Project Report

Data Storage Paradigms, IV1351

Lana Ryzhova

lana_lana67@ukr.net

Ångermannagatan 1B lgh 1003

162 64 Vällingby

2004 11 05

1 Introduction

1. **The purpose of this report is:**

   1.1 Creating a conceptual model of the Soundgood music school database.

   1.2 Construction of a diagram that is created based on the description of the requirements for solving the problem of creating a conceptual model.

   1.3 Transforming a conceptual model into a functioning database.

2. **The task was solved independently.**

2 Literature Study

1. A lecture on the conceptual model from the book "Fundamentals of Database Systems (7th Edition recommended, 6th edition is also fine) by Ramez Elmasri and Shamkant B. Navathe" was read and understood..

2. Videos on conceptual modeling were viewed.

3. Was read in articles from the Internet about inheritance.

4. The PostgreSQL program has been installed and its basic functionality has been analyzed.

5. Using Data Modeling using the Entity–Relationship (ER) Model, using crow's Foot Notation, the task of creating a conceptual model of the Soundgood music school was solved.

6. The report and solution to this problem have been posted in my Git repository.

   https://github.com/Lana-1167/KTX-1167.git

7. The document tips-and-tricks-task1.pdf was read and understood.

3 Method

Basic Components of Conceptual Modeling:

Entities: represent key objects or concepts in the domain, such as Student, Instructor, Lesson, or Musical Instrument, etc**.**

Attributes: define properties of objects such as Student Name, Person number, Data of rent instruments, Address, etc.

Relationships: represent connections between entities, such as a student placing multiple orders for different lessons on different instruments, musical instruments belonging to different brands and having different types, or an instructor who can be assigned to group, individual and ensemble lessons, etc.

Procedure for creating a conceptual model.

Description of the procedure to follow to create a conceptual model, following the steps covered in the conceptual modeling videos.

The steps to create a conceptual data model include several stages:

1. It is necessary to define key database objects:

    1.1. List the key database objects that will be included in this database.

    1.2. Determine which database objects are of primary importance and require storage and retrieval.

2. It is necessary to define the attributes of each entity that are relevant to the scope of the conceptual data model.:

    2.1. Identify the attributes of each entity that fall within the scope of the conceptual data model.

    2.2. Define the basic properties of each entity without going into details such as data types or constraints.

3. Establish relationships that make sense from a business perspective:

    3.1. Make an analysis of the relationships between objects.

    3.2. Based on the analysis, determine the existing relationships.

4. Review and refine the conceptual model for inconsistencies, redundancies and missing information.

    4.1 Analyze the initial conceptual model for inconsistencies, redundancies, and missing information.

    4.2 If necessary, update the conceptual model to improve its accuracy and completeness.

At the end of the conceptual data modeling process, a high-level representation of the conceptual data model is obtained, which will serve as the basis for the next stage of the process - logical data modeling.

This solution meets the basic requirements for solving the problem of creating a conceptual model based on information about the organizational description of the Soundgood music school and can be transformed into a functioning database.

The created conceptual model covers the entire description of the Soundgood music school company.

When creating the conceptual model, I used Data Modeling using the Entity–Relationship (ER) Model, using crow's Foot Notation.

4 Result

The main goal of conceptual data modeling is to understand business requirements and enable communication between stakeholders such as business analysts, developers, and end users.

4.1 Application for attend to the Soundgood music school

One of the key objects of the database is "Application for attending the Soundgood music school", which is created in accordance with the following requirements.

«Soundgood sells music lessons to students who want to learn to play an instrument. When someone wants to attend the school, they submit contact details, which instrument they want to learn, and their present skill».

"Application for attending the Soundgood music school" will have a web interface.

4.1 Application for attending (tab. attend_student)

attend_id	name	surname	telephone	e_mail	instrument_id	level_id
1	Henry	Jacob		Henry. Jacob@google.com	Violin	beginner

The "Application for attending" relationship is of primary importance and requires data storage and retrieval. Data from "Application for attending" will be transferred to the following relationships using procedures: "Student", "Address", "Contact_details".

When data from the “Application for attending” is inserted into the Student table, the student is assigned a Person number. There are 3 types of Person numbers. For students, for instructors, and for other persons (e.g. student's parents). Different sequences of personal numbers will be used to separate these entities. For example,

100 to 999 are students, 1000 to 4999 are instructors, 5000 to 9000 are other people.

Separation of personal numbers is necessary to ensure data integrity, since students, instructors, and other persons have a similar set of attributes. In fact, separation of personal numbers is necessary to ensure the process of inheritance in the database, which allows a type or table to acquire the properties of another type or table.

The attributes of "Application for attending" are:

«attend_id»	primary key for "Application for attendance"
«name»	student name
«surname»	student's last name
«telephone»	contact details
«e_mail»	contact details
«instrument_id»	musical instrument
«level_id»	Current student skills

4.2 Invitation

According to the task condition "If there is a room, the student is offered a place". "Invitation" will have a web-interface.

4.2 Invitation (tab. invitations)

invitation_id	name_app	surname_app	instrument_id	level_id
1	Henry	Jacob	Violin	beginner

4.3 Lesson.

According to the conditions of the task “There is no concept like 'course' or sets of lessons. Instead, students continue to take lessons as long as they wish. There are individual lessons and group lessons.

4.3 lesson (tab. lesson)

level_id	level_stud
1	beginner
2	intermediate
3	advanced

4.4 Level of lesson.

According to the conditions of the task “There are three levels: beginner, intermediate and advanced”.

4.4 level (tab. levels)

lessons_id	type_ lessons
1	individual
2	group
3	ensemble

4.5 Ensemble genre

According to the conditions of the task «Ensembles have a specific target genre (e.g., punk rock, gospel band)».

4.5 Ensemble genre (tab. ensemble_genre)

ensembles_id	**genre_ensembles**
1	punk rock
2	gospel band

4.6 Type of musical instruments

According to the problem statement, musical instruments have such a property as instrument type.

4.6 type of musical instrument (tab. types_instruments)

type_id	**types_instrum**
1	Strings
2	Wind
3	Reed
4	Drums

4.7 Brand of musical instruments

In accordance with the condition of the problem, musical instruments have such a property as the brand of the instrument.

4.7 Brand of musical instrument (tab. brands)

brand_id	brand
1	Gibson
2	Harman Professional
3	Shure
4	Yamaha

4.8 Musical instruments

There is a wide selection of instruments, wind, string etc., supporting different brands and in different quantities in stock at the Soundgood music school.

4.8 Musical instruments (tab. types_instruments)

instrument_id	instruments	brand_id	type_id	quantities
1	Violin	1	2	5
2	Piano	6	1	2
3	Guitar	8	4	7
4	Drums	4	5	10

4.9 Student

According to the conditions of the task «Person number, name, address and contact details must be stored for each student».

4.9 student (tab. student)

student_numb	name	surname	address_id	details_id	contact_id	numb_sibling
117	Henry	Jacob	1	1	1	2

4.10 Address

The "Address" relation stores information about addresses.

4.10 Address (tab. address)

address_id	postal_code	city	street	home	room
117	123 55	Stockholm	Nygatan	11	20

4.11 Contact details

The "Contact details" relation stores information about contact details.

4.11 Contact details (contact_details)

contact_id	telephone	e_mail
117	741132337	Henry.Jacob@google.com

4.12 Contact person

The "Contact person" relation stores information about the contact person.

4.12 Contact person (tab. persons_conta

person_id	name	surname	student	contact_id	related_attitude
5000	Kris	Jacob	117	1	father

4.13 Sibling

The “Sibling” relation stores information about siblings. According to the assignment - “Furthermore, it must be possible to see which students are siblings, since there is a discount for siblings. It’s not sufficient to show just whether a student has siblings, it must be possible to see who’s a sibling of who.”

To implement this condition, the attribute "number of brothers and sisters" was added to the students table, and the "Sibling" relationship was implemented.

4.13 Sibling (tab. sibling)

sibling_id	student_numb	person_sibling_ numb
1	117	118

4.14 Instructor

In accordance with the task “Person number, name, address and contact details must be stored for each instructor”.

Since the "Address" and "Contact details" relationships for instructors have the same structure as for the "Student" relationship, we use inheritance and common

"Address" and "Contact details" relationships for instructors and students. For instructors, personal numbers will start with the 1000 number.

4.14 Instructor (tab. instructor)

instruktor_numb	name_inst	surname_inst	address_id	contact_id
1340	Mikael	Oliver	2	2

4.15 Persons contact

According to the task “It must also be possible to store contact details for a contact person (e.g., parent) for each student”.

Since the "Contact details" relationship for a contact person has the same structure as for the "Student" relationship, we use inheritance and common "Contact details" relationships for instructors and students. For a contact person, personal numbers will start with the 5000 number.

4.15 Person contact (tab. persons_contact)

person_id	name	surname	student_numb	contact_id	related_attitude
1	Kris	Jacob	117	1	father
2	Kris	Jacob	118	2	father
3	Kris	Jacob	119	3	father
4	Lucas	Levi	120	4	mother

4.16 Proficiency in musical instruments

In accordance with the task "A student can take any number of lessons, for any number of instruments". And also "An instructor can teach a specified set of instruments, and may also be able to teach ensembles". To implement these conditions, the "hold_instrument" relation was created. The "contact_id" attribute records the personal numbers of both students and instructors.

4.16 Proficiency (tab. hold_instrument)

hold_id	contact_id	instrument_id	level_id
1	1340	1	1
2	117	1	2

4.17 Price

In accordance with the task “Currently, there is one price for beginner and intermediate levels, and another price for the advanced level. Also, there are different prices for individual lessons, group lessons and ensembles. There is also a discount for siblings, if two or more siblings have lessons taken during the same month, they all get a certain percentage discount. Soundgood wants to have a high level of flexibility to change not just prices, but also pricing schemes. They might, for example, not always have the same price for beginner and intermediate lessons".

In addition, “Soundgood offers students the ability to rent instruments to be delivered at their home. The fee is paid the same way lessons are paid, each month students are charged for the instruments that where rented the previous month.”

4.17 Price (tab. price)

price_id	type_service	price_service	unit_measur
1	Rent instruments	100	SWK
2	Home delivery	0	SWK
3	Individual beginner	100	SWK
4	Individual intermedia	100	SWK
5	Individual advanced	200	SWK
6	Group lessons	50	SWK
7	Ensembles	5	SWK
8	Sibling discounts	5	%

4.18 Group_lessons

In accordance with the task “A group lesson has a specified number of places (which may vary), and is not given unless a minimum number of students enroll. A lesson is given for a particular instrument and a particular level. Group lessons are given at scheduled time slots".

“A group lesson has a specified number of places (which may vary), and is not given unless a minimum number of students enroll”.

“Maxsimum number of places on group lessons (for example, from 1 to a certain number of seats, which can change)”.

Application is used for booking of group lessons. Has a web-interface.

4.18 Group lesson (tab. group_lessons)

groupleson_id	lessons_id	period_start	period_end	min_places_id	max_places_id	curr_numb_students	instrument_id	level_id	instructor_id
1	2	09.01.2024	09.01.2025	1	3	2	1	1	1340
2	2	01.01.2024	31.01.2025	2	4	3	2	2	1340

4.19 Ensemble

In accordance with the task “Besides lessons for a particular instrument, there are also ensembles, where students playing different instruments participate at the same lesson. Ensembles have a specific target genre (e.g., punk rock, gospel band),

and there is a maximum and minimum number of students also for ensembles. Ensembles are given at scheduled time slots.

Application is used for booking Ensemble. Has a web-interface.

4.19 Ensemble (tab. ensemble)

ensem_id	lessons_id	ensembles_id	period_start	period_end	min_students	max_students	curr_numb_ students	instructor_id
1	3	1	09.01.2025	09.01.2025	2	15	6	1340
2	3	2	01.01.2025	31.01.2025	2	15	7	1340

4.20 Booking individual

In accordance with the assignment “A lesson is given for a particular instrument and a particular level. Individual lessons do not have a fixed schedule, but are rather to be seen as appointments, like for example an appointment with a dentist".

Application is used for booking individual lessons. Has a web-interface.

4.20 Booking individual (tab. booking individual)

indiv_id	lessons_id	student_numb	period_start	period_end	numb_lessons	instrument _id	level_id	instructor_id
1	1	117	09.01.2024	09.01.2025	3	1	1	1340
2	1	117	01.01.2024	31.01.2025	13	1	1	1340
3	1	120	01.01.2024	31.01.2025	30	9	1	1340

4.21 Administrative staff

In accordance with the task “Administrative staff must be able to make bookings, it must therefore be possible to understand which instructor is available when, and for which instruments”.

Application is used by administrative staff. Has a web-interface.

4.21 Administrative staff (tab. admin_ staff)

admin_ id	lessons_ id	period_start	period_end	time_id	price_id	student_id	instructor_id	note
1	1	01.01.25	31.01.25	6	3	120	1340	instructor is available
1	2	01.01.25	31.01.25	7	5	8	1340	instructor is available

4.22 Time slot

In accordance with the assignment "Group lessons and ensembles are given at scheduled time slots".

4.22 Time slot(tab. start_time)

start_time	end_time
8	9
10	11
12	13
14	15
16	17
18	19

4.23 Student Payment

In accordance with the task “Students are charged monthly for all lessons taken during the previous month. There is no concept like 'course' or sets of lessons, a student who has been offered a place, and accepted, continue to take lessons as long as desired, and can either book one lesson at the time or book many lessons during a longer time period”.

Application is used for payment by students. Has a web-interface.

4.23 Student Payment (tab. student_payment)

st_payment_id	lessons_id	student_numb	period_start	period_end	price_id	sibling_prices_id	total_numb_classes	total_paid
1	1	117	01.01.25	31.01.25	5	8	16	(16 *200*2)*5%
2	1	120	01.01.25	31.01.25	3	0	30	30 * 100

4.24 Instructor Payment

In accordance with the quest “There are no instructors with fixed monthly salaries, instead they are paid monthly for all lessons given during the previous month. Instructor payments depend on the same things as student fees (see above), namely level of lesson and whether a given lesson was a group or individual lesson. Instructor payments are not affected by sibling discounts".

Application is used to receive payment for lessons for students. Has a web-interface.

4.24 Instructor Payment (tab. instructor_payment)

ins_payment_id	instruktor_numb	lessons_id	period_start	period_end	price_id	total_number_classes	total_received
1	1340	1	01.01.25	31.01.25	5	16	16 *200
2	1340	1	01.01.25	31.01.25	3	30	30 * 100

4.25 Renting Instruments

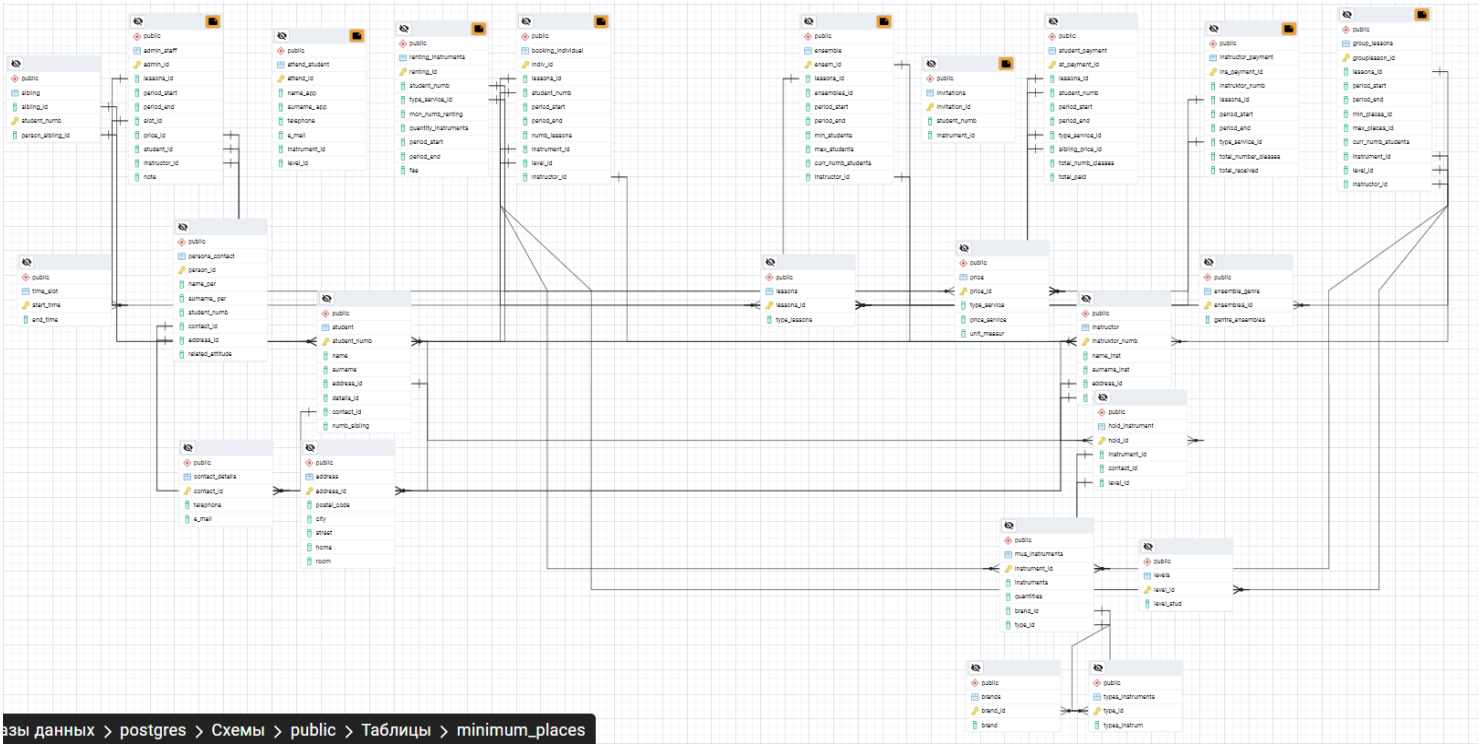
In accordance with the task “Each student can rent up to two specific instruments at any given period, the renting happens with a lease up to 12 month period. Students can list and search current instruments and rent them if they don't exceed their two-instrument quota. Instruments are rented per month. The fee is paid the same way lessons are paid, each month students are charged for the instruments that where rented the previous month".

Application is used for renting tools. Has a web-interface.

4.25 Renting Insrtruments (tab. renting_instruments)

renting_id	student_numb	price_id	mon_numb_renting	quantity_instruments	period_start	period_end	fee
1	117	1	1	2	01.01.25	31.01.25	2 * 100
2	117	2	12	1	01.01.25	31.01.25	2 * 1

Entity–Relationship (ER) Model, using crow's Foot Notation of the Soundgood Music School



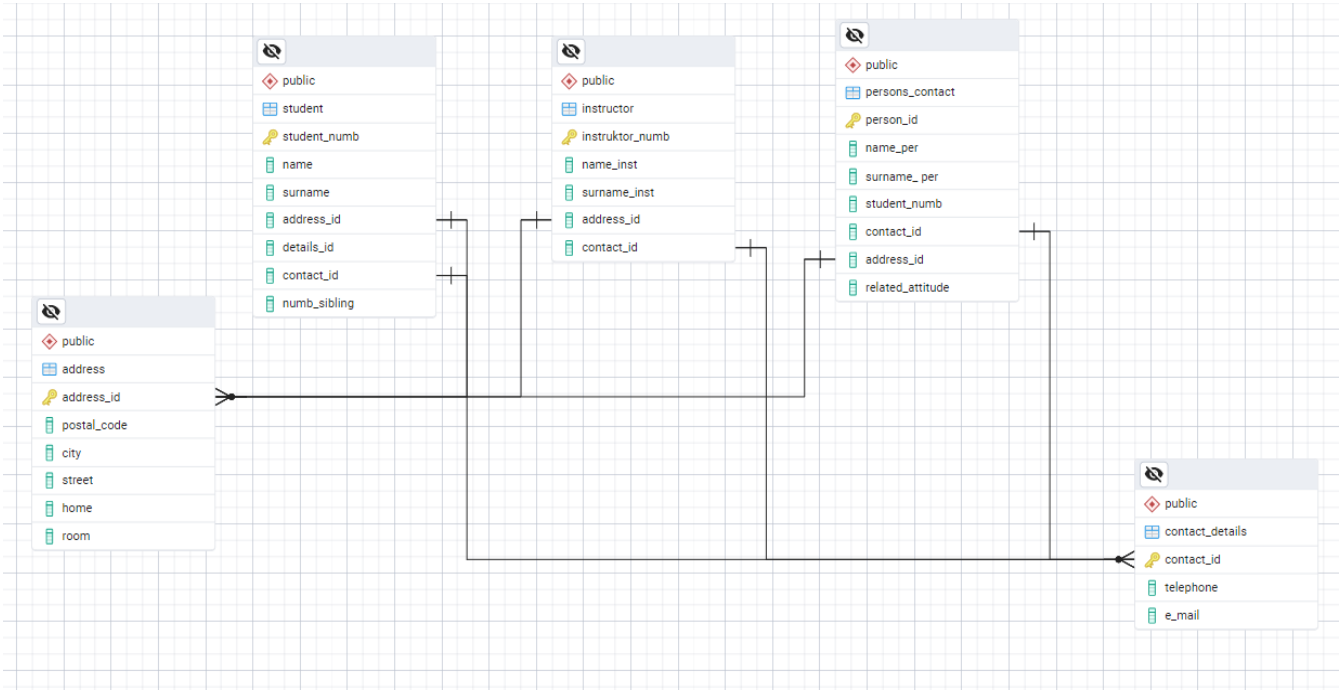
Inheritance in a Relational Database the Soundgood Music School

We have a table **student**, table **instructor** with their own fields, and a table **persons_contact**. For tables **student, instructor** and **persons_contact** have child tables **address** and **contact_details**.

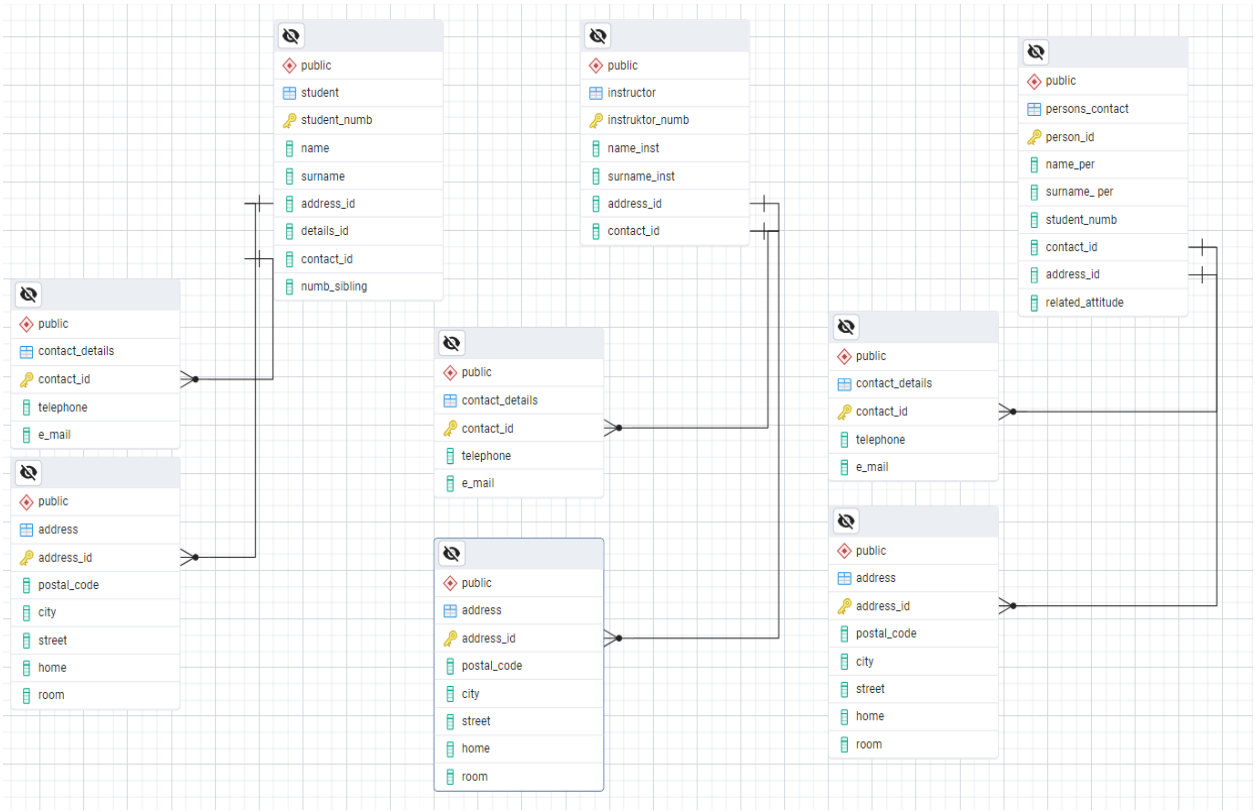
This is inheritance via composition, where children refer to the parent, and ideally have the same Primary Key. In this case, the data of one instance of the class will be physically separated in different tables, and retrieving all the fields of the object will require a JOIN query.

In Entity–Relationship (ER) Model it looks like this:

Entity–Relationship (ER) Model



A conceptual model diagram **without inheritance** that models the same thing without inheritance looks like this:



5 Discussion

1. **The Benefit of Using Inheritance in the Soundgood Music School Database**

In this database, inheritance is implemented through composition, where children refer to the parent, and ideally have the same Primary Key. In this case, the data of one instance of the class will be physically separated in different tables, and retrieving all the fields of the object will require a JOIN query.

In this solution, there are three identical Primary Keys that are identifiers of students, instructors, and other persons. "Person number" is the Primary Key for the tables student, instructor, persons_contact.

To ensure data integrity and separation of the Primary Key, different sequences of personal numbers will be used in these relationships. For example, 100 to 999 are students, 1000 to 4999 are instructors, 5000 to 9000 are other people.

Thus, the advantage of using inheritance is:

1.1 Ensuring data integrity. Ensuring data normalization.

1.2 Ensuring efficient use of disk space.

1.3 To speed up the selection (reading) of data from the database, you can use indexes.

The disadvantage of using inheritance is:

1.4 The code for SQL queries to select and add data will be more complex, since it is necessary to take into account the separation of personal numbers.

1.5 To obtain the necessary reports, a JOIN query will be required that will select data from different tables of the normalized representation of the database,

Accordingly, the only benefit of not using inheritance is that the code for SQL queries will be simpler.

2. **Does the CM contain all information needed by Soundgood?**

A conceptual data model is a high-level abstract representation of an organization's data. This model focuses on capturing entities, their attributes, and relationships without specifying any implementation details..

3. **Is it easy, that is a reasonable number of hops, to collect information related to all of the major entities (student, lesson, instructor, etc).**

The task is quite labor-intensive. It involves the use of 4 mandatory steps.

Identify objects: List the key objects that will be included in the database. Determine which objects are of primary importance and require storage and retrieval.

Define Attributes: Define the attributes of each entity that are relevant to the scope of your data model. Use the basic properties of each entity without going into details such as data types or constraints.

Establish relationships: Analyze the relationships between objects and identify existing relationships, ensuring that the proposed relationships make sense from a business perspective.

Review and refine. Analyze the original conceptual model for inconsistencies, redundancies, and missing information. Update the model as needed to improve its accuracy and completeness.

4. **Does the CM have a reasonable number of entities? Are important entities missing? Are there irrelevant entities, for example entities without attributes?**

In the solution presented here, the CM has a reasonable number of entities. All the important entities are present in this solution. For example, student, lessons, instructor, etc. There are no irrelevant entities in this solution, such as entities without attributes. Useless or irrelevant entities do not have any meaningful value or significance for the context or purpose of the database.

5. **Are there attributes for all data that shall be stored? Do all attributes have cardinality? Is the cardinality correct? Are the correct attributes marked as NOT NULL and/or UNIQUE?**

In the solution presented here, there are attributes for all the data that needs to be stored. This includes student data, instructors, student and instructor fees, musical instrument rentals, etc. Cardinality is the uniqueness of the data values contained in a column. Not all attributes have a cardinality. For example, phone or email in the contact_details relationship do not have a cardinality. In other entities, all attributes have the correct cardinality. Attributes that have the correct cardinality are marked as NOT NULL and/or UNIQUE.

6. **Does the CM have a reasonable number of relations? Are important relations missing? Are there irrelevant relations? Do all relations have cardinality at both ends and name at least at one end?**

Yes, the developed CM has a reasonable number of links. Important links are present. There are no irrelevant links. The power is the number of tuples in the relation (in simple terms, the number of records), the power of the relation can be any (from 0 to infinity), the order of the tuples is not important. Yes, all links have a power at both ends and a name at least at one end.

7. **Are naming conventions followed? Are all names sufficiently explaining?**

This solution follows naming conventions. All names are self-explanatory.

8. **Is the notation (UML or crow foot) correctly followed?**

Yes, the notation is correct. but the project used Entity–Relationship (ER) Model, using crow's Foot Notation.

9. **Are all business rules and constraints that are not visible in the diagram explained in plain text?**

All business rules and constraints that are not visible in the diagram are explained in plain text.

10 **Is the method and result explained in the report? Is there a discussion? Is the discussion relevant?**

Yes, the report explains the method and the result. There is a discussion that is relevant.

6 Comments About the Course

The conceptual data modeling assignment, including lectures, installation and learning of the required software, practical modeling, and preparation for the workshop, has taken up all the time since the beginning of the course. But since conceptual data modeling is the first step in the data modeling process, I believe that solving the practical problem will greatly help me in my future practice.