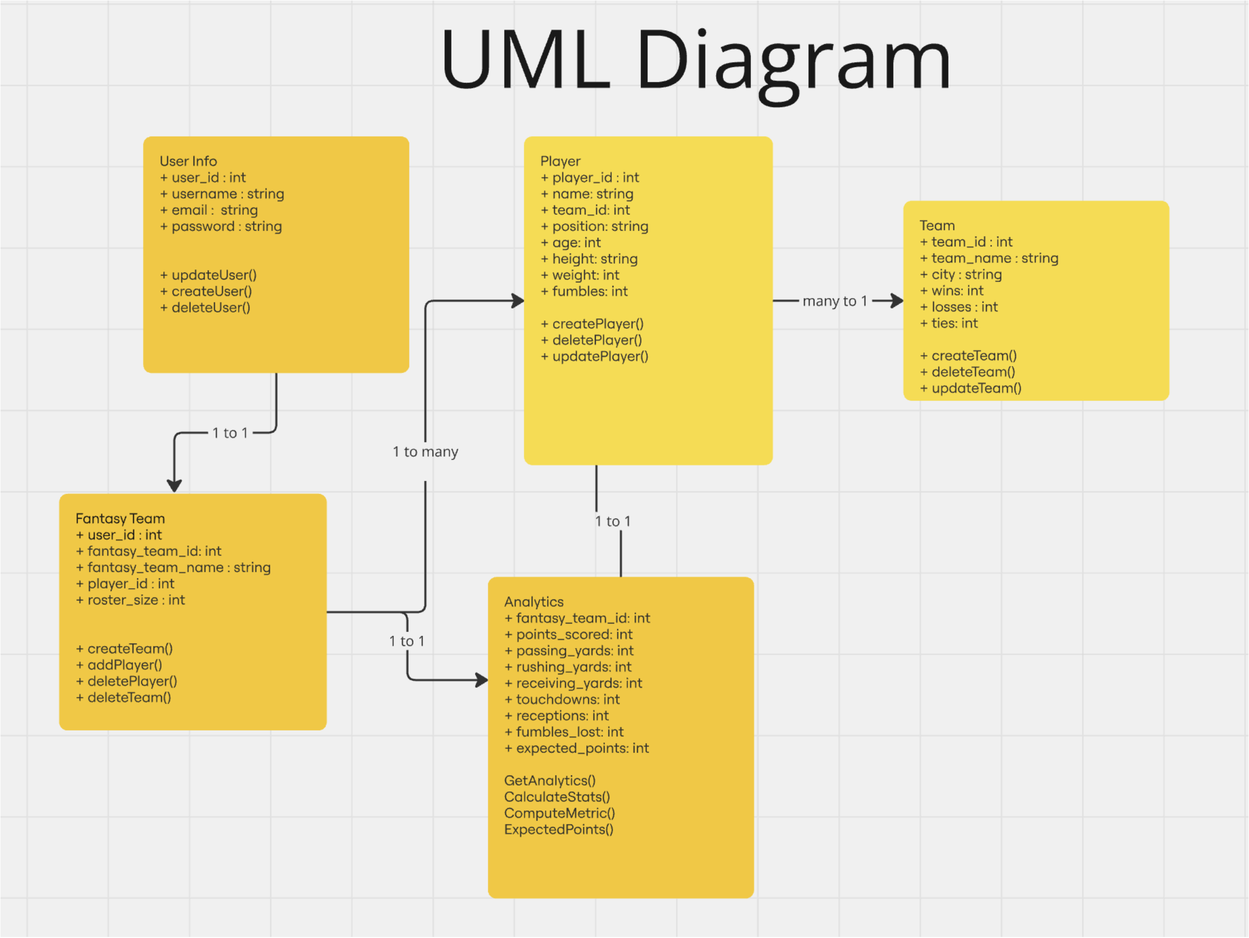
Stage 2 Database Design

1 Draw ER/EML Diagram



2 Assumptions

Our project consists of 5 tables: User, FantasyTeam, Analytics, Player, and Team. There are 1 to 1 relationships from User to Fantasy Team, Fantasy Team to Analytics, and Player to Analytics. We made these assumptions because each user should only have one fantasy team. The Analytics table serves almost as a “black-box” that can take care of all the calculations for both fantasy team and player. This is to simplify our backend and this way, analytics can be flexible to work with both tables. There is also a 1 to many relationship from Fantasy Team to Player and a many to one relationship from Player to Team. Each fantasy team should be able to have multiple players, hence the 1 to many. Many players can also be on the same team so that’s why there is a many to one relationship for those tables. For the cardinality of the fantasy team we have designed it so that you can create multiple fantasy teams to analyze, the cardinality of the players, the total amount of players in nfl is 1696, so we expect this table to be about that size, there are 32 nfl teams which will be inside teams.

Since our app is designed to be a fantasy analytics helper tool, there doesn't need to be leagues with teams or schedules. All we need is functionality for us to compute stats for players and fantasy rosters so we don't need more overhead. The reason we separated analytics from players was so that analytics could also compute statistics for an entire fantasy roster. The player table is more used to show basic information about the players, similar to what you would see them introduced as at the NFL combine or on TV during games.

4 Normalizing database

We begin finding all the functional dependencies and then creating a mapping with a letter to simplify the calculation

Player.team_id = A
Team.team_id = D
FantasyTeam.user_id = B
Userinfo.user_id = E
Analytics.team_id = C
FantasyTeam.fantasy_team_id = F

R (A, B, C, D, E, F)
FD = {A ->D; B -> E; C -> F}

After we have obtained the functional dependencies we calculate the 3nf and found our database to already be in 3nf form
Therefore, our schema is 3nf.

5 Converting to conceptual database design

```
UserInfo(
        user_id: INT [PK],
        username: VARCHAR(100),
        email: VARCHAR(100),
        password: VARCHAR(100)
)
Team(
        team_id: INT [PK],
        team_name: VARCHAR(100),
        city: VARCHAR(100),
        wins: INT,
        losses: INT,
        ties: INT
)
Player(
        player_id: INT [PK],
        name: VARCHAR(100),
        team_id: INT [FK to Team.team_id],
        position: VARCHAR(50), age: INT,
        height: DECIMAL(5,2),
        weight: DECIMAL(5,2),

)
FantasyTeam(
        fantasy_team_id: INT [PK],
        user_id: INT [FK to UserInfo.user_id],
        fantasy_team_name: VARCHAR(100),
        roster_size: INT
)
Analytics(
        team_id: INT [PK, FK to FantasyTeam.fantasy_team_id],
        points_scored: INT,
        passing_yards: INT,
        rushing_yards: INT,
        receiving_yards: INT,
        touchdowns: INT,
        receptions: INT,
        fumbles_lost: INT,
```

expected_points: INT
)

Fix for suggestions from previous stage from canvas:

“The project is interesting. I think you may want to give more details about the creative component. Currently, the description for the creative component is not enough and I cannot understand your plan to implement this component, like how you will analyze the player performance and what data and technical methods you will use to support the analysis. You also need to point out the detailed technically challenging points to implement this component.”

More details for creative component:

For the creative component we want to provide performance metrics and ranking players. We wanted to use linear regression to create models to show the user how their player is expected to perform in future games based on their past performance, and use nearest neighbor technique learned in cs 441 to find players who have similar attributes which a user could use to make informed decisions in their draft. The way we will quantify performance is based on their prior performance using linear regression and computing totals on their stats from the Player Table and populating it inside the Analytics table