Project Step 2 Draft

Group 4: High Strung
Group Members: Michael McGuan and Derek Woodard
Project Title: Music Studio Administration System

Fixes based on Feedback from Step 1

Approved:

- Removed “Payments” table completely (feedback from TA on Step 1 Draft submission).
- Changed “People” table to “Student” table. While we had originally rejected this suggestion for our Step 1 Draft submission, by removing the “Payments” table, we no longer have a need to differentiate between types of people; everyone is now considered a student (feedback from Paige Wiley).
- More detail about assignments and pieces was added to the overview so that it now mentions all entities named in the outline. Examples of the kinds of reports a user would like to see have also been included (response to feedback from Eric Peters).
- The purpose of the Lessons entity is to track attendance. The description of Lessons in the outline has been revised to make this function more clear (feedback from Eric Peters).
- Corrections made to *PersonID* and *PaymentID*. The first is now known as *studentID,* but *paymentID* is no longer applicable as a result of removing the “Payments” table completely from our design (similar feedback from Eric Peters, Noah Kiedaisch and Michael Zardonella).

Not Approved:

- No action was taken on the suggestion to make junction table names plural. It is a widely-accepted and standard convention to keep junction table names singular (feedback from Paige Wiley).
- No action was taken on the suggestion to reposition lines to show a more clear view of the relationships. We intentionally positioned the lines to prioritize readability, showing that relationships are clearly defined without creating unnecessary visual clutter (feedback from Eric Peters).

Our Design Decision Changes

- As a result of the above suggestions, any table containing the word *person* or *people* has been changed to *student* for consistency.
    - Changed table name “People” to “Student”
    - Changed table name “PersonPiece” to “StudentPiece”
    - Changed table name “PersonInstrument” to “StudentInstrument”

    - Changed all foreign keys from *personID* to *studentID*.

- Made several changes to Student table for better readability and flow of logic:
    - Removed *isParent* attribute
    - Removed *isStudent* attribute
    - Removed *familyID* attribute
    - Removed *dateStarted* attribute; this was redundant, as this date can be referenced as *date* in the "Lessons" table
    - Added *parentFirstName* and *parentLastName* attributes

- Removed the following relationship from the "Students" table as the "Payments" table no longer exists:
    - A 1:M relationship between Student and Payments is implemented with studentID as a foreign key in Payments.

- During the normalization process, it was noted that there was a transitive dependency involving zipCode, city, and state in the Students table. The city and state of an address are functionally dependent on the zip code, which in turn depends on the studentID. We have elected not to put this table into 3rd normal form because the benefits do not outweigh the additional complexity of creating a separate zip code table.

Overview

Independent music instructors may have studios with as many as 40 private students, each of whom is studying three to four different pieces at a time. A teacher may offer instruction in four or more musical instruments, and a student may be learning multiple instruments as well. Motivated by the need to simplify the bookkeeping tasks involved in running a studio, the system proposed here is an interactive web page driven by a database that will store data related to a music teacher's private studio.

Teachers frequently need to communicate with parents regarding scheduling and billing, and the system will meet this need by maintaining up-to-date contact information. Lesson attendance will also be tracked, with a record of every lesson stored in the database. One of a teacher's responsibilities is choosing musical pieces for students to learn, and the system will facilitate repertoire selection by providing a searchable list of pieces in a teacher's library. Each piece will be linked to its composer and instrument. The teacher will be able to add multiple assigned pieces to each student, and it will also be possible to produce a list of all students who have studied a given piece.

With the administrative details of managing a small studio streamlined and consolidated, users of our solution will be able to provide better quality music instruction to their students.

Database Outline

- **Students:** holds data about current and former students. In this project, it is assumed that no parent is also a student.
  - studentID: INT, AUTO_INCREMENT, UNIQUE, NOT NULL, PK
  - firstName: VARCHAR(50), NOT NULL
  - lastName: VARCHAR(50), NOT NULL
  - streetAddress: VARCHAR(255)
  - city: VARCHAR(50)
  - state: VARCHAR(50)
  - zipCode: VARCHAR(15)
  - phone: VARCHAR(15)
  - email: VARCHAR(100)
  - parentFirstName: VARCHAR(50)
  - parentLastName: VARCHAR(50)
  - isActive: TINYINT(1), NOT NULL DEFAULT 1
  - Relationships:
    - A M:M relationship between Student and Instruments is implemented with studentID as a foreign key in the junction table StudentInstrument.
    - A M:M relationship between Student and Pieces is implemented with studentID as a foreign key in the junction table StudentPiece.
    - A 1:M relationship between Student and Lessons is implemented with studentID as a foreign key inside Lessons.

- **Lessons:** holds data about the lessons a teacher has given. A record in this table indicates that a student attended a lesson on the date specified. The system does not track missed or canceled lessons.
  - lessonID: INT, AUTO_INCREMENT, UNIQUE, NOT NULL, PK
  - studentID: INT, NOT NULL, FK
  - instrumentID: INT, NOT NULL, FK
  - date: DATE, NOT NULL
  - comment: VARCHAR(150)
  - Relationships:
    - A M:1 relationship between Lessons and Student is implemented with studentID as a foreign key inside Lessons
    - A M:1 relationship between Lessons and Instruments is implemented with instrumentID as a foreign key inside Lessons.

- **Instruments:** holds data about musical instruments.
  - instrumentID: INT, AUTO_INCREMENT, UNIQUE, NOT NULL, PK
  - name: VARCHAR(50), NOT NULL
  - comment: VARCHAR(150)
  - Relationships:
    - A M:M relationship between Student and Instruments is implemented with instrumentID as a foreign key in the junction table StudentInstrument.
    - A 1:M relationship between Instruments and Pieces is implemented with instrumentID as a foreign key in Pieces.

        - A 1:M relationship between Instruments and Lessons is implemented with instrumentID as a foreign key in Lessons.

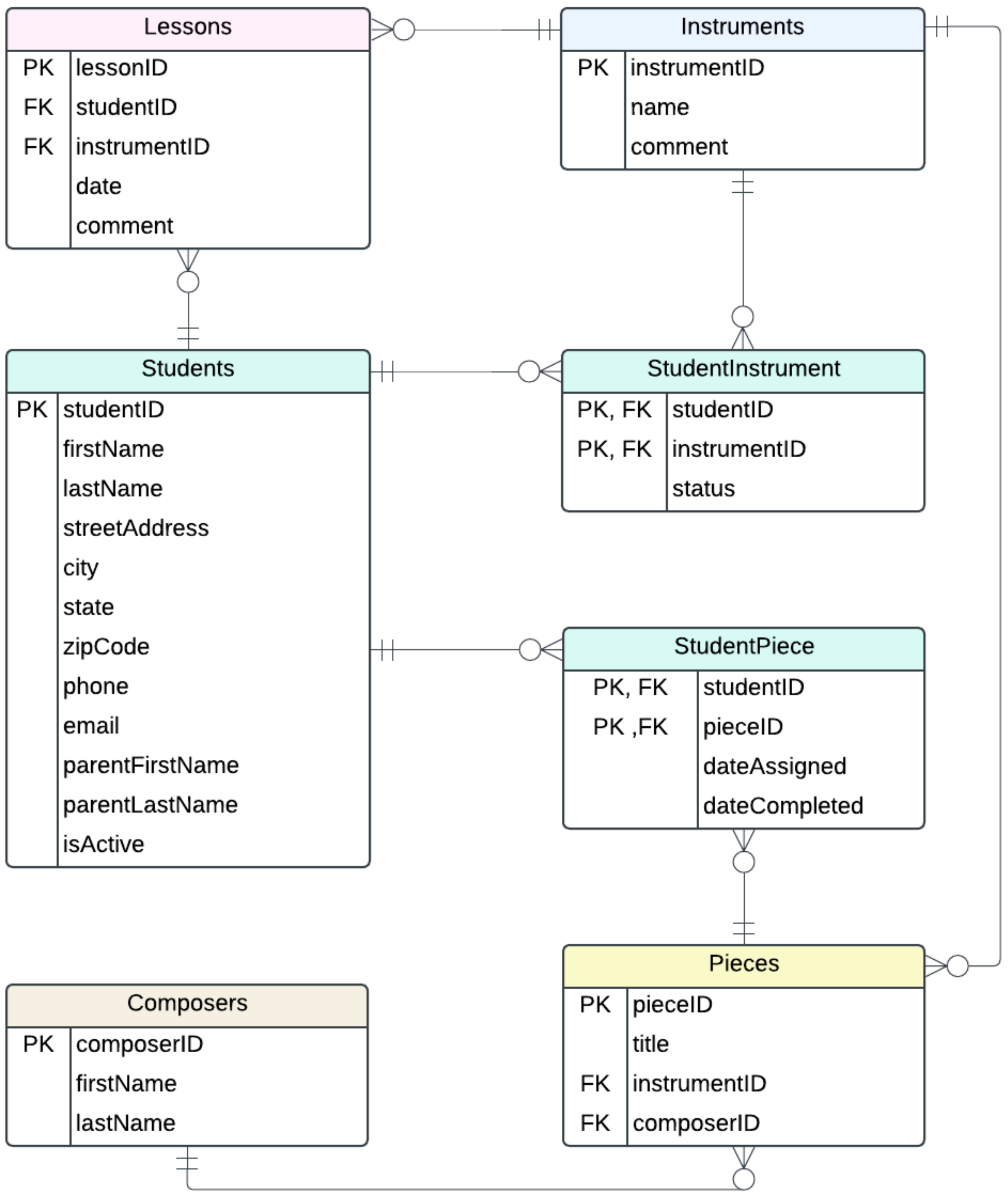
- **StudentInstrument:** tracks which students play which instruments.
    - studentID: INT, NOT NULL, FK (composite PK)
    - instrumentID: INT, NOT NULL, FK (composite PK)
    - status: ENUM(‘Owned’, ‘Rented’), NOT NULL
    - Relationships: A M:M relationship between Student and Instruments is implemented with studentID and instrumentID as foreign keys in this table.

- **StudentPiece:** tracks the pieces assigned to each student.
    - studentID: INT, NOT NULL, FK (composite PK)
    - pieceID: INT, NOT NULL, FK (composite PK)
    - dateAssigned: DATE, NOT NULL
    - dateCompleted: DATE
    - Relationships: A M:M relationship between Student and Pieces is implemented with studentID and pieceID as foreign keys in this table.

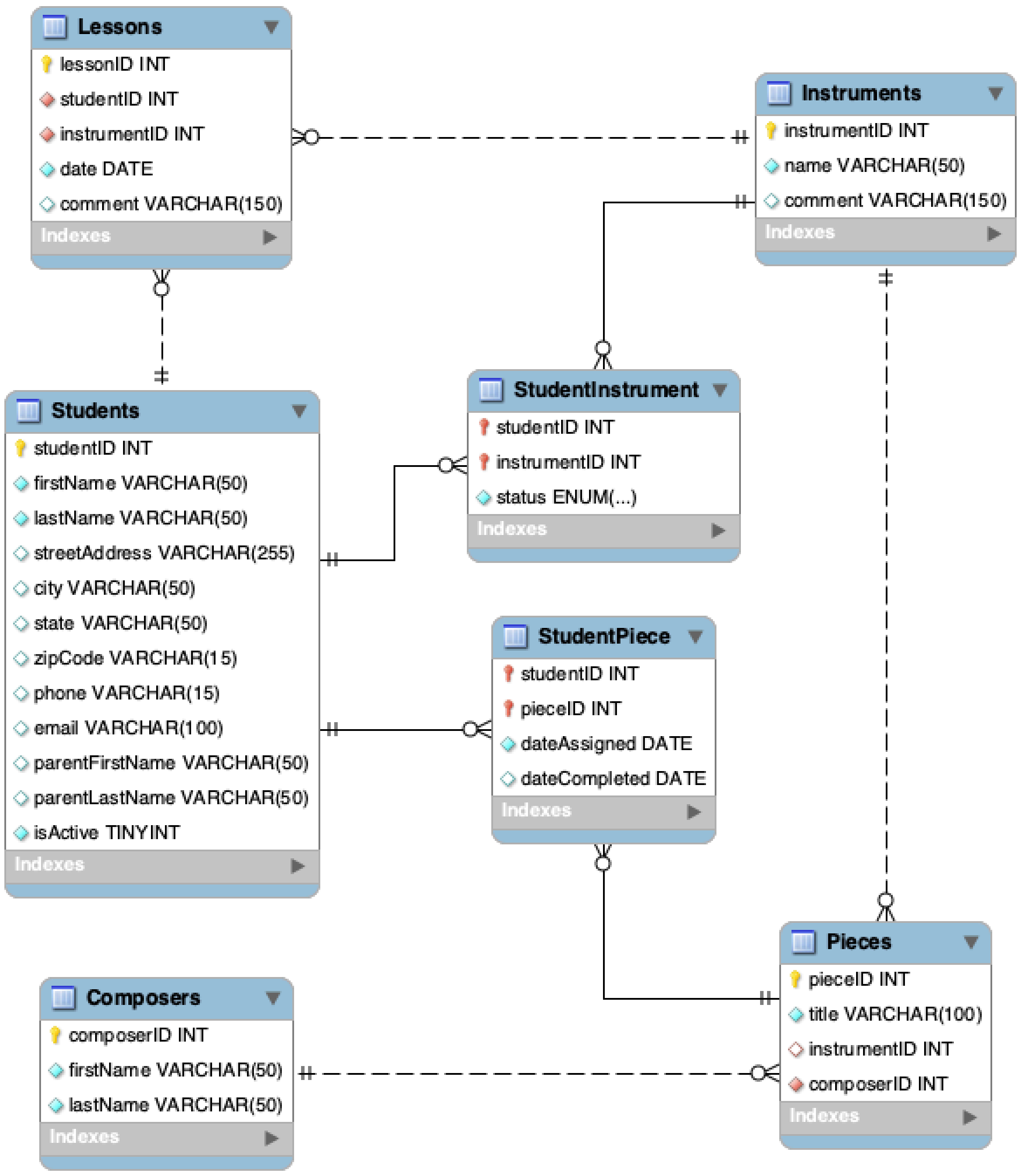
- **Pieces:** holds data about musical pieces. In this project, a piece is related to one composer and one instrument. The database does not store information on lyricists and arrangers.
    - pieceID: INT, AUTO_INCREMENT, UNIQUE, NOT_NULL, PK
    - title: VARCHAR(100), NOT NULL
    - instrumentID: INT, NOT NULL, FK
    - composerID: INT, NOT NULL, FK
    - Relationships:
        - A M:1 relationship between Pieces and Composers is implemented with composerID as a foreign key in Pieces.
        - A M:1 relationship between Pieces and Instruments is implemented with instrumentID as a foreign key in Pieces.
        - A M:M relationship between Pieces and Student is implemented with pieceID as a foreign key in the junction table StudentPiece.

- **Composers:** holds data about musical composers.
    - composerID: INT, AUTO_INCREMENT, UNIQUE, NOT NULL, PK
    - firstName: VARCHAR(50), NOT NULL
    - lastName: VARCHAR(50), NOT NULL
    - Relationships: A 1:M relationship between Composers and Pieces is implemented with composerID as a foreign key in Pieces.

Entity-Relationship Diagram



Schema


Lessons
lessonID INT
studentID INT
instrumentID INT
date DATE
comment VARCHAR(150)
Indexes
Instruments
instrumentID INT
name VARCHAR(50)
comment VARCHAR(150)
Indexes
StudentInstrument
studentID INT
instrumentID INT
status ENUM(...)
Indexes
Students
studentID INT
firstName VARCHAR(50)
lastName VARCHAR(50)
streetAddress VARCHAR(255)
city VARCHAR(50)
state VARCHAR(50)
zipCode VARCHAR(15)
phone VARCHAR(15)
email VARCHAR(100)
parentFirstName VARCHAR(50)
parentLastName VARCHAR(50)
isActive TINYINT
Indexes
StudentPiece
studentID INT
pieceID INT
dateAssigned DATE
dateCompleted DATE
Indexes
Pieces
pieceID INT
title VARCHAR(100)
instrumentID INT
composerID INT
Indexes
Composers
composerID INT
firstName VARCHAR(50)
lastName VARCHAR(50)
Indexes

Example Data

Students table

studentID	firstName	lastName	streetAddress	city	state	zipCode	phone	email	parentFirstName	parentLastName	isActive
1	Jonathan	McCoy	7556 Elgin St	Irving	Iowa	47125	(594) 273-5475	jonathan.mccoy@example.com	Roberta	Elliot	1
2	Gary	Collins	8748 Adams St	Los Lunas	New Jersey	29784	(972) 960-8131	gary.collins@example.com	Marcia	Collins	1
3	Johnny	Fleming	4623 E Sandy Lake Rd	Princeton	Maine	78296	(577) 536-0700	johnny.fleming@example.com	Gloria	Fleming	0
4	James	Jones	6546 Spring Hill Rd	Fort Collins	North Carolina	21782	(847) 304-0243	james.jones@example.com	Grace	Jones	0
5	Kristin	Bryant	8251 Sunset St	Topeka	Nevada	16396	(655) 547-8159	kristin.bryant@example.com	Andy	Bryant	1
6	Bessie	Douglas	4449 Cackson St	El Monte	Nevada	46077	(947) 330-4671	bessie.douglas@example.com	NULL	NULL	1

Instruments table

instrumentID	name	comment
1	Violin	NULL
2	Guitar	NULL
3	Piano	NULL

Lessons table

lessonID	studentID	instrumentID	date	comment
1	1	1	2025-01-20	NULL
2	2	1	2025-01-27	NULL
3	5	3	2025-01-27	Begin learning Mozart 2nd mov.
4	6	2	2025-01-29	NULL
5	1	1	2025-01-27	NULL
6	2	2	2025-01-27	NULL

Composers table

composerID	firstName	lastName
1	Johann Sebastian	Bach
2	Wolfgang Amadeus	Mozart
3	Fernando	Sor
4	Max	Bruch

Pieces table

pieceID	title	instrumentID	composerID
1	Partita No.3 in E major, BWV 1006	1	1
2	Invention in C major, BWV 772	3	1
3	Scottish Fantasy, Op.46	1	4
4	Sonata No.16 in C major, K.545	3	2
5	Introduction and Variations on a Theme by Mozart, Op.9	2	3

StudentInstrument table

studentID	instrumentID	status
1	1	Owned
2	1	Rented
2	2	Rented
5	3	Owned
6	2	Owned

StudentPiece table

studentID	pieceID	dateAssigned	dateCompleted
1	1	2024-12-09	NULL
1	3	2024-11-11	NULL
2	1	2024-10-28	2025-01-27
2	5	2024-11-04	NULL
5	2	2024-11-04	NULL
5	4	2024-11-11	NULL
6	5	2024-06-11	2024-09-23