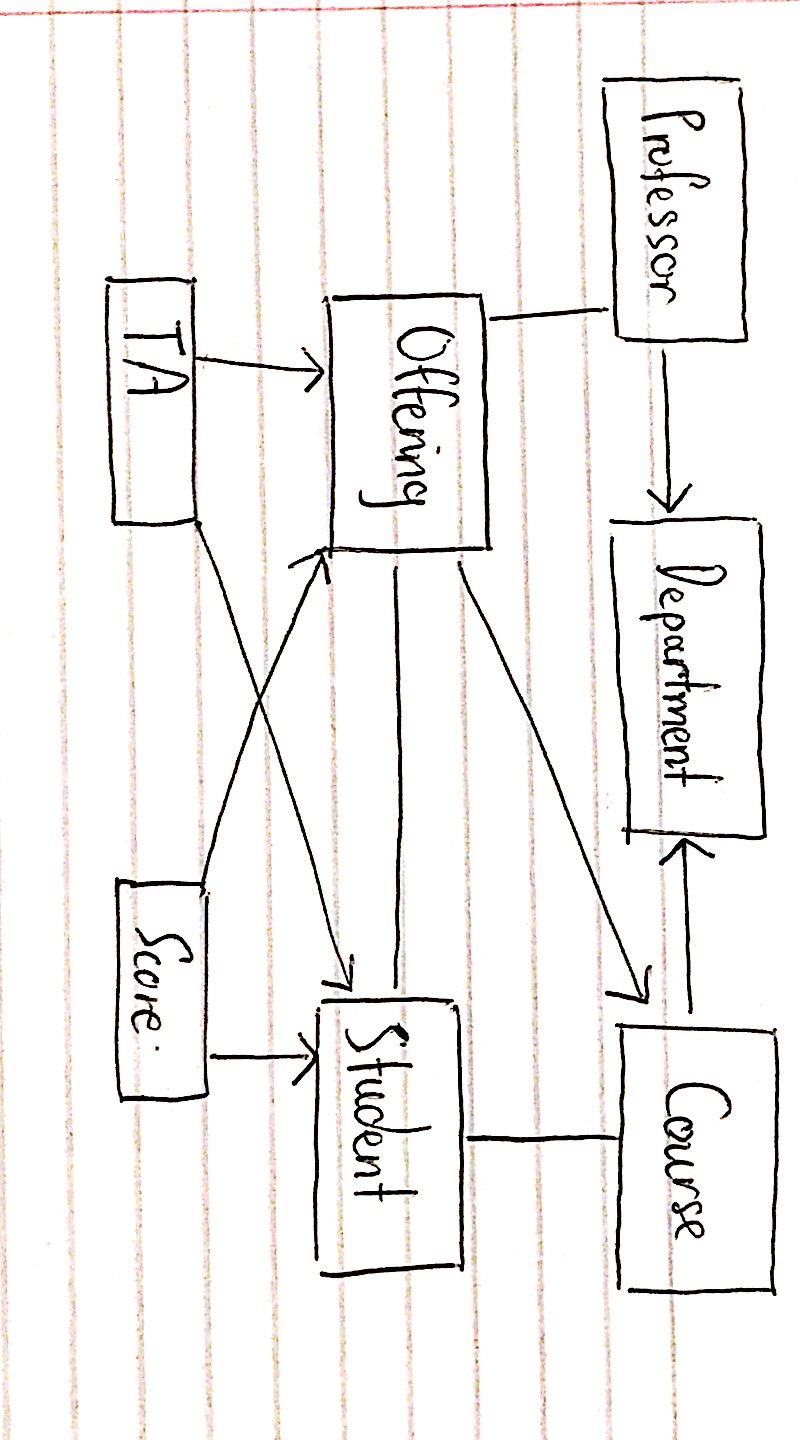
Phan Anh Nguyen Prof. Jack Orenstein

Comp 115

Assignment 3

**Part 1.1:**

Entity/Relationship diagram:



**Part 1.2:**

**Professor:**

University ID

Department

Chair (bool)

Name

Email address

Phone number

Website link

**Department:**

Department name

Department chair

Address

Phone number

Website link

**Course:**

Department

Catalog number

Title

Description

Prerequisite

**Student:**

Name

University ID

Courses taken

**Offering:**

Course

Semester

Professor

TAs

Task

**Score:**

Offering

Student ID

Task

Score (Weight is incorporated here)

**TA:**

Student ID

Offering

**Part 2.1:**

create table Professor (

p\_id int(8) not null **primary key**,

dept\_name varchar not null references Department,

chair bool not null;

prof\_name varchar not null,

email varchar not null,

phone int(10) not null,

website varchar);

create table Department (

dept\_name varchar not null **primary key**,

prof\_name varchar not null references Professor // Used for selective dept chair name

address varchar not null,

phone int(10) not null,

website varchar not null);

create table Course (

c\_id varchar not null **primary key**,

dept\_name varchar not null references Department,

title varchar not null,

description varchar not null,

prerequisite varchar);

create table Student (

name varchar not null,

s\_id(8) not null **primary key**,

c\_id varchar not null references Course); // To reference courses taken

create table Offering (

c\_id varchar not null references Course,

p\_id varchar not null references Professor,

semester varchar not null,

task varchar not null, // To denote evaluation tasks.

Primary key (c\_id, p\_id, semester));

Create table TA (

c\_id varchar not null references Course,

p\_id varchar not null references Professor,

semester varchar not null,

s\_id int(8) not null references Student, // To reference TAs

Primary key (c\_id, p\_id, semester, s\_id));

create table Score (

c\_id varchar not null references Course,

p\_id varchar not null references Professor,

semester varchar not null,

s\_id not null references Student, //To reference students enrolled

Task not null references Offering,

score int(3), // Note that score incorporates weight as max score for a task is weight

Primary key (c\_id, p\_id, semester, s\_id, task);

**Part 2.2:**

For my schema, we know that there are 4 main entities that have natural keys which are Courses, Students, Professor, and Department. The entities Offering, TA and Score arise from relationships between courses, students and professors. For this schema, I tried to not use surrogate keys as due to the nature of the 3-entity relationship, it is possible to use a combination of the natural keys p\_id, c\_id and a semester string (i.e FALL2018) to uniquely identify an Offering. Within an offering is TA and Score. The primary key of a TA relationship between an offering and a student can be modelled by incorporating the key used for an offering and the natural key for Student (s\_id). Similarly, Score can be identified based off of the primary used for offerings, in addition with an s\_id of a student taking the course, followed by the task that is being evaluated.

**Part 3.1:**

List of possible FDs is:

X -> Y

X -> Z

Y -> X

Y -> Z

Z -> Y

Z -> X

X -> Y, Z

Y -> X, Z

Z -> X, Y

X, Y -> Z

X, Z -> Y

Y, Z -> X

**Part 3.2:**

FDs that can be ruled out are:

X -> Y

X -> Z

Y -> X

Z -> X

Z -> Y

X -> Y, Z

Y -> X, Z

Z -> X, Y

Y, Z -> X

X, Z -> Y

**Part 3.3:**

FDs that can be ruled out are:

X -> Y

X -> Z

X -> Y, Z

**Part 3.4:**

No FDs can be ruled out.

**Part 4.1:**

Candidate keys are {V}, {X}

FD that violates BCNF: X -> Y, Z

Closure of dependency: {X, Y, Z}

Decompose (V,W,X,Y,Z) using {X}

C = {X, Y, Z}

S = {X, Y, Z}

T = {V, W, X}

Now we have both S and T in BCNF.

**Part 4.2:**

Candidate keys are {W,X}, {X}, {Y}

**Step 1:**

FD that violates BCNF: X -> Z

Closure of dependency: {X, Z}

Decompose (W,X,Y,Z) using {X}

C = {X, Z}

S = {X, Z}

T = {W, X, Y}

Now we have both S and T in BCNF. But we are not finished yet

**Step 2:**

FD that violates BCNF: Y -> W

Closure of dependency: {Y, W}

Decompose (W,X,Y) using {W}

C = {W, Y}

S = {W, Y}

T2 = {X, W}

Now we have both S and T and T2 in BCNF.