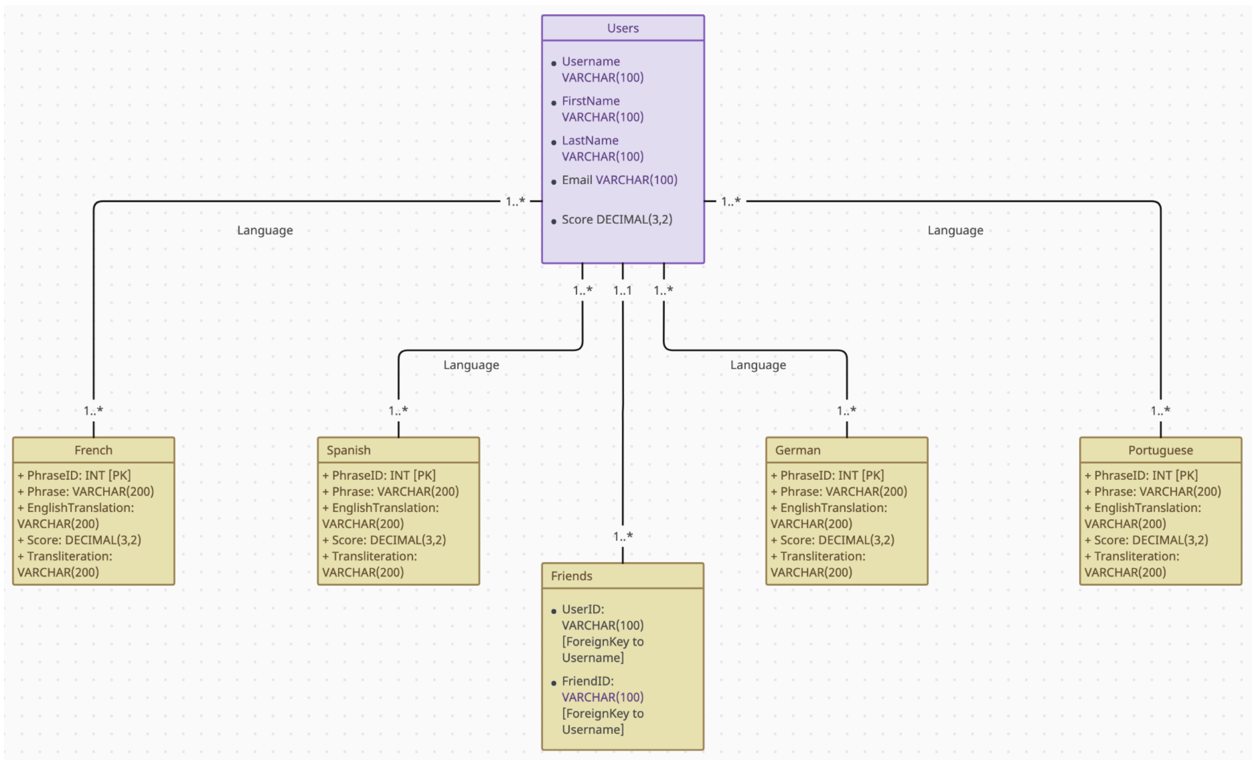
Stage 2



Assumptions

Users:

We've modeled this as an entity since each user has multiple attributes. Also, the score component is compiled from all the other entities (except friends), hence modeling users as a single entity rather than an attribute in each language will make it simpler to do computations and keep track of user information.

Friends:

We originally modeled this as a self-relation from users to users as a 1-to-many relationship, but we realized that this may cause issues in keeping a list of friends as a list and we found it easier to understand friends as a separate entity that takes userID and friendID as foreign keys from users. This will help us conveniently have a table with the first column being a user and the other being their friend. We can also easily retrieve a leaderboard from this as well.

French, Spanish, German, and Portuguese:
We've decided to model each language as a separate entity since each language has its own set of phrases and translations. The website we gather our information from does not have the same list of common phrases for each language and hence we are required to make a separate entity for each language.

Relationships:

- Users to languages is a many-to-many relationship since many users can use many phrases from each language.

- Users to friends is one-to-many as we only need to use a user as a foreign key once for many friends.

Relational Schema

Users(Username: VARCHAR(100) [PK], FirstName: VARCHAR[100], LastName: VARCHAR[100], Email: VARCHAR(100), Score: INT)

French(PhraseID: INT [PK], Phrase VARCHAR(200), EnglishTranslation VARCHAR(200), Score DECIMAL(5,2), Transliteration VARCHAR(200))

Spanish(PhraseID: INT [PK], Phrase VARCHAR(200), EnglishTranslation VARCHAR(200), Score DECIMAL(5,2), Transliteration VARCHAR(200))

German(PhraseID: INT [PK], Phrase VARCHAR(200), EnglishTranslation VARCHAR(200), Score DECIMAL(5,2), Transliteration VARCHAR(200))

Portuguese(PhraseID: INT [PK], Phrase VARCHAR(200), EnglishTranslation VARCHAR(200), Score DECIMAL(5,2), Transliteration VARCHAR(200))

Friends(UserID: VARCHAR(100) PK,FK to Username, FriendsID: VARCHAR(100) PK,FK to Username)

LanguageF(Username: VARCHAR(100) [PK, FK to Users.Username], PhraseID: INT [PK, FK to French.PhraseID])
LanguageS(Username: VARCHAR(100) [PK, FK to Users.Username], PhraseID: INT [PK, FK to Spanish.PhraseID])

LanguageG(Username: VARCHAR(100) [PK, FK to Users.Username], PhraseID: INT [PK, FK to German.PhraseID])

LanguageP(Username: VARCHAR(100) [PK, FK to Users.Username], PhraseID: INT [PK, FK to Portuguese.PhraseID])

Normalization:
Functional dependencies:
Users Relation

Username -> FirstName, LastName, Email, Score

French Relation

French.PhraseID -> French.Phrase, French.EnglishTransation, French.Score, French.Transliteration

Spanish Relation

Spanish.PhraseID -> Spanish.Phrase, Spanish.EnglishTransation, Spanish.Score, Spanish.Transliteration

German Relation

German.PhraseID -> German.Phrase, German.EnglishTransation, German.Score, German.Transliteration

Portuguese Relation

Portuguese.PhraseID -> Portuguese.Phrase, Portuguese.EnglishTransation, Portuguese.Score, Portuguese.Transliteration

Possibly in BCNF/3NF already because the LHS is a (super)key in all the FDs.

Addressing Concerns from Previous Checkpoints:

We were advised that our initial AI-focused creative component was unsuitable for a database class. Instead, we plan to implement a scoring system for sentences based on their length, providing users with more challenging phrases if their score exceeds a certain threshold. This approach ensures a smoother learning curve for users. Additionally, we are developing web scrapers to extract the necessary data.