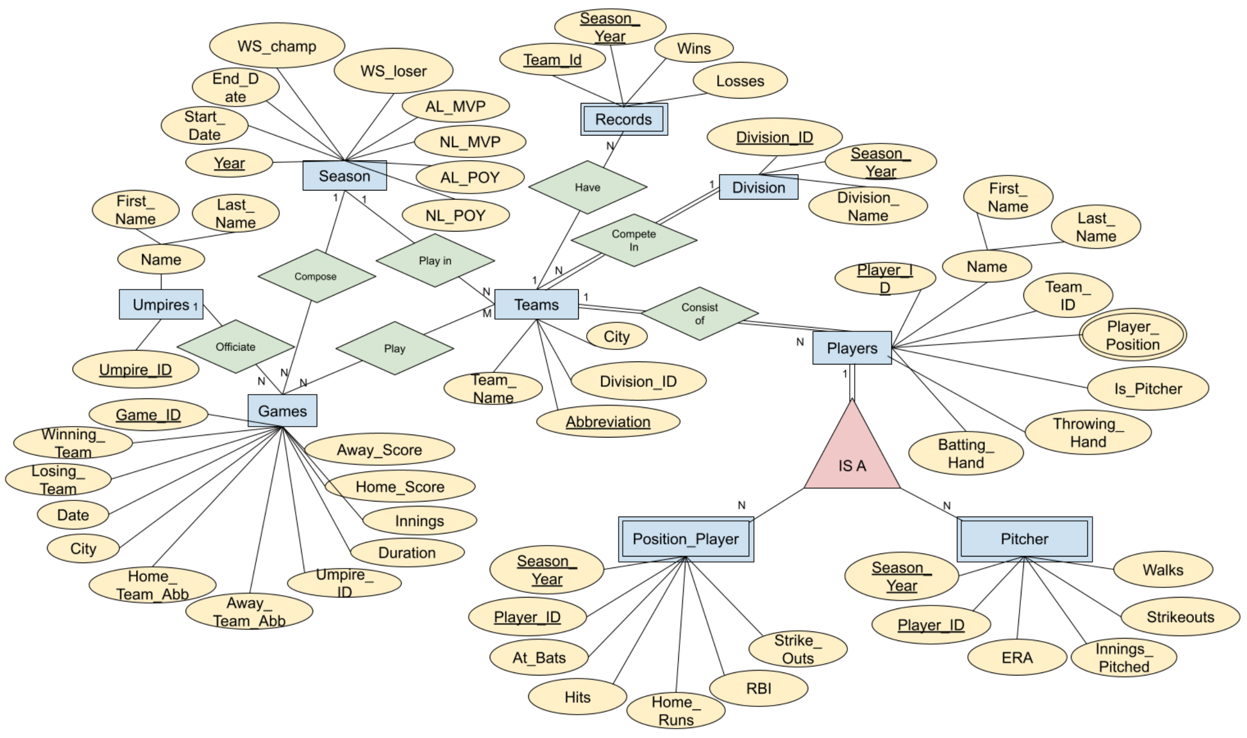
CSC 261 Project

Task A: ER Diagram



The **weak entities** in our ER diagram include Records, since it depends on a team and season for its existence, Position_Player and Pitchers, as they both depend on players for their existence. The players entity demonstrates **class hierarchy** and can be illustrated using "IS-A" notation, since a player is either a position player or a pitcher, but most fall into one of those categories and cannot be both. Our **cardinality** is shown in our diagram above. Umpires and Games have a 1:N relationship, since we are denoting there is one plate umpire per game but that umpire can officiate multiple games. For the relationship between season and games, we denoted this as 1:N since a single season is made of many games. The season and teams relationship is also 1:N since many teams play in a single season. Teams and games would be N:M since multiple (2) teams play a game, and many games are played by a single team. The division and teams relationship is 1:N since one division has many teams, but a team only competes in one division. Teams and Players is 1:N since a single team has many players, but each player plays for only one team. Then, as mentioned previously, there is class hierarchy surrounding the players table, since a player can only be a pitcher or a position player. The cardinality for both of these relationships is 1:N since many players can be a pitcher (or a position player), but a pitcher or a position player can only be a single player. Players and teams have **total participation** since every player must be associated with a team. Teams and divisions are also total since every team has an associated division (AL or NL). There is also total participation with the players table since every player must be either a position player or a pitcher.

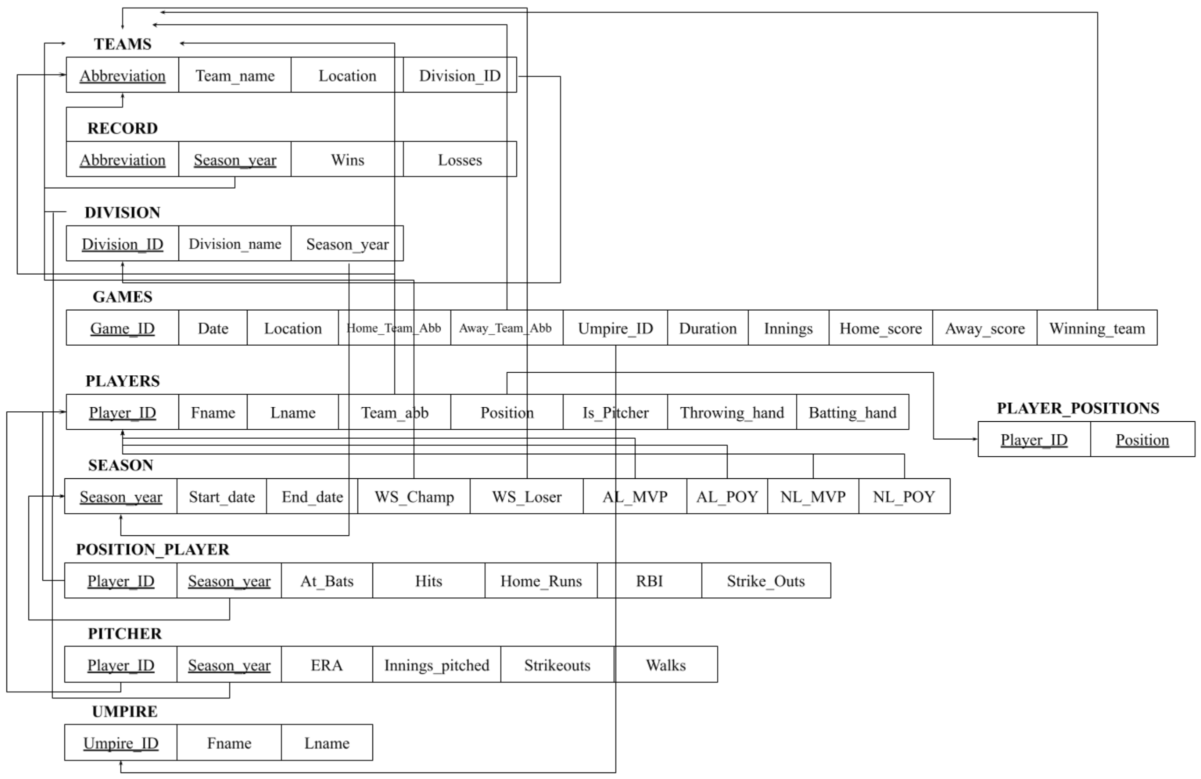
Our assumptions:

- A team must belong to one division

- A team has only one record per season
- Each game has exactly one home team and one away team
- Players must belong to a team
- A game has one umpire (home plate umpire), but each umpire can officiate multiple games
- Each player has exactly one statistical entry per season (this is a running sum)
- Each season determines one champion and one runner up

Task B: Relational Database Design Using ER-to-Relational Mapping

1. ER-to-Relational Mapping Algorithm



Step 1: Mapping of Regular Entity Types. We create the relations TEAMS, RECORD, DIVISION, GAMES, PLAYERS, POSITION_PLAYER, PITCHER, UMPIRES, and SEASON to correspond to the regular entity types TEAMS, RECORD, DIVISION, GAMES, PLAYERS, POSITION_PLAYER, PITCHER, UMPIRES, SEASON, and PLAYER_POSITIONS from our ER Diagram. The primary keys are Abbreviation, Abbreviation and Season_year, Division_ID, Game_ID, Player_ID, Player_ID and Season_year, Player_ID and Season_year, Umpire_ID, Season_year, and Player_ID and Position respectively.

Step 2: Mapping of Weak Entity Types.

Records (Weak Entity)
Depends on: Teams and Season

Primary Key: (Team_abb, Season_year)
Foreign Keys: Team_abb → Teams(Abbreviation), Season_year → Season(Season_year)
Deletion Rule: Cascade (If a team or season is deleted, its records should also be deleted.)

Pitcher Stats (Weak Entity)
Depends on: Players and Season

Primary Key: (Player_ID, Season_year)
Foreign Keys: Player_ID → Players(Player_ID), Season_year → Season(Season_year)
Deletion Rule: Cascade (If a player or season is deleted, their pitching stats should also be deleted.)

Position Player Stats (Weak Entity)
Depends on: Players and Season

Primary Key: (Player_ID, Season_year)
Foreign Keys: Player_ID → Players(Player_ID), Season_year → Season(Season_year)
Deletion Rule: Cascade (If a player or season is deleted, their batting stats should also be deleted.)

Step 3: Mapping of Binary 1:1 Relationship Types. We have no strict 1:1 relationships in our ER diagram that require additional tables. Relationships like Season to World Series Champion are mapped with foreign keys in the Season table.

Step 4: Mapping of Binary 1:N Relationship Types. For each of our 1:N relationships, the "N" side gets a foreign key referencing the "1" side:

- Teams (1) → Players (N):
  - Abbreviation as FK in Players
- Teams (1) → Record (N):
  - Abbreviation as FK in Record
- Division (1) → Teams (N):
  - Division_ID as FK in Teams
- Season (1) → Division (N):
  - Season_year as FK in Division
- Season (1) → Record (N):
  - Season_year as FK in Record
- Games (1) → Umpires (N):
  - Umpire_ID as FK in Games
- Games (1) → Teams (N) [Winning Team]:
  - Winning_team as FK in Games

Step 5: Mapping of Binary M:N Relationship Types. For M:N relationships, we created a new table with composite primary keys referencing both entities.

- Position_Player (Players ↔ Season)
  - Player_ID (FK), Season_year (FK), At_bats, Hits, Home_runs, RBI, Strike_outs
- Pitcher (Players ↔ Season)

- Player_ID (FK), Season_year (FK), Innings_pitched, ERA, Strikeouts, Walks

Step 6: Mapping of Multivalued Attributes. A player may have played multiple positions. A separate table Player_Positions stores these.

Player_Positions

Primary Key: (**Player_ID**, **Position**)
Foreign Key: **Player_ID** → Players(Player_ID)
Attributes: Position (varchar(20))

Step 7: Mapping of N-ary Relationship Types. Our database design does not incorporate N-ary Relationship Types, so this step is not in our relational mapping.

Step 8: Options for Mapping Specialization or Generalization. Our database design does not incorporate Specialization or Generalization, so this step is not in our relational mapping.

Step 9: Mapping of Union Types (Categories). Our database design does not incorporate Union Types, so this step is not in our relational mapping.

Relation Name	**ER Diagram Components**
Teams	E(Teams) + R(compete_in → Division)
Record	E(Record) + R(have → Teams)
Division	E(Division) + R(compete_in → Teams)
Games	E(Games) + R(play → Teams) + R(officiate → Umpires) + R(compose → Season)
Players	E(Players) + R(consist_of → Teams)
Position_Player	E(Position_Player) + R(is_a → Players)
Pitcher	E(Pitcher) + R(is_a → Players)
Umpires	E(Umpires) + R(officiate → Games)
Season	E(Season) + R(play_in → Teams) + R(compose → Games)
Player_Positions	E(Player_Positions)

2. Schema

Teams

Primary Key: **Abbreviation**
Foreign Key: **Division_ID** → REFERENCES Division(Division_ID)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
<u>Abbreviation</u>	varchar(3)	PRIMARY KEY	Identify every team with a unique 3 letter abbreviation. All abbreviations used in the MLB now.
Team_name	varchar(20)	NOT NULL	The team name, ex. Yankees
Location	varchar(20)	NOT NULL	The team location, ex. New York
Division_ID	varchar(2)	FOREIGN KEY	Stores the division of each team. This is a Foreign Key referenced from the DIVISION table.

Actions taken on Foreign Key - **Division_ID** - when the corresponding tuple for the referred table is deleted:

set null

If a division is removed, teams remain in the database but without a division.

Record

Primary Key: (**Abbreviation**, **Season_year**)
Foreign Key: **Abbreviation** → REFERENCES Teams(Abbreviation), **Season_year** → REFERENCES Season(Season_year)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
<u>Abbreviation</u>	varchar(3)	PRIMARY KEY, FOREIGN KEY	The Team in the relationship. This is a Foreign Key referenced from the TEAMS table.
<u>Season_year</u>	int(4)	PRIMARY KEY, FOREIGN KEY	The Year in the relationship. This is a Foreign Key referenced from the SEASON table.
Wins	int	NOT NULL	Count of the number of wins in a single season.
Losses	int	NOT NULL	Count of the number of losses in a single season.

Actions taken on Foreign Key - **Abbreviation** - when the corresponding tuple for the referred table is deleted:

cascade

If a team is deleted, its historical record is also removed.

Actions taken on Foreign Key - **Season_year -** when the corresponding tuple for the referred table is deleted:

cascade

If a season is deleted, all records for that season are removed.

Division

Primary Key: **Division_ID**
Foreign Key: **Season_year** → REFERENCES Season(Season_year)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
Division_ID	varchar(2)	PRIMARY KEY	Identify each division with a unique 2 letter abbreviation. All abbreviations used in the MLB now.
Division_name	varchar(20)	NOT NULL	Either National League or American League.
Season_year	int(4)	FOREIGN KEY	Stores the season each year. This is a Foreign Key referenced from the SEASON table.

Actions taken on Foreign Key - **Season_year -** when the corresponding tuple for the referred table is deleted:

cascade

If a season is removed, its division structure is no longer valid.

Games

Primary Key: **Game_ID**
Foreign Keys: **Home_Team_Abb** → REFERENCES Teams(Abbreviation)
Away_Team_Abb → REFERENCES Teams(Abbreviation)
Umpire_ID → REFERENCES Umpires(Umpire_ID)
Winning_team → REFERENCES Teams(Abbreviation)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
Game_ID	int	PRIMARY KEY, Auto increment	Identify every game with a unique number. The first tuple in the relation will have a value of 1.

Date	datetime	NOT NULL	Date and time the game was played.
Location	varchar(20)	NOT NULL	Where the game was held.
Home_Team_Abb	varchar(3)	FOREIGN KEY	The home team playing in the game. This is a Foreign Key referenced from the TEAMS table.
Away_Team_Abb	varchar(3)	FOREIGN KEY	The away team playing in the game. This is a Foreign Key referenced from the TEAMS table.
Umpire_ID	int	FOREIGN KEY	The umpire officiating the game. This is a Foreign Key referenced from the UMPIRES table.
Duration	decimal(5,2)	NOT NULL	Total duration of the game in hours.
Innings	int	NOT NULL	Total innings played.
Home_score	int	NOT NULL	Runs scored by the home team.
Away_score	int	NOT NULL	Runs scored by the away team.
Winning_team	varchar(3)	FOREIGN KEY	The team that won the game. This is a Foreign Key referenced from the TEAMS table.

Actions taken on Foreign Key - **Home_Team_Abb** - when the corresponding tuple for the referred table is deleted:

set null

If a home team is removed, the game remains, but the home team is set to NULL.

Actions taken on Foreign Key - **Away_Team_Abb** - when the corresponding tuple for the referred table is deleted:

set null

If an away team is removed, the game remains, but the away team is set to NULL.

Actions taken on Foreign Key - **Winning_Team** - when the corresponding tuple for the referred table is deleted:

set null

If a winning team is removed, the game remains, but the winner is set to NULL.

Actions taken on Foreign Key - **Umpire_ID** - when the corresponding tuple for the referred table is deleted:

set null

If an umpire is removed, the game remains, but the umpire is set to NULL.

Players

Primary Key: **Player_ID**
Foreign Key: **Team_abb** → REFERENCES Teams(Abbreviation)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
<u>Player_ID</u>	int	PRIMARY KEY, Auto increment	Identify every player with a unique number. The first tuple in the relation will have a value of 1.
Fname	varchar(30)	NOT NULL	Player's first name.
Lname	varchar(30)	NOT NULL	Player's last name.
Team_abb	varchar(3)	FOREIGN KEY	The team the player is associated with. This is a Foreign Key referenced from the TEAMS table.
Position	varchar(20)	NOT NULL	The player's primary position.
Is_Pitcher	boolean	NOT NULL	Indicates if the player is a pitcher.
Throwing_hand	varchar(10)	NULLABLE	Player's throwing hand. R or L.
Batting_hand	varchar(10)	NULLABLE	Player's batting hand. R, L, or S.

Actions taken on Foreign Key - **Team_Abb** - when the corresponding tuple for the referred table is deleted:

set null

If a team is removed, players remain in the database, but without a team.

Position_Player

Primary Key: (**Player_ID**, **Season_year**)
Foreign Keys: **Player_ID** → REFERENCES Players(Player_ID), **Season_year** → REFERENCES Season(Season_year)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
<u>Player_ID</u>	int	PRIMARY KEY, FOREIGN KEY	The player the stats belong to. This is a Foreign Key referenced from the PLAYER table.

Season_year	int(4)	PRIMARY KEY, FOREIGN KEY	The season this stat is for. This is a Foreign Key referenced from the SEASON table.
At_Bats	int	NOT NULL	Total at-bats.
Hits	int	NOT NULL	Total hits.
Home_Runs	int	NOT NULL	Total home runs.
RBI	int	NOT NULL	Total runs batted in.
Strike_Outs	int	NOT NULL	Total strikeouts.

Actions taken on Foreign Key - **Player_ID -** when the corresponding tuple for the referred table is deleted:

cascade

If a player is removed, their batting stats are also removed.

Actions taken on Foreign Key - **Season_year -** when the corresponding tuple for the referred table is deleted:

cascade

If a season is removed, all batting stats for that season are removed.

Pitcher

Primary Key: (**Player_ID**, **Season_year**)
Foreign Keys: **Player_ID** → REFERENCES Players(Player_ID), **Season_year** → REFERENCES Season(Season_year)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
Player_ID	int	PRIMARY KEY, FOREIGN KEY	The player the stats belong to. This is a Foreign Key referenced from the PLAYER table.
Season_year	int(4)	PRIMARY KEY, FOREIGN KEY	The season this stat is for. This is a Foreign Key referenced from the SEASON table.
Innings_pitched	decimal(5,2)	NOT NULL	Total innings pitched.
ERA	decimal(4,2)	NOT NULL	Earned Run Average.

Strikeouts	int	NOT NULL	Total strikeouts.
Walks	int	NOT NULL	Total walks allowed.

Actions taken on Foreign Key - **Player_ID -** when the corresponding tuple for the referred table is deleted:

cascade

If a player is removed, their pitching stats are also removed.

Actions taken on Foreign Key - **Season_year -** when the corresponding tuple for the referred table is deleted:

cascade

If a season is removed, all pitching stats for that season are removed.

Umpire

Primary Key: **Umpire_ID**
Foreign Keys: none
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
Umpire_ID	int	PRIMARY KEY, Auto increment	Identify every umpire with a unique number. The first tuple in the relation will have a value of 1.
Fname	varchar(30)	NOT NULL	Umpire's first name.
Lname	varchar(30)	NOT NULL	Umpire's last name.

Season

Primary Key: **Season_year**
Foreign Keys: **WS_Champion** → REFERENCES Teams(Abbreviation)
WS_Loser → REFERENCES Teams(Abbreviation)
AL_MVP → REFERENCES Players(Player_ID)
AL_POY → REFERENCES Players(Player_ID)
NL_MVP → REFERENCES Players(Player_ID)
NL_POY → REFERENCES Players(Player_ID)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
Season_year	int	PRIMARY KEY	The year of the season. Identify every year with a year number.

Start_date	datetime	NOT NULL	When the season started.
End_date	datetime	NOT NULL	When the season ended.
WS_Champ	varchar(3)	FOREIGN KEY	Team that won the World Series. This is a Foreign Key referenced from the TEAMS table.
WS_Loser	varchar(3)	FOREIGN KEY	Team that lost the World Series. This is a Foreign Key referenced from the TEAMS table.
AL_MVP	int	FOREIGN KEY	Player that won the American League MVP. This is a Foreign Key referenced from the PLAYERS table.
AL_POY	int	FOREIGN KEY	Player that won the American League POY. This is a Foreign Key referenced from the PLAYERS table.
NL_MVP	int	FOREIGN KEY	Player that won the National League MVP. This is a Foreign Key referenced from the PLAYERS table.
NL_POY	int	FOREIGN KEY	Player that won the National League POY. This is a Foreign Key referenced from the PLAYERS table.

Actions taken on Foreign Key - **WS_Champ** - when the corresponding tuple for the referred table is deleted:

set null

If a World Series champion team is removed, the season record remains, but WS_Champ is set to NULL.

Actions taken on Foreign Key - **WS_Loser** - when the corresponding tuple for the referred table is deleted:

set null

If a World Series losing team is removed, WS_Loser is set to NULL.

Actions taken on Foreign Key - **AL_MVP** - when the corresponding tuple for the referred table is deleted:

set null

If an AL MVP player is removed, the record remains but the AL_MVP is set to NULL.

Actions taken on Foreign Key - **AL_POY** - when the corresponding tuple for the referred table is deleted:

set null

If an AL POY player is removed, the record remains but the AL_POY is set to NULL.

Actions taken on Foreign Key - **NL_MVP** - when the corresponding tuple for the referred table is deleted:

set null

If an NL MVP player is removed, the record remains but the NL_MVP is set to NULL.

Actions taken on Foreign Key - **NL_POY** - when the corresponding tuple for the referred table is deleted:

set null

If an NL POY player is removed, the record remains but the NL_POY is set to NULL.

Player_Positions

Primary Key: **(Player_ID, Position)**
Foreign Keys: **Player_ID** → Players(Player_ID)
Attributes:

Attribute	**Data Type**	**Default**	**Purpose & Description**
Player_ID	int	PRIMARY KEY, FOREIGN KEY	Foreign key referencing Players(Player_ID), uniquely identifying a player.
Position	varchar(20)	PRIMARY KEY, FOREIGN KEY	The position a player plays (e.g., "Shortstop", "Catcher"). Part of the primary key.

Actions taken on Foreign Key - **Player_ID** - when the corresponding tuple for the referred table is deleted:

cascade

If a player is removed, their position records in Player_Positions are also removed.

Actions taken on Foreign Key - **Position** - when the corresponding tuple for the referred table is deleted:

restrict

A position cannot be deleted if it is still assigned to any player in Player_Positions.

Group Work & Contribution

We collaborated on sharing documents online throughout the entire milestone. We both agree that we contributed equally to the assignment, with each of us completing roughly half of the work.