Project Report - Task 1

Data Storage Paradigms, IV1351

Rozhan Asadi - rozhana@kth.se

Fall term 2022

1 Introduction

The purpose of this project is to make a database to facilitate information handling and business transactions for the Soundgood music school company. Music good company is a music institute offering music lessons to students and it needs a database for keeping track of its data about the lessons, instruments, students and instructors. In the current task (task 1), we will be focusing on making a conceptual model for the future database so that we can start making the actual database in future tasks. This project was done by only one person (Rozhan Asadi) and not with a project partner.

To get started, we need to discuss what is a conceptual model. In short, conceptual model is an abstract, psychological representation of a system or a task. Calendars and diaries can be good example of mental models for creating appointments. The concepts in this type of model help people understand and simulate the subject that the model is about.

Now that we have an introduction about conceptual model, it is easier to discuss the conceptual model which was made for this project.

The changes that were needed

This task was submitted for the first seminar, and after getting the review from the peer review and the graded submission review, this task was changed and improved to fit the review criteria better.

Since the attribute types are not needed in a conceptual model, they were removed from the model. The class "student" was modified so it would not have too many attributes. The current attributes in the student class are all needed. The relations got named and more notes were written and added to the model to make it easier to understand. A class named "contact person" was added since it was missing. The classes student and instructor are related to lessons.

2 Literature Studies

To understand conceptual models and be able to make one for this project, first the "conceptual model" information page in Canvas was studied really carefully. In that page there was tutorial videos and guides that was watched. Then the "tips-and-tricks-task1" document was read to become more prepared. The resources which were given on canvas were enough to get started but to continue more information was needed. So, it was time for searching in internet and learning more about databases, conceptual models and crow feet notations. After reading from several websites and watching YouTube tutorials, making the conceptual model was started.

3 Method

As it was mentioned, to make a conceptual model there are steps that one needs to follow. This section will be explaining each step to give a better view about conceptual models.

The first step is "noun identification". In this step, you need to start reading the problem explaining text and mark every word which was a noun. Every noun in the text can be counted as a class and be declared as a class in Astah (or any other application or website which is used for making an ER diagram. Astah was used in case of this project since we have Astah tutorials on Canvas.).

After making a class for every noun in the text, it is time for the next step which is "using a category list". There is a category list which is made from possible categories that a class can be from. In this step we read the list and see if any of the nouns in the list are related to the subject of the problem and make those those nouns a new class. When we end this step we are supposed to have many many classes in our diagram.

Now that we have many classes, it is time to delete the unnecessary classes from the diagram. We go through all classes in the diagram and delete any class that can not be an actual class itself. Unnecessary and useless classes aree gone after this step.

Then we move to the next step which is "finding attributes". We look at each class and try to come up with attributes that are relevant and needed for the class and add the attribute as a new attribute in the class. In this step, all needed attributes are added to the diagram.

For the next step we connect the classes together with connections or also called relations. Looking at every two class, we question if there is a relation between them and if there is, we put a relation between them by a crow feet notation arrow/line. At the end we look at the whole diagram and see if it is effectively showing what the problem is about and how the database will look like. If some changes or additions are needed then we fix the model until we get to a conceptual model that satisfies us.

4 Result

In this section the conceptual model that was made in this project can be seen(Figure 1). The file of the project can also be seen in the personal Git repository with the link : Click Link.

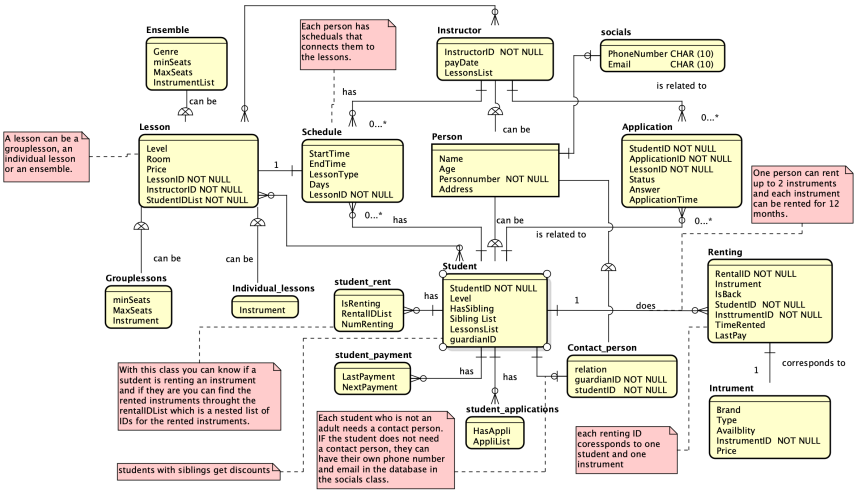

Figure 1: Conceptual model for the sound good music school database

It was tried to have the number of classes as minimum as possible. Two key classes of this model are "person" and "lesson". The class "person" is the parent of three classes "Student", "Instructor" and "Contact Person" and these classes inherit attributes from "person" A person in our database can be a student, an contact person or even an instructor. A person can have zero or 1 socials and by the "socials" we mean the phone number and contact info for that person.

For each student the data for their level, if they have siblings, the nested list of the IDs for their siblings, the nested list of the lessons they take part it and their guardian's (contact person's) ID. If the "HasSibling" attribute is TRUE then for finding the siblings we can check the nested list of "Sibling list". When a student rents an instrument, the instrument's ID is saved in the nested list of class "student-rent" and then the information of the renting can be accessed by the renting ID. Each "renting" corresponds to one instrument in the "Instrument" class where the brand, type, availability and the price can be viewed. If the student is a minor and needs a guardian as a contact person, the data of the person can easily be stored in the table "contact person".

The list of lessons ID can connect the student to the schedule of that lesson. The schedule shows the days of the class, the start time of the class, the end time of the class and the lesson's type.

To apply for lessons, the students make applications and those applications can be accessed by the instructors.

The sibling discounts will be handled by the system and there is no need to save it in the database. To calculate the price that a student needs to pay, the system can easily check the sibling list in the database.

5 Discussion

In this section, we will be discussing why this conceptual model can be a good model for the database. This Conceptual model includes all needed data from the music school in order to help the staff to run the school and manage the data. Although this model has every necessary data, it does not include unnecessary re-mentioning and waste of space. It was tried to keep the database as clear, efficient and useful as possible.

This conceptual model was improved after getting the feedback from the task 1 review and seminar 1. Every feedback about the model was paid attention to and changes were made based on the comments.

Every necessary attribute exist in the model and the attributes such as IDs that should not be NULL are marked as NOT NULL in the model. Making sure the ID attributes are not missing or NULL helps with keeping our database together. The IDs are what connects the tables/classes together like a key and losing keys or having NULL as a key will make the database to get into trouble and not be able to function well.

For this model, the IE notation was used along with marking numbers in the relation that was needed. The relations and the classes can be seen in figure 1. For clearer information, check GitHub repository.

5.1 Inheritance

To make this database more efficient, inheritance was used. If inheritance was not used then the attributes which used to be in the parent class, had to be in the child classes and the attributes would be repeated unnecessarily. In that way while the number of classes in the model would be less, the number of attributes would be many more. That can make handling the data harder and slower. It also takes more space to have more attributes in the database which will be a waste of space and money since more space is needed to be bought or rented to be available for the database.

Inheritance provides code re-usability since we can create classes that are built upon existing classes. It is a making a new implementation while maintaining the same behaviors which is important and useful when we want to manage the data. In our case, let's say the the music schools decides to make a new set of classes. To add the new form of classes to our database, we can re-use what written for the lesson class in general and then add attributes for that specific kind of class.

Inheritance adds flexibility to the database and gives it a better data structure which is more manageable. The other advantage of using inheritance in databases is that we can hide a part of data or make it not changeable. Let's

say we want to keep some data private and we do not want to make it alterable while we want to make another data public. Having the data in classes and sub-classes helps with restricting access to data to only classes that need that class to continue their functioning.

On the other hand, if the data in the base class are left unused that can lead to waste of memory and space which is a disadvantage. Also, if inheritance is not used properly, it can lead to wrong solutions and problems in the database.