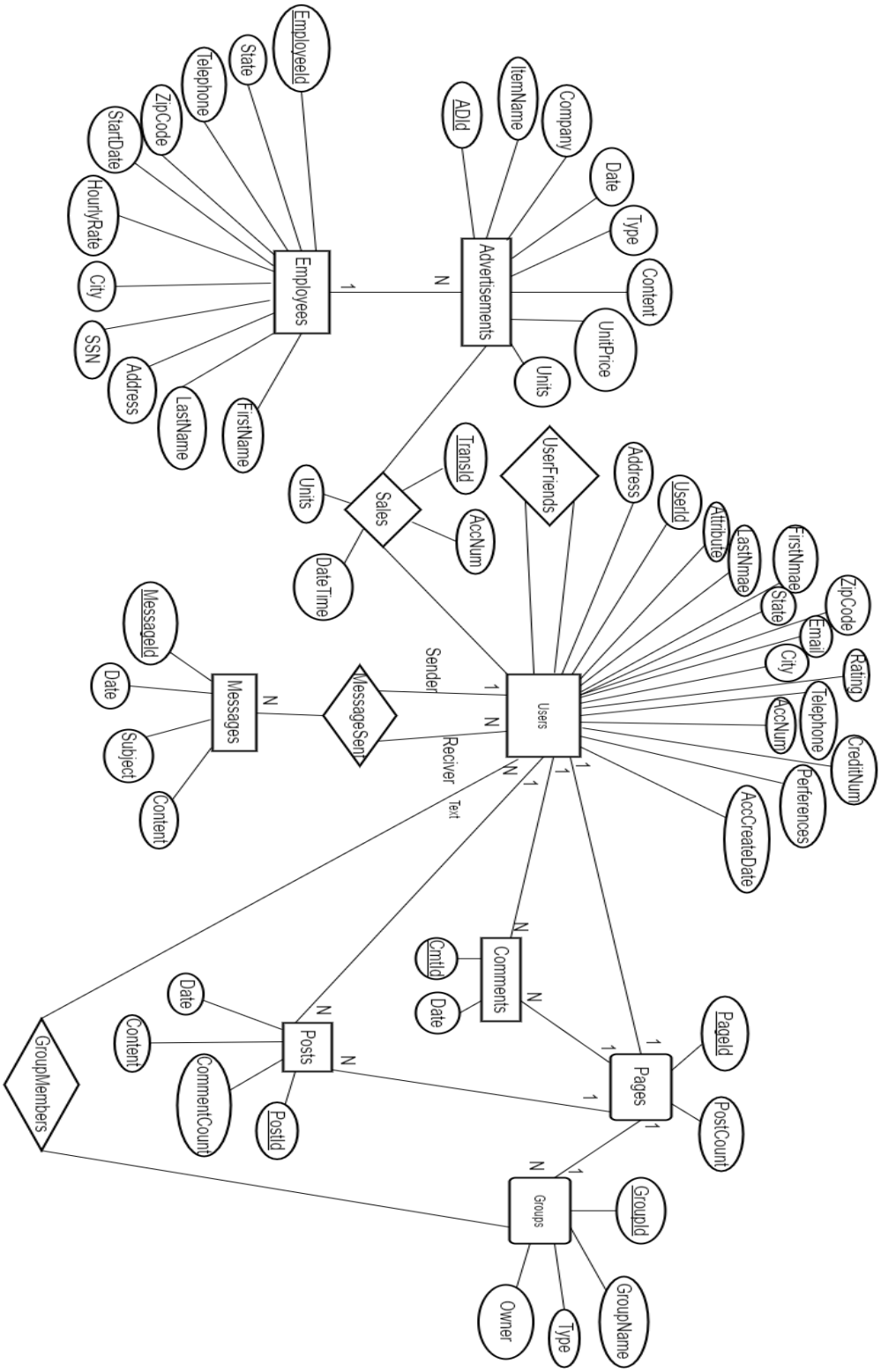
Team/Website Name: WithFriends

Team: Daniel Sha (danielsha94@gmail.com), Kaile Ye (kaile.ye@stonybrook.edu), Yida Yuan (yida.yuan@stonybrook.edu)



E-R Diagram Rationale

On Etities: (Sales,Advertisements, Employees, Users, Messages, Comments, Pages, Groups, Posts)
These entities are the necessary to act as the base of a social networking platform.

On relationships:
1) One Employee is in charge of any number of advertisements.
2) Advertisements are based off of Sales attributes in order to be targeted.
3) Sales are based off of the User entity and attributes from it such as Rating and number of messages sent.
4) Messages are sent by 1 specific User and can be recieved by many users.
5) Posts are created by 1 user and assigned to 1 page.
6) Each post can have up to N number of comments.
7) Each comment can only have 1 creator.
8) Each user has a page (profilepage/wall) that can be viewed by any number of users.
9) One user can form a group.
10) Once formed, A group is open to any number of users to join.
11) Each group has a page.

```
CREATE TABLE Users (
        UserId INTEGER,
        LastName VARCHAR(20),
        FirstName VARCHAR(20),
        Address VARCHAR(50),
        City VARCHAR(20),
        State VARCHAR(20),
        ZipCode INTEGER,
        Telephone INTEGER,
        Email VARCHAR(30),
        AccountNumber INTEGER,
        AccountCreationDate DATE,
        CreditCardNumber INTEGER,
        Preference VARCHAR(20),
        Ratings INTEGER,
        PRIMARY KEY (UserId),
        UNIQUE (AccountNumber)
)
```

A user table that contains standard information for customer users. The account number should be unique.

```
CREATE TABLE UsersFriends (
```

```
    UserId INTEGER,
    FriendId INTEGER,
    PRIMARY KEY (UserId, FriendId),
    FOREIGN KEY (UserId) REFERENCES Users (UserId)
    FOREIGN KEY (FriendId) REFERENCES Users (UserId),
)
```
A table to store the many-to-many friend relationship between users.

```
CREATE TABLE Groups (
    GroupId INTEGER,
    GroupName, VARCHAR(50),
    Type VARCHAR(20),
    OwnerId INTEGER,
    PRIMARY KEY (GroupId),
    FOREIGN KEY (OwnerId) REFERENCES Users (UserId)
)
```
A groups table that contains the group information and ownerId. Data on group members is in the following table.

```
CREATE TABLE GroupsMembers (
    GroupId INTEGER,
    UserId INTEGER,
    PRIMARY KEY (GroupId, UserId),
    FOREIGN KEY (GroupId) REFERENCES Groups (GroupId),
    FOREIGN KEY (UserId) REFERENCES Users (UserId)
)
```
A table to represent the many-to-many relation between groups and users.
```
CREATE TABLE Pages (
    PageId INTEGER,
    OwnerId INTEGER DEFAULT NULL,
    GroupId INTEGER DEFAULT NULL,
    PostCount INTEGER,
    PageType CHAR(5),
    PRIMARY KEY (PageId),
    FOREIGN KEY (OwnerId) REFERENCES User (UserId),
    FOREIGN KEY (GroupId) REFERENCES Group (GroupId),
    CHECK (!(OwnerId == NULL && GroupId == NULL) && !(OwnerId != NULL &&
GroupId != NULL)),
    CHECK (PageType IN (‘Group’, ‘User’)
)
```
Pages can either be for users or groups, but only one of these types should be initialized.
```
CREATE TABLE Posts (
    PostId INTEGER,
    PosterId INTEGER,
    PageId INTEGER,
    Date DATE,
```

```
    Content VARCHAR(2000),
    CommentCount INTEGER,
    PRIMARY KEY (PostId),
    FOREIGN KEY (PosterId) REFERENCES Users (UserId)
    FOREIGN KEY (PageId) REFERENCES Pages (PageId)
)
```

Posts table that contains information on the post and refers to the ID of the poster. It also refers to the page ID that it belongs to. This assumes a one-to-many relation between pages and posts. If posts can belong to multiple pages, then we can create a new table for this many-to-many relation.

```
CREATE TABLE Comments (
    CommentId INTEGER,
    PosterId INTEGER,
    PostId, INTEGER,
    Date DATE,
    Content VARCHAR(2000),
    PRIMARY KEY (CommentId),
    FOREIGN KEY (PosterId), REFERENCES Users (UserId),
    FOREIGN KEY (PostId), REFERENCES Posts (PostId)
)
```

A table for comments that refers to the commenter and the original post.

```
CREATE TABLE Messages (
    MessageId INTEGER,
    Date DATE,
    Subject VARCHAR(50),
    Content VARCHAR(2000),
    PRIMARY KEY (MessageId)
)
```

A table for messages between users. The sender and receivers are in the next table.

```
CREATE TABLE MessagesSent (
    MessageId INTEGER,
    SenderId INTEGER,
    ReceiverId INTEGER,
    PRIMARY KEY (MessageId, SenderId, ReceiverId),
    FOREIGN KEY (MessageId) REFERENCES Messages (MessageId),
    FOREIGN KEY (SenderId) REFERENCES Users (UserId),
    FOREIGN KEY (ReceiverId) REFERENCES Users (UserId)
)
```

A separate table designed for senders and receivers, because a message can be sent to multiple receivers.

```
CREATE TABLE Advertisements (
    AdvertisementId INTEGER,
    EmployeeId INTEGER,
    Type VARCHAR(20),
    Date DATE,
    Company VARCHAR(50),
```

```
    ItemName VARCHAR(50),
    Content VARCHAR(2000),
    UnitPrice INTEGER,
    Units INTEGER,
    PRIMARY KEY (AdvertisementId),
    FOREIGN KEY (EmployeeId) REFERENCES Employee (EmployeeId)
)
```

A table to store information about advertisements. It refers to the employee who made the advertisement.

```
CREATE TABLE Sales (
    TransactionId INTEGER,
    DateTime DATETIME,
    AdvertisementId INTEGER,
    Units INTEGER,
    AccountNumber INTEGER,
    PRIMARY KEY (TransactionId),
    FOREIGN KEY (AdvertisementId) REFERENCES Advertisements (AdvertisementId),
    FOREIGN KEY (AccountNumber) REFERENCES Users (AccountNumber)
)
```

The sales table records transaction information and refers to the advertisement of the transaction, as well as the customer through his/her account number, which should be unique.

```
CREATE TABLE Employees (
    EmployeeId INTEGER,
    SocialSecurityNumber INTEGER,
    FirstName VARCHAR(20),
    LastName VARCHAR(20),
    Address VARCHAR(50),
    City VARCHAR(20),
    State VARCHAR(20),
    ZipCode INTEGER,
    Telephone INTEGER,
    StartDate  DATE,
    HourlyRate DECIMAL (5,2),
    PRIMARY KEY (EmployeeId),
    UNIQUE (SocialSecurityNumber)
)
```

A table for employees that is much like the table for a user. An employee ID was added over the guidelines so that the social security number would not be the primary key, for security reasons. Instead, it is a candidate key.

Collaboration plan:

Using the examples on blackboard:

Teammate 1 will focus on groups, pages, postings, and comments.
Teammate 2 will focus on messaging, searching, liking and unliking postings and comments, and group-related privacy issues.
Teammate 3 will focus on all aspects of targeted advertising and sales.

Kaile: 1
Yida: 2
Daniel: 3