

Introduction to Databases – Project1 Proposal

Bin Qi (bq2130) Yunqiu Yao (yy2827)

1. UNI used on the course DB server: **yy2827**

2. 3 queries:

(1) How many transactions per person in the past week?

%%sql

```
with trans (username, name, trans_id) as (  
    select tt.actor_name, vu.full_name, tt.payment_id from venmo_users as vu  
    join transa_transf as tt on vu.user_name = tt.actor_name  
    where tt.created_time >= current_date - interval '7 days'  
    union  
    select tt.target_name, vu.full_name, tt.payment_id from venmo_users as vu  
    join transa_transf as tt on vu.user_name = tt.target_name  
    where tt.created_time >= current_date - interval '7 days'  
)  
select username, name, count(*) as transactions from trans  
group by username, name  
order by transactions DESC;
```

• Result table:

	username	name	transactions
0	NoahGershwin	Noah Gershwin	6
1	jjscarz	Julia Scarangella	6
2	christianromeroUP	Christian Romero	5
3	Guo-Zhiqi	Guo Zhiqi	4
4	Marisa-Chambers	Marisa Chambers	2
5	Michael-Caguioa	Michael Caguioa	2
6	elisadangelo	Elisa Dangelo	2
7	Brian-Du-3	Brian Du	2
8	Jeffrey-Kemper-1	Jeffrey Kemper	2
9	Andrew-Kwon-8	Andrew Kwon	1
10	MeganResnick	Megan Resnick	1
11	anthonynguyen1006	Anthony Nguyen	1

- Description: We are interested in computing the amount of transactions one Venmo user participated in (as an actor or target) during the period of data collection. This statistic can be used as one of the risk dimensions. Theoretically, the more the publicized

transactions we could retrieve from one Venmo account, the higher risk it may be exposed to.

(2) How many distinct people one user transacted with in the past week?

```
%%sql
with trans_person (username1, username2) as (
  select tt.actor_name, tt.target_name from transa_transf as tt
  where tt.created_time >= current_date - interval '7 days'
  union
  select tt.target_name, tt.actor_name from transa_transf as tt
  where tt.created_time >= current_date - interval '7 days'
)
select username1, vu.full_name as name, count(username1) as persons from
trans_person as tp
join venmo_users as vu on vu.user_name = tp.username1
group by username1, name
order by persons DESC;
```

- Result table:

	username1	name	persons
0	NoahGershwin	Noah Gershwin	5
1	jiscarz	Julia Scarangella	5
2	christianromeroUP	Christian Romero	4
3	Guo-Zhiqi	Guo Zhiqi	4
4	elisadangelo	Elisa Dangelo	2
5	Brian-Du-3	Brian Du	2
6	Marisa-Chambers	Marisa Chambers	1
7	Andrew-Kwon-8	Andrew Kwon	1
8	anthonynguyen1006	Anthony Nguyen	1
9	Michael-Caguioa	Michael Caguioa	1
10	Jeffrey-Kemper-1	Jeffrey Kemper	1
11	MeganResnick	Megan Resnick	1

- Description: This calculates the size of social network shown in transactions for each Venmo user. The larger the value, the higher the risk of privacy leakage.

(3) The largest length of message for each Venmo user's transactions

```
%%sql
drop table if exists mes cascade;
create table mes (username, name, trans_id, message) as (
```

```

select tt.actor_name, vu.full_name, tt.payment_id, tt.message from venmo_users as
vu
join transa_transf as tt on vu.user_name = tt.actor_name
where tt.created_time >= current_date - interval '7 days'
);
drop function if exists strlen cascade;
CREATE FUNCTION strlen(mes) RETURNS integer AS $$
    SELECT char_length($1.message);
$$ LANGUAGE SQL;
select username, name, max(strlen(mes.*)) as length from mes
group by username, name
order by length DESC;

```

- Result table:

	username	name	length
0	Brian-Du-3	Brian Du	99
1	NoahGershwin	Noah Gershwin	31
2	jjscarz	Julia Scarangella	19
3	MeganResnick	Megan Resnick	18
4	christianromeroUP	Christian Romero	14
5	Guo-Zhiqi	Guo Zhiqi	13
6	Jeffrey-Kemper-1	Jeffrey Kemper	10
7	Marisa-Chambers	Marisa Chambers	10
8	Michael-Caguioa	Michael Caguioa	7
9	elisadangelo	Elisa Dangelo	1

- Description: We use the length of “messages” under transaction as another risk factor. This captures the detailedness of notes one user would like to publicize when transacting on Venmo. More characters indicate more info leakage.

3. Changes made since last submission

Based on the mentor’s feedback, we revised the “interact” relationship part of E-R diagram, simplifying it into 2 separate relationships: Comments and Like (thanks to TA Ivy Chen).

Also, we added a summarized cache table “Risk_Rank” (thanks to TA Ivy Chen and our mentors’ advice). We revised the corresponding

“relationships&constraints” part. More details can be found in the revised proposal below.

To fulfill the requirement of entity/relationship numbers, we added a new entity “Bank_acciunts”, to record credit card info associated with each Venmo account. It is also a weak entity for “Venmo_users”.

We added 2 “check” constraints in the tables “transa_transf” and “bank_pay”.

The summarized cache table “Risk_Rank”, which will provide our app user a risk score and some google search results, has not yet been completed. We plan to discuss with our mentor in the first place, then enrich more queries to generate this application-level table.

(We attach the revised project proposal below, and look forward to your feedbacks and corrections. Thank you!)

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 various 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 message 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+public+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 in our app 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 risk related google search results.

It may appear that the table “Risk_Rank” is sort of redundant, as each attribute’s value can be generated by aggregating other tables. However, from the perspective of app use, as this application-level table may not be updated as frequently other (risk levels will be updated once a week, while other data can be daily updated) it seems necessary to include a cache table containing “delayed” risk levels in our database so that an existent risk score can be accessed any time by our user.

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),
Comments (com_id, contents)

Risk_Rank (rid, Risk_level, Search_results)

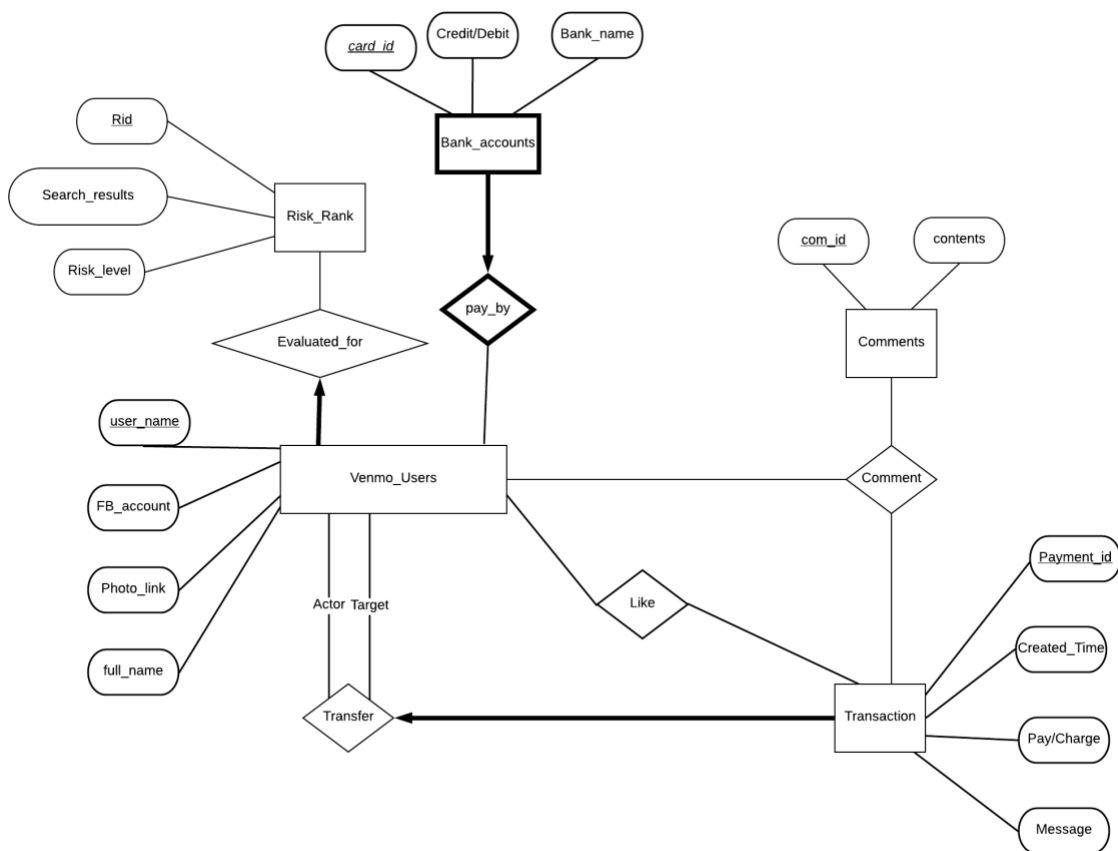
Bank_accounts (card_id, Credit/Debit, Bank_name)

Relationship & constraints:

Each transaction has exactly one actor (user) and one target (user); a Venmo user can comment on several transactions and one transaction for several times; a Venmo user can “like” a transaction only once; A Venmo account can relate to several bank accounts, while each bank account is

connected to exactly one Venmo user and is its weak entity; Each user has exactly one assigned risk rank.

E-R diagram:



Schema:

```

create table risk_rank(
    rid int primary key,
    risk_level int not null,
    search_result text not null
);
    
```

```
create table venmo_users(  
    user_name text primary key,  
    full_name text not null,  
    photo_link text,  
    fb_account text  
);
```

```
create table transa_transf(  
    payment_id int primary key,  
    created_time date not null,  
    pay_charge text not null check (pay_charge = 'pay' or pay_charge ='charge'),  
    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 likes(  
    payment_id int references transa_transf,  
    user_name text references venmo_users,  
    primary key(payment_id,user_name)  
);
```

```
create table comments(  
    com_id int primary key,  
    contents text not null  
);
```

```
create table comment(  
    com_id int references comments,  
    payment_id int references transa_transf,  
    user_name text references venmo_users,  
    primary key(com_id, payment_id,user_name)  
);
```

```
create table bank_pay(  
    card_id int primary key,  
    user_name text not null references venmo_users on delete cascade,  
    credit_debit text not null check (credit_debit = 'credit' or credit_debit ='debit'),  
    bank_name text not null  
);
```

```
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  
);
```