Icon

Description automatically generated

A red moon in the sky

Description automatically generated with low confidence

TIC2601 Database & Web Applications

Team 4 (Topic D):

Cai Xincheng

Chong Kean Siong

Poh Yi Tat

Qian Zijing

Wang Pengcheng

Contents

[1. Introduction 4](#_Toc116500211)

[1.1 View 4](#_Toc116500212)

[1.2 Search 4](#_Toc116500213)

[1.3 Post and Comment 4](#_Toc116500214)

[1.4 Upvote / Downvote and Reputation 4](#_Toc116500215)

[1.5 Administrator 4](#_Toc116500216)

[2. Technology Stack 5](#_Toc116500217)

[2.1 Architecture Overview 5](#_Toc116500218)

[2.2 Tools and Languages 5](#_Toc116500219)

[2.2.1 Web Server and Application Framework 5](#_Toc116500220)

[2.2.2 Language 5](#_Toc116500221)

[2.2.3 Database Management System 5](#_Toc116500222)

[3. Entity Relationship 6](#_Toc116500223)

[3.1 Entities 6](#_Toc116500224)

[3.2 Relationship 6](#_Toc116500225)

[3.3 ER Diagram 7](#_Toc116500226)

[4. Relational Schema 8](#_Toc116500227)

[4.1 user 8](#_Toc116500228)

[4.2 post 8](#_Toc116500229)

[4.3 comment 8](#_Toc116500230)

[4.4 category 8](#_Toc116500231)

[4.5 subscribe 8](#_Toc116500232)

[4.6 vote\_post 9](#_Toc116500233)

[4.7 vote\_comment 9](#_Toc116500234)

[5. Sample SQL Statements 10](#_Toc116500235)

[6. Web interface and project highlights 11](#_Toc116500236)

# Introduction

Rabbit is a social news platform that enable users to share information categorised by topics and interact with each other.

Below lists some of the majors features of Rabbit:

## View

All site visitors are able to view the posts and comments.

## Search

All site visitors are able to search posts with one or a combination of the below criteria:

* key words that appearing in post
* topic category

## Post and Comment

Registered users are able to post new contents and comments after logging in. They are also able to edit their posts after original post.

## Upvote / Downvote and Reputation

Registered users are able to upvote or downvote posts and comments. A net score will be computed based on the number of upvotes and downvotes received for each of the posts and comments which affects not only the visibility of the comments and contents, but also the reputation of their owners.

## Administrator

Administrators of Rabbit are able to perform create, read, update and delete operations on all data entries. They are also able to view useful statistics such as topics in trending in terms of total number of votes received, clickstream, etc.

# Technology Stack

## Architecture Overview

**Web Browser**

**Web Application**



**Database Connector**

Logo, icon

Description automatically generated

**Database**



## Tools and Languages

## Web Server and Application Framework

Web server: NodeJS

Application framework: Express

## Language

Client side: HTML, CSS, Javascript

Server-side: Javascript

## Database Management System

MySQL is used as the database management system for Rabbit.

# Entity Relationship

## Entities

There are 4 entities in our database schema:

* user – user profile
* post – post content
* comment – comment content of the post
* category – category list to group posts

## Relationship

* user – write – post
* user – write – comment of the post
* user – subscribe – category
* user – vote – post
* user – vote – comment of the post
* comment – of – comment
* comment – of – post
* post – have – category

## ER Diagram

(0,n)

(0,n)

(0,n)

(0,n)

(0,n)

(0,n)

(1,1)

(1,1)

(1,1)

(1,1)

(0,n)

(0,n)

username

user

password

is\_admin

created\_at

reputation

subscribe

have

of

is\_upvote

post\_id

content

header

vote

category

category

comment\_id

post

comment

write

created\_at

updated\_at

parent\_comment\_id

content

updated\_at

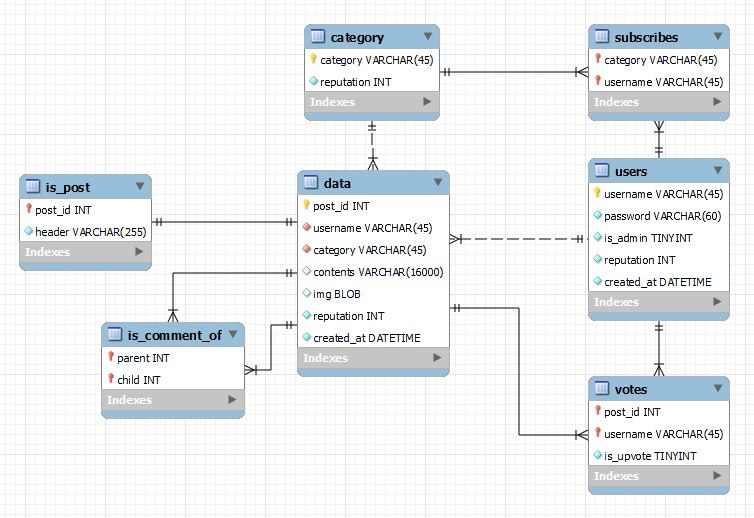
(0,n)

of

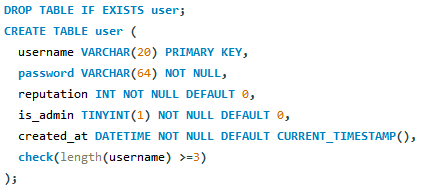
(1,1)

# Relational Schema

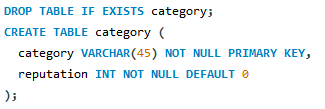
After optimizing our data requirements, we have ended with an ERD as shown below.



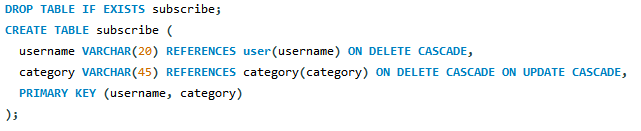
## user



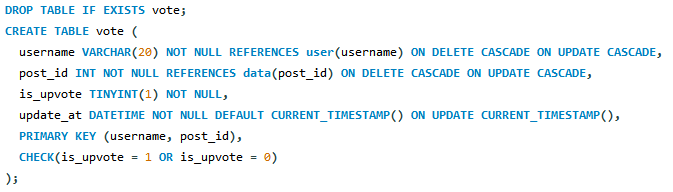
## category



## subscribe



## vote



And below are the tables where we consolidated the data that derived from post and comment into one.

## data

Graphical user interface, text, application

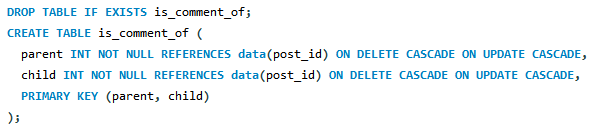
Description automatically generated

## post

Graphical user interface, text

Description automatically generated with medium confidence

## is\_comment\_of

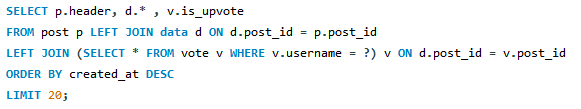


We have grouped the contents of “post” and “comment” in a single table as “post” and “comments” shares many other similarities. The difference will then be stored on a separate table called post which holds the information of “post” that “comment” does not have the attribute of, that includes the header of the post.

As a user should be allowed to infinitely reply to another comment, no matter the layer of comment it is to the post. The table “is\_comment\_of” is then used to record the hierarchy of the posts and comments.

# Sample SQL Statements

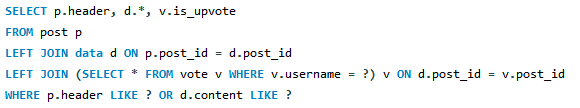
**Select all post**



Starting from the select statement from the index page, this query selects the latest 20 post using the “created\_at” datetime value from when the post was inserted into the database. “LIMIT 20” is used to limit the number of results to 20 to display to the user.

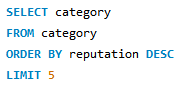
The left join statement in the 3rd line is used to check for the vote status if a user have previously voted on that post and displays them back to the user for their interaction. Through this, if a user has previously voted on a post, it would return a true or false value and if unvoted, it would return a null value. This line would appear frequently for the other select statements as well to check the interactions the user had previously with the post or comment.

**Search post**



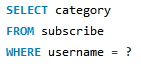
This select statement is for the search bar for the user to search contents that is similar to the input which the user had entered. However, this statement limits to only posts to be able to be searched up instead of post and comments. If we would want to include comments into the results of the search, we could simply swap the order of the left join of the post and data table and user post as the left join to retrieve the header if required.

**Select all categories**



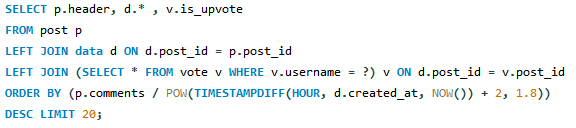
This select statement is used to retrieve the top 5 categories to be displayed in the index page.

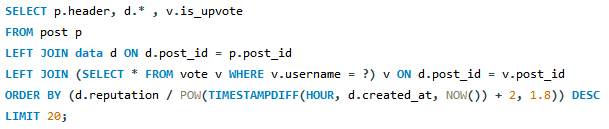
**Select all subscription**



This select statement is used to retrieve the subscribed categories the user has after they have logged in.

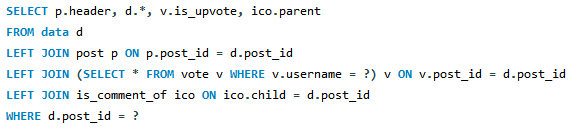
**Select by hot / trending**





The above 2 select statements are responsible for ranking the post for hot and trending sections in our website. We used a formula to rank the post based on their votes or comment interactions across the time it is generated. Using the number of comments in the post for hot and number of upvotes for trending, we then divide that by a time formula that gives it a score that would drastically decay after the 1 hour mark.

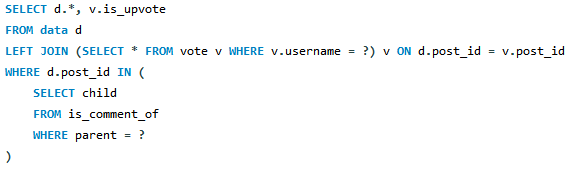
**Select post information**



This query is used to retrieve information on the post page. The post page will be able to display if it is a post or comment and will hide the header if no information of the header is retrieved, hence making this a universal query no matter the post\_id is a post or a comment.

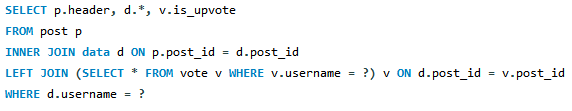
Another line is then added to retrieve information of the parent post if there is one, and the hyperlink of the back button will then redirect the user to the parent post or the index page if it is a post.

**Select replies**



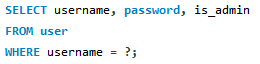
This select statement is used to retrieve the next layer of replies of a particular post.

**Select user’s posts**



This select statement is used to retrieve any post related to a particular user to be displayed in the user page.

**Login**



This statement is use to retrieve the user information when logged in, and the retrieved password hash will be used to compare with the inputted password’s hash value and if they are matching, the user information will be saved into the session of the webpage.

**Create, Edit & Delete**

The following sql are all create, edit and delete statements that are consolidated in a single js file named “crud.js”. These statements will only be run after checking for the permissions of the user in the script.

**Create post**





These two queries are used to insert a new post into the database. The first queries will insert the information into the data table through a json object, and through this, the returned value would be the newly inserted post\_id. And with that we would be able to insert the header information into the post table with the corresponding post\_id.

**Edit post**







This section consists of 3 sql statements, it is used to insert the information of replies of user to a post or comment. We would first retrieve the category of the replied post to be inserted into the comment. And the remaining queries work the same way as post, but this time we insert a record of the parent and the newly created record.

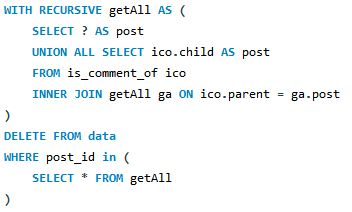






This part would be the section for editing posts, after the verification of the user matching with the username saved or the user is an admin, the content will be updated to the newly edited content. And the 3rd line will only be run if the result came back from the 1st query that it is a post and will update the header information.

**Delete post**



This query is used for the deletion of a post by the user or admin. As the “is\_comment\_of” table is a table matched with data table with both parent and child columns using post\_id as a foreign key, we would not be able to use “ON DELETE CASCADE” function to automatically delete when the key is deleted.

Hence, it is necessary to manually delete the comment data manually through this recursive function. This recursive function will go through the “is\_comment\_of” table and retrieve all layers of comments that belongs to the particular post or comment and deletes them together.

**Create, Edit & Delete vote**



This part will be run if a logged in user clicks on a vote button, this line will first check their existing vote of the particular post and return the value. And depending on the results one of the next three queries will be ran.







The delete query will be ran if it matches with the returned value to clears the record of the vote of the user. The update query will run if a different value was returned. And finally the insert query will run if the return value was null.





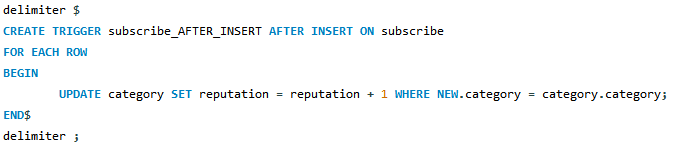
And after that, the post and the poster reputation will be changed accordingly. A value of -1 will be used if it’s a downvote, +1 for upvote and a +- 2 will be used when the update function is used.

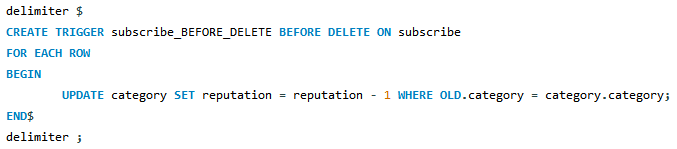
**Create & Delete subscription**

****

****

The following queries are used to record the user’s subscription interaction to a category.

****

****

The handling of the category’s reputation is a little different from the others where we used trigger to update its values when there is an insert or delete query is ran.

**Create user**





The application will first check if the username is used before allowing the insertion of the user account into the database.



This query will allow the user to update their username or password.



This query will allow the user to delete their account from the database.

****

This query allows an admin to change another account to an admin account.

The user crud functions mostly consist of verification on the JavaScript to check if the user is allowed to perform the above actions.

# Web interface and project highlights