University DBMS Conceptual Design

1. Develop a conceptual data model reflecting the following requirements:

ASSUMPTIONS:

- **Students can return to the same school to complete multiple majors over their lifetime, no reason to give a cap to the number of majors a student can declare.**
- **No two departments will have the same name**
- **No two majors will have the same name**
- **Students need a school-provided non-duplicative ID to be told apart in the system since students with the same major and name/initials can easily exist**
- **No two events with the same name can start on the same day**
- **The same professor cannot be a chair of more than 1 department**

a. Identify the main entity types.

By reading the given case study, we can see that the University wants objects called:

- **Departments**
- **Majors**
- **Students**
- **Events**

Although it mentions departments having a chair, it is explicitly stated that we should not store specific data about the chair.

b&c. Identify the main relationship types between the entity types identified in a.

All the relationships are also told to us explicitly in the given case study. It mentions that:

Entity 1	Relationship	Entity 2	Participation	Cardinality	Multiplicity	Type of Relationship
Students	Declare	Majors	1	*	*..*	Many to Many
Majors	Define	Students	1	*	*..*	Many to Many
Majors	Reference	Departments	1	1	1..*	One to One
Departments	Clafisy	Majors	1	*	1..*	One to Many

Departments	Host	Events	0	*	*..*	Many to Many
Events	Utilize	Departments	1	*	*..*	Many to Many
Students	Attend	Events	1	*	*..*	Many to Many
Events	Entertain	Students	1	*	*..*	Many to Many

d. Identify attributes and associate them with entity or relationship types.

Most of the time, the case study will state its entity attributes immediately after the entity itself, and we have already what relationships we need. From the case study, along with everything above, we get the following entities with attributes:

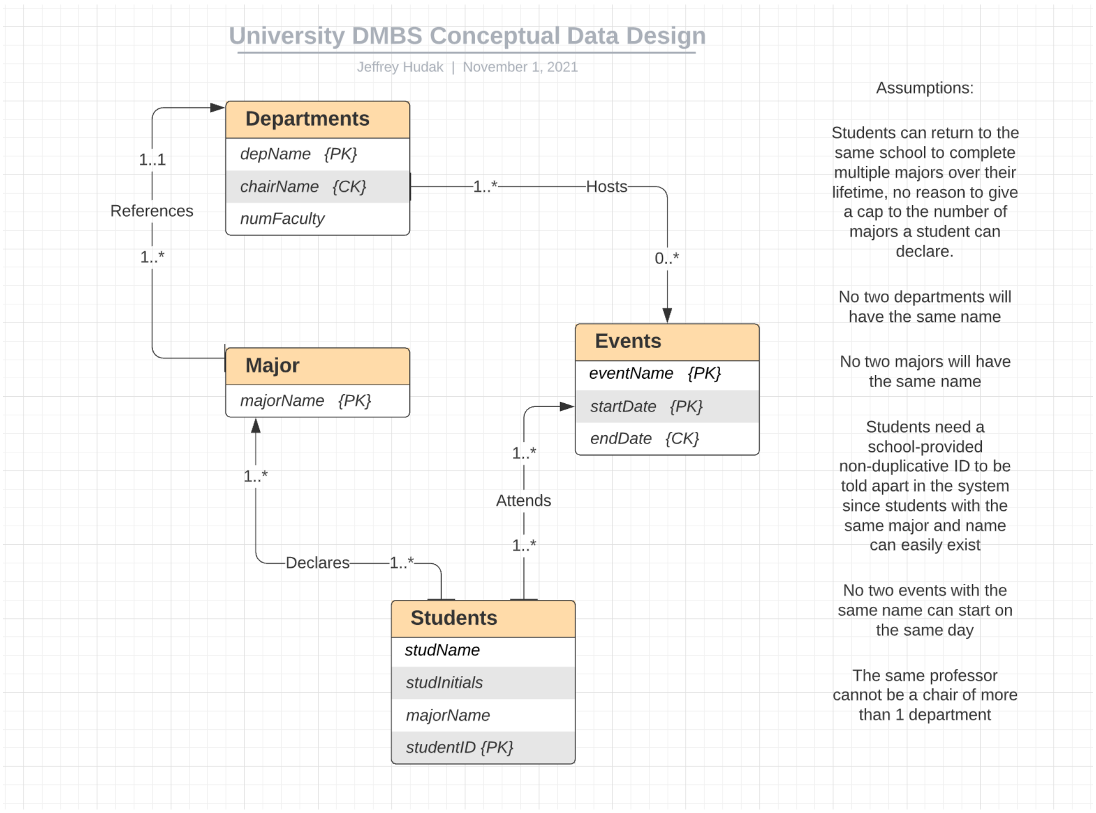
- **Departments**: have a departmentName (a unique name for each department), chairName (the name of the faculty member that is the chair of the department), and numFacultyMembers (the total number of faculty that work under the department)
- **Majors:** have a majorName (the name of the major offered)
- **Events:** have an eventName (a name for the event being held), startDate, and endDate
- **Students:** have a studName, studInitials, and a studentID (as mentioned in assumtions, a unique, school-provided identifier for each student)

e. Determine candidate and primary key attributes for each (strong) entity type

Since all of our entities are strong entities, they each have candidate and primary keys. There is no real help from the case study, so these were made by observation of the attributes of each entity along with the assumptions provided above:

- **Department:** departmentName is the primary key, as there would never be more than two departments with the same name. Also from our assumptions, since the same faculty member would not be the head of more than one department for workload reasons, the departmentChair is a candidate key, but making the departmentName the primary key just makes more logical sense.
- **Major:** The only attribute a major has is its name, so majorName is the primary key.
- **Events:** A tuple of the eventName and startDate is the primary key. The eventName and endDate is also a candidate key, as to whether we use the start or end date does not make a difference.
- **Students:** The school-provided studentID (in Assumptions) is the primary key for each student, as no other combination of the other attributes directly given in the case study allows for primary keys - students with the same names and/or initials can take the same majors.

The resulting ER Diagram can be found below:



University DMBS Conceptual Data Design
Jeffrey Hudak | November 1, 2021
Departments
depName {PK}
chairName {CK}
numFaculty
1..1
References
1..*
1..*
Hosts
0..*
Major
majorName {PK}
Events
eventName {PK}
startDate {PK}
endDate {CK}
1..*
Attends
1..*
1..*
Declares
1..*
Students
studName
studInitials
majorName
studentID {PK}
Assumptions:
Students can return to the same school to complete multiple majors over their lifetime, no reason to give a cap to the number of majors a student can declare.
No two departments will have the same name
No two majors will have the same name
Students need a school-provided non-duplicative ID to be told apart in the system since students with the same major and name can easily exist
No two events with the same name can start on the same day
The same professor cannot be a chair of more than 1 department