**Xbox One and Halo 5 Database**

**CSE 530A Group Project Report**

**Blake Myers, Zhiwei Yang**

**1. Read Me**

**1.1 Front end:**

Our project’s URL is

<http://ec2-54-68-132-200.us-west-2.compute.amazonaws.com/~zhiwei/cse530/xboxhalo5/>

The user enters a valid Xbox One Gamertag into the text field and selects an option of what data to pull from our database. The main Gamertag we used as a starting point was BigLew117. From there, we pulled his friends, and then all of their friends. To test, enter Biglew117 in the text field and bring up his friends list. All friends displayed can then be queried as well as all of their friends. Options are as follows:

|  |  |
| --- | --- |
| **Search Option:** | **Results:** |
| Friends | Displays the list of friends of the entered Gamertag |
| Games | Displays the list of games played by the entered Gamertag |
| Total Play Time | Calculated the Total Play Time of Player |
| Halo 5 Stats | Displays the Halo 5 statistics of the Player |
| Friend Recommendations | Run an algorithm to recommend friends based on Halo 5 stats |
| Game Recommendations | Run an algorithm to recommend games based on Halo 5 stats |

**1.2 Back end:**

Please follow the following steps to access the project’s database.

* Use command “ssh zhiwei@54.68.132.200” with password “zhiweiyang” to login the AWS machine.
* Use command “mysql -u zhiweiyang -p” with password “650230” to login MySQL databases.
* Use command “show databases;”, you will see a database named “xboxhalo5”, and that is our project’s database.
* The folder for the webpage is located here: /home/zhiwei/.html/cse530/xboxhalo5

**2. Project Description**

**2.1 Project goal**

Using in-game statistics from Halo 5, an Xbox One game, and profile information for Xbox One players, create a database that will allow a user to find mutual friends to play with who have similar skills and also recommend games based on Halo 5 play style.

The purpose would be for anyone who plays Halo 5 to have a better way to find new friends to play with who match your play style. For example, some of my friends are too competitive for me so playing with them is not enjoyable because they compete against players who are much better than me. Also, playing by yourself is not as enjoyable. The database would use data from your friend's friends list to find new mutual friends. Also, from another data source, game recommendations could be made based on finding what games people play who have a similar Halo 5 play style.

**2.2 Data source**

Halo 5 player stats and match data from developer.haloapi.com  
Xbox One player profile from xboxapi.com which includes a friends list and all Xbox One games played.  
The data sources above will be blended together based on the Xbox One Gamertag.

**2.3 How to get our data?**

The developers of Halo 5 have an API to access content within their Halo 5 database. The API allows a user to submit a player’s name or a unique match id and returns a JSON file with the data listed in it. We would need to request data on lots of players and strip out what useful information we need to put in our own database. Some of the data that we can use would be players matches, which include time played, whether they won or lost and how well they performed in the match. We will also grab a player’s overall statistics from their cumulative matches. In Halo 5, there are also several different game types that can be played online. We would grab all the different game types that a player has played (which are listed as Arena or Warzone) and determine more details about player preferences in Halo 5.  
To get the Xbox One player information (like Facebook account information, but for the Xbox One), there are several existing open APIs available to request certain information about a player. Using a player’s Gamertag, the different APIs allow the user to pull information about the player, including profile name, location, a list of games the player has played, and a list of friends of the player.

**2.4 Data scrape and clean**

Using a python script, we used the api keys obtained from each api source to query their database for the data we put into our database.

Once we have the data, which also comes in a JSON object, our Python script pulls the data we specified for our database and writes a SQL insert statement to a .sql file. Using the sql files with all the insert command, we are able to fill our database with the information we need.

**2.5 Data size**

Our database contains 8 tables. Four of the tables were filled with data from Xbox One player data, and the other 4 were filled with Halo 5 statistics:

In total, 318 player’s data was pulled for our database. Of those 318 players, 270 unique Xbox One games were played. Also, for the 318 players, 4421 game statistics were pulled, meaning that an average player has played about 14 unique games.

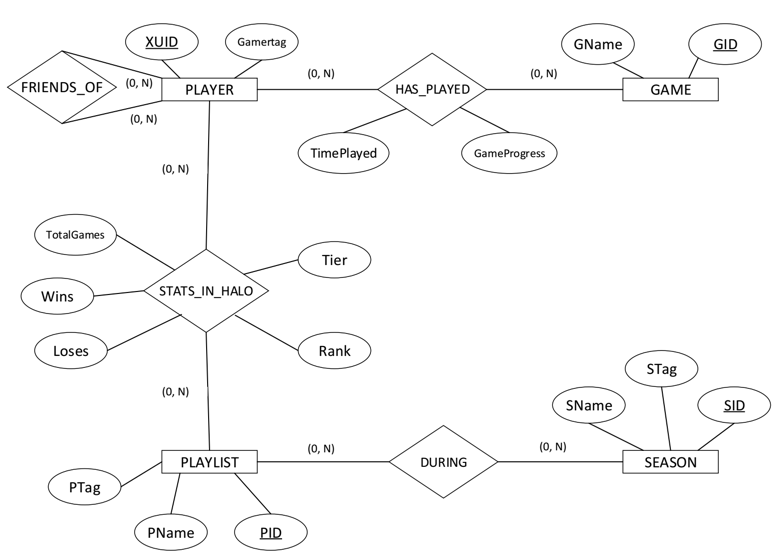
For Halo 5 statistics, a subset of about 120 of the players pulled had played Halo 5, so statistics were pulled for all 120 players. The 120 players had unique statistics in 821 playlists, which is sufficient enough for a basic recommendation system.

Here’s an outline of the contents of each table:

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Description | Number of Attributes | Number of Rows |
| PLAYERS | List of Xbox One Players: Gamertag and XUID | 2 | 318 |
| FRIENDS\_OF | List of Players Friends: PlayerXUID, FriendXUID | 2 | 375 |
| GAMES | List of all Games: GameTitle, GameID | 2 | 270 |
| GAMES\_PLAYED | List of Players Games: PlayerXUID, GameID, TimePlayed, GameProgress | 4 | 4421 |
| PLAYLISTS | List of Halo 5 Playlists: PlaylistName, PlaylistID | 3 | 61 |
| SEASONS | List of Halo 5 Seasons: SeasonName, SeasonID | 3 | 6 |
| RANKS | List of Halo 5 Ranks: RankName, RankID | 2 | 8 |
| PLAYER\_STATS | List of Players Halo 5 Stats: PlayerXUID, PlaylistID, SeasonID, TotalGames, Rank | 8 | 821 |

**3. Data Modeling**

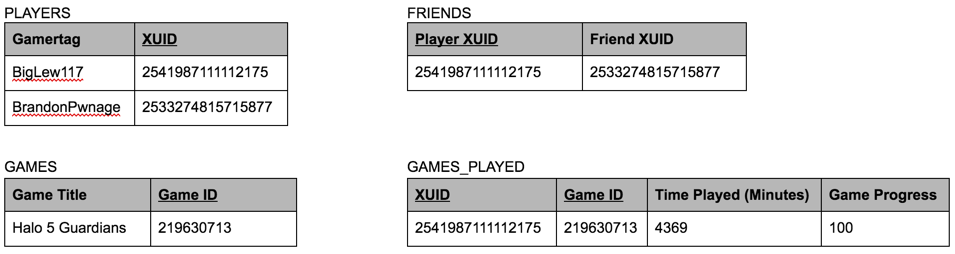
**3.1 ER diagram**



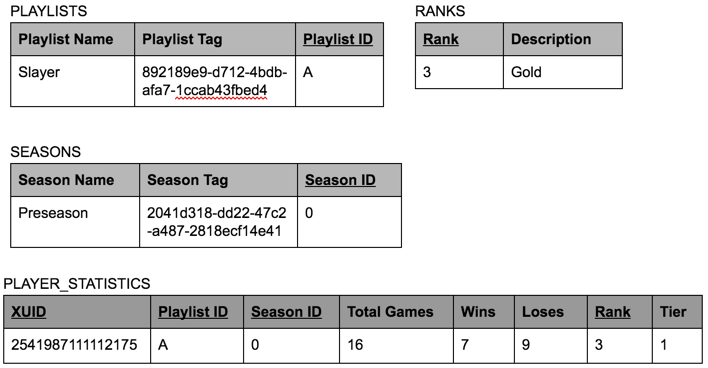
*Figure 3.1 Xbox Halo 5 ER Diagram*

**3.2 Relations**

All relations presented below are in the third normal form.



*Figure 3.2.1 Data from Xbox Platform sources*



*Figure 3.2.2 Data from Halo 5 sources*

**4. Programming**

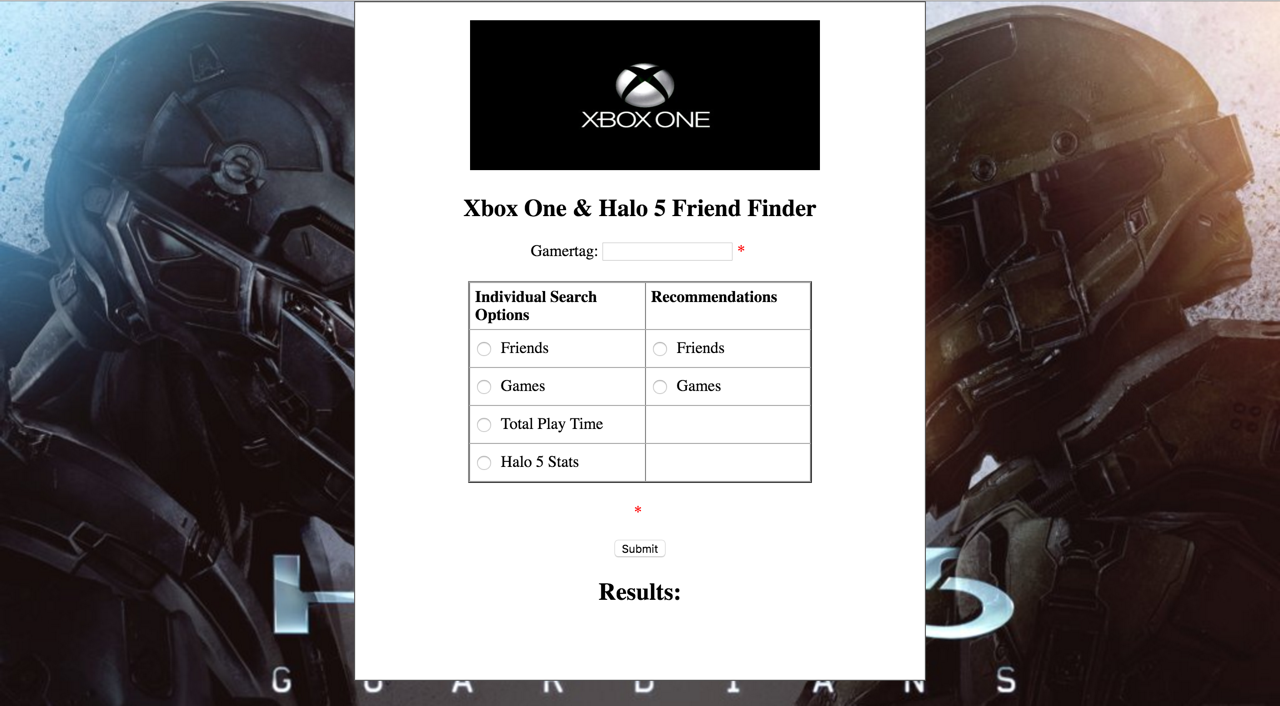
**4.1 System description**

For scraping and formatting our data, we used a Python script that pulled data from APIs, formatted the data, then wrote insert sql statements to a text file.

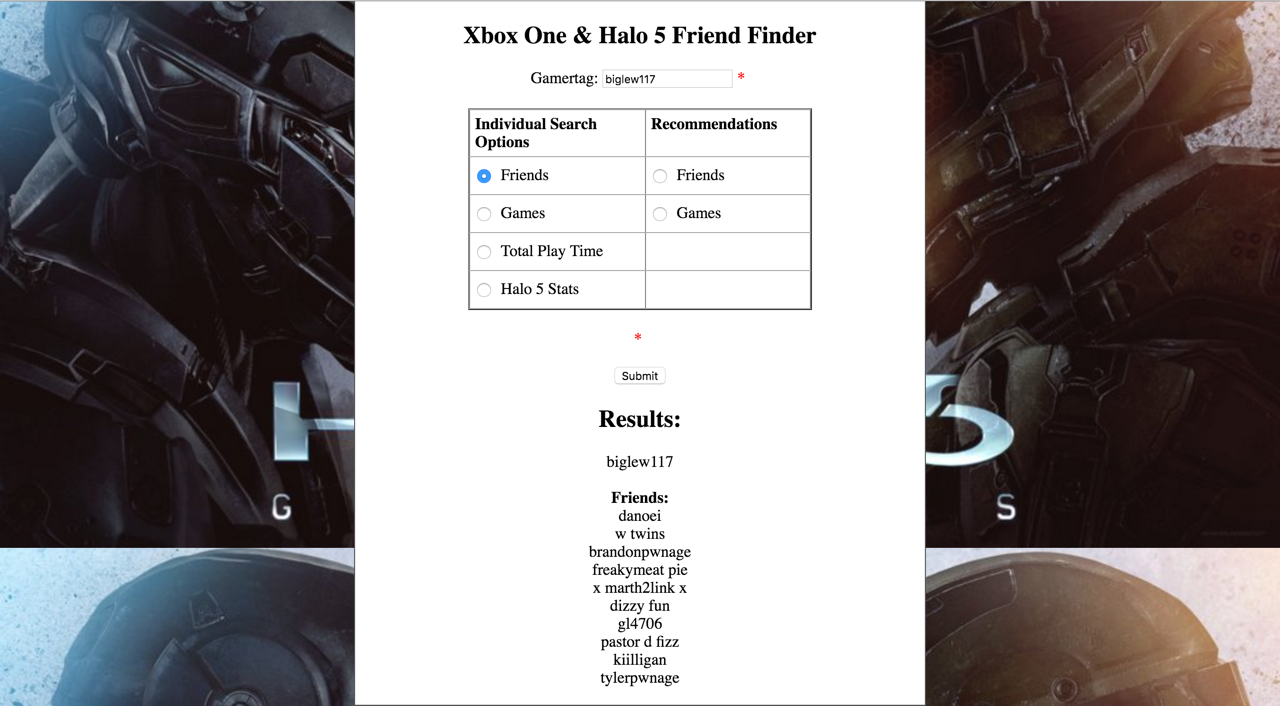
Our website is hosted on Amazon Web Services and uses Apache / PHP and MySQL. We have a MySQL database that is queried by our php files to get data to display on the webpage.

**4.1.1 Front end**

The UI we offer for uses is a website shown as Figure 4.1.1-1. Users enter their gamertag into “Gametag” field, and select the service they want using the radio buttons. After hitting “Submit”, they will get the results shown as Figure 4.1.1-2.



*Figure 4.1.1 Project Website*



*Figure 4.1.1-2 The Friend List of A User Named “biglew117”*

**4.1.2 Back end**

Our data is stored in a MySQL database called “xboxhalo5”, and uses PHP mysqli class to communicate.

**4.1.3 Data script**

As we mentioned before, we use a Python script (xboxhalo.py) to grab and clean the data we used in our database

**4.2 Sample SQL statements**

Here are SQL Statements that were used in our project:

// Query a players friends

select fxuid from FRIENDS\_OF where pxuid=?;

select gamertag from PLAYERS where xuid=?;

// Query a players games

select gid, timeplayed, gameprogress from GAMES\_PLAYED where playerxuid=?;

select gametitle from GAMES where gameid=?;

// Query a players Halo 5 stats and convert seasonid and playlistid to names

select \* from PLAYER\_STATS where gamerxuid=? order by seasonid;

select pname from PLAYLISTS where pid=?;

select sname from SEASONS where sid=?;

select rankname from RANKS where rankid=?

// Query games played by all users and list in order

select gid, count(gid) from GAMES\_PLAYED group by gid order by count(gid);

// Query average player ranking

select avg(rank) from PLAYER\_STATS where rank > '0';

// Query a specific XUID and grab the top three playlists played

select GamerXUID, PlaylistID, TotalGames, Rank from PLAYER\_STATS

where GamerXUID = '2541987111112175' AND TotalGames > 9

Order BY TotalGames DESC

Limit 3;

// Query a specific XUID and grab the top three games played

select PlayerXUID, GID, TimePlayed from GAMES\_PLAYED

where PlayerXUID = '2541987111112175'

Order BY TimePlayed DESC

Limit 3;

**4.2.1 Create table**

The table creation sql file can be found in the Supporting\_Files.zip under the SQL Files folder: createTables.sql

**4.2.2 Insert**

Our script create a sql file that inserts all our data into the database. The file insertAll.sql can be found in the Supporting\_Files.zip under SQL Files folder.

Example Insert statement:

INSERT INTO PLAYER\_STATS (GamerXUID, PlaylistID, SeasonID, TotalGames, Wins, Loses, Rank, Tier) VALUES (2533274801392686, '21', 'W', 2, 1, 1, 0, 0);

**4.2.3 Recommend friends or games for a player**

Our algorithm for recommending players takes the top 3 playlists of a player, searches through the other players top 3 playlists, and then sees if at least 2 of the three playlists match another player. If that matches, then the ranking of the player is considered.

For game recommendations, we pull the same list that the friend recommendations does, and then find out of those friends that were recommended, what games are most popular with the recommended friends.

**4.3 Sample script**

Our full python script xboxHalo.py can be found in the Supporting\_Files.zip in the Scripts folder. Our script uses the api keys to query the databases and then formats the data and writes insert sql statements to a sql file. Here is a sample function to generate INSERT queries for table GAME\_PLAYED.

def print\_player\_game\_stats():

myXuid = get\_xuid("DiZzY fUn")

response = get\_games(myXuid)

games\_list = response['titles']

for game in games\_list:

if game['name'].find('Xbox One') == 0:

print "\n"

elif game['titleType'].find('DGame') != -1:

timeFound = None

response = get\_game\_stats(myXuid, game['titleId'])

sublist = response['statlistscollection']

statsdict = sublist[0]

statslist = statsdict['stats']

for item in statslist:

if item['name'].find('GameProgress') != -1:

if 'value' in item:

gameProg = item['value']

if item['name'].find('MinutesPlayed') != -1:

timePlayed = item['value']

timeFound = True

if not timeFound:

sublist = response['groups']

subdict = sublist[0]

sublist2 = subdict['statlistscollection']

statsdict = sublist2[0]

statslist = statsdict['stats']

for item in statslist:

if item['name'].find('HoursPlayed') != -1:

timePlayed = item['value'] \* 60

print 'INSERT INTO GAMES\_PLAYED (PlayerXUID, GID, TimePlayed, GameProgress)'

print(' VALUES ({}, {}, {}, {});'.format(myXuid, game['titleId'], timePlayed, gameProg))

return

**5. Conclusion**

**5.1 Goal of Project**

The goal of our project was to create a database that contains both Xbox One player data and Halo 5 statistics. The database can then be queried to find and recommend friends for players as well as recommend games. Our data supports our goal. We were able to pull enough player data and create an algorithm that compares our entire database’s Halo 5 statistics to make friend and game recommendations.

**5.2 Additional Queries Supported By Database**

Our database is able to analyze the games played data as well as the halo 5 statistics to do more than recommendations. Of the players pulled, we can use the game data of each player to find the most popular games played, as well has how successful players were at certain games, based on the game progress pulled. Also, a player could easily compare statistics of all their friends to see who is the best at a certain game. For the Halo 5 statistics, we could analyze who has the highest rank in a playlist, what playlist is played the most, and the average rank of players in a playlists.

**6. Appendix**

All scripts can be found in the Supporting Files zip file under the Scripts folder.

Contents of Supporting\_Files.zip:

* Scripts:
  + xboxHalo.py
  + supporting files used or created by xboxHalo.py
* WebpageSource
  + Index.php, sql.php as well as all assets used on webage
* SQL Files
  + Create.sql, createTables.sql, insertAll.sql
* Tables.doc
* Xbox\_Halo5\_ER\_Diagram.pdf