## Assignment 1
## Group 52
Nicolas Johansson
Aapo Haavisto

## 1. Assumptions
We have primarily chosen the primary keys based on logic; for example id's often make good keys. When needed we have also looked at domain details and the data to check for appropriate ones. Most of the foreign keys were given in the domain schema, however some had to be added.
We introduced a reference from TeachingAssistants to Students (this required adding the attribute studentId to TeachingAssistants since teacherId is not equivalent to studentId).
For the programmeCourse relation, by looking at the data we realized that *programme* and *course* was not enough to identify the rows (there exists rows for fixed values on these), therefore we also use *academicYear* as a primary key.
We also assumed that AssignedHours could reference CourseInstances in the same way as ReportedHours do, allowing for a simpler schema with fewer attributes.
Here is our final schema:

Teachers(<u>teacherId</u>, name, dept, division)

SeniorTeachers(<u>teacherId</u>)
    teacherId → Teachers.teacherId

TeachingAssistants(<u>teacherId</u>, studentId)
    teacherId → Teachers(teacherId)
    studentId → Students(studentId)

Programmes(<u>programmeCode</u>, programmeName, dept, director)
    director → SeniorTeachers.teacherId

Courses(<u>courseCode</u>, courseName, credits, level, dept, division, ownedBy)
    ownedBy → Programmes.programmeCode

ProgrammeCourses(<u>programme</u>, <u>academicYear</u>, studyYear, <u>course</u>, courseType)
    programme→ Programmes.programmeCode
    course → Courses.courseCode

CourseInstances(courseCode, studyPeriod, academicYear, <u>instanceId</u>, examiner)
    courseCode→ courses.courseCode
    examiner → SeniorTeachers.teacherId

CoursePlanning(<u>course</u>, planningNumStudents, seniorHours, assistantHours)
    course → CourseInstances.instanceId

AssignedHours(<u>course</u>, <u>teacherId</u>, hours)
course → CourseInstances.instanceId
teacherId → Teachers.teacherId

ReportedHours(<u>course</u>, <u>teacherId</u>, hours)
course → CourseInstances.instanceId
teacherId → Teachers.teacherId

Students(<u>studentId</u>, name, programme, year, graduated)
(programme, year) → ProgrammeCourses.(programme, academicYear)

Registrations(<u>courseInstance</u>, <u>student</u>, status, grade)
course → CourseInstances.instanceId
student→ Students. studentId

## 2. Design decisions

We decided to add the entity “Teacher”. This was done to clarify that both senior teachers and teaching assistants are a subclass of Teachers.
Further, we assumed that Phd students fall under Students and don’t require any extra entity.
We also decided to make “programmeCourse” a relation instead of an entity. This is the way we felt made the most sense; programmeCourse is connecting the courses which are run by a study programme in a certain year to that programme.
Finally we decided to not create entities for departments and divisions, since even if they would describe an important relationship (divisions belonging to a specific department), the entities would contain only one attribute each.