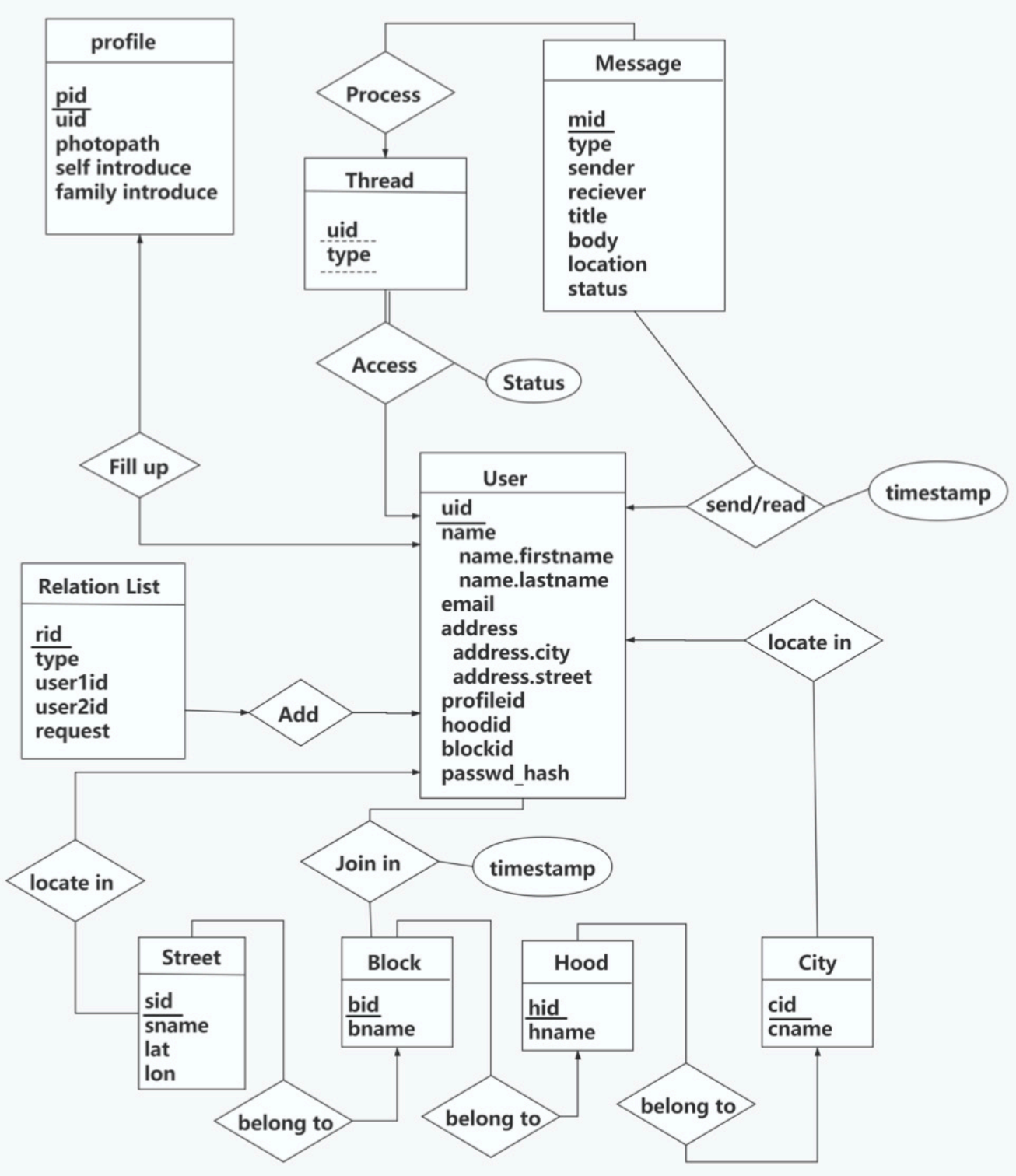
Project Part 1

Siling Liu sl7109

Through the research on project requirements and the research on the location-based social network "nextdoor.com", ER diagram is drawn as follows:



According to the ER diagram drawn, the various entities and relationships involved in the website are explained as follows:

User：

The User represents the registered users of the website and has eight attributes: uid, name, email, address, profileid, hoodid, blockid, password.
The uid is the primary key of the User, which is unique and generated by the database automatically.
The name is a composite attribute consisting of firstname and lastname, which is filled in by the user when registrating.
The email is also filled in by the user directly when the user registers.
The address is also a composite attribute, consisting of city and street.
The profileid is specified by the program when the User registers, and the default value after the User registers is -1, which means empty. The User has not filled in the profile. After the User registers and logs in, a profile row is automatically generated and inserted into the corresponding database, and the generated profileid is filled in the profileid field in the User table.
Blockid is the block number that the user joined, the initial value at registration is -1, and the corresponding value will be filled in after the user successfully joined the block.
The initial value of hoodid at registration is -1. When the user joins a block, it will be modified to the hood to which it belongs. The password stores the hash value of the users' password when they register, while not saving the password in clear text ensures the security of the user's password.

Message：

Message is used to represent the interactive information of the website. There are eight attributes in total: mid, type, sender, receiver, title, body, location and status.

Mid is the primary key of each message, which is unique and generated by the database automatically.
Type is a **multi-valued attribute**, which represents the type of information. Here, there are 7 kinds of messages. The meanings of the Numbers and representatives are as follows:

1 -apply to join a block
2-apply to add a friend
3 -send message to all friends
4 -send message to all neighbors
5 -send message to the entire block
6 -send message to the entire hood
7 -send message to a friend or neighbor(a specific person)

The sender is the user id that sends the message.
The reciever is the user id who has the access to receive message assigned by the user who sent it.
The title is the title of the information, given by the user who posted it.
The body is the content topic of the message, filled in by the user who posted the message.
A message attribute timestamp is generated when the user sends a message, which records the time the message was sent.
Location is an optional field, which is empty by default. The user can choose whether to add the location information when publishing the information, which determines whether to display the city and street information filled in by the user when registering.
Status is used to indicate whether the information has been read by the receiver, with both read and unread values.
0-unread
1-read

Thread:

Thread is used to handle requests for different messages sent by users and is a collection of weak entities, determined by the user id and type. Thread is responsible for processing the information in the mysql database based on user requirements and presenting the results to the user.

The user id represents the user who requests the message.

Type is the type of information requested by users. The back-end of the website establishes different threads to process user information according to different processing needs of users.

Profile:

Profile is used to store the user's profile, which consists of five attributes: pid, uid, photopath, self introduce, and family introduce.

Pid is the primary key of each user profile, which is unique, directly incremented by the program generation.

The uid is the user owner id of the profile. In the profile table, the uid is the foreign key of the table, the User table is the main table, and the id column in the User table is the reference. If a User in the User table logs out the User, the row in which the uid is located in the Profile also needs to be deleted, and the uid column in the Profile can only appear once for each uid, because each User can only have one Profile, thus ensuring data consistency.
Photopath is an image of an avatar uploaded by the user. This property is optional.

Self introduce is a user's personal introduction. , to be filled in by the user.
The user's family introduction shall be filled in by the user.

Relation List:

The Relation List is used for the storage of user friend and neighbor, and has five attributes: rid, type, user1id, user2id, and request.

The rid is the primary key which is unique, directly incrementally generated by the program.
Type represents the type of relationship, and there are two values, respectively 0 and 1, which represent the following meanings:
0 - neighbor
1 - friend
The value of type affects the relationship between user1id and user2id.
If the value of type is 1, it means that two people are friends and user1 applies to add user2. The request field records the specific process of adding friends. The default value of request field is 0, which means that the respondent has not replied to the friend request.
If the value of type is 0, the two are neighbor, and user1 unilaterally adds user2 as neighbor. When user1 sends an application to user2, the value of request is -1, that is, the default value of request is -1.

Where user1id and user2id belong to the foreign key, the User table is taken as the main table, and the id column is taken as the reference. The values of user1id and user2id must depend on the uid column of the User table

City Hood Block Street

The four positional models are defined from large to small.
Use the corresponding methods of id and name to store the location information data in mysql database, and change the references to strings in other tables from strings to Numbers, saving the storage space in the database.

Use longitude and latitude coordinates to locate the specific location of each street for use on subsequent maps.

The ER diagram in the figure above is transformed into the corresponding **relational schema** , as follows:

```
User(uid , name , email , address , profileid , hoodid , blockid,password)
Message(mid , type , sender , receiver , title , body , timestamp , location ,
status)
Profile(pid , uid , photopath , self introduce , family introduce)
Relation_List(rid , type , user1id , user2id , request)
Hood_and_Block(hbid , hid , bid)
Block_and_Street(csid,cid,sid)
Hood(hid,hname)
Block(bid,bname)
City(cid,cname)
Street(sid,sname)
```

Using mysql as the backend database for your website, the following sections describe the database and table construction statements.

create database

```
create DATABASE Web;
```

create table

```
#uid , name , email , address , profileid , hoodid , blockid
CREATE TABLE IF NOT EXISTS `User`(
   `uid` INT UNSIGNED AUTO_INCREMENT,
   `user_name` VARCHAR(40) NOT NULL,
   `email` VARCHAR(40) NOT NULL,
    `address_city_id` INT NOT NULL,
    `address_street_id` INT NOT NULL,
    `profile_id` INT NOT NULL DEFAULT -1,
    `hood_id` INT NOT NULL DEFAULT -1,
    `block_id` INT NOT NULL DEFAULT -1,
    `password` VARCHAR(80) NOT NULL,

   PRIMARY KEY ( `uid` )
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

#Message(mid , type , sender , receiver , title , body , timestamp , location ,
status)
CREATE TABLE IF NOT EXISTS `Message`(
   `mid` INT UNSIGNED AUTO_INCREMENT,
   `type` INT NOT NULL,
   `sender_id`  INT NOT NULL,
    `receiver_id` INT NOT NULL,
    `title` VARCHAR(40) ,
    `body` VARCHAR(200) ,
```

```
    `timestamp` TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    `location` bool ,
   `status`  INT NOT NULL,
   PRIMARY KEY ( `mid` )
   foreign key(sender_id)  REFERENCES User(uid)
   foreign key(receiver_id)  REFERENCES User(uid)
   ON DELETE RESTRICT
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

#Profile(pid , uid , photopath , self introduce , family introduce)
CREATE TABLE IF NOT EXISTS `Profile`(
   `pid` INT UNSIGNED AUTO_INCREMENT,
   `` INT NOT NULL,
    `photopath` VARCHAR(200) NOT NULL DEFAULT "***",
    `self_intro` VARCHAR(200) NOT NULL,
    `family_intro` VARCHAR(200) NOT NULL,
   PRIMARY KEY ( `pid` )
    foreign key(uid)  REFERENCES User(uid)
    ON DELETE RESTRICT
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER  TABLE Relation_List CHANGE  rid;
# Relation_List(rid , type , user1id , user2id , request)
CREATE TABLE IF NOT EXISTS `Relation_List`(
   `rid` INT UNSIGNED AUTO_INCREMENT,
   `type` INT NOT NULL,
   `user1_id` INT NOT NULL,
   `user2_id` INT NOT NULL,
   `request` INT NOT NULL  DEFAULT -1,
   PRIMARY KEY ( `rid` )
    foreign key(user1_id)  REFERENCES User(uid)
   foreign key(user2_id)  REFERENCES User(uid)
   ON DELETE RESTRICT
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

#Hood_and_Block(hbid , hid , bid)
CREATE TABLE IF NOT EXISTS `Hood_and_Block`(
   `hbid` INT UNSIGNED AUTO_INCREMENT,
   `hid` INT NOT NULL,
    `bid` INT NOT NULL,
   PRIMARY KEY ( `hbid` )
    foreign key(hid)  REFERENCES Hood(hid)
   foreign key(bid)  REFERENCES Block(bid)
   ON DELETE RESTRICT
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
#Block_and_Street(bsid,cid,sid)
CREATE TABLE IF NOT EXISTS `Block_and_Street`(
   `bsid` INT UNSIGNED AUTO_INCREMENT,
   `bid` INT NOT NULL,
    `sid` INT NOT NULL,
   PRIMARY KEY ( `cbid` )
    foreign key(sid)  REFERENCES Street(sid)
   foreign key(bid)  REFERENCES Block(bid)
    ON DELETE RESTRICT
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

#Hood(hid,hname)
CREATE TABLE IF NOT EXISTS `Hood`(
   `hid` INT UNSIGNED AUTO_INCREMENT,
    `hname` VARCHAR(40) NOT NULL,
   PRIMARY KEY ( `hid` )
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

#Block(bid,bname)
CREATE TABLE IF NOT EXISTS `Block`(
   `bid` INT UNSIGNED AUTO_INCREMENT,
    `bname` VARCHAR(40) NOT NULL,
   PRIMARY KEY ( `bid` )
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

#City(cid,cname)
CREATE TABLE IF NOT EXISTS `City`(
   `cid` INT UNSIGNED AUTO_INCREMENT,
    `cname` VARCHAR(40) NOT NULL,
   PRIMARY KEY ( `cid` )
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

#Street(sid,sname)
CREATE TABLE IF NOT EXISTS `Street`(
   `sid` INT UNSIGNED AUTO_INCREMENT,
    `sname` VARCHAR(40) NOT NULL,
   PRIMARY KEY ( `sid` )
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

design test data

According to the test data required by the website, different test data are designed for different entities as follows:

User:

#uid、name、email、profileid、streetid、blockid、hoodid、cityid、password、passwd_hash

uid	name	email	profileid	streetid	blockid	hoodid	cityid	password
1	James Smith	21322@gmail.com	1	3	6	3	1	123456a.
2	Robert Wilson	test@gmail.com	2	1	2	1	1	qweasd.
3	Paul Cook	third@gmail.com	3	2	3	1	1	zxcasdqwe
4	Linda Wood	lindaW@gmail.com	4	1	2	1	1	zxcvbnm.
5	Susan Hall	Susan@gmail.com	5	1	2	1	1	ujmyhnb.
6	Jean White	echoHelloWorld@163.com	-1	1	-1	-1	1	123456.
7	lily zhu	test@163.com	-1	1	2	1	1	qazwsx.

Profile:

#pid，uid，photopath、self introduce、family introduce

pid	uid	photopath	self introduce	family introduce
1	1	test.jpg	i am James Smith	single
2	2	test.jpg	i am Robert Wilson	single
3	3	test.jpg	i am Paul Cook	married
4	4	test.jpg	i am Linda Wood	single
5	5	test.jpg	i am Susan Hall	married

Message

#mid、type、sender、receiver、title、body、location、timestamp 、status

mid	type	senderid	receiverid	title	body	location	status
1	1	6	2				0
2	1	6	4				0
3	1	6	5				0
4	2	2	3				0
5	3	2	4	test1	content1	1	0
6	3	2	5	test2	content2	0	0
7	4	1	2	haha	this is a test	1	0
8	3	3	2	test4	content4	1	0
9	5	4	2	test5	content5	1	0
10	6	4	2	test6	content6	1	0

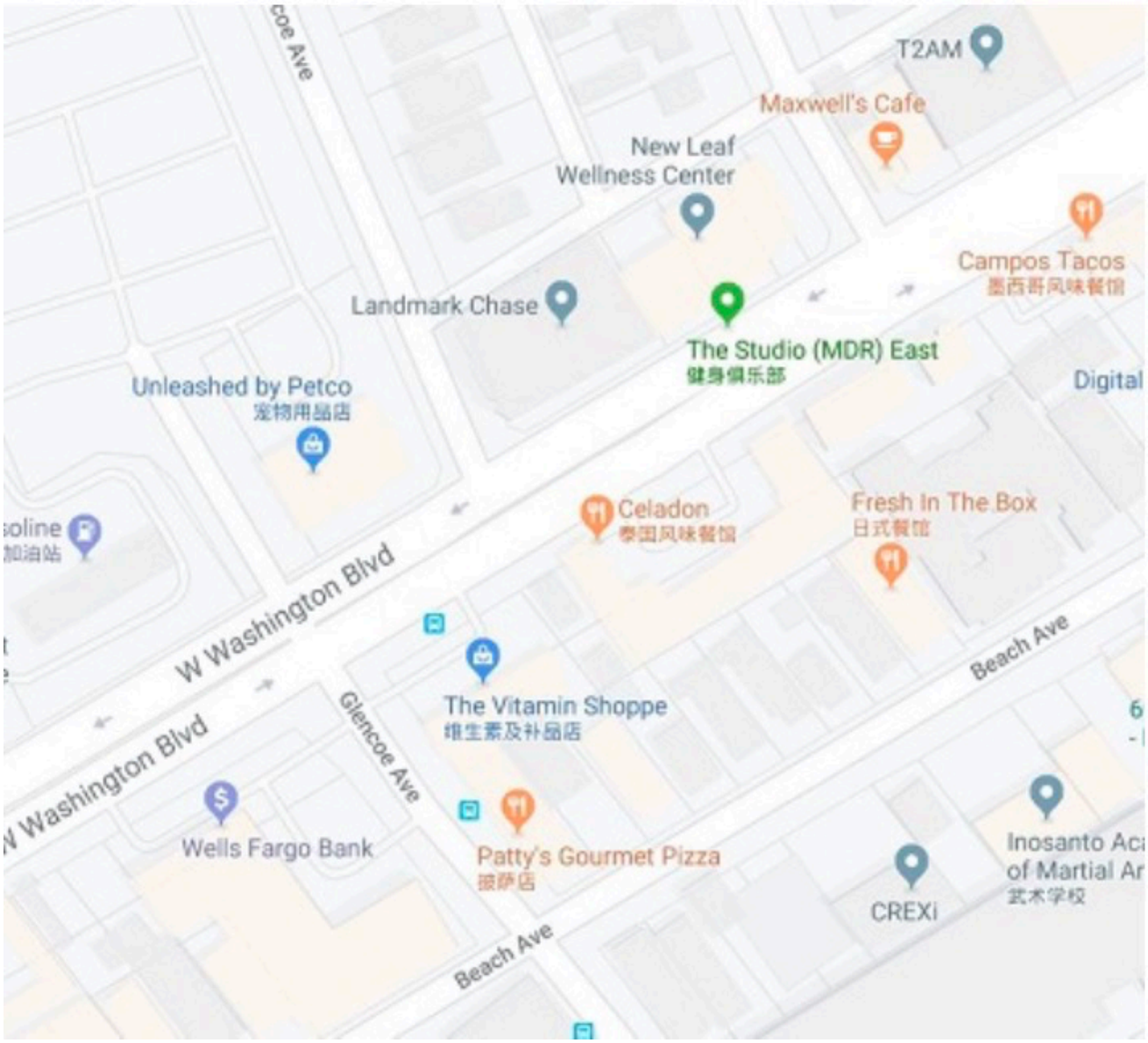
Relation List:

#rid、type、user1id、user2id、request

rid	type	user1id	user2id	request
1	1	2	4	1
2	1	2	5	1
3	1	2	3	-1
4	0	2	1	-1
5	0	3	4	-1
6	0	1	2	-1
7	1	3	2	-1

City Hood Block Street

Randomly select a region in Google map as the test data source, and the selected region screenshot is as follows:



First, list the cities to be selected:

cityid	cityname
1	Los Angeles
2	New York
3	Chicago
4	Detroit

The table below lists neighborhood options:

hoodid	hoodname
1	hood1
2	hood2
3	hood3

The table below lists block options:

blockid	blockname
1	New Leaf Wellness Center
2	Pause Studio
3	The Studio (MDR) East
4	Unleashed by Petco
5	OneWest Bank
6	Marina Collection

The table below lists street options:

hoodid	hoodname
1	Washington Blvd
2	Glencoe Ave
3	Beach Ave

The following table lists the ownership of hood and block

hbid	hoodid	Block id
1	1	1
2	1	2
3	1	3
4	2	4
5	2	5
6	3	6

The following table lists the ownership of block and street

bsid	bid	sid
1	1	1
2	1	2
3	2	1
4	2	2
5	3	1
6	3	2
7	4	1
8	4	2
9	5	1
10	5	2
11	6	3

After importing the above information into the database, write SQL test statements for testing

user login

```
INSERT INTO User (user_name , email , profile_id, address_street_id,
block_id,hood_id, address_city_id ,password) VALUES  ("Robert
Wilson","test@gmail.com",2,1,2,1,1,"$2y$10$KAhc2k5zGvx1PtBQDYkCve84kOK5Ex3pbNQ3Sr
0fLYhC6ewOCyeQG");
```

user view profile

```
select photopath, self_intro,family_intro from Profile where  uid= ?
```

user edit profile

```
INSERT INTO Profile (uid , photopath , self_intro, family_intro) VALUES  (
1,"test.jpg","i am  James Smith","single" );
```

user add a friend

```
INSERT INTO Relation_List (type , user1_id , user2_id, request) VALUES  (
1,2,4,1);
```

user add a neighbor

```
INSERT INTO Relation_List (type , user1_id , user2_id, request) VALUES  (0,2,1,0
);
```

change the reading status of a message

```
UPDATE Message SET status = 0 WHERE mid = 10;
```

query all the friends of a user

```
mysql> select * from Relation_List where type =1 and request =1 and user1_id = 2
;
+-----+------+----------+----------+---------+
| fid | type | user1_id | user2_id | request |
+-----+------+----------+----------+---------+
|   1 |    1 |        2 |        4 |       1 |
|   2 |    1 |        2 |        5 |       1 |
+-----+------+----------+----------+---------+
2 rows in set (0.00 sec)

mysql> select * from Relation_List where type =1 and request =1 and user2_id = 2
;
Empty set (0.00 sec)

mysql>
```

query all the neighbors of a user

```
mysql> select * from Relation_List where type =0 and user1_id = 2;
+-----+------+----------+----------+---------+
| fid | type | user1_id | user2_id | request |
+-----+------+----------+----------+---------+
|   4 |    0 |        2 |        1 |       0 |
+-----+------+----------+----------+---------+
1 row in set (0.00 sec)

mysql>
```

query unread messages from all friends of a user

```
select sender_id,mid,title,body,location from Message where type = 3 and  status
= 0 and receiver_id =?
```

query unread messages from all neighbors of a user

```
select sender_id,mid,title,body,location from Message where type = 4 and  status
= 0 and receiver_id =?
```

query unread messages of user's block

```
select sender_id,mid,title,body,location from Message where type = 5 and  status
= 0 and receiver_id =?
```

query unread messages of user's hood

```
select sender_id,mid,title,body,location from Message where type = 6 and  status
= 0 and receiver_id =?
```

one user sends a friend request to another

```
INSERT INTO Relation_List(type ,user1_id, user2_id) VALUES  (1,?,?)
```

When a user rejects another user's friend request, the corresponding friend request information is deleted

```
DELETE FROM Relation_List where type =1 and request = -1 and user2_id=? and
user1_id=?
```

When a user agrees to another user's friend request, the user modifies the corresponding friend request information and sets the status of the request to pass

```
UPDATE Relation_List SET request = 1 WHERE  user2_id=? and user1_id=?
```

user adds another user in the same block as his/her neighbor

```
INSERT INTO Relation_List(type,user1_id, user2_id) VALUES  (0,?,? )
```

Search all unread messages for that user based on the keywords that the user entered

```
select sender_id,mid,title,body,location from Message where   status = 0 and
receiver_id =? and body like '%?%'
```

user sends a message to all his/her friend

```
INSERT INTO Message (type , sender_id , receiver_id, title,body,location,status)
VALUES  (3,?,?,"","",?,0);
```

Here are some specific SQL functions:

```
#connect database
function conn_db()
{
$conn = mysqli_connect('127.0.0.1','xxx','xxx');
mysqli_select_db($conn,'Web');
return $conn;
}

#get city name by city id
function get_city_name($Cid)
{

    $conn = conn_db();
    $re = "";

    $stmt = $conn->prepare("select cname from City where cid=? ");
    $stmt->bind_param("i", $Cid);

    // bind para
    $cid = $Cid;
    $stmt->execute();
    $stmt->store_result();  //store result;
```

```
    if($stmt->num_rows==0)    // get re
     {
         //chenck next type;
         echo "cannnot fin city  ".$Cid;
         exit();
     }else{
        //  bind var
        $stmt->bind_result($cname);
        while ($stmt->fetch())
        {
            //save re
          $re = $cname;
        }
     }
    return $re;
}

#get user name by user name
function get_user_name($Uid)
{
    $conn = conn_db();
    $re = "";

    $stmt = $conn->prepare("select user_name from User where uid=? ");
    $stmt->bind_param("i", $uid);

    // bind para
    $uid = $Uid;
    $stmt->execute();
    $stmt->store_result();  //store result;

    if($stmt->num_rows==0)    // get all result
     {
         //chenck next type;
         echo "cannnot find user name  ".$Uid;
         exit();
     }else{
        //  bind result  to var
        $stmt->bind_result($uname);
        while ($stmt->fetch())
        {
            // save re
          $re = $uname;
        }
     }
```

```
     return $re;
}

#user login to web
function register_db($name,$Email,$City_id,$Street_id,$Passwd_hash)
{
    $conn = conn_db();

    $City_id = intval($City_id);
    $Street_id = intval($Street_id);


    $stmt = $conn->prepare("INSERT INTO User(user_name,email, address_city_id ,
address_street_id ,password) VALUES  ( ?,?,?,?,? ) ");
    $stmt->bind_param("ssiis",
$user_name,$email,$city_id,$street_id,$password_hash);

    //bind para and execute
    $user_name = $name;
    $email = $Email;
    $city_id = $City_id;
    $street_id = $Street_id;
    $password_hash = $Passwd_hash;
    $stmt->execute();

     if($stmt)
     {
         //echo yes
         return 1;
     }else{
        // echo no
         return 0;
     }

    $stmt->close();
    $conn->close();
}

#add a member as friend
function add_neighbor_rela($req,$sender)
{
    $conn = conn_db();

    $stmt = $conn->prepare("INSERT INTO Relation_List(type,user1_id, user2_id)
VALUES  (0,?,? ) ");
    $stmt->bind_param("ii", $sender,$rid);
```

```
    $sid = $sender;
    $rid = $req;

    $stmt->execute();

     if($stmt)
     {
         //echo yes
         return 1;
     }else{
        // echo no
         return 0;
     }

}
```