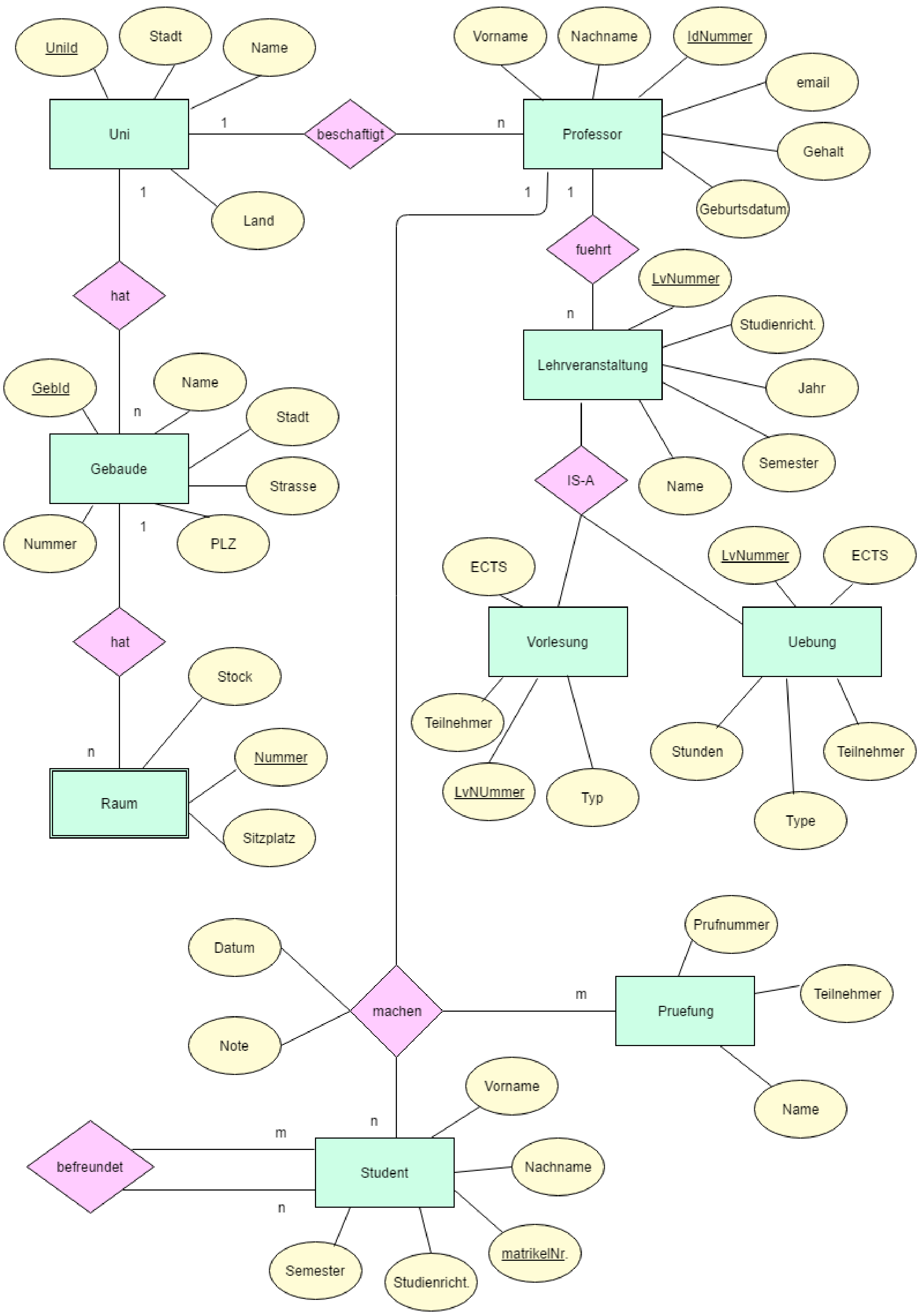
University Database

Assignment 1: Requirements Analysis & Conceptual Design

The project contains the database used to store data necessary in the university, as well as the Java program written to insert data in the database and the website (implemented using PHP) used to access the data.
The ER diagram from below models the entities and relationships to implement the database.
First we have the "University" entity, with the attributes: uniId, stadt, land, name. The university has buildings. Buildings are identified by idnumber. They also have such attributes as name, address that consists of stadt, strasse, plz, nummer. Each building has rooms. A room is a "weak entity", because the room cannot exist without the building. The room has such attributes: nummer, stock and number of sitzplaetze. Every university has professors that work there. Professors are identified by idnummer, they also have: vorname, nachname, email, gehalt, geburtsdatum. Each professor that is in the database can be found either by Id Number or by Surname. Professors conduct lessons. Each lesson has lvnummer. Lessons belong to concrete studienrichtung, they also have name, semester in which the students can study them (Wintersemester or Summersemester) and the year. The lessons have "IS-A" relationship, The lesson may be vorlesung or uebung. The attributes are: ECTSanzahl, teilnehmeranzahl,typ, stunden. Each lesson has an exam. Exam has such attributes: name, anzahlteilnehmer and nummer as primary key. Student has such attributes: vorname, nachname, studienrichtung, semester and matrikelnummer as primary key. The relationship prufungmachen has additional attributes: the day of the exam and the mark which the student has by passing this exam. Each student can be friends with other students.



UniId
Stadt
Name
Uni
1
beschaftigt
n
Professor
Vorname
Nachname
IdNummer
email
Gehalt
Geburtsdatum
1
Land
1
1
hat
fuehrt
n
LvNummer
Studienricht.
Lehrveranstaltung
Jahr
GebId
Name
n
Stadt
Gebaude
Strasse
Nummer
1
PLZ
IS-A
Name
Semester
ECTS
LvNummer
ECTS
hat
Vorlesung
Uebung
Stock
Teilnehmer
Nummer
n
Raum
LvNUmmer
Typ
Stunden
Teilnehmer
Sitzplatz
Type
Prufnummer
Datum
machen
m
Pruefung
Teilnehmer
Note
Name
Vorname
n
m
befreundet
Student
Nachname
n
matrikelNr.
Semester
Studienricht.

Assignment 2: Logical Design

- Uni(UniId, land, stadt, name)
  - *PK: {uniId}*
- Gebaude(name, stadt, strasse, plz, nummer, gebnummer)
  - *PK: {gebnummer}*
  - *FK: uniId <> Uni*
- Raum(*raumnummer, stock, sitzplaetze*)
  - *PK: {raumnummer}*
  - *FK: gebnummer<> Gebaude*
- Professor(vorname, nachname, email, gehalt, geburtsdatum, idnummer)
  - *PK: {idnummer}*
  - *FK: uniId <> Uni*
- Lehrveranstaltung (LvNummer, studienrichtung, jahr, semester, name)
  - *PK: {LvNummer}*
  - *FK: idnummer <> Professor*
- Vorlesung(*LvNummer, ects, teilnehmer, typ)*
  - *PK: {LvNummer}*
  - *FK: LvNummer <> Lehrveranstaltung*
- Uebung(LvNummer, ects,stunden, teilnehmer)
  - *PK: {LvNummerr}*
  - *FK: LvNUmmer<> Lehrveranstaltung*
- Student(*matrikelnummer*, vorname, nachname, studienrichtung, semester)
  - *PK: {matrikelnummer}*
- Prufung(prufNummer, name, teilnehmer)
  - *PK: {prufNummer}*
- Prufungmachen(*matrikelnummer*, *idnummer, prufnummer,* datum, note)
  - *FK1: Prufungmachen.matrikelnummer <> Student*
  - *FK2: Prufungmachene.Prufnummer <> Prufung*
  - *FK3: Prufungmachen.Idnummer<> Professor*

Assignment 4: Implementation

Java Implementation

The program implements a class called "TestDataGenerator". It initializes a static variable called "row" (which is later used to store each row from the "insertSQL.txt" file – explained later – and then to be run; a new dataset is to be added in the database) and it implements a method called "removeSemicolon", that removes the semicolon from any given String (if the given String is not NULL and has a semicolon).

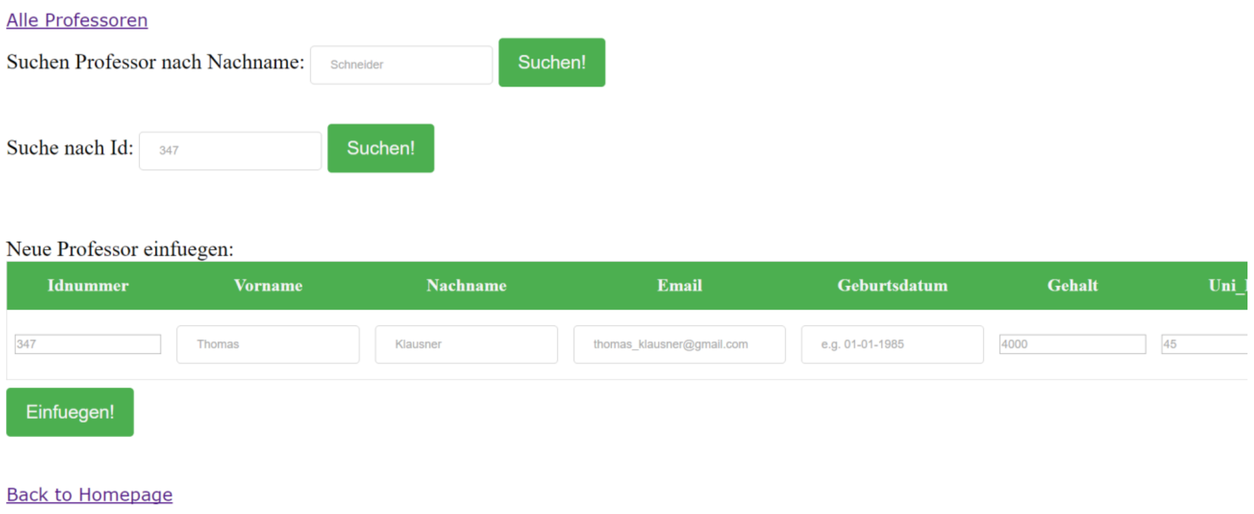
The "main" function then sets up the connection to the database, using *Connection con = DriverManager.getConnection(database, user, pass);* and then creates a statement variable: *Statement stmt = con.createStatement();* - which is later used to execute the insert statements needed to fill the database with testing values.

Using FileReader and BufferedReader is then created a connection to the .txt file "insertSQL.txt". In this file are all insert statements (and an alter session statement) that need to be run.

The program then reads every line from the file, removes the semicolon and executes the statements (first time with stmt.execute() – because the first row in the "insertSQL.txt" file is not an insert/update/delete, but an alter session, to modify the date format – and then with stmt.executeUpdate() – to insert the data into the tables).

The number of datasets in each table is then returned using the variable “rs” of type ResultSet and the method executeQuery().
In the end are all connections closed.
The code is tested for errors using the try statement and the caught errors are then handled in the catch statement (the error is printed as output, so that the user can see what is wrong).

PHP Implementation
The PHP project is structured in 5 .php files and one .css file.
The main page is index.php:
The first function is a search function in the Professor table with a given IdNummer, which returns all available data as a table. The second one with a given Nachname, and another one is a function to insert data in the table.


Alle Professoren

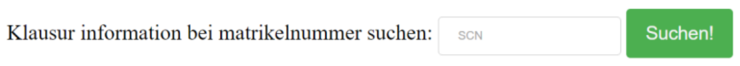
Suchen Professor nach Nachname: Schneider

Suche nach Id: 347

Neue Professor einfuegen:

Idnummer	Vorname	Nachname	Email	Geburtsdatum	Gehalt	Uni_
347	Thomas	Klausner	thomas_klausner@gmail.com	e.g. 01-01-1985	4000	45

Back to Homepage


Klausur information bei matrikelnummer suchen: SCN

Professor ProfessorByUni Pruefung_info Raum_info Note_info

This is a Heading

This is a paragraph.