Members: **Neil Isaac Hulbert, Ai Nguyen,**
Dung Thai, Jiarui Xiong

Abstract

Our group created a database that stores information about American college football teams, Pac-12. This was designed for football fans, professional recruiters, and anyone who is interested in the Pac-12 Conference. Users can view their favorite teams' past and current performance, their favorite players' statistics, the matches between teams, the coaches, and can use machine learning to see the most likely outcome of future matches. We also generated some web pages to enhance our database. This document will go in depth about our motivation, goals, planning, and provide details on how we implemented this project.

Introduction

There has been an increasing demand for data analysis from the Pac-12 themselves, from the fans participating in fantasy leagues, from professional scouts, and from the NFL looking to narrow down their top prospects. As there are no sports system sites that do data analysis, we decided on creating a sports system for Pac-12 that does the basic functionalities and expanding it using data analysis. As fans ourselves, we felt that we have a good understanding of what functionalities our database should include and ensure it meets the needs of users. Our database contains basic functionalities like allowing fans to view teams' past and current performance, players' statistics, the matches between teams, and the coaches. Then expand our database to be able to allow users to select two teams and see the winning percentage of future matches.

Objectives and Scope

Pac-base was built in response to requests from fans who support a team from the Pac-12 Conference. Our database can accomplish many of the capabilities that other sports systems can do including allowing fans to be able to view any information about their favorite team from performances to the matches, the coach of the team, and track players' statistics. In order to provide the information about their preferred team in the Pac-12 quickly and expertly, we structured our database efficiently to provide the most positive experience. In addition, we use machine learning to provide prediction of future matches.

Pac-base only covers all football teams from the Pac-12 Conference instead of from the NCAA league. Our database stores information about matches, season records, coaches, and players in Pac-12 teams after 2011. On our website, users can select their favorite team and view the information, and use the prediction functionality for the next match.

Relation to Other Work

Our idea came from looking at other existing sports systems including ESPN and Pac-12. In the ESPN system, they have millions of data on the American college football teams including information about matches, seasonal records, their players, and the players' statistics. Not only that, ESPN also has millions of data on other sports and their data goes as far as decades ago. As fans ourselves, sometimes it is difficult for us to search for information about the teams that we are interested in. On the other hand, we also noticed that there is an increasing demand for data analysis from the team themselves to improve their performance, from the fans participating in fantasy leagues, from professional scouts, and from the NFL looking to narrow down their top prospects. Since ESPN mainly focuses on presenting the data directly, and they do not analyze or process their data any further, we decided to create Pac-Base to accomplish the fans' demand. It is particularly difficult for a non-technical user to use machine learning on this data. We conducted a more professional and comprehensive analysis for the Pac-12 division.

Main Body of Work

All we had in the beginning of the project was an idea that we wanted to collect data and use that data to predict the result of the future match. As fans ourselves, we knew the basic ideas of the data entities we needed to generate. From there, we developed our initial tables and loaded our tables with sample data sets. We also added and modified our data as we learned how to construct a good database design. After that we were able to manipulate our data entries using SQL queries to give us information that is useful for our statistics. One of our initial queries was to get all the matches that Huskies team played, display the score of the Huskies team and the score of the opponents team for each match. This query was modified later for our website, where it would be able to get all the matches of the selected team, display the score of the selected team and the score of the opponents team for each match.

After our initial SQL database, we started to revise for the final integration. Since we modified our database as we learned more about constructing a good database design, we only had minor changes from phase II to phase III. The first thing we did was to sketch our database using an ER diagram to display the major entities within the system scope and the relationships among these entities. As we continued to revise our project, we modified our ER diagram many times until it best represented our database. Specifically, the relationships of the MATCH relation were difficult to pin down at first. After the ER diagram, we began to normalize our database using the Boyce-Codd Normalization Format, where we identified every determinant within all attributes and determined whether or not they were candidate keys. Luckily, we did not have to revise our schema or to create an extra table.

As we finalized the database implementation, we began the front end development. Since we don't have much experience with designing the front end and connecting with the database, we choose to work with Visual Studios using C# web form and SQL database for the main site. It carries out most of the work for us where we only need to drag the UI design around to get the arrangement that we prefer and connect it with the database while still having a grasp of web development.

For the machine learning component, we used a separate cloud provider: the Google App Engine connected to a Google Cloud SQL database. Since the prediction takes much more computation and database querying than any other function of the site, using a different instance for prediction prevented any non-prediction request from being processed too slowly. On this provider we used JSP for the web page, which is associated with Java Servlets. Each of these servlets run the Java LibSVM support vector machine library for training and prediction. Since model training doesn't take too long, we decided to allow the user to choose the past player statistics on which the prediction will be based, and then retrain the model using those statistics on each request. This is fast enough and this way we don't have to store the trained models in the database. These statistics can include rushing yards, passing yards, touchdowns, tackle yards for loss, sacks, and interceptions made. Each of the statistics can be taken as a team total, or as a list of the top 1-10 players on the team. The statistics chosen therefore determine the feature vector for the SVM, with each statistic scaled to fit within a range of [0,1]. For the parameters used for the LibSVM model, we chose a linear kernel and the C-SVM classification method, with a C value of 100, an epsilon value of 0.001 for the stopping criterion, the shrinking heuristic enabled, and a kernel cache size of 1GB.

We ensured that our interface reflects all of the database functionalities. We tried to use every table in the database so that our work would be reflected in the functionality of the interface. To navigate the web pages, we decided to put all the team on the front page instead just having the logo and the name.

The starting page with twelves teams is the first thing that the users see when the website opens. The users can select their favorite team or any team that we are interested in seeing. After the users select the team, it will bring users to the next page and they will first see the logo, team name, and university name. It will also display the matches between the selected team and all of their opponents, our website will display the opponents' name, the selected team's score and the opponents' score, and the date that they played. At the same time, the users can also see the coach information including their name, how many wins or losses in their coach career. Users can select the season record of the selected team, which will display the wins and losses in the Pac-12 Conference, the wins and losses in the season. On each of the team webpage, there is a menu bar with "Home", "Team", and "Rooster". If users click on "Home", it will bring them back to the front page. if users click on "Team", they select a different team that they want to see information on. If users click on "Roster", they will see all the offensive and defensive players on the selected team. There is also a fun stats clickable link that shows the players with the most sacks, coach for each team, and coach overall win percentage. On our "Fun Stats", we also included a link that leads users to prediction functionality for the next match page, the users can select two teams and it will display the probable result.

Architecture

We decided to make seven tables including TEAM, SEASONREC, COACH, MATCH, ROSTER, OFFPLAYER, DEFPLAYER based on BNCF while maintaining the characteristics of what we wanted in our database.

Since our database is about the American college football team, we decided to have a TEAM table that contains the basic information including an unique id, the name, and the university name.

To be able to allow users to view their favorite teams' past and current performance, we included a SEASONREC table, it contains team id, the year, wins and losses in the Pac-12 Conference, wins and losses in the season.

To grant users the ability see all the matches that their favorite teams have played. We included a MATCH table that includes an unique id, the id of the home team, the id of the away time, the score of the home team, the score of the away team, the date of the match.

As for the COACH table, it contains team id, their name, their start year, wins and losses in their career. Using this information will allow the school or professional recruiters to determine if the coach is suitable for the team.

We included information about the players, which is the ROSTER table that includes their name and an unique id, their position, how many games played, and what team they are from. There are multiple reasons why we included the ROSTER, OFFPLAYER, DEFPLAYER tables in our database. The first reason is to allow the professional recruiters to access the players' records including how many games played, if they are offensive or defensive players, and other useful information to help evaluate the player better. This will make their recruitment effortless. The second reason is to allow the coaches to effortlessly access the players' records, to understand their strengths and weaknesses, which will help the coaches train their players better. It will also assist the coaches in deciding who should be on the floor based on the team that they are playing against. At the same time, coaches can see their opponents' strengths and weaknesses, which allows coaches to quickly come up with strategies, which might improve their performance.

Putting all of those tables together allows us to achieve our primary purpose of this project, which is to allow the fans to choose any two teams, and we will provide the percentage of winning for the next match.

Conclusions

Overall, this project was well-organized to help us expand our knowledge on constructing, modeling, and normalizing our database using BCNF, as well as implementing the front end. At the end of the project, we felt that we have a strong and solid foundation about databases. We feel prepared to conduct industry projects. It is unbelievable about what we have learned and accomplished in a short period of time. Our project could not have gone smoothly without the contribution of everyone in the team and our professor for conveying the knowledge needed about databases to help us succeed not only for this project, but throughout our careers.

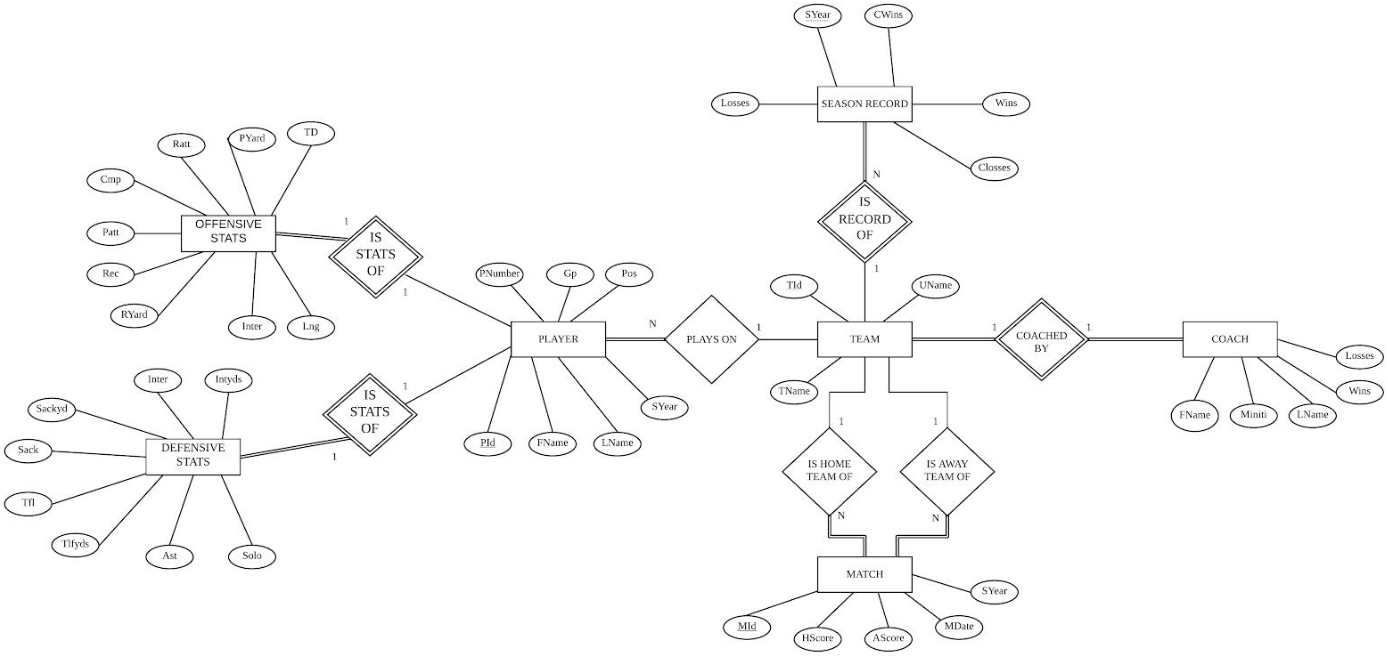
Thoughts about future work

The current state of our database and the website is at a good starting point for us to continue developing. A feature that we would like to implement is to be able to predict the future division winner, this would require an enormous amount of data. In addition, we can further extend our web application into mobile applications, which allows fans to have access to their favorite team's latest updates quickly.

References:

“12.” *Pac*, stats.pac-12.com/sports/fball/2019-20/teams.
“Football Statistics.” *Pac*, pac-12.com/content/football-statistics.
Utah Athletics. “Football Standings.” *Pac*, 1 Jan. 2020, pac-12.com/football/standings.
“NCAAF Teams.” *ESPN*, ESPN Internet Ventures, www.espn.com/college-football/teams.
“LibSVM.” GitHub, www.https://github.com/cjlin1/libsvm**.**

Appendix A: ER DIAGRAM



Appendix B: Normalization Proof to BCNF

I. Identifying Functional Dependencies:

1. TEAM = {Tid, Tname, Uname}
   a. Tid -> {Tname, Uname}
2. COACH = {Tid, Fname, Minit, Lname, Wins, Losses, SYear}
   a. Tid -> {Fname, Minit, Lname, Wins, Losses, SYear}
3. SEASONREC = {Tid, SYear, CWINS, WINS, CLOSSES, LOSSES}
   a. {Tid, SYear} -> {CWINS, WINS, CLOSSES, LOSSES}
4. MATCH = {Mid, Hid, Aid, HScore, AScore, MDate, SYear}
   a. Mid -> {Hid, Aid, Hscore, Ascore, MDate, SYear}
5. ROSTER = {Pid, FName, LName, PNumber, Pos, Gp, Tid, SYear}
   a. Pid -> {FName, LName, PNumber, Pos, Gp, Tid, SYear}
6. OFFPLAYER = {Pid, Patt, Ratt, Cmp, Rec, PYards, RYards, TD, Inter, Lng}
   a. Pid -> {Patt, Ratt, Cmp, Rec, PYards, RYards, TD, Inter, Lng}
7. DEFPLAYER = {Pid, Solo, Ast, Tfl, Tflyds, Sack, Sackyd, Inter}
   a. Pid -> {Solo, Ast, Tfl, Tflyds, Sack, Sackyd, Inter}

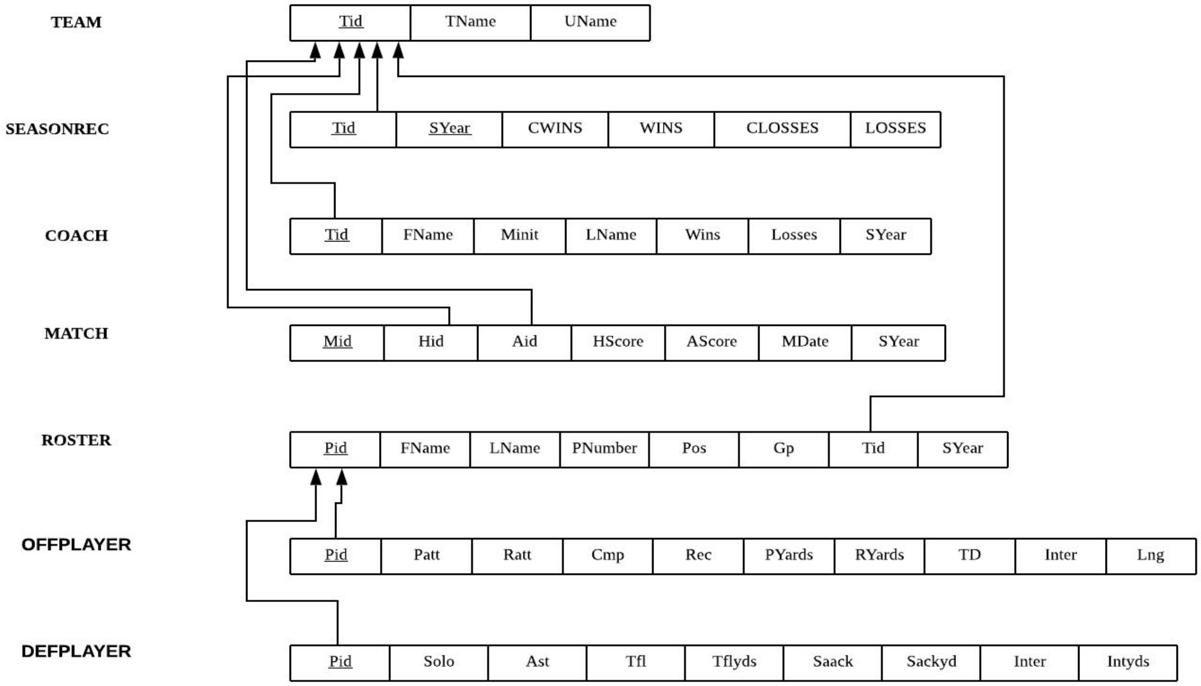
II. Identifying Candidate Keys:

1. TEAM
   a. Tid
2. SEASONREC
   a. (Tid, SYear)
3. COACH
   a. Tid
4. MATCH
   a. Mid
5. ROSTER
   a. Pid
6. OFFPLAYER
   a. Pid
7. DEFPLAYER
   a. Pid

III. Handling Functional Dependencies that have determinants which are not candidate keys:

None.

Appendix C: Relational Schema Diagram



TEAM: Tid | TName | UName

SEASONREC: Tid | SYear | CWINS | WINS | CLOSSES | LOSSES

COACH: Tid | FName | Minit | LName | Wins | Losses | SYear

MATCH: Mid | Hid | Aid | HScore | AScore | MDate | SYear

ROSTER: Pid | FName | LName | PNumber | Pos | Gp | Tid | SYear

OFFPLAYER: Pid | Patt | Ratt | Cmp | Rec | PYards | RYards | TD | Inter | Lng

DEFPLAYER: Pid | Solo | Ast | Tfl | Tflyds | Saack | Sackyd | Inter | Intyds

Appendix D:

Query 1:

Purpose: Get the team name, university name, and their coach first and last name. Display all coaches and their teams. Using the same table, list all coaches of the teams. It can help select who is the best head coach in the last 10 years in the pac-12 conference.

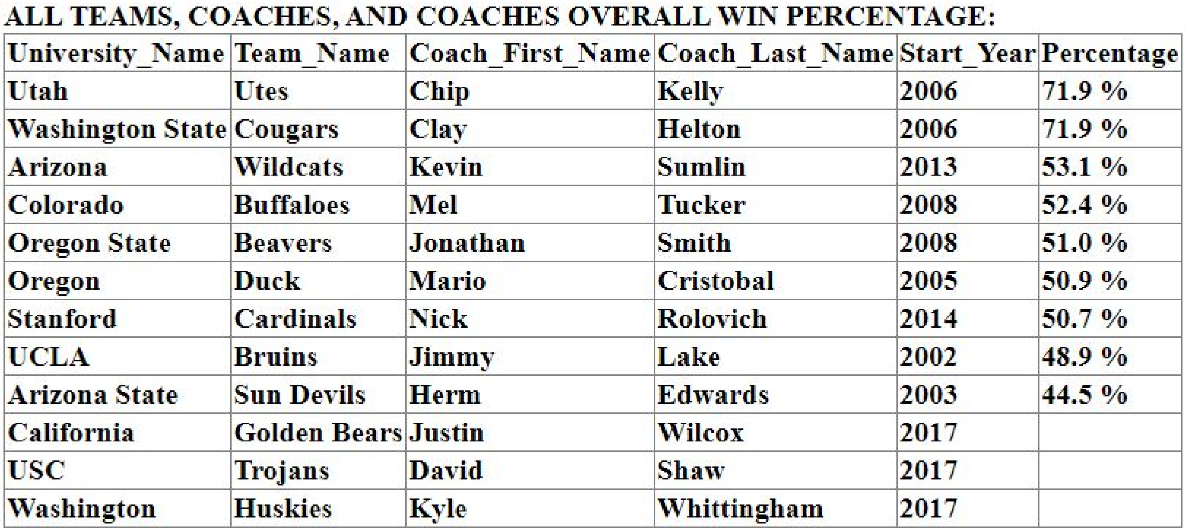
Description: Used a full join to get the team name, university name, and their coach first and last name. Return a list of coaches who have more than five years in the college football team, sorted in decreasing order of the win rate.

Query:

```
 Select w.University_Name, w.Team_Name, w.Coach_First_Name, w.Coach_Last_Name,
w.Start_Year, q.Percentage FROM(SELECT t.Uname University_Name, t.Tname Team_Name,
c.FName Coach_First_Name, c.LName Coach_Last_Name, c.SYear Start_Year
FROM TEAM t FULL JOIN COACH c on t.Tid = c.Tid) AS  w LEFT JOIN (SELECT
distinct(c.FName) FirstName, c.LName LastName,t.TName TeamName, t.UName University,
c.SYear StartYear,
Round(cast(Wins as decimal(11,2))/(Wins+Losses) * 100, 1)/100  as Percentage
FROM COACH c, TEAM t WHERE c.SYear <= 2014 and c.TID = t.TID) as q
ON q.TeamName = w.Team_Name Order By q.Percentage DESC
```

Expected: This table displays all university teams with their coaches. It will help fans to know who the coaches of the team are. The result is exactly as expected.

Result:



ALL TEAMS, COACHES, AND COACHES OVERALL WIN PERCENTAGE:

University_Name	Team_Name	Coach_First_Name	Coach_Last_Name	Start_Year	Percentage
Utah	Utes	Chip	Kelly	2006	71.9 %
Washington State	Cougars	Clay	Helton	2006	71.9 %
Arizona	Wildcats	Kevin	Sumlin	2013	53.1 %
Colorado	Buffaloes	Mel	Tucker	2008	52.4 %
Oregon State	Beavers	Jonathan	Smith	2008	51.0 %
Oregon	Duck	Mario	Cristobal	2005	50.9 %
Stanford	Cardinals	Nick	Rolovich	2014	50.7 %
UCLA	Bruins	Jimmy	Lake	2002	48.9 %
Arizona State	Sun Devils	Herm	Edwards	2003	44.5 %
California	Golden Bears	Justin	Wilcox	2017	
USC	Trojans	David	Shaw	2017	
Washington	Huskies	Kyle	Whittingham	2017	

Query 2:

Purpose: List all defensive players who have more than 5 sacks. It can help choose who is the best defensive player on the sack in the pac-12 conference.

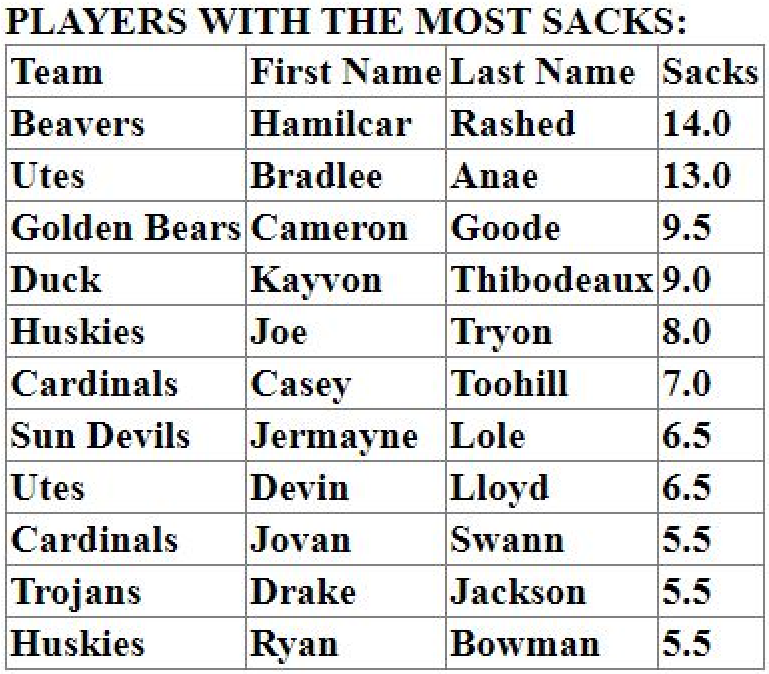
Description: select all the defensive players with more than 5 sacks using join.

Query:

```
SELECT t.Tname, r.Fname, r.Lname, o.sack
FROM TEAM t INNER JOIN ROSTER r on t.Tid = r.Tid
INNER JOIN DEFPLAYER o on r.pid = o.pid
WHERE  5 <  o.sack
ORDER BY o.sack DESC
```

Expected: The table displays Hamilcar Rashed from the Beavers is the best defensive player on the sack, and it also shows the rest of defensive players who have more than 5 sacks. The result is exactly as expected.

Result:



PLAYERS WITH THE MOST SACKS:

Team	First Name	Last Name	Sacks
Beavers	Hamilcar	Rashed	14.0
Utes	Bradlee	Anae	13.0
Golden Bears	Cameron	Goode	9.5
Duck	Kayvon	Thibodeaux	9.0
Huskies	Joe	Tryon	8.0
Cardinals	Casey	Toohill	7.0
Sun Devils	Jermayne	Lole	6.5
Utes	Devin	Lloyd	6.5
Cardinals	Jovan	Swann	5.5
Trojans	Drake	Jackson	5.5
Huskies	Ryan	Bowman	5.5

Query 3:

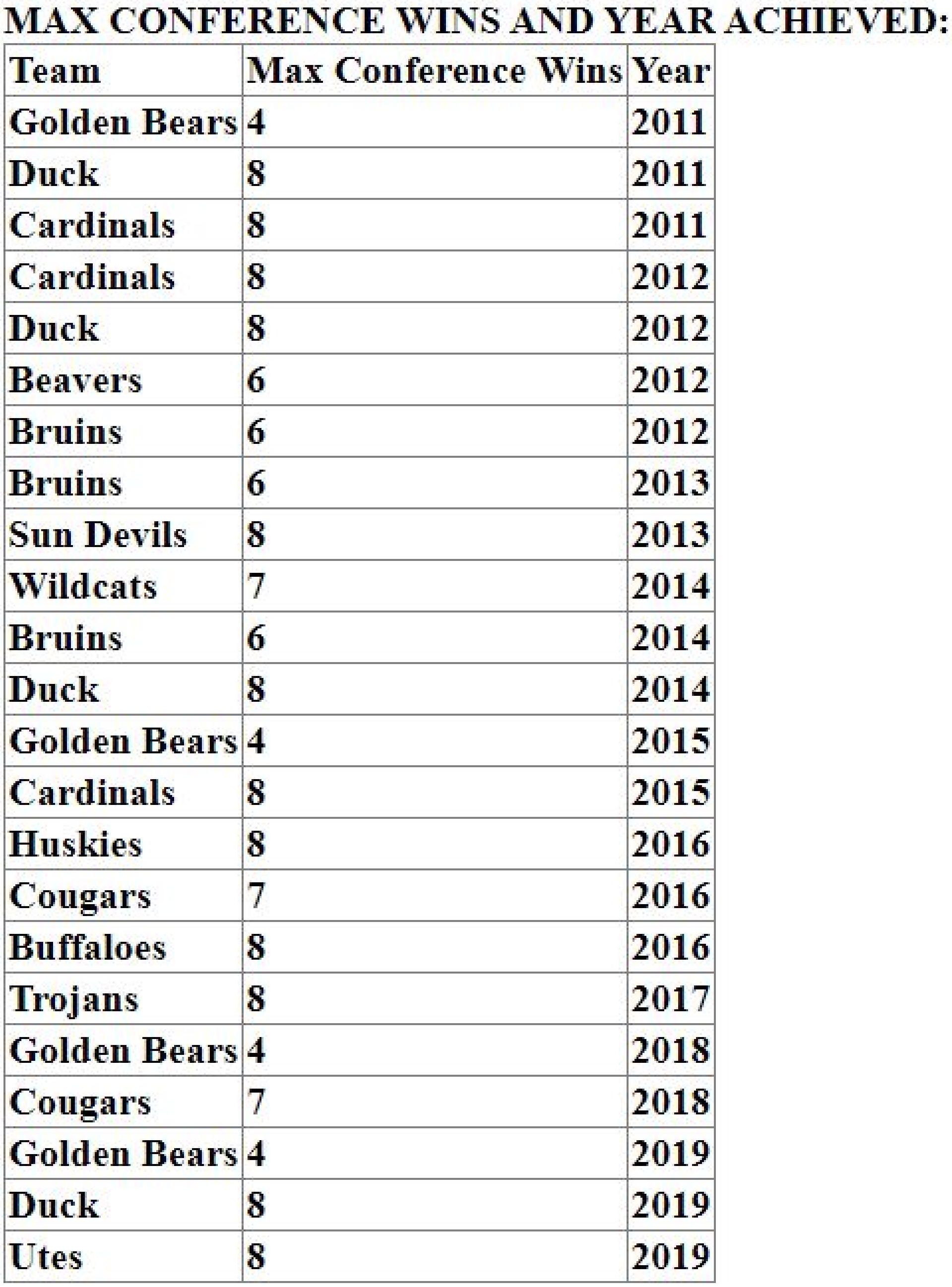
Purpose: Get the max conference wins for each team and the years that they achieved.

Description: Used correlated nested queries to get the max conference wins for each team and the years that they achieved.

Query:

```
SELECT t.TName, s.SYear, s.CWins
FROM TEAM t JOIN SEASONREC s on t.Tid = S.Tid
WHERE CWINS = (SELECT MAX(CWINS)
          FROM SEASONREC r
          WHERE s.Tid = r.Tid)
ORDER BY SYEAR;
```

Expected: This table displays when teams are most competitive in the pac-12 conference from 2011 to 2019. The result is exactly as expected.



MAX CONFERENCE WINS AND YEAR ACHIEVED:

Team	Max Conference Wins	Year
Golden Bears	4	2011
Duck	8	2011
Cardinals	8	2011
Cardinals	8	2012
Duck	8	2012
Beavers	6	2012
Bruins	6	2012
Bruins	6	2013
Sun Devils	8	2013
Wildcats	7	2014
Bruins	6	2014
Duck	8	2014
Golden Bears	4	2015
Cardinals	8	2015
Huskies	8	2016
Cougars	7	2016
Buffaloes	8	2016
Trojans	8	2017
Golden Bears	4	2018
Cougars	7	2018
Golden Bears	4	2019
Duck	8	2019
Utes	8	2019

Query 4:

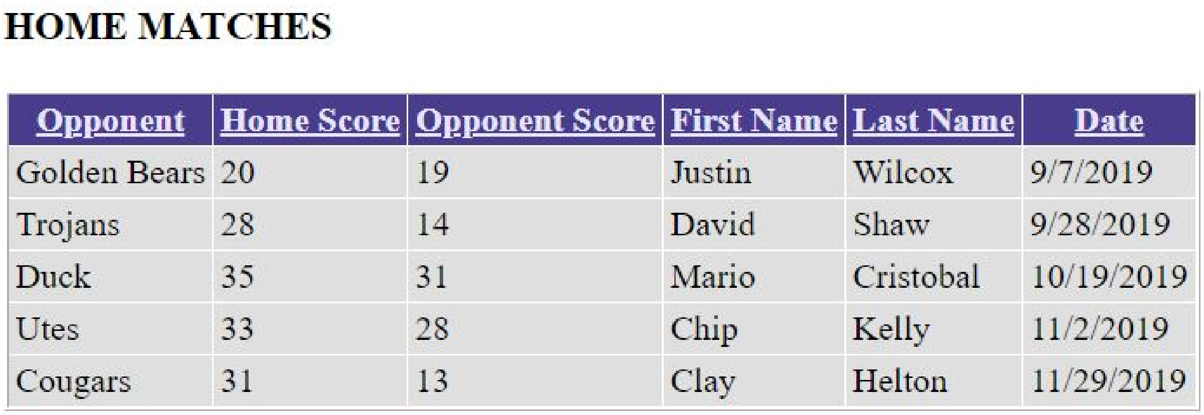
Purpose: Get the name of the home team and their score, and get the name of the away team and their score. All matches played with the Huskies.

Description: used inner join to get the name of home team and their score, and get the name of the away team and their score.

Query:

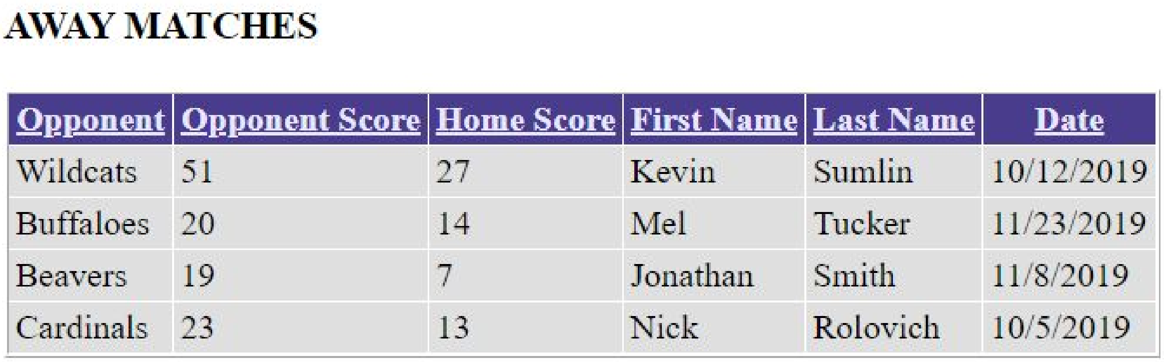
```
SELECT a.TName Away, m.HScore HomeScore, m.AScore AwayScore, c.FName FirstName,
c.LName LastName, m.MDate Date
FROM TEAM h INNER JOIN MATCH m ON h.Tid = m.Hid
INNER JOIN Team a ON m.Aid = a.Tid
INNER JOIN COACH c ON c.Tid = m.Aid
WHERE (m.Hid =10) AND YEAR(m.MDate) = 2019


SELECT h.TName Away, m.HScore HomeScore, m.AScore AwayScore, c.FName FirstName,
c.LName LastName, m.MDate Date
FROM TEAM h INNER JOIN MATCH m ON h.Tid = m.Hid
INNER JOIN Team a ON m.Aid = a.Tid
INNER JOIN COACH c ON c.Tid = m.Hid
WHERE (m.Aid = 10) AND YEAR(m.MDate) = 2019
```

Expected: This table shows all matches played with the Huskies. It can help the Huskies fans to follow their team record and the score of each match. The result is exactly as expected.



HOME MATCHES

Opponent	Home Score	Opponent Score	First Name	Last Name	Date
Golden Bears	20	19	Justin	Wilcox	9/7/2019
Trojans	28	14	David	Shaw	9/28/2019
Duck	35	31	Mario	Cristobal	10/19/2019
Utes	33	28	Chip	Kelly	11/2/2019
Cougars	31	13	Clay	Helton	11/29/2019



AWAY MATCHES

Opponent	Opponent Score	Home Score	First Name	Last Name	Date
Wildcats	51	27	Kevin	Sumlin	10/12/2019
Buffaloes	20	14	Mel	Tucker	11/23/2019
Beavers	19	7	Jonathan	Smith	11/8/2019
Cardinals	23	13	Nick	Rolovich	10/5/2019

Query 5:

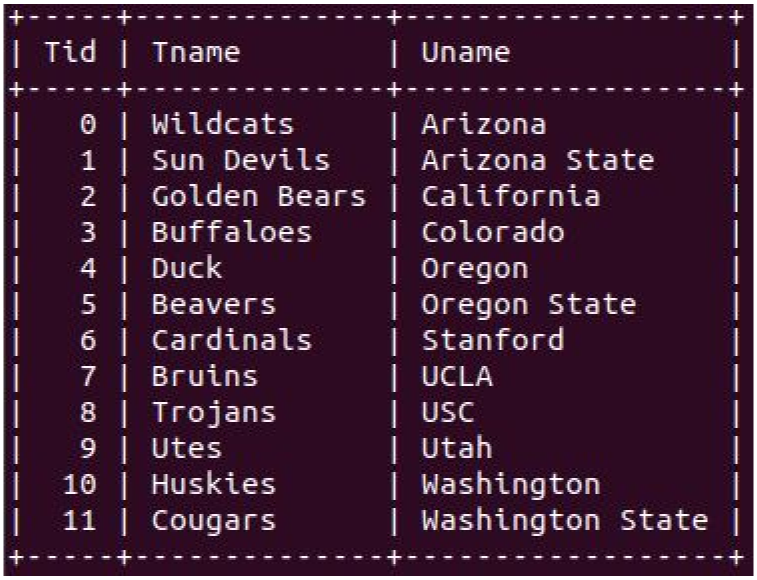
Purpose: Retrieve all the team table data so the user can predict a match between two teams by selecting them in a drop down menu.

Description: The statement is simply a select from statement, selecting the needed columns.

Query:

```
SELECT Tid, Tname, Uname FROM TEAM;
```

Expected: This table should show all the information in the TEAM relation.



```
+-----+--------------+------------------+
| Tid | Tname        | Uname            |
+-----+--------------+------------------+
|   0 | Wildcats     | Arizona          |
|   1 | Sun Devils   | Arizona State    |
|   2 | Golden Bears | California       |
|   3 | Buffaloes    | Colorado         |
|   4 | Duck         | Oregon           |
|   5 | Beavers      | Oregon State     |
|   6 | Cardinals    | Stanford         |
|   7 | Bruins       | UCLA             |
|   8 | Trojans      | USC              |
|   9 | Utes         | Utah             |
|  10 | Huskies      | Washington       |
|  11 | Cougars      | Washington State |
+-----+--------------+------------------+
```

Query 6:

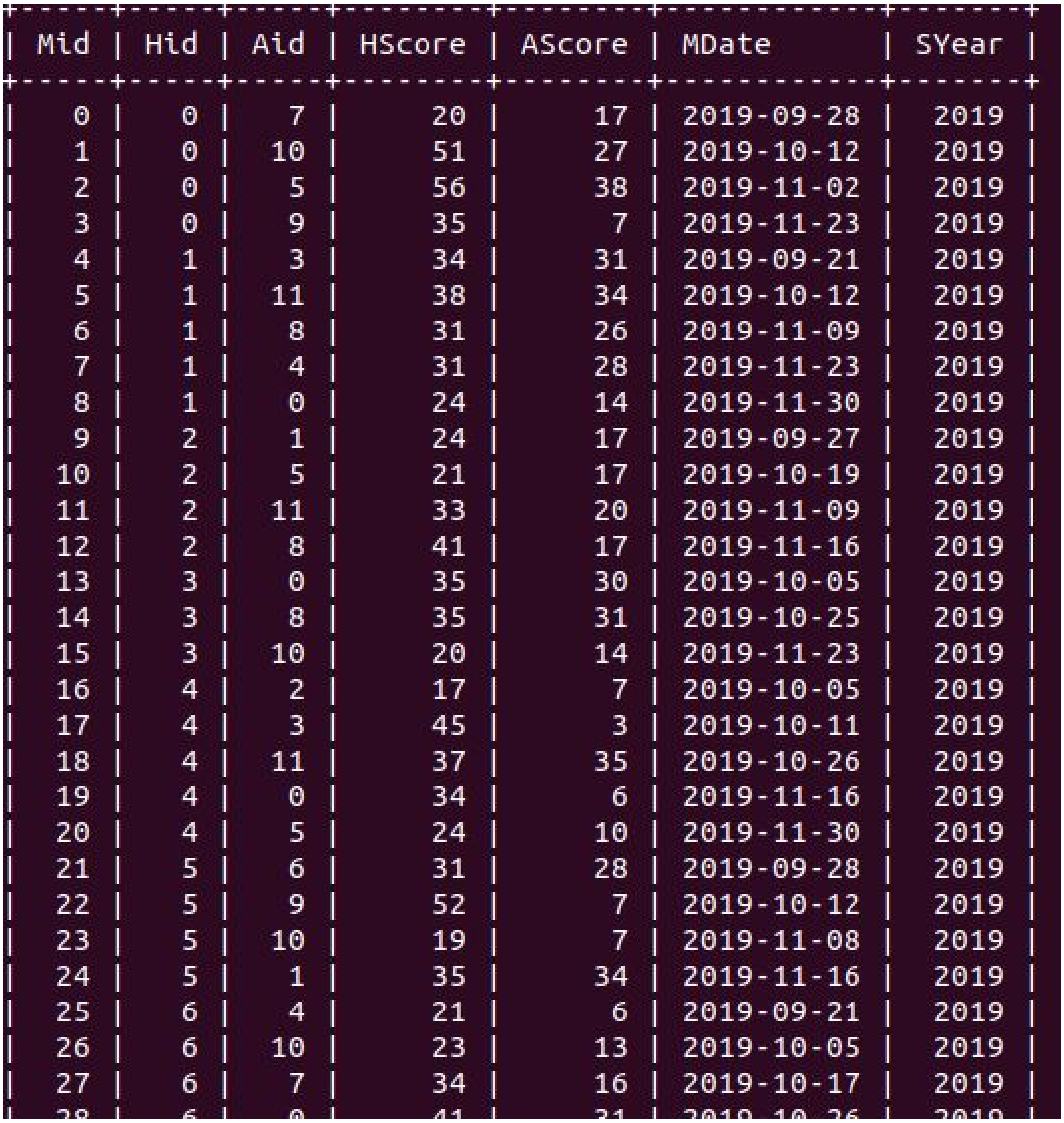
Purpose: Retrieve all the game data so the results can be used to train a support vector machine.

Description: The statement is simply a select from statement, selecting all columns.

Query:

```
select * from GAME;
```

Expected: This table should show all the information in the GAME relation.



```
+-----+-----+-----+--------+--------+------------+-------+
| Mid | Hid | Aid | HScore | AScore | MDate      | SYear |
+-----+-----+-----+--------+--------+------------+-------+
|   0 |   0 |   7 |     20 |     17 | 2019-09-28 |  2019 |
|   1 |   0 |  10 |     51 |     27 | 2019-10-12 |  2019 |
|   2 |   0 |   5 |     56 |     38 | 2019-11-02 |  2019 |
|   3 |   0 |   9 |     35 |      7 | 2019-11-23 |  2019 |
|   4 |   1 |   3 |     34 |     31 | 2019-09-21 |  2019 |
|   5 |   1 |  11 |     38 |     34 | 2019-10-12 |  2019 |
|   6 |   1 |   8 |     31 |     26 | 2019-11-09 |  2019 |
|   7 |   1 |   4 |     31 |     28 | 2019-11-23 |  2019 |
|   8 |   1 |   0 |     24 |     14 | 2019-11-30 |  2019 |
|   9 |   2 |   1 |     24 |     17 | 2019-09-27 |  2019 |
|  10 |   2 |   5 |     21 |     17 | 2019-10-19 |  2019 |
|  11 |   2 |  11 |     33 |     20 | 2019-11-09 |  2019 |
|  12 |   2 |   8 |     41 |     17 | 2019-11-16 |  2019 |
|  13 |   3 |   0 |     35 |     30 | 2019-10-05 |  2019 |
|  14 |   3 |   8 |     35 |     31 | 2019-10-25 |  2019 |
|  15 |   3 |  10 |     20 |     14 | 2019-11-23 |  2019 |
|  16 |   4 |   2 |     17 |      7 | 2019-10-05 |  2019 |
|  17 |   4 |   3 |     45 |      3 | 2019-10-11 |  2019 |
|  18 |   4 |  11 |     37 |     35 | 2019-10-26 |  2019 |
|  19 |   4 |   0 |     34 |      6 | 2019-11-16 |  2019 |
|  20 |   4 |   5 |     24 |     10 | 2019-11-30 |  2019 |
|  21 |   5 |   6 |     31 |     28 | 2019-09-28 |  2019 |
|  22 |   5 |   9 |     52 |      7 | 2019-10-12 |  2019 |
|  23 |   5 |  10 |     19 |      7 | 2019-11-08 |  2019 |
|  24 |   5 |   1 |     35 |     34 | 2019-11-16 |  2019 |
|  25 |   6 |   4 |     21 |      6 | 2019-09-21 |  2019 |
|  26 |   6 |  10 |     23 |     13 | 2019-10-05 |  2019 |
|  27 |   6 |   7 |     34 |     16 | 2019-10-17 |  2019 |
|  28 |   6 |   0 |     41 |     31 | 2019-10-26 |  2019 |
```

etc.

Query 7:

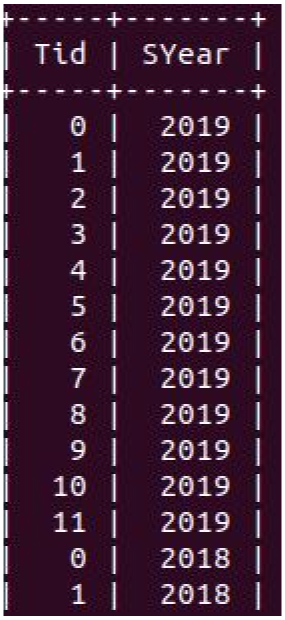
Purpose: Retrieve every possible team id/season year combination in order to eventually assign a strength vector to each.

Description: The statement is a nested select statement, using the distinct operator to avoid duplicate season years.

Query:

```
select Tid, SYear from TEAM, (select distinct(SYear) as SYear from GAME) as T;
```

Expected: This table should show every possible team id/season year combination.



```
+-----+-------+
| Tid | SYear |
+-----+-------+
|   0 |  2019 |
|   1 |  2019 |
|   2 |  2019 |
|   3 |  2019 |
|   4 |  2019 |
|   5 |  2019 |
|   6 |  2019 |
|   7 |  2019 |
|   8 |  2019 |
|   9 |  2019 |
|  10 |  2019 |
|  11 |  2019 |
|   0 |  2018 |
|   1 |  2018 |
```

etc.

Query 8:

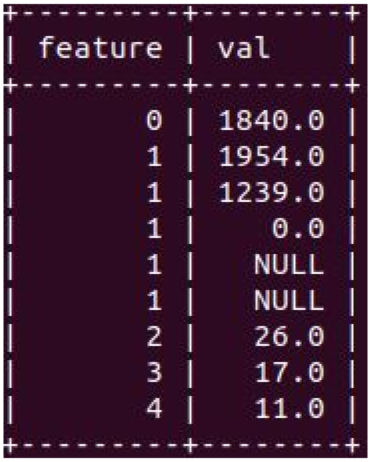
Purpose: Retrieve every possible team id/season year combination in order to eventually assign a strength vector to each.

Description: The statement is a nested select statement, using the distinct operator to avoid duplicate season years.

Query (Example, will be generated slightly differently depending on selected stats):

```
(select 0 as feature, SUM(RYards) as val
from (PLAYER left join OFFPLAYER on PLAYER.Pid = OFFPLAYER.Pid)
left join DEFPLAYER on PLAYER.Pid = DEFPLAYER.Pid where Tid = 0 and SYear = 2019  )
UNION ALL
(select 1 as feature, PYards as val
from (PLAYER left join OFFPLAYER on PLAYER.Pid = OFFPLAYER.Pid)
left join DEFPLAYER on PLAYER.Pid = DEFPLAYER.Pid where Tid = 0 and SYear = 2019
order by val desc limit 5)
UNION ALL
(select 2 as feature, Tflyds as val
from (PLAYER left join OFFPLAYER on PLAYER.Pid = OFFPLAYER.Pid)
left join DEFPLAYER on PLAYER.Pid = DEFPLAYER.Pid where Tid = 0 and SYear = 2019
order by val desc limit 1)
UNION ALL
(select 3 as feature, SUM(Sack) as val
from (PLAYER left join OFFPLAYER on PLAYER.Pid = OFFPLAYER.Pid)
left join DEFPLAYER on PLAYER.Pid = DEFPLAYER.Pid where Tid = 0 and SYear = 2019  )
UNION ALL
(select 4 as feature, SUM(Inter) as val
from (PLAYER left join OFFPLAYER on PLAYER.Pid = OFFPLAYER.Pid)
left join DEFPLAYER on PLAYER.Pid = DEFPLAYER.Pid where Tid = 0 and SYear = 2019  )
order by feature ASC, val DESC;
```

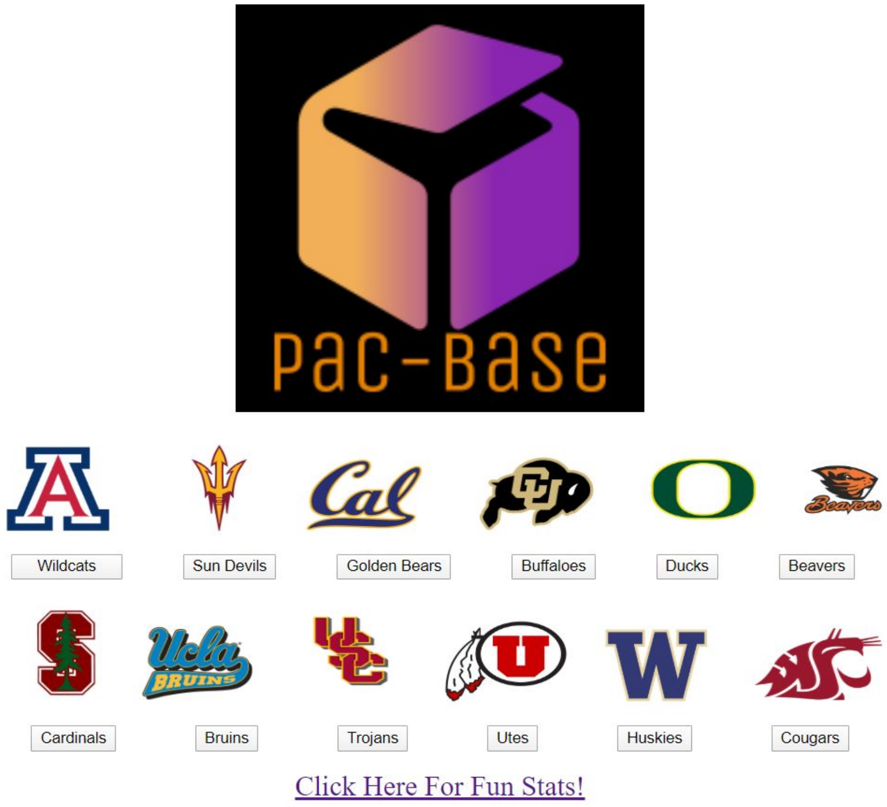
Expected: This table should show a combination of stats selected by the user that should represent the strength of a team in a given year (in the query above, this is Arizona in 2019). The vals within each feature should be decreasing, as each feature must either be a single aggregate value, or the statistic value of the top *n* players with the statistic. For example, the query above will show the top 5 players with feature 1 as shown below in the query's result.



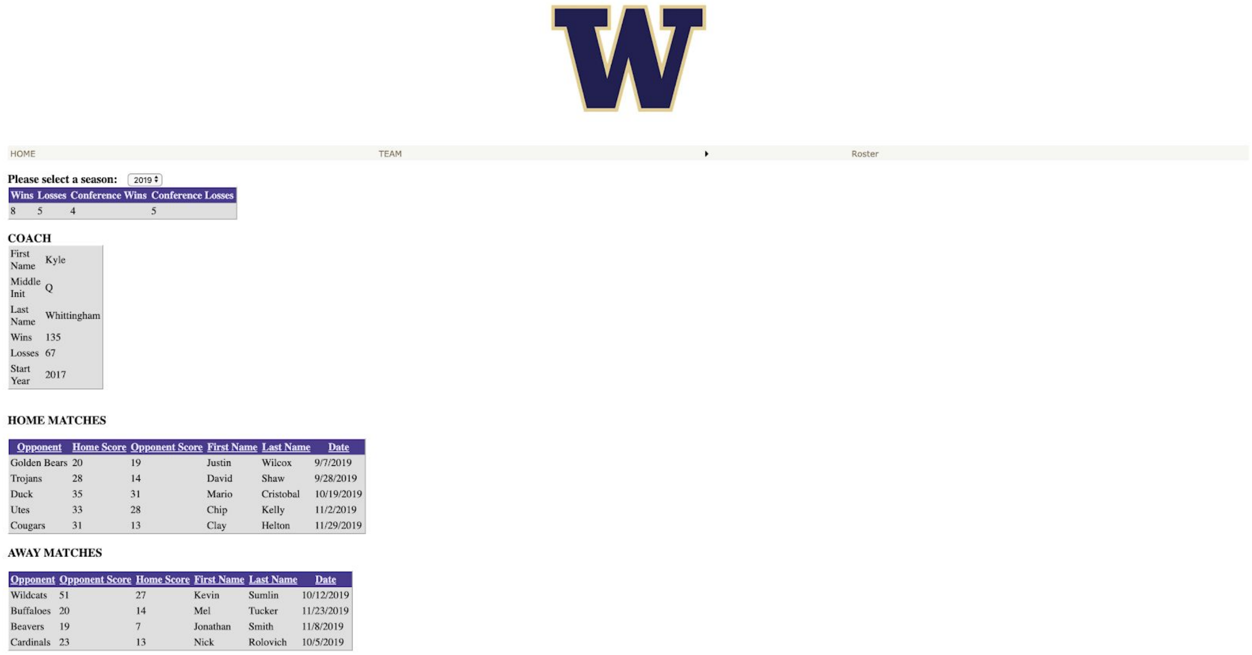
```
+---------+--------+
| feature | val    |
+---------+--------+
|       0 | 1840.0 |
|       1 | 1954.0 |
|       1 | 1239.0 |
|       1 |    0.0 |
|       1 |   NULL |
|       1 |   NULL |
|       2 |   26.0 |
|       3 |   17.0 |
|       4 |   11.0 |
+---------+--------+
```

Appendix E:

Below is our “Home” page that displays the logo of the database, all buttons with each team name and its icon, fun stats, and the instruction of the database. The page will navigate to team pages or fun stats.



Below is our “Team” page that displays the information about the selected team including their coach, their home and away matches, and the season record. There is a drop down list that users can change to see the season record of different years. Users (The page) can navigate to other team’s information using the “Team” tab and “Roster” page.



Please select a season: 2019

Wins	Losses	Conference Wins	Conference Losses
8	5	4	5

COACH

First Name	Kyle
Middle Init	Q
Last Name	Whittingham
Wins	135
Losses	67
Start Year	2017

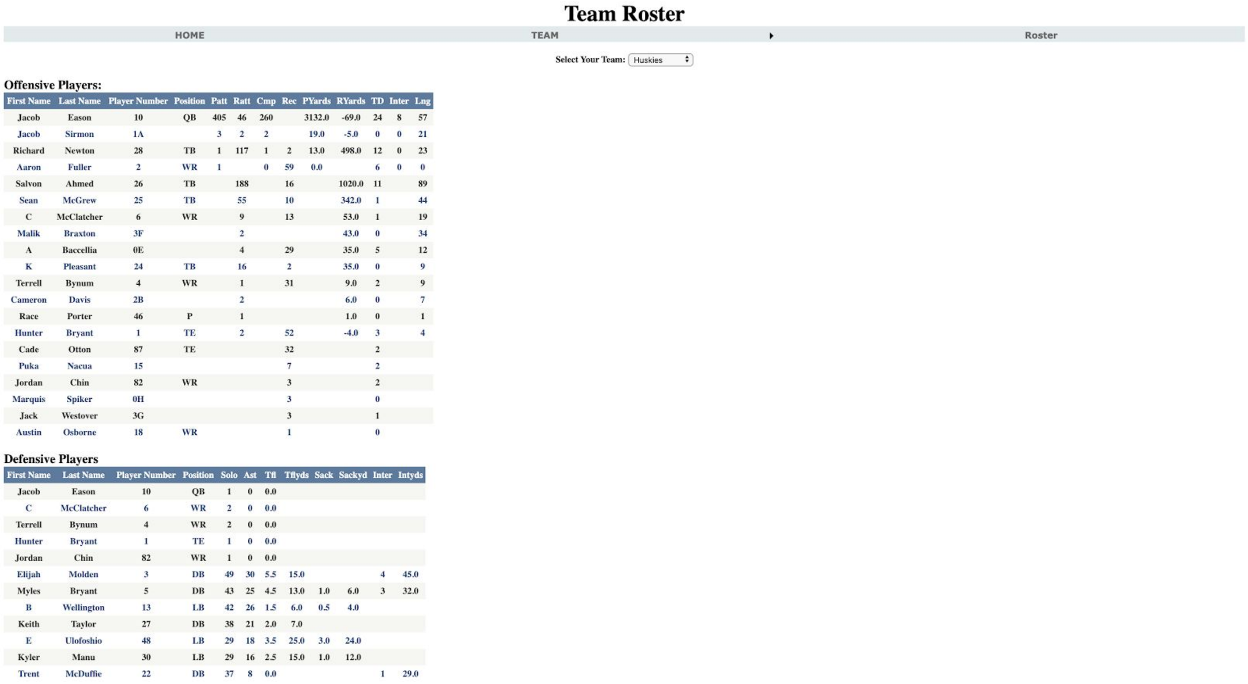
HOME MATCHES

Opponent	Home Score	Opponent Score	First Name	Last Name	Date
Golden Bears	20	19	Justin	Wilcox	9/7/2019
Trojans	28	14	David	Shaw	9/28/2019
Duck	35	31	Mario	Cristobal	10/19/2019
Utes	33	28	Chip	Kelly	11/2/2019
Cougars	31	13	Clay	Helton	11/29/2019

AWAY MATCHES

Opponent	Opponent Score	Home Score	First Name	Last Name	Date
Wildcats	51	27	Kevin	Sumlin	10/12/2019
Buffaloes	20	14	Mel	Tucker	11/23/2019
Beavers	19	7	Jonathan	Smith	11/8/2019
Cardinals	23	13	Nick	Rolovich	10/5/2019

Below is our “Roster” page that displays all the offensive and defensive players. There is also a drop down list that allows the users to change to different teams in the Pac-12 Conference. Users (The page) can navigate to the “Home” page and “Team” page.



Team Roster

Select Your Team: Huskies

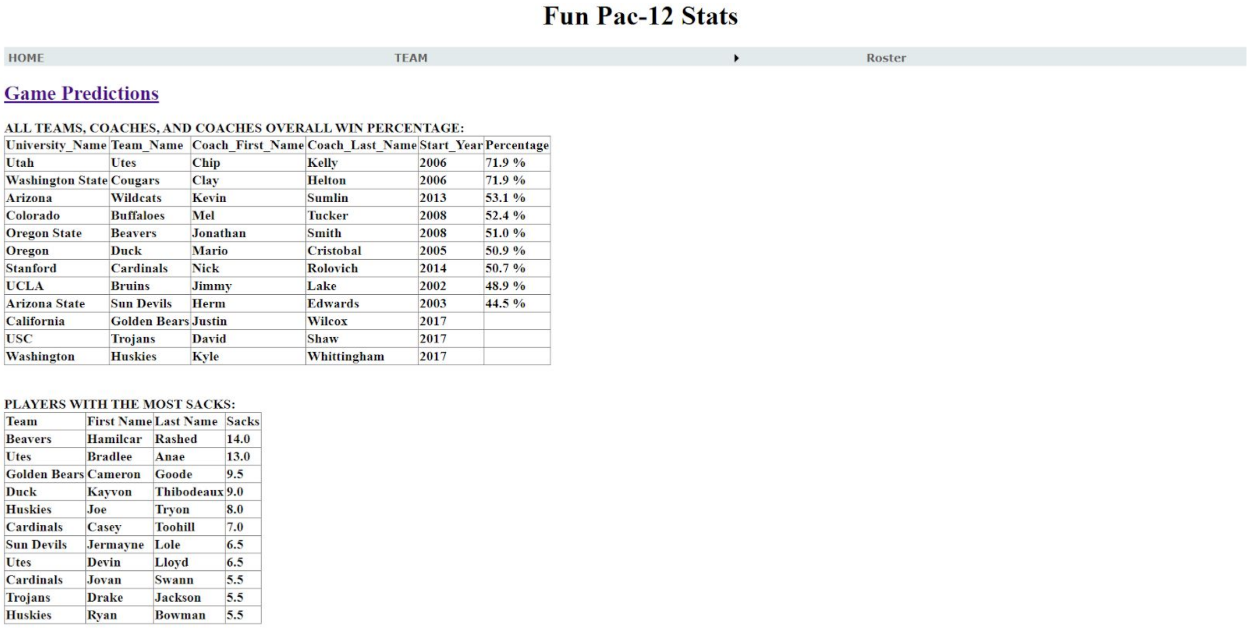
Offensive Players:

First Name	Last Name	Player Number	Position	Patt	Ratt	Cmp	Rec	PYards	RYards	TD	Inter	Lng
Jacob	Eason	10	QB	405	46	260		3132.0	-69.0	24	8	57
Jacob	Sirmon	1A		3	2	2		19.0	-5.0	0	0	21
Richard	Newton	28	TB	1	117	1	2	13.0	498.0	12	0	23
Aaron	Fuller	2	WR	1		0	59	0.0		6	0	0
Salvon	Ahmed	26	TB		188		16		1020.0	11		89
Sean	McGrew	25	TB		55		10		342.0	1		44
C	McClatcher	6	WR		9		13		53.0	1		19
Malik	Braxton	3F			2				43.0	0		34
A	Baccellia	0E			4		29		35.0	5		12
K	Pleasant	24	TB		16		2		35.0	0		9
Terrell	Bynum	4	WR		1		31		9.0	2		9
Cameron	Davis	2B			2				6.0	0		7
Race	Porter	46	P		1				1.0	0		1
Hunter	Bryant	1	TE		2		52		-4.0	3		4
Cade	Otton	87	TE				32			2		
Puka	Nacua	15					7			2		
Jordan	Chin	82	WR				3			2		
Marquis	Spiker	0H					3			0		
Jack	Westover	3G					3			1		
Austin	Osborne	18	WR				1			0		

Defensive Players

First Name	Last Name	Player Number	Position	Solo	Ast	Tfl	Tflyds	Sack	Sackyd	Inter	Intyds
Jacob	Eason	10	QB	1	0	0.0					
C	McClatcher	6	WR	2	0	0.0					
Terrell	Bynum	4	WR	2	0	0.0					
Hunter	Bryant	1	TE	1	0	0.0					
Jordan	Chin	82	WR	1	0	0.0					
Elijah	Molden	3	DB	49	30	5.5	15.0			4	45.0
Myles	Bryant	5	DB	43	25	4.5	13.0	1.0	6.0	3	32.0
B	Wellington	13	LB	42	26	1.5	6.0	0.5	4.0		
Keith	Taylor	27	DB	38	21	2.0	7.0				
E	Ulofoshio	48	LB	29	18	3.5	25.0	3.0	24.0		
Kyler	Manu	30	LB	29	16	2.5	15.0	1.0	12.0		
Trent	McDuffie	22	DB	37	8	0.0				1	29.0

Below is our “Query” page called fun states that displays three results of the queries from phase II. There is a link to connect to “Game Prediction”. The page also navigates to the “Home” page, “Team” page, or “Roster” page.



Fun Pac-12 Stats

Game Predictions

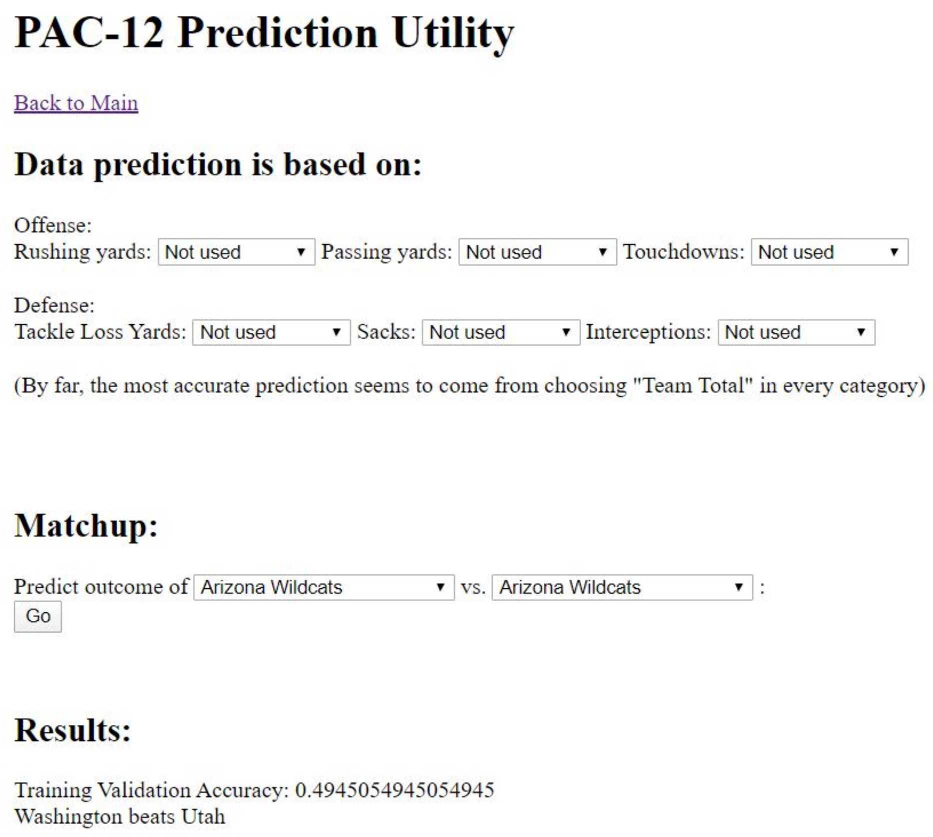
ALL TEAMS, COACHES, AND COACHES OVERALL WIN PERCENTAGE:

University_Name	Team_Name	Coach_First_Name	Coach_Last_Name	Start_Year	Percentage
Utah	Utes	Chip	Kelly	2006	71.9 %
Washington State	Cougars	Clay	Helton	2006	71.9 %
Arizona	Wildcats	Kevin	Sumlin	2013	53.1 %
Colorado	Buffaloes	Mel	Tucker	2008	52.4 %
Oregon State	Beavers	Jonathan	Smith	2008	51.0 %
Oregon	Duck	Mario	Cristobal	2005	50.9 %
Stanford	Cardinals	Nick	Rolovich	2014	50.7 %
UCLA	Bruins	Jimmy	Lake	2002	48.9 %
Arizona State	Sun Devils	Herm	Edwards	2003	44.5 %
California	Golden Bears	Justin	Wilcox	2017	
USC	Trojans	David	Shaw	2017	
Washington	Huskies	Kyle	Whittingham	2017	

PLAYERS WITH THE MOST SACKS:

Team	First Name	Last Name	Sacks
Beavers	Hamilcar	Rashed	14.0
Utes	Bradlee	Anae	13.0
Golden Bears	Cameron	Goode	9.5
Duck	Kayvon	Thibodeaux	9.0
Huskies	Joe	Tryon	8.0
Cardinals	Casey	Toohill	7.0
Sun Devils	Jermayne	Lole	6.5
Utes	Devin	Lloyd	6.5
Cardinals	Jovan	Swann	5.5
Trojans	Drake	Jackson	5.5
Huskies	Ryan	Bowman	5.5

Below is our “Game Prediction” page that will display the predicted result of any match between teams from the Pac-12 Conference. Users have to choose any data from offensive or defensive players and click “go”, it will use these statistics from past years to predict the result of the match. The page can navigate to the Homepage of the database.



PAC-12 Prediction Utility

Back to Main

Data prediction is based on:

Offense:
Rushing yards: Not used ▾ Passing yards: Not used ▾ Touchdowns: Not used ▾

Defense:
Tackle Loss Yards: Not used ▾ Sacks: Not used ▾ Interceptions: Not used ▾

(By far, the most accurate prediction seems to come from choosing "Team Total" in every category)

Matchup:

Predict outcome of Arizona Wildcats ▾ vs. Arizona Wildcats ▾ :

Results:

Training Validation Accuracy: 0.4945054945054945
Washington beats Utah