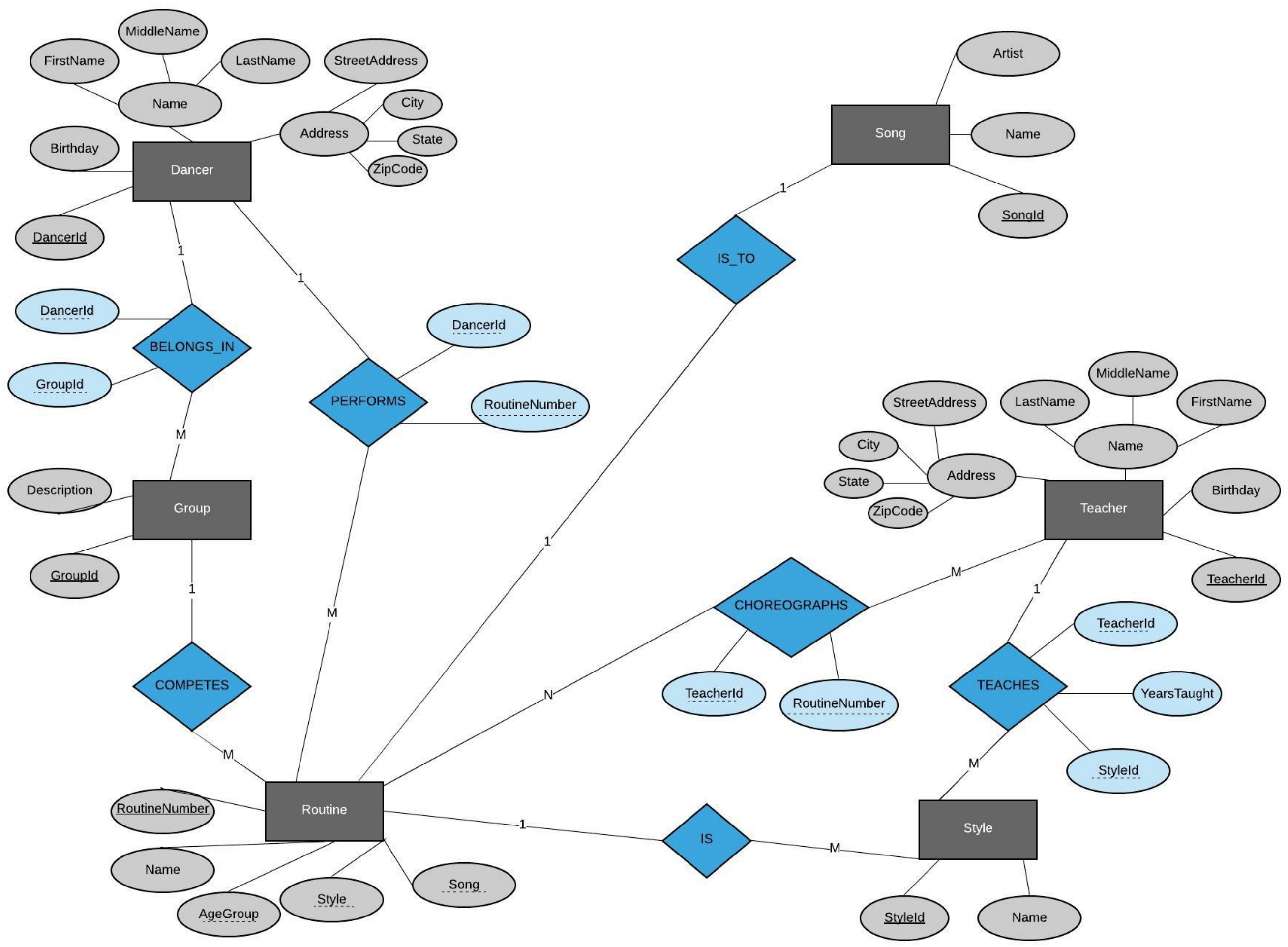
Dance Studio Organization Plan

Marissa Manata

Synopsis: A dance studio owner/director wanted to be able to keep track of all of their dancers, all of their teachers, and all of the routines for that year. They wanted to be able to tell what dancers are in which routines and what teacher(s) choreographed that routine. When it comes to the routines, they wanted to know what style, what song, and what group goes with that routine. They, also, wanted to know what styles a teacher taught and how long they have taught that style. The owner/director wanted to know a dancer's name, birthday, and address. They wanted to know a teacher's same information. A dancer could be in multiple age groups and could be in multiple routines during a year. A teacher could choreograph multiple routines during a year. One routine could only be to one song and could only be one style, but one style could be used by multiple routines. Teachers could teach multiple different styles for different time frames.

Database:



- *Dancer*: This table stores the data for each dancer. The reason that the name, birthday, and address of the dancer is stored is to have the information that the owner/director wants in order to keep a file on the students. The data has been used to put the dancer in groups and to send information to their house.
- *Teacher*: This table stores data for each teacher. The reason that the name, birthday, and address of the teacher is stored is to have the information that the owner/director wants to keep a file on the teacher. The data has been used to send paychecks to the teacher and see if there are certain age groups the teacher would work better with because of age.
- *Style*: This table holds the different styles that a dance can be. It just stores the name of the style because that is all that is needed. This information has been used to tell what styles the teachers teach and what style different routines are.
- *Routine*: This table holds all of the routines that the studio is performing that year. It stores the information such as the name of the routine and the song to tell the routines apart. The age group and style are provided to help with classification for competitions.
- *Group*: This table holds the different age groups that a dancer can be in. It stores a description to help the owner/director place dancers in a group that fits with their experience and age. It, also, helps classifying routines at competitions.
- *Song*: This table holds the songs a studio is using. It holds the name of the song and the artist, so that there are no repeats of the song. This information has been used with a routine to tell different dances apart from others.
- *BELONGS_IN*: This table is a relationship between the dancer and group tables. It holds the id number of the dancer and the group. This has been used to tell the dancer's age and ability. A dancer can be in multiple groups, but not perform in all the dances that each group performs.
- *PERFORMS*: This table is a relationship between the dancer and routine tables. It holds the id number of the dancer and the routine number in order to tell what dancer is in which routine since not every dancer is in every routine but can be in multiple routines.
- *CHOREOGRAPHS*: This table is a relationship between the teacher and routine tables. It holds the id number of the teacher and the routine number in order to tell what teacher choreographed which routine. Multiple teachers can choreograph a routine together, which is why it is a one-to-many relationship.
- *TEACHES*: This table is a relationship between the teacher and style tables. It holds the id number of the teacher and the style, as well as, the number of years the teacher has been teaching that style. This information has been kept in order to tell the experience of the teachers and can help decide what groups they should choreograph for.

Functionality:

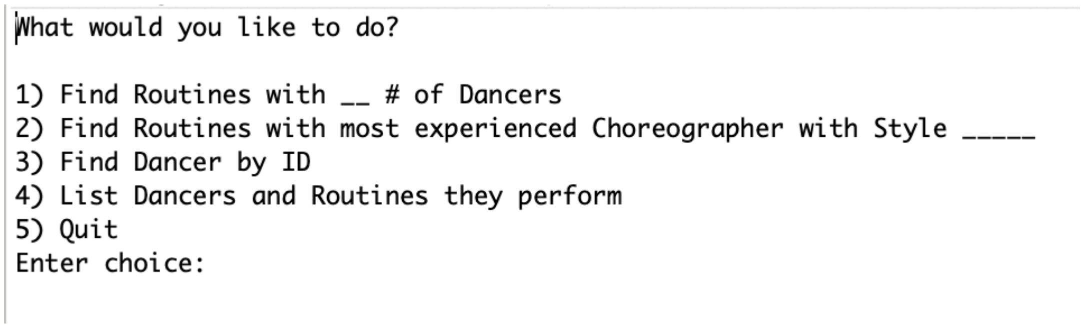
- Is able to group by dancer to find out what routine(s) they are in
- Is able to group by routine to find out what routines have a certain number of dancers
- Is able to group by routines of a certain style that were choreographed by the teachers with the most experience

Stakeholders: The user(s) are dance studio owners and/or directors who need a little more organization within their studio. The use of the database may change depending on the number of dancers within a studio or how many routines the studio maybe competing versus not competing.

Technological Requirements: I made this as a desktop application that uses SQLite and Java. I used the JDBC as a connector. I have had about 3 years of Java experience now and can use it pretty well. While I am still learning how to use SQLite, I do have experience with SQL programming before and as I learn more the easier it becomes to me.

Screenshots:

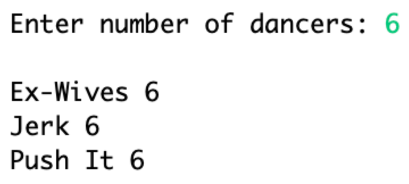
- Menu Screen



```
What would you like to do?

1) Find Routines with __ # of Dancers
2) Find Routines with most experienced Choreographer with Style _____
3) Find Dancer by ID
4) List Dancers and Routines they perform
5) Quit
Enter choice:
```

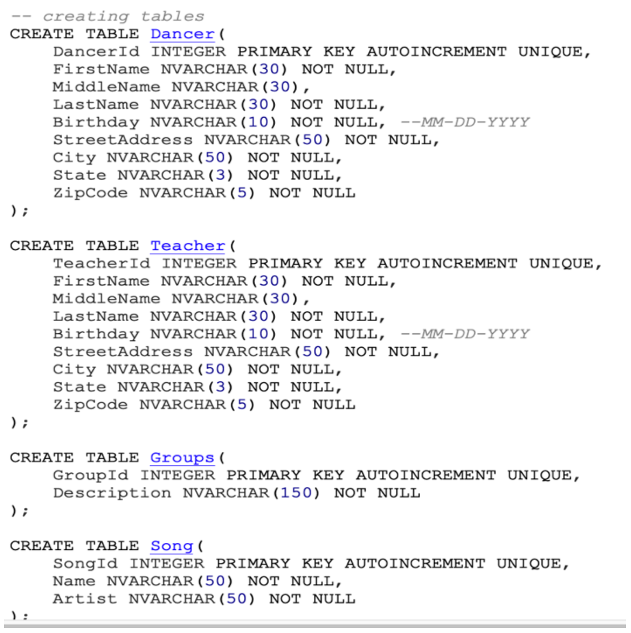
- Number of Dancer Output



```
Enter number of dancers: 6

Ex-Wives 6
Jerk 6
Push It 6
```

- Part of Script



```
-- creating tables
CREATE TABLE Dancer(
    DancerId INTEGER PRIMARY KEY AUTOINCREMENT UNIQUE,
    FirstName NVARCHAR(30) NOT NULL,
    MiddleName NVARCHAR(30),
    LastName NVARCHAR(30) NOT NULL,
    Birthday NVARCHAR(10) NOT NULL, --MM-DD-YYYY
    StreetAddress NVARCHAR(50) NOT NULL,
    City NVARCHAR(50) NOT NULL,
    State NVARCHAR(3) NOT NULL,
    ZipCode NVARCHAR(5) NOT NULL
);

CREATE TABLE Teacher(
    TeacherId INTEGER PRIMARY KEY AUTOINCREMENT UNIQUE,
    FirstName NVARCHAR(30) NOT NULL,
    MiddleName NVARCHAR(30),
    LastName NVARCHAR(30) NOT NULL,
    Birthday NVARCHAR(10) NOT NULL, --MM-DD-YYYY
    StreetAddress NVARCHAR(50) NOT NULL,
    City NVARCHAR(50) NOT NULL,
    State NVARCHAR(3) NOT NULL,
    ZipCode NVARCHAR(5) NOT NULL
);

CREATE TABLE Groups(
    GroupId INTEGER PRIMARY KEY AUTOINCREMENT UNIQUE,
    Description NVARCHAR(150) NOT NULL
);

CREATE TABLE Song(
    SongId INTEGER PRIMARY KEY AUTOINCREMENT UNIQUE,
    Name NVARCHAR(50) NOT NULL,
    Artist NVARCHAR(50) NOT NULL
);
```

Advanced Queries:

a) Fulfills Group 1
b) This query shows the name of the dancer and routine name of all of the routines they are in.

```
SELECT FirstName, LastName, Name
  FROM Dancer JOIN PERFORMS JOIN Routine
    ON Dancer.DancerId = PERFORMS.DancerId
   AND PERFORMS.RoutineNumber = Routine.RoutineNumber
ORDER BY Dancer.FirstName, Dancer.LastName;
```

a) Fulfills Group 2
b) This query shows the name of the routine and the number of dancers in the routine when the number of dancers is equal to the user input.

```
SELECT Name, count(*) AS NumDancers
  FROM (SELECT Name, DancerId
          FROM PERFORMS JOIN Routine
            ON PERFORMS.RoutineNumber = Routine.RoutineNumber
        ORDER BY Name) AS DancerCount
GROUP BY Name
HAVING count(*) = ?
```

a) Fulfills Group 3
b) This query shows the routine name, the most experienced teacher name, and style of routine for the style that the user identifies.

```
SELECT Routine.Name AS RoutineName, FirstName, LastName, StyleName
  FROM (SELECT Teacher.TeacherId AS Teacher, FirstName, LastName, Style.StyleId, Style.Name AS StyleName, YearsTaught
          FROM Teacher JOIN TEACHES JOIN Style
            ON Teacher.TeacherId = TEACHES.TeacherId
           AND TEACHES.StyleId = Style.StyleId
        GROUP BY Name
        HAVING max(YearsTaught) AND Style.Name = ?) AS Experienced JOIN CHOREOGRAPHS JOIN Routine
    ON Experienced.Teacher = CHOREOGRAPHS.TeacherId
   AND CHOREOGRAPHS.RoutineNumber = Routine.RoutineNumber
   AND Experienced.StyleId = Routine.Style
GROUP BY Routine.Name
```