I. System analysis and ER model

Problem Statement: The registrar wants to effectively manage the course registration system at Baruch College. Students can be identified by their EMPLID, First Name, Last Name, Phone, Address, Email, Major, minor, major concentration, Date of Birth, Admission Date, Classification(year), GPA, Academic Status, and proof of ID. Each student can enroll in multiple courses and multiple students can enroll in one course. They are further categorized by their start date. A course is defined by its Course_ID, Course Name, Course Duration, Description, Credits. A course is then further separated into sections. Section would include Section_ID, Semester, Capacity(#students), MeetingDay, Meeting Time, year. Each section would then represent a class. A class could be represented with Classroom_ID, room number, classroom capacity, classroom type, reservation status. A professor is responsible for teaching the students. A professor can be defined by their Professor_ID, First Name, Last Name, Email, Phone Number, Office Hours, Office, Courses specialization, Education history. Each professor can teach a minimum of 14 students. At the same time, a professor has to teach at least one course but can teach multiple courses. Each professor is a part of a department. A department is defined by its Department_ID, Department head, Phone number, address, email, specialization, Courses Offered. There can be multiple professors within a department and the department can offer one or multiple courses to its students. The students can also make an appointment to meet with an advisor. An appointment is defined by its Appointment ID, appointment Date, and appointment time. An advisor is defined by their advisor ID, first name, last name, and email. An advisor can have no appointments at all but can take multiple appointments. They also can advise multiple students, but a student can choose not to meet with an advisor, which means no appointments for the student either. However, a student can only have one appointment per semester.

Assumptions:

- Baruch only has one registrar office
- Students cannot enroll in classes before the starting date or after the deadline date, without exceptions
- The registrar office hands out all the holds
- Students have met the prerequisites of the class they are trying to enroll in
- Students don't have any holds during their enrollment period
- 14 students are required for a new course to be created and a professor can only teach a course with a minimum of 14 students.

Relationship Sentences:

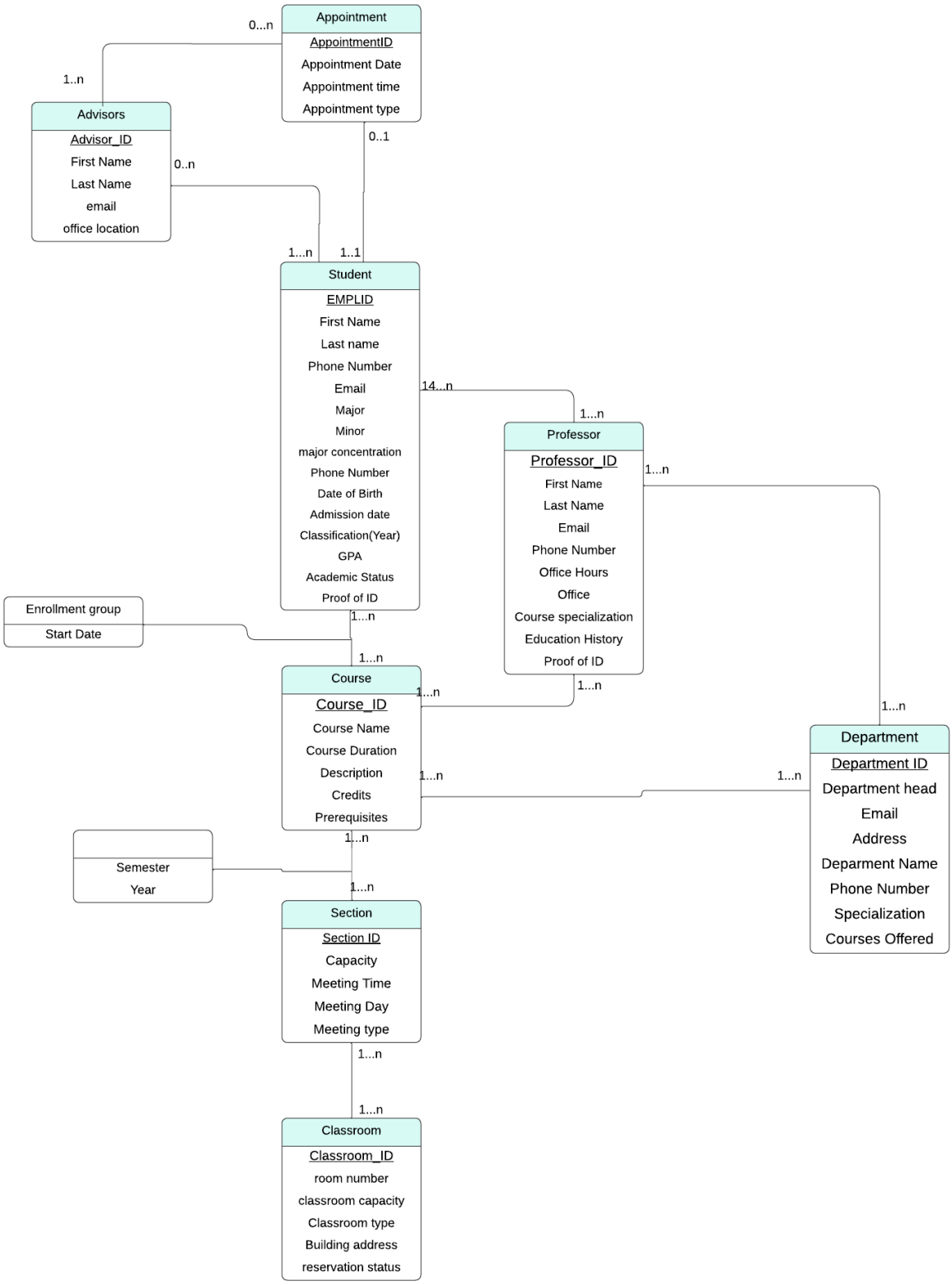
- Each student can enroll in multiple courses and multiple students can enroll in one course.
- Each professor can teach a minimum of 14 students.
- a professor has to teach at least one course but can teach multiple courses.
- There can be multiple professors within a department and the department can offer one or multiple courses to its students.
- An advisor can have no appointments at all but can take up to multiple appointments.
- They also can advise multiple students, but a student can choose not to meet with an advisor, which means no appointments for the student either.
- a student can only have one appointment per semester
- A student can have no holds or up to multiple holds at the same time.

Entities and Attributes

Our First Iteration

Entities	Attributes
Advisors	Advisor_ID, First Name, Last Name, Email, office location
Section	Section_ID, Semester, Capacity(#students), MeetingDay, Meeting Time, year
Department	Department_ID, Department head, Phone number, address, email, specialization, Courses Offered
Course	Course_ID, Course Name, Course Duration, Description, Credits, Prerequisites
Student	EMPLID, First Name, Last Name, Phone, Address, Email, Major, Minor, major concentration, Date of Birth, Admission Date, Classification(year), GPA, Academic Status, Proof of identification
Professor	Professor_ID, First Name, Last Name, Email, Phone Number, Office Hours, Office, Courses specialization, Education history, Proof of Identification
Appointment	appointment date, appointment time, appointmentID, appointment type
Classroom	Classroom_ID, room number, classroom capacity, classroom type, reservation status

Entity Relation Diagram


Appointment
AppointmentID
Appointment Date
Appointment time
Appointment type
0...n
1..n
Advisors
Advisor_ID
First Name
Last Name
email
office location
0..n
0..1
1...n
1..1
Student
EMPLID
First Name
Last name
Phone Number
Email
Major
Minor
major concentration
Phone Number
Date of Birth
Admission date
Classification(Year)
GPA
Academic Status
Proof of ID
14...n
1...n
Professor
Professor_ID
First Name
Last Name
Email
Phone Number
Office Hours
Office
Course specialization
Education History
Proof of ID
1...n
Enrollment group
Start Date
1...n
1...n
Course
Course_ID
Course Name
Course Duration
Description
Credits
Prerequisites
1...n
1...n
1...n
1...n
Department
Department ID
Department head
Email
Address
Deparment Name
Phone Number
Specialization
Courses Offered
Semester
Year
1...n
1...n
Section
Section ID
Capacity
Meeting Time
Meeting Day
Meeting type
1...n
1...n
Classroom
Classroom_ID
room number
classroom capacity
Classroom type
Building address
reservation status

Logical Modeling

Relationship	**Attributes**
Student	EMPLID, First Name, Last Name, Phone, Address, Email, Major, Minor, major concentration, Date of Birth, Admission Date, Classification(year), GPA, Academic Status, proof of ID
Student_Course	Student_Course_ID, Course ID(fk), EMPLID(fk), Enrollment group, startdate
Course	Course ID, Course name, description, course duration, credits
Course_Section	Course_Section_ID, CourseID(fk), SectionID(fk), Semester, year
Section	SectionID, Capacity, meeting time, meeting day
Classroom	Classroom_ID, room number
Department	Department_ID, Department head, Phone number, address, email, specialization, Courses Offered
Department_Course	Department_Course_ID, Department_ID(fk), Course_ID(fk),
Professor	Professor_ID, First Name, Last Name, Email, Phone Number, Office Hours, Office, Courses specialization, Education history, proof of ID
Professor_Department	Professor_Department_ID, Professor ID(fk), Department ID(fk)

Professor_Student	Professor_Student ID, EMPLID(fk), ProfessorID(fk)
Professor_Course	Professor_Course_ID, Professor ID(fk), Course ID(fk)
Appointment	Appointment ID, Appointment Date, Appointment time
Advisors	Advisor_ID, Firstname, last name, email
Appointment_Advisor	Appointment_Advisor_ID, Appointment ID(fk), Advisor ID(fk)
Advisors_Student	Advisor Student ID, EMPLID(fk), Advisor ID(fk)

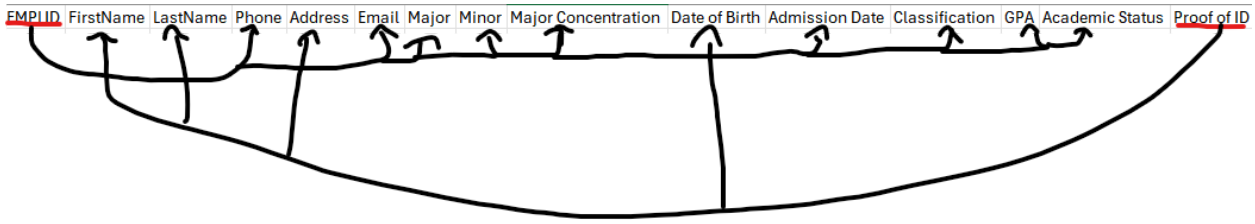
Normalization

Assumptions about normalization exercise:

- All tables not included were considered normalized after our first iteration
- Intended to demonstrate how we worked through this process
- Conceptualization necessitated expansion of attributes during this process

Process: I chose student as the relationship to normalize, as it was only in 2NF

Relationship	Attribute
Student	EMPLID, First Name, Last Name, Phone, Address, Email, Major, Minor, major concentration, Date of Birth, Admission Date, Classification(year), GPA, Academic Status, proof of ID



Functional dependencies

Key:EMPLID, Proof of ID
FD1:EMPLID→ Phone, Email, Major, Minor, major concentration, Date of Birth, Admission Date, Classification(year), GPA, Academic Status
FD2: Proof of ID → First Name, Last Name, Address, Date of Birth

Normalization Process

1NF: Student'sCourse(EMPLID, First Name, Last Name, Phone, Address, Email, Major, Minor, major concentration, Date of Birth, Admission Date, Classification(year), GPA, Academic Status, Proof of ID)
It was defaulted to 1NF before I began the normalization process

2NF:AcademicRecord(EMPLID, Phone, Email, Major, Minor, major concentration, Admission Date, Classification(year), GPA, Academic Status)
StudentInfo(Proof of ID, First Name, Last Name, Address, Date of Birth**)**
2NF sees that there are no partial dependencies. Academic Record is created

3NF: StudentInfo(Proof of ID,First Name, Last Name, Address, Date of Birth)
Contact Info(EMPLID, Phone, Email,)
StudyConcentration (EMPLID, Major, MajorConcentration, Minor)
AcademicRecord(EMPLID, Admission Date, Classification(year), GPA, Academic Status)
3NF sees that the transitive dependencies are removed and put into a new tables and in addition, created separate tables for both major and minors

Result:

Relationship	Attributes
StudentInfo	Proof of ID, First Name, Last Name, Address, Date of Birth
StudyConcentration	EMPLID, Major, MajorConcentration, Minor
AcademicRecord	EMPLID, Admission Date, Classification(year), GPA, Academic Status
ContactInfo	EMPLID, Phone, Email

Database Schema with Structured Query language

CREATE TABLE Student (
EMPLID INT NOT NULL,
FirstName VARCHAR(45) NOT NULL,
LastName VARCHAR(45) NOT NULL,
Email VARCHAR(45) NULL,
Phone VARCHAR(12) NULL,
Major VARCHAR(45) NULL,
Minor VARCHAR(45) NULL,
Majorconcentration VARCHAR(45) NULL,
DateofBirth DATE NOT NULL,
AdmissionDate DATE NOT NULL,
Classification VARCHAR(9) NOT NULL,
GPA INT NULL,
AcademicStatus Double NOT NULL,
PRIMARY KEY (EMPLID));

```
CREATE TABLE Professor (
ProfessorID INT NOT NULL,
FirstName VARCHAR(45) NULL,
LastName VARCHAR(45) NULL,
Email VARCHAR(45) NULL,
Phone VARCHAR(12) NULL,
OfficeHours VARCHAR(45) NULL,
Office VARCHAR(12) NULL,
CoursesSpecialization VARCHAR(45) NULL,
DateofBirth DATE NOT NULL,
EducationHistory VARCHAR(100) NULL,
PRIMARY KEY (ProfessorID));

CREATE TABLE Department (
DepartmentID INT NOT NULL,
DepartmentHead VARCHAR(45) NULL,
Email VARCHAR(45) NULL,
Phone VARCHAR(12) NULL,
Address VARCHAR(50) NULL,
Specialization VARCHAR(45) NULL,
CoursesOffered VARCHAR(100) NULL,
PRIMARY KEY (DepartmentID));

CREATE TABLE Classroom (
ClassroomID INT NOT NULL,
RoomNumber VARCHAR(12) NULL,
ClassroomCapacity INT NULL,
ClassroomType VARCHAR(45) NOT NULL,
ReservationStatus BIT NOT NULL,
PRIMARY KEY (ClassroomID));

CREATE TABLE Section (
SectionID INT NOT NULL,
Semester VARCHAR(45) NOT NULL,
Capacity INT NOT NULL,
MeetingDay DOUBLE NOT NULL,
```

```
MeetingTime DOUBLE NOT NULL,
Year INT NOT NULL ,
PRIMARY KEY (SectionID));

CREATE TABLE Course (
CourseID INT NOT NULL,
CourseName VARCHAR(45) NULL,
CourseDuration DOUBLE NOT NULL,
Description VARCHAR(120) NULL,
Credit INT NOT NULL,
Prerequisites VARCHAR(100) NULL ,
PRIMARY KEY (CourseID));

CREATE TABLE Appointment (
AppointmentID INT NOT NULL,
AppointmentDate DATE NOT NULL,
AppointmentTime TIME NOT NULL,
AppointmentType VARCHAR(24) NULL,
PRIMARY KEY (AppointmentID));

CREATE TABLE Advisors (
AdvisorID INT NOT NULL,
FirstName VARCHAR(45) NOT NULL,
LastName VARCHAR(45) NOT NULL,
Email VARCHAR(45) NULL,
OfficeLocation VARCHAR(45) NULL,
PRIMARY KEY (AdvisorID));

CREATE TABLE DepartmentCourse (
DepartmentCourseID INT NOT NULL,
DepartmentID INT NOT NULL,
CourseID INT NOT NULL,
PRIMARY KEY (DepartmentCourseID),
FOREIGN KEY (CourseID) REFERENCES Course,
FOREIGN KEY (DepartmentID) REFERENCES Department);

CREATE TABLE Professor_Department (
```

```
Professor_Department_ID INT NOT NULL,
ProfessorID INT NOT NULL,
DepartmentID INT NOT NULL,
PRIMARY KEY (Professor_Department_ID),
FOREIGN KEY (DepartmentID) REFERENCES Department,
FOREIGN KEY (ProfessorID) REFERENCES Professor);

CREATE TABLE Professor_Student (
Professor_Student_ID INT NOT NULL,
ProfessorID INT NOT NULL,
EMPLID INT NOT NULL,
PRIMARY KEY (Professor_Student_ID),
FOREIGN KEY (EMPLID) REFERENCES Student,
FOREIGN KEY (ProfessorID) REFERENCES Professor);

CREATE TABLE Professor_Course (
Professor_Course_ID INT NOT NULL,
ProfessorID INT NOT NULL,
CourseID INT NOT NULL,
PRIMARY KEY (Professor_Course_ID),
FOREIGN KEY (CourseID) REFERENCES Course,
FOREIGN KEY (ProfessorID) REFERENCES Professor);
```

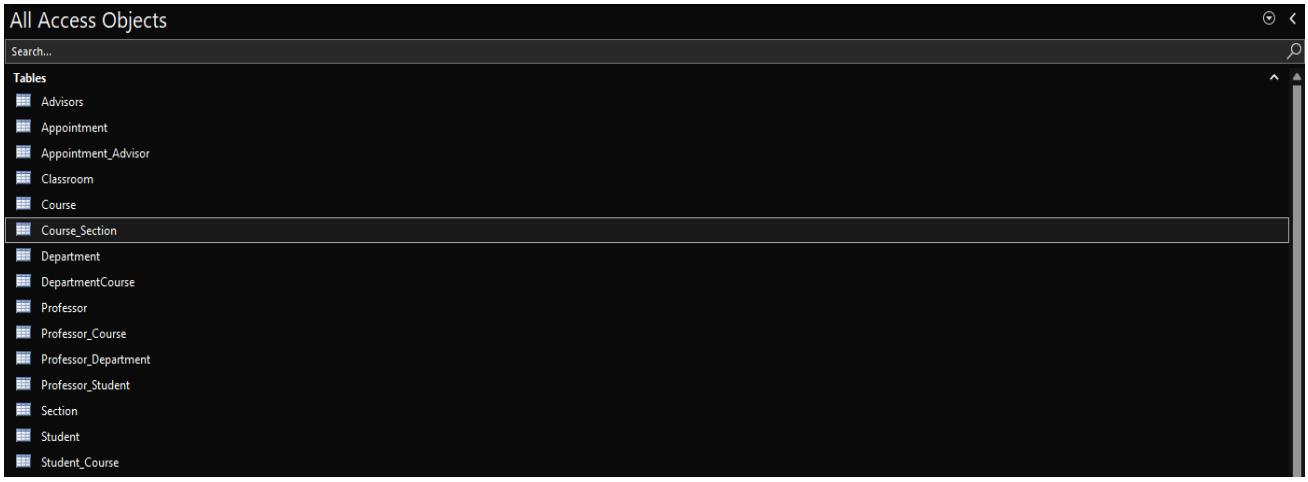
Updates on the table:

```
ALTER TABLE Student ADD COLUMN ProofofID VARCHAR(45) NOT NULL;

ALTER TABLE Professor ADD COLUMN ProofofID VARCHAR(45) NOT NULL;
```

Database Schema:

After creating the table and adding the foreign key constraints, the database schema now looks like the following:



Relationship View: Using the Relationship View under Database Tools, we can see the relationships (foreign keys) between the tables: