Adam Gordon

Samantha Goleco

Bryan Trinh

Kelly Suen

1. An introductory paragraph providing information about the organization

The organization we are writing about is San Francisco State University. It is a local university and the area we are focusing on is academics and class registration. Most data about courses, colleges, and departments can be easily obtained from the school’s website, and student data can be obtained from the registrar's office. For this project, we will make up data about instructors, students, and other hard to obtain data.

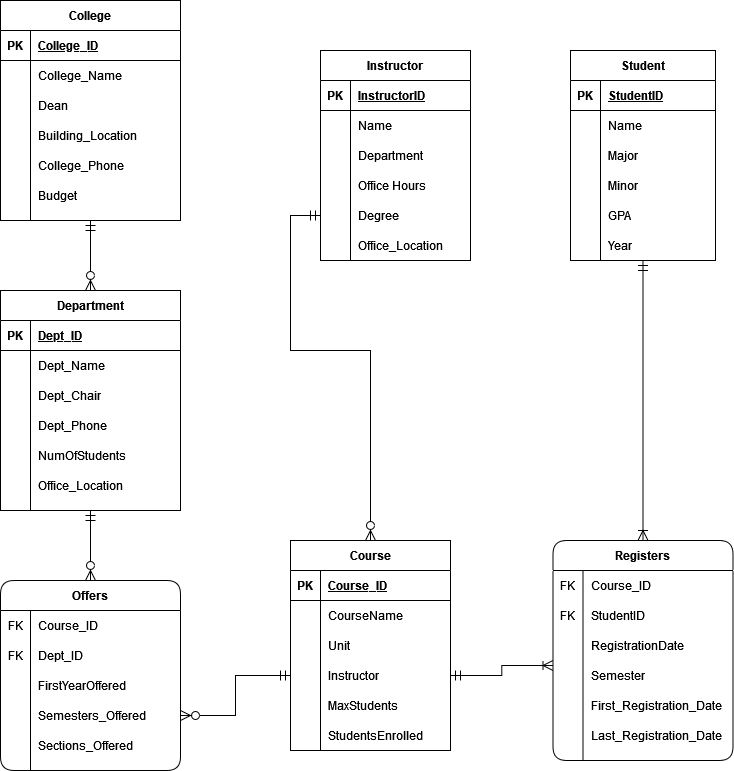
1. A list of all the entities (not the attributes) and a short description for each.

* College - subset of the university that focuses on a subject
* Department - the subset of a college that focuses even deeper on a particular academic discipline.
* Student - person that takes at least one course
* Instructor - person that teaches zero or more courses
* Course - a student registers for a course and an instructor teaches a course
* Registers - when a student enrolls in a course
* Offers - a department offers a course

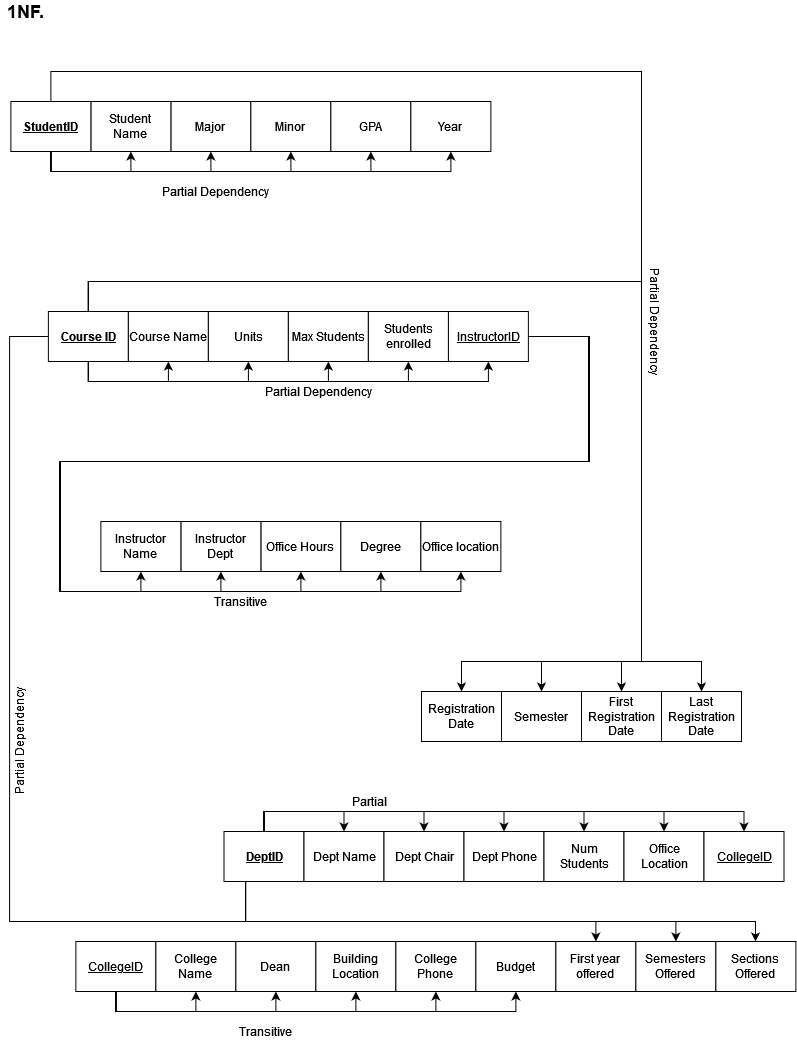
1. A list of ALL the Business Rules used by your organization.

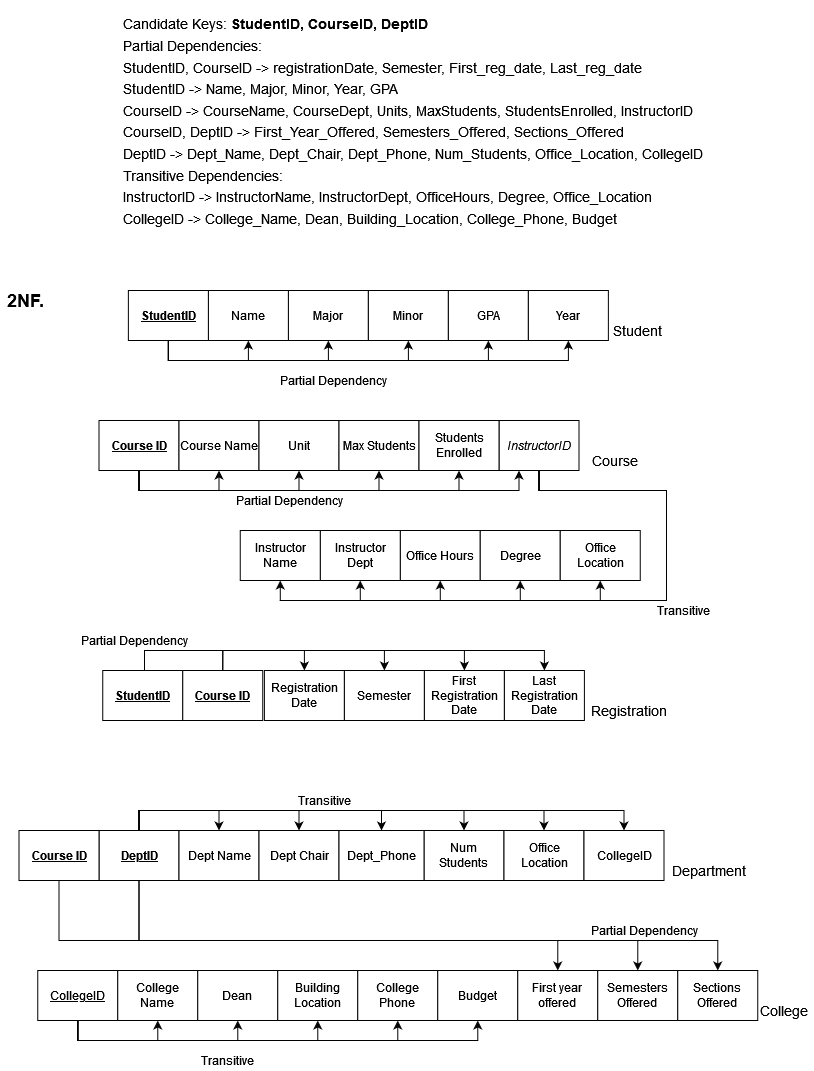
* Each student must register for at least one course.
* Each course must have at least one student registered.
* Each course must have one instructor.
* Each instructor may teach many courses.
* Each course may be offered by many departments.
* Each department may offer many courses.
* Each department must belong to one college.
* Each college may contain more than one department.

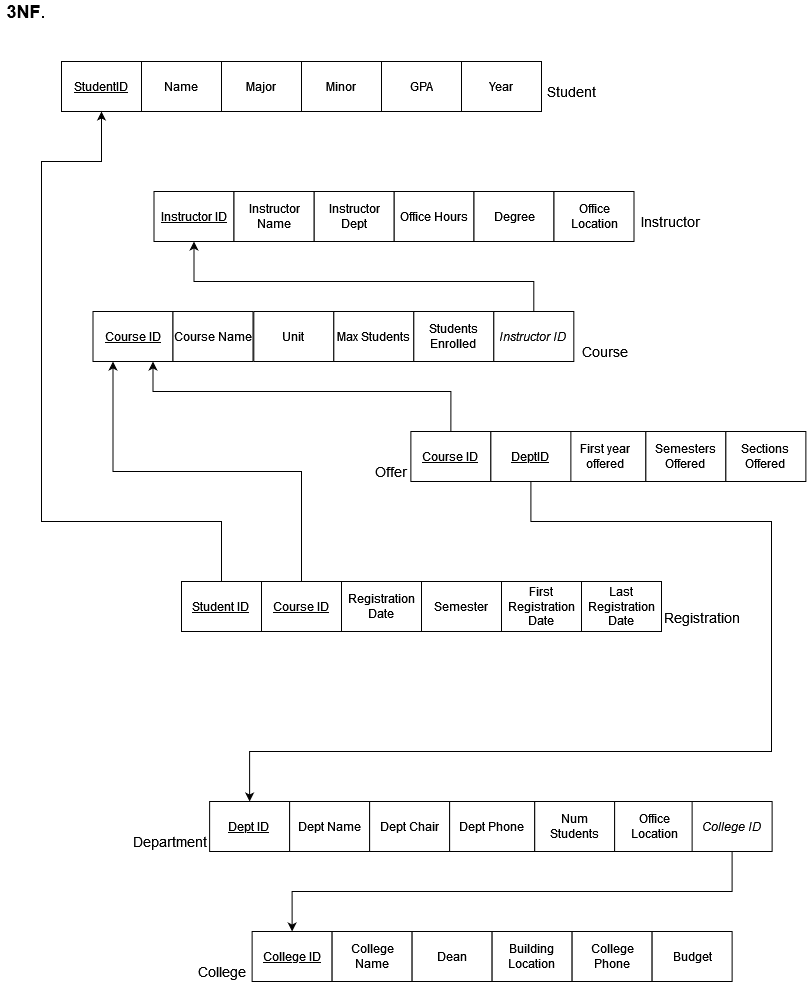
1. An ER diagram



1. A relation model







-- 6a. A list of SQL statements used to create our tables:

CREATE SCHEMA IF NOT EXISTS university\_db;

USE university\_db;

-- Adam Gordon

-- Samantha Goleco

-- Bryan Trinh

-- Kelly Suen

CREATE TABLE Student (

StudentID INT NOT NULL,

StudentName VARCHAR(20),

Major VARCHAR(20) DEFAULT "Undeclared",

Minor VARCHAR(20),

GPA DOUBLE,

StudentYear VARCHAR(20)

CHECK(StudentYear IN ("Freshman", "Sophomore", "Junior", "Senior")),

CONSTRAINT student\_pk PRIMARY KEY (StudentID)

);

CREATE TABLE Instructor (

InstructorID INT NOT NULL,

InstructorName VARCHAR(20),

InstructorDept VARCHAR(20),

OfficeHours VARCHAR(20),

Degree VARCHAR(20),

OfficeLocation VARCHAR(20), -- building name

CONSTRAINT instr\_pk PRIMARY KEY (InstructorID)

);

CREATE TABLE College (

CollegeID INT NOT NULL,

CollegeName VARCHAR(50),

Dean VARCHAR(20),

BuildingLocation VARCHAR(20), -- building name

CollegePhone VARCHAR(15),

Budget INT,

CONSTRAINT college\_pk PRIMARY KEY (CollegeID)

);

CREATE TABLE Department (

DepartmentID INT NOT NULL,

DepartmentName VARCHAR(50),

DepartmentChair VARCHAR(50),

DepartmentPhone VARCHAR(50),

NumOfStudents INT,

OfficeLocation INT, -- room number

CollegeID INT NOT NULL,

CONSTRAINT dept\_pk PRIMARY KEY (DepartmentID),

CONSTRAINT dept\_fk FOREIGN KEY (CollegeID) REFERENCES College (CollegeID)

);

CREATE TABLE Course(

CourseID INT NOT NULL,

CourseName VARCHAR(50),

Unit INT,

MaxStudents INT,

StudentsEnrolled INT,

InstructorID INT NOT NULL,

CONSTRAINT course\_pk PRIMARY KEY (CourseID),

CONSTRAINT course\_fk FOREIGN KEY (InstructorID) REFERENCES Instructor(InstructorID)

);

CREATE TABLE Offer(

CourseID INT NOT NULL,

DepartmentID INT NOT NULL,

FirstYearOffered YEAR, -- the year the course was first offered

Semesters\_Offered VARCHAR(6)

CHECK(Semesters\_Offered IN ("Spring", "Summer", "Fall", "Winter")),

Sections\_Offered INT,

CONSTRAINT offer\_pk PRIMARY KEY (CourseID, DepartmentID),

CONSTRAINT offer\_fk1 FOREIGN KEY (CourseID) REFERENCES Course (CourseID),

CONSTRAINT offer\_fk2 FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)

);

CREATE TABLE Registration (

StudentID INT NOT NULL,

CourseID INT NOT NULL,

RegistrationDate DATE,

Semester VARCHAR(6)

CHECK(Semester IN ("Spring", "Summer", "Fall", "Winter")),

FirstRegistrationDate DATE,

LastRegistrationDate DATE,

CONSTRAINT reg\_pk PRIMARY KEY (StudentID, CourseID),

CONSTRAINT reg\_fk1 FOREIGN KEY (StudentID) REFERENCES Student (StudentID),

CONSTRAINT reg\_fk2 FOREIGN KEY (CourseID) REFERENCES Course (CourseID)

);

-- 6b. SQL statements used to load data into our tables

INSERT INTO Student (StudentID, StudentName, Major, Minor, GPA, StudentYear) VALUES

(900000001, "Jerry Seinfeld", "English", "History", 3.5, "Senior"),

(900000002,"George Costanza","Business", null, 2.5, "Senior"),

(900000003, "Rachel Green", "Film", null, 3.1, "Freshman"),

(900000004,"Stephen Yoon","Psychology", "Liberal Arts", 3.3, "Sophomore"),

(900000005,"Alex Martinez","Art","Theater", 3.6, "Junior"),

(900000006,"Mary Burrell","Biology","Psychology", 3.8, "Sophomore"),

(900000007, "Stu Dent", "Art", null, 3.0, "Senior"),

(900000008, "Mordecai Rigby", default, null, 2.0, "Senior");

INSERT INTO Instructor (InstructorID, InstructorName, InstructorDept, OfficeHours, Degree, OfficeLocation) Values

(100000001, "Tea Cher","Biology", null, "Biology","Sci"),

(100000002, "Eric Son","Computer Science ", Wednesday, noon, "Computer Science","Bus"),

(100000003, "Johnny Suh","Business Administration", null, "Business","Bus"),

(100000004, "Chase Cooper", "Science", "Monday, morning", "Biology", "Bio"),

(100000005, "Chris Bridgmen", "English", "Mondays, noon", "English", "Humanities");

INSERT INTO College (CollegeID, CollegeName, Dean, BuildingLocation, CollegePhone, Budget) VALUES

(98, "College of Science and Engineering", "Carmen Domingo", "TH", "415-338-1571", 6541234),

(45, "College of Liberal & Creative Arts", "Andrew Harris", "CA", "415-338-1471", 5555555),

(56, "College of Education", "Cynthia Grutzik", "BH", "415-338-2687", 2957111),

(34, "College of Health & Social Sciences", "Alvin Alvarez", "HSS", "415-338-3326", 3928545),

(64, "College of Business", "Eugene Sivadas", "BUS", "415-338-2138", 42764321),

(25, "College of Ethnic Studies", "Amy Sueyoshi", "EP", "415-338-1693",2363781 ) ;

INSERT INTO Department (DepartmentID, DepartmentName, DepartmentChair,

DepartmentPhone, NumOfStudents, OfficeLocation, CollegeID) VALUES

( 987, "Biology", "Laura Burrus", "415-338-1548", 574, 538, 98),

( 456, "Information Systems", "Lutfus Sayeed", "415-338-2138", 1000, 310, 64),

( 295, "Counseling", "Rebecca Toporek", "415-338-2005", 516, 021, 34),

( 256, "Criminal Justice", "Elizabeth Brown", "415-405-4129", 980, 424, 34),

( 436, "Economics", "Anoshua Chaudhuri", "415-338-2108",786 ,215 , 64),

(460 , "Accounting", "Amy Chang", "415-338-1107",187 ,415 ,64) ;

INSERT INTO Course (CourseID, CourseName, Unit, MaxStudents, StudentsEnrolled, InstructorID) VALUES

( 7894, "Introductory Biology", 4, 25, 20, 100000001),

( 7109, "Introduction to Computer Science", 3, 30, 27, 100000002),

( 7124, "Introduction to Information Systems", 3, 28, 23, 100000002),

( 2432, "Microeconomics",3 ,30 ,28 , 100000003),

( 7562 , "Building Business Applications",3 ,30 ,25 ,100000002 ),

( 2932, "Managerial Accounting", 3, 50, 30, 100000005),

( 8765, "English Literature", 3, 20, 20, 100000005);

INSERT INTO Offer (CourseID, DepartmentID, FirstYearOffered, Semesters\_Offered, Sections\_offered) VALUES

( 7894, 987, 2009, "Spring", 1),

( 2432, 436, 2013, "Fall", 2),

( 7124, 456, 2019, "Spring", 4),

( 7562 , 456, 2017, "Fall",4 ),

( 2932, 460 ,2020 , "Fall", 5);

INSERT INTO Registration (StudentID, CourseID, RegistrationDate, Semester, FirstRegistrationDate, LastRegistrationDate) VALUES

( 900000001, 7894, "2021-05-15", "Fall", "2021-05-10", "2021-08-10"),

( 900000001, 7124, "2021-05-15", "Fall", "2021-05-10", "2021-08-10"),

( 900000002, 2432, "2021-05-10", "Fall", "2021-05-10", "2021-08-10"),

( 900000003, 7124, "2021-05-13", "Fall", "2021-05-10", "2021-08-10"),

( 900000004, 8765 ,"2021-11-29", "Spring","2021-11-25","2021-01-23"),

( 900000004, 7124 ,"2021-11-29", "Spring","2021-11-25","2021-01-23"),

( 900000005, 7894 , "2021-12-02", "Spring", "2021-11-25", "2022-01-23");

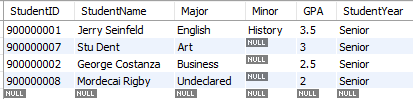
-- 6c. Retrieve information from our tables

-- 1. We want to know the GPAs of all senior students. List all senior students in descending GPA order.

SELECT \* FROM Student

WHERE StudentYear = "Senior"

ORDER BY GPA DESC;



-- 2. We want to know the ID and name of all students who have registered for an introductory (or introduction) course. Do not show duplicates.

SELECT Student.StudentID, StudentName

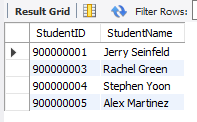
FROM Student, Registration

WHERE Student.StudentID = Registration.StudentID

AND Registration.CourseID IN

(SELECT CourseID FROM Course WHERE CourseName LIKE "Intro%" )

GROUP BY Student.StudentID;



-- 3. We want to know the number of students taught by each professor, including professors that are not currently teaching.

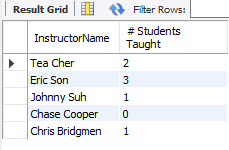
SELECT InstructorName, COUNT(Registration.StudentID) AS "# Students Taught"

FROM

(Instructor LEFT OUTER JOIN Course ON Instructor.InstructorID = Course.InstructorID)

LEFT OUTER JOIN Registration ON Course.CourseID = Registration.CourseID

GROUP BY InstructorName;



-- 4. We want to know the name, ID, and number of courses of instructors that are teaching more than 1 class.

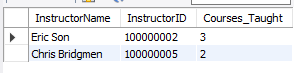
SELECT InstructorName, Instructor.InstructorID, COUNT(CourseID) AS Courses\_Taught

FROM Instructor, Course

WHERE Course.InstructorID = Instructor.InstructorID

GROUP BY InstructorID

HAVING Courses\_Taught > 1;



-- 5. We want to know the ID, name, and GPA of all senior students with GPAs greater than 3.0, using a view. Sort by descending GPA order.

CREATE VIEW SeniorStudents\_V

AS

SELECT \* FROM Student

WHERE StudentYear = "Senior";

SELECT StudentID, StudentName, GPA FROM SeniorStudents\_V

WHERE GPA >= 3.0

ORDER BY GPA DESC;



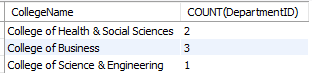
-- 6. Display the number of departments in each college. EXCLUDE colleges that do not have departments.

SELECT College.CollegeName, COUNT(DepartmentID)

FROM College RIGHT OUTER JOIN Department

ON College.CollegeID = Department.CollegeID

GROUP BY College.CollegeID;

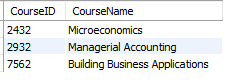


-- 7. Display course name and course ID of all courses offered in the fall.

SELECT Course.CourseID, Course.CourseName FROM Course, Offer

WHERE Course.CourseID = Offer.CourseID

AND Semesters\_Offered = "Fall";



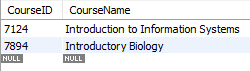
-- 8. We want to know what courses Jerry Seinfeld has registered for using a subquery.

SELECT CourseID, CourseName FROM Course

WHERE CourseID IN

(SELECT CourseID FROM Registration

WHERE StudentID = 900000001);



-- 9. We want to know the number of students enrolled in Introductory Biology.

SELECT StudentsEnrolled FROM Course

WHERE CourseName = "Introductory Biology";



7. Conclusion

Some of the benefits we as a group gained from working together is learning how to code sql for a college database. Along with creating sql, we were able to work over time to get everything together such as our ERD, 1NF-3NF, and sql database. Being in a university such as SFSU allowed us to already have some knowledge to be able to handle this database. Since we are already familiar with the SFSU’s database when signing up for classes it helped us to create our own.