Projektuppgift

Databaser DT003G

Databas för social media

Elias Eriksson

Mittuniversitetet

MID SWEDEN UNIVERSITY

Campus Härnösand Universitetsbacken 1, SE-871 88. **Campus Sundsvall** Holmgatan 10, SE-851 70 Sundsvall.
Campus Östersund Kunskapens väg 8, SE-831 25 Östersund.
Phone: +46 (0)771 97 50 00, Fax: +46 (0)771 97 50 01.

MITTUNIVERSITETET
Avdelningen för informationssystem och -teknologi

Författare: Elias Eriksson, eler2006@student.miun.se
Utbildningsprogram: Webbutveckling, 120 hp
Huvudområde: Datateknik
Termin, år: VT, 2021

Sammanfattning

En databas för en enklare webbapp designas och implementeras i mariadb. Databasen har stöd för användare att göra inlägg på exempelvis en webbplats där användare också har möjlighet att kommentera på inlägg samt att möjligheten för att gilla inlägg och kommentarer finns.

Innehållsförteckning

1 Introduktion

1.1 Bakgrund och problemmotivering

En databas byggs med syftet att kunna användas för en social media webbplats. Databasen kommer att innehålla inloggningsuppgifter för en användare som så att en användare kan kopplas samman med vad användaren gör på webbplatsen. Den inloggade användaren kan sedan kopplas samman med dess profil som innehåller mer personlig data i form av förnamn, efternamn, en valfri beskrivning och profilbild. Det ska även vara möjligt att veta när användaren skapade sin profil.

Det ska sedan vara möjligt för användaren att skapa bloggposter som kan innehålla bilder. En blogg post ska gå att koppla samman med vilken användare det är som har gjort posten. När en post är gjord ska andra användare sedan ha möjlighet att kommentera på bloggposten. Det ska även gå att se vilken person det är som lagt upp kommentaren.

Både kommentarer samt bloggposts ska gå att gilla och det ska stå när inläggen och kommentarerna är gjorda.

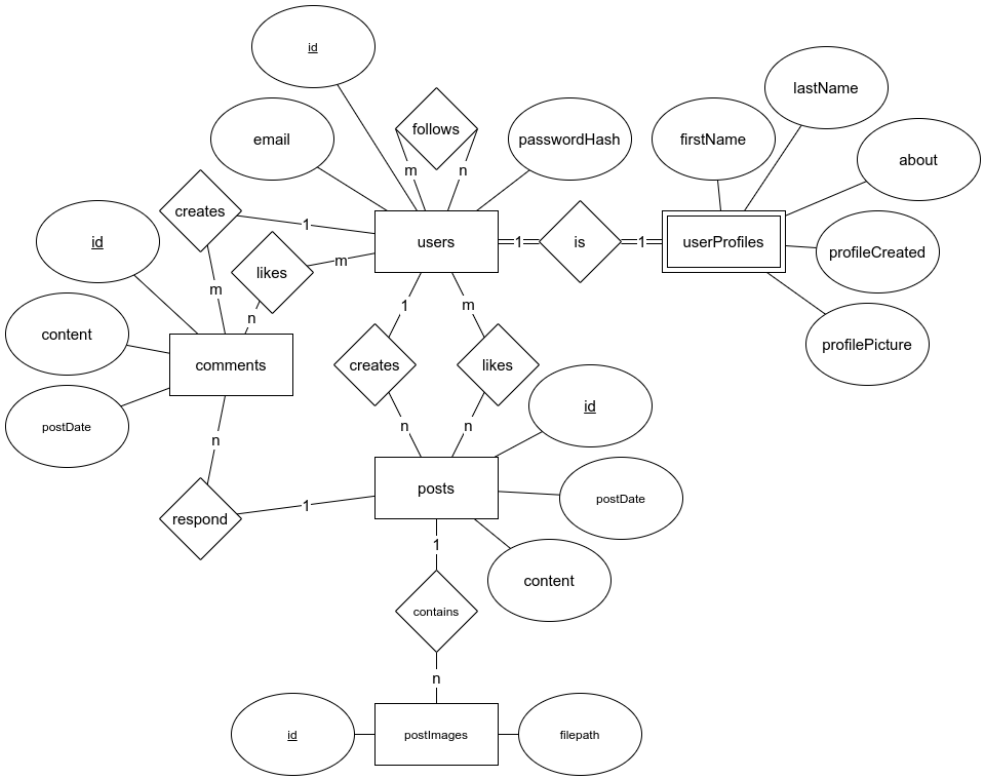
Med databasen klar så kommer det bland annat vara möjligt att få ut information såsom vilken användare som har skrivit vad på sidan, vilka användare som kommenterat på sidan, vem den nyaste användaren är och mycket mer.

2 Metod

För att skissa databasen skapades ett ER diagram i webbappen draw.io. Där den textuella beskrivningen låg som grund till denna skiss. ER diagrammet översattes sedan till relationer för att mer noggrant visa vilka tabeller som kommer finnas samt vilka främmande nycklar som pekar på vilka primär nycklar. När skisserna och relationerna innehöll all data som beskrivits i den textuella beskrivningen så implementerades databasen i SQL med mariadb som DBMS. SQL kod skapades i 4 olika filer för att kunna installera databasen, fylla den med test data samt testa om det går att lägga in konstiga värden och om det går att göra sökningar i databasen som tänkt. Testdatafilen är utformad på ett sådant sätt att testa om det går att sätta in konstiga värden på varje tabells PK, FK samt tabellens constraints för andra attribut.

3 Konstruktion

I den textuella beskrivningen nämns en användare med möjlighet att logga in samt kunna kopplas till en profil. Detta representerades i ER diagrammet [Figur 1] som en entitet users med attributen id, email och passwordHash. Även fast email kommer vara unik så lades fortfarande ett id till till users då en email kommer vara lång och upprepas många gånger via främmande nycklar och en int är alltid 4 bytes medan en hel email alltid kommer vara längre än 4 tecken och därmed alltid större än 4 bytes. Användarprofilen som nämns representeras som en svag entitet userProfiles där användaren har ett fullständigt deltagande till userProfiles. userProfiles har attributen firstName, lastName, about, profilePicture och profileCreated.



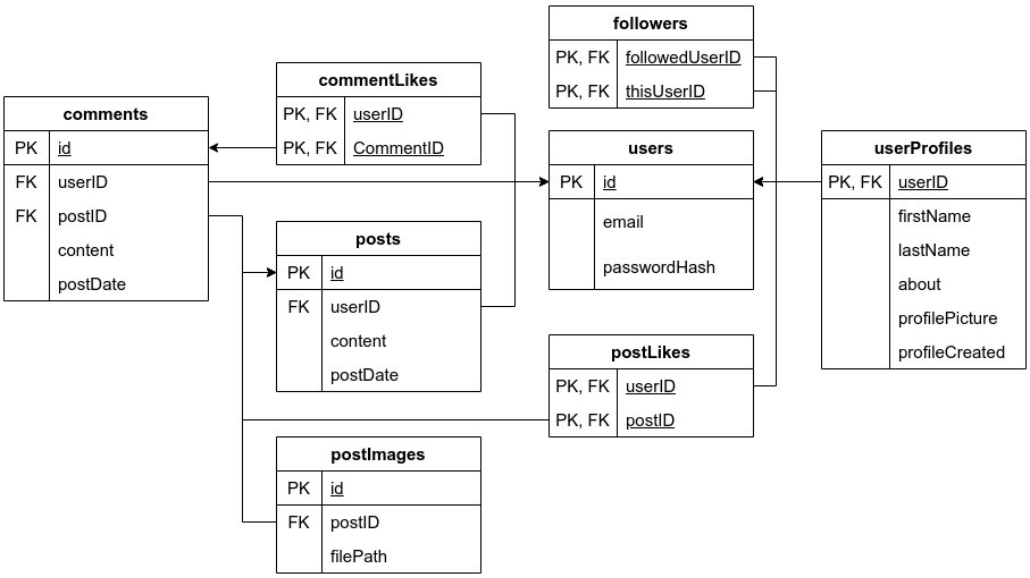
Figur 1: ER diagrammet som skapades från den textuella beskrivningen. Diagrammet visar hur de 5 entitererna users, userProfiles, posts, postImages och comments relaterar till varandra.

En entitet posts skapas för att representera en blogg post som sammanlänkas till användare med ett flera till 1 samband då en användare kan skapa flera posts. posts innehåller attributen id, content samt postDate. Blogg posts ska även ha stöd för att innehålla bilder vilket representeras med en entitet postImages för att kunna stödja ett ospecificerat antal bilder till varje post. PostImages håller attributen id samt filepath och länkas till posts med ett flera till 1 samband.

En entitet comments skapas för att representera en kommentar till en post. comments innehåller attributen id, content samt postDate och sammanlänkas till posts med ett flera till 1 samband då en post kan ha flera kommentarer. comments sammanlänkas även med users med ett flera till ett samband då en användare kan skapa flera kommentarer.

För att användare ska kunna följa andra användare skapades ett många till många samband followers från users till users. Detsamma gjordes mellan posts och användare samt comments och användare för att visa gillningar på posts och comments.

ER diagrammet översattes sedan till relationer [Figur 2].



Figur 2: Relationer över alla entiteter från ER diagrammet och m-n samband. Relationerna visar vilka vad det är som kommer vara PKs i databas tabellerna samt vilka tabeller det är som har en FK mot en annan tabell och vad den FKn pekar på.

I relationerna märks det ut att users primär nyckel är dess id attribut. userProfiles primär nyckel är även dess främmande nyckel som pekar på users id för att kunna koppla den inloggade användaren till en profil.

posts primär nyckel är id och har en främmande nyckel userID som pekar på users för att avgöra vilken användare det är som skapat blogg posten. postImages primär nyckel är dess id och har en främmande nyckel till posts för att visa vilken post bilden tillhör.

comments primär nyckel är id och har en foregn key som pekar på users för att avgöra vilken användare det är som skapat komentaren samt en foregn key till posts för att visa vilken post kommentaren kommenterar.

Flera till flera sambandet followers blir här representerat som en tabell och får en sammansatt nyckel som primär nyckel där båda delarna i den sammansatta nyckel

är en varsin främmande nyckel som pekar på olika users id. Ett id krävs altså inte då en person bara kan följa en annan person en gång.

Samma sak görs för postLikes och commentLikes. Båda dessa tabellers främmande nycklar blir en sammansatt primärnyckel och främmande nycklarna som pekar på users samt posts respektive comments. Det funkar här med då en användare bara kan gilla en kommentar eller ett inlägg en gång.

Med relationerna klara så började arbetet med att implementera databasen i SQL med mariadb som DBMS. Vid implementationen skapades fyra filer. Den fösta filen install.sql inehåller all SQL för att återskapa alla tabellers struktur med alla deras constraints. Alla primära nycklar är tillagda med namngivda constraints och alla främmande nycklar är tillagda med namngivda constraitns samt att on delete cascade är tillagt för att uppehålal databasens integritet vid radering utav rader som har främmande nycklar som pekar på sig. Vid implementering av denna fil lades några extra constrains till för specifika tabellers attribut.

En extra regex constraint (^[^@]+@[^.]+\..+$) lades till på users.email för att kontrollera att alla e-postadresser följer formatet x@y.z. Ett krav på att lösenords hashen ska vara minst 1 tecken långt lades även till.

userProfile har en extra regex constrain (^/.+) för profilePicture för att se till att alla filsökvägar till profilbilderna är absoluta. En extra regex constraint (^[[:alpha:]]+$) för lastName och firstName finns även för att filtrera ut alla tecken som inte räknas som bokstäver.

posts samt comments har en extra constraint som ser till att inehållet måste innehålla minst 1 tecken.

postImages har samma regex constraint (^/.+) users.profilePicture som ser till att alla fil sökvägar är absoluta.

followers har en extra constraitns som inte tillåter thisUserID att vara lika med followedUserID för att förhindra att en användare följer sig själv.

Med install.sql filen klar kan databasen installeras genom att köra filen install.sql i terminalen med kommandot source och databasen kan börja fyllas med data. Alla inserts för exempel data skrevs ner i en fil data.sql och databasen kan fyllas med data från data.sql med kommandot source i terminalen efter att install.sql har körts.

Testdata för att försöka bryta mot de tänkta constraintsen skapades sedan och skrevs ner i en fil tests.sql där alla rader förväntas ge ett fel. Alla tabellers primära nycklar testas genom att försöka duplicera dom och tabellernas främmande nycklar testas genom att försöka sätta in ett värde på en kolumn som inte finns i referens tabellen. Constraintsen som beskrivits ovan testas också. Filen tests.sql kan köras med source i terminalen efter att data.sql har körts.

Efter att alla tester va skapad och testkörd gjordes några sökningar i databasen för att se till att det går att söka igenom den som tänkt. Dessa frågor skrevs ner i en fil

queries.sql som kan köras efter att tests.sql har körts. Frågorna testar bland annat om tabellernas främmande nycklar kopplar samman tabellerna som tänkt.

4 Resultat

Databasen är utformad enligt beskrivningen i introduktionen. Databasen skulle kunna användas för en enklare webbapp som önskar att sina användare kan posta små bloggposts med bilder samt ha andra användare kommentera inläggen. Databasen är sökbar och det är möjligt att få ut bland annat vilka användare som har postat vad, hur många användare sidan har, hur många det är som kommenterat på ett vist inlägg. Det är även möjligt att ta bort saker från databasen och den bevarar automatiskt sin referens integritet då all främmande nycklar har referensintegritetsåtgärden `on delete cascade`.

5 Slutsatser

Databasen fungerar för det som den är utformad för men som med allt så går det alltid att göra det bättre. Just nu är det inte möjligt för användare att kommentera kommentarer vilket är väldigt vanligt på dagens sociala media. Det skulle potentiellt vara önskvärt för en eventuell kund att inte ha den möjligheten men det skulle alltid vara bättre att ha stödet för det. Bortsett från detta är jag nöjd med strukturen på på databasen i sin nuvarande struktur.