Udiddit, A Social News Aggregator (Part I)

REVIEW

HISTORY

Meets Specifications

Kudos 🎉

It was a great submission 🏆
You nailed it !!

Keep up the good work :)
Happy Learning

SQL Syntax

All SQL code submitted should be properly syntax highlighted in order to make it easier to read.

See this StackOverflow post for some tips on how to do so.

Great work here! Your code has relevant and the appropriate syntax highlighting.

Any SQL query that is more than about 80 characters should be multi-lined at the appropriate places (FROM,

JOIN, WHERE, etc.) and indented to make it easier to read.

Good codding skills.
Really well written and organized queries

Analysis of Provided (“bad”) Data Schema

At least three flaws related to data modeling best-practices on the provided data schema are noted.

The analysis of the provided data schema should contain bullet-point recommendations instead of only pointing out shortcomings. For example, instead of “such column shouldn’t be such data type”, you should say “such column should be this data type and have an index added to it”.

Nice work! Your suggestions are relevant and well communicated.

Data Modeling and DDL

Copy/pasting each DDL query in the workspace should produce a new table or index, and run without errors.

Great job! Your DDL statements run without errors when pasted "as-is".

While the data model is allowed to contain extra fields not required to satisfy the features requested, like for example the addition of some audit fields, the data model should at least be able to answer all the features and queries outlined.

The data model provided should satisfy first, second, and third normal forms.

Well done for designing your schema to conform to first, second, and third normal forms.

The data model provided should not allow any non-existent IDs to be inserted in related tables.

Good job for enforcing referential integrity, so that non-existent IDs are not inserted into created tables.

The data model provided should have appropriate indexes in order for the queries in guidelines 1 and 2 to run quickly and efficiently. Note as some constraints automatically add indexes, it's important to make sure that you don't duplicate any indexes.

- Each username has to be unique
- Usernames can be composed of at most 25 characters
- Usernames can't be empty
- We won't worry about user passwords for this project

- Topic names have to be unique.
- The topic's name is at most 30 characters
- The topic's name can't be empty
- Topics can have an optional description of at most 500 characters.

- Posts have a required title of at most 100 characters
- The title of a post can't be empty.
- Posts should contain either a URL or a text content, but not both.
- If a topic gets deleted, all the posts associated with it should be automatically deleted too.
- If the user who created the post gets deleted, then the post will remain, but it will become dissociated from that user.

- A comment's text content can't be empty.
- Contrary to the current linear comments, the new structure should allow comment threads at arbitrary levels.
- If a post gets deleted, all comments associated with it should be automatically deleted too.
- If the user who created the comment gets deleted, then the comment will remain, but it will become dissociated from that user.
- If a comment gets deleted, then all its descendants in the thread structure should be automatically deleted too.

Well done for completing this, and especially for implementing the comment thread structure.

- A user can only cast one vote on a given post.
- If the user who cast a vote gets deleted, then all their votes will remain, but will become dissociated from the user

from the user.

- If a post gets deleted, then all the votes for that post should be automatically deleted too.

Data Migration and DML

The migrated data will be in normalized form, with each table having its own primary key, and related tables using IDs and foreign keys. While the database will verify that the related IDs are consistent, it can't actually verify that they're correct. Thus, the DML used to migrate the data needs to make sure that the right IDs are associated together, not just any valid ID.

Great job done here for ensuring IDs belong together, especially by using appropriate JOINS.

Since the data has to be migrated following a certain order and logic — some data has dependencies — it's important that the DML that is provided can be literally copy/pasted as-is, and run without errors.

Excellent! Your schema takes into account table dependencies, and hence, maintains the appropriate execution order for migrating data.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review