[Report]

Assignment 2

2DV513- Database Theory

Author: Amelie Löwe, Johan Eriksson
Date: 2019-12-20
Course: Database Theory
Course code: 2DV513

1.1 Introduction

Following assignment is part of the Database theory course- 1DV513 and focuses on relational algebra and SQL. The main task is creating and hosting data from Reddit on a database designed by our team. According to the assignment description ”Your main objective is to balance ease of implementation and performance; the database should perform as well as possible without sacrificing how understandable your design is”.

1.2 Resources and Equipment

For our assignment we used following relational database management system
to host our tables: MySQL workbench, version 8.0.18.
The programming language chosen was javascript to automate the importing data process. The IDE is Visual Studio Code, version: 1.41.0. We also chose to work with Node.js MySQL.

2.1 Relational Algebra

student(id, name); enrolledIn(id, code); subject(code, lecturer). First lets make a few tables to help us out:

student	
id	**name**
1	A
2	B

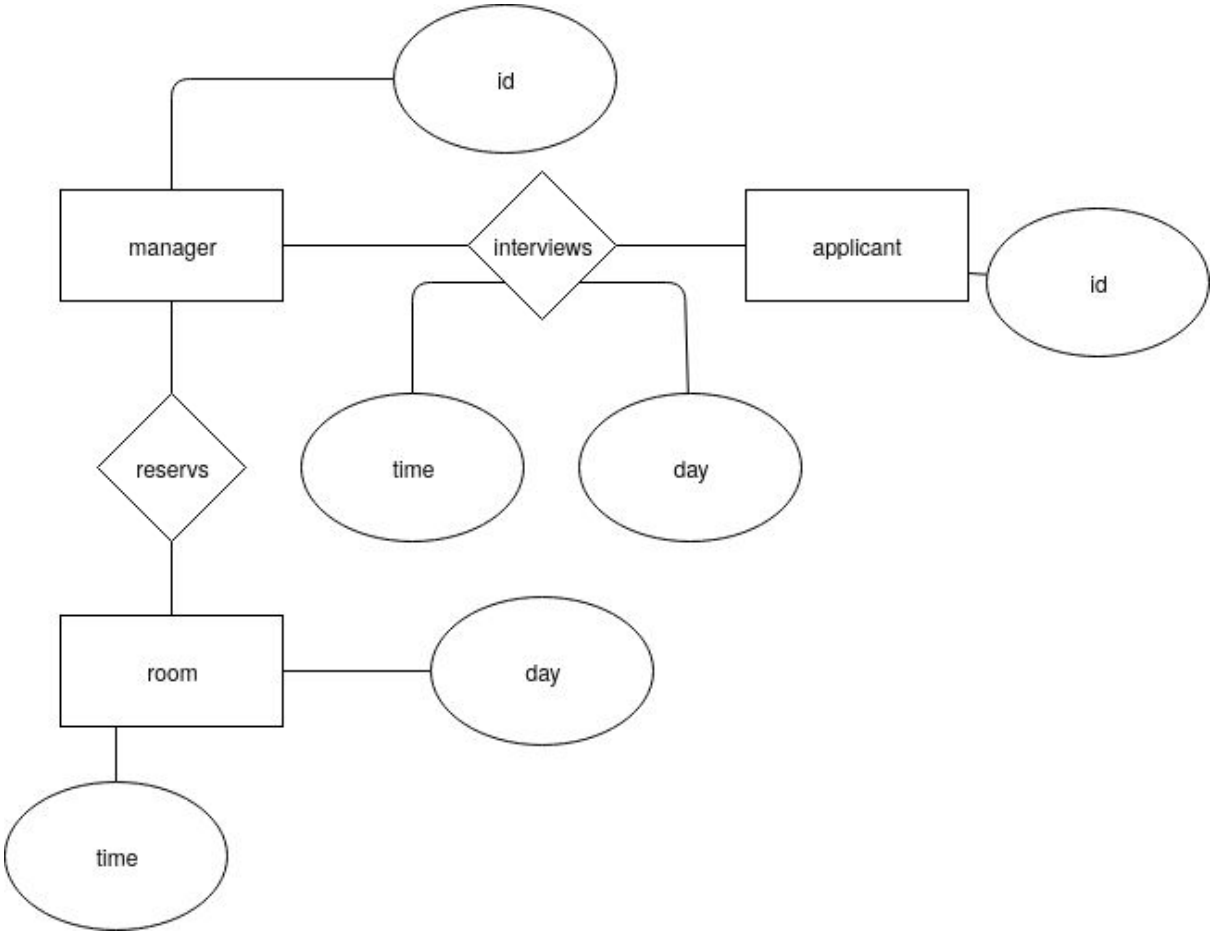
enrolledIn	
id	**code**
1	2dv513
2	2dv513
1	1dv513
1	2dv610

subject	
code	**lecturer**
2dv513	Ilir
1dv513	ilir
2dv610	Mr X

1. What are the names of students enrolled in 2dv513?
   answer: $\pi_{name}(\sigma_{code='2dv513'}(student \bowtie enrolledIn))$
2. What are the names of students in both 1dv513 and 2dv513?
   answer: $\pi_{name}(\sigma_{code='1dv513'}(student \bowtie enrolledIn)) \cap \pi_{name}(\sigma_{code='2dv513'}(student \bowtie enrolledIn))$
3. Who teaches 2dv610?
   answer: $\pi_{lecturer}(\sigma_{code='2dv610'}(subject))$
4. Who teaches 1dv513 and 2dv513?
   answer: $\pi_{lecturer}(\sigma_{code='1dv513' \wedge code='2dv513'}(subject))$
5. What are the names of students who are taking a subject not taught by Ilir?
   answer: $\pi_{name}(\sigma_{lecturer<>Ilir}(student \bowtie subject \bowtie enrolledIn)$

3.1 FDs and Normalization

1. Find Functional dependencies
    a. Since each manager uses the same room during a specific day for all the interviews that day a functional dependency would be:
        manager day → room
    b. Since each applicant has a maximum of being interviewed one time a day by one manager for that day and in one room for that day during a certain time a functional dependency would be:
        applicant day → manager time room
    c. Since a room is booked by a manager for a certain time for one interview with a specific applicant a functional dependency would be:
        room day time → applicant manager
2. Find the keys of the relation
    From the answer in question one we can see that {manager, day} is a key and {applicant, day} is an other key.
3. Show that the relation is in 3NF but not in BCNF
    BCNF requires a superkey. From the first question answer a we have a key {manager, day} however it is not a superkey.
    If we look at the same key for 3NF and looking at the right-hand side "room", is part of a key (1C).
4. Decompose the relation in relations that are in BCNF
    The functional dependency that needs to be used is manager day → room (of the ones we have found). From the book (pdf-page 127/ p.90) we should have one relation where all attributes from the functional dependency is: {manager, room, day} and another where the attributes on the left hand side is added together with the attributes that are not included in the functional dependency (excluding the right hand side): {manager, day, applicant, time}
5. Draw an E/R diagram that describes the system. Try to incorporate all dependencies.


id
manager
interviews
applicant
id
reservs
time
day
room
day
time

4.1 Setting up the Reddit database

The database was set up according to the picture below, provided by the assignment description in the a2_2dv513 pdf file. Hence the grouping of the keys was done by considering their actual purpose and using the description column in the picture. Therefore following key-values where found and the corresponding tables set up.

Comment table:

- com_body: data.body,
- com_link: data.link_id,
- com_utc: data.created_utc,

User table:

- user_id: data.parent_id,
- user_name: data.name,
- user_author: data.author,
- user_score: data.score,

Subreddit table:

- sub_id: data.subreddit_id,
- sub_name: data.subreddit

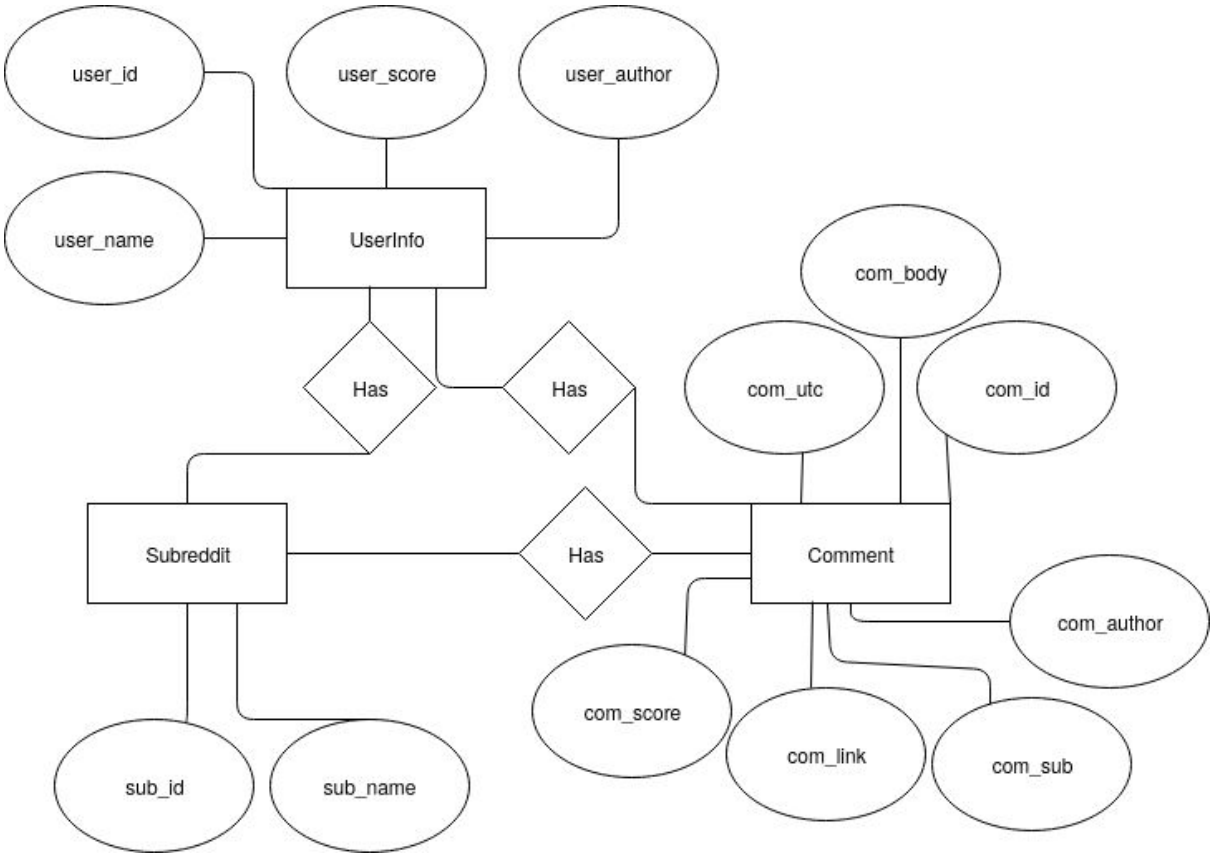
Key	Description
`id`	The id of a comment. The id is an integer value encoded using base36
`parent_id`	The id of the thing this comment is a reply to, either the link or a comment in it
`link_id`	The id of the link this comment is a reply to, can be same as `parent_id`
`name`	The name is a combination of a type prefix and the `id` of a post. The prefix `t1` indicates that it is a comment. You can find a full list of type prefixes at https://www.reddit.com/dev/api
`author`	The posters name
`body`	The comment's contents
`subreddit_id`	The id of the subreddit in base36
`subreddit`	The name of the subreddit
`score`	The combination of up and down votes. Note that `ups` and `downs` are not reliable in this dataset, `score` is often (always?) the same as `ups`
`created_utc`	When the comment was posted (UTC epoch-second format)

4.2 Connecting

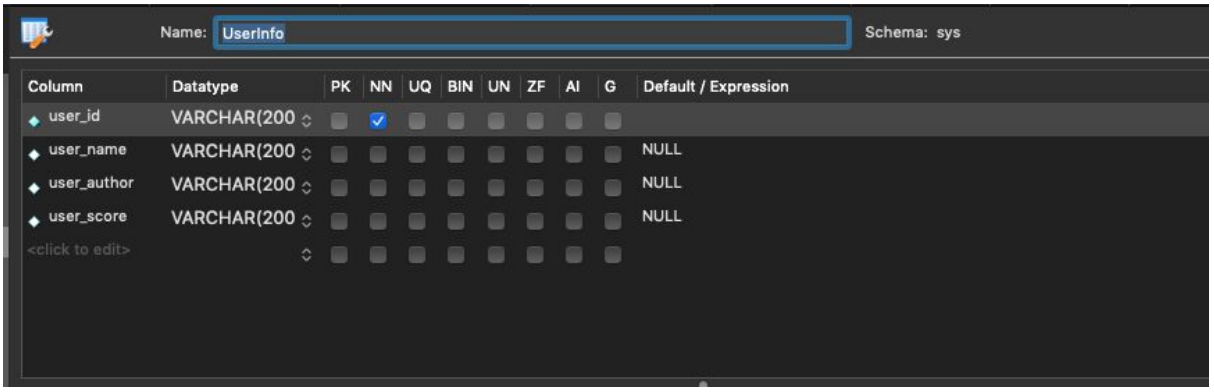
The connection to the database was created by providing a host, port, user and password.
For this section we used w3schools tutorial as a source and inspiration material- “Node.js MySQL - Get Started, Link: https://www.w3schools.com/nodejs/nodejs_mysql.asp”

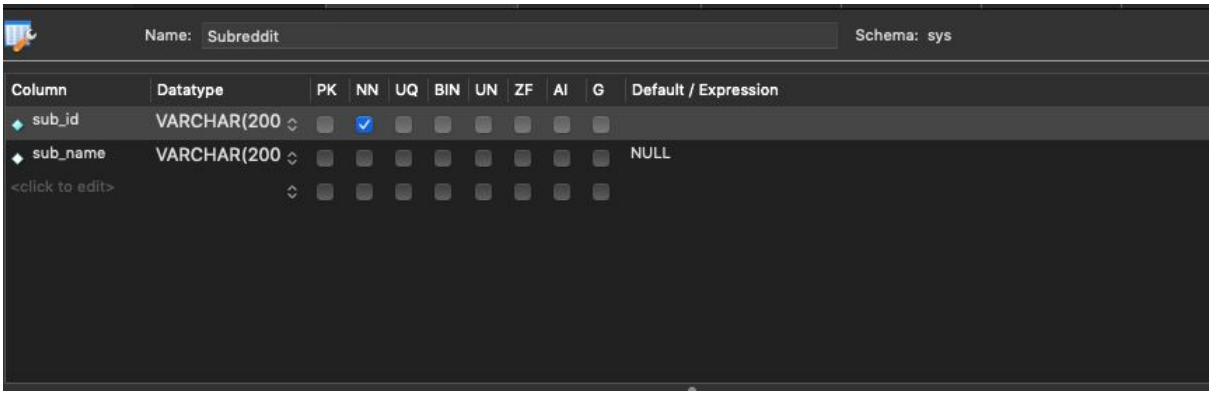
```
var con = mysql.createConnection({
  host: "localhost",
  port: 3306,
  user: "root",
  password: "Ulrik123456",
  database: "sys"
});

con.connect(function(err) {
  if (err){
    console.log("IT DIDNT WORK");
    throw err;
  } else{
    console.log("Connected!");
    fetchData('./RC_2007-10');
  }
});
```

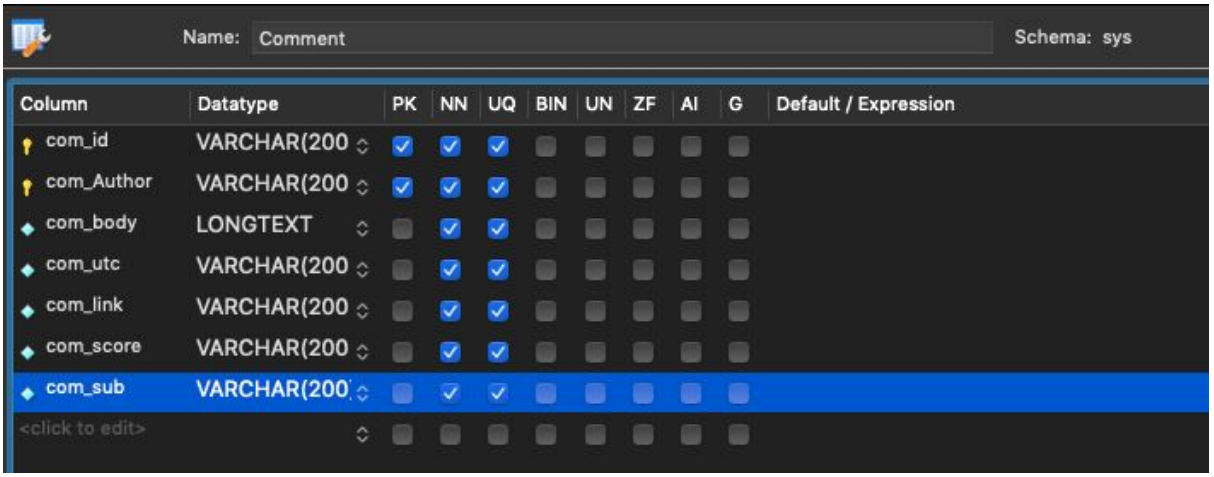
4.3 E/R Diagram



4.4 The Tables







5.1 Importing data

For automating the process of import the data, we read each line of the json file and parse it to the saveFile method which inserts the data into the sql queries.
However while importing the data of the RC-2011-07 file we encountered a severe problem. After about 59490 milliseconds (almost at the one minute mark) the javascript heap ran out of memory. The memory was configured to 2GB.

Therefore we had to change the settings to allow for larger files.

```
{
    "files.maxMemoryForLargeFilesMB": 7066,
    "git.ignoreLimitWarning": true
}
```

```
<--- Last few GCs --->

[14547:0x10256b000]    69537 ms: Mark-sweep 1397.3 (1432.1) -> 1397.0 (1424.1) MB, 1510.7 / 0.0 ms  (+ 0.0 ms in 9 steps
ms, walltime since start of marking 1521 ms) (average mu = 0.155, current mu = 0.088) finaliz[14547:0x10256b000]    69574
1428.1) MB, 20.4 / 0.0 ms  (average mu = 0.155, current mu = 0.088) allocation failure


<--- JS stacktrace --->

FATAL ERROR: Ineffective mark-compacts near heap limit Allocation failed - JavaScript heap out of memory
 1: 0x100038c65 node::Abort() [/usr/local/bin/node]
 2: 0x100038e41 node::OnFatalError(char const*, char const*) [/usr/local/bin/node]
 3: 0x1001b43a5 v8::internal::V8::FatalProcessOutOfMemory(v8::internal::Isolate*, char const*, bool) [/usr/local/bin/node
 4: 0x10057ee02 v8::internal::Heap::FatalProcessOutOfMemory(char const*) [/usr/local/bin/node]
 5: 0x1005818d5 v8::internal::Heap::CheckIneffectiveMarkCompact(unsigned long, double) [/usr/local/bin/node]
 6: 0x10057d77f v8::internal::Heap::PerformGarbageCollection(v8::internal::GarbageCollector, v8::GCCallbackFlags) [/usr/l
 7: 0x10057b954 v8::internal::Heap::CollectGarbage(v8::internal::AllocationSpace, v8::internal::GarbageCollectionReason,
]
 8: 0x10058441b v8::internal::Heap::FinalizeIncrementalMarkingIfComplete(v8::internal::GarbageCollectionReason) [/usr/loc
 9: 0x1005942df v8::internal::IncrementalMarkingJob::Task::RunInternal() [/usr/local/bin/node]
10: 0x10009ba8b node::PerIsolatePlatformData::RunForegroundTask(std::__1::unique_ptr<v8::Task, std::__1::default_delete<v
11: 0x10009b456 node::PerIsolatePlatformData::FlushForegroundTasksInternal() [/usr/local/bin/node]
12: 0x100974b6a uv__async_io [/usr/local/bin/node]
13: 0x1009843bb uv__io_poll [/usr/local/bin/node]
14: 0x100974fdd uv_run [/usr/local/bin/node]
15: 0x100040569 node::Start(v8::Isolate*, node::IsolateData*, int, char const* const*, int, char const* const*) [/usr/loc
16: 0x10003f990 node::Start(uv_loop_s*, int, char const* const*, int, char const* const*) [/usr/local/bin/node]
17: 0x10003f4ff node::Start(int, char**) [/usr/local/bin/node]
18: 0x100001034 start [/usr/local/bin/node]
19: 0x2
```

```
1 •  SELECT * FROM sys.Comment;
```

100% 27:1

Result Grid Filter Rows: Search Export: Fetch rows:

com_id	com_body	com_utc
c029a1z	pr0n for girls?...I'm intrigued	1192458801
c029a20	Debatable issue? I am not going to spend 5 minutes on this. *YOU* go look up pi...	1192458809
c029a21	I doubt it, also from what I can tell most military leaders fucking hate Bush, they j...	1192458811
c029a22	but amount of methane emitted by vegetarians is probably much less than metha...	1192458811
c029a23	What drives me insane is that Gore preaches incessantly about how everyone sh...	1192458829
c029a24	You submitted 11 stories in 39 minutes. For some reason you live to spam Redd...	1192458830
c029a25	[deleted]	1192458837
c029a26	Anyone who can (and I realise there are some that can't), should vote with thei...	1192458843
c029a27	How Stuff Works is my guiltiest pleasure. A fascinating show.	1192458851
c029a28	By the great beard of Odin, no! You know as soon as some Wiccan, or god forbi...	1192458863
c029a29	Test article	1192458871
c029a2a	> From bash.org	1192458874
c029a2b	oh i want to try	1192458883
c029a2c	Let's not use the words "Bush" and "Hitler" in the same sentence, since this is no...	1192458887
c029a2d	absurd web 2.0 startups	1192458892
c029a2e	Subdual damage.	1192458909
c029a2f	I love Colbert for being the first person to actually stand up to the president and...	1192458921
c029a2g	Nice. Now they just need to have comments refresh in real time.	1192458928
c029a2h	[deleted]	1192458935

Comment 3

```
Comment body:  It is not lunacy if your objective is to wipe the slate clean as per "[the Shock Doctrine](http://www
 Naomi Klein.
Comment body: Er, how about someone who thinks Dick's a madman?
Comment body: &gt;Grants zebra is greeted by Brandy, an Atlantic bottlenose dolphin while out on a daily walk around

I didn't know what angle is better, a zebra fish, or a dolphin that takes a daily walk?
Comment body: Finally.

As I  mentioned  [a while ago,](http://politics.reddit.com/info/5z3ee/comments/c02bafr) Reddit is exactly like that
Comment body: You're assuming that evolution is capable of solving any problem.  It's quite the opposite: when facin
 out, and evolution has done its job, just as it is doing now amongst the Amish.
Comment body: 2.76 MB is "gigantic" for an online image. And photos.cx doesn't seem to like that file, with a crypti

In any case, I got it working via using the URL version, though I guess the original link is working now too.

http://e.photos.cx/cmx-Abstraction-nsfw-2.7mb.jpg-d9e.jpg
Comment body: There's no such thing as the "scientific method". There is a scientific *attitude* but no method. You
't guess, and even terms which don't have any meanings. It's sickening.

As for what can be attributed to any one man, literally **nothing** can be. So shut the fuck up with your overly lit
Comment body: [deleted]
Comment body: Have you ever experienced that "draining"Amelies-MacBook-Air-3:JS amelielowe$
```

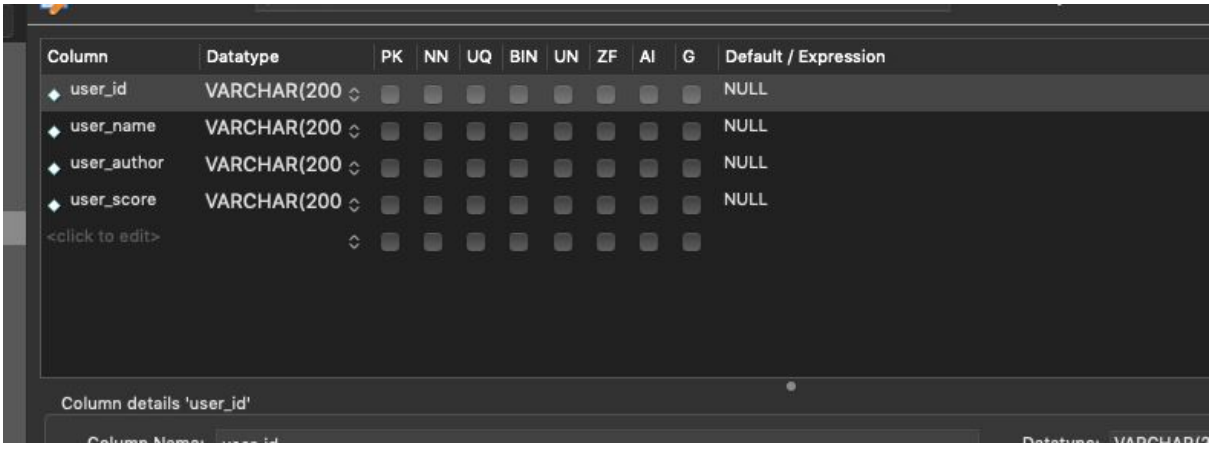
5.2 Two Schema versions

This part of the report will test the cost of constraint by creating to versions of the schema. As well as discussing the results and how the version with constraints compares to the version without any. Furthermore I will evaluate these tests by using a PerformanceObserver to time each test.

```
var timend = performance.now();
console.log("The queries took "+ (timend-timer)+ " milliseconds!");
```

5.2.1 Without Constraints

No primary, not null and unique keys are used, as well as in the javascript program only the INSERT INTO command is given.



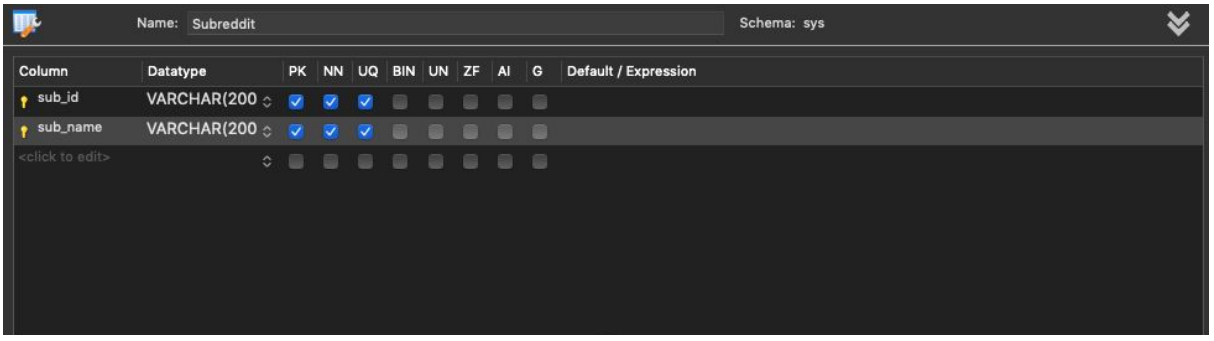
```
var sqlTable2 = "INSERT INTO Subreddit(sub_id,sub_name) VALUES ?";
con.query(sqlTable2, [subValues], function (err, result) {
  if (err) throw err;
});
var sqlTable3 = "INSERT INTO Comment (com_id, com_body,com_utc) VALUES ?";
con.query(sqlTable3, [comValues], function (err, result) {
  if (err) throw err;
  console.log("Comment body: " + data.body);
  var timerEnd = performance.now();
  console.log('The time in milliseconds is: ' + (timerEnd-timer))

});

}
```

The database was tested by using the 2007-10 file. When applying no constraints on the data and printing the user name, userId and comment body the time took 1810769 ms = 30.17948333 min.

```
As for what can be attributed to any one man, literally **nothing** can be. So
The time in milliseconds is: 1810754.4938940005
UserInfo name: [deleted]
UserInfo ID: c02bo4x
Comment body: [deleted]
The time in milliseconds is: 1810769.790426
```

5.2.2 With constraints

Importing the data into the schema with constraints, thus all keys (not null, primary, and unique) where added to the tables, as well as using the “INSERT IGNORE INTO” to discard all duplicate entries and other erroneous data imports.



```
var sqlTable1 = "INSERT IGNORE INTO UserInfo (user_id, user_name, user_author,user_score) VALUES ?";
con.query(sqlTable1, [userValues], function (err, result) {
  if (err) throw err;
  console.log("UserInfo name: " + data.author);
  console.log("UserInfo ID: " + data.id);
});
var sqlTable2 = "INSERT IGNORE INTO Subreddit(sub_id,sub_name) VALUES ?";
con.query(sqlTable2, [subValues], function (err, result) {
  if (err) throw err;
});
var sqlTable3 = "INSERT IGNORE INTO Comment (com_id, com_body,com_utc) VALUES ?";
con.query(sqlTable3, [comValues], function (err, result) {
  if (err) throw err;
  console.log("Comment body: " + data.body);
  var timerEnd = performance.now();
  console.log('The time in milliseconds is: ' + (timerEnd-timer))

});
```

```
Comment body: [deleted]
The time in milliseconds is: 1235144.5312329996
UserInfo name: rbhambha
UserInfo ID: c02bo4y
```

The time with constraints is resulted in 1235145 ms = 20.58575 min.

5.3 Discussion

We were slightly surprised by the results of this test. The hypothesis provided by this team was that the schema with constraints would be significantly slower than the one without, because the the schema without constraints just adds the duplicate entries and the other version has to physically discard them when an error occurs. However this was not the case. We ran these test multiple times without much difference in the time complexities observed. The result might have been different if we further optimized the fetching of data.

Would it be reasonable to import and turn on constraints after?
It would not be reasonable to import the data and then turning on the constraints after we believe, since a lot of erroneous data, such as duplicate entries can be imported into the tables while not having any constraints. Removing the faulty data will most likely add a lot of extra work and is not very cost effective. Furthermore the imported data will also use a lot of valuable space which in some cases might be very sparse.

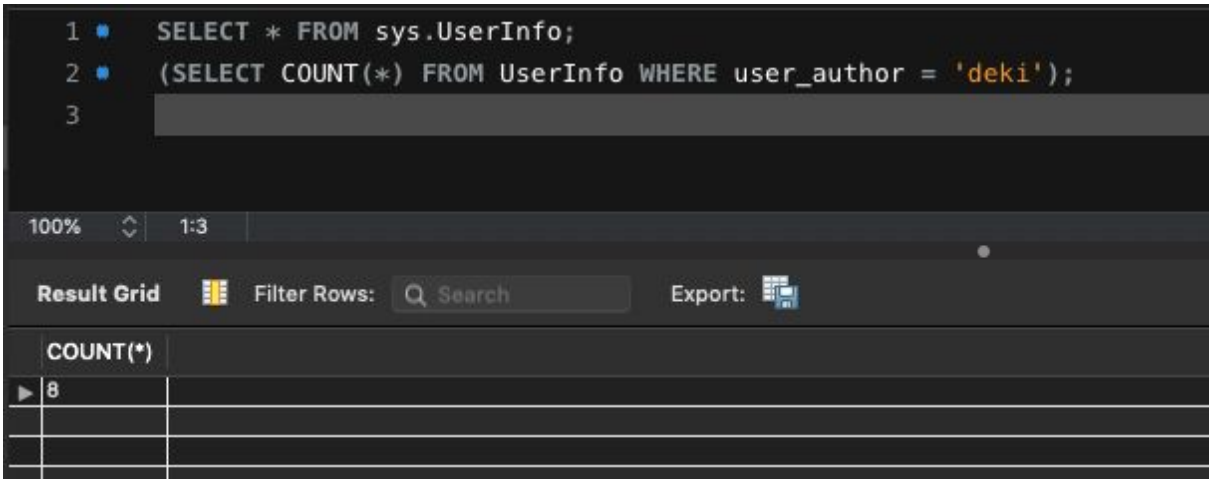
6.1 Queries

The questions are answered by using the test file data added to the tables without any constraints. Therefore a lot of duplicate entries are in the tables and the solutions might seem a bit high in some screenshots.

1. How many comments have a specific user posted?

Solution:

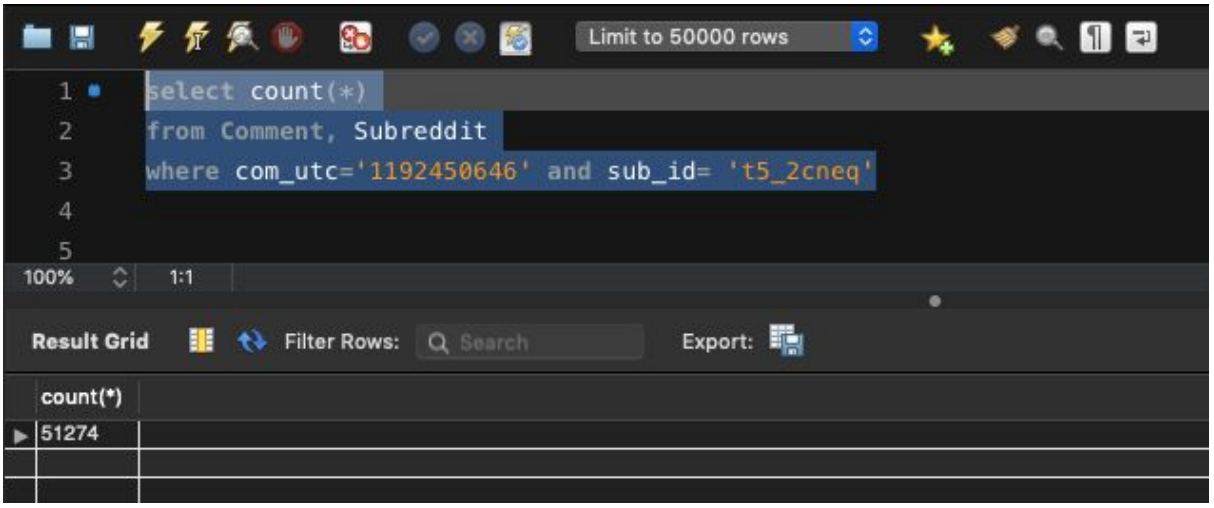
(SELECT COUNT(*) FROM UserInfo WHERE user_author = 'userName');



2. How many comments does a specific subreddit get per day?

Solution:

select count(*) from Comment, Subreddit where com_utc='1192450646' and sub_id= 't5_2cneq'



3. How many comments include the word 'lol'?

Solution:

(SELECT COUNT(*) FROM Comment WHERE com_body LIKE '%lol%');

```
1  SELECT * FROM sys.Comment;
2  (SELECT COUNT(*) FROM Comment WHERE com_body LIKE '%lol%');
3
```

100% 60:2

Result Grid Filter Rows: Search Export:

COUNT(*)
568

4. Users that commented on a specific link has also posted to which subreddits?

Solution:

SELECT DISTINCT sub_name FROM Subreddit JOIN (SELECT com_author FROM Comment WHERE com_link = 't3_5ybc2') AS test1

```
1  SELECT DISTINCT sub_name FROM Subreddit JOIN (SELECT com_author FROM Comment WHERE com_link = 't3_5ybc2')
2
3
```

100% 83:1

Result Grid Filter Rows: Search Export:

sub_name
ads
netsec
features
de
fr
es
ja
tr
eo
it

5. Which users have the highest and lowest combined scores? (combined as the sum of all scores)

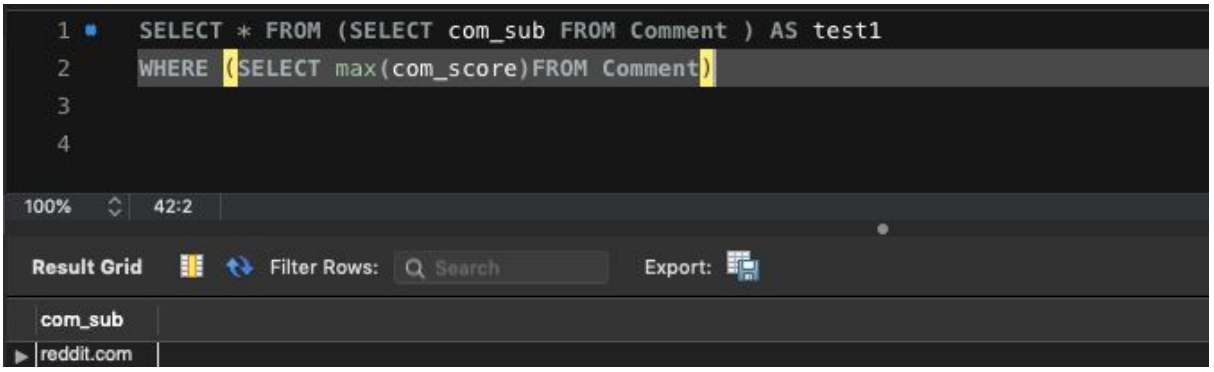
Solution:

```
1  SELECT (SELECT com_Author, SUM(com_score)
2  AS sum
3  FROM Comment GROUP BY com_Author) AS test1
4  UNION DISTINCT
5  SELECT com_Author, MIN(sumMIN)
6  FROM (SELECT com_Author, SUM(com_score) AS sumMIN
7  FROM Comment
8  GROUP BY com_Author) As test2
9
10
11
```

This didnt turn out as planned, we got a lot of errors such as Error Code: 1241 Operand should contain 1 column(s) as well as Error Code: 1248. Every derived table must have its own alias, which we didnt manage to solve.

6. Which subreddits have the highest and lowest scored comments?
Solution:

SELECT * FROM (SELECT com_sub FROM Comment) AS test1
WHERE (SELECT max(com_score)FROM Comment)



Highest : SELECT max(com_score)FROM Comment
Lowest: SELECT min(com_score)FROM Comment

7. Given a specific user, list all the users he or she has potentially interacted with (i.e., everyone who as commented on a link that the specific user has commented on).
Solution:

```
select distinct com_Author
 from (select com_link from Comment where com_Author='mcm69') AS test1
 join
 (select com_link, com_Author from Comment) AS test2
```

```
1 • select distinct com_Author
2   from (select com_link from Comment where com_Author='mcm69') AS test1
3   join
4   (select com_link, com_Author from Comment) AS test2
```

100% 54:4

Result Grid Filter Rows: Search Export:

com_Author
bostich
igiveyoumylife
Arve
[deleted]
gigaquack
Percept
mcm69
deki
llimllib
raldi
jezmck
sickofthisshit
hfaber
paternoster
zoomzoom83
donh

AS test1 and AS test2 are added to avoid the SQL Error (1248): Every derived table must have its own alias.

8. Which users has only posted to a single subreddit?
Solution: SELECT com_Author, COUNT(*) as count FROM Comment GROUP BY com_Author HAVING COUNT(*) = 1

```
1 • SELECT com_Author, COUNT(*) as count FROM Comment GROUP BY com_Author HAVING COUNT(*) = 1
```

100% 13:1

Result Grid Filter Rows: Search Export:

com_Author	count
tashbarg	1
lyon	1
orangepeel	1
crm114	1
walrod	1
sakri	1
tubelight	1
hejog	1
edf825	1
bushwakko	1
Bobsickle	1
flosch	1
tinha	1
greentangent	1
JimJones	1
chiarafox	1

6.1.2 Discussion

“Report the queries you found to work best, together with a brief motivation for why you think that worked best.”
As a group we found that the more direct queries were easier to write since these types of queries need less combinations when making the query. For example the 3. How many comments include the word 'lol'? query was fairly straightforward opposed to others with unions and groups.
The queries where derived tables needed to be created were way more difficult to write we believe.