Principles of Database Systems (CS6083)

Project 1

April 15, 2019

Mingyu Zhao, Hangbo Fan
New York University
Tandon School of Engineering
m.zhao@nyu.edu, hf9292@nyu.edu

1 Introduction

1.1 Assumption

We have the following assumptions when designing our system:

1. Each email address can only be used to register one account, and users will not be about to change their email since signed up. After created an account, users will login with their email address and password.

2. The *wmtype* shows the role of the member which is either administrator or user in the **Workspace**.

3. Because a user cannot send two messages in the same channel of the sam workspace at a single second, we use (*wid*, *cname*, *uemail*, *mtime*) as the primary key of the **Message** relation.

4. Instead of using trigger in our database, we will use multiple SQL statements if cascading operations are needed.

2 Entity-Relationship Model

Our entity-relationship diagram is modeled as follows:

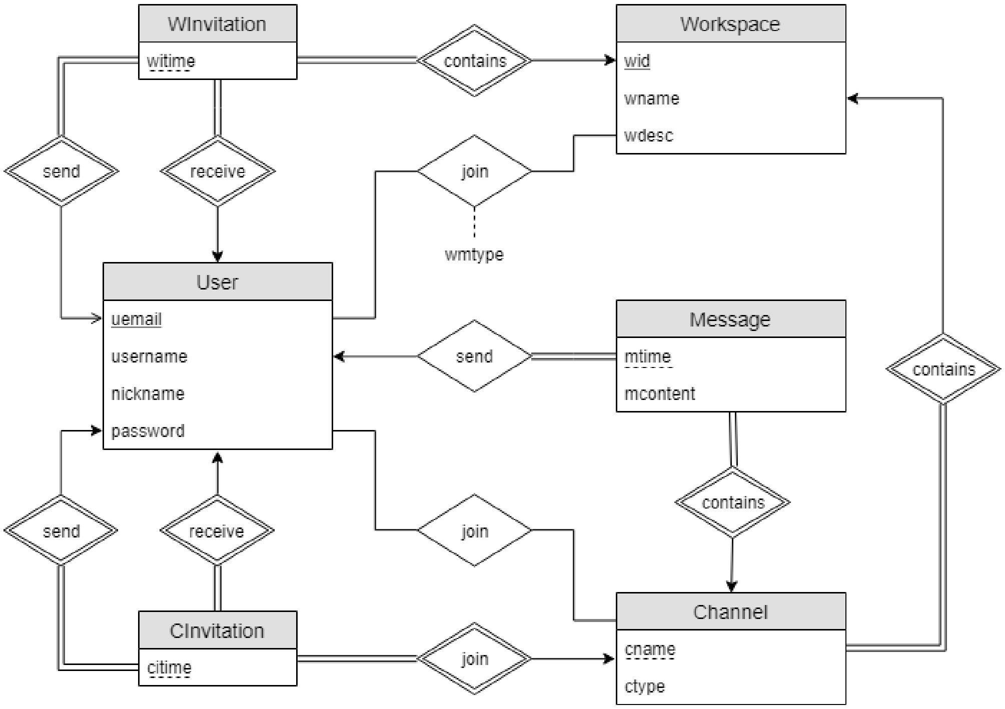


Figure 1: Entity-Relationship Diagram

Here, the **Message**, **Channel**, **CInvitation** and **WInvitation** become weak entities. And the **join** relationship between **User** and **Workspace** has an attribute *wmtype*, indicating the user in a particular workspace is an administrator or a coworker.

3 Relational Schema Design

Translating from the entity-relationship model, we have the following relational schema (the underlined attributes are chosen as the primary keys):

- **User** (uemail, username, nickname, password)
- **Workspace** (wid, wname, wdesc)
- **Channel** (wid, cname, ctype, ctime)
- **Message** (wid, cname, uemail, mtime, mcontent)
- **WInvitation** (semail, remail, wid, witime)
- **CInvitation** (semail, remail, wid, cname, citime)
- **WMember** (uemail, wid, wmtype)
- **CMember** (uemail, wid, cname)

And we also have foreign key constraints among these relations:

- *wid* is a foreign key from **Channel**, referencing **Workspace**
- (*wid*, *cname*) is a foreign key from **Message**, referencing **Channel**
- *uemail* is a foreign key from **Message**, referencing **User**
- *semail* is a foreign key from **WInvitation**, referencing **User**
- *remail* is a foreign key from **WInvitation**, referencing **User**
- *wid* is a foreign key from **WInvitation**, referencing **Workspace**
- *semail* is a foreign key from **CInvitation**, referencing **User**
- *remail* is a foreign key from **CInvitation**, referencing **User**
- (*wid*, *cname*) is a foreign key from **CInvitation**, referencing **Workspace**
- *uemail* is a foreign key from **WMember**, referencing **User**
- *wid* is a foreign key from **WMember**, referencing **Workspace**
- *uemail* is a foreign key from **CMember**, referencing **User**
- (*wid*, *cname*) is a foreign key from **CMember**, referencing **Channel**

The visualized relational schema is also shown below:

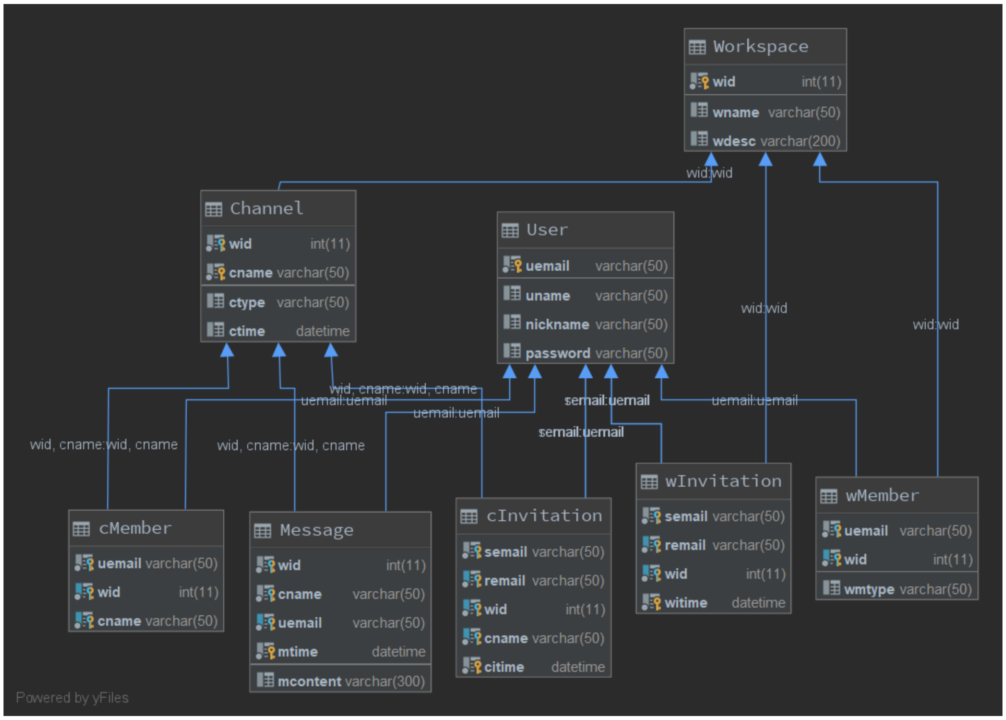


Figure 2: Visualized Relational Schema

4 Database Implementation

For implementation, we are using MySQL (8.0).

4.1 Create Tables

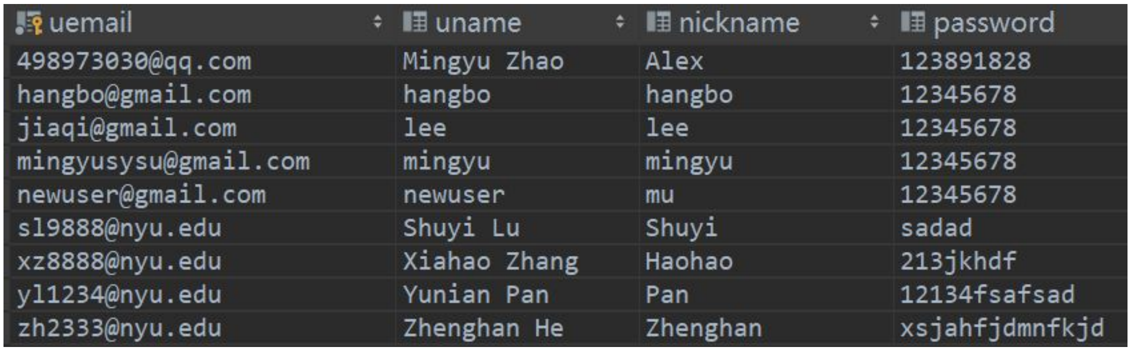
```
create table User (
    uemail varchar(50) not null primary key,
    uname varchar(50) null,
    nickname varchar(50) null,
    password varchar(50) null
);

create table Workspace (
    wid int auto_increment primary key,
    wname varchar(50) null,
    wdesc varchar(200) null
);

create table Channel (
    wid int not null,
    cname varchar(50) not null,
    ctype varchar(50) null,
    ctime datetime null,
    primary key (wid, cname),
    foreign key (wid) references Workspace (wid)
);

```

```sql
create table Message (
    wid int not null,
    cname varchar(50) not null,
    uemail varchar(50) not null,
    mtime datetime not null,
    mcontent varchar(300) null,
    primary key (wid, cname, uemail, mtime),
    foreign key (wid, cname) references Channel (wid, cname),
    foreign key (uemail) references User (uemail)
);

create table cInvitation (
        semail varchar(50) not null,
        remail varchar(50) not null,
        wid int not null,
        cname varchar(50) not null,
        citime datetime not null,
        primary key (semail, remail, wid, cname, citime),
    foreign key (semail) references User (uemail),
    foreign key (remail) references User (uemail),
    foreign key (wid, cname) references Channel (wid, cname)
);

create table cMember (
    uemail varchar(50) not null,
    wid int not null,
    cname varchar(50) not null,
    primary key (uemail, wid, cname),
    foreign key (uemail) references User (uemail),
    foreign key (wid, cname) references Channel (wid, cname)
);

create table wInvitation (
    semail varchar(50) not null,
    remail varchar(50) not null,
    wid int not null,
    witime datetime not null,
    primary key (semail, remail, wid, witime),
    foreign key (semail) references User (uemail),
    foreign key (remail) references User (uemail),
    foreign key (wid) references Workspace (wid)
);

create table wMember (
    uemail varchar(50) not null,
    wid int not null,
    wmtype varchar(50) null,
    primary key (uemail, wid),
    foreign key (uemail) references User (uemail),
    foreign key (wid) references Workspace (wid)
);
```

4.2 Insert Testing Data

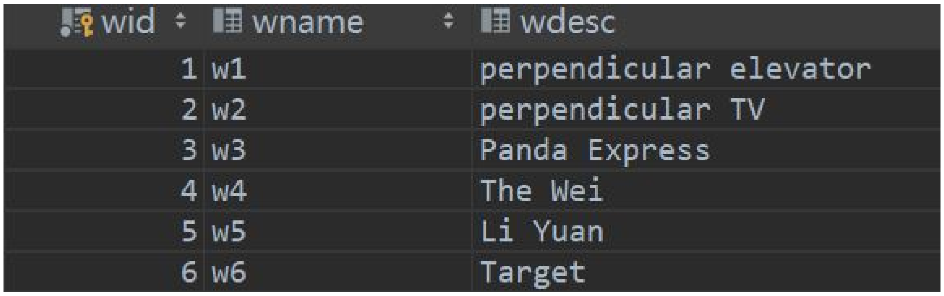
For testing purpose, we generated some sampled data as follows:

- First, we generated 9 users, 6 workspaces and 7 channels with different types and corresponding workspace.
- Then we insert the messages and the relationship between different entities so that we ensure at least some result will show up for each query.



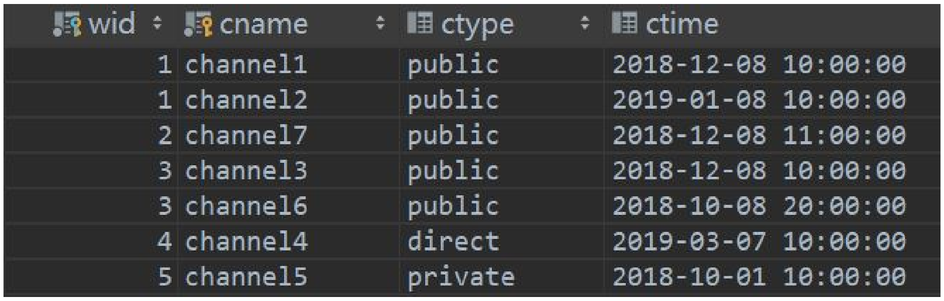
uemail	uname	nickname	password
498973030@qq.com	Mingyu Zhao	Alex	123891828
hangbo@gmail.com	hangbo	hangbo	12345678
jiaqi@gmail.com	lee	lee	12345678
mingyusysu@gmail.com	mingyu	mingyu	12345678
newuser@gmail.com	newuser	mu	12345678
sl9888@nyu.edu	Shuyi Lu	Shuyi	sadad
xz8888@nyu.edu	Xiahao Zhang	Haohao	213jkhdf
yl1234@nyu.edu	Yunian Pan	Pan	12134fsafsad
zh2333@nyu.edu	Zhenghan He	Zhenghan	xsjahfjdmnfkjd

Figure 3: Sampled Data for the **User** Relation



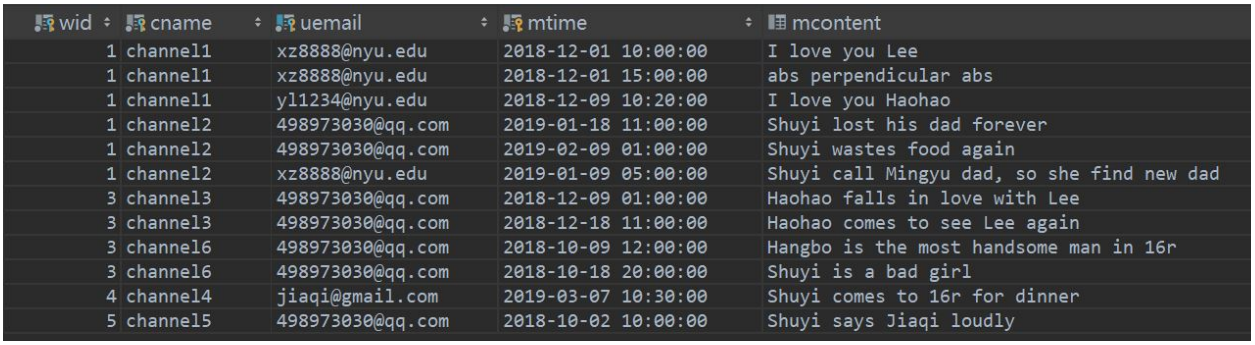
wid	wname	wdesc
1	w1	perpendicular elevator
2	w2	perpendicular TV
3	w3	Panda Express
4	w4	The Wei
5	w5	Li Yuan
6	w6	Target

Figure 4: Sampled Data for the **Workspace** Relation



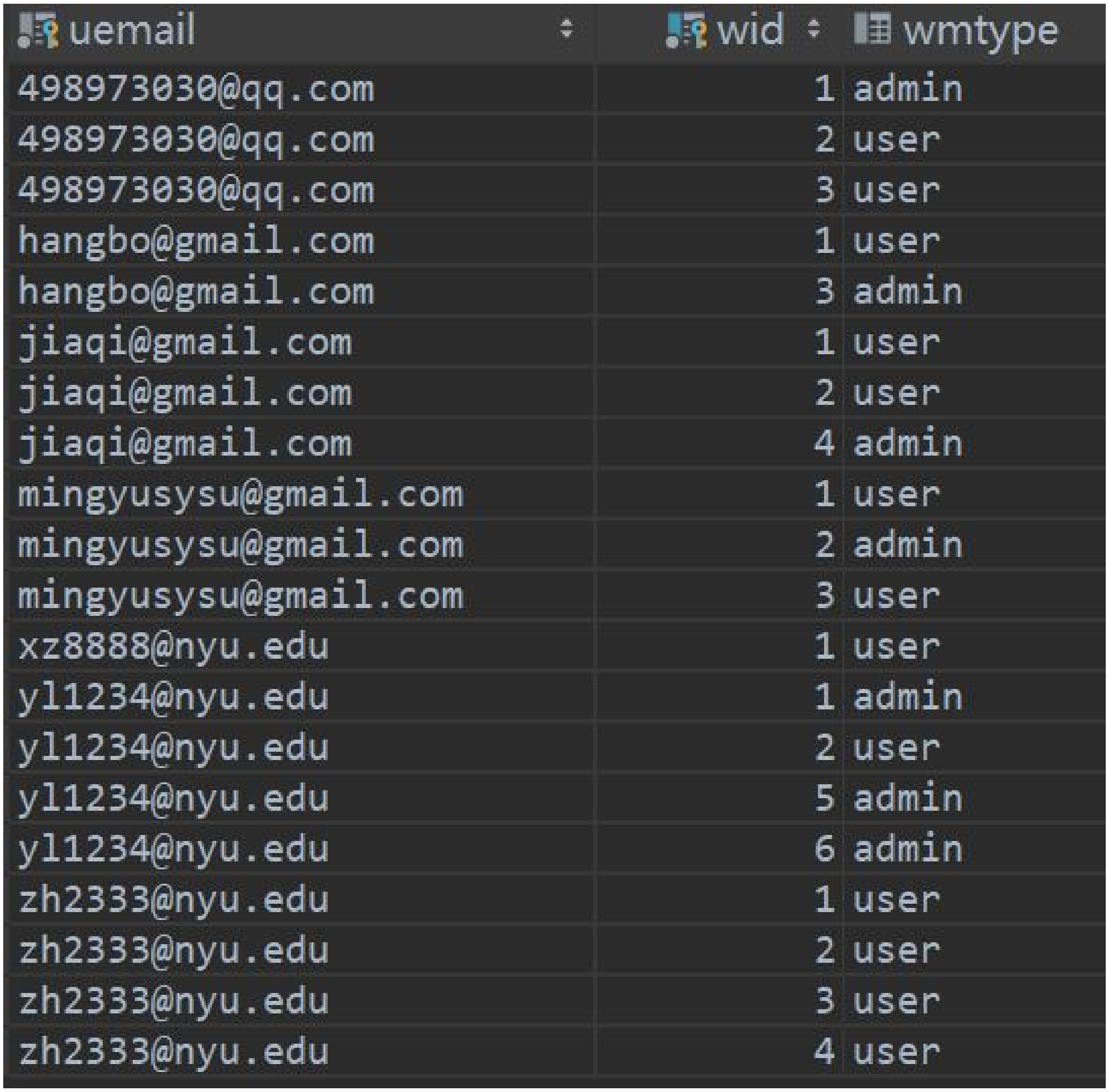
wid	cname	ctype	ctime
1	channel1	public	2018-12-08 10:00:00
1	channel2	public	2019-01-08 10:00:00
2	channel7	public	2018-12-08 11:00:00
3	channel3	public	2018-12-08 10:00:00
3	channel6	public	2018-10-08 20:00:00
4	channel4	direct	2019-03-07 10:00:00
5	channel5	private	2018-10-01 10:00:00

Figure 5: Sampled Data for the **Channel** Relation



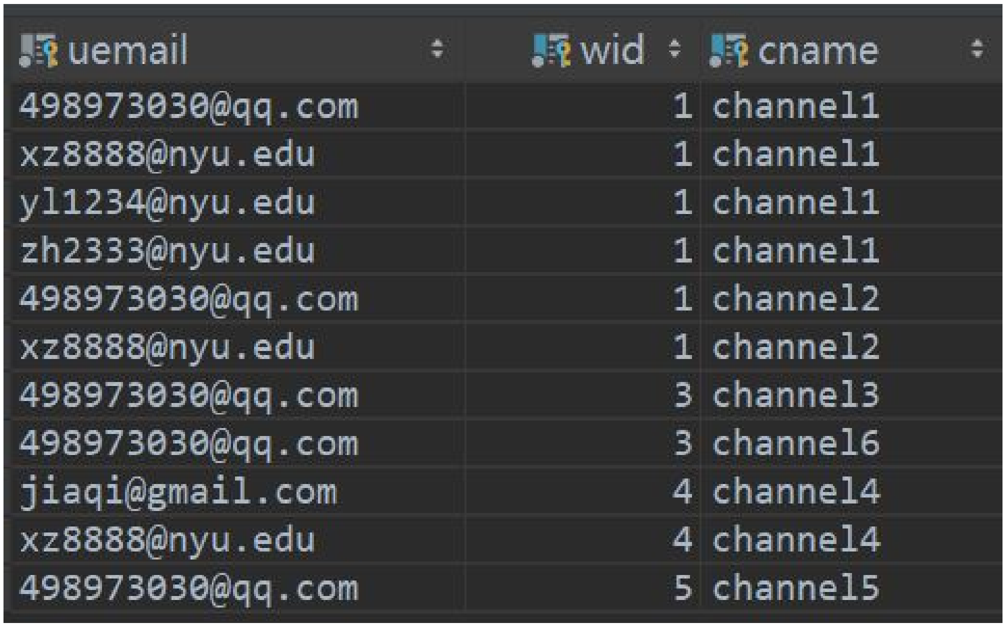
wid	cname	uemail	mtime	mcontent
1	channel1	xz8888@nyu.edu	2018-12-01 10:00:00	I love you Lee
1	channel1	xz8888@nyu.edu	2018-12-01 15:00:00	abs perpendicular abs
1	channel1	yl1234@nyu.edu	2018-12-09 10:20:00	I love you Haohao
1	channel2	498973030@qq.com	2019-01-18 11:00:00	Shuyi lost his dad forever
1	channel2	498973030@qq.com	2019-02-09 01:00:00	Shuyi wastes food again
1	channel2	xz8888@nyu.edu	2019-01-09 05:00:00	Shuyi call Mingyu dad, so she find new dad
3	channel3	498973030@qq.com	2018-12-09 01:00:00	Haohao falls in love with Lee
3	channel3	498973030@qq.com	2018-12-18 11:00:00	Haohao comes to see Lee again
3	channel6	498973030@qq.com	2018-10-09 12:00:00	Hangbo is the most handsome man in 16r
3	channel6	498973030@qq.com	2018-10-18 20:00:00	Shuyi is a bad girl
4	channel4	jiaqi@gmail.com	2019-03-07 10:30:00	Shuyi comes to 16r for dinner
5	channel5	498973030@qq.com	2018-10-02 10:00:00	Shuyi says Jiaqi loudly

Figure 6: Sampled Data for the **Message** Relation



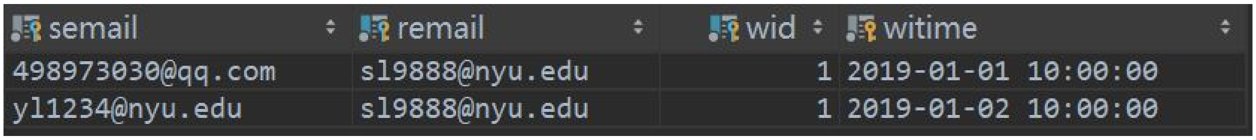
uemail	wid	wmtype
498973030@qq.com	1	admin
498973030@qq.com	2	user
498973030@qq.com	3	user
hangbo@gmail.com	1	user
hangbo@gmail.com	3	admin
jiaqi@gmail.com	1	user
jiaqi@gmail.com	2	user
jiaqi@gmail.com	4	admin
mingyusysu@gmail.com	1	user
mingyusysu@gmail.com	2	admin
mingyusysu@gmail.com	3	user
xz8888@nyu.edu	1	user
yl1234@nyu.edu	1	admin
yl1234@nyu.edu	2	user
yl1234@nyu.edu	5	admin
yl1234@nyu.edu	6	admin
zh2333@nyu.edu	1	user
zh2333@nyu.edu	2	user
zh2333@nyu.edu	3	user
zh2333@nyu.edu	4	user

Figure 7: Sampled Data for the **wMember** Relation



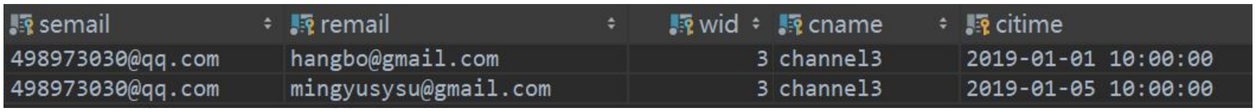
uemail	wid	cname
498973030@qq.com	1	channel1
xz8888@nyu.edu	1	channel1
yl1234@nyu.edu	1	channel1
zh2333@nyu.edu	1	channel1
498973030@qq.com	1	channel2
xz8888@nyu.edu	1	channel2
498973030@qq.com	3	channel3
498973030@qq.com	3	channel6
jiaqi@gmail.com	4	channel4
xz8888@nyu.edu	4	channel4
498973030@qq.com	5	channel5

Figure 8: Sampled Data for the **cMember** Relation



semail	remail	wid	witime
498973030@qq.com	sl9888@nyu.edu	1	2019-01-01 10:00:00
yl1234@nyu.edu	sl9888@nyu.edu	1	2019-01-02 10:00:00

Figure 9: Sampled Data for the **wInvitation** Relation



semail	remail	wid	cname	citime
498973030@qq.com	hangbo@gmail.com	3	channel3	2019-01-01 10:00:00
498973030@qq.com	mingyusysu@gmail.com	3	channel3	2019-01-05 10:00:00

Figure 10: Sampled Data for the **cInvitation** Relation

4.3 Sample Queries

4.3.1 Query 1: Create a new user account, with email, name, nickname, and password.

SQL Query:

```sql
    insert into User
    value ('newuser@gmail.com', 'newuser', 'nu', '12345678');
```

4.3.2 Query 2: Create a new public channel inside a workspace by a particular user.

SQL Query:

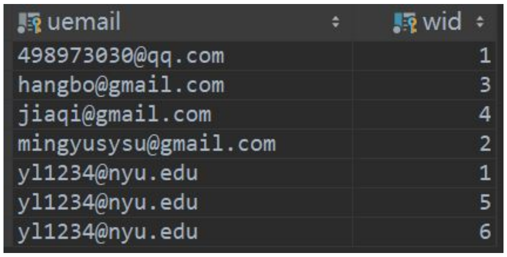
```sql
    insert into Channel
    value (2, 'channel7', 'public', '2018-12-08 11:00:00');
```

4.3.3 Query 3: For each workspace, list all current administrators.

SQL Query:

```sql
    select uemail, wid
    from Workspace natural join wMember
    where wmtype = 'admin'
    group by uemail, wid;
```

Query Result:



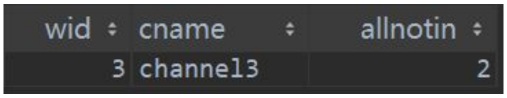
uemail	wid
498973030@qq.com	1
hangbo@gmail.com	3
jiaqi@gmail.com	4
mingyusysu@gmail.com	2
yl1234@nyu.edu	1
yl1234@nyu.edu	5
yl1234@nyu.edu	6

4.3.4 Query 4: For each public channel in a given workspace, list the number of users that were invited to join the channel more than 5 days ago and that have not yet joined.

SQL Query:

```sql
    select wid, cname, count(remail) as allnotin
from cInvitation natural join Channel
where timestampdiff(Day, (date_format(ctime,'%Y%m%d')),date_format(now(),'%Y%m%d')) > 5
    and wid = 3
  and Channel.ctype = 'public'
      and remail not in (
        select uemail
        from cMember
        where wid = 3 and cInvitation.cname = cMember.cname
    )
group by wid, cname;
```

Query Result:



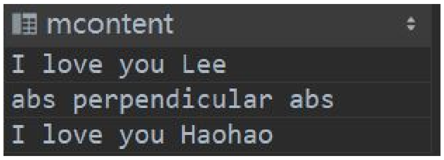
wid	cname	allnotin
3	channel3	2

4.3.5 Query 5: For a particular channel, list all messages in chronological order.

SQL Query:

```
select mcontent
from Channel natural join Message
where cname = 'channel1'
order by mtime;
```

Query Result:



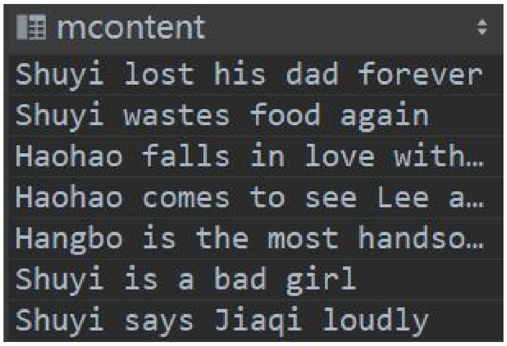
mcontent
I love you Lee
abs perpendicular abs
I love you Haohao

4.3.6 Query 6: For a particular user, list all messages they have posted in any channel.

SQL Query:

```
select mcontent
from Message
where uemail = '498973030@qq.com';
```

Query Result:



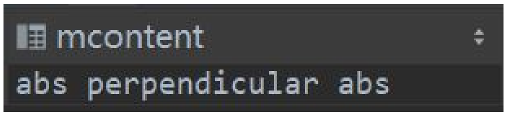
mcontent
Shuyi lost his dad forever
Shuyi wastes food again
Haohao falls in love with…
Haohao comes to see Lee a…
Hangbo is the most handso…
Shuyi is a bad girl
Shuyi says Jiaqi loudly

4.3.7 Query 7: For a particular, list all messages that are accessible to this user and that contain the keyword "perpendicular" in the body of the message.

SQL Query:

```
select mcontent
from Message natural join cMember natural join wMember
where uemail = 'xz8888@nyu.edu' and mcontent like '%perpendicular%';
```

Query Result:



mcontent
abs perpendicular abs