CS 370 - Database Management Systems
Project Report

FREE TO PLAY - TOURNAMENT ORGANIZER

April 29th, 2024

Isaac, Scott, and Winter
Bradley University

1 Project Proposal

Project Title: Tourngonizer (Gaming Tournament Organizer)
Objective: Our goal is to design a gaming tournament organizer using databases. Gaming tournaments are extremely common as video games tend to be online and have a massive amount of people who want to play. Games like Counter-strike get tens of millions of users, and in those games there are modes like ranked to prove one's skills. We want to apply our knowledge to design a tournament organizer that pits players together and has a winner emerge.

Project Description: The features will include the ability to edit and change the bracket if things go awry, the ability to register an account, keep the score, and schedule the time of the tournament.

Bare Necessities:

- Creation of a user/guest account. This includes the ability to have and manage an account
- Ability to manage or edit the bracket if the need arises (admin / player roles)
- Include the ability to schedule the tournament (date and time feature must be included)
- Pairing feature to match two people against each other
- Feature to keep track of your wins and losses in the tournament
- Ability to name a tournament and share it with other people so they can join

Extra Features:

- Record data on user IDs over multiple tournaments
- Different types of tournaments, double elim, round robin, etc.
- Ability to pair more than one player in a bracket node
- Ability to make a tournament for a specific game (ex: Preset for CS:go tourney vs Smash tourney)
- Ability to customize what the tournament page will look like

Data Source: Our two data sources will include user input, and the second will be the results that are generated inside of the tournament. The first source will take into account the username and password of an account. This will also include tournament inputs for things such as the amount of registers, the name of the tournament, and the date of the tournament. The last thing that the user will need to enter is the records of the tournament, such as winning, losing, and disqualifying. The data we generate in the back end will be the wins and losses recorded throughout the tournament. This will then be used to decide where the player will proceed. User input is the most important part of what

Front end: For the front-end side of things, we will be using HTML, CSS, and JavaScript to build the website page. We will be using frameworks such as react to help with the development of the front page. We will use these to build a multi-layered website that has different tabs for users, tournament pages, and is simple to use and operate for the user on the other end.

Back end: For the back-end, we will be using tools such as MySQL for making the databases and looking into frameworks such as Django for working with the databases and having them interact with the front end side of things. We want to store everything here, from tournament and user IDs to the wins and losses in the tournaments. We will be using the info from the databases to store information about the tournaments and users safely.

2 Database Conceptual Design

Figure 3 Our Database has four entities in it. The player, the Tournament, the Match, and the Organizer. To start, the player has an ID, username, and display name. These are all separate and will be displayed for them to see. The seed, login, and password will be for the individual to see besides the seed. The seed will determine where the player is placed in the tournament and will determine how skilled the player is. A low seed means they are more likely to get first place. A high seed means that they are more likely to lose. The ID is the primary key for the Player.

The next block is the tournament, which has an ID, name, and date. The tournament is meant to hold match data. The ID is for the system to recognize the tournament. The name is for the players to recognize and will be shown on the front end. The date is to determine when the tournament will start. The Organizer has a name and an ID, where the ID is the primary key and the name is for the player recognizing them. They are given the ability to make a tournament and set the date on it. They can set the Name as well.

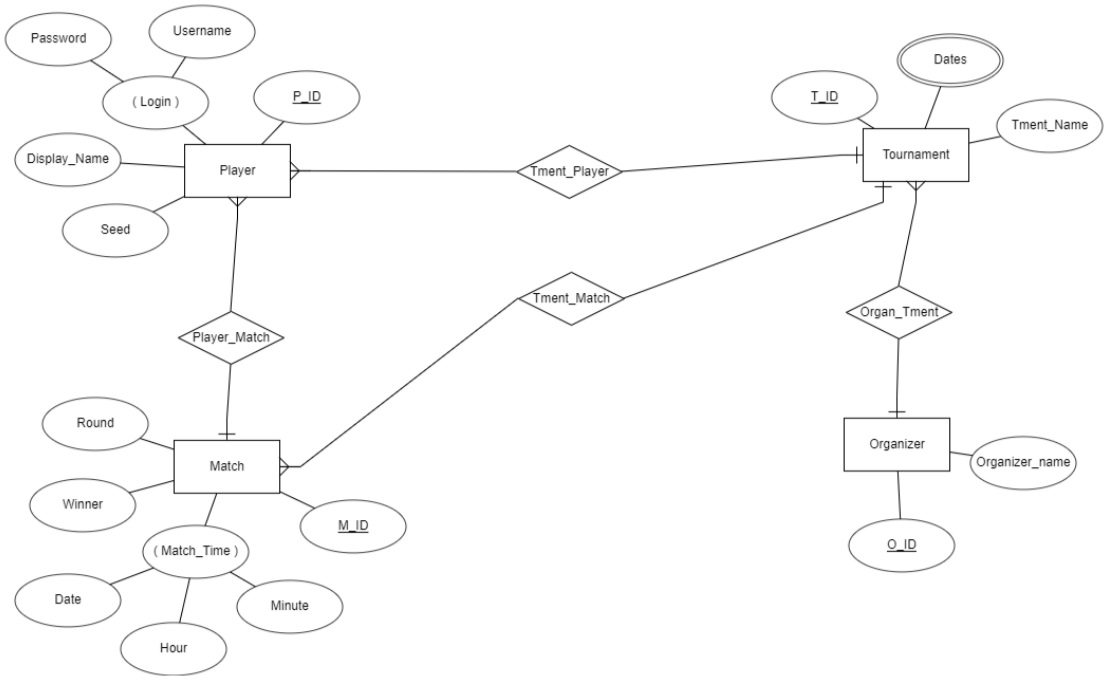

Figure 1: Our ER diagram for our project

The Match is the last entity in our project. It will hold an ID as the primary key, as well as the rounds and winner of the tourney. It will also hold the Match time, which is a multi-valued attribute. It will have the minutes and hours, as well as the date. These will be used to tell the player when the match time is set to happen. This is important if we were to add features like disqualifying the player if they do not attend the match.

The relations between them are the Player Match, the Tournament Player, the Tournament Match, and the Organizer Tournament. These will hold the IDs of aforementioned entities. As an example, the Player Match will hold the Player and Match ID. The important relations are that the Player has a many to one relationship with the Match. Match has the same with the Tournament. The player has a many to one with the Tournament, and Tournament has the same with the Organizer.

As for the Notation of the diagram, the squares are entities, the circles are attributes. The underlined text in the circles are primary keys, and the text in parentheses are composite attributes. The Double circle is the multi valued attribute. The arrow on the entity means many, and the line on the entity means one. The relationship is the diamond between the line.

For the relationship of the diagram, two good examples would be the player to matches and the player to tournament. The player will register for a tournament after seeing their name and assuring the date will work for them. Then the player will be moved to the next relationship of the match. The match will take the player data from those who are registered for the tournament. The match will then display when the player match is taking place, as well as the round the player is on and the winner of the match. It will then move the winning player up to the next match.

2.1 ER to Relations

Figure 3 To convert Our diagram over to a relational schema, we first start out with

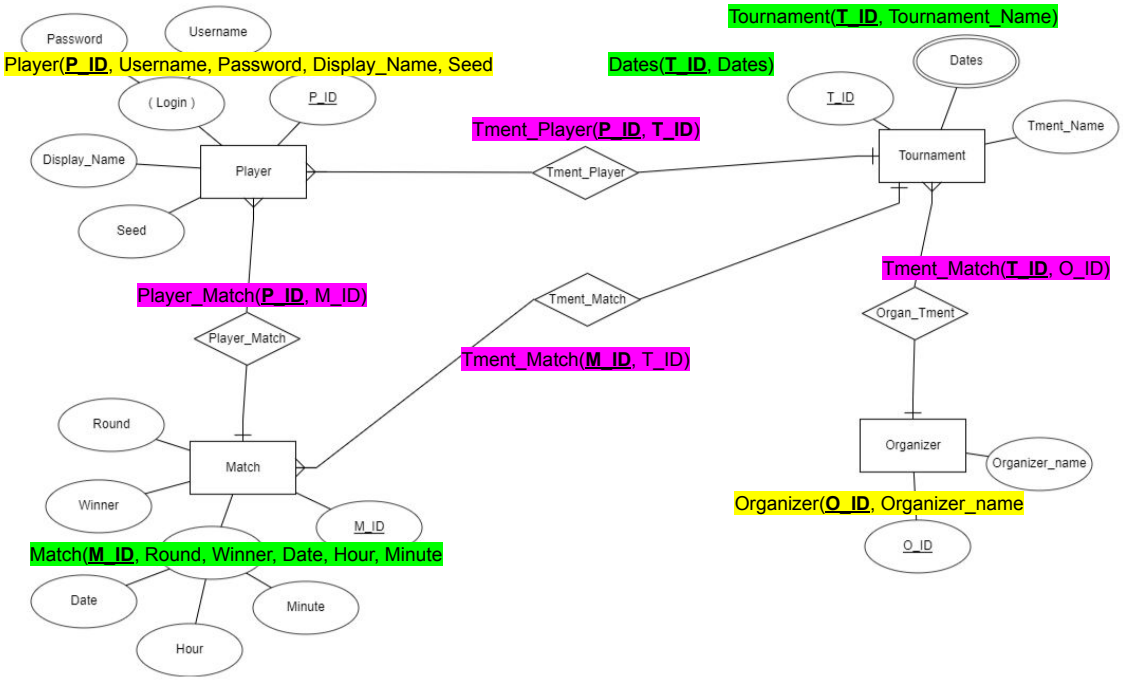

Figure 2: Our ER diagram for our project

grouping the entities with their attributes. All of the relations are weak with a few complex

attributes. Those with complex attributes are labeled with Green, and the weak ones are labeled in yellow. The next step is to label the relationships between the entities. These hold the IDs of the entities in communication, and the many entity has the primary key over the one key. These relationships are labeled in pink.

Figure 3 To finish the conversion, we need to combine groups to get rid of redundant

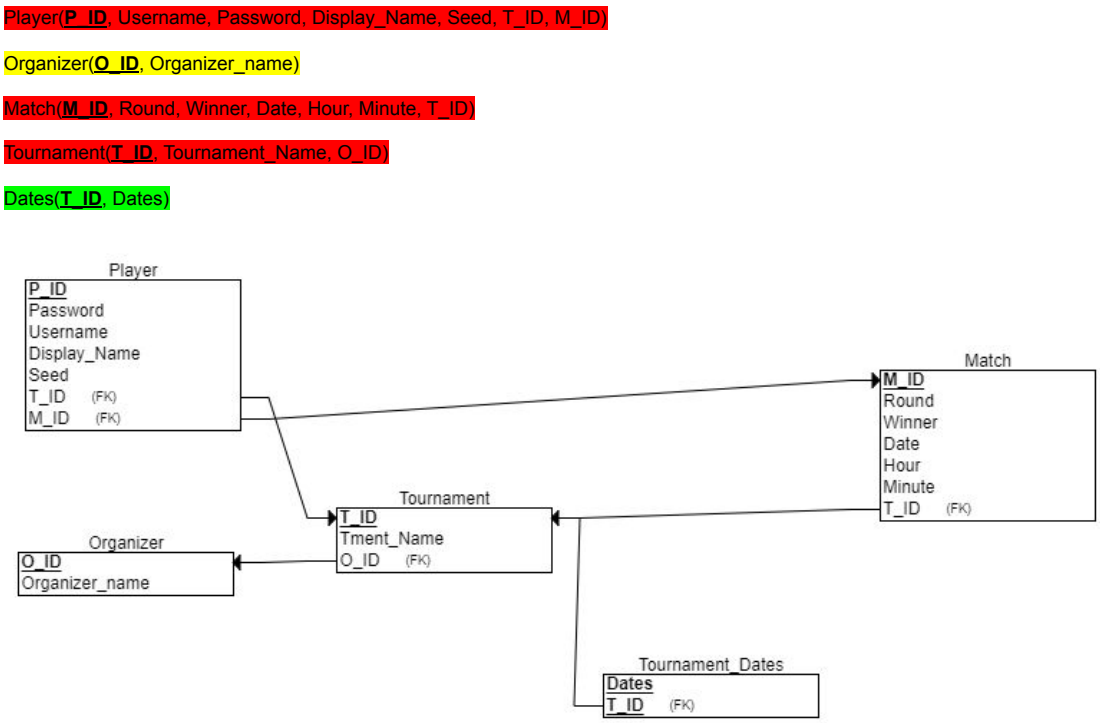

Figure 3: Our Relational Schema diagram for our project

data. We do this by taking the relationships and applying them to the many element of the schema. This means as an example that the tournament and match ids go to the player. After these are sorted, all that needs to happen is making the table for each entity and adding the arrows to represent the relationships in the database.

3 Normalization

For Normalization, we made sure that our data is already at level three. In the process of making the database schema, we went through quite a bit of redundancy of data, such as having matches hold a player one and a player two. We found that we could just take from the player table to simplify the table. Level one was removing repeating groups which we did not have in our first draft. The second step was removing partial dependency which we did when we removed the players from matches. The last thing we needed to do was remove transitive dependency, which we didn't have by the time we simplified it. Everything is able to depend on itself and calls other tables when needed. As such, our database is normalized.

4 SQL Implementation

Our SQL Implementations Include:

1. An authorization for organizer password and username. This allows the organizer to login to the page only allowed to them

2. the player is given a view of the database when they enter the website, acting as a guest account

3. The Organizer can insert:
    a) a tournament for players to enter
    b) a player with a unique ID and username
    c) a match as well as the time and date for it

4. The Organizer can delete:
    a) a tournament and removes the players assigned to it
    b) a player without having to delete the tournament
    c) a match and removes the players from the match assigned to it

5. A select for the organizer to view tournaments after creating or deleting them

6. Natural Join combines Player and M ID so the players are assigned a match

*Note: we are considering subsections to be unique SQL Injections

5 Project Application Conversion

To convert our project into a workable application, we would want to finish the features we set out to complete in the original project proposal. Due to time constrains and knowledge, we were forced to reign back some of the features presented. The first thing we would want to do are to complete the bare necessitates. This would include the ability to create a guest account and a user page for those logged in. The other feature we will need is the ability to edit a bracket if the need arises. This would be done for the tournament organizer account, as they should be the only ones allowed to edit the tournament. We need to add a way to schedule tournaments in a way that people could search for them that way, and add a search function in general for the tournament. The last few things we would need to do is keep a history of wins and losses for proof of placement in the tournament. We also need to add the ability to share around the tournament join link so people can join it. Overall, if we deployed the website we would need to flesh out the features that we have already completed. We would also add some security functions to the website as well to make sure that accounts could not be accessed and there is no way injections that we wouldn't want don't open up as we expand the website.

Some other features that we would need to be able to convert this into a working website is to add some of the extra features. As mentioned before we not only want to keep placement history for a single tournament, but adding a tournament history for the account would be a nice feature for the user to have. Then we would want to add the ability to change the tournament type. So far we only have player matching, but there are types of tournaments like round robin or team battles that are better to run with a small bracket size. We also would want to configure the tournament to work with certain games, this would be mostly for stylizing, but it would go a long way for the user to use the bracket. The last thing we would want to add is the ability to customize what the profiles for everything can look like, such as profile pictures for players and banners for tournaments. All of these features would help the project go from an assignment to actual software that could be used and presented.