|  |
| --- |
| Radio Club  Member Management Database Documentation |
| Josiah Bennett University of Pittsburgh at Bradford April 19, 2012 |

# Introduction

*Statement of the Problem*

The University of Pittsburgh at Bradford Radio Club currently has no true way to track what members attend meetings. Many members make a valid effort to make each and every meeting, but there are no methods in place to manage all of the members. Currently all of the members on the executive board go to the meetings and then wait for the members to arrive. This can allow for the members to attend meetings whenever they decide to. While this may be convenient it is less than ideal.

While allowing members to decide when they would like to attend meetings is convenient, it severely hurts club productivity. Without a stable and consistent member base all of the club’s projects aren’t completed in a timely manner. Many projects end up being pulled from the planned events. With all of the cancelled events go potential funds for future projects. These funds are vital for the club to run efficiently. Without funds coming in the radio club would be left without the ability to broadcast new shows or order new equipment.

Goals and Objectives

The goal of this project is to create a management system in order to manage the attendance of members, committees, and responsibilities in the University of Pittsburgh at Bradford Radio Club. The system will be built on a program created by Visual C# for the front-end structure in order to be easy and simple to use for the executive board. The program will also utilize an SQL backbone in order to create the database of members as well as their attendance records. The SQL backbone will give the program designer the ability to see what members are working on. The database will also give out statistics like how much money the members owe the club in dues as well as what committee the members belong to. The system will allow for more responsibility to be put on the members in order to make sure they attend meetings, and manage their projects efficiently.

Description of the Project

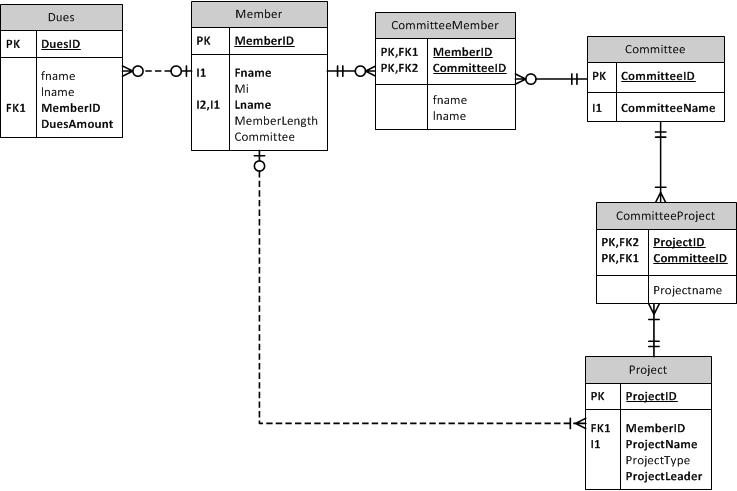
The system developed will utilize SQL Server as well as Visual C#. Visual C# will provide the GUI for the management program. With a simple yet thorough interface, the executive board will be able to run the club better. Creating a Visual C# program to help management will also open the possibility to port the system to another operating system for use in other clubs and organization.

A SQL Server database will be the backend part of the program. The SQL database will house all of the members, committees, attendance rates, projects, majors, and money they have helped to raise. The SQL scripts will be integrated with the Visual C# program to create drop down boxes that will allow the executive committee to choose what members they want to query. This will help allow the management staff to manage the members more effectively. The SQL scripts could potentially be distributed to other clubs in order for the other clubs or organizations to take advantage of this project.

Tentative Timeline

|  |  |
| --- | --- |
| February 12, 2012 | Conceptual Model Due |
| March 7, 2012 | Logical Model Due |
| April 1, 2012 | Physical Database Model Due |
| April 15,2012 | Database Views Due |
| April 15,2012 | Insert Statements Due |
| April 21,2012 | Stored Procedures Due |
| April 26, 2012 | Final Documentation Due |

ER Diagram



Physical Database Model

/\*

Bennett, Josiah

Web Databases

Finalized Physical Model

4/25/12

\*/

use master;

--Drop the database if it already exists

IF EXISTS (

SELECT name

FROM sys.databases

WHERE name = 'radioclub'

)

DROP DATABASE radioclub;

GO

--creates the database for radioclub

create database radioclub

go

use radioclub;

go

--Tables

--Create table member.

create table member(

memberid int identity primary key,

fname varchar(25)not null,

mi varchar(1),

lname varchar(25)not null,

memberlength varchar(2),

committee varchar(25));

--Create table project.

create table project(

projectid int identity primary key,

memberid int,

projectname varchar(25)not null,

projecttype varchar(10),

projectleader varchar(25)not null

);

--Create table committee.

create table committee(

committeeid int identity primary key,

committeename varchar(25));

--Create table committeeproject.

create table committeeproject(

projectid int,

projectname varchar(25),

committeeid int,

primary key(projectid,committeeid));

--Create table committeemember.

create table committeemember(

memberid int,

fname varchar(25),

lname varchar(25),

committeeid int,

primary key(memberid,committeeid));

--Create table dues.

create table dues(

duesid int identity primary key not null,

fname varchar(25),

lname varchar(25),

memberid int not null,

duesamount decimal

);

--Indexes

--Creates an index on the column (lname) on table member.

create index idx\_member\_lname on member(lname);

--Creates an index on the columns (fname and lname) on table member.

create index idx\_member\_fname\_lname on member(fname,lname);

--Creates an index on the column (committeename) on table committee.

create index idx\_committee\_name on committee(committeename);

--Creates an index on the column (projectname) on table project.

create index idx\_project\_name on project(projectname);

--Foreign Keys

--Constraint for table dues.

alter table dues add constraint fk\_dues\_member foreign key (memberid)references member(memberid);

--Constraint for table project.

alter table project add constraint fk\_project\_member foreign key (memberid)references member(memberid);

--Constraints for table committeeproject.

alter table committeeproject add constraint fk\_committeeproject\_project foreign key (projectid)references project(projectid);

alter table committeeproject add constraint fk\_committeeproject\_committee foreign key (committeeid)references committee(committeeid);

--Constraints for table committeemember.

alter table committeemember add constraint fk\_committeemember\_member foreign key (memberid)references member(memberid);

alter table committeemember add constraint fk\_committeemember\_committee foreign key (committeeid)references committee(committeeid);

--Insert Statements

--Inputs data into table member.

insert into member (fname,mi,lname,memberlength,committee)values('Josiah','J','Bennett',2,'Executive');

insert into member (fname,mi,lname,memberlength,committee)values('Alexandria','K','Young',2,'Studio');

insert into member (fname,mi,lname,memberlength,committee)values('John','C','Stahlman',2,'Executive');

insert into member (fname,mi,lname,memberlength,committee)values('Vincent',null,'Petrone',2,'Executive');

insert into member (fname,mi,lname,memberlength,committee)values('Tyler',null,'Crozure',2,'Studio');

insert into member (fname,mi,lname,memberlength,committee)values('Johnny','R','Jefferson',3,'Executive');

insert into member (fname,mi,lname,memberlength,committee)values('Kali',null,'Learn',1,'Studio');

insert into member (fname,mi,lname,memberlength,committee)values('Connor',null,'Holton',1,'Studio');

insert into member (fname,mi,lname,memberlength,committee)values('Cody','A','Souder',2,'Executive');

insert into member (fname,mi,lname,memberlength,committee)values('Robert',null,'Small',3,'Studio');

--Inputs data into table dues.

insert into dues (fname,lname,memberid,duesamount)values('Josiah','Bennett',1,'10.00');

insert into dues (fname,lname,memberid,duesamount)values('Alexandria','Young',2,'10.00');

insert into dues (fname,lname,memberid,duesamount)values('John','Stahlman',3,'10.00');

insert into dues (fname,lname,memberid,duesamount)values('Vincent','Petrone',4,'10.00');

insert into dues (fname,lname,memberid,duesamount)values('Tyler','Crozure',5,'10.00');

insert into dues (fname,lname,memberid,duesamount)values('Johnny','Jefferson',6,'5.00');

insert into dues (fname,lname,memberid,duesamount)values('Kali','Learn',7,'15.00');

insert into dues (fname,lname,memberid,duesamount)values('Connor','Holton',8,'15.00');

insert into dues (fname,lname,memberid,duesamount)values('Cody','Souder',9,'10.00');

insert into dues (fname,lname,memberid,duesamount)values('Robert','Small',10,'5.00');

--Inputs data into table committee.

insert into committee (committeename) values ('Executive');

insert into committee (committeename) values ('Studio');

--Inputs data into table project.

insert into project(memberid,projectname,projecttype,projectleader)values(3,'Alumni Weekend Event','Community','John Stahlman');

insert into project(memberid,projectname,projecttype,projectleader)values(3,'Weekend Radio Show Events','Community','John Stahlman');

insert into project(memberid,projectname,projecttype,projectleader)values(4,'Club Night Event','Membership','Vincent Petrone');

insert into project(memberid,projectname,projecttype,projectleader)values(1,'Member Management','Membership','Josiah Bennett');

--Inputs data into table committeemember.

insert into committeemember (memberid,fname,lname,committeeid)values(1,'Josiah','Bennett',1);

insert into committeemember (memberid,fname,lname,committeeid)values(2,'Alexandria','Young',2);

insert into committeemember (memberid,fname,lname,committeeid)values(3,'John','Stahlman',1);

insert into committeemember (memberid,fname,lname,committeeid)values(4,'Vincent','Petrone',1);

insert into committeemember (memberid,fname,lname,committeeid)values(5,'Tyler','Crozure',2);

insert into committeemember (memberid,fname,lname,committeeid)values(6,'Johnny','Jefferson',1);

insert into committeemember (memberid,fname,lname,committeeid)values(7,'Kali','Learn',2);

insert into committeemember (memberid,fname,lname,committeeid)values(8,'Connor','Holton',2);

insert into committeemember (memberid,fname,lname,committeeid)values(9,'Cody','Souder',1);

insert into committeemember (memberid,fname,lname,committeeid)values(10,'Robert','Small',2);

--Inputs data into table committeeproject.

insert into committeeproject(projectid,projectname,committeeid)values(1,'Alumni Weekend Event',2);

insert into committeeproject(projectid,projectname,committeeid)values(2,'Weekend Radio Show Events',2);

insert into committeeproject(projectid,projectname,committeeid)values(3,'Club Night Event',2);

insert into committeeproject(projectid,projectname,committeeid)values(4,'Member Management',1);

--Views

--Gives all member dues without filter.

go

CREATE VIEW memberdues AS

SELECT m.memberid as 'Member ID',m.fname as 'First Name',m.lname as 'Last Name',d.duesamount as 'Amount Due'

FROM member m

join dues d on m.memberid=d.memberid

group by d.duesamount,m.memberid,m.fname,m.lname

--Gives the member's name alone with the project they are currently working on.

go

CREATE VIEW memberproject AS

SELECT m.memberid as 'Member ID',m.fname as 'First Name',m.lname as 'Last Name',p.projectname as 'Project'

FROM member m

join project p on m.memberid=p.memberid

--Gives the member name along with the total amount of dues they owe based on a selected amount.

go

CREATE VIEW membertotaldues AS

SELECT m.memberid as 'Member ID',m.fname as 'First Name',m.lname as 'Last Name', d.duesamount as 'Amount Due'

FROM member m

join dues d on m.memberid=d.memberid

where d.duesamount in ('10.00')

group by d.duesamount,m.memberid,m.fname,m.lname

--Helps users find a member along with that member's committee.

go

CREATE VIEW membercommittee AS

SELECT cm.memberid as 'Member ID',cm.fname as 'First Name',cm.lname as 'Last Name', c.committeename as'Committee'

FROM committeemember cm

join committee c on cm.committeeid=c.committeeid

--Stored Procedures

--Looks up members based on committee ranking.

go

create procedure membercommitteelookup

@committee varchar(15)

as

begin

select memberid as 'Member ID',fname+lname as 'Member Name' from member

where committee = @committee;

end

go

exec membercommitteelookup 'Studio';

--Updates a member's committee ranking.

go

create procedure updatecommittee

@fname varchar(25),

@lname varchar(25),

@newcommittee varchar(15)

as

begin

update member

set committee=@newcommittee

where fname=@fname and lname=@lname;

end;

go

exec updatecommittee 'John','Stahlman','Executive';

select\*from member

--Find a member by memberid

go

create procedure membername

@memberid int,

@fname varchar(25) output,

@lname varchar(25) output

AS

BEGIN

select @fname=fname,@lname=lname

from member

where memberid=@memberid;

END;

Go

declare @firstname varchar(25);

declare @lastname varchar(25);

EXEC membername '1',@firstname output,@lastname output;

SELECT @firstname+@lastname as Member;

What Has Changed, What Did I learn?

There were a lot of things that changed throughout the entire project. In the beginning part of the project there were many different ideas I had planned for my project. Initially I had hoped to be able to create an entire Visual C# interface for my database to run off of. I still may create the program in my free time to implement into the actual radio club’s meetings. This would be a very useful program to have running during our meetings.

Initially, I had also though I would need a lot more tables to house all of the information that I planned to have integrated into the database. This turned out to be a false assumption, because I was able to house all of my information in only a couple of tables. Learning how to normalize data taught me how to minimize the amount of information in my database that would end up being redundant. At the end of my project it turned out that I used nearly every component I planned to except for numerous extra unneeded tables.

Throughout the creation of my database I learned a lot about how to use SQL to create and use databases. I feel more confident in my abilities to use SQL in a business setting. My experience with SQL Server has created an interest in me to teach myself how to use MySQL and other database programs. I feel more confident to be able to list SQL experience on my resume and while doing so bettering my chances of getting a job that deals with SQL.

References

Pratt, P. J., & , M. Z. (2008). *A guide to sql*. (8th ed.).

Course Technology Ptr.