



Music
at **MARIST**

Singers

Marist College Singers

Database Project

Tori Palazzolo

December 2 2013

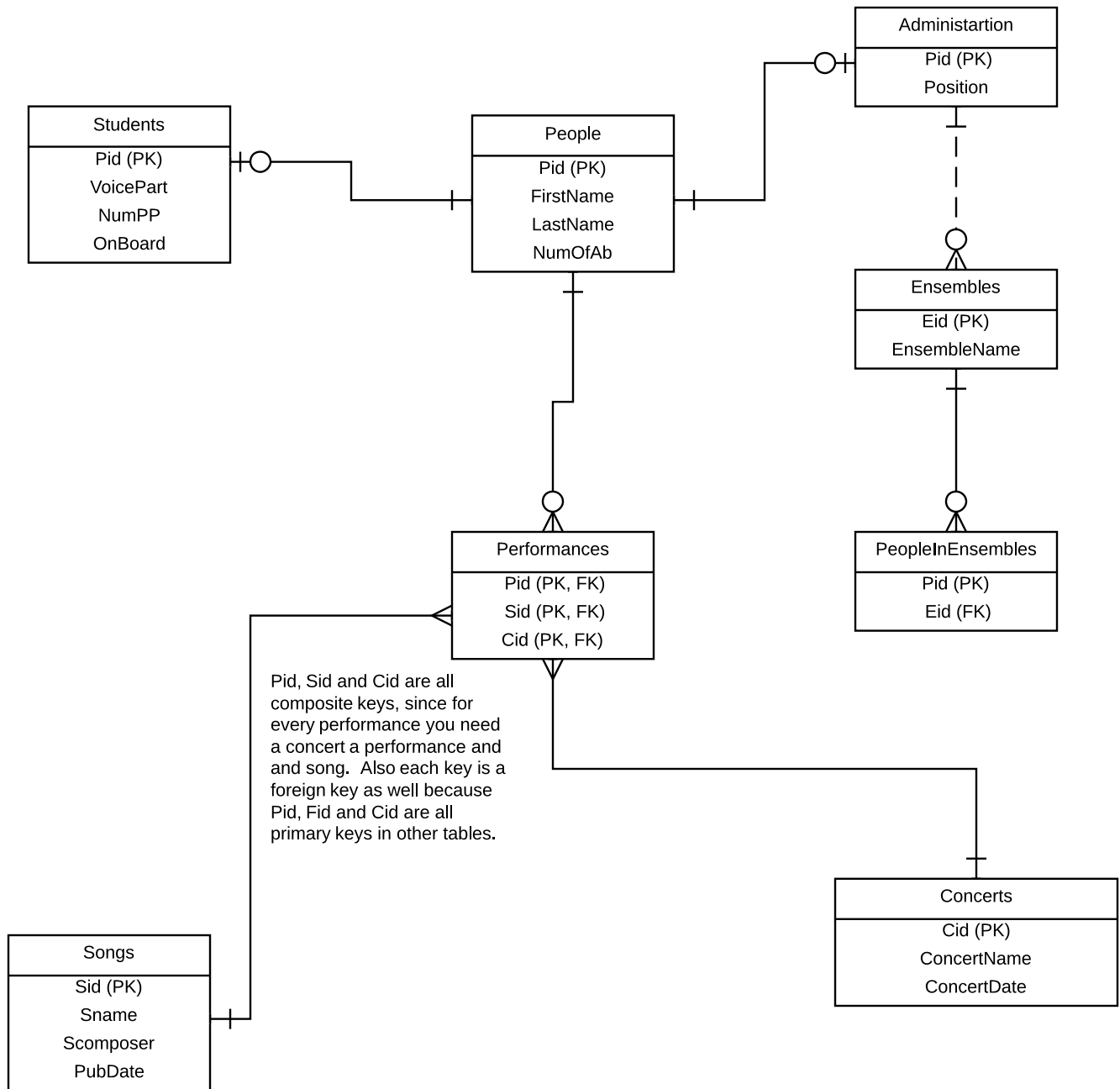
Executive Summary

Marist Singers, is one of the largest groups on campus, containing over 150 members. Within the club there is one huge ensemble, called the Marist singer with many smaller ensembles. Being a member of the Marist singers enables you to audition for smaller ensembles. The club puts on three concerts a semester. Each concert has around five or six songs, with many other additional soloist who perform. The club has many rules and regulations that you must obey in order to get priority points towards your housing. The club also has an administration that works within the music department, and a student board as well. The singer's administration, makes sure that all of the rules are being followed and are also responsible for recording everything that goes on within the club.

By creating a database, we can see every member within the club, their basic information, whether or not they are in an ensemble or not , and their attendance record. The database will also contain every concert with their song record, soloist and date. The database will also be able to tell you the student board as well as the administrators. This databases overall goal is to organize this large club to make it easier to see people's involvement within the club, as well as the keep a song record for the concerts.

The Databases Main goals are to include:

- A record of people who are in the overall club.
- People who are executives with in the club. I.E Student Board and Administrators.
- Concert Records: Including who performed, what song they performed and what concert it was for.
- Priority point and attendance records. Which is essential for your housing and involvement within the club.



Create Statements

People Table

The people table includes everyone in singers, including administration. When you are entered into the database you will receive a pid, to make sure that every student entered is unique. The peoples table also includes the persons first name, last name, and number of absences.

```
CREATE TABLE People (
  Pid      char(4) not null,
  Firstname char(20),
  LastName  char(20),
  NumOfAb   int,
  primary key(Pid)
);
```

Pid	FirstName	LastName	NumOfAb
P01	Ashley	Morris	1
P02	Kenny	Lane	2
P03	Maggie	Gunther	3
P04	Kate	Smith	1
P05	Nick	Bayer	1
P06	Kerry	Flanagan	2
P07	Art	Hemel	1
P08	Sarah	Williams	0
P09	Mike	Napolitano	2
P10	Fran	Green	5
P11	Margie	Zarcone	2
P12	Tommy	Heart	4
P13	Callie	Parmale	1
P14	Harrison	Davis	2
P15	Kat	Marie	3

Functional Dependencies:

Pid → FirstName, LastName, NumOfAb

Songs Table

The songs table where all of the songs, that were performed in singers are located. Every song has a song id, the song name, composer and the publishing date. The song table is just a way for the singer's administration to see what songs they have on file, so they know what they sung in the past, and optional songs for future concerts. (Note my sample data is only a fraction of the song record that Marist College singers has).

```
CREATE TABLE Songs (
  Sid      varchar(4) not null,
  SName    char(100),
  SComposer char(20),
  PubDate  Date,
  primary key(Sid)
);
```

Functional Dependencies:

Sid → Sname, SComposer, PubDate

Sid	SName	SComposer	Pub Date
S01	Come Sail Away	Styx	12/10/1981
S02	Lacrymosa	Mozart	6/6/1561
S03	Aint No Mountain High Enough	Martin Grey	3/5/1973
S04	Im Alive	Jack Lock	7/18/1991
S05	Silent Night	Chris Berk	6/9/1431
S06	You Can't Stop The Beat	Hair Spray	8/21/1982
S07	On My Own	Les Mes	12/05/1981
S08	Some Nights	Fun	5/7/2011
S09	Always Be My Baby	Mariah Carey	7/8/1990
S10	Faithfully	Journey	10/10/1989
S11	Wicked	Emory Green	11/13/1984
S12	Imagine	John Lennon	12/14/1970
S13	Hark The Harold	Santa	09/25/1890
S14	Marist Fight Song	Art Hemel	05/14/2002

Students Table

The students table is where everyone who is not an administrator is located. For this table we have the primary key which is the pid as well as information about the student such as their voice part, number of priority points they have and whether or not a student is on the board.

```
CREATE TABLE Students (
  Pid      varchar(4) not null,
  VoicePart char(20),
  NumPP    int,
  OnBoard  boolean,
  primary key(Pid)
);
```

Functional Dependencies:

Pid → VoicePart, NumPP, OnBoard

Pid	VoicePart	NumPP	On Board
P01	Soprano	2	T
P02	Bass	2	T
P03	Alto	2	T
P04	Soprano	2	T
P05	Tenor	2	T
P06	Soprano	2	T
P10	Alto	2	F
P11	Soprano	2	F
P12	Tenor	2	F
P13	Soprano	2	F
P14	Tenor	2	F
P15	Alto	2	F

Administration

The administration table tells you who is in the administration, as well as what their position is.

```
CREATE TABLE Administration(
  Pid      varchar(4) not null,
  Position  char(16),
  primary key(Pid),
  foreign key (Pid) References people(pid)
);
```

Functional Dependencies:

Pid → Position

Pid	Position
P07	Dean Of Music
P08	Director
P09	Advisor

Concerts

The concert table is where all of the concert information is located. Within this table to have a Concert Id (Cid) and a Concert Name along with the Concert Date.

```
CREATE TABLE Concerts (
  Cid      varchar(4) not null,
  ConcertName char(20),
  ConcertDate Date,
  primary key(Cid)
);
```

Functional Dependencies:

Cid → ConcertName, ConcertDate

Cid	Concert Name	Date
C01	Parents Weekend	09/12/2013
C02	Night On Broadway	11/14/2013
C03	Lessons In Carols	12/3/2013

C04	LITA	2/16/2014
C05	Barnavon	4/10/2014

Performances

The performance table is the associative table in the singers database. It was created to avoid many to many relationships within the database. The performance database consist of every performance throughout the year. The table has three entities like Pid, Sid and Cid. These entities are all composite keys, because in each performance you need to must have people who performed, the concert they performed at and the name of the song that they performed. Each entity is also a foreign key, since they are all primary keys from other tables. For example Cid is the primary key in the Concert Table.

```
CREATE TABLE Performances (
  Pid  varchar not null,
  Cid  varchar(4) not null references concerts(cid),
  Sid  varchar(3) not null references songs(sid),
  primary key(Pid, Sid, Cid)
);
```

Functional Dependencies:

{Pid, Cid, Sid} → Pid, Cid, Sid

Cid	Pid	Sid
C01	P01	S01
C01	P02	S01
C01	P03	S01
C01	P04	S01
C01	P05	S01
C01	P06	S01
C01	P07	S01
C01	P08	S01
C01	P09	S01
C01	P10	S01
C01	P11	S01
C01	P12	S01
C01	P14	S01
C01	P15	S01
C01	P05	S04
C01	P06	S09

C01	P01	S03
C01	P02	S03
C01	P03	S03
C01	P04	S03
C01	P05	S03
C01	P06	S03
C01	P07	S03
C01	P08	S03
C01	P09	S03
C01	P10	S03
C01	P11	S03
C01	P12	S03
C01	P14	S03
C01	P15	S03
C02	P15	S06
C02	P01	S06
C02	P04	S06
C02	P01	S11
C02	P02	S11
C02	P03	S11
C02	P04	S11
C02	P05	S11
C02	P06	S11
C02	P07	S11
C02	P08	S11
C02	P09	S11
C02	P10	S11
C02	P11	S11
C02	P12	S11
C02	P14	S11
C02	P15	S11
C02	P11	S07
C03	P02	S05
C03	P01	S02
C03	P02	S02
C03	P03	S02
C03	P04	S02
C03	P05	S02
C03	P06	S02
C03	P07	S02
C03	P08	S02
C03	P09	S02
C03	P10	S02
C03	P11	S02
C03	P12	S02
C03	P14	S02

C03	P15	S02
C03	P10	S13
C04	P01	S08
C04	P02	S08
C04	P03	S08
C04	P04	S08
C04	P05	S08
C04	P06	S08
C04	P07	S08
C04	P08	S08
C04	P09	S08
C04	P10	S08
C04	P11	S08
C04	P12	S08
C04	P14	S08
C04	P15	S08
C05	P01	S14
C05	P02	S14
C05	P03	S14
C05	P04	S14
C05	P05	S14
C05	P06	S14
C05	P07	S14
C05	P08	S14
C05	P09	S14
C05	P10	S14
C05	P11	S14
C05	P12	S14
C05	P14	S14
C05	P15	S14
C05	P14	S12

People In Ensembles

People in Ensembles table contains people in singers that are in smaller ensemble. To be in a smaller ensemble on campus you must be in the marist singers. For example to be in our all-female acapella group you must be in marist singers as well. The table contains Eid and Pid which are composite primary keys because you for every ensemble there must be people.

```
CREATE TABLE PeopleInEnsembles(
  Pid      varchar(4) not null References people(Pid),
  Eid      varchar(4) not null References Ensembles(Eid),
  primary key(Pid, Eid)
);
```

Functional Dependencies:

{Pid, Eid} → Pid, Eid

Pid	Eid
P01	E01
P01	E04
P02	E03
P03	E04
P04	E01
P05	E01
P05	E03
P10	E02
P11	E04
P12	E01
P12	E02
P12	E03

Ensembles

The Ensemble Table is all of the smaller ensembles within the Marist Singers. The table contains two entries giving every ensemble name a unique identification key (Eid) in order to make each ensemble unique.

```
CREATE TABLE Ensembles(
  Eid      varchar(4) not null,
  EnsembleName char(16),
  primary key(Eid)
);
```

Functional Dependencies:

Eid → EnsembleName

Eid	EnsembleName
E01	Chamber Choir
E02	Gospel Choir
E03	Time Check
E04	Sirens

Views

In Marist Singers if you have four or more absences you can be asked to leave the club. Therefore I decided to create a view that shows the administration, who has four or more absences and they can see who will ultimately be kicked out of the club. My view takes the NumofAb entity from the people table and sees who has four or more absences.

```
Create View NumberofAbs AS
Select FirstName, LastName, NumofAb
From People
Where NumofAb >= 4;
```

	firstname character(20)	lastname character(20)	numofab integer	
1	Fran	Green	5	
2	Tommy	Heart	4	

Fran Green and Tommy Heart, both risk getting thrown out of Marist Singers.

Marist singers is run with a combination of the student board as well as the administration. In the student table I create an entity therefore you can see who is on the board or not. This view was created so that singer's admin, college activities etc. can see who is on the board, so they know who to contact if they have any concerns.

```

Create View PeopleOnBoard As
Select students.pid, students.OnBoard, people.FirstName, people.LastName
From Students
Inner Join People
On students.pid=people.pid
Where Students.OnBoard='T';

```

Here are the students who are on the Singers Board.

	pid character varying(4)	onboard character(4)	firstname character(20)	lastname character(20)
1	P01	T	Ashley	Morris
2	P02	T	Kenny	Lane
3	P03	T	Maggie	Gunther
4	P04	T	Kate	Smith
5	P05	T	Nick	Bayer
6	P06	T	Kerry	Flanagan

Reports And Their Queries

Chamber Choir is a smaller ensemble in Marist Singers. In order to get the output who had to join people and People in Ensembles to see where the pids match up. Than in order for the where clause to work you had to inner join ensemble to see where the EnsembleName was Chamber Choir.

Get the names of everyone who is in chamber choir.

```
Select people.FirstName, people.LastName
From people
Inner Join PeopleInEnsembles
On people.pid=PeopleInEnsembles.pid
Inner Join Ensembles
On PeopleInEnsembles.Eid=Ensembles.Eid
Where EnsembleName='Chamber Choir';
```

Names Of Students who are in Chamber Choir

	firstname character(20)	lastname character(20)	
1	Ashley	Morris	
2	Kate	Smith	
3	Nick	Bayer	
4	Tommy	Heart	

The main purpose of the song table is an overall record of what songs they have on file. A song record is there so the singers know what songs they have sung in past performances and what songs they can sing in future performances. For each performance you cannot sing a song that has been performed over the past five years. The main purpose of this query what songs have not been performed yet, so that Marist college singers has that option in future years. For this query we must find the nulls in the performance table, and locate the Sid and song name in the Songs table, therefore we must use a full/outer join.

Song(s) that have not be sung in a Concert this year.

```

Select Songs.Sid, Songs.SName
From Songs
Full Join Performances
On Songs.Sid=Performances.Sid
Where Performances.Sid is Null |

```

The Song Faithfully has not been sung in a concert this year. Therefore the Marist singers can perform the song in future years.

	sid character varying(4)	sname character(100)
1	S10	Faithfully

An attendance record is ideal for Marist singers. The administrators like to see how everyone is doing in terms of attendance, and where does everyone stand. For this reason I decided to see who has more than the average absences, so we can see who is not on track, and how many people have more attendances than they should.

See who is above the average attendance record.

```

Select NumofAb, FirstName, LastName
From People
Where NumOfAb>(Select Avg(NumofAb) From People)
Group By People.FirstName, People.LastName, People.NumofAb

```

These people have above the average attendance record. Meaning that they missed more than other people in the club.

	numofab integer	firstname character(20)	lastname character(20)
1	3	Maggie	Gunther
2	4	Tommy	Heart
3	5	Fran	Green

Stored Procedures

Stored procedures main purpose is to store certain procedures/code in the system catalog, therefore you do not need to write continuous code. Since the database is so big, attendance records can be quite big. My stored procedure will see who has four or more attendances. (Similar to the view that I wrote before). The purpose of the stored procedure is that for every modification, example someone gets another absence etc, they do not keep having to write more code. My stored procedure also has an alert, so that the administration can notify. In every database there is isolation levels, with isolation levels there is different types of reads such as dirty reads and phantom reads.

In the Singers Database I chose to set the default isolation level of Read committed, which guarantees that any data read is committed and read by the database, at that moment. Isolation are set up in order to keep the database in third normal form, by ensuring that the database follows ACID (Atomic, Consistent, Independent, and Durable)

```
Create Procedure HighNumAb
As Warning_Abs
Begin
If NumofAb >=4
Than Return 'Must Notify them for being absent to much'
End If;
Select People.NumofAb
From People
Where People.Pid=People.NumofAb
End
```


Triggers



A trigger is a something that the Database create sets on a view or a table, to change the results whenever a certain clause or event happens. For example in Singers if you are on the Student board you get three priority points instead of two. Therefore in my database I set up a trigger that will increase the amount of priority points if someone is on the Singer's board.

```
Create Trigger BoardPP
Before Insert On Students
For Each Row
Begin
Print 'Increase PP because on Board'
select NumofPP
declare NumofPP
select OnBoard From Students
update Students
set NumofPP= '3'
Where OnBoard ='T'
End
```

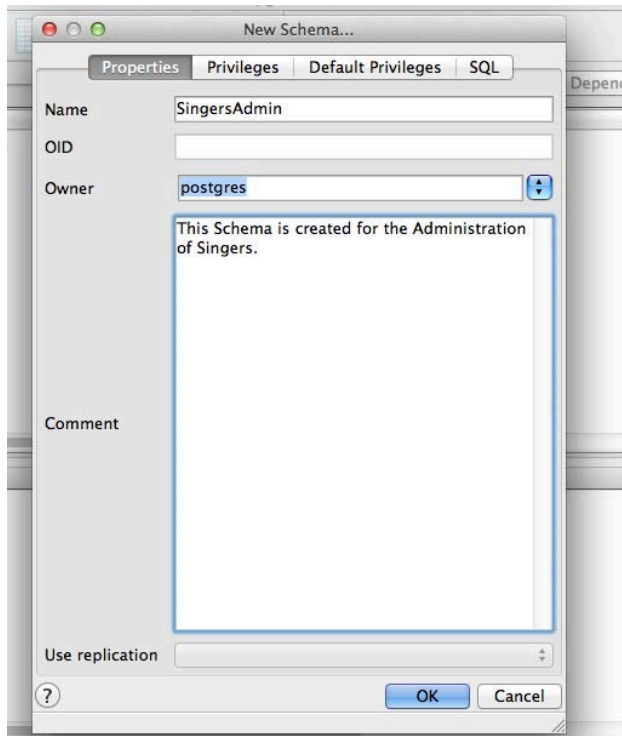
Security

To make this database secure we must create a separate schema for Singers Administration. What Creating a schema does is that it allows separation with managing the database. Therefore if we group object and files together and only allow access for the Admin of singers, we get a more secure database.

In the PHP my admin homepage you have the option to create new schemas rather than just the public one. As you can see here I created a schema for the singers administration, and a public one as well.

Schema	Owner	Comment
 SingersAdmin	postgres	This Schema is created for the Administration of Singers.
 public	postgres	standard public schema

These are the two schemas that are in the singers database.



Schemas allow you to insert different domains, templates views, triggers etc. To make this database more secure we have public schema and a singers schema. To restrict access from the public schema we can revoke privileges on certain tables such as the students table, and people table because they have information that can be desired to be changed. For example in singers if you have more than four absences you risk getting kicked out of the club, therefore you would want to revoke number of absences in the public schema because you do not want someone to change that information. Now for the SingersAdmin schema they will have access to every table, and have permission to change the amount of absences and have access to priority points if needed. Therefore the singers admin can Grant Update Search and Delete, all rows and columns from every table

```
Create User Singers_Administration
Grant Select, Insert, Update, Delete
On Students, People
In Schema SingersAdmin
To Singers_Administration
```

Implementation Notes

- Create a user interface that the Student Board and Administration can access with ease. Most people do not know how to use Postgress therefore a user interface must be implemented.
- More security- in the future we could add more schemas insert create and revoke to add more security between the administration and the public.
- Create more views, since the club is so large if we create more views it would be easier for use to distinguish different people. For example we could create a view for each ensemble therefore we can view who is in each ensemble.
- The club goes to Disney every year, one thing that we could implement more tables concerning the trip, such as people who are going, their flights hotel rooms etc.
- In some of our concerts, some ensembles perform, in the future I would like to include Eid in the performance table, and create views and or queries to see what ensemble performed at what concert.

Known Problems

- The database can use more tables, which would make it more detailed, in certain aspects. Also table can include more entities, making each table more detailed.
- Over the years the amount of people who performed can be very large, making the performance table massive. In order to not compromise speed with our database we could do separate tables for each year, or implement an index.
- As of right now, the ensembles table does not have a director associated with it.
- The club does not have a way to contact any of their members. Therefore we must create more entities to the student or peoples table, allowing us to add more contact information.

Future Enhancements

- Include a table that includes members of MCTA (Marist College Theater Association) and Singers. The schedule of both the clubs usually conflict, and there is a different attendance policy concerning members that are within that club.
- This December the Singers Club is going to Disney, So it can be in the Database Development favor to create a table concerning people that are going Disney, who paid in full vs. who just paid the deposit.
- Time Check and Sirens have separate concerts to singers, therefore we can create another table(s) having to do with the separate concerts that sirens and time check are a part of.
- Create a table concerning with the practice rooms and meeting times. Since singers is such a huge club you the practice room changes periodically.
- You are allowed to get out of a concert if you have an excusal letter. For future enhancements the developer can create a table in order to include concert excusals.