2. Designing the tables and using the oracle DBMS

```
create table account (id NUMBER,
        email VARCHAR2(50) not null,
        pwd VARCHAR2(50) not null,
        PRIMARY KEY (id),
        UNIQUE(email));

create table users (userid NUMBER,
                first_name VARCHAR2(50) not null,
                last_name VARCHAR2(50) not null,
                email VARCHAR2(50),
                country VARCHAR2(50) not null,
                zip NUMBER not null,
                regDate DATE not null,
                regTime TIMESTAMP not null,
                status VARCHAR2(50) not null,
                PRIMARY KEY (userid),
                FOREIGN KEY (userid) REFERENCES account(id)
                ON DELETE CASCADE
                );

create table lgroup (gid NUMBER,
                 gname VARCHAR2(50) not null,
                 PRIMARY KEY (gid),
                 FOREIGN KEY (gid) REFERENCES account(id)
                 ON DELETE CASCADE
                 );

create table company (cid NUMBER,
                 cname VARCHAR2(50) not null,
                 PRIMARY KEY (cid),
                 FOREIGN KEY (cid) REFERENCES account(id)
                 ON DELETE CASCADE
                 );

create table usergrp (userid NUMBER not null,
                 grpid NUMBER not null,
                 datetime TIMESTAMP not null,
                 PRIMARY KEY (userid,grpid),
                 FOREIGN KEY (userid) REFERENCES users(userid),
                 FOREIGN KEY (grpid) REFERENCES lgroup(gid)
                 ON DELETE CASCADE
                 );
```

```
create table follower (userid NUMBER not null,
                cid NUMBER not null,
                datetime TIMESTAMP not null,
                PRIMARY KEY (userid,cid),
                FOREIGN KEY (userid) REFERENCES users(userid),
                FOREIGN KEY (cid) REFERENCES company(cid)
                ON DELETE CASCADE
                );

create table user_conn (userid NUMBER NOT NULL,
                connectid NUMBER NOT NULL,
                PRIMARY KEY (userid,connectid),
                FOREIGN KEY (userid) REFERENCES users(userid),
                FOREIGN KEY (connectid) REFERENCES users(userid)
                ON DELETE CASCADE
                );

create table resources (rsid NUMBER,
                r_type VARCHAR2(50) NOT NULL,
                r_link VARCHAR2(50) NOT NULL,
                PRIMARY KEY (rsid)
                );

create table user_resource (userid NUMBER NOT NULL,
                rsid NUMBER NOT NULL,
                PRIMARY KEY (userid,rsid),
                FOREIGN KEY (userid) REFERENCES users(userid),
                FOREIGN KEY (rsid) REFERENCES resources(rsid)
                ON DELETE CASCADE
                );


create table post ( pid NUMBER,
                sender_id NUMBER not null,
                receiver_id NUMBER not null,
                post_type VARCHAR2(50) not null,
                post_content VARCHAR2(50),
                share_type VARCHAR2(50) not null,
                attach_res NUMBER,
                datetime TIMESTAMP not null,
                PRIMARY KEY (pid),
                FOREIGN KEY (sender_id) REFERENCES account(id),
                FOREIGN KEY (receiver_id) REFERENCES account(id),
                FOREIGN KEY (attach_res) REFERENCES resources(rsid)
```

```
);

create table comments ( commentid number,
        sender_id NUMBER not null,
        pid NUMBER not null,
        c_content VARCHAR2(50),
        islike NUMBER,
        isshare NUMBER,
        PRIMARY KEY(commentid),
        FOREIGN KEY (sender_id) REFERENCES account(id),
        FOREIGN KEY (pid) REFERENCES post(pid)
        );
```

4. Table design

Inner join: It is a complete join i.e it checks for matches and returns rows. If there are multiple matching rows in the second table, multiple rows will be returned. Also, when two tables are joined, columns can be returned from either.

IN: It is a semi-join, a join that checks for matches but does not return rows. If there are multiple matching tables in the result only one disticnt row from the first table will be returned. Also, because the rows are not returned, columns from the table referenced in the IN cannot be returned.

For large data ser if we need to check for matching rows in the other tables but don't need any columns from that table, we should use IN and if we need columns from the second table, use Inner Join.

To improve performance by reducing the number of tables by Normalization and using partitioned views with triggers to support dynamic content changes to be reflected in views generated and if tables values are in ordered set IN performs better than Inner Join.

Assumptions

1) Query 2 - find_names.sql : Group Name USC is considered.

2) Query 4 - find_posts_without_file.sql: year 2014 considered.

3) Query 7 - get_comments_names.sql: 2014 considered.

4) Query 8 - find_registers.sql: 2013 considered and Time format HH24:Mi:SS.