Seminar 1 - Conceptual Model

Data Storage Paradigms, IV1351

Seema Bashir

9/11-2023

1 Introduction

The aim of of this task is to create a detailed and thorough conceptual model for the Soundgood Music Schools database. In order to successfully create the model, it needs to cover several aspects concerning the reality such as managing information at the school, including planning lessons, student details, assigning instructors, processing payments, and handling instrument rentals. These are all processes which concern the Soundgood Music school and how they manage data for all parties concerned.

The task was to meticulously adhere to the data specified in the requirements and the descriptions which highlighted how the Soundgood Music school manages their routines. This included how the Soungood Music school establishes and maintains the management of various lesson types, monitoring student records and their associated monthly costs, handling particulars concerning instructors and their monthly salaries, and formulating a system for instrument rental. Another vital aspect which was included is the implementation of a sibling discount. The student discount is applied to a students month cost. It should also be noted that the database is exclusively designed for the computation of payments within Soundgood's financial system and does not extend its functionality to encompass other financial tasks such as bookkeeping or tax calculations.

The higher grade task is also implemented in the solution which highlights the use of inheritance relationships between the different entities.

In collaboration with Razan Yakoub, this report details the approach employed and executed, providing an overview of the collaborative results in meeting the specified requirements.

2 Literature Study

In order to prepare for this seminar, chapter three in the 7th edition of Fundamentals of Database Systems by Elmasri and Navathe is evaluated. The chapter delves into the modeling concepts of the entity-relationship (ER) model, which consists of several fundamental components. Entities represent distinct objects or concepts within the domain. Thereby it is the attributes that describe the characteristics or properties of

these entities. Each attribute has a name, data type, and cardinality (range of possible values).

The chapter further gives insight on weaker entities and various other attribute types such as simple, composite and multivalued. Composite attributes are often used to represent attributes that have a natural hierarchical structure. Thus these attributes may need their own class.

Chapter three further illustrates the CM modelling process. The CM modelling process typically involves the identification of entities and subsequent attributes. Thereby, entity relation types are identified which highlight the relationships between entities and classify them as one-to-one, one-to-many, or many-to-many.

Furthermore, lectures provided by Leif Lindbeck including, *Introduction to UML, How to create a domain model, IE notations* and *Conceptual model* are watched and evaluated.

Lastly, the "Tips and Tricks" documents provided vital details regarding the CM model. The document highlights 5 important points which are crucial to consider when constructing the diagram. The first point emphasises on the importance of avoiding entities without attributes. Since the CM is used as a basis for data storage development. As the data is stored as attributes in the model, it is vital to include attributes in the entities. Additionally, it is important to avoid displaying actions in the CM model. Incase the action taking place is not data related, it simply does not have any value being displayed. Furthermore, another significant aspect to consider when creating the model is the concept of derived data. Derived data is data which can be calculated from other data in the data storage. In a database, derived data poses a disadvantage as it slows down the database. This is because more data must be written and more entities might be updated than if there wasn't any derived data. Hence, data is best stored when not duplicated.

3 Method

Approaching this task, a similar methodology is employed in crafting a Domain Model in the IV1350 course. The pivotal alteration, however, lies in the conceptual model. Here, the preference leans towards entities rather than classes, with a primary focus on the relationships between these entities and their associated data attributes, giving rise to a data-centric model. Notably, every entity now mandates the inclusion of data; none are permitted to exist in an empty state.

Consequently, the "noun identification" stage marked the outset, designating every noun in the school's management description as an entity. This allowed for a collection of entities which were potential candidates for the model. Progressing from there, the "category list" phase unfolded, featuring a table with 13 categories. Within each category, additional entities linked to it were identified, even if not explicitly articulated as nouns in the school's description. Essentially, this step allowed for the identification of less apparent entities, allowing the list to expand.

Following that, a thorough examination of the entity list led to the next step: the

"eliminate unnecessary entities" phase. The objective here was to discard entities that could be more aptly described using alternative terms or those that proved unnecessary for the model. In essence, the focus was on removing entities devoid of a direct relevance to the actual data, making them unnecessary for storage in the database.

Following the refinement of the entity list, the next step involved "finding attributes". This involved identifying entities that could function more effectively as attributes within another entity. These attributes assumed various forms—ranging from integers, strings, booleans, to timestamps. Each attribute was assigned a cardinality, signifying its allowable range or the number of instances within the model's context. Consequently, the primary entities incorporated these attributes. This systematic approach also facilitated the identification and removal of additional entities lacking associated attributes, resulting in a more streamlined and efficient model.

In concluding the procedural phase, the identification of "appropriate relations" among diverse entities was undertaken. Significantly, the conceptual model emphasized relationships that illuminate the processing of data and the interconnection of various data components, accentuating the flow and interactions within the system.

Furthermore, IE notation (crow's foot) are employed in order to establish the relationships. The IE notation has two primary relationship types. The first entails the "identical" relations, elucidating similarities and shared attributes. Secondly, there are the "non-identical" relations, depicting connections between entities that are distinct yet interrelated. Additionally, the incorporation of the "subtype" relation, a specialized type illustrating inheritance and hierarchy within the conceptual model, facilitated the depiction of how certain entities inherit attributes and characteristics from parent entities.

4 Results

Considering the higher grade task, the initial step involved constructing the main conceptual model using inheritance. Subsequently, the model was duplicated without incorporating inheritance to allow for a comparison and evaluation of associated advantages and drawbacks, in line with the assignment's higher grade requirement

4.1 Diagram

Figure 1 illustrates the conceptual model diagram, encompassing key entities necessary for the effective representation and management of the school's business data. These include Student, Instructor, Instrument, and Lesson. The diagram also incorporates tailored entities to meet specific needs, such as instrument rentals, scheduling availability for Instructors to facilitate administrative booking of individual lessons, and a pricing scheme detailing the cost per lesson based on lesson type and skill level. The pricing scheme plays a pivotal role in determining both the student's monthly expenses and the instructor's monthly salary.

To accommodate specific conditions, such as the need for a contact person for students under the age of 18, an entity named ContactPerson is introduced. Consequently, a

one-to-one connection is established between Student and ContactPerson, necessitating the presence of each student (parent) while allowing for the potential inclusion of a ContactPerson (child) ranging from zero to one. This design introduces the necessary adaptability within the model.

Another notable condition concerns the management of instrument rentals, wherein each student is restricted to renting a maximum of two instruments concurrently, and each instrument is available for rental for a duration of up to 12 months. To address this constraint, a new entity is introduced named "RentingPeriod," representing the rental duration for each instrument per student. This entity is associated with both the instrument and the student. Despite defining the relationship as "one-to-zero-or-many," the condition ensures that each student can maintain a maximum of two "RentingPeriod" connections simultaneously, thereby establishing it as a "one-to-zero-or-two" relationship.

Moreover, the determination of overall expenses for students and total salaries for instructors is facilitated through the utilization of the "PricingScheme" entity. It's noteworthy that both costs and salaries are derived data, calculable using alternative attributes. For instance, in the computation of a student's monthly cost, the "takenLesson 0.." attribute is employed, representing an array of Lesson data types that encompasses all lessons a student has undertaken in a given month. With each lesson carrying an associated price, the total cost is readily computed through the application of logical methods. This same principle is applicable to the calculation of instructor salaries, where they also possess an attribute, "givenLesson 0..," denoting a similar array type that includes Lesson data for the lessons provided in the previous month.

Additionally, another important condition is addressed, pertaining to the application of a sibling discount for students with siblings enrolled in the school and who have simultaneously taken lessons in the same month as their siblings. To implement this an integer attribute named "siblingDiscount" is incorporated under the Student entity. The calculation of this attribute is expected to be carried out through a logical application method at a later stage. As a result, the student discount will be applied to the total cost of the lessons undertaken by each student.

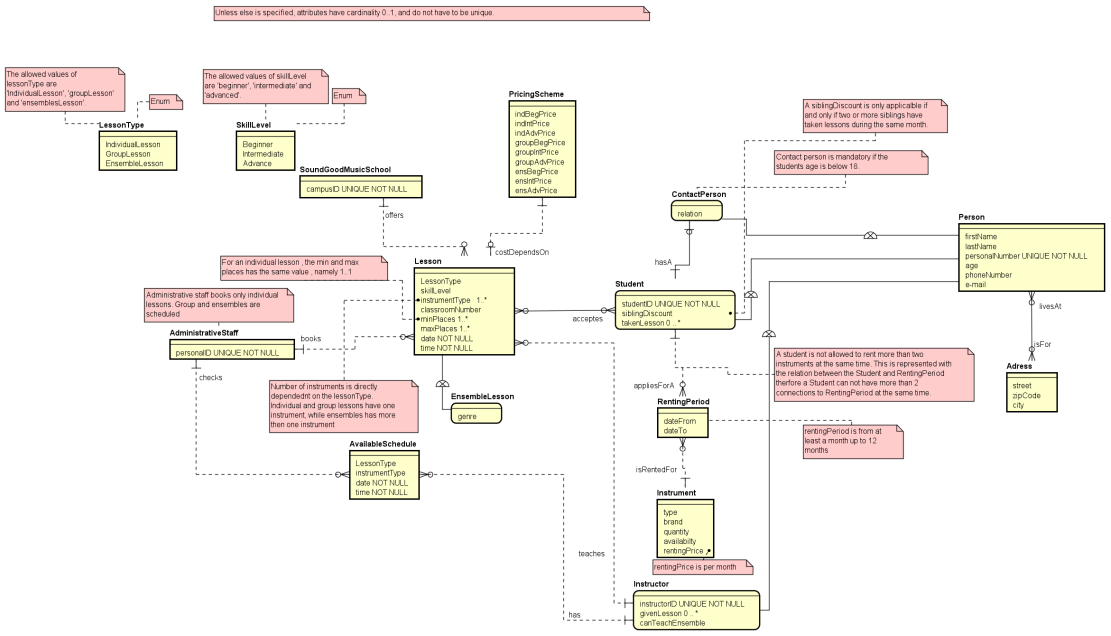

Figure 1: The Conceptual Model diagram featuring the utilization of Inheritance.

4.2 With vs. Without Inheritance

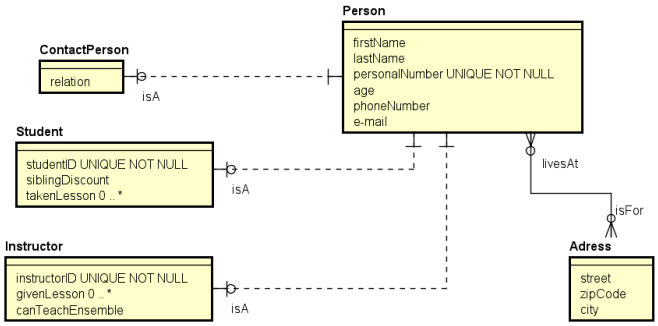

Figure 2: A segment of the Conceptual Model diagram devoid of the use of Inheritance.

Incorporating inheritance between the super entity "Person" and its derived sub-entities, namely "Student," "Instructor," and "Contact Person," was undertaken. This decision is made to centralize common attributes, such as name, personal number, and contact details, which are shared across multiple entities. Introducing a parent entity allowed for the seamless inheritance of these attributes by other entities. Similarly, in the context of "Lesson," encompassing three types (individual, group, and ensembles) with shared attributes like lesson type, skill level, date, and time, a parent entity is established for these common attributes. This resulted in the individual and group lesson entities

becoming redundant as they inherited all attributes from the parent entity "Lesson." Consequently, adherence to the rules of a Conceptual Model diagram led to the removal of the entities for group lessons and individual lessons since they had no attricbutes (data) in them.

Conversely, the "Ensemble" entity maintained its distinct attribute, genre. Adhering to the specified instructions, an alternative method of establishing the connection is established without utilizing inheritance for one of them. To be more specific, the model is replicated however now there are one-to-one relation instead of a subtype relation between "Person" and its children, exemplified by the relationship "Student isA Person."

5 Discussion

Upon revision, a robust conceptual model (CM) should thoroughly encompass all pertinent information tailored to the specific needs of a given business—specifically, in our context, capturing the business framework of Soundgood Music School. As previously emphasized, my partner and I successfully achieved this objective. In the results section, we systematically incorporated all essential entities, as outlined earlier, effectively addressing the unique requirements relevant to Soundgood's operations. The model adeptly reflects Soundgood's data management needs and aligns seamlessly with the organization's specific business rules and constraints, establishing a solid foundation for efficient information management within the school.

In terms of retrieving information related to significant entities, our conceptual model is strategically designed to ensure a seamless process for gathering data on key components such as students, lessons, and instructors. The number of "hops" required to access related information is kept at a reasonable level, thereby enhancing the user-friendliness of the model. For instance, when retrieving specific student data, direct navigation to the "Student" entity within our model is facilitated. Subsequently, by traversing the established relationships, access to details regarding the student's enrolled lessons and associated instructors can be easily accomplished. This streamlined structure minimizes the steps necessary to collect pertinent information. In practical terms, this equips Soundgood's administrative staff to efficiently procure the required details without unnecessary complications or detours, contributing to the development of a user-friendly and accessible system.

Regarding the number of entities, our model has been thoughtfully created. We are confident that we have incorporated all the essential entities necessary for effective data management at Soundgood Music School. Additionally, a meticulous process was undertaken to eliminate any unessary entities, notably those initially devoid of content, such as "IndividualLesson" and "GroupLesson," both of which were excluded in the final version of the Conceptual Model (CM). This careful curation ensures that the model remains focused on relevant data, avoiding unnecessary elements.

Moreover, our model incorporates a suitable number of connections that aptly depict how data is managed within the system. We have been diligent in excluding relations that are not pertinent to data management, thereby guaranteeing the model's conciseness

and alignment with Soundgood's business requirements.

In our conceptual model, we approach attributes with a particular emphasis on cardinality, which defines how one entity can be related to another in terms of the number of instances. The default cardinality setting is established at 0..1, indicating the potential for an attribute to have zero or one instance. This default approach allows for adaptability to various situations.

In instances requiring greater precision, specific cardinality values are assigned. An illustration of this can be observed in the "minPlace" and "maxPlace" attributes linked to the "Lesson" entity. Here, these attributes are configured with a cardinality of 1.., indicating that both the minimum and maximum places must have a minimum value of 1, while accommodating maximum values as needed. This cardinality is in precise alignment with the specified requirements of the "Lesson" entity.

Furthermore, attributes such as "takenLesson" are endowed with a cardinality of 0.., reflecting the concept that a student may engage in multiple lessons. This strategic approach ensures that our model accurately encompasses potential scenarios without unduly constraining possibilities.

Laslty,inheritance is used in the CM model to establish a connection between the "Person" entity and its offspring entities, namely "Student," "Instructor," and "ContactPerson," resulting in distinct advantages. This method facilitates the streamlined representation of shared personal attributes, such as name and contact details, consolidating them within the parent "Person" entity. This promotes code reusability and ensures data consistency. Conversely, when we linked "Person" to the same entities—"Student," "Instructor," and "ContactPerson"—without utilizing inheritance, opting instead for one-to-one non-identical relations, we discerned additional advantages. This alternative allows for heightened flexibility and autonomy, enabling each entity to evolve independently without impacting others. Such an approach simplifies the model in scenarios where entities lack extensive shared attributes, striking a balance between consistency and adaptability. This contributes to a clearer and more versatile representation of Soundgood Music School's data management needs.