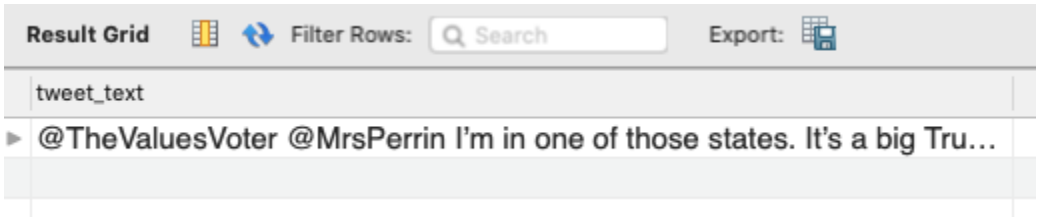
Last Name: **First Name:** **Student ID:**

1. [10 pts] Find the text of all tweets that were posted by the tweeter with the handle ‘patgotweet’.

a) [7 pts] SQL Query:

```
SELECT TW.tweet_text
FROM Tweet TW, Tweeter TR
WHERE TW.tweeter_id = TR.tweeter_id AND
      TR.handle = "patgotweet";
```

b) [3 pts] Result: (1 Row)

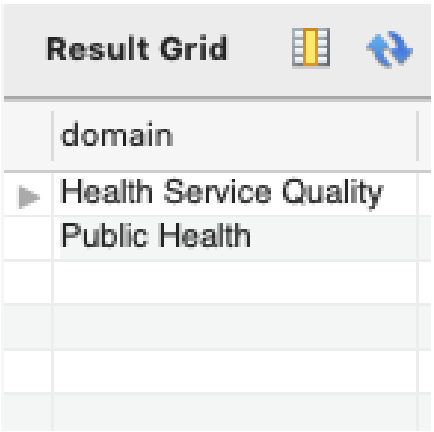


2. [10 pts] List the **distinct** domains of expertise for checkers who have verified tweets that have the hashtag "*COVID19*". (**Note: The hashtag value is all in capital letters.)**

a) [7 pts] SQL Query:

```
SELECT DISTINCT e.domain
FROM Expertise e, Verification v, Hashtags ht
WHERE ht.hashtag = 'COVID19'
AND ht.tweet_id = v.tweet_id
AND v.user_id = e.user_id
```

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

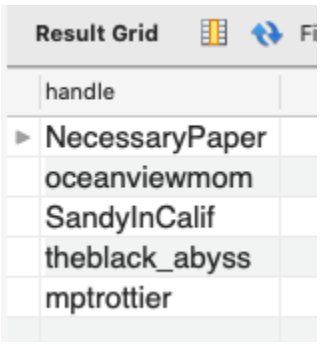


3. [10 pts] List the handles of Tweeters who have posted a tweet that has been verified by a Checker who started as a checker after the date “2020-01-31 03:41:49”.

a) [7 pts] SQL Query:

```
SELECT DISTINCT TR.handle
FROM Checker C, Verification V, Tweet TW, Tweeter TR
WHERE C.user_id = V.user_id AND
      V.tweet_id = TW.tweet_id AND
      TW.tweeter_id = TR.tweeter_id AND
      C.checker_since > '2020-01-31 03:41:49';
```

b) [3 pts] Result: (5 Rows)



4. [15 pts] For verified tweets that contain the hashtag "COVID19", find the associated evidence URLs, verification comments, and checkers' first and last names (**Again: "COVID19" is in all caps**.)

a) [12 pts] SQL Query:

```
SELECT u.name_first, u.name_last, v.comment, e.url
FROM Evidence e, Verification v, Hashtags ht, VerifiedUsing vu, User u
WHERE ht.hashtag = 'COVID19'
AND   ht.tweet_id = v.tweet_id
AND   v.user_id = u.user_id
AND   v.ver_id = vu.ver_id
AND   vu.ev_id = e.ev_id
```

b) [3 pts] Result (12 rows):

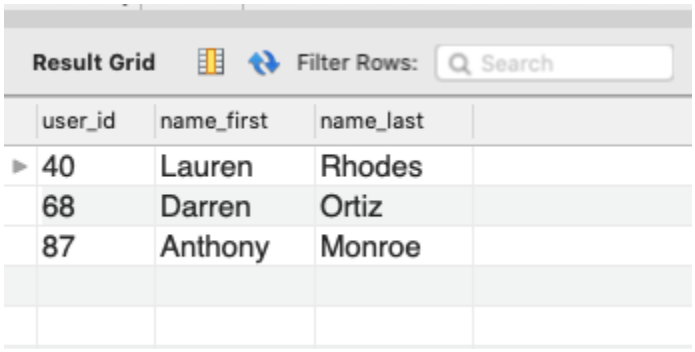
name_first	name_last	comment	url
Jonathan	Howard	Masks works! Check the CDC	http://states-covid-numbers.org
Jonathan	Howard	Masks works! Check the CDC	http://florida-covid19.gov
Jonathan	Howard	Masks works! Check the CDC	https://cdc.gov
Antonio	Olson	Masks works! Check the CDC	http://florida-covid19.gov
Antonio	Olson	Masks works! Check the CDC	https://cdc.gov
Antonio	Olson	Masks works! Check the CDC	http://states-covid-numbers.org
Gina	Miranda	Masks works! Check the CDC	http://states-covid-numbers.org
Gina	Miranda	Masks works! Check the CDC	http://florida-covid19.gov
Gina	Miranda	Masks works! Check the CDC	https://cdc.gov
Courtney	White	Masks works! Check the CDC	https://cdc.gov
Courtney	White	Masks works! Check the CDC	http://mask-works.info
Courtney	White	Masks works! Check the CDC	http://covid-is-not-hoax.net

5. [15 pts] Find the user IDs, first names, and last names of checkers that have **all** the domains of expertise from the user with ID = 68. (Note: Your answer will include the "ID = 68" checker as well, of course.)

a) [12 pts] SQL Query:

```
SELECT OU.user_id, OU.name_first, OU.name_last
FROM User OU, Checker OC
WHERE OU.user_id = OC.user_id AND
      NOT EXISTS (
            SELECT *
            FROM Expertise OE
            WHERE OE.user_id = 68 AND
                  NOT EXISTS (
                      SELECT *
                      FROM Expertise IE
                      WHERE IE.user_id = OC.user_id AND
                            IE.domain = OE.domain
                  )
      );
```

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



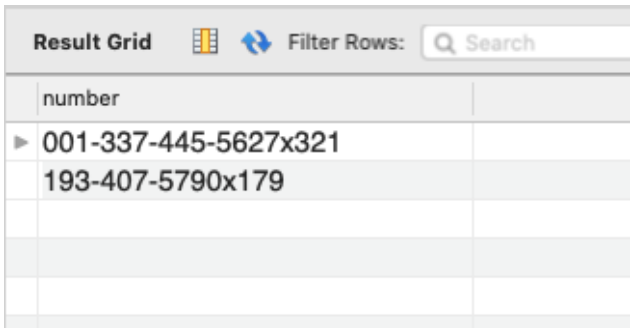
user_id	name_first	name_last
40	Lauren	Rhodes
68	Darren	Ortiz
87	Anthony	Monroe

6. [10 pts] List the phone numbers of checkers who have verified the tweet with the id "1321211561046933514" ***and*** who are experts in "Infectious Diseases" (Note the use of the word "and" instead of "or" from the previous assignment!)

a) [7 pts] SQL Query:

```
SELECT DISTINCT p.number
FROM Verification v, Expertise e, Phone p
WHERE v.tweet_id = '1321211561046933514'
AND e.domain = 'Infectious Diseases'
AND v.user_id = p.user_id
AND e.user_id = p.user_id
```

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



7. [15 pts] Find tweet ids and the number of replies for each tweet that has one or more replies. List only the top five tweets that have the highest number of replies.

a) [12 pts] SQL Query:

```sql
SELECT t1.tweet_id,
       COUNT(*) AS rep_cnt
FROM Tweet t1, Tweet t2
WHERE t1.tweet_id = t2.replied_to_tweet
GROUP BY t1.tweet_id
HAVING rep_cnt >= 1 -- Optional
ORDER BY rep_cnt DESC
LIMIT 5;
```

b) [3 pts] Result: (5 Rows)

tweet_id	rep_cnt
1321470312509300738	3
1321293727105765376	3
1321457297441214464	2
1321497818146635776	2
1321418974886854656	2

8. [15 pts] For tweets that have two or more reactions (replies and/or quotes), print their tweet id along with their number of replies and number of quotes. (Note that for such tweets, the sum of replies and quotes should be 2 or more). Order the result by the number of reactions in largest-first order.

a) [12 pts] SQL Query:

```
SELECT t1.tweet_id,
       (SELECT COUNT(*) FROM Tweet t2 WHERE t2.replied_to_tweet =
t1.tweet_id) AS rep_cnt,
       (SELECT COUNT(*) FROM Tweet t3 WHERE t3.quoted_tweet = t1.tweet_id)
AS qt_cnt
FROM Tweet t1
WHERE (SELECT COUNT(*)
       FROM Tweet t3
       WHERE t3.quoted_tweet = t1.tweet_id) +
      (SELECT COUNT(*)
       FROM Tweet t2
       WHERE t2.replied_to_tweet = t1.tweet_id) >= 2
ORDER BY rep_cnt + qt_cnt DESC
```

b) [3 pts] Result (9 rows):

tweet_id	rep_cnt	qt_cnt
1321293727105765376	3	0
1321470312509300738	3	0
1321211561046933514	1	1
1321418974886854656	2	0
1321457297441214464	2	0
1321493086120210432	2	0
1321494210185342976	2	0
1321496681217548288	2	0
1321497818146635776	2	0