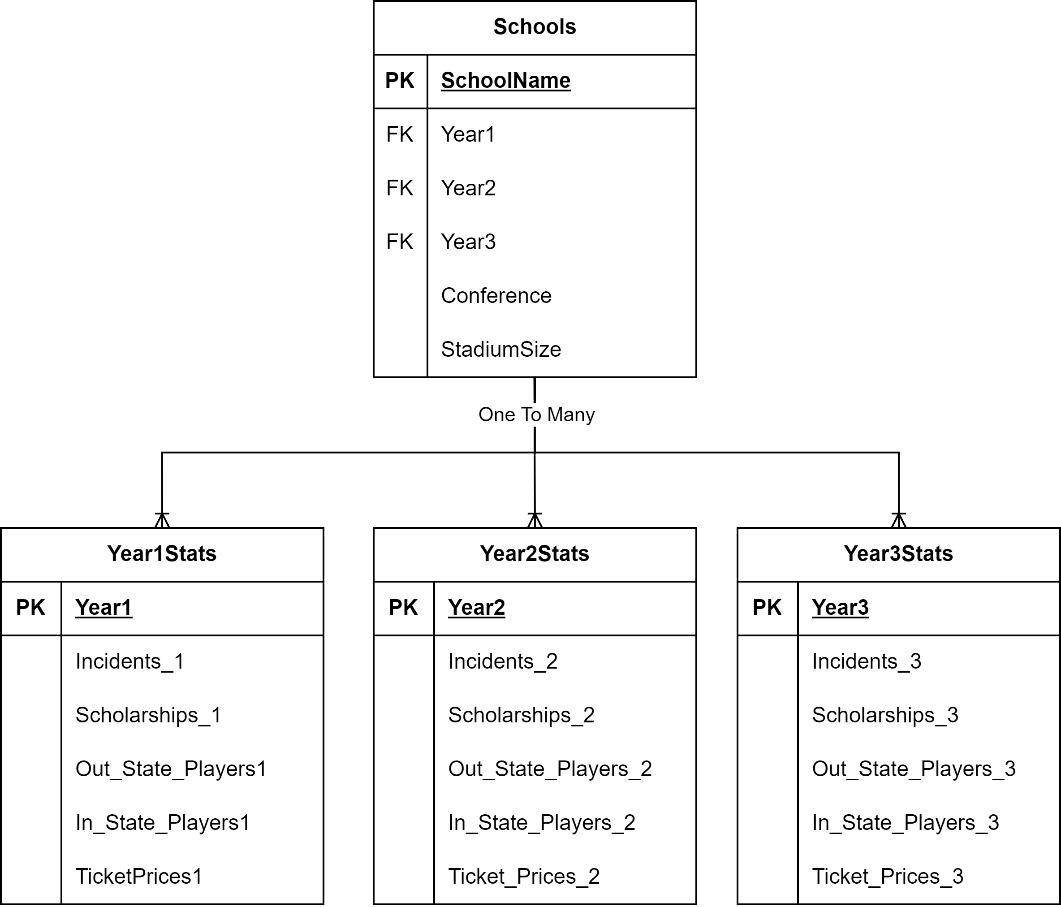
**Data Modeling Project – NCAA**

1. **Schools ER Diagram**

One Liner:

* Schools are in conferences and record stadium sizes. They also consist of a ticket price, number of incidents, a scholarship count, a number of out of state players, and a number of in state players.

ER Diagram:



Maximum Cardinality:

* Schools can belong to only 1 conference
* Schools can only record 1 stadium size
* Schools can have many out of state players
* Schools can have many in state players
* Schools can have many recruiting incidents
* Schools can have 3 listed ticket prices
* Schools can have one or more Incidents

Minimum Cardinality:

* Schools must have scholarships counted for every year
* Schools must have more than one in state player
* Schools must have more than one out of state player
* Schools must have ticket prices for every year

Tables Needed:

* For this, we would need 4 tables, one holding the data for year one, one holding the data for year 2, one holding the data for year 3, and finally, one holding the data for the school as well as all the year’s data. Each year table has a uniqueID that is stored as a primary key in the main table, Schools, which also holds its own uniqueID. This makes it easier for a Database admin to draw the calculations from each year to record averages in the reports and allows for other mathematical relationships.

SQL:

// Schools Table

CREATE TABLE Schools (

SchoolName VARCHAR(30) PRIMARY KEY,

Year1 INT,

Year2 INT,

Year3 INT,

Conference VARCHAR(25),

StadiumSize INT);

//Year1Stats Table

CREATE TABLE Year1Stats (

Year1 INT PRIMARY KEY,

Incidents\_1 DECIMAL(5,2),

Scholarships\_1 DECIMAL(2,0),

Out\_State\_Players1 DECIMAL(2,0),

In\_State\_Players1 DECIMAL(2,0),

TicketPrices1 DECIMAL(5,2),

FOREIGN KEY (Year1) REFERENCES Schools(Year1));

//Year2Stats Table

CREATE TABLE Year2Stats (

Year2 INT PRIMARY KEY,

Incidents\_2 DECIMAL(5,2),

Scholarships\_2 DECIMAL(2,0),

Out\_State\_Players2 DECIMAL(2,0),

In\_State\_Players2 DECIMAL(2,0),

Ticket\_Prices\_2 DECIMAL(5,2),

FOREIGN KEY (Year2) REFERENCES Schools(Year2));

//Year3Stats Table

CREATE TABLE Year3Stats (

Year3 INT PRIMARY KEY,

Incidents\_3 DECIMAL(5,2),

Scholarships\_3 DECIMAL(2,0),

Out\_State\_Players3 DECIMAL(2,0),

In\_State\_Players3 DECIMAL(2,0),

Ticket\_Prices\_3 DECIMAL(5,2),

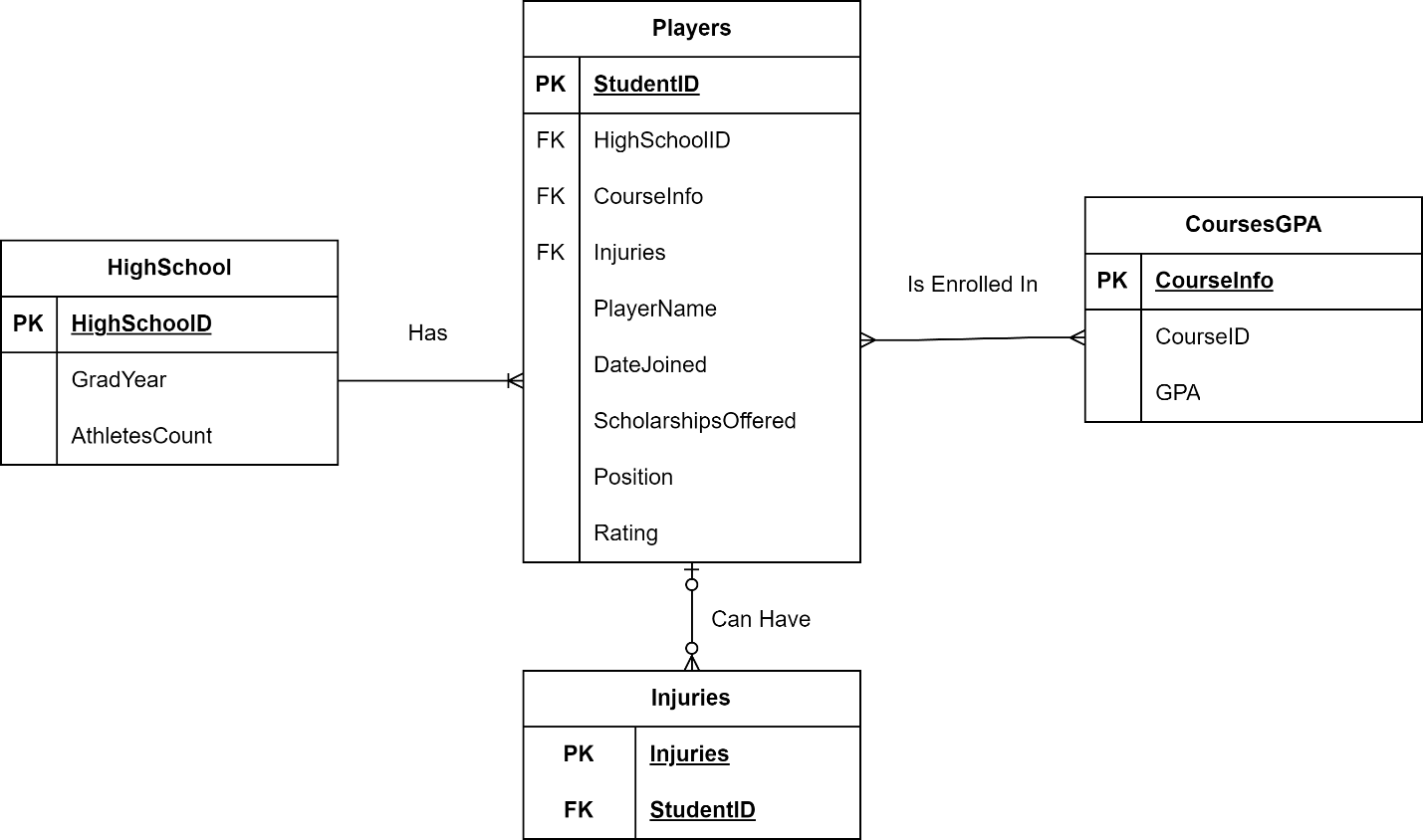
FOREIGN KEY (Year3) REFERENCES Schools(Year3));

1. **Players ER Diagram**

One Liner:

* A player has a studentID, name, date joined, High school they went to, number of scholarships offered, position, rating, and can be enrolled in 1 or more courses.

ER Diagram:



Bridge Table Diagram for Players-CoursesGPA Many-to-Many relationship:



Many to Many Relationship:

* The diagram illustrates the relationships between three entities: Players, Players\_Courses, and CoursesGPA. The Players\_Courses table is a bridge table that connects the Players and CoursesGPA tables, indicating which courses players are enrolled in. The primary keys in Players\_Courses are a composite of StudentID and CourseInfo, both of which are foreign keys referencing the Players and CoursesGPA tables respectively. This bridge table facilitates a many-to-many relationship between Players and CoursesGPA, allowing players to be enrolled in multiple courses, and each course to have multiple players.

Maximum Cardinality:

* Players can only have 1 high school ID
* Players Can only have 1 date joined
* Players can have many positions
* Players can have many scholarships offered
* Players can have many Course Information.
* Players can have maximum of 1 rating

Minimum Cardinality:

* Players must have 1 Highschool ID
* Players must be enrolled in a course
* Players must have a date joined
* Players must have 1 rating
* Players enrolled in courses must have a GPA
* Players must have a position

Tables Needed:

SQL:

//Players Table

CREATE TABLE Players (

StudentID INT PRIMARY KEY,

HighSchoolID INT,

CourseInfo VARCHAR(30),

Injuries VARCHAR(40),

PlayerName VARCHAR(30),

DateJoined DATE,

ScholarshipsOffered INT(2),

Position VARCHAR(25),

Rating INT,

FOREIGN KEY (HighSchoolID) REFERENCES HighSchool(HighSchoolID),

FOREIGN KEY (CourseInfo) REFERENCES CoursesGPA(CourseInfo));

//HighSchool Table

CREATE TABLE HighSchool (

HighSchoolID INT PRIMARY KEY,

GradYear VARCHAR(4),

AthletesCount INT(3));

//CoursesGPA table

CREATE TABLE CoursesGPA (

CourseInfo VARCHAR(30) PRIMARY KEY,

CourseID INT,

GPA VARCHAR(2));

//Injuries Table

CREATE TABLE Injuries (

Injuries VARCHAR(40),

StudentID INT,

PRIMARY KEY (Injuries, StudentID),

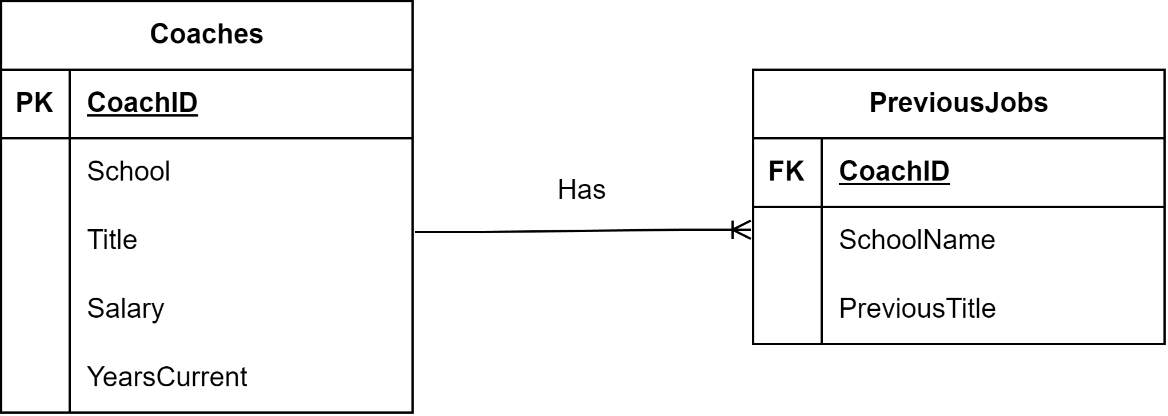
FOREIGN KEY (StudentID) REFERENCES Players(StudentID));

1. **Coaches ER Diagram**

One Liner:

* Coaches can have many previous jobs.

ER Diagram:



Relationship:

* The relationship between the Coaches and PreviousJobs tables is a one-to-many relationship where a single coach can have multiple previous jobs recorded. Each previous job entry must be associated with one coach, while a coach may or may not have any previous jobs listed.

Maximum Cardinality:

* One coach can have many previous jobs
* Coaches can only have one current school
* Coaches can only have 1 salary
* Coaches can have many years currently at their school.

Minimum Cardinality:

* Coaches must have a current school
* Coaches must have a coachID
* Coaches Must have a salary

Tables Needed:

* The Coaches table contains information about coaches, with CoachID as the primary key. The PreviousJobs table stores details of each coach's past job experiences, with CoachID as a foreign key linking each job to a specific coach. This diagram ensures that each previous job entry is directly associated with one coach, while a coach can have multiple previous job records.

SQL:

//Coaches Table

CREATE TABLE Coaches (

CoachID INT PRIMARY KEY,

CoachName VARCHAR(30),

School VARCHAR(30),

Title VARCHAR(20),

Salary DECIMAL(8, 2),

YearsCurrent INT(2));

// PreviousJobs table

CREATE TABLE PreviousJobs (

JobID INT PRIMARY KEY,

CoachID INT,

SchoolName VARCHAR(50),

PreviousTitle VARCHAR(50),

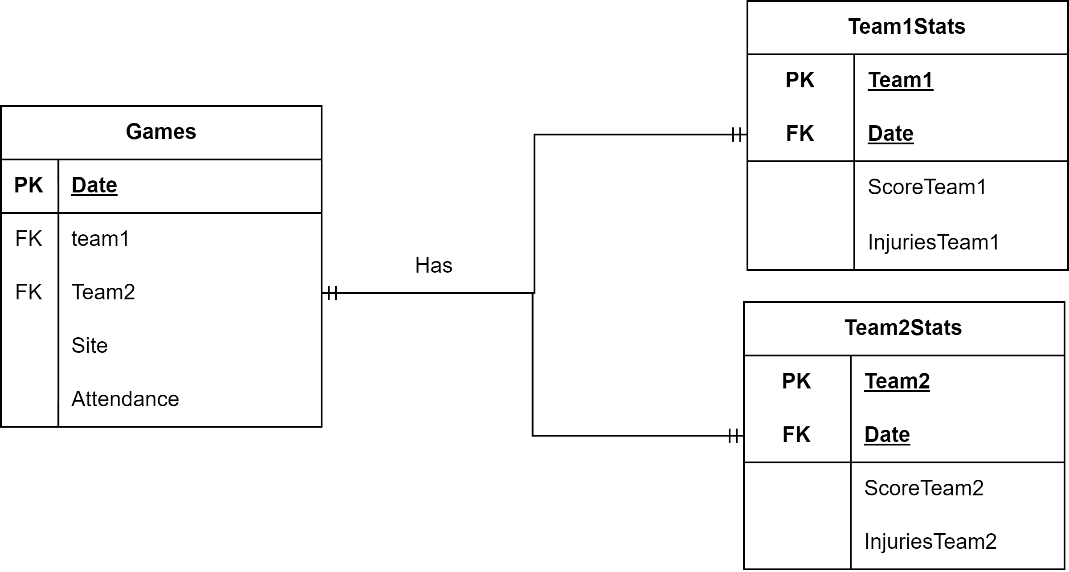
FOREIGN KEY (CoachID) REFERENCES Coaches(CoachID));

1. **Games ER Diagram**

One Liner:

* Games are played at a site with an attendance. Games have score stats and injury stats for both teams that play that game.

ER Diagram:



Relationships:

* The diagram represents the relationships between the Games, Team1Stats, and Team2Stats tables. Each game is associated with one set of statis for both Team 1 and Team 2 that are recorded in their respective Team1Stats and Team2Stats tables. The relationship ensures that for every game, the scores and injuries of both teams are tracked and linked back to the specific game.

Maximum Cardinality:

* Each team can only have 1 set of stats per game
* Each Game can only have 1 set of stats per team
* There can only be 1 site
* There can only be 1 attendance

Minimum Cardinality:

* There must be a recorded attendance
* There must be a recorded site
* There must be team 1 stats
* There must be team 2 stats

Tables Needed:

* The Games table has Date as the primary key. The table also includes team1, Team2, Site, and Attendance, where team1 and Team2 are foreign that link to the teams involved in the game. The Team1Stats table has a composite primary key consisting of Team1 and Date, with Team1 serving as both the primary and foreign key referencing the Games table. Similarly, the Team2Stats table uses Team2 and Date as its composite primary key, with Team2 also acting as a foreign key in the Games table. This model ensures that the statistics for each team are properly associated with the corresponding game.

SQL:

//Games table

CREATE TABLE Games (

Date DATE PRIMARY KEY,

Team1 VARCHAR(30),

Team2 VARCHAR(30),

Site VARCHAR(20),

Attendance INT(6));

//Team1Stats table

CREATE TABLE Team1Stats (

Team1 VARCHAR(30),

Date DATE,

ScoreTeam1 INT(3),

InjuriesTeam1 INT(3),

PRIMARY KEY (Team1, Date),

FOREIGN KEY (Date) REFERENCES Games(Date));

//Team2stats Table

CREATE TABLE Team2Stats (

Team2 VARCHAR(30),

Date DATE,

ScoreTeam2 INT(3),

InjuriesTeam2 INT(3),

PRIMARY KEY (Team2, Date),

FOREIGN KEY (Date) REFERENCES Games(Date));

Summary

The above diagrams represent each entity that must exist in order to create each required report the NCAA listed. These entities required different relationships with one another in order to exist. There are examples of one-to-one relationships, many-to-many, and some on-to-many. There are also instances of superclass and subclass relationships. This was a very unique project that helped me grasp what a real situation would be like in a real-world incident where data modeling is required, and I appreciate the challenge.