Jonas Zeitler, jonze168, Tekniska Högskolan - Linköpings Universitet
2012-12-07

RUT-fileManager

TNM055 - databaser, projektrapport

Problemformulering

Bakgrund

Under slutet av oktober kontaktades jag av Rektorsutbildningen, en avdelning på Uppsala Universitet. De hade ett problem med filhantering i de forskar-/utbildargrupper som arbetar på avdelningen. Dessa utbildare uttryckte en frustration över att ständigt förlora digitalt material i form av pdf-filer, bilder etc. Dessutom kände de att de inte hade några ändamålsenliga system för att dela material med sina kollegor.

Syfte

Syftet för detta projekt har främst varit att utveckla en applikation som löser de problem som formuleras ovan. Utöver detta har lärande och personlig utveckling för mig som webbutvecklare varit stora faktorer i projektets beslutsprocesser.

Konkretiserat

Den applikation som är under utveckling är en plattform för materialåtkomst. En önskvärd egenskap för applikationen är att kunden vill att den ska fungera som ett virtuellt bibliotek. Filer som finns tillgängliga i databasen och på servern ska vara åtkomliga på flera olika sätt. Detta innebär höga krav på applikationens möjlighet att indexera all tänkbar info om de filer som laddas upp. Det ställer dessutom krav på såväl användargränssnitt som databas-/ kodstruktur så att användarupplevelsen blir så intuitiv som möjligt.

Metod

Bakgrundsundersökning

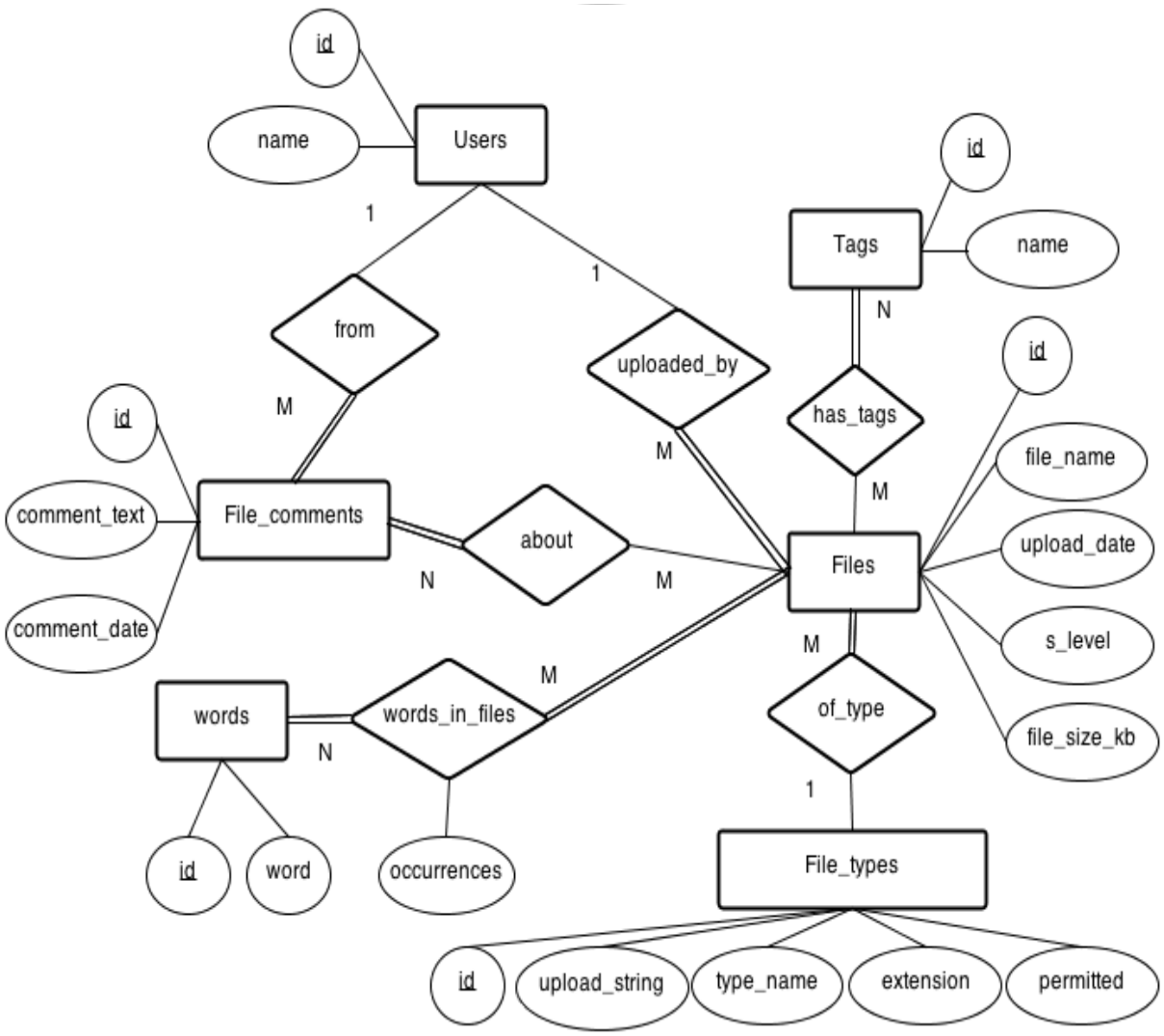
För att få en bra bakgrund till projektet inleddes en undersökning av avdelningens anställda och deras uppleveler angående filstruktur. Denna undersökning innefattade ett antal korta intervjuer och en enkät. Responsen var omfattande och gav en tydlig bild av vad kunden önskade.

Projektstruktur

Arbetet med projektet har strukturerats genom användning av en versionshanterare (Git/ github). För själva projektets struktur beslutades att använda ett designmönster, i detta fall en MVC (Model-View-Controller). MVC-ramverket är *helt* självkodat.

Databasstruktur

Ett utkast för *entitetsrelationsdiagram* över databasens struktur framställdes, det slutliga ER-diagramet visas i figur 1.



Figur 1: ER-diagram

Detta ER-diagram utvecklades till ett *relationsschema* (figur 2) över databasen. Databasen är, enligt figur 2, i *Boyce-Codd Normal Form*.

Users(id, name)

File_types(id, type_name, upload_string, extension, permitted)

Files(id, file_name, upload_date, s_level, user+_id, file_size_kb, file+_type)

Tags(id, name)

Has_tag(tag_id, file_id)

File_comments(id, comment_text, comment_date, user_id)

Comments_about_files(comment+_id, file+_id)

Words(id, word)

Words_in_files(word+_id, file+_id, occurrences)

Figur 2: Relationsschema

Databasen innehåller alltså information om filer på en grundläggande nivå. Prejektet har hittills betått i att indexera unika ord från speciellt pdf-filer. Databasen är organiserad så att funktioner för *sökning* och *bläddring* kan implementeras i applikationen. Dessutom görs filkommentarer och *tags* för filer tillgängliga genom *join*-operationer. Tanken är att en användare ska kunna bläddra i biblioteket med *tags* som kategorier. En möjlig utökning för att förhöja användarens upplevelse av ett bibliotek är att lägga till tabeller/attribut om författare/upphovsorganisation för de filer som indexeras.

SQL-queries

Exempel på en relevant query från projektet är en sökningsquery:

```
SELECT files.fileName, files.id,
ROUP_CONCAT(DISTINCT matched_words.word SEPARATOR ', ') AS 'word',
GROUP_CONCAT(DISTINCT matched_comments.comment_text SEPARATOR ', ')
AS 'comment_text',
files.upload_date,
users.name
FROM files
      LEFT OUTER JOIN comments_about_files
      ON files.id = comments_about_files.comment_id
      LEFT OUTER JOIN matched_comments
      ON comments_about_files.file_id = matched_comments.id
      LEFT OUTER JOIN words_in_files
      ON words_in_files.file_id = files.id
      LEFT OUTER JOIN matched_words
      ON matched_words.id = words_in_files.word_id
      INNER JOIN users
      ON files.user_id = users.id
      WHERE files.fileName LIKE '$term%'
      OR matched_words.word LIKE '%'
      OR matched_comments.comment_text LIKE '%'
GROUP BY files.id
ORDER BY files.upload_date DESC
```

Queryn innehåller dessutom information från två views som skapas dynamiskt vid varje sökning, detta är en av dem:

```
$query = "  CREATE VIEW matched_words AS
            SELECT *
            FROM words JOIN words_in_files
            ON words.id = words_in_files.word_id
            WHERE words.word REGEXP '^";
$query .=   implode('|^', $terms)."' ORDER BY
      words_in_files.occurrences DESC";

// $terms refererar här en array innehållandes de sökord som en
// användare har skrivit in
```

Avslutningsvis

Detta är ett projekt som jag lagt ner ca 100 timmars arbete på hittills. Det är inte klart i någon mening men är en bra grund för fortsatt arbete.