Danrong Li (dl4111) & Yuqi Tang (yt2101)

Port number: 8702

A High-level Description of Your Application

Our application is an online information system for algorithm/programming classes in NYU. We created this database to document student users working on the online coding questions (eg. leetcode problems) that are assigned by their classes which are offered by departments and instructed by professors in NYU.

All Entity Sets, Relationship Sets, and Business Rules

1. The entity sets are as follows:

Students, Classes, Departments, Professors, Questions, Examples, Constraints, Language

2. Entity cluster: University
3. Relationship sets are as follows:

with_details, assigned, instructed_by, offer_by, comment. Weak relationship sets: have, follow.

4. Business rules are as follows:
    I. Students can be assigned any number of questions. Questions can be completed by any number of students. Students can complete any number of questions. Students can use any number of languages to complete the question. The languages can be used by any number of students to complete the questions.
    II. Students can comment on any number of questions. Questions can be commented on by any number of students. Students can be inside any number of classes. Classes can have any number of students. Classes can assign any number of questions.
    III. Every detail will belong to exactly one student. Every comment is made by exactly one student.
    IV. All questions have examples. And for every example, it only has 1 matching question. Some questions follow constraints. And for every constraint, it only has 1 matching question.
    V. Each class is instructed by exactly one professor. Classes can be offered in any number of departments. Departments can offer any number of classes. Professors can instruct any number of classes.
    VI. Classes are offered by departments during different terms. For each department, only the most recent offering was recorded.

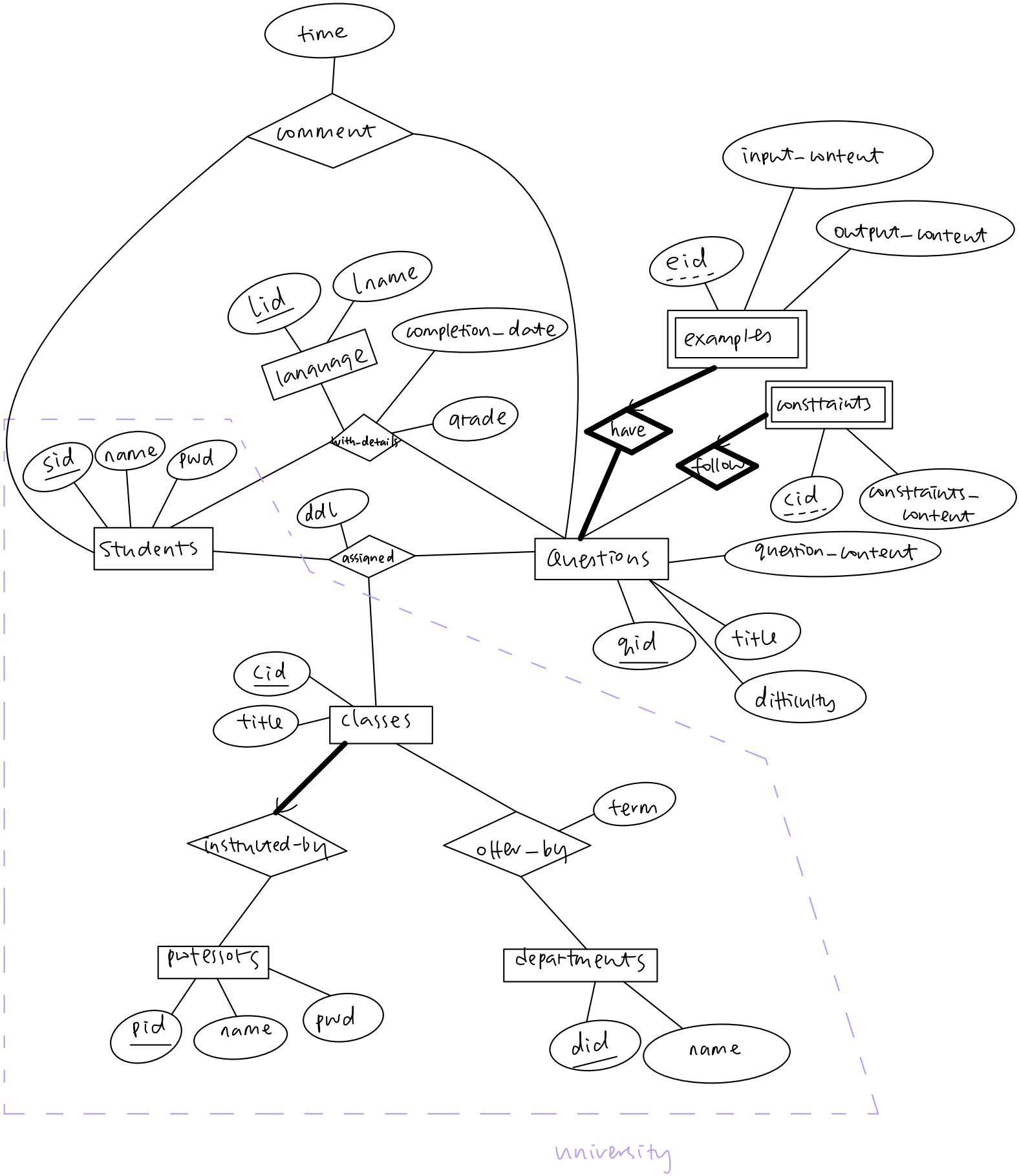
Plan to Acquire Data

We are planning to use leetcode as our questions. We will get the questions and question attributes from the leetcode website. Then we can go to the NYU Albert course catalog to obtain our classes, departments and professors data. Some trivial data, like the language, grade, date etc, we will make them up realistically. The data we acquired are in csv format inside the data folder.

User Interaction

Users would ask the following questions:

1. Given a leetcode question id, list the question difficulty level, question title, question content, input example and output example.
2. Given a student name, list the number of questions the student completed. For each question completed, show the question id, completion timestamp, grade received and language used.
3. Given a student name, list all the assigned leetcode questions as well as their ddl.
4. Given a student name, list all the questions that he/she commented on before, as well as the timestamp.
5. Given a class name, list the number of students who took the class, as well as their names.
6. Given class name, list all students who attempted the leetcode problems.(may complete after ddl)
7. Given class name, list all students who have finished their assignments before ddl.
8. Given a class name, list the students' performance(percentage of students completed the assigned problems before the ddl) in the class.
9. Given a term, list the number of classes offered in that term as well as the class names.
10. Given a term, list the names of professors who are teaching in that term.
11. Given a term, list the names of departments that are offering classes in that term.
12. Given a department name, list all the classes they provided.
13. Given a professor name, list the names of classes that are instructed by that professor.
14. Given a professor name, list the questions that they assigned to their classes.



team members: Dantong Li & Yuqi Tang

note: we cannot represent participation of Questions

```
drop table assigned;
drop table offer_by;
drop table with_details;
drop table have_examples;
drop table follow_constraints;
drop table language;
drop table comment;
drop table students;
drop table questions;
drop table departments;
drop table classes_instructed;
drop table professors;


create table questions(
        qid int primary key,
        difficulty int,
        title varchar(255) ,
        question_content varchar(1024)
);

create table have_examples(
        eid int ,
        qid int ,
        primary key(eid,qid),
        input_content varchar(255) ,
        output_content varchar(255) ,
        foreign key(qid) references questions(qid) on delete cascade
);

create table follow_constraints(
        cid int ,
        constraints_content varchar(1024) ,
        qid int ,
        primary key(cid,qid),
        foreign key(qid) references questions(qid) on delete cascade
);


create table students(
        sid int primary key,
        name varchar(255) ,
        pwd varchar(255)
);

create table comment(
        sid int ,
        qid int ,
        time timestamp ,
        primary key(sid,qid),
```

```
        foreign key(sid) references students(sid),
        foreign key(qid) references questions(qid)
);



create table language(
        lid int primary key,
        lname varchar(255)
);

create table with_details(
        grade varchar(255) ,
        completion_date timestamp ,
        lid int ,
        sid int ,
        qid int ,
        primary key(lid,sid,qid),
        foreign key(sid) references students(sid),
        foreign key(qid) references questions(qid),
        foreign key(lid) references language(lid)
);

create table professors(
        pid int primary key,
        name varchar(255) ,
        pwd varchar(255)
);

create table classes_instructed(
        cid int primary key,
        title varchar(255) ,
        pid int not null,
        foreign key(pid) references professors(pid)
);

create table assigned(
        ddl timestamp ,
        sid int ,
        qid int ,
        cid int,
        primary key(sid,qid, cid),
        foreign key(sid) references students(sid),
        foreign key(qid) references questions(qid),
        foreign key(cid) references classes_instructed(cid)
);

create table departments(
        did int primary key,
        name varchar(255)
```

```
);

create table offer_by(
        term varchar(255) ,
        cid int ,
        did int ,
        primary key(cid, did),
        foreign key(cid) references classes_instructed(cid),
        foreign key(did) references departments(did)
);
```