Marist College Music Department

A Database Design Proposal

By Andrew Bauman

CMPT 308 - Fall 2017
The Marist Music Department is comprised of both instrumental and vocal performance ensembles containing nearly 400 students.

The purpose of this database system is to simplify the process of managing many of the daily bookkeeping tasks within the Marist College Music Department to afford the department the ability of remaining focused on producing music rather than becoming distracted by day-to-day tasks.
Entity-Relationship Diagram
People

Stores the general information for the people within the database.

Fundamental Dependencies

\[
\text{person_id} \rightarrow \text{firstName, lastName, dateOfBirth, phoneNumber, address, emailAddress}
\]

```
CREATE TABLE People (  
    person_id char(8) not null,  
    firstName text not null,  
    lastName text not null,  
    dateOfBirth date,  
    phoneNumber text,  
    address text,  
    emailAddress text not null,  
    primary key(person_id)
);
```
Musicians

Includes those who perform in ensembles.

Fundamental Dependencies

\[ \text{person}_\text{id} \rightarrow \text{expectedGradSemester}, \text{favoriteClef}, \text{favoriteGenre} \]

<table>
<thead>
<tr>
<th>person_id</th>
<th>expectedGradSemester</th>
<th>favoriteClef</th>
<th>favoriteGenre</th>
</tr>
</thead>
<tbody>
<tr>
<td>20083100</td>
<td>Spring 2020</td>
<td>Bass</td>
<td>Funk</td>
</tr>
<tr>
<td>20083101</td>
<td>Fall 2019</td>
<td>Treble</td>
<td>Pop</td>
</tr>
<tr>
<td>20083102</td>
<td>Spring 2018</td>
<td>Tenor</td>
<td>Marches</td>
</tr>
<tr>
<td>20083103</td>
<td>Spring 2021</td>
<td>Treble</td>
<td>Classical</td>
</tr>
<tr>
<td>20083104</td>
<td>Fall 2020</td>
<td>Bass</td>
<td>Jazz</td>
</tr>
<tr>
<td>20083105</td>
<td>Fall 2020</td>
<td>TAB</td>
<td>Hard Rock</td>
</tr>
<tr>
<td>20083106</td>
<td>Spring 2019</td>
<td>Treble</td>
<td>Swing</td>
</tr>
<tr>
<td>20083107</td>
<td>Fall 2018</td>
<td>Bass</td>
<td>Soft Rock</td>
</tr>
<tr>
<td>20083108</td>
<td>Spring 2018</td>
<td>Bass</td>
<td>Jazz</td>
</tr>
<tr>
<td>20083109</td>
<td>Fall 2021</td>
<td>Treble</td>
<td>Pop</td>
</tr>
<tr>
<td>20083110</td>
<td>Spring 2019</td>
<td>Tenor</td>
<td>Metal</td>
</tr>
</tbody>
</table>

CREATE TABLE Musicians (  
  person_id char(8) not null references People(person_id),  
  expectedGradSemester text,  
  favoriteClef text,  
  favoriteGenre text,  
  primary key(person_id)  
);
Instrumentalists

References those who perform in band ensembles.

```
CREATE TABLE Instrumentalists (  
   person_id  char(8) not null references Musicians(person_id),  
   primary key(person_id)  
);
```
Vocalists

References those who perform in vocal ensembles

Fundamental Dependencies
person_id → vocalRange

CREATE TABLE Vocalists (  
  person_id  char(8) not null references Musicians(person_id),  
  vocalRange  text,  
  primary key(person_id)  
);
Create Table Statements

Directors

<table>
<thead>
<tr>
<th>person_id</th>
<th>almaMater</th>
<th>favoriteCigaretteBrand</th>
<th>favoriteCoffeeBlend</th>
</tr>
</thead>
<tbody>
<tr>
<td>30042002</td>
<td>University of Michigan</td>
<td>Lucky Strike Menthol</td>
<td>black</td>
</tr>
<tr>
<td>30042003</td>
<td>Music U</td>
<td></td>
<td>Starbucks Holiday Blend</td>
</tr>
<tr>
<td>30042001</td>
<td>Marist College</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Includes those who direct ensembles.

Fundamental Dependencies

person_id → almaMater, favoriteCigaretteBrand, favoriteCoffeeBlend

```sql
CREATE TABLE Directors(
    person_id char(8) not null references People(person_id),
    almaMater text,
    favoriteCigaretteBrand text,
    favoriteCoffeeBlend text,
    primary key(person_id)
);
```
Department_Roles

Stores the possible roles and salaries office staff may have

Fundamental Dependencies

role_id → roleName, hourlyRate, salary, jobDesc

CREATE TABLE Department_Roles (
    role_id char(4) not null,
    roleName text,
    hourlyRate numeric(10,2),
    salary numeric(10,2),
    jobDesc text,
    primary key (role_id)
);

<table>
<thead>
<tr>
<th>role_id</th>
<th>rolename</th>
<th>hourlyrate</th>
<th>salary</th>
<th>jobdesc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>Director of Music and Director of Bands</td>
<td>73000.00</td>
<td></td>
<td>Responsible for the management and direction of the Marist Band and Music Department</td>
</tr>
<tr>
<td>0002</td>
<td>Director of Choral Activities</td>
<td>75000.00</td>
<td></td>
<td>Responsible for the management and direction of the Marist Singers Program and related ensembles</td>
</tr>
<tr>
<td>0003</td>
<td>Operations Manager</td>
<td>10.00</td>
<td></td>
<td>Manages the operations regarding the Marist Music Department</td>
</tr>
</tbody>
</table>
Administrators

Includes those who hold roles within the department.

Fundamental Dependencies

\[ \text{person_id} \rightarrow \text{role_id} \]

<table>
<thead>
<tr>
<th>person_id</th>
<th>role_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>30042002</td>
<td>0001</td>
</tr>
<tr>
<td>30042003</td>
<td>0002</td>
</tr>
<tr>
<td>30042004</td>
<td>0003</td>
</tr>
</tbody>
</table>
Ensembles

Includes the ensembles offered within the department.

Fundamental Dependencies
ensemble_id → ensembleName

```
CREATE TABLE Ensembles (  
    ensemble_id    char(5) not null,  
    ensembleName   text,  
    primary key(ensemble_id)  
);
```

<table>
<thead>
<tr>
<th>ensemble_id</th>
<th>ensemblename</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>Symphonic Band</td>
</tr>
<tr>
<td>00002</td>
<td>Wind Symphony</td>
</tr>
<tr>
<td>00003</td>
<td>Orchestra</td>
</tr>
<tr>
<td>00004</td>
<td>Jazz Foxes</td>
</tr>
<tr>
<td>00005</td>
<td>Brass Ensemble</td>
</tr>
<tr>
<td>00006</td>
<td>Flute Choir</td>
</tr>
<tr>
<td>00007</td>
<td>Marist Singers</td>
</tr>
<tr>
<td>00008</td>
<td>Time Check</td>
</tr>
<tr>
<td>00009</td>
<td>Sirens</td>
</tr>
<tr>
<td>00010</td>
<td>Pit Band</td>
</tr>
<tr>
<td>00011</td>
<td>Computer Society</td>
</tr>
</tbody>
</table>
Instruments

Includes the instruments which students can play in the ensembles

Fundamental Dependencies

\[
\text{instrument\_id} \rightarrow \text{instrumentName}, \text{instrumentFamily}
\]

CREATE TABLE Instruments (
    instrument_id char(4) not null,
    instrumentName text not null,
    instrumentFamily text not null,
    primary key(instrument_id)
);
Member_Of

Determines who is in which ensemble and if they play an instrument in it.

Fundamental Dependencies

\[ \text{person_id, ensemble_id} \rightarrow \text{instrument_id} \]

```
CREATE TABLE Member_of (
    person_id  char(8) not null references Musicians(person_id),
    ensemble_id char(5) not null references Ensembles(ensemble_id),
    instrument_id char(4) references Instruments(instrument_id),
    primary key(person_id, ensemble_id)
);
```
Can_Play

Stores which instruments each instrumentalist can play

<table>
<thead>
<tr>
<th>person_id</th>
<th>instrument_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>20083100</td>
<td>0001</td>
</tr>
<tr>
<td>20083100</td>
<td>0002</td>
</tr>
<tr>
<td>20083101</td>
<td>0004</td>
</tr>
<tr>
<td>20083102</td>
<td>0003</td>
</tr>
<tr>
<td>20083103</td>
<td>0006</td>
</tr>
<tr>
<td>20083104</td>
<td>0008</td>
</tr>
<tr>
<td>20083105</td>
<td>0010</td>
</tr>
</tbody>
</table>

Fundamental Dependencies
person_id → instrument_id

CREATE TABLE Can_Play (person_id char(8) not null references Instrumentalists(person_id), instrument_id char(4) not null references Instruments(instrument_id), primary key(person_id, instrument_id) );
Directs_In

References which people direct ensembles

Fundamental Dependencies

person_id $\rightarrow$ ensemble_id

```
CREATE TABLE Directs_In (  
  person_id char(8) not null references Directors(person_id),  
  ensemble_id char(5) not null references Ensembles(ensemble_id),  
  primary key(person_id, ensemble_id)
);
```
Locations

Stores the locations where ensembles can perform and rehearse in.

Fundamental Dependencies

\[\text{location}_\text{id} \rightarrow \text{roomName}, \text{maxCapacity}\]

```
CREATE TABLE Locations (  
    location_id  char(4) not null,  
    roomName  text,  
    maxCapacity  text,  
    primary key(location_id)  
);
```

<table>
<thead>
<tr>
<th>location_id</th>
<th>roomName</th>
<th>maxcapacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>Symphonic Hall</td>
<td>410</td>
</tr>
<tr>
<td>0002</td>
<td>Fusco Recital Hall</td>
<td>314</td>
</tr>
<tr>
<td>0003</td>
<td>Nelly Golleti Theatre</td>
<td>400</td>
</tr>
<tr>
<td>0004</td>
<td>River Rooms</td>
<td>350</td>
</tr>
<tr>
<td>0005</td>
<td>Bardavon</td>
<td>800</td>
</tr>
<tr>
<td>0006</td>
<td>CIA Recital Hall</td>
<td>900</td>
</tr>
<tr>
<td>0007</td>
<td>Longview Park</td>
<td>1000</td>
</tr>
<tr>
<td>0008</td>
<td>The Caberet</td>
<td>210</td>
</tr>
<tr>
<td>0009</td>
<td>HC2020</td>
<td>40</td>
</tr>
<tr>
<td>0010</td>
<td>Small Ensembles Room</td>
<td>100</td>
</tr>
</tbody>
</table>
Rehearses_In

References the locations where ensembles rehearse.

Fundamental Dependencies

ensemble_id → location_id, dayOfTheWeek, startTime, endTime

CREATE TABLE Rehearses_In (  
ensemble_id char(5) not null references Ensembles(ensemble_id),  
location_id char(4) not null references Locations(location_id),  
dayOfTheWeek text not null, --CheckRestraints  
startTime time not null,  
endTime time not null,  
primary key(ensemble_id)  
);
Performs_In

Stores the department’s music repertoire

### Fundamental Dependencies

\[ \text{lit}_\text{id} \rightarrow \text{songTitle, genre} \]

```sql
CREATE TABLE Literature (  
  lit_id char(5) not null,  
  songTitle text,  
  genre text,  
  primary key(lit_id)  
);
```

<table>
<thead>
<tr>
<th>lit_id</th>
<th>songTitle</th>
<th>genre</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>Sleigh Ride</td>
<td>Christmas</td>
</tr>
<tr>
<td>00002</td>
<td>The Marist College Fight Song</td>
<td></td>
</tr>
<tr>
<td>00003</td>
<td>The Marist Song</td>
<td></td>
</tr>
<tr>
<td>00004</td>
<td>The Great Locomotive Chase</td>
<td></td>
</tr>
<tr>
<td>00005</td>
<td>Everlong</td>
<td></td>
</tr>
<tr>
<td>00006</td>
<td>We Wish you a Merry Christmas</td>
<td>Christmas</td>
</tr>
<tr>
<td>00007</td>
<td>First Suite in E Flat</td>
<td></td>
</tr>
<tr>
<td>00008</td>
<td>Second Suite in F</td>
<td></td>
</tr>
<tr>
<td>00009</td>
<td>The Planets</td>
<td></td>
</tr>
<tr>
<td>00010</td>
<td>All Star</td>
<td></td>
</tr>
</tbody>
</table>
Songs_Playing

References which songs each ensemble is playing

```
CREATE TABLE Songs_Playing (
    ensemble_id char(5) not null references Ensembles(ensemble_id),
    lit_id      char(5) not null references Literature(lit_id),
    primary key(ensemble_id, lit_id)
);
```
v_Administrators

Returns a list of administrators containing their names, position, and contact information.

```
CREATE VIEW v_Administrators AS
    SELECT firstName, lastName, roleName, emailAddress
    FROM People, Administrators, Department_Roles
    WHERE people.person_id = administrators.person_id
    AND administrators.role_id = department_roles.role_id;
```

<table>
<thead>
<tr>
<th>firstame</th>
<th>lastame</th>
<th>rolename</th>
<th>emailaddress</th>
</tr>
</thead>
<tbody>
<tr>
<td>Art</td>
<td>Himmelberge</td>
<td>Director of Music and Director of Bands</td>
<td><a href="mailto:arthur.himmelberger@marist.edu">arthur.himmelberger@marist.edu</a></td>
</tr>
<tr>
<td>Sarah</td>
<td>Williams</td>
<td>Director of Choral Activities</td>
<td><a href="mailto:sarah.williams@marist.edu">sarah.williams@marist.edu</a></td>
</tr>
<tr>
<td>Mikey</td>
<td>Napolitano</td>
<td>Operations Manager</td>
<td><a href="mailto:michael.napolitano@marist.edu">michael.napolitano@marist.edu</a></td>
</tr>
</tbody>
</table>
v_BandMembers

Returns the names, instrument, and graduation semesters of each band student.

```
CREATE VIEW v_BandMembers AS
SELECT firstName, lastname, expectedGradSemester, instrumentName
FROM People p, Musicians m, Instrumentalists i, Can_Play, Instruments
WHERE p.person_id = m.person_id
AND m.person_id = i.person_id
AND i.person_id = Can_Play.person_id
AND Can_Play.instrument_id = Instruments.instrument_id
ORDER BY lastname ASC;
```

<table>
<thead>
<tr>
<th>firstname</th>
<th>lastname</th>
<th>expectedGradSemester</th>
<th>instrumentName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andrew</td>
<td>Bauman</td>
<td>Spring 2020</td>
<td>Trombone</td>
</tr>
<tr>
<td>Andrew</td>
<td>Bauman</td>
<td>Spring 2020</td>
<td>Euphonium</td>
</tr>
<tr>
<td>Marsha</td>
<td>Mellow</td>
<td>Fall 2019</td>
<td>Flute</td>
</tr>
<tr>
<td>Cadence</td>
<td>Smith</td>
<td>Spring 2018</td>
<td>Clarinet</td>
</tr>
<tr>
<td>I.M.</td>
<td>Tired</td>
<td>Fall 2020</td>
<td>Marimba</td>
</tr>
<tr>
<td>Adagio</td>
<td>Turner</td>
<td>Spring 2021</td>
<td>Harpsichord</td>
</tr>
<tr>
<td>Melody</td>
<td>Wind</td>
<td>Fall 2020</td>
<td>Baritone Sax</td>
</tr>
</tbody>
</table>
v_Singers

Returns the names, instrument, and graduation semesters of each student in Singers.

CREATE VIEW v_Singers AS
SELECT firstName, lastName, expectedGradSemester, vocalRange
FROM People p, Musicians m, Vocalists v
WHERE p.person_id = m.person_id
AND m.person_id = v.person_id
Order by lastname ASC;

<table>
<thead>
<tr>
<th>firstname</th>
<th>lastname</th>
<th>expectedgradsemester</th>
<th>vocalrange</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maureen</td>
<td>Biologist</td>
<td>Spring 2018</td>
<td>Baritone</td>
</tr>
<tr>
<td>Rita</td>
<td>Book</td>
<td>Spring 2019</td>
<td>Tenor</td>
</tr>
<tr>
<td>Johnny</td>
<td>Cash</td>
<td>Spring 2019</td>
<td>Alto</td>
</tr>
<tr>
<td>Ann</td>
<td>Chovey</td>
<td>Fall 2021</td>
<td>Alto</td>
</tr>
<tr>
<td>Marsha</td>
<td>Mellow</td>
<td>Fall 2019</td>
<td>Baritone</td>
</tr>
<tr>
<td>Isabelle</td>
<td>Ringing</td>
<td>Fall 2018</td>
<td>Soprano</td>
</tr>
</tbody>
</table>
MembersIn(ensemble_id, ‘results’)

CREATE OR REPLACE FUNCTION MembersIn(char(5), REFCURSOR) RETURNS refcursor AS $$
DECLARE
  inputNumber char(5) := $1;
  resultSet REFCURSOR := $2;
BEGIN
  OPEN resultSet for
  SELECT firstName, lastName, expectedGradSemester
  FROM People p, Musicians m, Member_of, Ensembles
  WHERE p.person_id = m.person_id
  AND m.person_id = member_of.person_id
  AND member_of.ensemble_id = ensembles.ensemble_id
  AND ensembles.ensemble_id = inputNumber;
  RETURN resultSet;
END;
$$
LANGUAGE plpgsql;

SELECT MembersIn('00001', 'results');
FETCH ALL FROM results;

<table>
<thead>
<tr>
<th>firstname</th>
<th>lastname</th>
<th>expectedgradsemester</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andrew</td>
<td>Bauman</td>
<td>Spring 2020</td>
</tr>
<tr>
<td>Cadence</td>
<td>Smith</td>
<td>Spring 2018</td>
</tr>
<tr>
<td>Melody</td>
<td>Wind</td>
<td>Fall 2020</td>
</tr>
</tbody>
</table>
**PerformsIn(location_id, ‘results’)***

CREATE OR REPLACE FUNCTION PerformsIn(char(4), REFCURSOR) RETURNS refcursor AS $$

DECLARE
    inputNumber char(4) := $1;
    resultSet REFCURSOR := $2;

BEGIN
    OPEN resultSet for
        SELECT eventDate, eventTime, ensembleName
        FROM Ensembles e, Performs_In, Locations l
        WHERE l.location_id = performs_In.location_id
        AND e.ensemble_id = performs_In.ensemble_id
        AND l.location_id = inputNumber;

    RETURN resultSet;

END;
$$

LANGUAGE plpgsql;

Returns a list of the ensembles performing in a specified location along with their respective date and times.

---

<table>
<thead>
<tr>
<th>eventdate</th>
<th>eventtime</th>
<th>ensemblename</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-12-02</td>
<td>14:00:00</td>
<td>Symphonic Band</td>
</tr>
<tr>
<td>2017-12-03</td>
<td>14:00:00</td>
<td>Orchestra</td>
</tr>
</tbody>
</table>
Returns a list of where and when a given ensemble is practicing in.
In this case, Symphonic Band.

```
SELECT dayOfTheWeek, startTime, endTime, roomName
FROM Ensembles e, Rehearses_In r, Locations l
WHERE l.location_id = r.location_id
AND e.ensemble_id = r.ensemble_id
AND e.ensemble_id = '00001';
```

Returns a list of the songs a given ensemble is playing.
In this case, Symphonic Band

```
SELECT songTitle
FROM Literature l, Songs_Playing s, Ensembles e
WHERE l.lit_id = s.lit_id
AND s.ensemble_id = e.ensemble_id
AND e.ensemble_id = '00001';
```

<table>
<thead>
<tr>
<th>dayoftheweek</th>
<th>starttime</th>
<th>endtime</th>
<th>roomname</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday</td>
<td>20:00:00</td>
<td>22:00:00</td>
<td>Symphonic Hall</td>
</tr>
</tbody>
</table>

- Sleigh Ride
- The Marist College Fight Song
- First Suite in E Flat
Operation: Save Alan

This trigger prevents anyone from removing anyone named Alan Labouseur from the database.

CREATE OR REPLACE FUNCTION saveAlan() RETURNS TRIGGER AS $$
BEGIN
    IF OLD.firstName = 'Alan' AND OLD.lastName = 'Labouseur'
    THEN RAISE EXCEPTION 'CANNOT DELETE ALAN, THE GREATEST DATABASE PROFESSOR WHO EVER LIVED';
    END IF;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER savingAlan BEFORE DELETE ON People
FOR EACH ROW EXECUTE PROCEDURE saveAlan();

DELETE FROM People
where lastName = 'Labouseur';

ERROR: CANNOT DELETE ALAN, THE GREATEST DATABASE PROFESSOR WHO EVER LIVED
CONTEXT: PL/pgSQL function savealan() line 4 at RAISE
********** Error **********
The administrators should have the ability to modify all tables as well as to create new views and stored procedures to be used by other users.

The office staff should have the ability to query data as well as upgrade records, they should not have the ability to delete records.

Band and Singers eBoard members should have the ability to generate rosters, view the music library, and check to make sure rehearsals and performances are scheduled. They should not be capable of viewing administrators information.
The purpose of this database system is to manage many of the daily bookkeeping tasks for the Marist College Music Department.

If this database were fully implemented, person_id would most likely be replaced by a CWID number, which it is already equipped to handle.
As of this moment, there is no way to confirm that the instruments a student is playing in an ensemble is one of the instruments they have reported knowing how to play.
This is a small-scale representation of what the larger, fully developed database would look like.

If someone were to continue developing this project, I would imagine the database would have **more check constraints** to ensure the database remain normalized. Additionally **more views and stored procedures** should exist to allow for a simplified user experience. The developer can also **take into account semesters**, in that, some ensembles may not be offered every semester. **Additional data could be tracked for Literature**, such as instrument parts, how many times a piece has been performed, the last time a piece has been performed, etc. The database could also begin to handle **time clock and payroll information for office staff members**.