Seminar 1

Datalagring, IV1351

Adrian Jonsson Sjödin
adriansj@kth.se

November 10, 2022

Contents

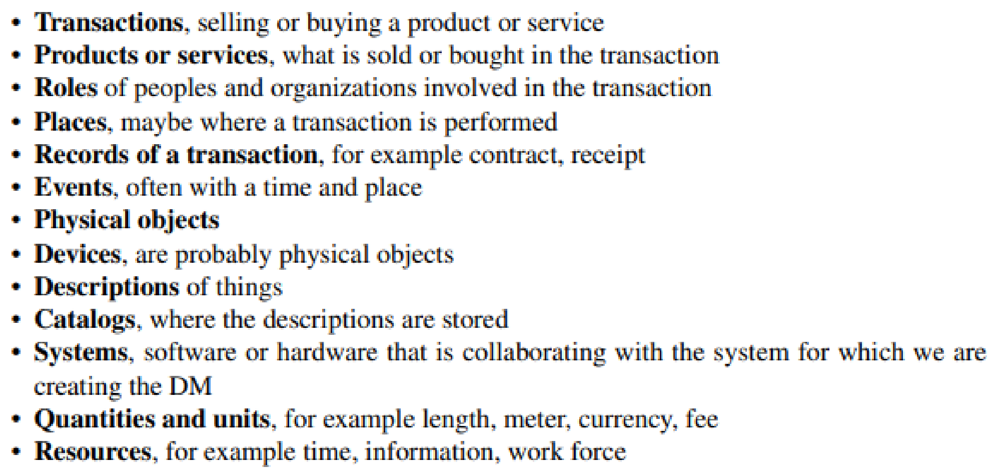
1 Introduction

The purpose of this seminar is to practice modeling information based on an organizational description that will later be turned into a functioning database. The first step in this is to produce a conceptual model. The creation and reasoning behind this conceptual model is what will be covered in this report.

2 Method

To get our conceptual model we followed the same process as in IV1350 Object Oriented Design when we created our DM. We started with noun identification, followed by using a category list (see 2.1) to try and get all the entities. After those two steps where done we have quite a lot of entities, some of which where duplicates but with slightly different names. The next step now was to go trough all of the entities and decide which we should keep and which where unnecessary, and then from there we started grouping similar entities together and turn some of them to attributes. At the same time as we turned entities to attributes and found new attributes we also specified the cardinality for them. Finally when we where satisfied with this we started to draw associations between entities.

The software used to create the conceptual model is Astah Professional. This was chosen because I'm already familiar with it from the previous mentioned course. One thing with it though is that it wasn't possible to specify 0..1 cardinality for the associations. Thus all associations have the bubble indicating 0 on them.



- **Transactions**, selling or buying a product or service
- **Products or services**, what is sold or bought in the transaction
- **Roles** of peoples and organizations involved in the transaction
- **Places**, maybe where a transaction is performed
- **Records of a transaction**, for example contract, receipt
- **Events**, often with a time and place
- **Physical objects**
- **Devices**, are probably physical objects
- **Descriptions** of things
- **Catalogs**, where the descriptions are stored
- **Systems**, software or hardware that is collaborating with the system for which we are creating the DM
- **Quantities and units**, for example length, meter, currency, fee
- **Resources**, for example time, information, work force

Figure 2.1: Category list.
L Lindbäck, A First Course in Object Oriented Development, A Hands-On Approach, 2022

3 Result

In figure 3.1 we have the result from using noun identification and a category list.

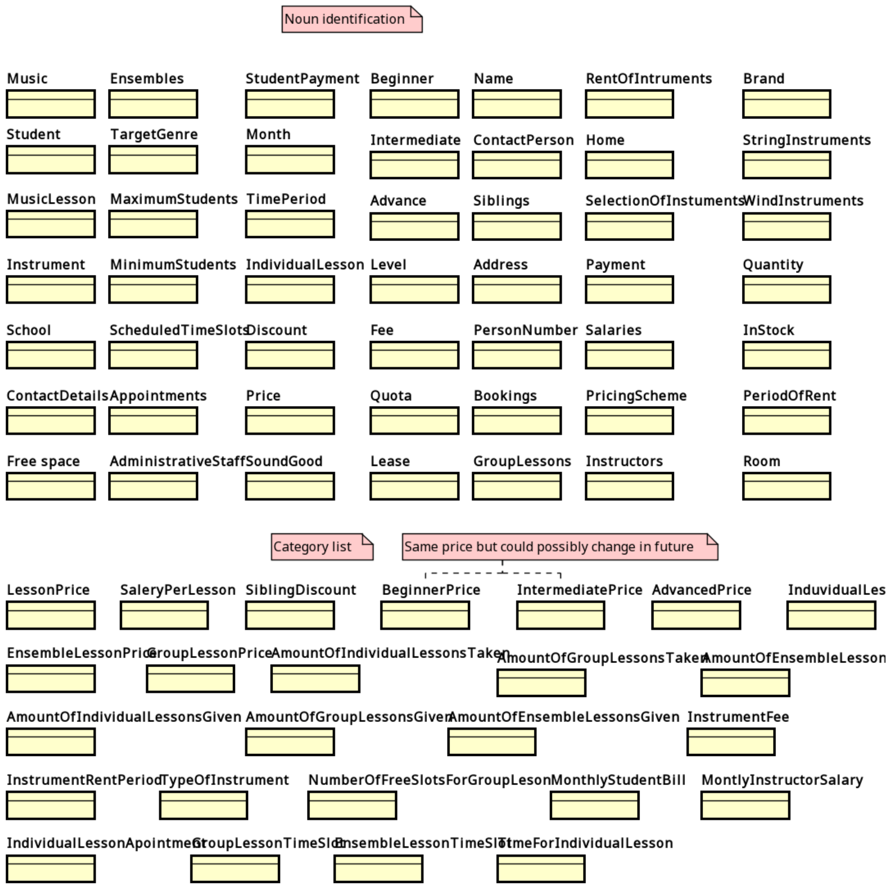


Figure 3.1: Found entities from noun identification and category list

In figure 3.2 we can see the finished conceptual model. I tried to have zero duplicated data and derived data. That is why I have an entity called *person* who contains all information that both a student and an instructor should have, and had the *student* and *instructor* entity just contain relevant attributes unique to them. I also don't have an attribute *total* anywhere in the conceptual model since that would be derived data, and can for example in the *MonthlyStudentBill* be calculated from tha information in *PricingScheme* and its own attributes.

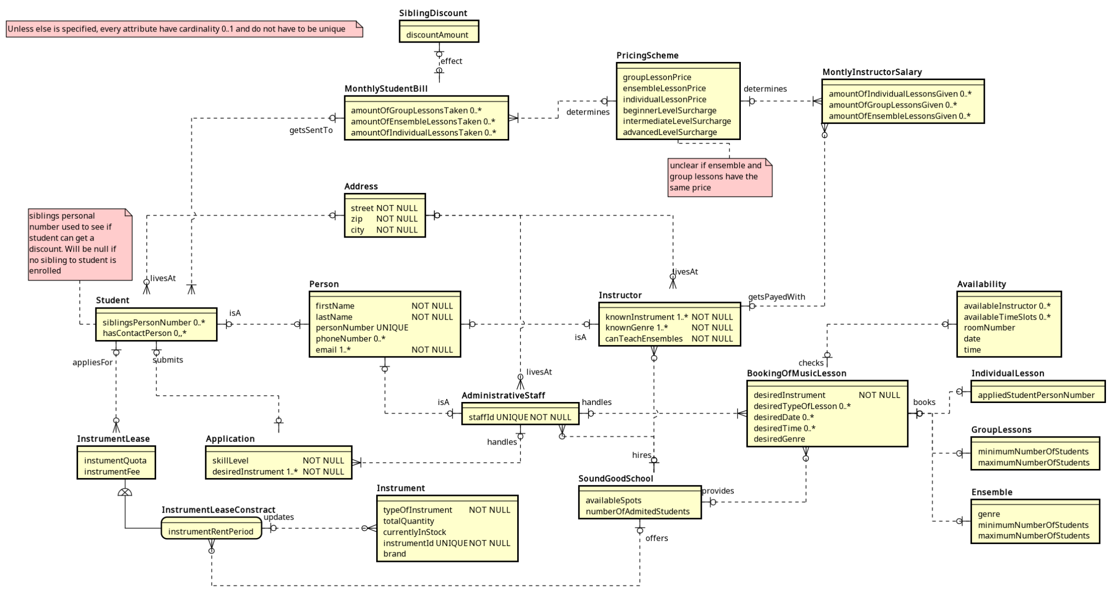


Figure 3.2: Conceptual Model for the SoundGood music school database

4 Discussion

This is my first attempt at modeling a database but I think I've gotten everything that was specified in the customer description. I have quite a few attributes that are not allowed to be nulled and for those I've simply decided that upon what I think the customer would want since it wasn't specified if null values where allowed. For example in the entity *Instructor* all three attributes are not allowed to be null, this because it stands to reason that an employer would want to know which instruments an instructor knows and what genres they can teach.

The customer description mentioned that there should be possible to store a contact person for a student, but not that they must have one. Because of this I had an entity called *ContactPerson* in an earlier version of my conceptual model, however] since I strived to have no duplicated data I had to remove it since that entity then became empty. Instead we have an attribute *hasContactPerson* in *Student* that will have a reference to *Person* for the contact person. This is also the reason why the attribute *personNumber* can be null, since there is no point in storing the person number for a contact person.

There is also an entity *BookingOfMusicLesson* for which it can be argued that its associations might be a bit weird. My thought behind it was that the administrative staff handles a booking by checking what the student wants and then look at what is available and can offered to the student. That then books one out of the three lesson types. But reading the name now, for example "*BookingOfMusicLesson books IndividualLesson*" does come across a bit strange. Perhaps a better name might have been just *MusicLesson*.

Apart from the above mentioned I believe the conceptual model is sufficient and that it follows the rules and constraints that a conceptual model should have. Naming conventions have been followed and every entity has an associations and every attribute have its cardinality specified.