

Databases Mini-Project

● Design flaws of the giant table

The `bad_giant_table` contains lots of repeating data across the table as it is poorly normalised. It is not even in the first normal form (1NF) as a few attributes violate the condition such that every attribute should take atomic value. Other than the three attributes below, every other attribute takes atomic value. And all attributes take only a single value from the same domain.

- 1) The `user_name` attribute stores both first name and surname.
- 2) The `created_at` and `user_created_at` attributes stores both timestamp and timezone.

And functional dependencies are not treated properly as well. I could find a few functional dependencies in the `bad_giant_table`.

- 1) `tweet_id` → `created_at`, `text`, `retweet_count`, `tweet_source`, `user_id`, `hashtag`
- 2) `user_id` → `user_name`, `user_sceen_name`, `user_location`, `user_utc_offset`, `user_timezone`, `user_followers_count`, `user_friends_count`, `user_lang`, `user_description`, `user_status_count`, `user_created_at`
- 3) `user_time_zone` → `user_utc_offset`

Even though `tweet_id`, `user_id` and `user_time_zone` should be the primary key in separate relation, only `tweet_id` is set to a primary key in `bad_giant_table`.

● Redesigned database schema

I redesigned the `bad_giant_table` in the third normal form (3NF).

Exceptions:

- 1) Theoretically, `user_name` attribute should be divided into `user_surname` and `user_firstname`, however, I kept it with `user_name` for simplicity as query parts do not explicitly ask to extract surname or first name separately.
- 2) Theoretically, to normalise the Twitter dataset in the third normal form, `user_location`, `user_timezone` and `user_utc_offset` should separately form a relation. Because the `user_timezone` and `user_utc_offset` depend on `user_location` attribute, which is not the primary key of the users relation. However, in this Twitter dataset, `user_location` attribute contains some random words such as 'FarFarAway', 'In My Own World', 'Behind you', etc, which are not related to geographic location. So, functional dependencies are not formed.
- 3) There were issues with foreign key constraint while copying values from `bad_giant_table` to redesigned database as some tweets replied to tweets which are not included in the `bad_giant_table`. Theoretically, in reply relation, `reply(in_reply_to_status_id)` and `reply(in_reply_to_user_id)` attributes should be a foreign key to the PK of 'tweets' and 'users', respectively. However, I did not specify those attributes as a foreign key as it violates referential integrity.

DETAIL: Key (in_reply_to_status_id)=(260212715052756993) is not present in table "tweets".

Key (in_reply_to_user_id)=(165989321) is not present in table "users".

* The underlined attribute represents Primary Key (PK).

- Tweets = (tweet_id, created_at, text, retweet_count, tweet_source, user_id)
Attribute 'Tweets(user_id)' is a foreign key to the PK of 'Users'
- Reply = (tweet_id, in_reply_to_screen_name, in_reply_to_status_id, in_reply_to_user_id)
Attribute 'Reply(tweet_id)' is a foreign key to the PK of 'Tweets'
Attribute 'Reply(in_reply_to_status_id)' is a foreign key to the PK of 'Tweets'
Attribute 'Reply(in_reply_to_user_id)' is a foreign key to the PK of 'Users'
- Retweet = (tweet_id, retweet_of_tweet_id)
Attribute 'Retweet(tweet_id)' is a foreign key to the PK of 'Tweets'
Attribute 'Retweet(retweet_of_tweet_id)' is a foreign key to the PK of 'Tweets'
- Hashtag = (tweet_id, hashtag)
Attribute 'Hashtag(tweet_id)' is a foreign key to the PK of 'Tweets'
- Users = (user_id, user_name, user_screen_name, user_location, user_timezone, user_followers_count, user_friends_count, user_lang, user_description, user_status_count, user_created_at)
Attribute 'Users(user_timezone)' is a foreign key to the PK of 'Timezone'
- Timezone = (timezone, utc_offset)

In particular, in the bad_giant_table hashtags are stored in 6 attributes from hashtag 1 to hashtag 6 and a huge portion of those attributes are null (There were only 507, 332, 222 values in hashtag 4, hashtag 5, and hashtag 6, respectively, out of 110,574 tweets). This is badly inefficient in terms of both memory usage and writing queries. So, I separated hashtag relation and did not specify tweet_id as a primary key so that all hashtags can be copied to one column.

● Outputs of queries

1. Tweets, users and languages

1-1. How many tweets are there in total? 110574

1-2. How are these tweets distributed across languages?

Query:

```
select user_lang, count(tweet_id) from users inner join tweets on users.user_id = tweets.user_id
group by user_lang;
```

1-3. Compute, for each language, the fraction of total tweets that have that language setting, as well as the fraction of the number of users that have that language setting.

Query:

- *the fraction of total tweets that have that language setting*

```
with nr_tweets as (select count(tweet_id) from tweets)
select user_lang, count(tweet_id)::float / (select * from nr_tweets) as fraction_of_tot_tweets
from users inner join tweets on users.user_id = tweets.user_id
group by user_lang;
```

- the fraction of the number of users that have that language setting

with nr_users as (select count(user_id) from users)
 select user_lang, count(user_id)::float / (select * from nr_users) as fraction_of_nr_users
 from users group by user_lang;

1-2 & 1-3 Results:

From the left side, results for 1-2, 1-3 fraction of total tweets that have that language setting, and 1-3 fraction of the number of users that have that language setting.

user_lang	count	user_lang	fraction_of_tot_tweets	user_lang	fraction_of_nr_users
ar	1405	ar	0.01270642284804746	fr	0.0021293610953202233
ca	7	ca	6.330602130699803e-05	tr	0.0011658492874830181
cs	2	cs	1.8087434659142295e-05	en	0.6675113453515373
de	75	de	0.000678278799717836	fi	9.63511807837205e-06
el	2	el	1.8087434659142295e-05	ru	0.0036806151059381232
en	74077	en	0.6699314486226419	cs	1.92702361567441e-05
es	17910	es	0.16197297737261923	ja	0.0924585930800582
eu	1	eu	9.043717329571148e-06	sv	1.92702361567441e-05
fi	1	fi	9.043717329571148e-06	fil	8.671606270534846e-05
fil	9	fil	8.139345596614032e-05	eu	9.63511807837205e-06
fr	231	fr	0.0020890987031309347	de	0.0006840933835644156
hu	1	hu	9.043717329571148e-06	nl	0.0005781070847023231
id	1423	id	0.012869209759979742	id	0.013103760586585989
it	26	it	0.00023513665056884982	zh-cn	0.00020233747964581306
ja	9861	ja	0.08918009658690108	zh-tw	0.0001348916530972087
ko	691	ko	0.006249208674733662	ar	0.012795436808078082
msa	14	msa	0.00012661204261399606	no	9.63511807837205e-06
nl	61	nl	0.0005516667571038399	th	0.002090820623006735
no	1	no	9.043717329571148e-06	ca	6.744582654860436e-05
pl	4	pl	3.617486931828459e-05	pt	0.03549577500072263
pt	3977	pt	0.03596686381970445	ur	9.63511807837205e-06
ru	396	ru	0.0035813120625101742	hu	9.63511807837205e-06
(28 rows)		(28 rows)		(28 rows)	
sv	2	sv	1.8087434659142295e-05	pl	3.85404723134882e-05
th	233	th	0.002107186137790077	el	1.92702361567441e-05
tr	123	tr	0.001112377231537251	ko	0.006291732105176949
ur	1	ur	9.043717329571148e-06	it	0.00024087795195930125
zh-cn	26	zh-cn	0.00023513665056884982	es	0.16101245820767535
zh-tw	14	zh-tw	0.00012661204261399606	msa	0.00012525653501883666

2. Retweeting habits

2-1. What fraction of the tweets are retweets? 0.194

2-2. Compute the average number of retweets per tweet. 121

2-3. What fraction of the tweets are never retweeted? 0.669

2-4. What fraction of the tweets are retweeted fewer times than the average number of retweets? 0.936

From the result, we can see that approximately 94% of tweets are retweeted fewer times than the average number of retweets, and only 6% of tweets have more retweeted than the average. This means the distribution of retweet_count values is highly biased.

3. Hashtags

3-1. What is the number of distinct hashtags found in these tweets? 10158

Rank	Hashtag
1	ReasonsIFailAtBeingAGirl
2	RED
3	oomf
4	HonestyHour
5	EresGuapaSi
6	10PeopleYouTrulyLove
7	TeamFollowBack
8	TweetLikeAGirl
9	ImSingleBecause
10	WeAllGotThatOneFriend

Query:

Result:

user_lang	hashtag		
ar	سوريا	nl	TeamFollowBack
ar	غرد بصورة	nl	sweetdreams
ar	الكويت	nl	1000ADAY
ar	مسيرة كرامة وطن	nl	500ADAY
ca	SoyeIngrato	nl	Truste
ca	EseMomentoEnElQue	nl	FollowBack
cs	continuous	nl	Spotify
cs	fermentation	nl	voetbalnieuws
de	Hamburg	nl	TEAMFOLLOWBACK
de	Noporquemeenamoro	nl	TEAMFOLLOWWACK
de	Thesis	nl	F4F
de	Hannover	nl	TeamFollow
de	gratis	nl	actrices
de	Wittmund	nl	Verveel
de	tracking	nl	voetbal
de	A7	nl	TFBJP
de	Telmi	nl	FOLLOWNGAIN
de	Koblenz	nl	acteurs
de	jazz	nl	sport
de	nowplaying	nl	teek
de	typo12	nl	Breaking
		nl	TEAMASOCIAAL

[illegible]

If we add another condition such that a hashtag has to be used more than ten times to be classified as a popular hashtag, then we can get a more meaningful result than the previous one.

Queries:

```
with hashtag_rank as (with nr_hashtag_per_lang as (select user_lang, hashtag, count(hashtag) as nr_used
from users inner join tweets on users.user_id = tweets.user_id inner join hashtag
on tweets.tweet_id = hashtag.tweet_id group by user_lang, hashtag)
select user_lang, hashtag, nr_used, rank() over (partition by user_lang order by nr_used desc) as rank
from nr_hashtag_per_lang)
select user_lang, hashtag from hashtag_rank where rank < 4 and nr_used > 10;
```

Result:

user_lang	hashtag
ar	سوريا
ar	غرد_بصورة
ar	مسيرة_كرامة_وطن
ar	الكويت
en	ReasonsIFailAtBeingAGirl
en	RED
en	oomf
es	EresGuapaSi
es	LoAdmito
es	SoloElVenezolano
id	TeamFollowBack
ja	countkun
ja	sougofollow
ja	followmejp
pt	YGDONTFORGETBRAZIL
ru	gameinsight

(16 rows)

4. Replies

4-1. How many tweets are neither replies, nor replied to? 88136

4-2. If a user user1 replies to another user user2, what is the probability that they have the same language setting?
0.142

4-3. How does this compare to the probability that two arbitrary users have the same language setting?