Last Name: SUH First Name: JOOWON Student ID: 44414081

1. [10pts] Return the handles of Tweeters and their number of Covid-tagged tweets if they've used the hashtag "**covid19**" more than 3 times. Your query should normalize the hashtags to lowercase (e.g., **Covid19** should be converted to **covid19** in order to properly consider all Covid-tagged tweets.

a) [7pts] SQL Query:

```
from Tweeter
where Tweeter.tweeter_id IN ( select tweeter_id
from( select  count(*) as cnt, tweeter_id
        from (    select lowercase.hashtag, lowercase.tweet_id, count(*)
                  from (    select LOWER(hashtag) as hashtag, tweet_id
                                    from Hashtags ) as lowercase
                  where lowercase.hashtag = 'covid19'
                  group by tweet_id ) as covidtweets, Tweet
        where covidtweets.tweet_id = Tweet.tweet_id
        group by tweeter_id ) as cnttb
where cnttb.cnt >3 )
```

b) [3pts] Result (2 rows):

CupofJoeintheD2
ppl4justice

2. [10pts] Return the handles of Tweeters who have a followers count greater than 500,000 and who have posted a tweet that contains one or more of the top ten most popular hashtags. (Note: You can break popularity ties arbitrarily.)

a) [7pts] SQL Query:

```
select *
from Tweeter
where followers_count > 500000 and Tweeter.tweeter_id IN (select tweeter_id
from
        (select Hashtags.tweet_id
        from (    select count(*), lowercase.hashtag
                        from ( select hashtag as hashtag, tweet_id from Hashtags ) as lowercase
                        group by lowercase.hashtag
                        order by count(*) desc
                        limit 10) as topten, Hashtags
        where topten.hashtag = Hashtags.hashtag) as tweetwithtopten, Tweet
where tweetwithtopten.tweet_id = Tweet.tweet_id)
```

b) [3pts] Result (3 rows):

display_name	followers_count	handle	tweeter_id	verified
CTV News	1001621	CTVNews	203123011	1
BrooklynDad_Defiant!	722723	mmpadellan	1640929196	0
Refinery29	1309078	Refinery29	19546942	1

3. [10pts] Find the tweet ids for tweets that have been **verified** using at least two **different** pieces of evidence and that have a popularity greater than 2.4. Remember from HW1 (😌) that the popularity of a tweet can be computed using the formula:

$$Popularity = 0.4\ (Number\ of\ quotes) + 0.6\ (Number\ of\ replies)$$

a) [7pts] SQL Query:

b) [3pts] Result (4 rows):

4. **Views** [20 pts]

Congratulations! For obvious reasons, the CTO of ***CheckedTweets.org*** is setting up a data science team to analyze election tweets that contain one or more of the following hashtags: **"election2020", "trump", "biden", "bidenharris2020", "trumppence2020", "pennsylvania", "northcarolina", "wisconsin", "michigan".** (*You will need to normalize the hashtags to lowercase*.) The CTO has made you the head of that team. As the team leader, you have been asked to create a SQL view so that the rest of the team can simply look at the data and draw meaningful conclusions without having to deal with all of its underlying complexity.

The view should provide simple tabular access to a combination of the following pieces of information:

- ❑ Tweeter info (tweeter_id, handle, followers_count, verified)
- ❑ Tweet info (tweet_id, tweet_text, *popularity*, *quality*)

Remember that tweet *popularity* and *quality* are derived attributes and can be computed as follows:

$$Popularity = 0.4\ (Number\ of\ quotes) + 0.6\ (Number\ of\ replies)$$
$$Quality = Amount\ of\ associated\ evidence\ used\ for\ verification$$

a) [15 pts] Create the desired view (ElectionTweets) by writing an appropriate CREATE VIEW statement.

CREATE VIEW ElectionTweets…;

```
DROP VIEW IF EXISTS `ElectionTweets`;
CREATE VIEW ElectionTweets (tweeter_id, handle, followers_count, verified, tweet_id, tweet_text )
as

    select TWT.tweeter_id, TWT.handle, TWT.followers_count, TWT.verified, TW.tweet_id, TW.tweet_text
    from (  select *
            from Tweet
            where tweet_id in ( select tweet_id
            from (
            select lower(hashtag) as hashtag, tweet_id
            from Hashtags
            )as lowercase
            where lowercase.hashtag ="election2020" or
            lowercase.hashtag ="trump"or
            lowercase.hashtag = "biden" or
            lowercase.hashtag = "bidenharris2020" or
            lowercase.hashtag = "trumppence2020" or
            lowercase.hashtag = "pennsylvania"or
            lowercase.hashtag = "northcarolina" or
            lowercase.hashtag = "wisconsin" or
            lowercase.hashtag = "michigan" )) as TW, Tweeter TWT
    where TW.tweeter_id = TWT.tweeter_id ;
```

(Hint: your view should have 699 rows)

tweeter_id	handle	followers_count	verified	tweet_id	tweet_text	popularity	quality
3424914034	jtksandstormer	2	0	1321194058656681986	SCOTUS rules Wisconsin ball…	0.0	3
2251868828	BelleBelle410	40	0	1321194075517693952	After just filling out my ballot, I…	1.2	2
1314354148234625024	Emile_L_Tellah	54	0	1321194079246581766	Why didnt 80 million of us thin…	0.6	3
15096075	idealist	94401	1	1321194099588956166	68% of adults say that the 22	0.6	0

b) [5 pts] Show the usefulness of your view by writing a SELECT query against the view that prints the Tweet id, the Tweeter's handle, and the popularity and quality of tweets that have the maximum popularity.

Result (1 row):

5. **Stored Procedures** [20 pts]

a) [15 pts] Create and exercise a SQL stored procedure called RegisterChecker(…) that the application can use to add a brand new checker with an office phone to the database. You **may not** change the signature of this procedure. Hint: To get the current time, use the NOW() function.

```
DELIMITER //
CREATE PROCEDURE RegisterChecker(
    user_id integer,
    name_first varchar(50),
    name_last varchar(50),
    email varchar(100),
    password varchar(30),
    profile_pic varchar(500),
    address_country varchar(30),
    address_state varchar(30),
    address_city varchar(30),
    office_number varchar(20)
)
BEGIN
    ...
END;  //
DELIMITER ;
```

```
BEGIN
    insert into User
    values( user_id, name_first, name_last, email, password, NOW(), profile_pic, address_country, address_state, address_city );
    insert into Checker
    values( user_id, NOW() );
    insert into Phone
    values( user_id, 'OFFICE', office_number);
END;  //
DELIMITER ;
```

b) [5pts] Verify that your new stored procedure works properly by calling it as follows to add a new checker and then running a SELECT query to show the stored procedure's after-effects:

```
CALL RegisterChecker (3000, "Peter", "Anteater",
    "peter-anteater2020@gmail.com", "pretend-this-is-hashed",
    null, "USA", "California", "Irvine", "(949) 824-5011");

SELECT U.user_id, U.email, U.profile_pic, C.checker_since, P.number, P.kind
FROM User U, Checker C, Phone P
WHERE U.user_id = C.user_id AND
      P.user_id = C.user_id AND
       U.user_id = 3000;
```

Result (1 row):

user_id	email	profile_pic	checker_since	number	kind
3000	peter-anteater2020@gmail.com	NULL	2020-11-21 19:09:09	(949) 824-5011	OFFICE

6. **Alter Table** [10 pts]

As your schema currently stands, evidence can only be submitted in the form of URLs to websites. Your boss would like to enrich the Evidence entity by also allowing books (specifically, 13-character ISBNs) to be used as evidence. This changes your ER model in two ways: 1) URL now becomes an optional field in Evidence, and 2) ISBN is now an additional optional field in Evidence.

Note: *The current datatype for URL is* `VARCHAR(500)`.

a) [5 pts] Write and execute the ALTER TABLE statement(s) needed to modify the Evidence table to reflect the new requirements above. (Hint: Refer to the MySQL documentation online if you need more information about how to use the ALTER TABLE statement.)

```
1 •  alter table Evidence
2    modify url varchar(500) NULL;
3
4 •  alter table Evidence
5    add isbn varchar(13) null after url;
6
```

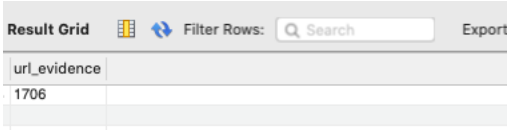
b) [5 pts] Execute the following INSERT and SELECT statements to show the effect of your change. Report the results (just the counts) for each SELECT statement.

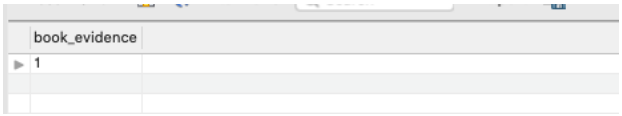
```
INSERT INTO Evidence (ev_id, url, isbn)
VALUES (2000, NULL, "0-1306-3278-3");

SELECT COUNT(*) AS url_evidence
FROM Evidence
WHERE url IS NOT NULL;

SELECT COUNT(*) AS book_evidence
FROM Evidence
WHERE isbn IS NOT NULL;
```

Result:





7. **Triggers** [20 pts]

a) [15 pts] To help tie your newfound SQL knowledge back to the seemingly mysterious initial ER model, you are tasked with defining a trigger called update_tweet_info(…). When raw tweets are deposited into the database by your application, this trigger will insert tuples into the Tweet, Tweeter, and Hashtags tables using the information found in the newly deposited raw tweets. *If a Tweeter already exists at the time of a deposit, you should only update their follower count, display name, and handle.* To specify an update action for an INSERT statement when you have a duplicate primary key, i.e., when an object with that key already exists, see here.

```
USE cs122a_fall20;
DROP TRIGGER IF EXISTS update_tweet_info;
--
-- Question 7a (trigger template)
--
DELIMITER //
CREATE TRIGGER update_tweet_info
after insert on RawTweet
FOR EACH ROW
BEGIN
    INSERT INTO Tweeter
    values(JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.user.screen_name')),
    JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.user.followers_count')),
    JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.user.name')),
    JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.user.id_str')),
    CASE WHEN JSON_EXTRACT(new.content, '$.user.verified') THEN 1 ELSE 0 END
    )
    ON DUPLICATE KEY UPDATE display_name = JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.user.screen_name')),
                            followers_count = JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.user.followers_count')),
                            handle = JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.user.name'));

    INSERT INTO Tweet
    values( JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.created_at')),
    JSON_EXTRACT(new.content, '$.geo.coordinates[0]'),
    JSON_EXTRACT(new.content, '$.geo.coordinates[1]'),
    JSON_EXTRACT(new.content, '$.quoted_status_id'),
    JSON_EXTRACT(new.content, '$.in_reply_to_status_id'),
    JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.id')) ,
    JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.text')),
    JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.user.id_str'))
    );


    CALL UpdateHashtags(JSON_UNQUOTE(JSON_EXTRACT(new.content, '$.id')));
END; //
DELIMITER ;
```

Hint 1: To get all tweeter-associated information for an arbitrary single raw tweet, we can perform the query:

```
SELECT JSON_UNQUOTE(JSON_EXTRACT(content, '$.user.screen_name')) AS display_name,
    JSON_UNQUOTE(JSON_EXTRACT(content, '$.user.followers_count')) AS
followers_count,
    JSON_UNQUOTE(JSON_EXTRACT(content, '$.user.name')) AS handle,
    JSON_UNQUOTE(JSON_EXTRACT(content, '$.user.id_str')) AS tweeter_id,
    CASE WHEN JSON_EXTRACT(content, '$.user.verified') THEN 1 ELSE 0 END AS
verified
```

```
FROM RawTweet T
LIMIT 1;
```

Hint 2: To get all tweet-associated information for an arbitrary single raw tweet, we can perform the query:

```
SELECT JSON_UNQUOTE(JSON_EXTRACT(T.content, '$.created_at')) AS posting_datetime,
    JSON_EXTRACT(T.content, '$.geo.coordinates[0]') AS posting_location_latitude,
    JSON_EXTRACT(T.content, '$.geo.coordinates[1]') AS posting_location_longitude,
    JSON_EXTRACT(T.content, '$.quoted_status_id') AS quoted_tweet,
    JSON_EXTRACT(T.content, '$.in_reply_to_status_id') AS replied_to_tweet,
    JSON_UNQUOTE(JSON_EXTRACT(T.content, '$.id')) AS tweet_id,
    JSON_UNQUOTE(JSON_EXTRACT(T.content, '$.text')) AS tweet_text,
    JSON_UNQUOTE(JSON_EXTRACT(T.content, '$.user.id_str')) AS tweeter_id
FROM RawTweet T
LIMIT 1;
```

Hint 3: To **update** the Hashtag table for a particular raw tweet, we can call the following stored procedure that we have provided for you in the updated load script:

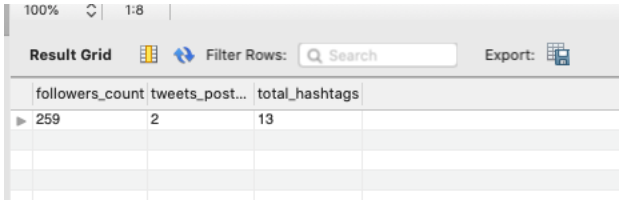
```
CALL UpdateHashtags(tweet_id);
```

```
DELIMITER //
CREATE TRIGGER update_tweet_info
...
FOR EACH ROW
BEGIN
...
END; //
DELIMITER ;
```

b) [5 pts] Execute the following INSERT and SELECT statements to show the effect of your trigger. Report the follower count, the number of tweets posted, and the number of distinct hashtags associated with the specified tweeter before and after each INSERT.

```
SELECT TW.followers_count, COUNT(DISTINCT T.tweet_id) AS tweets_posted,
                           COUNT(DISTINCT H.hashtag) AS total_hashtags
FROM Tweeter TW
LEFT OUTER JOIN Tweet T ON TW.tweeter_id = T.tweeter_id
LEFT OUTER JOIN Hashtags H ON T.tweet_id = H.tweet_id
WHERE TW.tweeter_id = T.tweeter_id
AND T.tweeter_id = '1121141389696413697';
```

Result (1 row):

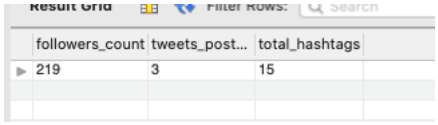



```
INSERT INTO RawTweet VALUES (
    "1321203503310762222",
    '{"id": "1321203503310762222", "created_at": "2020-06-12 14:56:40",
"extended_tweet": {"entities": {"hashtags": [{"indices": [136, 145], "text":
"criminals"}, {"indices": [147, 158], "text": "Obidengate2020"}, {"indices": [159,
169], "text": "obamagate"}, {"indices": [170, 178], "text": "treason"}, {"indices":
[179, 198], "text": "politicalespionage"}, {"indices": [199, 219], "text":
"seditiousconspiracy"}, {"indices": [220, 235], "text": "4yrsrussialies"},
{"indices": [236, 262], "text": "fbidocsproveHRCrussiahoax"}], "user_mentions":
[{"indices": [0, 9], "screen_name": "LLinda_W", "id_str": "3303621560", "name":
"LindaW- #VoteBlueDownBallot", "id": 3303621560}, {"indices": [10, 23],
"screen_name": "WardDPatrick", "id_str": "3353220134", "name": "Pat Ward", "id":
3353220134}]}, "full_text": "@LLinda_W @WardDPatrick Truly nauseating to hear the
excuses and defensive epic fails coming from antiamerican actually voting for this
#criminal  #Obidengate #obamagate #treason #politicalespionage #seditiousconspiracy
#4yrsrussialies #fbidocsproveHRCrussiahoax   https://t.co/Q98eyjmpBC"},
"retweet_count": 0, "retweeted": false, "filter_level": "low",
"in_reply_to_screen_name": "LLinda_W", "is_quote_status": false, "id_str":
"1321203503310762222", "favorite_count": 0, "text": "@LLinda_W @WardDPatrick Truly
nauseating to hear the excuses and defensive epic fails coming from antiamerican
actu\u2026 https://t.co/tDmveFKMDf", "lang": "en", "quote_count": 0, "favorited":
false, "possibly_sensitive": false, "truncated": true, "timestamp_ms":
"1603834143603", "reply_count": 0, "user": {"friends_count": 911,
"profile_image_url_https":
"https://pbs.twimg.com/profile_images/1280133701599404032/9HYrUqHg_normal.jpg",
"listed_count": 1, "profile_background_image_url": "", "default_profile_image":
false, "favourites_count": 9141, "description": "\u26a1 #Seditiousconspiracy",
"is_translator": false, "profile_background_image_url_https": "", "protected":
false, "screen_name": "WrathchiksMama", "id_str": "1121141389696413697",
"profile_link_color": "1DA1F2", "translator_type": "none", "id":
1121141389696413697, "geo_enabled": false, "profile_background_color": "F5F8FA",
"profile_sidebar_border_color": "C0DEED", "profile_text_color": "333333",
"verified": false, "profile_image_url":
"http://pbs.twimg.com/profile_images/1280133701599404032/9HYrUqHg_normal.jpg",
"url": "https://www.altcensored.com/watch?v=9HFxVvrXjCg", "contributors_enabled":
false, "profile_background_tile": false, "profile_banner_url":
"https://pbs.twimg.com/profile_banners/1121141389696413697/1594042636",
"statuses_count": 7648, "followers_count": 219, "profile_use_background_image":
```

```
true, "default_profile": true, "name": "ThePlotAgainstPresident", "location":
"\ud83d\udc47\ud83d\udc47WATCH&SHARE!#SaveOur1A\ud83d\udc47\ud83d\udc47",
"profile_sidebar_fill_color": "DDEEF6"}}'
);

SELECT TW.followers_count, COUNT(DISTINCT T.tweet_id) AS tweets_posted,
                           COUNT(DISTINCT H.hashtag) AS total_hashtags
FROM Tweeter TW
LEFT OUTER JOIN Tweet T ON TW.tweeter_id = T.tweeter_id
LEFT OUTER JOIN Hashtags H ON T.tweet_id = H.tweet_id
WHERE TW.tweeter_id = T.tweeter_id
AND T.tweeter_id = '1121141389696413697';
```

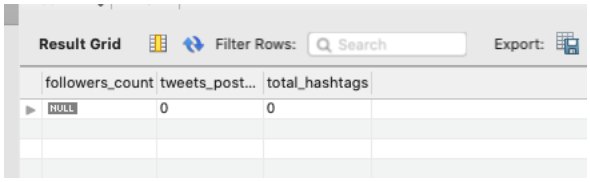
Result (1 row):



followers_count	tweets_post...	total_hashtags
219	3	15

```
SELECT TW.followers_count, COUNT(DISTINCT T.tweet_id) AS tweets_posted,
                           COUNT(DISTINCT H.hashtag) AS total_hashtags
FROM Tweeter TW
LEFT OUTER JOIN Tweet T ON TW.tweeter_id = T.tweeter_id
LEFT OUTER JOIN Hashtags H ON T.tweet_id = H.tweet_id
WHERE TW.tweeter_id = T.tweeter_id
AND T.tweeter_id = '992109555483103122';
```

Result (1 row):



Result Grid Filter Rows: Search Export:

followers_count	tweets_post...	total_hashtags
NULL	0	0

```
INSERT INTO RawTweet VALUES (
    "1321203503310762322",
    '{"id": "1321203503310762322", "created_at": "2020-04-29 05:29:03",
"retweet_count": 0, "retweeted": false, "filter_level": "low", "is_quote_status":
false, "id_str": "1321203503310762322", "favorite_count": 0, "text": "Harris County
early voting hours extended to 10 p.m. until Thursday https://t.co/VF9uvQlJwK)
**LETS GO, HARRIS COU\u2026 https://t.co/k7pt7SRIN5", "lang": "en", "quote_count":
0, "favorited": false, "possibly_sensitive": false, "truncated": true,
```

```
"timestamp_ms": "1603832187724", "reply_count": 0, "entities": {"urls":
[{"display_url": "chron.com/news/election2\u2026", "indices": [68, 91],
"expanded_url": "https://www.chron.com/news/election2020/article/Harris-County-
early-voting-hours-extended-to-10-
15677986.php?utm_campaign=CMS%20Sharing%20Tools%20(Premium", "url":
"https://t.co/VF9uvQlJwK"}, {"display_url": "twitter.com/i/web/status/1\u2026",
"indices": [117, 140], "expanded_url":
"https://twitter.com/i/web/status/1321194070757330945", "url":
"https://t.co/k7pt7SRIN5"}]}, "user": {"friends_count": 22468,
"profile_image_url_https":
"https://pbs.twimg.com/profile_images/1057463467286773760/wKFC-Ixa_normal.jpg",
"listed_count": 6, "profile_background_image_url":
"http://abs.twimg.com/images/themes/theme1/bg.png", "default_profile_image": false,
"favourites_count": 20682, "description": "JUJUS BU PRU VETTED NICE LADY, BUT,
PLEASE DONT HIT ON ME. MARRIED CLOSE TO 40 YRS. CRITTER LOVER. PRO CLIMATE 4 ALL
KIDS. MAIN ACCT@jjsmokkieBOY57", "is_translator": false,
"profile_background_image_url_https":
"https://abs.twimg.com/images/themes/theme1/bg.png", "protected": false,
"screen_name": "smokesdad289", "id_str": "992109555483103122",
"profile_link_color": "FF691F", "translator_type": "none", "id":
992109555483103122, "geo_enabled": false, "profile_background_color": "000000",
"profile_sidebar_border_color": "000000", "profile_text_color": "000000",
"verified": false, "profile_image_url":
"http://pbs.twimg.com/profile_images/1057463467286773760/wKFC-Ixa_normal.jpg",
"contributors_enabled": false, "profile_background_tile": false,
"profile_banner_url":
"https://pbs.twimg.com/profile_banners/992109555483103232/1541111709",
"statuses_count": 62229, "followers_count": 20476, "profile_use_background_image":
false, "default_profile": false, "name": "jujus other", "location": "texas",
"profile_sidebar_fill_color": "000000"}}'
);
```


```
SELECT TW.followers_count, COUNT(DISTINCT T.tweet_id) AS tweets_posted,
                           COUNT(DISTINCT H.hashtag) AS total_hashtags
FROM Tweeter TW
LEFT OUTER JOIN Tweet T ON TW.tweeter_id = T.tweeter_id
LEFT OUTER JOIN Hashtags H ON T.tweet_id = H.tweet_id
WHERE TW.tweeter_id = T.tweeter_id
AND T.tweeter_id = '992109555483103122';
```

Result (1 row):