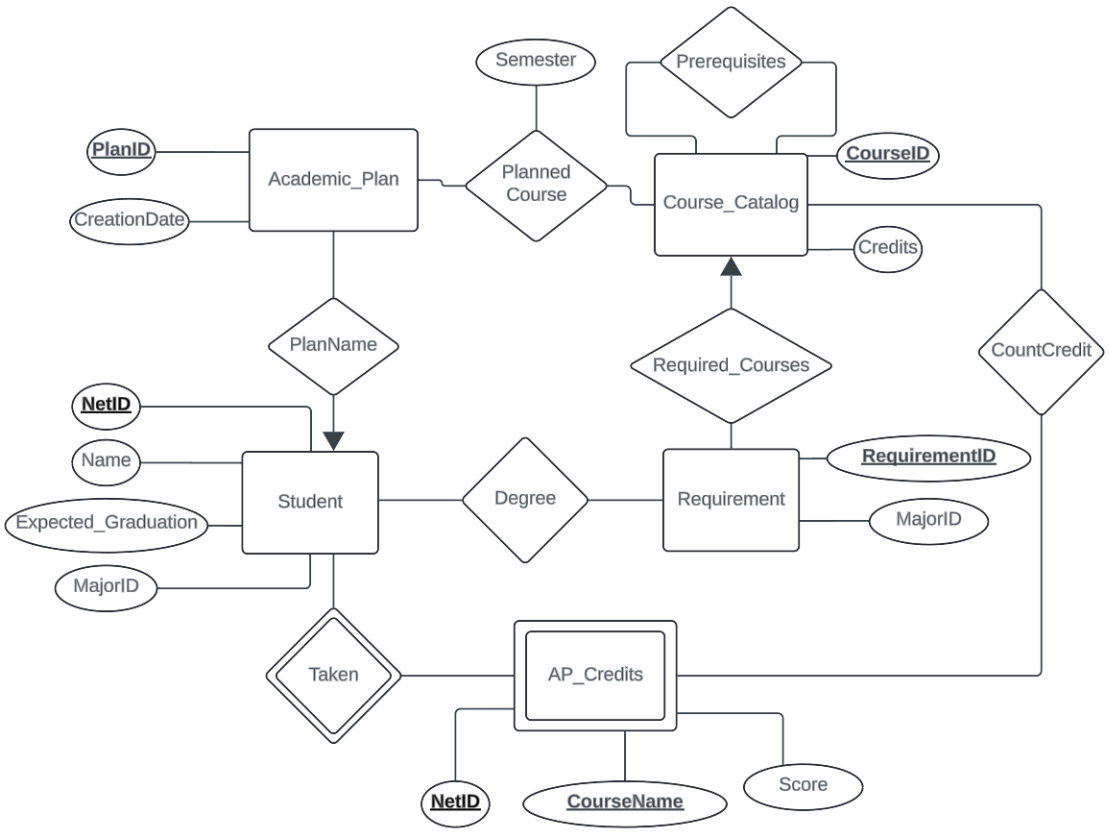
Stage 2



Entities and Attributes

1. Student:
Each student has a unique NetId(PK), defined by attributes such as `Name`, `Expected_Graduation`, and `MajorID`. Since a student has multiple attributes, and is also associated with other entities (e.g., academic plan), it is modeled as an entity rather than as an attribute of an academic plan.

2. Academic_Plan:
Represents a set of planned courses and uniquely defined as a PlanID(PK) for a student with Attributes: `CreationDate`, `PlanName` . An academic plan involves multiple attributes that describe the plan itself and is linked to multiple courses, which makes it more suitable as a distinct entity.

3. Course_Catalog:
This entity holds information about courses offered by the institution. With Attributes: `CourseID`(PK), `Credits`. The catalog contains multiple courses that are shared across multiple students and plans, hence it is more appropriate as a separate entity.

4. Requirements:
This entity represents requirements relationships among majors and courses. Attributes: `RequirementID`(PK), `MajorID`. Requirements involve one-to-many relationships between courses, which makes it necessary to have this as a distinct entity rather than an attribute and one-many relationships with students since each major can have multiple requirements.

5. AP_Credits:
This is a weak entity that represents student-taken AP courses and the respective score. Attributes: 'NetID' (PK), 'CourseName' (PK), 'Score'. The following entity is a weak entity to the Student entity, as we need both the NetID of the student and the CourseName in order to determine the score that they received in the exam.

Relationships and Cardinalities

1. Student and Academic_Plan (One-to-Many):
A student can have multiple academic plans, but each plan belongs to one student.This allows for students to have different plans, perhaps for different degrees or revisions.

2. Academic_Plan and Course_Catalog(Many-to-Many):
An academic plan can include courses, but each course belongs to one specific academic plan. It reflects the structure of courses within a single degree plan. The relationship table (PlannedCourse) has an attribute 'Semester' that indicates the semester in which the course was taken for the academic plan.

3. Course_Catalog and AP_Credits (Many-to-Many):
Courses can be fulfilled by multiple different AP credits (Ex. Calculus 1 is fulfilled assuming you get an AP score of 3 and higher on Calculus AB), and an AP Course credit can be applicable to multiple courses (Ex. Calculus 1 and 2 is fulfilled assuming

you get an AP score of 4 and higher on Calc BC exam). A separate relationship (CountCredit) is required to accurately model this many-to-many association.

4. Student and Requirements (Many-to-Many):
A student has multiple major requirements and a major requirement can apply to multiple students. Since each student has at least one major, they would have multiple major requirements to complete their degree. On the other hand, a major will have multiple students, meaning that the same requirement can apply to multiple students.

5. Requirements and Course_Catalog (Many-to-One):
A requirement corresponds to one course, but one course from the course catalog can be a requirement to multiple different majors. This allows for us to map different major requirements to their corresponding courses in the catalog without needing any additional attributes.

6. Student and AP_Credits (Many-to-Many):
A student can have multiple AP Credits. An AP Credit is given to many students. This is a weak entity because we need NetID from the student entity along with the AP Course name to uniquely identify the score for each student.

7. Course_Catalog and Course_Catalog (Many-to-Many):
Courses can have multiple prerequisites and a course can be a prerequisite to multiple courses. This self loop helps reduce redundancy in the number of entities that we are representing in the database.

Entities List:
- Student (NetID, Name, ExpectedGrad, MajorID)
- CourseCatalog(CourseID, Credits)
- Requirement (RequirementID, MajorID)
- AcademicPlan (PlanID, CreationDate)
- APCredits(NetID, CourseName, Score)

FUNCTIONAL DEPENDENCIES

1) Student
   NetID → Name, ExpectedGrad, MajorID
2) Course_Catalog

CourseID → Credits

3) Requirement
RequirementID → MajorID

4) Academic Plan
PlanID → CreationDate

5) AP_Credits
NetID, CourseName → Score

RELATIONAL

1. PlanName
   PlanID -> NetID, Name, Expected_Graduation, MajorID
2. Required_Courses
   RequirementID -> CourseID, Credits
3. Planned_Course
   PlanID, CourseID → Semester

L	M	R	NONZ
RequirementID	CourseID	Name, ExpectedGrad, MajorID	
PlanID	NetID	Semester, Credits	
CourseName		CreationDate	

CANDIDATE KEYS: RequirementID, PlanID, CourseName

MINIMAL BASIS:

NetID → Name, ExpectedGrad, MajorID
CourseID → Credits
RequirementID → MajorID, CourseID
NetID, CourseName → Score
PlanID → NetID, CreationDate

PlanID, CourseID → Semester

COMPLETED 3NF

A(NetID, Name, ExpectedGrad, MajorID);
B(CourseID, Credits);
C(PlanID, CourseID, Semester);
D(PlanID, NetID, CreationDate);
E(RequirementID, MajorID, CourseID);
F(CourseID, Credits);
G(RequirementID, PlanID, CourseName)

The following normalized dependencies adheres to 3NF for the following reasons:

- Every non prime attribute in the entity is fully dependent on the primary key
- This ensures that there are no partial dependencies
- There are also no transitive dependencies since all of the attribute is dependent on the primary key, this ensures that there are no dependencies from non-prime to another non-prime attribute

The dependencies above are 1NF because all of the attribute values are atomic (no lists or tuple values). The dependencies are also 2NF because the entity attributes are all fully dependent on the primary key. And finally the dependencies are 3NF because there are no transitive dependencies (non-prime to another non-prime attribute dependencies)

Relational Schema

Student(
  NetID: VARCHAR(20) [PK],
  Name: VARCHAR(100),
  Expected_Graduation: REAL,
  MajorID: VARCHAR(50)
)

Course_Catalog(
 CourseID: VARCHAR(20) [PK],
 Credits: INT
)

Requirement(
 RequirementID:INT [PK],
 MajorID: VARCHAR(50),
 Credits: INT [FK to Course_Catalog.Credits]
)

Academic_Plan(
 PlanID: INT [PK],
 StudentID: INT [FK to Student.StudentID],
 CreationDate: DATE,
 NetID: VARCHAR(20) [FK to Student.NetID]
)

Planned_Course(
 PlanID: INT [FK to AcademicPlan.PlanID, PK],
 CourseID: VARCHAR(10) [FK to Course_Catalog.CourseID, PK],
 Semester: VARCHAR(20)
)

Prerequisite(
 CourseID: VARCHAR(20) [FK to Course_Catalog.CourseID, PK],
 PrerequisiteID: VARCHAR(10) [FK to Course.CourseID]
)

AP_Credits (

NetID: VARCHAR(20) [FK to Student.NetID, PK],
CourseName: VARCHAR(100),
Score: INT,
CourseID: VARCHAR(20) [FK to Couse_Catalog.CourseID]
)