Logical Model

Data Storage Paradigms, IV1351

Robert Furuvald, rfu@kth.se

2023-01-10

1 Introduction

This task consists of developing a logical model based on the conceptual model that was developed in the previous assignment. The logical model shall consist with enough physical attributes to be able to create a database. The diagram must have crow foot notation, and create the database based on the model.

2 Literature Study

For this task the author watched the lectures given about logical and physical models read chapter 14 and 15 that covered logical and physical models and normalization and all the different steps in developing a good relation database schema. For the SQL scripts the author used a generator and watched the lecture on sql. I have worked on my own.

3 Method

The first step in this assignment was to make a logical model that illustrates the database for *Soundgood Music School* based on the conceptual model from the previous assignment. There are some steps to follow in order to make the model as follows:

1. Deciding what relations should be used
2. Specify column type
3. Add column constraints
4. Check model is normalized
5. Set attributes
6. Set relations cardinality

The first step consists of deciding which relations should be used in the logical model. The first thing was to create a table for each entity from the conceptual model and then create the columns that consisted with a cardinality with 0..1 and 1..1. Attributes with a

cardinality of one-to-many or many-to-many will have an own table. For example, the author created a one-to-many relations between student and sibling. This is because there can be multiple students for each sibling.

The second step was to set a type for each column. The author where using PostgreSQL and therefore SERIAL was used for all primary key which is used to find a specific row in the table. The other types that where used was varchar, date and time. Next step is to add column constraints which is to decide what should happen when certain actions are performed like what should happen if some removed a phone number from a person, in this model the author uses ON DELETE CASCADE.

The fourth step is to verify that the model is normalized and that all operations will be possible to perform.

The fifth step is to set all attributes to the model which are the same as the conceptual model except from the attributes that became an own table, like phone in my model.

The last step is to set the cardinality for all relations.

4 Result

The SQL script was created by using Astah's export function and then modified to add ON DELETE CASCADE. Git where the scripts can be found for creating and inserting fake data is
The result of the logical model can be seen in figure 4.1

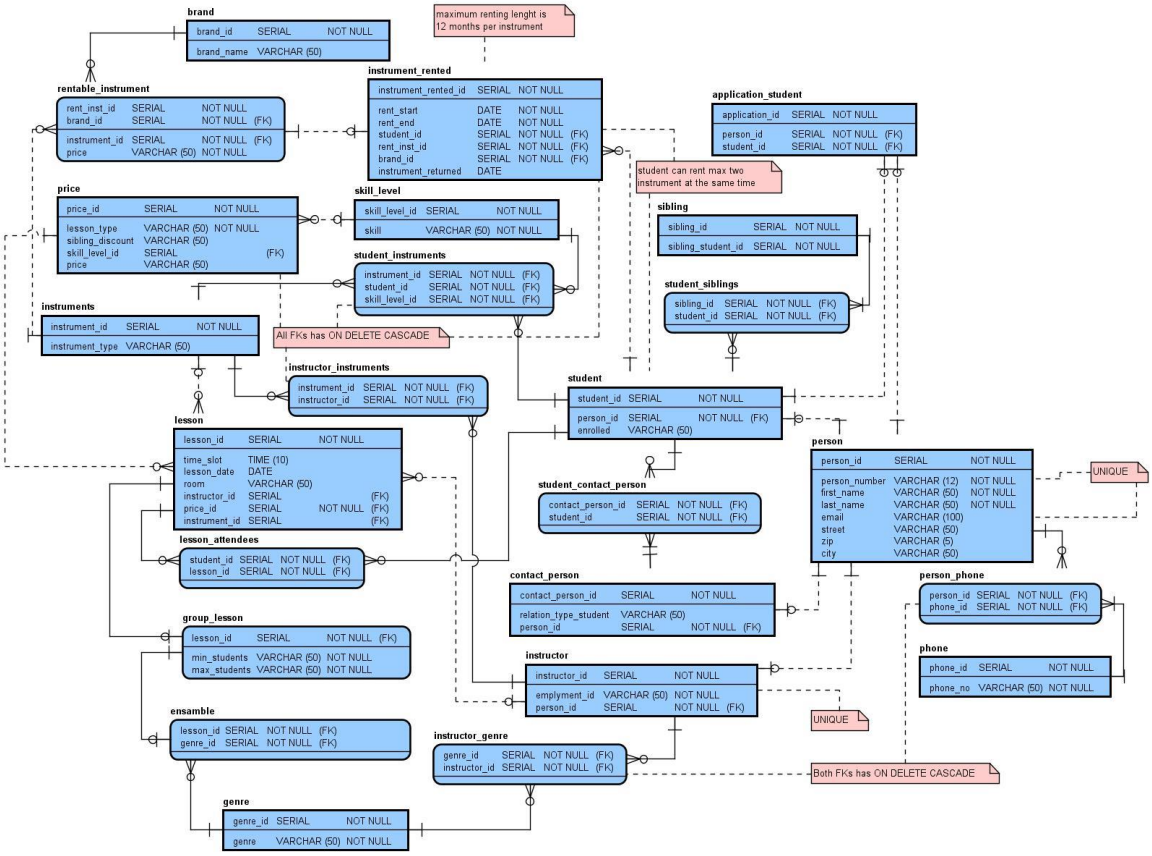

Figure 4. 1: Displays the logical model for Soundgood Music School.

5 Discussion

The naming convention is followed in this model. I've attempted to make all the names self-explanatory, and the naming convention for the model is lower case and combined words are connected with underscore. To my best knowledge, the model is in 3NF. The author made sure that all column constraints and foreign key constraints are specified with either ON DELETE SET NULL, ON DELETE CASCADE or ON DELETE NO ACTION. After designing the database thoughtfully, it can run all the different operations, no tables are missing, and no unnecessary tables exist in the database. All primary keys and naming are well chosen so it can be easily understood and easy to use to select specific data. Each table have relevant data and there are columns for all data that shall be stored and other information can be retrieved based on the data that is stored. For example there is no column for if an payment of a students lessons, but this can be gathered by all lessons a student have taken on a specific month. The column types are primarily varchar and time with a few date, this could have been improved to a timestamp instead of time and date

but none generator for fake data could handle timestamp. All primary keys are set to be unique automatically since it is of the SERIAL type.