

Project Report

Data Storage Paradigms, IV1351

Date

Project members:

[Elias Tosteberg, eliaστο@kth.se]

Declaration:

By submitting this assignment, it is hereby declared that all group members listed above have contributed to the solution. It is also declared that all project members fully understand all parts of the final solution and can explain it upon request.

It is furthermore declared that the solution below is a contribution by the project members only, and specifically that no part of the solution has been copied from any other source (except for lecture slides at the course IV1351), no part of the solution has been provided by someone not listed as a project member above, and no part of the solution has been generated by a system.

1 Introduction

I have done the Mandatory Part of Task 1. Logical and Physical Model. Where a model of the database and scripts for generating the database and its data will be created.

2 Literature Study

When preparing for the project information was gathered from the digital Logical and Physical model lecture. There it showed how to create the model.

3 Method

When creating the model Astah was used. The first step was to create all the necessary tables. The next Step was to add all the attributes that are not multivariable attributes. Then Primary keys were then decided upon either a unique attribute or a surrogate key. The relations were then added for anything that were not a many-to-many relation. For any many-to-many relations a cross-reference table was added that contained the primary keys of both tables. Any multivariable attributes were then added as a separate table containing the Primary key of the original table and the attribute. After the Astah model was finished Astah was used to generate the creation script of the database. The data script was then created with a combination of online data generation and manual work.

4 Result

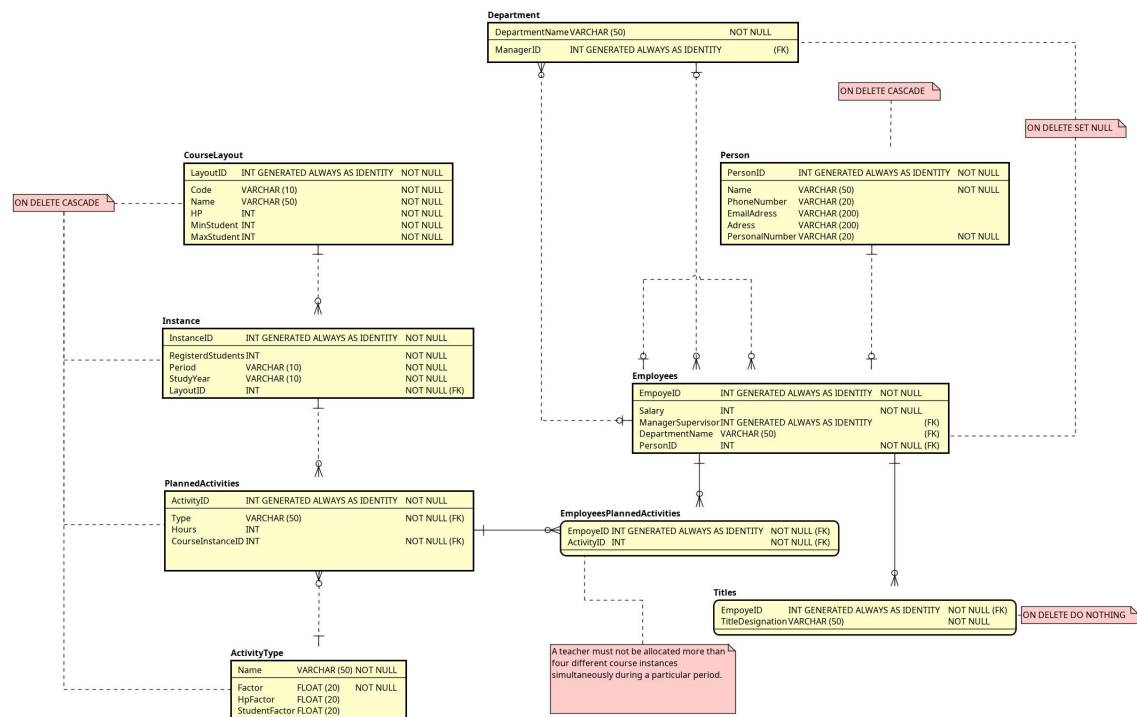


Figure 1: Astah diagram

The scrips and Astah are stored here: github.com/Sailet03/Seminar-IV1351-2025.

4.1 Primary keys

Most tables will use substitute keys for there primary keys since some needed it and for consistency it was better to use them for most. All substitute primary keys are of the Postgres datatype INT GENERATED ALWAYS AS IDENTITY so that the keys are generated automatically by Postgres.

4.2 CourseLayout

CourseLayout contains the content of the layouts. It contains the information of every course layout. Code is not unique since there can be multiple different layouts for the same course code if the course changes after a certain year or period. CourseLayout contains the min/max amount of students, the course code, course name and HP.

4.3 Instance

Instance contains instances of every course. It contains the year and period of the instance. It also contains how many students are registered. Furthermore, it contains a Foreign key that contain what CourseLayout it is.

4.4 PlannedActivities

PlannedActivities conin the all the activities that is a part of every instance. It contains how many hours the activity is going to take and what type of activity it is.

4.5 ActivityType

ActivityType contains information of the existing types of activities. It also contains the multiplayer for how much extra time is required for the teachers.

4.6 Person

Person contains the personal information of all persons.

4.7 Employees

Employees contain the information about the people that work at the university. It has a relation to itself since the manager of the employee also is an employee.

4.8 Department

This table contains what department exists and what manager it has.

4.9 Titles

Titles is a multivariable attribute just in case one employee has multiple titles.

4.10 EmployeesPlannedActivities

EmployeesPlannedActivities is a cross-reference table that connects Employees to ActivityType since multiple employees can be a part of the same activity and every teacher can have multiple activities.

5 Discussion

All the tables seem to be mostly relevant. The only exception might be the person table since it theoretically could be merged with the employee table. This would however become problematic if later another type of person would be added to the database like a student table. Then it would be much better to have the person table than to remove it. The titles' table might also be redundant if we assume that an employee can only have one title.

The models seem to be in 3NF since no attribute depend on anything that is not the primary key and there are no places where derived data is stored in the database. All the data that is required can be accessed. And the crow's foot notation seems to be correctly followed.