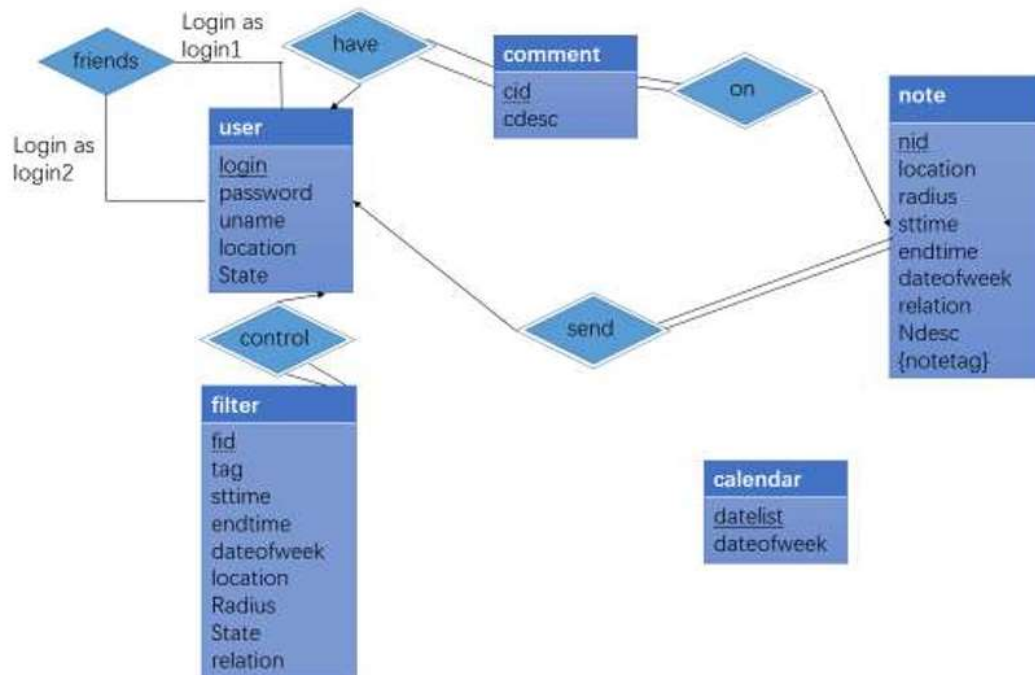


Datebase Project

N11803298 Zixiao Li

Project1

1. ER diagram



2. Introduction





































The schema is like the ER diagram above. I assume: I. Each note has its note id for each user, beginning from 0. II. Each filter also has its filter id for each user, beginning from 0. III. The friends relationship can be a single-direction relationship, which means Alice can regard Bob as a friend when Bob does not regard Alice as his friend. So when Alice and Bob both regard each other friends, there should be two tuples (Alice, Bob) and (Bob, Alice) in the friends table. IV. The calendar table stores a large list of dates and their days of week (Monday, Tuesday, ...), so that people can use dateofweek column in note and filter to identify which day of week they want their note to be shown or filter to be used by us comparing the dateofweek columns of table note or filter to the dateofweek column of table calendar. The dateofweek is set to 0 when there isn't a constraints about days of week. V. The same person can comment on the same note several times so comment should be a weak entity. And since each note can have several tags, tag should be multivalued. VI. Not like many social apps, when relationship is set to friends in a filter, users will not receive notes from themselves here.

3. Tables

users

		login	passwords	uname	location	state
<input type="checkbox"/>	 编辑  复制  删除	Barney@nyu.edu	123456	Barney	[GEOMETRY - 25 字节]	lunch break
<input type="checkbox"/>	 编辑  复制  删除	H&M	123456	H&M	[GEOMETRY - 25 字节]	NULL
<input type="checkbox"/>	 编辑  复制  删除	Lily@nyu.edu	123456	Lily	[GEOMETRY - 25 字节]	just chilling
<input type="checkbox"/>	 编辑  复制  删除	Marshall@nyu.edu	123456	Marshall	[GEOMETRY - 25 字节]	just chilling
<input type="checkbox"/>	 编辑  复制  删除	Robin@nyu.edu	123456	Robin	[GEOMETRY - 25 字节]	lunch break
<input type="checkbox"/>	 编辑  复制  删除	Ted@nyu.edu	123456	Ted	[GEOMETRY - 25 字节]	at work

friends

		login1	login2
<input type="checkbox"/>	 编辑  复制  删除	Barney@nyu.edu	Marshall@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Barney@nyu.edu	Ted@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Lily@nyu.edu	Marshall@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Lily@nyu.edu	Robin@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Marshall@nyu.edu	Barney@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Marshall@nyu.edu	Lily@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Marshall@nyu.edu	Ted@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Robin@nyu.edu	Lily@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Robin@nyu.edu	Ted@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Ted@nyu.edu	Barney@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Ted@nyu.edu	Marshall@nyu.edu
<input type="checkbox"/>	 编辑  复制  删除	Ted@nyu.edu	Robin@nyu.edu

note

login	nid	location	radius	stime	endtime	dateofweek	relation	ndesc
Barney@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-17 18:00:00	2018-11-29 19:00:00	0	friends	I am awesome
Barney@nyu.edu	1	[GEOMETRY - 25 字节]	100	NULL	NULL	0	all	I am always awesome
H&M	0	[GEOMETRY - 25 字节]	100	2018-10-01 00:00:00	2018-11-30 00:00:00	3	all	on sale
Robin@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-28 00:00:00	2018-11-28 23:00:00	0	all	traffic jam
Ted@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-17 18:00:00	2018-11-29 19:00:00	0	friends	good restaurant

notetag

		login	nid	tid
<input type="checkbox"/>	 编辑  复制  删除	Barney@nyu.edu	0	#me
<input type="checkbox"/>	 编辑  复制  删除	Barney@nyu.edu	1	#me
<input type="checkbox"/>	 编辑  复制  删除	H&M	0	#shopping
<input type="checkbox"/>	 编辑  复制  删除	Robin@nyu.edu	0	#transportation
<input type="checkbox"/>	 编辑  复制  删除	Ted@nyu.edu	0	#food

filter

login	fid	tid	stime	endtime	dateofweek	location	radius	state	relation
Barney@nyu.edu	0	#food	2018-11-16 09:23:23	2018-11-30 12:27:29	0	[GEOMETRY - 25 字节]	100	lunch break	friends
Barney@nyu.edu	1	#transportation	2018-11-16 09:23:23	2018-12-31 12:27:29	0	[GEOMETRY - 25 字节]	100	just chilling	all
Lily@nyu.edu	0	#me	NULL	NULL	0	NULL	NULL	just chilling	friends
Lily@nyu.edu	1	#shopping	2018-10-01 00:00:00	2018-12-01 00:00:00	0	[GEOMETRY - 25 字节]	100	just chilling	all
Marshall@nyu.edu	0	#me	NULL	NULL	0	NULL	NULL	just chilling	friends
Robin@nyu.edu	0	#me	NULL	NULL	0	NULL	NULL	NULL	friends
Robin@nyu.edu	1	#food	NULL	NULL	0	NULL	NULL	NULL	friends
Robin@nyu.edu	2	#shopping	NULL	NULL	0	NULL	NULL	NULL	friends
Robin@nyu.edu	3	#transportation	NULL	NULL	0	NULL	NULL	NULL	friends
Robin@nyu.edu	4	#tourism	NULL	NULL	0	NULL	NULL	NULL	friends
Ted@nyu.edu	0	#me	2018-11-01 00:00:00	2018-11-30 00:00:00	0	NULL	100	at work	friends

calendar(partial)

				datelist	dayofweek
<input type="checkbox"/>				2018-11-26	1
<input type="checkbox"/>				2018-11-27	2
<input type="checkbox"/>				2018-11-28	3
<input type="checkbox"/>				2018-11-29	4
<input type="checkbox"/>				2018-11-30	5
<input type="checkbox"/>				2018-12-01	6
<input type="checkbox"/>				2018-12-02	7
<input type="checkbox"/>				2018-12-03	1
<input type="checkbox"/>				2018-12-04	2
<input type="checkbox"/>				2018-12-05	3
<input type="checkbox"/>				2018-12-06	4
<input type="checkbox"/>				2018-12-07	5
<input type="checkbox"/>				2018-12-08	6
<input type="checkbox"/>				2018-12-09	7
<input type="checkbox"/>				2018-12-10	1
<input type="checkbox"/>				2018-12-11	2
<input type="checkbox"/>				2018-12-12	3
<input type="checkbox"/>				2018-12-13	4
<input type="checkbox"/>				2018-12-14	5

Since the table calendar is too large to be shown in this description, I just show part of the table. And table comments is empty since no queries in this project use this table.

4. Constraints

Keys: calendar(primary key(datelist)); users(primary key(login)); friends(primary key(login1, login2), foreign key(login1)references users(login), foreign key(login2)references users(login)); note(primary key(login, nid), foreign key(login)references users(login)); logintag(primary key(login, nid, tid), foreign key(login, nid)references note(login, nid)); filter(primary key(login, fid), foreign key(login)references users(login)); Other constraints: I. tid should be in (#tourism, #shopping, #food, #transportation, #me), relation should be in (friends, me, all and NULL). We use enum function to realize those constraints. II. location in table note can not be NULL because according to the project description, people use filter to identify which kind of note(i.e. tag) and from which kind of people(relation) they want to

see. So since the purpose of this app is to let the users see the notes around them, they shouldn't receive any note far away because of a NULL location. Other constraints about time and location in note and filter can be NULL.

5. Queries

(1) insert into users(login, passwords, uname) values('Ted@nyu.edu','123456','Ted');

Results:

	login	passwords	uname	location	state
<input type="checkbox"/> 编辑 复制 删除	Ted@nyu.edu	123456	Ted	NULL	NULL

(2) insert into note(login, nid, location, radius, sttime, endtime, dateofweek, relation, ndesc)
select 'Ted@nyu.edu', MAX(nid)+1, POINTFROMTEXT('POINT(10 15)'), 100, '2018-11-17 18:00:00', '2018-11-29 19:00:00', 0, 'friends', 'good restaurant' from note where login='Ted@nyu.edu'; Results:

	login	nid	location	radius	sttime	endtime	dateofweek	relation	ndesc
<input type="checkbox"/> 编辑 复制 删除	Ted@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-17 18:00:00	2018-11-29 19:00:00	0	friends	good restaurant

(3) select login2

from friends where login1='Ted@nyu.edu';

Results:

login2
Barney@nyu.edu
Marshall@nyu.edu
Robin@nyu.edu

(4) Explanation: I first create three views about tags they want to see from all people, their friends and themselves, respectively. Then using these views, I can decide which notes they should receive. Time and location constraints should be tested in both note and filter, and here I just discuss the difficult part, relation. I. for tags they want to see from all people, they should receive all notes with relation 'all', notes from friends with relation 'friends' and all notes from themselves. II. for tags they want to see from friends, they should receive all notes with relation not 'me' from friends with relation 'friends'. III. for tags they want to see from themselves, they should receive all notes from themselves. By union the results of procedure I to III, I can get the right results. Then I drop all the temporary views.

Example SQL using 'Barney@nyu.edu' as login:

```
create view allneedtag(tid) as select tid from users join filter using(login) where
login='Barney@nyu.edu' and (st_distance(users.location,filter.location)<filter.radius or
filter.location is NULL) and (now()<filter.endtime or filter.endtime is NULL) and
(now())>filter.sttime or filter.sttime is NULL) and (dateofweek=0 or dateofweek in (select
dateofweek from calendar where date(now())=datelist)) and (users.state=filter.state or
filter.state is NULL) and filter.relation='all';
create view friendsneedtag(tid) as select tid from users join filter using(login) where
```

```
login='Barney@nyu.edu' and (st_distance(users.location,filter.location)<filter.radius or
filter.location is NULL) and (now()<filter.endtime or filter.endtime is NULL) and
(now())>filter.sttime or filter.sttime is NULL) and (dateofweek=0 or dateofweek in (select
dateofweek from calendar where date(now())=datelist)) and (users.state=filter.state or
filter.state is NULL) and filter.relation='friends';
```

```
create view meneedtag(tid) as select tid from users join filter using(login) where
login='Barney@nyu.edu' and (st_distance(users.location,filter.location)<filter.radius or
filter.location is NULL) and (now()<filter.endtime or filter.endtime is NULL) and
(now())>filter.sttime or filter.sttime is NULL) and (dateofweek=0 or dateofweek in (select
dateofweek from calendar where date(now())=datelist)) and (users.state=filter.state or
filter.state is NULL) and filter.relation='me';
```

```
(select note.* from note natural join notetag natural join allneedtag, users where
users.login='Barney@nyu.edu' and (st_distance(users.location,note.location)<note.radius)
and (now()<note.endtime or note.endtime is NULL) and (now())>note.sttime or note.sttime is
NULL) and (dateofweek=0 or dateofweek in (select dateofweek from calendar where
date(now())=datelist)) and (note.relation='all' or (note.relation<>'all' and
note.login='Barney@nyu.edu') or (note.relation='friends' and note.login in (select login2
from friends where login1='Barney@nyu.edu'))))
```

union

```
(select note.* from note natural join notetag natural join meneedtag join users using(login)
where users.login='Barney@nyu.edu' and
(st_distance(users.location,note.location)<note.radius) and (now()<note.endtime or
note.endtime is NULL) and (now())>note.sttime or note.sttime is NULL) and (dateofweek=0
or dateofweek in (select dateofweek from calendar where date(now())=datelist))
```

union

```
(select note.* from note natural join notetag natural join friendsneedtag , users where
users.login='Barney@nyu.edu' and (st_distance(users.location,note.location)<note.radius)
and (now()<note.endtime or note.endtime is NULL) and (now())>note.sttime or note.sttime is
NULL) and (dateofweek=0 or dateofweek in (select dateofweek from calendar where
date(now())=datelist)) and note.relation<>'me' and note.login in (select login2 from
friends where login1='Barney@nyu.edu'));
```

drop view allneedtag;

drop view friendsneedtag;

drop view meneedtag;

Results:

login	nid	location	radius	sttime	endtime	dateofweek	relation	ndesc
Ted@nyu.edu	0	POINT(10 15)	100	2018-11-17 18:00:00	2018-11-29 19:00:00	0	friends	good restaurant

(5) Explanation: I first create views about all users that satisfy the constraints of this note. Then by checking their filter, I can decide whether they should receive this note. Time and location constraints should be tested in both note and filter, and here I just discuss the difficult part, relation. If there is a filter with the same tag of one of the tags of this note, we check its relation column to see (assuming the note is from Ted and its note id is 0. Since

they are primary keys I can assume I already know their value somewhere): I. relation is all, the user should receive this note.. II. relation is friends, I check if Ted is the users' friend and decide whether the user should receive this note. III. relation is me, I check if the user is Ted to decide if the user should receive this note. By union the results of procedure I to III, I can get the right results. Then I drop all the temporary views.

Example SQL using 'Ted@nyu.edu' as login and 0 as nid:

```
create view needcheckfilter(login,location,state,tid) as select
users.login,users.location,users.state,tid from users,note natural join notetag where
note.login='Ted@nyu.edu' and nid=0 and
(st_distance(users.location,note.location)<note.radius) and (now()<note.endtime or
note.endtime is NULL) and (now()>note.sttime or note.sttime is NULL) and (dateofweek=0
or dateofweek in (select dateofweek from calendar where date(now())=datelist)) and
(note.relation='all' or (note.relation='friends' and 'Ted@nyu.edu' in (select login2 from
friends where friends.login1=users.login)) or users.login='Ted@nyu.edu');
```

```
(select distinct needcheckfilter.login from needcheckfilter join filter using(login) where
(st_distance(needcheckfilter.location,filter.location)<filter.radius or filter.location is NULL)
and (now()<filter.endtime or filter.endtime is NULL) and (now()>filter.sttime or filter.sttime is
NULL) and (dateofweek=0 or dateofweek in (select dateofweek from calendar where
date(now())=datelist)) and (needcheckfilter.state=filter.state or filter.state is NULL) and
filter.tid = needcheckfilter.tid and filter.relation='all')
```

union

```
(select distinct needcheckfilter.login from needcheckfilter join filter using(login) where
(st_distance(needcheckfilter.location,filter.location)<filter.radius or filter.location is NULL)
and (now()<filter.endtime or filter.endtime is NULL) and (now()>filter.sttime or filter.sttime is
NULL) and (dateofweek=0 or dateofweek in (select dateofweek from calendar where
date(now())=datelist)) and (needcheckfilter.state=filter.state or filter.state is NULL) and
filter.tid = needcheckfilter.tid and (filter.relation='me' and
needcheckfilter.login='Ted@nyu.edu'))
```

union

```
(select distinct needcheckfilter.login from needcheckfilter join filter using(login) where
(st_distance(needcheckfilter.location,filter.location)<filter.radius or filter.location is
NULL) and (now()<filter.endtime or filter.endtime is NULL) and (now()>filter.sttime or
filter.sttime is NULL) and (dateofweek=0 or dateofweek in (select dateofweek from calendar
where date(now())=datelist)) and (needcheckfilter.state=filter.state or filter.state is NULL)
and filter.tid = needcheckfilter.tid and (filter.relation='friends' and 'Ted@nyu.edu' in (select
login2 from friends where friends.login1=needcheckfilter.login)));
```

drop view needcheckfilter; results:

login
Barney@nyu.edu
Robin@nyu.edu

(6) Explanation: I create a temporary table(since we cannot create index on a view in mysql)

to store the result of (4) and create full text index on the notes' description. Then we search the key words in natural language mode in the notes using the index. Finally I drop all the temporary views and tables.

```
create view allneedtag(tid) as select tid from users join filter using(login) where
login='Barney@nyu.edu' and (st_distance(users.location,filter.location)<filter.radius or
filter.location is NULL) and (now()<filter.endtime or filter.endtime is NULL) and
(now())>filter.sttime or filter.sttime is NULL) and (dateofweek=0 or dateofweek in (select
dateofweek from calendar where date(now())=datelist)) and (users.state=filter.state or
filter.state is NULL) and filter.relation='all';
create view friendsneedtag(tid) as select tid from users join filter using(login) where
login='Barney@nyu.edu' and (st_distance(users.location,filter.location)<filter.radius or
filter.location is NULL) and (now()<filter.endtime or filter.endtime is NULL) and
(now())>filter.sttime or filter.sttime is NULL) and (dateofweek=0 or dateofweek in (select
dateofweek from calendar where date(now())=datelist)) and (users.state=filter.state or
filter.state is NULL) and filter.relation='friends';
create view meneedtag(tid) as select tid from users join filter using(login) where
login='Barney@nyu.edu' and (st_distance(users.location,filter.location)<filter.radius or
filter.location is NULL) and (now()<filter.endtime or filter.endtime is NULL) and
(now())>filter.sttime or filter.sttime is NULL) and (dateofweek=0 or dateofweek in (select
dateofweek from calendar where date(now())=datelist)) and (users.state=filter.state or
filter.state is NULL) and filter.relation='me';
create table allresult as
(select note.* from note natural join notetag natural join allneedtag, users where
users.login='Barney@nyu.edu' and (st_distance(users.location,note.location)<note.radius)
and (now()<note.endtime or note.endtime is NULL) and (now())>note.sttime or note.sttime is
NULL) and (dateofweek=0 or dateofweek in (select dateofweek from calendar where
date(now())=datelist)) and (note.relation='all' or (note.relation<>'all' and
note.login='Barney@nyu.edu') or (note.relation='friends' and note.login in (select login2
from friends where login1='Barney@nyu.edu'))))
union
(select note.* from note natural join notetag natural join meneedtag join users using(login)
where users.login='Barney@nyu.edu' and
(st_distance(users.location,note.location)<note.radius) and (now()<note.endtime or
note.endtime is NULL) and (now())>note.sttime or note.sttime is NULL) and (dateofweek=0
or dateofweek in (select dateofweek from calendar where date(now())=datelist)))
union
(select note.* from note natural join notetag natural join friendsneedtag , users where
users.login='Barney@nyu.edu' and (st_distance(users.location,note.location)<note.radius)
and (now()<note.endtime or note.endtime is NULL) and (now())>note.sttime or note.sttime is
NULL) and (dateofweek=0 or dateofweek in (select dateofweek from calendar where
date(now())=datelist)) and note.relation<>'me' and note.login in (select login2 from
friends where login1='Barney@nyu.edu')); ALTER TABLE allresult ADD FULLTEXT(ndesc);
SELECT * FROM allresult WHERE MATCH (ndesc) AGAINST ('good' IN NATURAL LANGUAGE
```

MODE);
 drop view allneedtag;
 drop view friendsneedtag;
 drop view meneedtag; drop table allresult;
 Results:

login	nid	location	radius	stime	endtime	relation	ndesc
Ted@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-17 18:00:00	2018-11-29 19:00:00	friends	good restaurant

6. Test

Queries(1)(2)(3) is simple, I'll just show the test results of (4)(5)(6). Again I put the table note and filter here for TA to check my answer more conveniently.

note:

login	nid	location	radius	stime	endtime	dateofweek	relation	ndesc
Barney@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-16 09:23:23	2018-11-30 12:27:29	0	friends	I am awesome
Barney@nyu.edu	1	[GEOMETRY - 25 字节]	100	NULL	NULL	0	all	I am always awesome
H&M	0	[GEOMETRY - 25 字节]	100	2018-10-01 00:00:00	2018-11-30 00:00:00	3	all	on sale
Robin@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-28 00:00:00	2018-11-28 23:00:00	0	all	traffic jam
Ted@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-17 18:00:00	2018-11-29 19:00:00	0	friends	good restaurant

filter:

login	fid	tid	stime	endtime	dateofweek	location	radius	state	relation
Barney@nyu.edu	0	#food	2018-11-16 09:23:23	2018-11-30 12:27:29	0	[GEOMETRY - 25 字节]	100	lunch break	friends
Barney@nyu.edu	1	#transportation	2018-11-16 09:23:23	2018-12-31 12:27:29	0	[GEOMETRY - 25 字节]	100	just chilling	all
Lily@nyu.edu	0	#me	NULL	NULL	0	NULL	NULL	just chilling	friends
Lily@nyu.edu	1	#shopping	2018-10-01 00:00:00	2018-12-01 00:00:00	0	[GEOMETRY - 25 字节]	100	just chilling	all
Marshall@nyu.edu	0	#me	NULL	NULL	0	NULL	NULL	just chilling	friends
Robin@nyu.edu	0	#me	NULL	NULL	0	NULL	NULL	NULL	friends
Robin@nyu.edu	1	#food	NULL	NULL	0	NULL	NULL	NULL	friends
Robin@nyu.edu	2	#shopping	NULL	NULL	0	NULL	NULL	NULL	friends
Robin@nyu.edu	3	#transportation	NULL	NULL	0	NULL	NULL	NULL	friends
Robin@nyu.edu	4	#tourism	NULL	NULL	0	NULL	NULL	NULL	friends
Ted@nyu.edu	0	#me	2018-11-01 00:00:00	2018-11-30 00:00:00	0	NULL	100	at work	friends

(4) On Lily:

login	nid	location	radius	stime	endtime	dateofweek	relation	ndesc
H&M	0	POINT(20 20)	100	2018-10-01 00:00:00	2018-11-30 00:00:00	3	all	on sale

On Marshall:

login	nid	location	radius	stime	endtime	dateofweek	relation	ndesc
Barney@nyu.edu	0	POINT(10 15)	100	2018-11-17 18:00:00	2018-11-29 19:00:00	0	friends	I am awesome
Barney@nyu.edu	1	POINT(20 20)	100	NULL	NULL	0	all	I am always awesome

On Robin:

login	nid	location	radius	stime	endtime	dateofweek	relation	ndesc
Ted@nyu.edu	0	POINT(10 15)	100	2018-11-17 18:00:00	2018-11-29 19:00:00	0	friends	good restaurant

On Ted:

login	nid	location	radius	stime	endtime	dateofweek	relation	ndesc
-------	-----	----------	--------	-------	---------	------------	----------	-------

I set Ted's location fall away and so he cannot receive any notes here.

(5) On note

H&M	0	[GEOMETRY - 25 字节]	100	2018-10-01 00:00:00	2018-11-30 00:00:00	3	all	on sale
-----	---	--------------------	-----	---------------------	---------------------	---	-----	---------

Results:

login
Lily@nyu.edu

On note

Barney@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-17 18:00:00	2018-11-29 19:00:00	0	friends	I am awesome
----------------	---	--------------------	-----	---------------------	---------------------	---	---------	--------------

Results:

login
Marshall@nyu.edu

On note

Barney@nyu.edu	1	[GEOMETRY - 25 字节]	100	NULL	NULL	0	all	I am always awesome
----------------	---	--------------------	-----	------	------	---	-----	---------------------

Results:

login
Marshall@nyu.edu

We can compare the result of this part to the former part and see that they are in compliance with each other. (6) Since Marshall get two notes, we can check if this SQL is right by searching key word 'always' in the result. Results:

login	nid	location	radius	stime	endtime	dateofweek	relation	ndesc
Barney@nyu.edu	1	POINT(20 20)	100	NULL	NULL	0	all	I am always awesome

Now only the notes containing 'always' are left.

Project2

I. Overview

1. login

login page:

Application Login Page

Enter your account id:
Enter your password:

Administrator

Enter current time:

As a user we use the upper interface. We can first click sign up button to make our own account:

sign up

login:

password:

name:

sign up

login:

password:

name:

Sign up successfully!

And in the data base the tuple is added:

	login	passwords	uname	location	state
<input type="checkbox"/>	1	1	1	NULL	NULL
<input type="checkbox"/>	Barney@nyu.edu	123456	Barney	[GEOMETRY - 25 字节]	lunch break
<input type="checkbox"/>	H&M	123456	H&M	[GEOMETRY - 25 字节]	NULL
<input type="checkbox"/>	Lily@nyu.edu	123456	Lily	NULL	just chilling
<input type="checkbox"/>	Marshall@nyu.edu	123456	Marshall	[GEOMETRY - 25 字节]	just chilling
<input type="checkbox"/>	Robin@nyu.edu	123456	Robin	[GEOMETRY - 25 字节]	lunch break
<input type="checkbox"/>	Ted@nyu.edu	123456	Ted	[GEOMETRY - 25 字节]	at work
<input type="checkbox"/>	zixiaoli@nyu.edu	1	1	[GEOMETRY - 25 字节]	1

Then we are able to login in to the main page(using [Barney@nyu.edu](#) as example):

username	noteid	description	latitude	longitude	
Ted@nyu.edu	0	good restaurant	40.7	-74	<input type="text" value="comment"/>



2. main page

In the main page first I post the notes user is able to see. Users can use the comment button to submit their comment on the notes. The notes are also shown on the map as yellow points. When click the yellow point users are able to see some contents of the notes (here I choose login and note description) in an alert window.

In the main page there are also interfaces to deal with your account, add friends, change filters and post notes.

localhost:81 显示

Ted@nyu.edu

good restaurant

manage your account

friends

change filters

post notes

Comment result:

username	noteid	description	latitude	longitude	
Ted@nyu.edu	0	good restaurant	40.7	-74	1

[manage your account](#)

make comment successfully!

[return to main page](#)

		login1	login2	nid	cid	cdesc
	 编辑	 复制	 删除	Barney@nyu.edu	Ted@nyu.edu	0 0 1

3. Manage your account

Click the mange your account button on the main page:

your account

login	passwords	uname	latitude	longitude	state
Barney@nyu.edu	123456	Barney	40.7	-74	lunch break

[return to main page](#)

manage account

passwords:

state:

name:

location:



[change state](#)

You are able to see your current account, and also change your account on the lower interface. You can choose your location by click on the map:

manage account

passwords:

state:

name:

location:



[change state](#)

There are also a button to give you choice to go back to the main page.

4. add friends

Click the friends button on the main page:

all friends

friends
Marshall@nyu.edu

your friend request

friends	accept
Ted@nyu.edu	accept

make new friends

friend id:

send request

return to main page




You are able to see all your friends and requests from other users. You can accept the request by click the accept button. You can also send request to other users here.

Click on accept:

accept successfully!

return to main page

And a tuple is added to the database:

<input type="checkbox"/>	 编辑	 复制	 删除	Barney@nyu.edu	Ted@nyu.edu
--------------------------	--	--	--	----------------	-------------

Also, here I add button to give you choice to go back to the main page.

5. change filters

Click the change filters button on the main page:

all filters

login	fid	latitude	longitude	start time	end time	radius	date of week	relation	state	tag
Barney@nyu.edu	0	40.7	-74	2018-12-01 00:00:00	2018-12-31 00:00:00	100	0	friends	lunch break	food
Barney@nyu.edu	1	40.7	-74	2018-12-01 00:00:00	2018-12-31 00:00:00	100	0	all	just chilling	transportation

[return to main page](#)

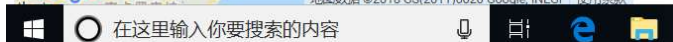
make filters

radius:
start time:
end time:
day of week:
relation:
state:
tag:
location:



Users are able to see their current filters and make new filters. Also they can click on the map to decide the location of their filter:

radius:
start time:
end time:
day of week:
relation:
state:
tag:
location:



And also here I add button to give you choice to go back to the main page.

6. post notes

Click the post notes button on the main page:

all notes

login	nid	latitude	longitude	start time	end time	radius	date of week	relation	description	tag	cid	comment
Barney@nyu.edu	0	40.8	-74	2018-12-01 00:00:00	2018-12-31 00:00:00	100	0	friends	I am awesome.	me		
Barney@nyu.edu	1	40.6	-74			100	0	all	I am always awesome.	me		

[return to main page](#)

post notes

radius:
start time:
end time:
day of week:
relation:
description:
tag:

[make note](#)

Here we can manage our own notes, and post new notes.

Post new notes:

post notes

radius:
start time:
end time:
day of week:
relation:
description:
tag:

[make note](#)

make note successfully!

And a tuple is added to the database(line 3):

	login	nid	location	radius	stime	endtime	dateofweek	relation	ndesc
<input type="checkbox"/> 编辑 复制 删除	Barney@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-12-01 00:00:00	2018-12-31 00:00:00	0	friends	I am awesome.
<input type="checkbox"/> 编辑 复制 删除	Barney@nyu.edu	1	[GEOMETRY - 25 字节]	100	NULL	NULL	0	all	I am always awesome.
<input type="checkbox"/> 编辑 复制 删除	Barney@nyu.edu	2	[GEOMETRY - 25 字节]	0	0000-00-00 00:00:00	0000-00-00 00:00:00	0	all	11111
<input type="checkbox"/> 编辑 复制 删除	H&M	0	[GEOMETRY - 25 字节]	100	2018-10-01 00:00:00	2018-12-30 00:00:00	3	all	on sale
<input type="checkbox"/> 编辑 复制 删除	Robin@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-28 00:00:00	2018-12-28 23:00:00	0	all	traffic jam
<input type="checkbox"/> 编辑 复制 删除	Ted@nyu.edu	0	[GEOMETRY - 25 字节]	100	2018-11-17 18:00:00	2018-12-29 19:00:00	0	friends	good restaurant

II. Project requirements

1. Two interface:

Application Login Page

Enter your account id:

Enter your password:

Log in

sign up

Administrator

Enter current time:

Log in

We can also use the administrator interface, where we can decide current time. After we click on the login button:

username	noteid	description	latitude	longitude		
Barney@nyu.edu	0	I am awesome.	40.8	-74		comment
Barney@nyu.edu	1	I am always awesome.	40.6	-74		comment
Robin@nyu.edu	0	traffic jam	40.8	-73.9		comment
Ted@nyu.edu	0	good restaurant	40.7	-74		comment

manage your account

friends

change filters

post notes



We can see all notes here, i.e., the restrictions are only on the time and location. Since In the main page there are still interfaces to deal with your account, add friends, change filters and post notes, we can adjust our location just like normal users.

2. guard against SQL injection and cross-site scripting attacks
- I use function `mysqli_real_escape_string()` and clean all user inputs to guard against SQL injection and cross-site scripting attacks.

```

function mysqlclean($array, $index, $maxlength, $connection)
{
    if (isset($array["{$index}"]))
    {
        $input = substr($array["{$index}"], 0, $maxlength);
        $input = mysqli_real_escape_string($connection, $input);
        return ($input);
    }
    return NULL;
}

```

3. protection against concurrency

The mysql DB system automatically add read and write lock when select and update the database. There are not sql queries that need other protections in this model so I think use the default setting is enough.