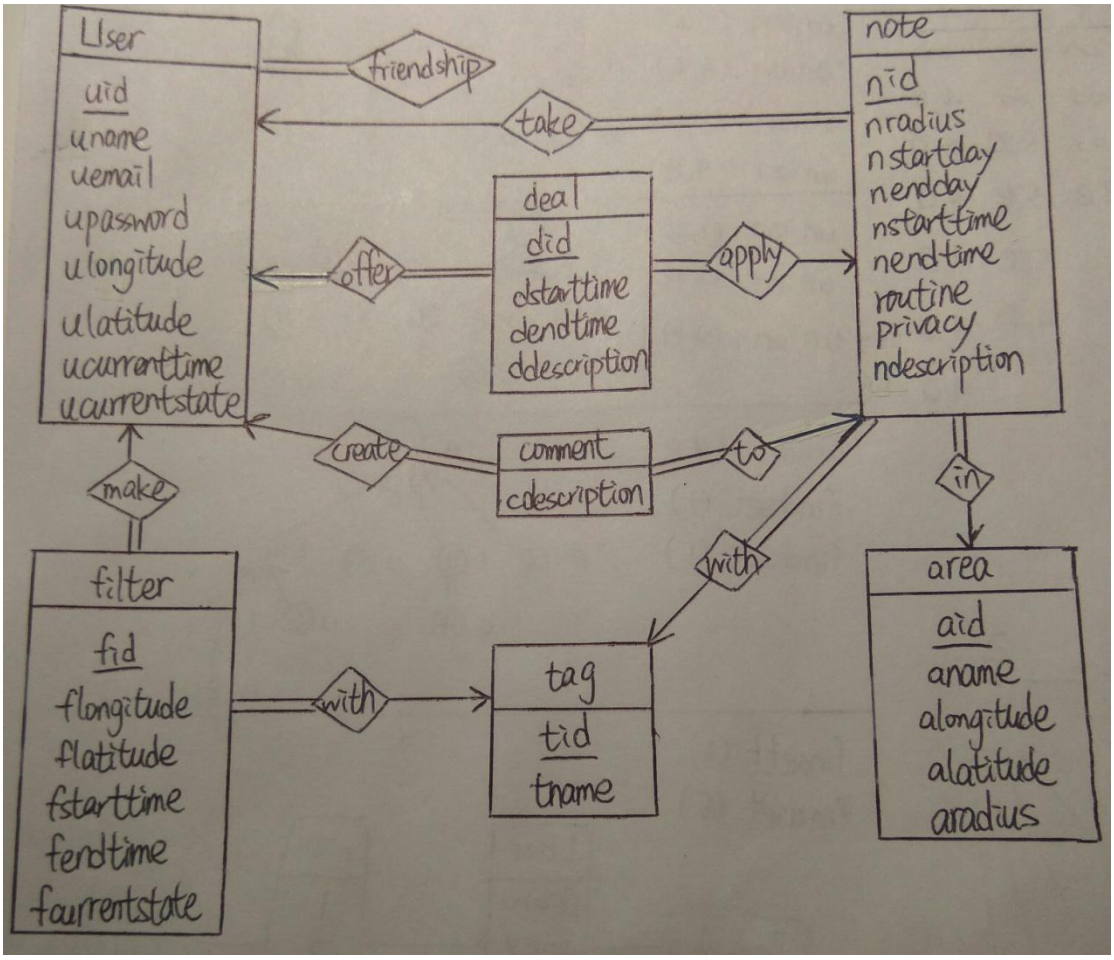
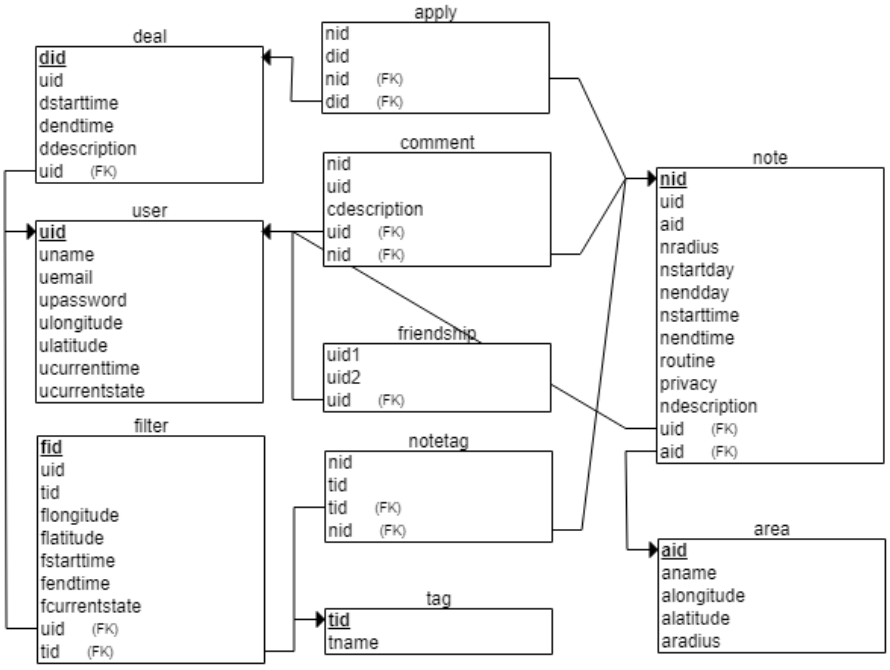
(a)

ER diagram:



Relational schema:



Foreign keys → Referenced Table
filter(uid) → user(uid)
filter(tid) → tag(tid)
friendship(uid1) → user(uid)
friendship(uid2) → user(uid)
note(uid) → user(uid)
note(aid) → area(aid)
comment(uid) → user(uid)
comment(nid) → note(nid)
notetag(nid) → note(nid)
notetag(tid) → tag(tid)
deal(uid) → user(uid)
apply(nid) → note(nid)
apply(did) → deal(did)

user:
uid is the primary key.
uname, uemail, and upassword are user's profile.
ulongitude and ulatitude record user's location once in a while.
ucurrenttime records current time once in a while.
ucurrentstate is user's current status such as "at work", "lunch break", "just chilling", or whatever they choose as the description or default state.

ulongitude, ulatitude, and ucurrenttime will update when a note is posted by a user, when the user performs any action, and also at regular intervals.
A society or business is just another user.

filter:
fid is the primary key.
uid represents the filter's owner.
tid represents the filter's tag.
flongitude and flatitude filter the note's location.
fstarttime and fendtime filter the note's time.
fcurrentstate corresponds to user's status when user creates the filter.

Filters are stored forever and there could be many filters that refer to the same state. Each filter only has one tag. If the user uses two filters, one filter results in the notes {1, 2, 3, 4} and another filter results in the notes {4, 5, 6, 7}. The notes displayed on

the page would be {1, 2, 3, 4, 5, 6, 7}. A user can have multiple filters for a given location and time and state. A filter is active when the user that defined it is in this particular state and in the area specified by the filter. Users can define partial filter. If the state condition is null, that means state does not matter, and that part of the conjunction evaluates to true.

friendship:
Use two uids to record friendship between two people.

tag:
tid is the primary key.
Each tag has a name such as 'tourism', 'shopping', 'food', 'transportation', 'me'.

area:
aid is the primary key.
Take "SoHo" as an example. The area table records the place's name and the definitions of the approximate boundaries by a circle (point plus radius).

note:
nid is the primary key.
uid represents the note's owner.
aid represent the note's area.
nradius represent the note's radius.
nstartday represent the note's start day.
nendday represent the note's end day.
nstarttime represent the note's start time.
nendtime represent the note's end time.
routine represents a schedule when the note can be seen, such as Monday, Tuesday, everyday, null…. People should not be able to use "the second Thursday of each month" or "on Easter Sunday" as a routine.
privacy implements content access restrictions with three types: me, friend, everyone.
ndescription records description related to the notes.

comment:
The comment table records user's comment on notes with the description.

notetag:
Each note can have many tags.

deal:
did is the primary key.
uid represents the deal’s owner.
dstarttime represent the deal’s start time.
dendtime represent the deal’s end time.
The deal table records user’s deal with the description.

apply:
Each note can have many deals.

User can model schedules by the routine column in the note table. Routine can represent a schedule when the note can be seen. If routine is Monday, it means user wants to schedule every Monday. Routine can also be everyday, null…. People should not be able to use "the second Thursday of each month" or "on Easter Sunday" as a routine.

Oingo can model user, filter, and area locations with longitude and latitude.
User’s location is user’s real time location. Filter’s location depends on where do users want the filter locate. Area’s location represents a certain place’s location.

There is also radius column in note and area table. Oingo can model the place’s approximate boundaries by a circle (point plus radius).

(b)
Use MySQL to create the schema with keys, foreign keys, and other constraints.
MySQL file: proj1_create_tables.sql

(c)
1.
INSERT INTO `user` (`uname`, `uemail`, `upassword`, `ulongitude`, `ulatitude`, `ucurrenttime`, `ucurrentstate`) VALUES
('ben', 'ben123@gmail.com', '123ben', '-73.986515 ', '40.694456', '2018-11-28 00:00:00', 'null'),
('andy', 'andy456@gmail.com', '456andy', '-73.986515', '40.694456', '2018-11-28 12:00:00', 'at work'),
('roy', 'roy789@gmail.com', '789roy', '-73.986515', '40.694456', '2018-11-28 15:00:00', 'null');

2.
add areas then add notes:

```
INSERT INTO `area` (`aname`, `alongitude`, `alatitude`, `aradius`) VALUES
('NYU Tandon', '-73.986515', '40.694456', '100'),
('Time Square', '-73.984637', '40.759114', '200');

INSERT INTO `note` (`uid`, `aid`, `nradius`, `nstartday`, `nendday`, `nstarttime`,
`nendtime`, `routine`, `privacy`, `ndescription`) VALUES
('1', '2', '300', '2018-11-28', '2018-11-28', '00:00', '12:00', 'Monday', 'everyone',
NULL),
('2', '1', '100', '2018-11-28', '2018-11-28', '09:00', '21:00', 'everyday', 'me', NULL),
('3', '2', '500', '2018-11-28', '2018-11-28', '12:00', '18:00', 'null', 'friend', NULL);
```

add tags and add to the notes:

```
INSERT INTO `tag` (`tname`) VALUES
('tourism'),
('shopping'),
('food'),
('transportation'),
('me');

INSERT INTO `notetag` (`nid`, `tid`) VALUES
('1', '1'),
('1', '2'),
('1', '3'),
('2', '4'),
('2', '5'),
('3', '1'),
('3', '2');
```

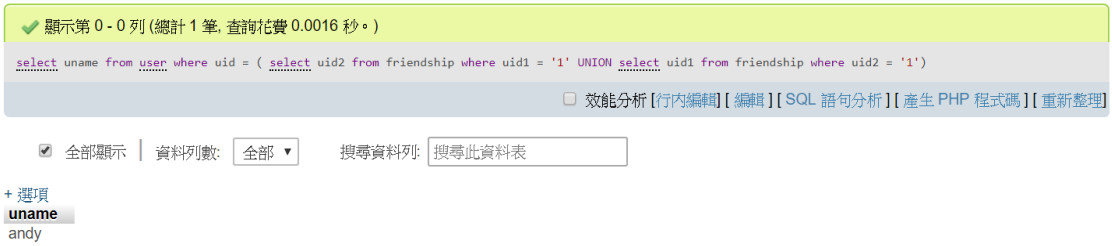
3.

```
INSERT INTO `friendship` (`uid1`, `uid2`) VALUES
('1', '2'),
('2', '3');
```

For a given user ben with uid=1, list all his friends:

```
select uname from user
where uid = (
```

select uid2 from friendship
where uid1 = '1'
UNION
select uid1 from friendship
where uid2 = '1')



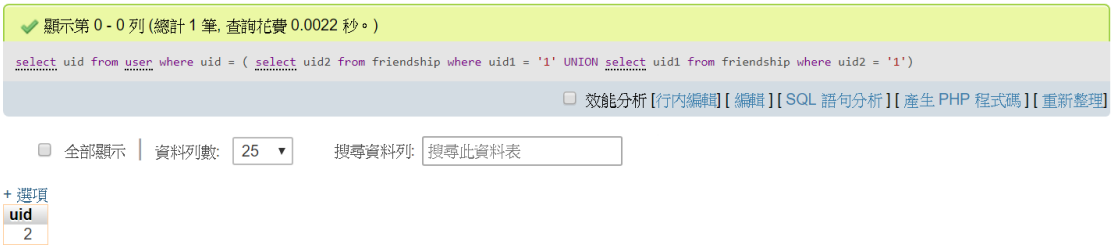
4.

INSERT INTO `filter` (`uid`, `tid`, `flongitude`, `flatitude`, `fstarttime`, `fendtime`, `fcurrentstate`) VALUES
('1', '1', '-73.986515', '40.694456', '2018-11-28 06:00:00', '2018-11-28 18:00:00', 'null'),
('1', '4', '-73.986515', '40.694456', '2018-11-28 06:00:00', '2018-11-28 18:00:00', 'null'),
('2', '1', '-73.986515', '40.694456', '2018-11-28 06:00:00', '2018-11-28 18:00:00', 'null'),
('2', '3', '-73.986515', '40.694456', '2018-11-28 06:00:00', '2018-11-28 18:00:00', 'null'),
('3', '1', '-73.986515', '40.694456', '2018-11-28 06:00:00', '2018-11-28 18:00:00', 'null'),
('3', '3', '-73.986515', '40.694456', '2018-11-28 06:00:00', '2018-11-28 18:00:00', 'null'),
('3', '5', '-73.986515', '40.694456', '2018-11-28 06:00:00', '2018-11-28 18:00:00', 'null');

Given a user ben and his current location, current time, and current state, output all notes that he should currently be able to see given the filters he has set up.

Find who is ben’s friend:
select uid from user
where uid = (
select uid2 from friendship
where uid1 = '1'

UNION
select uid1 from friendship
where uid2 = '1')



Find ben's current location, current time, and current state:
select ulongitude,ulatitude,ucurrenttime,ucurrentstate
from user
where uid='1'

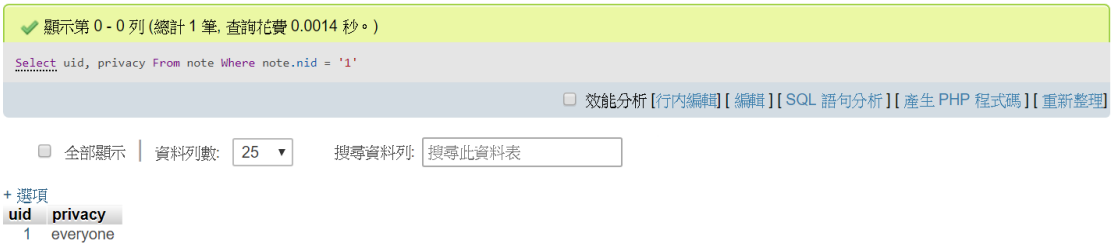
Find the corresponding area:
select aid
from area
where alongitude like '-73.98651%'
and alatitude like '40.69445%'

Use the information above:
Select nid
From note
Where timeDiff('2018-11-28 00:00:00',nstarttime)>0
And timeDiff('2018-11-28 00:00:00',nendtime)<0
And (privacy = 'me'

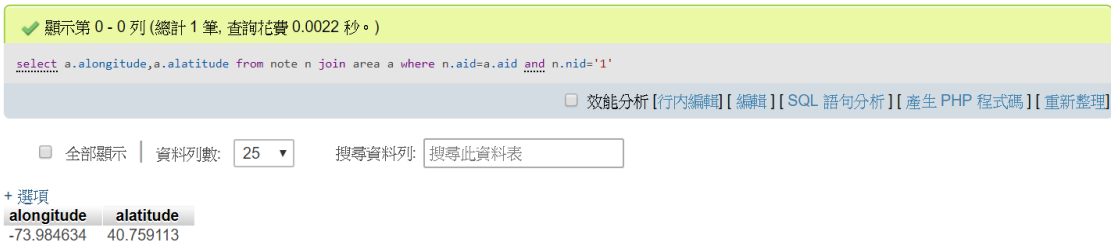
Or (privacy = 'friend' And uid ='2')
Or privacy = 'everyone')
And aid='1'

5.
Given note 1 and the current time, output all users that should currently be able to see this note based on their filter and their last recorded location.

Find the note's owner and privacy:
Select uid, privacy From note Where note.nid = '1'



Find the note's location:
select a.alongitude,a.alatitude
from note n join area a
where n.aid=a.aid
and n.nid='1'

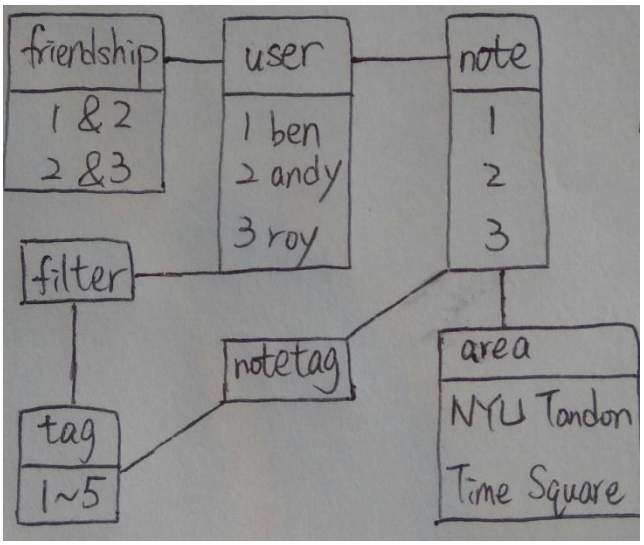


Use the current time and the information above to output all users that should currently be able to see this note (privacy is 'everyone'):
Select uid
From filter
Where timeDiff('2018-11-28 00:00:00',fstarttime)>0
And timeDiff('2018-11-28 00:00:00',fendtime)<0
And flongitude like '-73.98463%'
And flatitude like '40.75911%'

(d)

Use MySQL to generate test cases.

MySQL file: proj1_insert_statements.sql



User table

uid	uname	uemail	upassword	ulongitude	ulatitude	ucurrenttime	ucurrentstate
1	ben	ben123@gmail.com	123ben	-73.986511	40.694454	2018-11-28 00:00:00	null
2	andy	andy456@gmail.com	456andy	-73.986511	40.694454	2018-11-28 12:00:00	at work
3	roy	roy789@gmail.com	789roy	-73.986511	40.694454	2018-11-28 15:00:00	null

Area table

aid	aname	alongitude	alatitude	aradius
1	NYU Tandon	-73.986511	40.694454	100
2	Time Square	-73.984634	40.759113	200

Note table

nid	uid	aid	nradius	nstartday	nendday	nstarttime	nendtime	routine	privacy	ndescription
1	1	2	300	2018-11-28	2018-11-28	00:00:00	12:00:00	Monday	everyone	*NULL*
2	2	1	100	2018-11-28	2018-11-28	09:00:00	21:00:00	everyday	me	*NULL*
3	3	2	500	2018-11-28	2018-11-28	12:00:00	18:00:00	null	friend	*NULL*

Tag table

tid	tname
1	tourism
2	shopping
3	food
4	transportation
5	me

Notetag table

nid	tid
1	1
1	2
1	3
2	4
2	5
3	1
3	2

Friendship table

uid1	uid2
1	2
2	3

Filter table

fid	uid	tid	flongitude	flatitude	fstarttime	fendtime	fcurrentstate
1	1	1	-73.986511	40.694454	2018-11-28 06:00:00	2018-11-28 18:00:00	null
2	1	4	-73.986511	40.694454	2018-11-28 06:00:00	2018-11-28 18:00:00	null
3	2	1	-73.986511	40.694454	2018-11-28 06:00:00	2018-11-28 18:00:00	null
4	2	3	-73.986511	40.694454	2018-11-28 06:00:00	2018-11-28 18:00:00	null
5	3	1	-73.986511	40.694454	2018-11-28 06:00:00	2018-11-28 18:00:00	null
6	3	3	-73.986511	40.694454	2018-11-28 06:00:00	2018-11-28 18:00:00	null
7	3	5	-73.986511	40.694454	2018-11-28 06:00:00	2018-11-28 18:00:00	null

If users want to filter notes based on their current location and the current time, Oingo will compare filter's start time and end time with note's start time and end time. Oingo will also compare filter's longitude and latitude with area's longitude and latitude after using aid to connect note and area. Furthermore, Oingo will check the note's routine and privacy like the cases below.

Case1:
If anyone wants to filter notes for Monday schedule, he can find note 1 since note 1 is open to everyone.

Case2:
If ben wants to filter notes for null schedule, he can only find note 1 since note 2 can only be seen by andy and ben is not roy's friend.
If andy wants to filter notes for null schedule, he can find note 1, 2, 3 since andy is roy's friend.
If roy wants to filter notes for null schedule, he can only find note 1, 3 since note 2 can only be seen by andy.

Case3:
If ben and roy want to filter notes for everyday schedule, they cannot find note 2 since note 2 can only be seen by andy.