Zadanie 6

Autor: Samuel Schmidt

AIS id: 103120

Github odkaz

GitHub odkaz

GitHub classroom: https://github.com/FIIT-DBS/zadanie-pdt-schmidt-8

Uloha 1

Zadanie: Navrhnite dátový model (kolekcie a formát dokumentov) v MongoDB pre dataset tweetov, ktorý bude využívaný mobilnou aplikáciou, ktorá bude: a. Zobrazovať tweety jednotlivých používateľov vo forme feedov b. Zobrazovať jednotlivé tweety a ich retweets

Model som si rozdelil na dve kolekcie a to tweets a authors. Authors model obsahuje len dáta o autorovi, zatiaľ čo tweets obsahuje zvyšné informácie vytiahnuté z tabuliek (podla prvého zadania). Konkrétny príklad pre tieto modely, na konkrétnom modely uvádzam nižšie:

Tweets:


```
{
    "id": 1078758509397884930,
    "author_id": 1023900821895950336,
    "content": "You see BACK HOME we are big FAMOUS for work we do. Here in
MISERABLE TINY ISLAND we are made IGNORED ☹. Maybe if new REFERENDUM we not help
AGAIN. United England no make LAUGHTER then! RU💔GB https://t.co/y5c6IgRSNf",
    "possibly_sensitive": false,
    "language": "en",
    "source": "Twitter Web Client",
    "retweet_count": 25,
    "reply_count": 0,
    "like_count": 34,
    "quote_count": 1,
    "created_at": "2018-12-28T21:04:05+00:00",
    "hashtags": null,
    "links": [
        {
            "url":
"https://twitter.com/JuliaDavisNews/status/1078680196394479616",
            "title": null,
            "description": null
        }
    ],
    "annotations": [
        {
            "type": "Place",
```

```
            "value": "United England",
            "probability": 0.8113
        }
    ],
    "context_annotations": [
        {
            "domain": {
                "name": "Person",
                "description": "Named people in the world like Nelson Mandela"
            },
            "entity": {
                "name": "Donald Trump",
                "description": "45th US President, Donald Trump"
            }
        }
    ],
    "conversation_references": [
        {
            "parent_id": 1078680196394479616,
            "type": "replied_to"
        }
    ]
}
```


Authors:


```
{
    "id": 1023900821895950336,
    "name": "Proud Bear ʀᴜ",
    "username": "Pr0ud_Bear",
    "description": "Collective of GRU agents celebrating NOT-ENOUGH-FAMOUS #BREXIT
collaboration https://t.co/n8ALqDYuuy 📧 informatsiya(at)https://t.co/n8ALqDYuuy
DMs OPEN",
    "tweet_count": 183,
    "listed_count": 4,
    "followers_count": 1842,
    "following_count": 1
}
```


Author-Tweet vzťah je typ `document referencing` vzťahu a vybraný bol z toho dôvodu, že v opačnom prípade ak by sme u autorov drzali id tweetov, vznikali by veľmi nevyvážené dokumenty (niekto rád popíše niekto nie). Tento istý typ vzťahu ale v rámci jednej kolekcie sa dá nájsť v conversation_references Tweet-Tweet. Zvyšné parametre majú embedded vzťah k Tweetom, napríklad Hashtagy.

Ak by sme chceli Zobraziť tweety jednotlivých používateľov vo forme tweetov, stačilo by nám pri týchto modeloch následovná query:

```
db.tweets.find( { author_id: <author_id> } )
```

Zobraziť tweety a retweety vieme dosiahnúť pomocou atribútu `conversation_references`. Každý Tweet môže mať teda niekoľko parent_id a ak je jeho typ 'replied_to', takto by sme vedeli vyhľadať všetky ostatné retweety daného tweetu(kedze dany parent_id retweetoval aktuálny).

```
db.tweets.find(
    { "conversation_references":
        { $elemMatch:
            { parent_id: <source_tweet_id>, type: "replied_to" }
        }
    }
)
```

Uloha 2

Zadanie: Nainštalujte alebo využite online inštanciu MongoDB servera, do ktorého importujte všetky tweets (a s nimi spojené data – anotácie, referencie, odkazy a informácie o kontexte) zo dňa 24.02.2022.

Na začiatku som si vytvoril indexy, aby sa znížil čas query

```
CREATE INDEX IF NOT EXISTS fk_index_1 ON public.conversations(author_id);
CREATE INDEX IF NOT EXISTS fk_index_2 ON
public.conversation_hashtags(conversation_id);
CREATE INDEX IF NOT EXISTS fk_index_3 ON public.conversation_hashtags(hashtag_id);
CREATE INDEX IF NOT EXISTS fk_index_4 ON
public.context_annotations(conversation_id);
CREATE INDEX IF NOT EXISTS fk_index_5 ON
public.conversation_references(conversation_id);
CREATE INDEX IF NOT EXISTS fk_index_6 ON
public.conversation_references(parent_id);
CREATE INDEX IF NOT EXISTS fk_index_7 ON public.annotations(conversation_id);
CREATE INDEX IF NOT EXISTS fk_index_8 ON public.conversations(id);
CREATE INDEX IF NOT EXISTS fk_index_9 ON public.links(conversation_id);
CREATE INDEX IF NOT EXISTS fk_index_10 ON public.conversations(created_at);
```

Vytvoril som si query pre autorov a tweety, podobnu ako v predošlom zadaní a pomocou nich exportoval 5000 záznamov ako vzorku slúžiacu na demonštráciu danej úlohy.

Query pre autorov:

```
COPY( SELECT row_to_json (result) FROM (
SELECT authors.*
FROM authors WHERE '2022-02-24' in (SELECT created_at::date FROM conversations
WHERE author_id = authors.id)
GROUP BY authors.id LIMIT 5000
) AS result
) TO 'path\authors_5k.jsonl' WITH(HEADER FALSE);
```

Query pre tweety:

```
COPY (SELECT row_to_json(result)
FROM (
    select  conversations.id,
        conversations.author_id,
        regexp_replace(conversations.content,
E'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029"“]+', ' ', 'g' ) as content,
        conversations.possibly_sensitive,
        regexp_replace(conversations.language,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' )as language,
        regexp_replace(conversations.source,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' )as source,
        conversations.retweet_count,
        conversations.reply_count,
        conversations.like_count,
        conversations.quote_count,
        conversations.created_at,
        (SELECT json_agg(distinct regexp_replace(hashtags.tag,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' ))
         FROM hashtags WHERE hashtags.id in (SELECT hashtag_id FROM
conversation_hashtags WHERE conversation_id = conversations.id)) as hashtags,
        (SELECT json_agg(distinct jsonb_build_object(
            'url', regexp_replace(links.url,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' ),
            'title', regexp_replace(links.title,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' ),
            'description', regexp_replace(links.description,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' )))
         FROM links WHERE conversation_id = conversations.id) as links,
        (SELECT json_agg(distinct jsonb_build_object(
            'value', annotations.value,
            'type', annotations.type,
            'probability', annotations.probability
        ))FROM annotations WHERE conversation_id = conversations.id) as
annotations,
        (SELECT json_agg(distinct jsonb_build_object(
            'domain', (SELECT jsonb_build_object(
                'name',  regexp_replace(context_domains.name,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' ),
                'description', regexp_replace(context_domains.description,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' ))
                        FROM context_domains WHERE id =
context_annotations.context_domain_id),
            'entity', (SELECT jsonb_build_object(
                'name', regexp_replace(context_entities.name,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' ),
                'description', regexp_replace(context_entities.description,
e'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029""]+', ' ', 'g' ))
                        FROM context_entities WHERE id =
context_annotations.context_entity_id)
        )) FROM context_annotations WHERE conversation_id = conversations.id) as
```

```
context_annotations,
--       (SELECT json_agg( json_build_object( 'parent_id', parent_id, 'type', type
)) as conversation_references
--          FROM
            (SELECT json_agg(distinct jsonb_build_object(
                'parent_id',  conversation_references.parent_id,
                'type', conversation_references.type,
                'id', (SELECT conversations.id FROM conversations WHERE id =
conversation_references.parent_id),
                'content', regexp_replace((SELECT conversations.content FROM
conversations WHERE id = conversation_references.parent_id),
E'[\\n\\r\\f\\u000B\\u0085\\u2028\\u2029"“]+', ' ', 'g' ),
                'author', (SELECT jsonb_build_object(
                    'id', authors.id,
                    'name', authors.name,
                    'username', authors.username) FROM authors WHERE id = (SELECT
conversations.author_id FROM conversations WHERE id =
conversation_references.parent_id)),
                'hashtags', (SELECT json_agg(distinct hashtags.tag) FROM hashtags
WHERE hashtags.id in (SELECT hashtag_id FROM conversation_hashtags WHERE
conversation_id = (SELECT conversations.id FROM conversations WHERE id =
conversation_references.parent_id)))
            )) FROM conversation_references WHERE conversation_id =
conversations.id) as  refconversation_references

from conversations
WHERE '2022-02-24' = created_at::date
group by conversations.id
limit 5000
) AS result
) TO 'C:\Users\Samuel
Schmidt\Desktop\Files\UNI\ING\1.rocnik_zimny\PDT\Zadanie_6\tweets_5k.jsonl'
WITH(HEADER FALSE);
```

Nepodarilo sa mi však dalej rozbehat Mongo, preto som nestihol urobit import a tym padom ani dalsie zadanie som nezvladol dokoncit 😟.