Introduction to Databases – Project1 Proposal

Bin Qi (bq2130) Yunqiu Yao (yy2827)

General:

Our application tries to monitor the private information leakage on Venmo, a popular peer-to-peer payment app with social media features. There are tons of transactions between different users in our app as Venmo displays private data in a way more public than we could imagine. With our app, a user could track her transaction history and the extent of personal data publicized by Venmo. Also, she could search for someone else's personal/social/financial info on Venmo that she is interested in. While free and convenient to use, Venmo users could be extremely vulnerable to info leakage. Our app can be a timely alert that some Venmo users should change their security settings and using behavior in order not to fall for potential online scams.

To each user, by inputting a specific Venmo username, our app could generate a risk level indicating the comprehensive safety status of that Venmo account. The domain of "risk level" is {0,1,2,3,4}, and the higher the value, the riskier. We calculate this score by aggregating different dimensions of privacy i.e., the volume of public transaction history; the size of social network shown in transactions and interactions; any accessible connection to his/her facebook account; the length of notes attached to each transaction. More precisely:

rid	risk_level	public_transaction>5	friends>3	link_to_Facebook	Notes_length>10	Google_search_results
1	0	0	0	0	0	null
2	1	1	0	0	0	venmo+risk
3	1	0	1	0	0	venmo+friends+risk
4	1	0	0	1	0	venmo+facebook+risk
5	1	0	0	0	1	venmo+message+risk
6	2	1	1	0	0	public+venmo+friends+risk
7	2	1	0	1	0	public+venmo+facebook+risk
8	2	1	0	0	1	public+venmo+message+risk
9	2	0	1	1	0	venmo+friends+facebook+risk
10	2	0	1	0	1	venmo+friends+message+risk
11	2	0	0	1	1	venmo+facebook+message+risk
12	3	1	1	1	0	public+venmo+friends+facebook+risk
13	3	0	1	1	1	venmo+friends+facebook+message+risk
14	3	1	0	1	1	public+venmo+facebook+message+risk
15	3	1	1	0	1	public+venmo+friends+message+risk
16	4	1	1	1	1	public+venmo+friends+facebook+message+risk

Furthermore, relevant google search results regarding the potential risk factors can also be posted to inform and remind our users. The last column of the table above lists the key words we may apply in a UDF query which presents the google search results.

Data source:

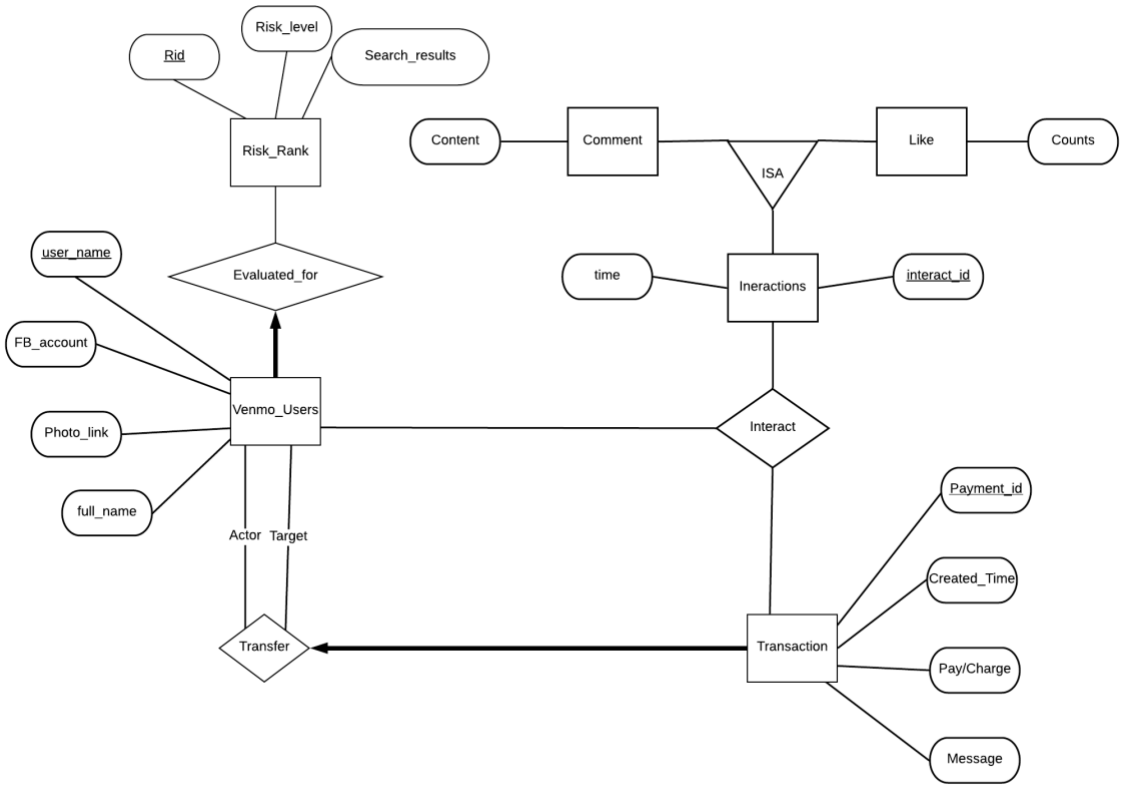
We will obtain the transaction feed and personal data via Venmo public API. It appears that Venmo is updating the transactions marked as “public” dynamically to the website. Scraping those publicly available data from the site, we can see all the information about the transactions and the initiators and respondents, such as their full names, photos, transaction time, message, etc. In other words, all the private information that we value is readily shared with the rest of the world.

Examples:

Entities: Venmo_Users (user_name, full_name, FB_link, Photo_link), Transaction (Payment_id, Pay/Charge, Message, Create_Time), Interaction (interact_id, time), Comment (Content), Like (Counts), Risk_Rank(rid, Risk_level, Search_results).

Relationship & constraints: each transaction has exactly one actor (user) and one target (user); a Venmo user can interact with several transactions and one transaction for several times; an interaction can be a “Comment” with words, or a “like” with counts. Each user has exactly one risk rank.

E-R diagram:



Schema:

```
create table venmo_users_evaluate(
    user_name text primary key,
    full_name text not null,
    photo_link text,
    fb_account text,
    rid int not null references risk_rank
);

create table risk_rank(
    rid int primary key,
```

```
    risk_level int not null,
    search_results text
);

create table transa_transf(
    payment_id int primary key,
    created_time date not null,
    pay_charge text not null,
    actor_name text not null references venmo_users(user_name),
    target_name text not null references venmo_users(user_name),
    message text not null
)

create table interaction(
    interaction_id int primary key,
    created_time date not null
);

create table comment(
    interaction_id int primary key references interaction,
    content text
);

create table like(
    interaction_id int primary key references interaction,
    rating int
);

create table interact(
    payment_id int references transa_transf,
    user_name text references venmo_users,
    interact_id int references interaction,
    primary key(payment_id,user_name,interact_id)
);
```