Requirement Analysis PolSim

Table of Contents:

Project Description

Our project is a politics simulator using a database, that is flexible enough to simulate many different systems using a simple table structure. We named it PolSim as it’s a nice contraction of Politics and Simulator. This simulator lets its participants create groups with each other and share their political power. It is also possible to take over and become a tyrannical dictator. Finally, if everybody has a "government" they’re happy with, we can poll the whole system to see how the decision-making process plays out on a global scale. The users get to feel what it’s like to live under various ruling systems. From pure, direct democracy all the way to a glorious, absolute monarchy.

The way it works is: To start with, every user is simply a member of "the world". Everybody in the world has the same amount of political power and can create votes. If some people decide to create a "country" they can form a group. Let’s say — for example — The UK. These people will start off with a direct democracy. Everybody in the group has equal decision power and if a vote is started, everybody’s vote counts the same. Except the group leader who has slightly more power, in order to avoid ties. However, the people of this United Kingdom might decide to create a parliament, and so create a group called "Parliament". Then they make an agreement with the people. If the people all give their political power to the parliament for a fixed term, then the people get to chose who is elected. If the majority agrees, then they all surrender their power to the parliament and any decisions made for all of the country is handled by the parliament alone.

These groups can be nested as deep as the user wishes. You could also have political parties within the groups as their own smaller groups. Inside those parties you could set up factions or conspiracies. This also applies in the opposite direction. If you already have some pseudo-countries set up you could join together and form grand alliances like the united nations.

Requirements

Functional Requirements

- Register/Login
- Create groups
- Join/leave groups
- Change group leader
- Remove someone from group
- Transfer political power
- Organise revolution

Non-Functional Requirements

- Database is in the 5th normal form
- All errors are handled correctly

Interview

Preparation

For the customer interview, we wanted to get as many specifications as possible, given the complexity and scale of this concept. We decided to ask some of the more specific questions we were wondering about how the system should allow many different systems to be included.

We discussed together which questions would be most important in order to improve the quality and decrease the overall quantity of questions we had to berate the customer with.

Our goals for the interview were to:

- Have a better understanding of the general idea behind the project.
- Specify exactly how the system should work and agree with the customer on the specification.
- Answer some questions about unclear details.

Interview Details:

We decided to meet at TBZ on tuesday for the interview. Samuel will lead the interview. All members of ou team would be present for the interview should any last-minute questions arise.

Protocol

Question 1:

Question:

How should the many layers of disparate government systems be represented and implemented?

Answer:

The people should be able to create groups. Every person can only be in a single group at any given time. However, these groups can be nested inside each other. For example: A party is a group in the parliament which is inside the country which is inside an alliance, et cetera.

Question 2:

Question:

How can people vote on the topics? How are the votes counted and processed?

Answer:

Each person has a percentage of the total available political power in each layer. groups can also have a percentage of the power. For example if a group of people join a group, all the power they owned in the layer above is given to the group as a whole and inside the group each person holds an equal share of the power of that layer.

Question 3:

Question:

What should be done, if anything, to avoid ties?

Answer:

Whoever originally creates a group becomes leader and their decision is worth slightly more than the the others. But to avoid tyrrany, who becomes the leader is voted on by everyone in the group.

Question 4:

Question:

What kind of actions should the people be able to vote on and perform?

Answer:

As long as the people have voting power, they can:

- Create new groups
- Add person to a group
- Remove person from a group
- Transfer political power
- Change the current group leader
- Revolt against the government

Question 5:

Question:

How are revolutions handled? Can anybody start one?

Answer:

Anybody, no matter whether they have political power or not, can start a vote for revolution. Everybody in the affected group who votes in favour of the revolution joins the revolution and all those against join the defense. Each pair of one revolutionary and defender have "battles". In these battles each side has a 50% chance of winning the battle. this continues until one side has no more participators. This introduces an element of luck to the revolts while simultaneously eliminating the possibility of a tie.

Question 6:

Question:

How are the different layers differentiated?

Answer:

There are three main kinds of votes: Local votes are votes that involve all the people and groups inside the current group. Global votes are votes that involve all the people and groups in the group one layer above the current group. Universal votes are votes that involve all the people and groups in the entire system.

Entity Relationship Model

[Entity Relationship model] | *ERM_153.png*

Entity Descriptions

Person

Naturally, we will need an entity to store information about each user. This entity will have the ability to join groups and vote on decisions in that group. Each person also requires a username that is unique.

Attributes:

- Username: unique title for referencing users
- Group: this is the group the person is in
- Can Vote: a boolean value whether this person has the right to vote
- Power: the amount of weight this one person's vote has in a decision

Group

This entity stores all the information of the hierarchical structure of the different governments. This entity is also self-referential, so that the groups can be nested inside themselves.

Attributes:

- Groupname: unique title for referencing groups
- Leader: person who created this group and also who breaks ties in voting
- Parent Group: the group this group is in
- Power: the amount of weight this one group's vote has in a decision

Vote

This records what every person votes on the decisions. It references the decision and the person who cast the vote, as well as whether the person agrees or disagrees to approving the decision.

Attributes:

- Person: the person casting this particular vote
- Group: the group casting this vote. Only one of these two can be set.
- Decision: the decision this person is deciding on
- Agreement: whether the person agrees to approve this decision.

Decision

Every decision that needs to be made, and has already been made, will be stored in this table. The outcome of this vote is gotten, by counting the number of unique and valid votes cast in favour of the decision.

Attributes:

- Title: a short description of the vote topic
- Creator: the person who initiated the decision
- Close Time: the time when voting is closed and the decision is either accepted or rejected.
- Voting: this is the reference to the group that will be affected by this change, and that get to vote on it's outcome.
- Create Date: the date that the decision was created on.
- Last Vote: the timestamp of the latest vote and therefore last change to the outcome

Action:

When a decision is approved it can perform any number of actions. An action is a group of SQL statements that get performed on the database. For example, if you want to change the leader of a group, that would be put in the database.

Attributes:

- Decision: the decision that will approve or deny that this action can pass
- Description: a short explaination of what this action will perform upon being approved.
- Exec Time: the time when this action is performed if it has been approved

Command:

Actions constitute groups of commands to be executed. You can't just create a group, you must also add all the potential members to the group as well. All those commands are performed in a single action, simply called "Create group".

Attributes:

- Action: the action that would perform this command
- SQL Command: the SQL code to be executed with this command with its arguments substituted

Argument:

Commands can have varying numbers and types of arguments, so this table contains the value of all the arguments to be passed to an SQL command. Luckily all the commands we need to be able to apply to the databse have only integer arguments, so we don't need to store the data type.

Attributes:

- Command: the command that accepts this argument
- Value: the integer value to be passed
- Position: the order in which the arguments should be inserted into the commands

DBMS Decision

IMPORTANT	Insert DBMS Decision matrix here

Specifications

Decision Method

Decisions are approved or denied by a majority vote. Votes are either on the local, global or universal level. Local means all people and groups within the current group. Global means all the people and groups within the group one layer above the current level. Universal means all the people and groups in "the world". "The World" is the highest level group that contains all people and groups in the current simulation. Each group and person has a certain amount of political power on their level. Decisions are scaled based on the voters political power. Take for example this system:

```
The World (100%)

    Switzerland (50%)
        Bill  (25%)
        David (25%)
        James (25%)
        Bob   (25%)

    United Kingdom (50%)
        Parliament   (100%)
            Boris      (33%)
            Nigel      (33%)
            Jeremy     (33%)
        Jim      (0%)
        Jeremiah (0%)
```

The political power of each person and group is written in parentheses "()". Jim and Jeremiah's votes won't be counted at all, as they aren't in the parliament. Boris, Nigel and Jeremy's votes each count for a third of the result. Bill, David, James and bob live in a direct democracy and each have one quarter of the decision power of switzerland. Both countries hold half of all power in the world.

Action Method

In order to keep the database a simple, flexible and future-proof as possible, there is chain of tables to run actions and allow them to be completely configurable.

The action table is what is easily visible externally as an "action". (e.g. Create group with set members) The action itself essentially contains a list of SQL commands to be executed on the server to change the system as intended by the person who created this action.

Each command is simply a single SQL command with all necessary arguments substituted by a question-mark (?). This allows the commands to still be flexible and able to have as many arguments as are necessary.

The arguments that are provided to the commands are stored in a seperate table called "argumen". This table holds the type and value information for one of the arguments to be used by the commands.

Naming Convention

Table Names: `<tblname>, lowercase`
Primary Key: `ID_<tblname>, lowercase`
Foreign Key: `<name>_<reftable>_ID, lowercase`
Column Names: `<colname>, lowercase`
View Names: `v_<name>, lowercase`
Index Names: `idx_<name>, lowercase`
Trigger Names: `trg_<name>, lowercase`
Stored Procedure Names: `stp_<name>, lowercase`