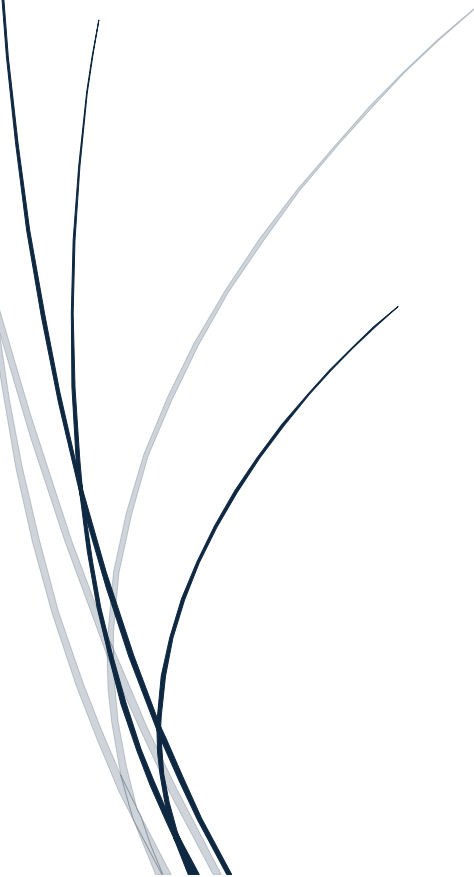




12/26/2024

CONCEPTUAL AND LOGICAL DESIGN OF A SOCIAL MEDIA ANALYTICAL PLATFORM



***Saifullah Imtiaz BSE233062
Ahmad Shehryar BSE221094
Ghazi Muhammad Qazi BSE231017***

1. Description of Entities, Attributes, and Relationships

I have designed a database for a Social Media Analytics Platform. The platform is intended to provide users with the ability to interact socially (through posts, comments, and connections) while allowing for the analysis of data such as user engagement and content popularity.

Entities:

1. Users

Attributes:

User ID (Primary Key, Integer)

Username (Varchar)

Email (Varchar)

Password Hash (Varchar)

Full Name (Varchar)

Profile Picture URL (Varchar)

Date Joined (Date)

Date Last Login (Date)

Status (Varchar) Active, Inactive, Banned

2. Posts

Attributes:

Post ID (Primary Key, Integer)

User ID (Foreign Key, Integer)

Content (Text)

Date Created (Date Time)

Likes Count (Integer)

Shares Count (Integer)

Comments Count (Integer)

Visibility (Varchar) Public, Private, Friends Only

3. Comments

Attributes:

Comment ID (Primary Key, Integer)

Post ID (Foreign Key, Integer)

User ID (Foreign Key, Integer)

Content (Text)

Date Created (Date Time)

4. Connections

Attributes:

Connection ID (Primary Key, Integer)

UserID1 (Foreign Key, Integer)

UserID2 (Foreign Key, Integer)

Connection Status (Varchar) Pending, Accepted, Blocked, Removed

Date Connected (Date)

5. Analytics

Attributes:

Analytics ID (Primary Key, Integer)

Post ID (Foreign Key, Integer)

Likes Count (Integer)

Shares Count (Integer)

Comments Count (Integer)

Engagement Rate (Decimal)

Date Collected (Date Time)

Relations:

Users can make Posts (one-to-many relationship).

Users may comment on Posts (many-to-one relationship).

Users can connect with each other as Connections (many-to-many relation, expressed through the connections table).

Posts can have Comments, Likes, Shares, and Analytics.

Posts are related to its Analytics, which consists of analytics records tracking engagement data for a post over the time period.

2. Issues or Concerns with the Design Process

a. Data Redundancy and Normalization:

A big issue was reducing redundancy of data yet making access to that information as simple as possible. For instance, engagement metrics (likes, shares, comments) were originally placed directly in the Posts table. However, it became evident that these metrics should be split into the Analytics table to facilitate easier querying and easier updates as data is accrued over time. This caused further normalization and prevented repetition of engagement data unnecessarily.

b. Handling Many-to-Many Relationships:

The relationship between Users and Connections is a classic example of a many-to-many relationship, which requires a junction table. At first, I considered directly linking users through foreign keys, but the implementation of a Connections table made it easier to track multiple connection statuses (e.g., pending, accepted, blocked) and handle the dynamic nature of user relationships.

c. Dynamic User Data:

The nature of the platform is analytical in scope, so dealing with real-time dynamic data such as engagement (likes, shares, comments) posed the challenge. This demanded that there be a separate table called Analytics that would hold engagement data per post so that the system could track and update analytics over time without compromising the performance.

d. Data Integrity and Security:

Security was a key consideration, especially when handling sensitive user data such as passwords. For password storage, I decided to store a hashed version of the password (using algorithms like crypt) rather than the raw password. Additionally, user status and visibility rules in posts were carefully considered to ensure data consistency.

3. Defining Tables, Fields, and Data Types

Table Definitions:

1. User

SQL

```
CREATE TABLE Users (  
    User ID INT PRIMARY KEY,  
    Username VARCHAR (50) UNIQUE,  
    Email VARCHAR (255) UNIQUE,  
    Password Hash VARCHAR (255),  
    Full Name VARCHAR (255),  
    Profile Picture URL VARCHAR (255),  
    Date Joined DATE,  
    Date Last Login DATE,
```

Status VARCHAR (20)

);

2. Posts

SQL

CREATE TABLE Posts (

Post ID INT PRIMARY KEY,

User ID INT,

Content TEXT,

Date Created DATETIME,

Likes Count INT DEFAULT 0,

Shares Count INT DEFAULT 0,

Comments Count INT DEFAULT 0,

Visibility VARCHAR (20),

FOREIGN KEY (User ID) REFERENCES Users (User ID)

);

3. Comments

SQL

CREATE TABLE Comments (

Comment ID INT PRIMARY KEY,

Post ID INT,

User ID INT,

Content TEXT,

Date Created DATETIME,

FOREIGN KEY (Post ID) REFERENCES Posts (Post ID),

FOREIGN KEY (User ID) REFERENCES Users (User ID)

);

4. Connections

SQL

CREATE TABLE Connections (

Connection ID INT PRIMARY KEY,

UserID1 INT,

UserID2 INT,

Connection Status VARCHAR (20),

Date Connected DATE,

FOREIGN KEY (UserID1) REFERENCES Users (User ID),

FOREIGN KEY (UserID2) REFERENCES Users (User ID)

);

5. Analytics

SQL

CREATE TABLE Analytics (

Analytics ID INT PRIMARY KEY,

Post ID INT,

Likes Count INT DEFAULT 0,

Shares Count INT DEFAULT 0,

Comments Count INT DEFAULT 0,

Engagement Rate DECIMAL (5, 2),

Date Collected DATETIME,

FOREIGN KEY (Post ID) REFERENCES Posts (Post ID)

);

Data Types:

INT is used for identifiers and counts like User ID, Post ID, Comment ID, etc.

VARCHAR is used for textual data such as Username, Email, Full Name, etc.

TEXT is for longer content, such as the Content field in Posts and Comments.

DATE and DATETIME are for date and time data that will be collected when users register, when posts are created, and analytics data is collected.

DECIMAL is used for calculated values like Engagement Rate.

4. Relationships Between Tables

One-to-Many:

A User can create many Posts (User ID in Posts table).

A User can comment on many different posts (User ID in Comments table).

A Post can have many comments (Post ID in Comments table).

Many-to-Many:

A User can be connected to many other Users, and so vice versa. This is reflected in the Connections table by the presence of the fields UserID1 and UserID2 which point to the Users table.

One-to-One:

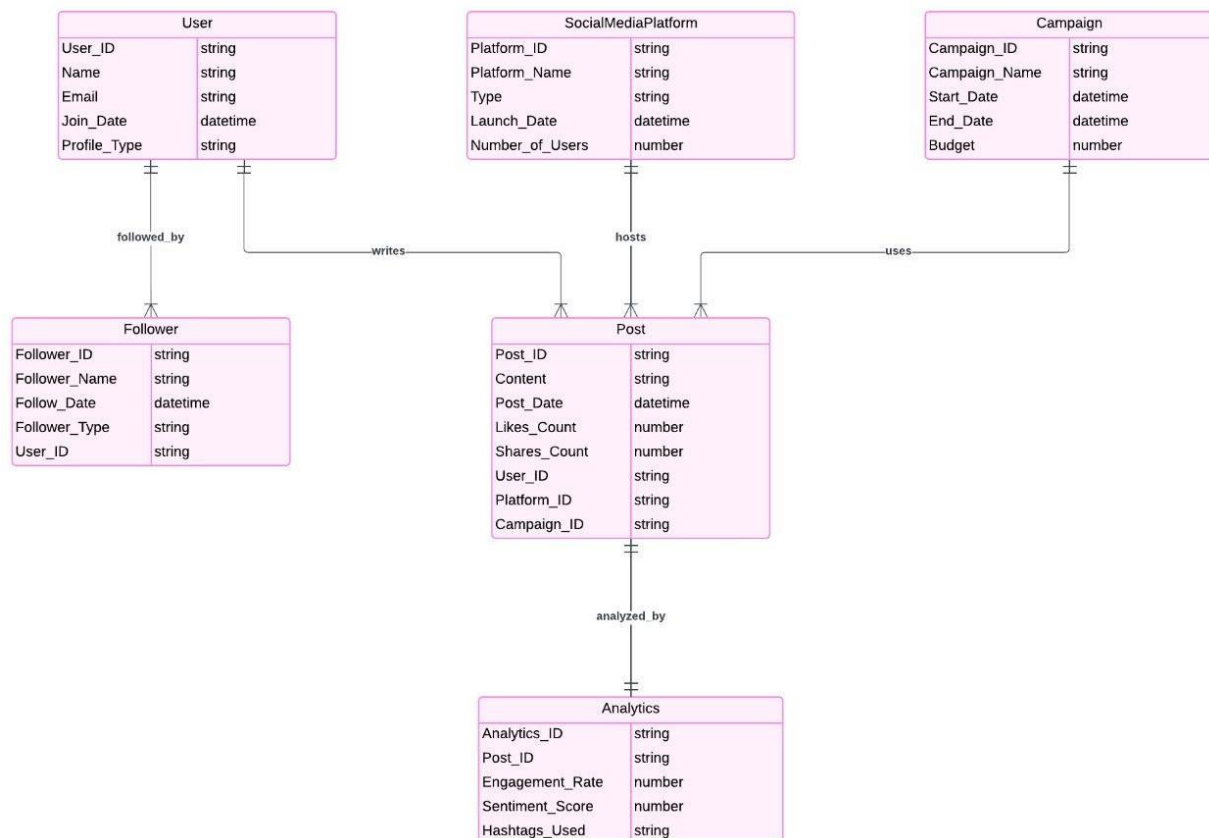
Each Post will have a corresponding entry in the Analytics table, where the Post ID in Analytics references the Post ID in the Posts table.

5. Converting the ERD to an RDM

The Entity-Relationship Diagram (ERD) for the Social Media Analytical Platform will depict the relationships described above. The transformation into the Relational Data Model (RDM) will be done by defining primary keys, foreign keys, and establishing clear relationships between entities using SQL tables. Every table in the RDM corresponds to an entity, and the relationships between them are captured through foreign keys.

The Posts table stores user generated content and engagement metrics, while the Analytics table tracks the performance of posts over time. The Connections table handles the dynamic relationships between users. By ensuring that each relationship is clearly defined, the RDM facilitates smooth data management and efficient querying.

ER:



RM:

User:

User_ID (Primary Key)

Name

Email

Join_Date

Profile_Type

Follower:

Follower_ID (Primary Key)

Follower_Name

Follow_Date

Follower_Type

User_ID (Foreign Key referencing User)

SocialMediaPlatform:

Platform_ID (Primary Key)

Platform_Name

Type

Launch_Date

Number_of_Users

Campaign:

Campaign_ID (Primary Key)

Campaign_Name

Start_Date

End_Date

Budget

Post:

Post_ID (Primary Key)

Content

Post_Date

Likes_Count

Shares_Count

User_ID (Foreign Key referencing User)

Platform_ID (Foreign Key referencing SocialMediaPlatform)

Campaign_ID (Foreign Key referencing Campaign)

Analytics:

Analytics_ID (Primary Key)

Post_ID (Foreign Key referencing Post)

Engagement_Rate

Sentiment_Score

Hashtags_Used

Relationships:

User "followed by" Follower: One-to-Many

User "writes" Post: One-to-Many

SocialMediaPlatform "hosts" Post: One-to-Many

Campaign "uses" Post: One-to-Many

Post "analyzed by" Analytics: One-to-One

6. Brief Report on Design Decisions

The design of the Social Media Analytical Platform was thus driven by handling dynamic user generated content along with rich analytical features toward both tracking engagement and the management of user interaction. Several critical design decisions that were considered are:

a. Normalization:

The database is normalized to reduce data redundancy and ensure that there is no data loss during transactions. The content in the Posts table, while the metrics like likes, shares, comments are stored in Analytics, which makes this application scalable and able to calculate the metrics over a long time without any duplicated data.

b. Connections with Many-to-Many Relationship:

The Connections table was aimed at depicting the many-to-many relationship between users as they connect with each other in different statuses (pending, accepted, blocked). This was necessary to catch the dynamics of user interaction on the platform.

c. Security Concerns:

Security has been a major concern with handling sensitive user information, such as passwords. This design incorporates password hashing, which protects user credentials.

d. Flexibility and Scalability:

This system was to be designed flexible enough to take care of dynamic interactions such as posts, comments, and likes and scalable enough to process large data especially in the Analytics table that will offer real-time insights into how the post is performing.

In conclusion, the design of the Social Media Analytical Platform has put a focus on consistency in data, tracking interaction between users, and ability to scale with the platform's growth. Using normalized tables, foreign key constraints, and efficient data handling practices will enable the system to support both social features and detailed analytics for the end user.