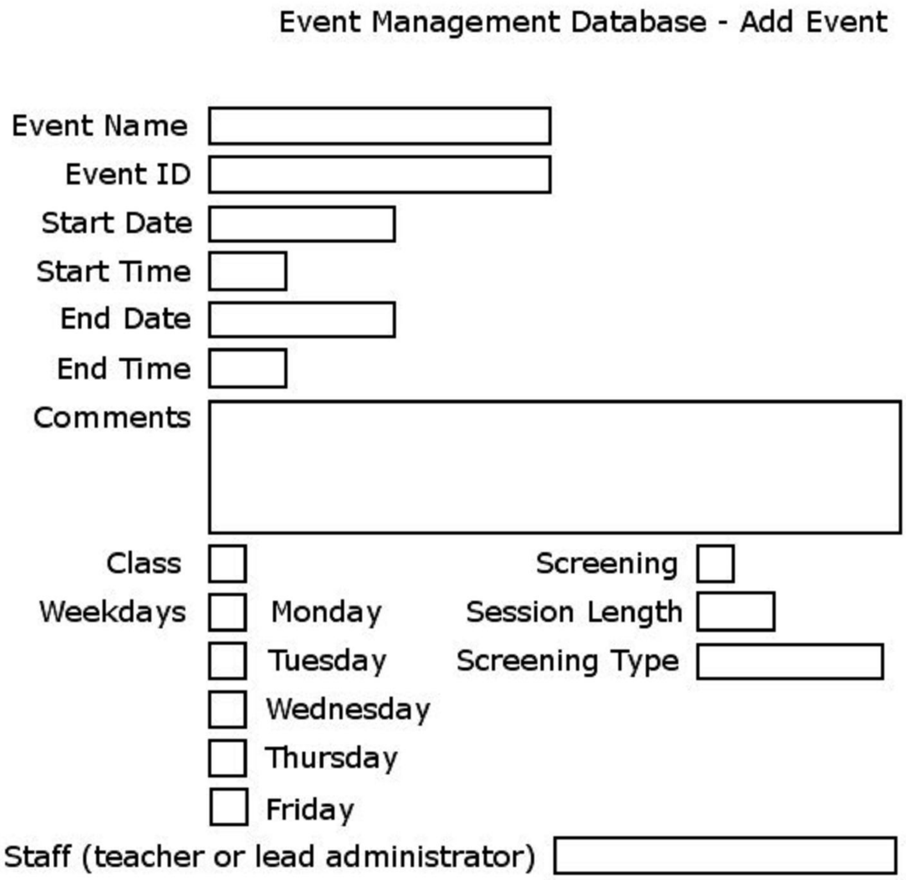
Phone 1

Phone 2

Comments

Password

The above screen is seen when adding a staff member to the database. All attributes for a staff member will be added to the STAFF table (the TEACH and ADMINISTER tables are updated when an event is added to the database).



Event Management Database - Add Event

Event Name

Event ID

Start Date

Start Time

End Date

End Time

Comments

Class □ Screening □

Weekdays □ Monday Session Length

□ Tuesday Screening Type

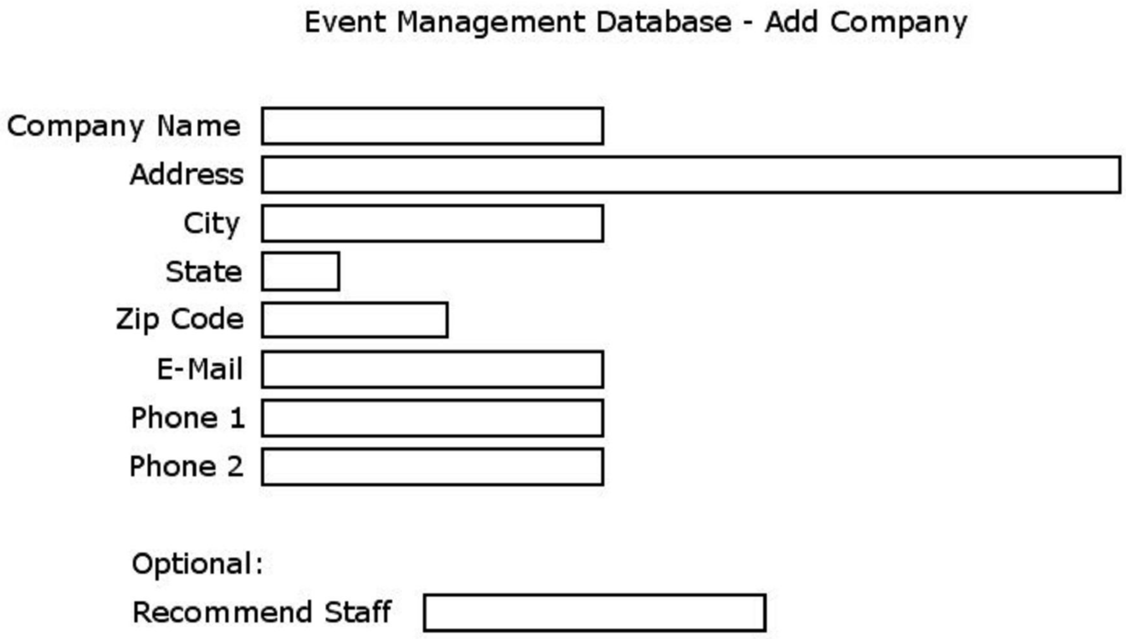
□ Wednesday

□ Thursday

□ Friday

Staff (teacher or lead administrator)

The above screen is seen when adding an event (either class or screening) to the database. All attributes for an event will be added to the CLASS/SCREENING and TEACH/ADMINISTER (depending on the check box selected) tables.



Event Management Database - Add Company

Company Name

Address

City

State

Zip Code

E-Mail

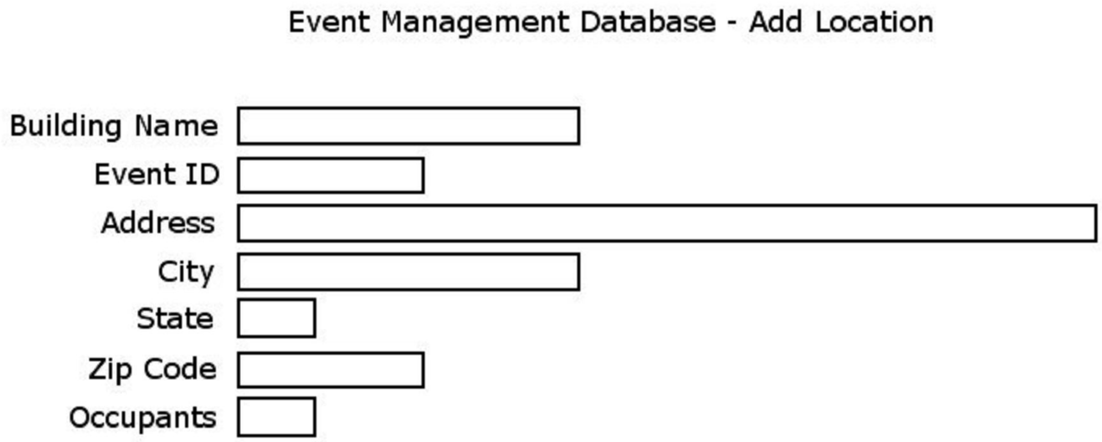
Phone 1

Phone 2

Optional:

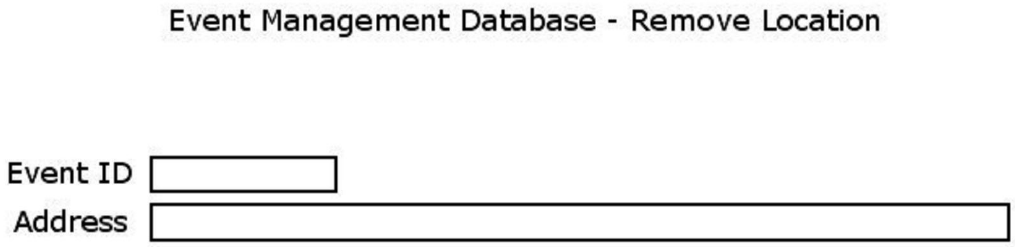
Recommend Staff

The above screen is seen when adding a company to the database. All attributes for a company will be added to the COMPANY and RECCOMENDS tables.



The above screen is seen when adding a location to the database. All attributes for a location will be added to the LOCATION and HELD_IN tables.

Screens that remove from database:



The above screen is seen when removing a location from the database. All attributes for a location will be removed from the LOCATION and HELD_IN tables.

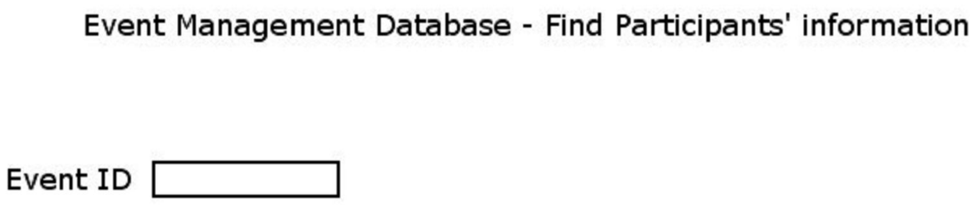
NOTE: The screens seen when removing a class/screening, company, screening participant/class participant are very similar to screen above. See the descriptions below to learn which database relations and attributes in that entity that are being modified for each remove.

The screen for removing a class or screening takes in the Event_ID, removes all attributes for the corresponding event from the CLASS/SCREENING and TEACH/ADMINISTER tables.

The screen for removing a company takes in the company's e-mail, removes all attributes for the corresponding company from COMPANY and RECCOMENDS tables.

The screen for removing a screening participant or class participant takes in the person's e-mail, removes all attributes for the corresponding person from CLASS_PARTICIPANT/SCREEN_PARTICIPANT and C_ATTEND/S_ATTEND tables.

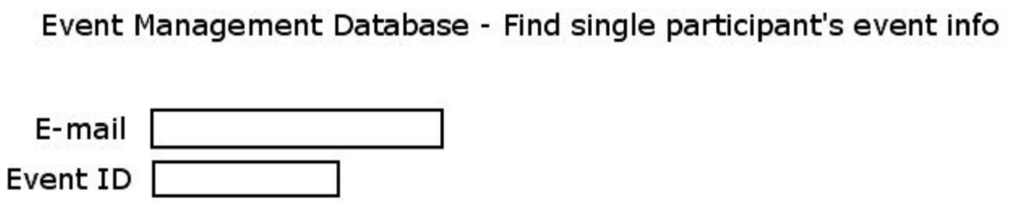
Screens that perform simple queries:



Event Management Database - Find Participants' information

Event ID

The above screen is seen when the user is requesting information about the participants in a given event. All participants attending the screening or class are output.

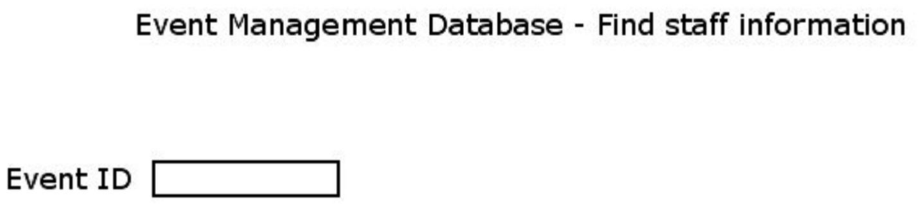


Event Management Database - Find single participant's event info

E-mail

Event ID

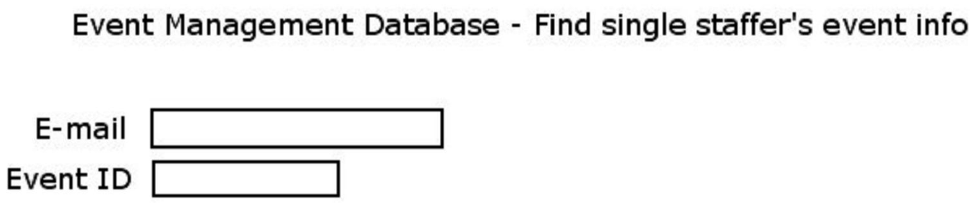
The above screen is seen when the user is requesting information about a single staffer in a given event. All of the selected staffer's info is output.



Event Management Database - Find staff information

Event ID

The above screen is seen when the user is requesting information about the staff in a given event. A list of staff members adminstering the screening/teaching the class are output.

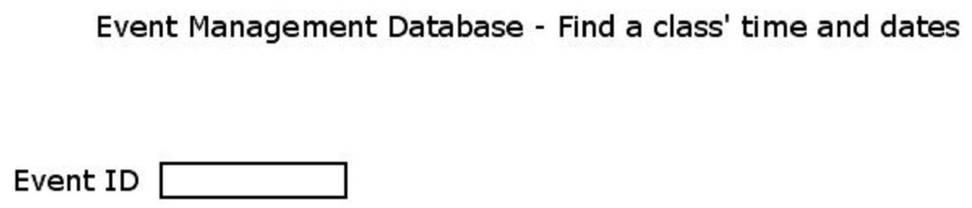


Event Management Database - Find single staffer's event info

E-mail

Event ID

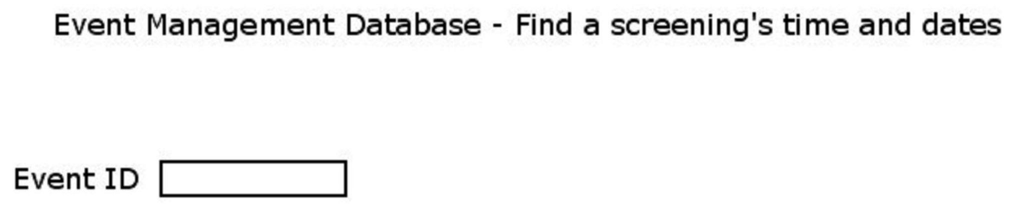
The above screen is seen when the user is requesting information about a single staffer in a given event. All information about the given staff member is output.



Event Management Database - Find a class' time and dates

Event ID

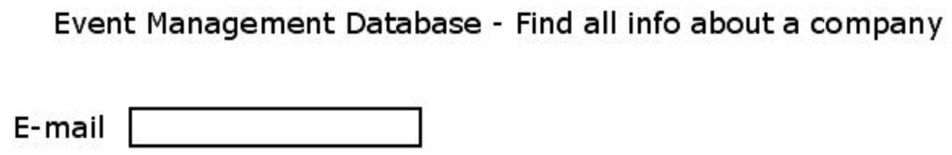
The above screen is seen when the user is requesting times and dates associated with a given class. All of the selected class' dates and times are output.



Event Management Database - Find a screening's time and dates

Event ID

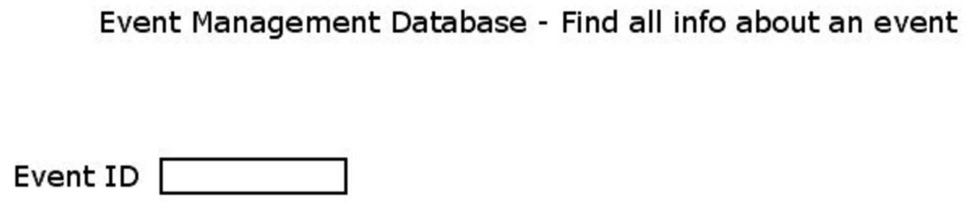
The above screen is seen when the user is requesting times and dates associated with a given screening. All of the selected screening's dates and times are output.



Event Management Database - Find all info about a company

E-mail

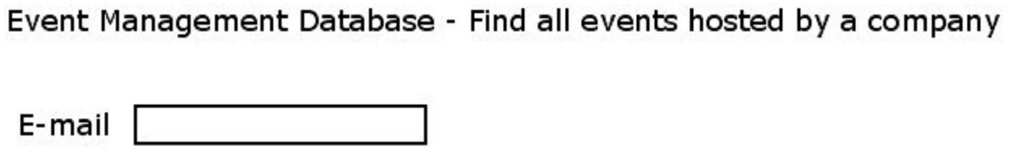
The above screen is seen when the user is requesting information about a given company. All of the selected company's information is output.



Event Management Database - Find all info about an event

Event ID

The above screen is seen when the user is requesting information about a given event. All of the selected event's information is output.

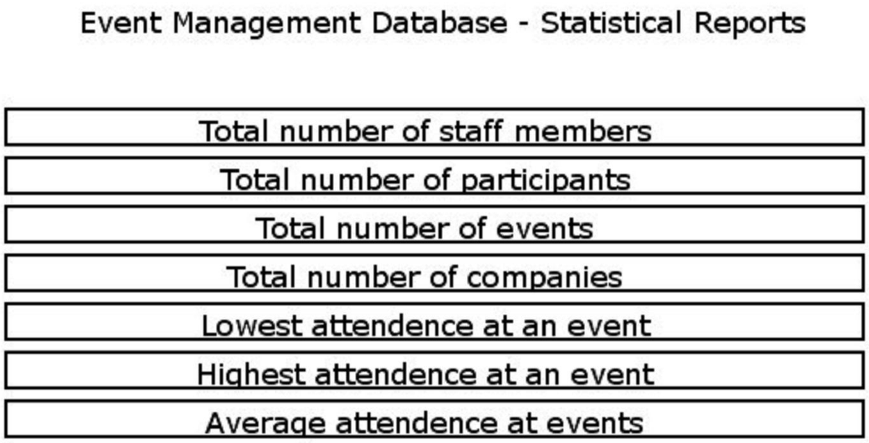


Event Management Database - Find all events hosted by a company

E-mail

The above screen is seen when the user is requesting all events hosted by a given company. All of the events the selected company is hosting is output.

Screens that perform statistical reports:


Event Management Database - Statistical Reports
Total number of staff members
Total number of participants
Total number of events
Total number of companies
Lowest attendence at an event
Highest attendence at an event
Average attendence at events

The above screen is seen when the user wants to generate general reports about the database. The outputs from this screen depend on the option selected. For additional information about the queries see section 6.

8. Implementation and Testing Plans

- **Distribution of processes**

  - The processes for implementation, documentation, debugging, testing, etc. for the project should not be executed by a single individual. The following shows how the work load is to be distributed among group members.

  - Entity creation schedule:
    - Person (3.3 man-weeks)
      - Implementation: Dale
      - Documentation: Dale
      - Testing: Dale, Gary, Matt
    - Event (2.3 man-weeks)
      - Implementation: Matt
      - Documentation: Matt
      - Testing: Dale, Gary, Matt
    - Company (1.83 man-weeks)
      - Implementation: Gary
      - Documentation: Gary
      - Testing: Dale, Gary, Matt
    - Location (.83 man-weeks)
      - Implementation: Gary
      - Documentation: Gary
      - Testing: Dale, Gary, Matt

  - Overall documentation (incorporation of individual documentation): Dale, Gary, Matt

  - Timeline manager: Dale

- **Testing Plans (with time estimation)**

  - Once the database is created it will be populated with test data and every requirement listed in section 4 and 6 will be tested. Each individual will be responsible for testing the Entity he creates (as listed above). This plan makes the most sense since the person who has spent the most time developing the Entity will know it the best. (1 man-week)

  - After the it has been decided that the DBMS performs the required functions properly, the interface will be created using PHP. This step will be more thoroughly evaluated when the testing is complete. There is a distinct possibility that this step will be cut short. (3 man-weeks)

**Note: Time estimations do not include time required for documentation or debugging of implementation. Total time consumption will be greater than the number listed.*

9. Estimate of effort

- **Individual item estimations**

    - For these estimations, we count...
        - each attribute as taking 30 minutes to implement.
        - each subclass as taking 60 minutes to implement.
        - each relationship as taking 60 minutes to implement.

    - Entity: Person
        - 9 attributes = 270 minutes.
        - 3 subclasses = 180 minutes.
        - 0 relationships = 0 minutes.
        - *Subtotal time = 450 minutes.*

        - Subclass Entity: Class_Participant
            - 3 attributes = 90 minutes.
            - 0 subclasses = 0 minutes.
            - 1 relationship = 60 minutes.
            - *Subtotal time = 150 minutes.*

        - Subclass Entity: Screen_Participant
            - 1 attribute = 30 minutes.
            - 0 subclasses = 0 minutes.
            - 1 relationship = 60 minutes.
            - *Subtotal time = 90 minutes.*

        - Subclass Entity: Staff
            - 1 attribute = 30 minutes.
            - 0 subclasses = 0 minutes.
            - 3 relationships = 240 minutes.
            - *Subtotal time = 270 minutes.*

    - Entity: Event
        - 7 attributes = 210 minutes.
        - 2 subclasses = 120 minutes.
        - 1 relationship = 60 minutes.
        - *Subtotal time = 390 minutes.*

        - Subclass Entity: Class
            - 3 attributes = 90 minutes.
            - 0 subclasses = 0 minutes.
            - 2 relationships = 120 minutes.
            - *Subtotal time = 210 minutes.*

        - Subclass Entity: Screening
            - 2 attributes = 60 minutes.
            - 0 subclasses = 0 minutes.
            - 2 relationships = 120 minutes.
            - *Subtotal time = 180 minutes.*

    - Entity: Company
        - 7 attributes = 210 minutes.
        - 0 subclasses = 0 minutes.
        - 2 relationships = 120 minutes.
        - *Subtotal time = 330 minutes.*

- Entity: Location
    - 3 attributes = 90 minutes.
    - 0 subclasses = 0 minutes.
    - 1 relationship = 60 minutes.
    - *Subtotal time = 150 minutes.*

- **Total time estimation***

    - *Note: At our estimation, 3 hours per week are able to be dedicated to this project. We will use this to estimate the number of weeks for completion. This unit will be called a "man-week".*

    - Entity: Person (including subclasses)
        - 960 minutes (3.3 man-weeks).
    - Entity: Event (including subclasses)
        - 600 minutes (2.3 man-weeks).
    - Entity: Company
        - 330 minutes (1.83 man-weeks).
    - Entity: Location
        - 150 minutes (.83 man-weeks).

    - Total time estimation:
        - *2040 minutes = 34 hours = 11.3 hours/person (3 people)*
        - *11.3/3 = 3.8 weeks to complete project implementation.*
        - *4 weeks total including documentation time necessary.*

Appendix 1

Well*Life is an organization sponsored by Carondelet Health. They help organize health-oriented classes for people and for companies to help improve people's daily life. For more information, visit their website: http://www.carondelethealth.org/welllife/.

When entering a new person into the database for a class, it is plausible that they are a BlueCross BlueShield and will qualify for additional benefits, depending on the event. BlueCross BlueShield is an insurance organization that provides cost-effective health insurance for many American citizens. For more information, visit their website: http://www.bcbs.com/.