CSE 412 Project: Phase 3

Notes:

- Each query can be referred to by their number in example_queries.sql
- It's assumed that user input is when the user takes a direct action to tweak how the query is ran.

Query Descriptions

Query 1: Making a Post/Modlist on the Website

Description: Main functionality of the web application. This query takes in the parameters of the user's post description, title, game, mods for the game, and a config/importance rating for each mod. Each post must have at least one mod, and optionally for each mod an importance rating chosen for each mod.
Parameters taken from user: Title, description, game, mods for post, importance ratings for each mod.
Type: Insert

Query 2: Most Posted About Games

Description: Retrieves the top 25 posted about game with or without a given genre.
Parameters taken from user: Genre (optional), Offset # (Load more button)
Type: Select
Changes: We added an offset into the query so the user can request to view more posts instead of loading just 25 posts.

Query 3: Leaderboard of Users with the most favorited posts

Description: This query retrieves the usernames of those who have been favorited the most.
Parameters taken from user: Offset # (Load more button)
Type: Select
Changes: An offset was added into the query so the user can request to view more posts.

Query 4: Most Visited Posts

Description: This query selects posts with the most unique visits. Given optional parameters the user can narrow down posts selected by selecting game genre to look for, and post title itself.
Parameters taken from user: Title (optional), Genre (optional)
Type: Select
Changes: We no longer are looking for most visited posts for a game. Instead we are looking for the most visited posts in general but if the user wants, they can input a title and game genre to narrow down the results.

Query 5: Populating the User's News Feed
Description: This retrieves posts where the author is followed by the user. This query is used to populate the Followed Users' Posts page.
Parameters taken from user: Offset # (Load more button)
Type: Select

Query 6: Create New User on the Website
Description: Inserts a new row taking in the user parameters of username, email, and password. This is used to actually create a new user so for login and authentication purposes.
Parameters taken from user: Username, email, password
Type: Insert

Query 7: Login Authentication
Description: When a user attempts to login, this query will select the user's information check against it the user's inputted password. This has ch
Parameters taken from user: Username, password
Type: Select
Changes: Previously, the query would retrieve a 1 if password was correct, and 0 if it didn't since it selected a count. We decided it would be simpler to instead query the user's information to retrieve the password so that when the password is checked server side our application would have an easier time receiving the response to work with.

Query 8: Add new mod
Description: This query adds a new mod for the website with the given user parameters.
Parameters taken from user: Game, name of mod, link to the mod, type of mod (optional), description
Type: Insert

Query 9: Add new game
Description: Enables functionality of adding a new game into the website.
Parameters taken from user: Name of game, year of release, genre, description
Type: Insert

Query 10: Change email
Description: Upon request this query will update the user's email address. This is used on the login page once the user is logged in.
Parameters taken from user: Email
Type: Update

Query 11: Commenting
Description: Inserts a new comment for a given post.
Parameters taken from user: User comment
Type: Insert

Query 12: Get comments for given post
Description: For a post it retrieves its comments.
Parameters taken from user: None from user
Type: Select

Query 13: Favoriting a post
Description: Allows users to favorite a post, this query keeps track of favorited post for users.
Parameters taken from user: Favorite button (which contains post information)
Type: Insert

Query 14: Unfavoriting a post
Description: This allows users to unfavorite a post, removing it from the table.
Parameters taken from user: Unfavorite button
Type: Delete

Query 15: Get user's favorite posts
Description: When requested this query will retrieve posts that are favorited by the user. This is used on the user's profile page.
Parameters taken from user: No user parameters
Type: Select

Query 16: Record a visit
Description: When a user visits a mod post we record it with either a select, insert, or update. This is used tracking most recent visits for a post for a user.
Parameters taken from user: Just visiting the page logged in
Type: Select, Insert, Update

Query 17: View visited posts of a user
Description: This will retrieve posts of a user that they have visited. This is used on the user profile page.
Parameters taken from user: None from the user, they just have to visit a page
Type: Select

Query 18: Get number of followers
Description: Retrieves count for the number of users a given user is following. This is used on the user profile page.
Parameters taken from user: None by the user
Type: Select
Changes: Removed unnecessary group by clause

Query 19: Retrieve followers
Description: This helps populate a list of followers with a given user. This is used on the user profile page.
Parameters taken from user: None from user
Type: Select

Query 20: Get number people user is following
Description: Query the number of people a given user is following. This is used on the user profile page.
Parameters taken from user: None from user
Type: Select
Changes: Removed unnecessary group by clause

Query 21: Retrieve people user is following
Description: This retrieves the users that the user is following. This is used on the user profile page.
Parameters taken from user: None from user
Type: Select

Query 22: Follow a user
Description: Records a new record of user following another user. This is used by other queries to help populate the Followed Users' Posts page.
Parameters taken from user: Follow button
Type: Insert

Query 23: Unfollow a user
Description: This is used to unfollow a user.
Parameters taken from user: Unfollow button
Type: Delete

Query 24: Retrieve information for mod list
Description: When a user requests to view information of a mod list, this query gets the information for it.
Parameters taken from user: None from user
Type: Select
Changes: Added the importance rating when retrieving mod information

Query 25: Get posts by user
Description: This retrieves a list of posts that are created by the user.
Parameters taken from user: When a user requests a user's page
Type: Select

Query 26: Deactivate account
Description: This is used to deactivate a user's account. Can be done within the login page.
Parameters taken from user: Delete button, user password
Type: Select, Update
Changes: We added a check for authentication before deactivating the account, it's like Facebook so we only want the user to delete themselves if they're sure

Query 27: Get user information (NEW)
Description: Gets the username and email for a given user.
Parameters taken from user: None from user
Type: Select

Query 28: Retrieve all games (NEW)
Description: This retrieves a list of all games, which is used for submitting a mod list so that a user can select from a drop down.
Parameters taken from user: None from user
Type: Select

Query 29: Get list of mods for a game (NEW)
Description: This query is used on the create a mod list page, where the user is given a list of mods for a given game from the user's selection.
Parameters taken from user: The game selected
Type: Select

Query 30: Retrieve latest post (NEW)
Description: This retrieves the latest posts created, populating the home page.
Parameters taken from user: Offset (Load more button)
Type: Select

Query 31: Check if user follows another (NEW)
Description: This is used to create the follow button to show a follow or unfollow button and use a proper query when pressed.
Parameters taken from user: None from user
Type: Select

Query 32: Check if user favorited a post (NEW)
Description: This helps check if a user has favorited a post, similarly to Query 31 we use it to show the right button and proper query is used.
Parameters taken from user: None from user
Type: Select

Query 33: Retrieve genres (NEW)
Description: This retrieves the list of genres and is used for filtering when searching for leaderboards that filter by genre.
Parameters taken from user: None from user
Type: Select

Query 34: Get experts for a game (NEW)
Description: This gets users and their respective game where the user has written about every mod for the given game. This is only used in the expert tab of the User Leaderboards page.
Parameters taken from user: Game name, Offset (Load more button)
Type: Select

Query 35: Get most talkative users (NEW)
Description: We retrieve a list of the users who have commented on other people's posts the most. This has a having clause that checks if the user has commented on a certain number of unique users' posts. This allows us not only to see who is commenting on the most posts, but also who's talking to the greatest amount of different users.
Parameters taken from user: Number of unique different users' post, Offset (Load more button)
Type: Select

Query 36: Get information for a game (NEW)
Description: This retrieves information for a given game. This is used on the game's page.
Parameters taken from user: None from user
Type: Select

Query 37: Get top visited posts for a game (NEW)
Description: For a given game this retrieves the most visited posts. This is used on the game's page.
Parameters taken from user: None from user
Type: Select

Query 38: Get addons for a game (NEW)
Description: This retrieves add ons for a given game, used on the game's page itself.
Parameters taken from user: None from user
Type: Select

Query 39: Get graphical mods for a game (NEW)
Description: This retrieves graphical add ons for a given game, used on the game's page itself.
Parameters taken from user: None from user
Type: Select

Query 40: Get unofficial patches for a game (NEW)
Description: This retrieves all unofficial patch add ons for a given game, used on the game's page itself.
Parameters taken from user: None from user
Type: Select

Query 41: Get mods without a type for a game (NEW)
Description: This retrieves all mods that do not have a specific category for a given game. This is also used on the game's page itself.
Parameters taken from user: None from user
Type: Select

Description for Most Important Indexes

Index Operations:

```
CREATE INDEX mod_for_game_index_for_game on
mod_for_game(game_name,game_release_year);
CREATE INDEX pfm_index_for_post on post_features_mod(title,date,time);
CREATE INDEX post_username_index on post(username);
CREATE INDEX post_game_index on post(game_name,game_release_year);
```

Description:
We want to talk about these four indexes instead of just one because these four helped speed up our main universal quantifier query significantly. We chose these indexes because they were the attributes that were used in the join conditions of the different queries (and thus, were the ones being looked up the most). Using postgresql's analyze query, we saw a significant increase in runtime from before and after the indexes were created. While these indexes will not affect any of the update queries that we are using for the website, they will slow down inserts for all of the tables because the index will also have to be updated for the new entries.

Run Time Comparison:
Original Run Time: 110818.362 ms
Run Time After Indexes: 254.788 ms

Changes from Phase II

There were several changes from Phase II, as well as many additional queries added. The new queries and changes of the queries are identified above in the report. We also have added new indexes, that can be found within the indexes.sql file. However we didn't make any changes to the schema, or make any new assertions or triggers. For the most part we kept most of the design the same, we didn't stray from our ER diagram, and have made mostly improvements and additional queries for the final phase.