Table of Contents

[Overview 1](#__RefHeading___Toc541_3202133802)

[Minimum Viable Product 1](#__RefHeading___Toc543_3202133802)

[Long-term Enhancements 1](#__RefHeading___Toc545_3202133802)

[Workflow – Weekly Game Processing 1](#__RefHeading___Toc547_3202133802)

[Workflow – Rankings 2](#__RefHeading___Toc549_3202133802)

[Classes 2](#__RefHeading___Toc551_3202133802)

[Team 2](#__RefHeading___Toc553_3202133802)

[Record 2](#__RefHeading___Toc555_3202133802)

[Modules 3](#__RefHeading___Toc557_3202133802)

[GameResults 3](#__RefHeading___Toc559_3202133802)

[Ranking 3](#__RefHeading___Toc561_3202133802)

[DB Manager? 3](#__RefHeading___Toc563_3202133802)

[Database 3](#__RefHeading___Toc565_3202133802)

# Overview

Create a program to assist with the process of ranking CFB teams each week. Short-term the program will collate the data and perform an initial pass to assist me with determing weekly rankings. Long-term, the program may be a standalone ranking system.

# Minimum Viable Product

* Obtain data necessary for basic ranking calculation (weekly game data)
* Store the data for each team
* Output the ranking results in a .csv file

# Long-term Enhancements

* Web-server to host a front-end for managing data collection and distribution
* Increase complexity of ranking algorithm so that it can provide a standalone ranking
* Include outside polls for comparison/use

## Workflow – Weekly Game Processing

1. Obtain game data for all games in a given week
2. For each game, determine the winning and losing team, point differential and location
3. Update win/loss record and point differential for both teams in DB
   1. How can I get the win/loss streak if I store the data that way? What if I had a streak column that started 1 to -1 and onwards?
   2. It’ll be tricky for me to capture trends in a team’s performance if I don’t store the weekly game data in some fashion.
4. Add opponent played in most recent week to DB

## Workflow – Rankings

For each team of known teams...

1. Get list of opponents

2. Get win-loss record of each opponent (aggregate home/awa/neutral) and combine into a total

3. Get win-loss of team (aggregate home/away/neutral information)

4. Get point diff of team

5. Print out information into a .csv

## Classes

### Team

Represents each team that we will include to evaluate in our rankings.

* Attributes
  + ID, Name, Record
* Methods
  + get\_team\_id
  + get\_win\_loss
  + update\_record
  + get\_opponents
  + get\_opponent\_win\_loss

### Record

Record is an aggregated total of the types of wins/losses, opponents and point differential for a given season. Each team will have one record per season.

* Attributes
  + team\_name
  + Home/Away/Neutral Wins/Losses
  + season
  + opponents
  + point\_diff
  + game\_location
  + win\_loss\_type (reg season, bowl, playoff, etc)
* Methods
  + add\_opponent
  + update\_win\_loss

## Modules

### GameResults

Fetches weekly results for each game played that week and stores information in DB.

* Attributes
  + season
  + week
* Methods
  + get\_weekly\_game\_results

#### Ranking

* Get list of teams
* Do ranking

#### DB Manager?

* Open connection
* Method that accepts sql statement, executes it and returns result
* Method that accepts sql statement, executes it (redundant?)

## Database

teams - Generic, mostly static team information

|  |  |  |  |
| --- | --- | --- | --- |
| team\_id | name | conference | stadium\_capacity |
| 100 | Michigan | Big10 | 100000 |
| 101 | Rutgers | Big10 | 50000 |

records - Aggregate of each team’s record for a given season

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| record\_id | team\_id (team\_id\_fk) | home\_wins | away\_losses | point\_diff | season |
| 1 | 100 | 3 | 0 | 30 | 2020 |
| 2 | 101 | 5 | 1 | 10 | 2020 |

games – record specific to a team for a given game

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| game\_id | home\_team (team\_id\_fk) | away\_team (team\_id\_fk) | game\_date | overtime |
| 1 | 100 | 102 | 09/30/22 | false |
| 2 | 101 | 103 | 09/30/22 | true |

season\_opponents – Collection of teams that other teams have played

|  |  |  |
| --- | --- | --- |
| season\_opponents\_pk | record\_id (record\_id\_fk) | opponent\_pk (team\_id\_fk) |
| 50 | 1 | 200 |
| 51 | 2 | 201 |

ranking - Output of my weekly rankings for each team

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ranking\_id | team\_id (team\_id\_fk) | season | Week | MyRanking | Wins | Losses | Opponent win % |
| 60 | 100 | 2020 | 1 | 5 | 1 | 0 | 100% |
| 70 | 100 | 2020 | 5 | 20 | 1 | 1 | 75% |

***MAY NOT BE NEEDED*** Games - Details for each game of the week

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| GameID | Week | Date | HomeTeam | AwayTeam | HomeTeam Points | AwayTeam Points | NeutralSite | Overtime |
| 456 | 1 | 9/14/2020 | 001 | 002 | 24 | 7 | false | true |
| 457 | 1 | 9/14/2020 | 003 | 004 | 13 | 30 | false | false |

# How To Setup a New Environment

1. Ensure postgres is installed and running
2. Create a cfb\_rankings database and run the scripts included in the project:
   1. teams.sql, records.sql, season\_opponents.sql, games.sql
3. Change psycopg2 connection string as needed for user/password/port
4. Run the import\_team\_data.py individually to fill out all of the team information
5. Should be good to go!