Project Report

Link to Video: https://youtu.be/jJ7WF1kJjqY

1. Changes in the directions of our project

We simplified our original what-if idea, which was over-ambitious and too difficult given the time frame allotted. We decided not to maintain original data and what-if data simultaneously, but used our stored procedure as an opportunity to consider a scenario of our original what-if idea. We also originally wanted the users to be able to insert/delete into any table but due to structure of the tables and dependencies, we were unable to do so. Further, we didn't implement the visualizations because our time constraints didn't give us a chance to do so, and we thought the other aspects deserved a higher priority.

2. What our application achieved regarding its usefulness

One of our successes was figuring out how to make our stored procedure relevant and interesting in the scope of our application. It gave us a chance to explore one facet of our original what-if idea that we found interesting. Our well-functioning and simple user interface was also a major achievement because it made the overall application easy to use. In particular, we are proud of our search functionality, which allows the user to search any keyword in the database in a straightforward way.

Unfortunately, we did fail to meet the standard that we set for ourselves concerning making the application useful for the user. We weren't able to add the functionality that allows rankings and weightedRankings to update automatically on an insert, delete, or update to the database. We also weren't able to give the user the opportunity to insert into tables other than NOC, due to the constraints that existed between the tables. Implementing this would have required a lot more error checking and case handling, which our timeline simply didn't provide.

3. Changes to the schema or source of the data

Within stages 2 and 3, when we first created our schema, we made some minor changes to our schema. Once those were decided on and approved by our TA, we did not alter the schema or source of the data further. The only modifications we made concerned the effects of our CRUD applications on the original data. Ultimately, based on the feedback we received, we maintained the original schema and source of data that we started with.

4. Changes to ER diagram and table implementations

The feedback that was given didn't give us recommendations on changes to make so we didn't make any changes since our ER diagram and tables.

5. Functionalities we added/removed

As mentioned above, a major functionality we removed was allowing the user to insert into any of the six tables of our schema, as a result of dependencies among records between tables. As a result, we weren't able to explore our original what-if idea in its entirety. We addressed this limitation by crafting a trigger and stored procedure that interacted with the other tables.

On the frontend, we removed interactive graphs from our application, as a consequence of our time constraint and not being able to figure out an efficient and meaningful way to complete this. Instead, we opted to incorporate our creative component, the wikipedia extension, which still provided a way for the user to explore and learn about the data present in our database. Further, we focused much of our efforts on making the user interface easy to use and added dropdowns to streamline input validation.

6. How our advanced database programs complement our application

We designed our two advanced queries to show an interesting subset of data from our database. We picked our queries to get data we believe most users would be interested in: countries whose gold medal count is higher than the avg gold medal count and coaches of athletes who play basketball or football.

The functionality of our stored procedure is to give underrepresented countries more of an equal chance in the Olympics and see how that affects the overall rankings. We implemented this by applying a bonus to the number of gold/silver/bronze medals that these smaller countries earned and calculating the country's new ranking as a result of this change.

The functionality of our trigger is to make the countries' rankings more dynamic so that any new country that is inserted has an accurate ranking. For instance, when the user inserts a new country into the NOC table, the trigger automatically inserts a new athlete into the Athlete table where the Athlete's country is equal to the newly inserted country. This is to ensure that every country is represented by at least one athlete.

7. Technical Challenges

Hansen – We face many challenges while implementing the creative component. The first challenge about this involved importing the python modules. People should have a clear mind about where the installed packages are and append the route to the package before trying to import. The second challenge is to create a backend that connects to wikipedia function, frontend input, and notification of chrome to display the output. Chrome.runtime.SendMessage and Chrome.runtime.onMessage.addListener are useful and we rely on these two functions to achieve the goal. The third challenge is to run the server. We spend lots of time figuring out the problems brought by version unmatches of the packages.

Maninder – One challenge we faced was displaying the data from the database. In our stage 4 demo, we were simply displaying a JSON object in a text box. While this did satisfy the requirement, it definitely could have been improved upon. So for stage 5, we found out how to use matching and the "usestate" variable in React to display the result in a structured and dynamic manner. This not only made our application look much better, but it also displayed the information in a much more readable manner.

Shloaka – Because the stored procedure inherently has a lot of components – including the declaration of variables, declaration of cursor and handler, loop, and final table – it took a lot of time to debug our issues. On top of that, we had to integrate 2 advanced queries within our stored procedure. To tackle this challenge, we looked at past notes and GAs to remind us of the basic

outline of a stored procedure. We also tested our 2 advanced queries separately and ensured they were delivering the correct output before implementing them in our procedure.

Emily – A major challenge on the frontend was connecting the user input to the backend with easy-to-use interface components. At first, we wanted to only use textboxes, which required the user to type important pieces of information that were not thoroughly validated on our end. Clearly, this was not a feasible way to set up the program. After a lot of trial and error, we arrived at using drop-downs. In hindsight, it would have been better to plan the UI beforehand and consider all the necessary edge cases for input validation.

8. Other components that changed

We created a wikipedia extension for users to easily search about the countries that show up in the results which they are not familiar with but interested in investigating. It is a chrome extension, which will show up when users click the right-up corner's extension button. They can type in the country they are interested in into the search box and the summary of the country's wikipedia page will show up seconds later, in the form of chrome notification.

9. Future work that our application can improve on

One thing our application can definitely work on is optimizing the CRU application. In our application we allow users to only insert into a specific table since the data dependencies would arise if we allowed them to insert into other tables. Not to mention NOC is the only table that does not depend on another table unlike the others. For update we correctly allow them to update one part of a DB entry using drop down menus however we would want to give the choice to pick an entry from a list rather than using dropdowns to get a specific entry. It is the same thing for the delete since we heavily rely on dropdowns to make the user narrow down to one entry but we want to give them the choice to delete from a lot of entries.

Another improvement we would like to make would be to make the ranking and weighted ranking dynamic. As it stands now we only change the ranking of a new insert rather than going back to the others. We would make all the ranking changes from the insertion of a new entry. We would also want to apply the same idea to the update and delete since these two would also cause the ranking to shift drastically.

One final improvement we would want to make would be to include visualization. In our original idea we wanted to include various visualizations but we didn't have the time to do so. So we would want to do this as an improvement. Since seeing a visualization of the data reveals a lot more than just displaying records. Not to mention it would help it look a little nicer. 🙂

10. Division of labor and teamwork

We all worked together for every part of the project, and so we were all aware of what changes were being made at each step. We made sure each team member had a good understanding before moving to the next component which allowed us to learn from one another and develop a healthy group dynamic.