GrahamLiangSun

Professor Durant

CS3200 - Database Design

20 June 2019

*NFL Fantasy Football Database Final Report*

README for NFL Project

Important for the recreation of our project is to first download and store all of our project files into the same directory of your computer since some files are reliant on reading the data contained within other files. After this has been done, opening and running the self-contained database dump will initially create our database with all of the tables and their keys, constraints, triggers, procedures, functions, and details set up. Additionally, the values for some of our tables will be added to the database as well, as we hard coded some of the values for some tables, such as the nfl teams table. If for some reason there is no data in the players table, there is a program called populate\_db.py that will parse the players.json file and insert the records into the database. Since we are including a database dump, this will not be needed, but we have included it in our submission as it was used to set up our databases for the project. For the sake of actually recreating exactly what we did for the project, this is what one would have to do to recreate the backend code of our project.

For recreating the front end of our project, once again a Python-supported interface is needed, and the file nflproject.py should be used to run our front end code. This program runs on the command line, and various options that the user can select will be printed to the terminal which will accept user input. For recreating the rest of our project, and actually utilizing the front end of our application as a user, please refer to the step-by-step section of our report.

The username and password used to access the database assumes that you have a user “root” on your database server with the password of “root”. This can be changed accordingly in the code if need be.

**PYTHON/TECHNICAL DEPENDENCIES**

In order to run both the database program and the main program, there are a few dependencies that need to be installed. Our program runs on Python 3. The json package is used in the database program to parse through the json file, and datetime is used in both programs for manipulating date objects. Most importantly however, you will need to install mysql-connector. This can be installed from the link below, or by using the following command:

python3 -m pip install mysql-connector-python

<https://dev.mysql.com/downloads/connector/python/>

Step-by-step Description of User Operations and User Activity Flow

1. User runs the nflproject.py program
2. User is prompted to select one of the following options: 'exit', 'create user', 'delete user', 'create fantasy team', 'delete fantasy team', 'view fantasy teams', 'view players', 'view players on fantasy team', 'add player to fantasy team', 'remove player from fantasy team', or 'change fantasy team owner'
3. User selects ‘view players’. Program will print out a valid list of player positions. User enters the position abbreviation and program prints out a list of players matching that position.
4. User selects “create user” to create a user.
5. User selects “create fantasy team” to create a fantasy team. User is prompted to enter a team name and select the owner of the team from one of the created users.
6. User selects “add player to fantasy team”. User is prompted to select the team and to enter a player’s NFL ID.
7. The remaining options can be selected to remove a player from a fantasy team, delete a fantasy team, and deleting a user. Deleting a user will delete all fantasy teams owned by that user, and deleting a fantasy team will remove all the player-fantasy team relationships in the database for that team. The user can also select a fantasy team and designate a new owner to it from the list of users.
8. User selects “exit” and the program will terminate.

Planned Uses And Users of our NFL Project

The section above which illustrates a step-by-step flow of the activity of users of our command-line program highlights the available uses of our project. Users can use our application code to view a database of NFL players with their teams, and then can use our application code to create fantasy football teams from these players, along with creating users/owners of these fantasy teams. They can then use our program to add players that they want, using players by position, and can then view these teams, view the data of the players in our database, view the data of the players in the fantasy teams, and even delete these teams or players they’ve added to the teams, as well as updating which user owns each team. Based on these uses, it is clear that our application program is geared towards the inexperienced or unprofessional database user, and is geared towards users who want to create a simple collection of fantasy football teams from the players of the NFL. Users will use our program mainly to create and manage fantasy football teams that they want to create, and because our program keeps track of various users and owners of these teams, the users of our application will also include those interested in starting whole leagues of fantasy football teams. Users will be able to create fantasy football teams and maintain leagues among themselves with our program in a more simple manner than writing it down or maintaining a league manually, which is a method that some people use today to keep track of their fantasy leagues, and can be rather complex. However, because our program makes the storing of these teams simple with a simple command line program, this whole process of keeping track of leagues of fantasy football teams will be easier for our application users. Additionally, users who want to have fantasy football teams and leagues that are free have the benefit of using our program instead of having to pay for a program to store and track their teams. Lastly, our program supports fantasy football users who are fed up with some other fantasy football application’s rules. For example, some other fantasy football applications from our experience require that no one have the same player on their fantasy teams. However, our program is designed so that this rule does not apply, as well as some other rules such as how many players are stored on the team. Therefore, it is up to the user and their league that they are perhaps running to declare and set rules, and they can design their teams using our application to fit those rules. Thus, overall, our application is designed for all kinds of non-professional fantasy football enthusiasts, and can be used in a variety of ways by our users according to how they want to set up their fantasy football teams or leagues.

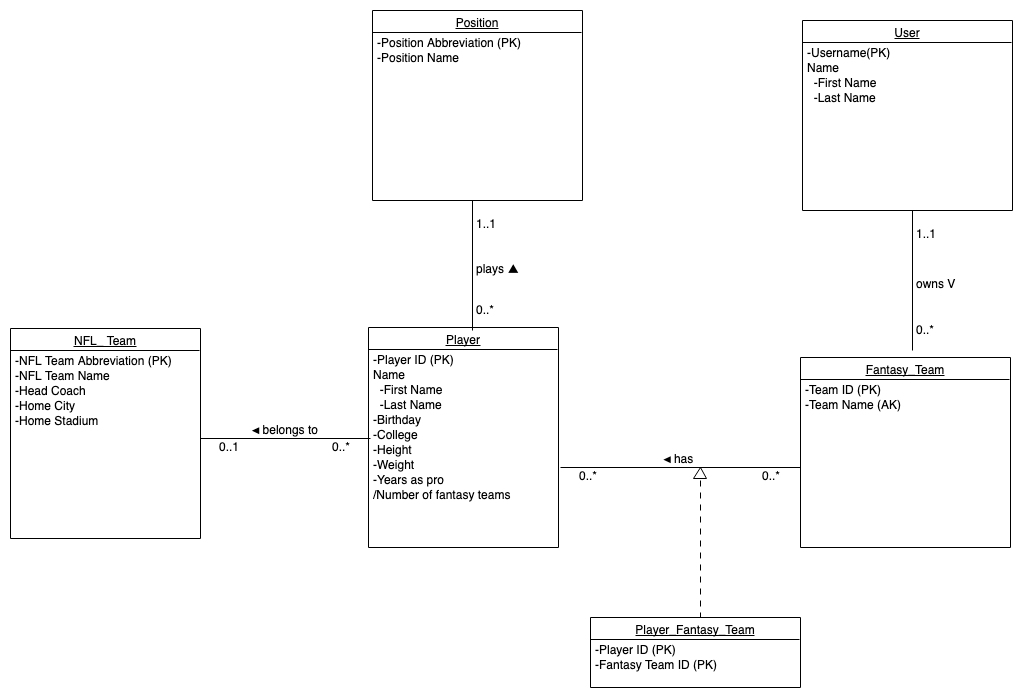
Technical Specifications

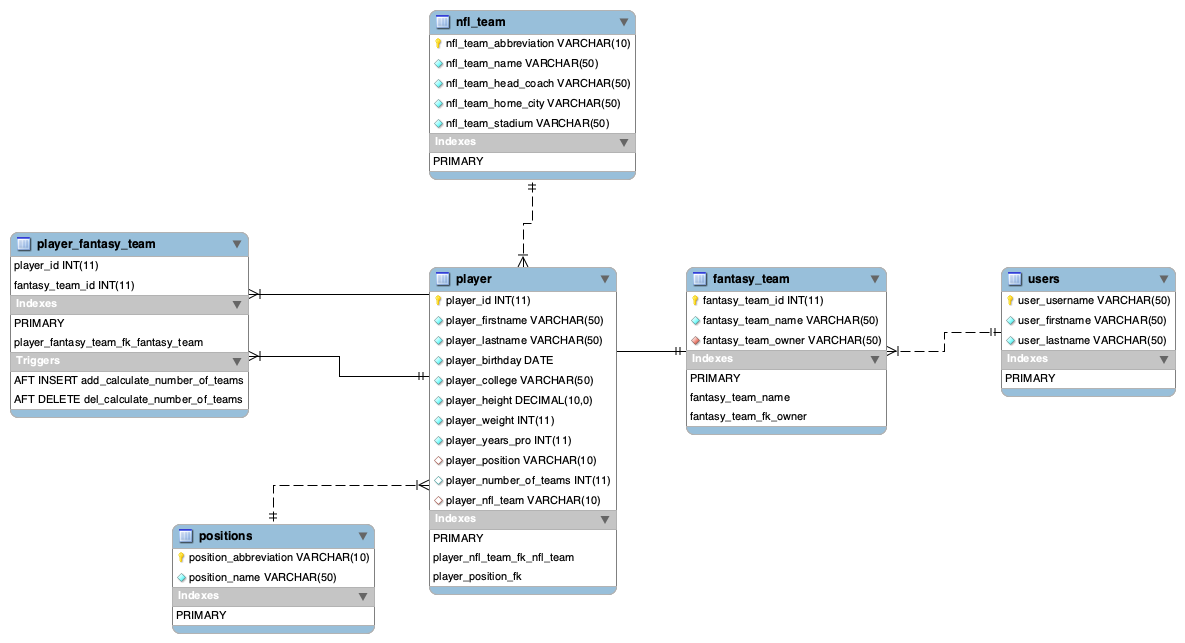
The plan for the NFL Project was to include a front end representation of our application using html. However, a front end that utilizes a simple command line interface ended up being the front end instead. Our host language of Python was used in order to create the user application, and it was connected to the SQL database for our project. The data is derived from a large-volume JSON of the active players in the NFL that was retrieved. This JSON file contains various statistics such as the player IDs of the players as recorded on the NFL’s website, a player’s position, their name, and other metrics such as height, weight, and birthday. This information will be stored as records in the players table in the database.

As far as actual hardware and machine specifications, there are none for this project besides having a computer which can run our needed programs, code files, and applications. Additionally, a working command line/terminal is needed for running the front end command-line functionality of our project.

UML and EER Diagram for NFL Project

The UML Diagram in Figure 3.1 and EER diagram in Figure 3.2 for the NFL Project database both make it easier to visualize the relationships that exist between the data and the tables within the database.

**



Flow of Project Overview

The NFL Project database provides several CRUD operations for all four of the CRUD operations that the user can perform through the front end of our database which lives in the command line. These make it easy for users to specify what type of information they need from the database without having to go through the entire large volume of information we have stored on the NFL. Some CRUD operations in our database are exemplified by several different operations available to a user within our database, however examples of at least one are provided below:

1. Creating a Fantasy Team - the user is allowed to create a fantasy team by adding players by their player id and assigning these fantasy teams with an owner name and team name.
2. Read players (i.e. by position) - this operation allows the user to return a list of players they wish to read by specifying, for example, which position that the players play as.
3. Updating fantasy team - this operation allows the user to modify any existing fantasy teams by changing the owner.
4. Deleting aspects of a fantasy team - a user can modify an existing fantasy team that exists not only by the specified actions mentioned above but also by deleting players off of a fantasy team.

Additional CRUD operations that we have include the creating and deleting of users, the deletion of fantasy teams, and viewing the current fantasy teams. Our tables are set up so that if a user is deleted, all fantasy teams owned by that user and the associated player-team relationships are deleted.

Lessons Learned

The NFL Project was able to give us invaluable experience into creating an entire database essentially from scratch (save for the values located within that are derived from a retrieved JSON file). Keying in on specifics, some of the group that was relatively new to the world of programming was able to learn the basics of a new programming language Python and the syntax that is involved with writing Python programs. Additionally, the integration of SQL with a high-level programming language such as Python was essential to the development of each one of us as computer scientists. Some lessons that were learned from the creation of this database involved the realization of better design methods that could’ve been user and implemented. For example, in our code we could use a lot more abstraction for actions such as getting the user input into the team name. Additionally, more formatting of our code would help make our code cleaner and better. The addition of GUI would make the front end more attractive and make the CRUD operations slightly more obvious and easier to use. This was the original plan as well, which included creating a web app that would integrate all of our operations and the database itself for the user to use. Finally, the project could some more search functions, including being able to filter from multiple specified fields.

One particularly noteworthy issue that we ran into early into the project was the method we used to gather player data. We originally wanted to use the NFL’s API to do a live search of the data. However, we were not able to query the API using its endpoints, as it was giving permission issues. From what we saw on Google, it seems that many users also had errors using the API endpoints, and one needs to email the NFL to explicitly request access to the API. Therefore, we were unable to reliably use the API to gather our data. In a real world application for NFL data, it is important to have a live source of data as players can often change teams, become injured, or otherwise have statistics change.

Future Work

The NFL Project data, as mentioned before, would greatly benefit from having a live source such as from a web API. Player statistics are constantly updating, teams can change ownerships and even cities, and players often are traded or injured.

Another worthwhile function to add to the program itself is a backup function. As the current program stands, the user selects a function such as deleting a user, and is then presented to select the user to delete. There currently is no way to change your mind unless you kill the application entirely. Therefore, one worthwhile addition is a “back up” feature that allows the user to cancel an operation. A GUI would also be beneficial to this app, as it would allow for a much clearer graphical representation of the data. Especially for visualizing users, teams, and players. We originally had wanted to build a web application, but as we did not have much experience, we decided to do the project on the command line for simplicity’s sake. More search functions would also be beneficial, and having a GUI may also help with adding better result filtering. For example, there could be checkboxes to view only players of certain positions, or multiple combinations, as well as filtering by name, team, etc.