

Datenbanksysteme SS17

Projektdokumentation

Freie Universität Berlin
Fachbereich Informatik

Ralitsa Dineva, Nedelcho Petrov, Gulnaz Kazakbaeva

2. Juni 2017

Einleitung

Dieses Projekt findet statt im Rahmen des Moduls Datenbanksysteme. Das Ziel ist eine Web-Anwendung anzufertigen, die die Daten aus dem Datensatz american-election-tweets visualisiert und Abfragen beantworten kann. Der Datensatz enthält Information über die tweets von der Präsidentschaftskandidaten in der USA in 2016. Mit Hilfe dieser Daten wird mit SQL eine Datenbank erstellt. Danach kann durch verschiedene Operationen die abgefragte Information abgeleitet werden. Die Web-Anwendung wird benutzerfreundlich und leicht zu bedienen sein und die Abfrageergebnisse schön und klar dargestellt.

Das Projektentwicklung wird in drei Iterationen verlaufen:

1. Modellierung

- 1.2 Explorative Datenanalyse des Datensatzes
- 1.3 ER-Modell
- 1.4 Relationales Modell
- 1.5 Erstellung von Datenbank

2. Datenimport

- 2.1 Datenbankschema erstellen
- 2.2 Datenbereinigung
- 2.3 Datenimport
- 2.4 Webserver

3. Data Mining und Visualisierung

Unser Team

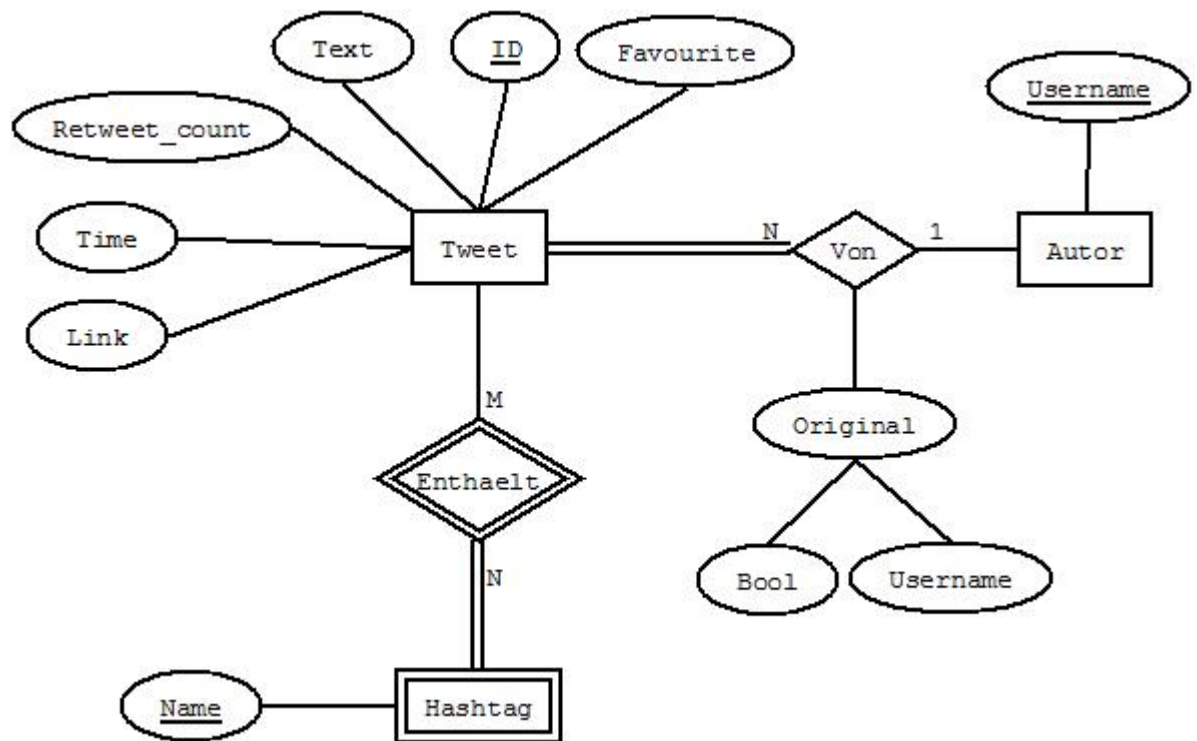
Auf dem Projekt arbeitet eine Gruppe von drei Studenten: Ralitsa Dineva, Nedelcho Petrov und Gulnaz Kazakbaeva. Sie sind alle Informatik Studenten an der FU Berlin. Gesammelt von verschiedenen Ecken der Erde um zusammen das Projekt zu erstellen, sie versuchen sich tief in der Lehre von Datenbanksystemen zu vertiefen und werden ihre Herzen und Schwitze bis den letzten Moment für das Erfolg geben.

1.2 Explorative Datenanalyse des Datensatzes

Der Datensatz ist eine Darstellung aller Tweets von Trump und Clinton vor den Präsidentschaftswahl in USA 2016. Die Hauptinformation in der Tabelle sind alle verschiedene Tweets. Zu jeden Tweet steht von wem und wann er gepostet war (Trump/Clinton). Falls er ein Retweet war, dann noch zusätzlich der Username des Originalautors. Im weiteren zeigen die üblichen Spalten ob der Tweet ein Zitat war, wie viele Benutzer den retweetet haben, wie viele ihn als favorite markiert haben und seine Internet Adresse, bzw mit welcher Gerät das getweetert wurde.

Die Information in den Spalten 'is quote status' und 'truncated' ist unnötig für das Ziel des Projekts und wird in seine weitere Entwicklung nicht betrachtet.

1.3 ER-Modell



Erklärung des Diagrams und der wichtigsten Entscheidungen:

Das Diagramm besteht aus den drei Entitätstypen *Tweet*, *Autor* und *Hashtag* (schwach) und die zwei Relationen *Von* und *Enthält*.

Die Entitätstypen:

- *Tweet*: steht für alle Tweets in dem Datensatz mit Schlüsselattribut *ID* und noch:
 - *Text*: Inhalt des Tweets.
 - *Time*: Die Zeit der Erscheinung.
 - *RetweetCount*: Anzahl von Retweets.
 - *Favourite*: Anzahl von Likes.
 - *Link*: URL Link des Tweets
- *Autor* steht für Hillary Clinton undrealDonaldTrump. Der Schlüsselattribut ist *Username*
- *Hashtag* steht für alle Hashtags mit Schlüsselattribut *Name* die Name des Hashtags. Das ist einen schwachen Entitätstyp, da er von *Tweet* durch einen Algorithmus* erzeugt wird.

*Der Algorithmus liest den Text des Tweets durch und macht aus jedem Wort das mit '#' anfängt eine Instanz.

Die Relationen:

- *Von* ist Relation zwischen *Tweet* (1:1) und *Autor* (1:N) die zeigt wer den Tweet getweetet hat. Um solche Tweets, die originell von Trump oder Clinton geschrieben sind nachvollziehen zu können, hat *Von* den Attribut *Original*. Er ist ein Tupel (Bool, Username), der falls der Tweet von jemand anderen geschrieben ist die Form (False, < Username des originellen Autors>) hat und sonst (True, NULL).
- *Enthält* zeigt welcher Tweet welche Hashtags in sich enthält.

1.4 Relationales Modell

Entitäten:

- *TWEET*(ID, Text, Time, Favourite, Retweet_count, Link)
- *HASHTAG*(Name)
- *AUTOR*(Username)

Relationen:

- *VON*(TID, Username, Original)
- *ENTHAELT*(TID, HName)

TID ist in beide Fällen der Fremdschlüssel von *Tweet* und *HName* ist von *Hashtag*.

Begründung des Funktionalitäts:

- Welche 'Tweets' waren am wichtigsten?
Man hat die favourite_count und retweet_count als Attributen und kann dann Operationen wie MAX und AVG und eigene Kriterien nutzen um die wichtigsten Tweets zu erzeugen.
- Welche Hashtags wurden am meisten verwendet?
Es wird für jede Instanz von *Hashtag* die Anzahl von Tweets in dem er vorkommt genommen:

```
SELECT Name, Count(*)
FROM ENTHAELT
WHERE
GROUPBY Name
```

Man konnte auch den favourite_count jedes einzelnes Tweet betrachten und dadurch noch besser auswählen.

- Wann traten insgesamt am meisten Hashtags auf?

Mittels:

$$\Pi_{Name, Time}((\text{Enthaelte}) \bowtie_{\{TID = ID\}} (\text{Tweet}))$$

entsteht eine Relation mit der Zeit der Erscheinung von jeden Hashtag. Davon kann berechnet werden wann traten am häufigsten Hashtags auf.

- Paaren (#A, #B) von Hashtags;
Welche Hashtags traten besonders häufig gemeinsam an?;
Wie hat sich die Häufigkeit der Verwendung eines speziellen Hashtags über die Zeit entwickelt?

```
SELECT A.Name, B.Name, TID
FROM ENTHAELT.A, ENTHAELT.B
WHERE A.TID = B.TID
```

Davon entstehen alle Paare von Hashtags und den Tweet in dem sie gemeinsam vorkommen. Der Attribut *Time* kann auch genommen werden um Häufigkeiten zu messen. Ähnlich kann jeden Hashtag einzeln betrachtet werden.

1.5 Erstellung von Datenbank

Siehe den Screenshots.

2.1 Datenbankschema erstellen

Relationeles model (siehe 1.4) in PSQL:

```
DROP TABLE IF EXISTS VON;
DROP TABLE IF EXISTS ENTHAELT;
DROP TABLE IF EXISTS HASHTAG;
DROP TABLE IF EXISTS AUTOR;
DROP TABLE IF EXISTS TWEET;
```

```
CREATE TABLE TWEET(
    ID SERIAL PRIMARY KEY,
    Text text NOT NULL,
    Time date,
    Favourite integer,
    Retweet_count integer,
    Link text);
```

```
CREATE TABLE HASHTAG(
    Name text PRIMARY KEY );
```

```
CREATE TABLE AUTOR(
    Username text PRIMARY KEY);
```

```
CREATE TABLE VON(
    TID integer REFERENCES TWEET(ID),
    Username text REFERENCES AUTOR(Username),
    Original text,
    PRIMARY KEY(TID, Username));
```

```
CREATE TABLE ENTHAELT(
    TID integer REFERENCES TWEET(ID),
    HName text REFERENCES HASHTAG(Name),
    PRIMARY KEY(TID, HName));
```

Link zu Repository: https://github.com/ralitsadineva/DBSProjekt/blob/master/create_tables.sql

2.2 Datenbereinigung

Operationen die auf den Daten angewendet sind:

- Hashtags in den Tweets finden und rausnehmen.
- Das 'T' zwischen Den und Uhrzeit mit Leerzeichen ersetzen.
- Das Link zu dem Tweet von dem Text rausnehmen.

Code:

```
import csv

#Opening the data file:
csvfile = open('american-election-tweets.csv')
creader = csv.DictReader(csvfile, delimiter=';')

read_file = [row for row in creader]

#Finding hashtags:
def extract_hash_tags(s):
    s = s.replace("# ", "#")
    return set(part[1:] for part in s.split() if part.startswith('#'))

#Extracting hashtags:
def get_all_hashtags(creader):
    hashtags = set()
    for row in creader:
        hashtags.update(extract_hash_tags(row['text']))
    return hashtags

#Formats the date and time from YY-MM-DD"T"HH-MM-SS
#to YY-MM-DD HH-MM-SS:
def format_timestamp(timestamp):
    return timestamp.replace("T", " ")
```



```
#Separates the text of a tweet from the link at the end of it:
def extract_links(text):
    links = [part for part in text.split() if part.startswith('https ')]
    if links == []:
        return []
    return links[len(links) - 1]
```

Link zu Repository: https://github.com/ralitsadineva/DBSProjekt/blob/master/read_and_correct.py

2.3 Datenimport

Code der den Datensatz eignet und dann in den Datenbank 'election' importiert:

```
import psycopg2
from read_and_correct import *

db = "election"
username = "postgres"
password = "ENTER YOUR PASSWORD"
host_ = "localhost"
port_ = "5432"
conn = psycopg2.connect(database = db, user = username, ...
... password = password, host = host_, port = port_)
print("Opened database successfully")

cur = conn.cursor()

def populate_tweets():
    print("Populating tweets...")
    for row in read_file:
        link = extract_links(row['text'])
        if link != []:
            url = link
        else:
            url = row['source_url']
        cur.execute('INSERT INTO TWEET (Text, Time, Favourite, ...
... Retweet_count, Link) VALUES (%s, %s, %s, %s, %s)', ...
...(row['text'], format_timestamp(row['time']), ...
...row['favorite_count'], row['retweet_count'], url))
    print("Finished populating tweets!")

def populate_hashtags():
    print("Populating hashtags...")
    hashtags = get_all_hashtags(read_file)
    for hashtag in hashtags:
        if len(hashtag) > 0:
```

```

        cur.execute("INSERT INTO HASHTAG VALUES (%s)", (hashtag, ))
    print("Finished populating hashtags!")

def populate_autor():
    print("Populating autor...")
    authors = set()
    for row in read_file:
        authors.add(row[ 'handle '])

    for author in authors:
        cur.execute("INSERT INTO AUTOR VALUES(%s)", (author, ))
    print("Finished populating autor!")

def populate_von():
    print("Populating von...")
    cur.execute("SELECT * FROM TWEET")
    rows = cur.fetchall()
    for row in rows:
        id = row[0]
        text = row[1]
        time = row[2]
        favourite = row[3]
        retweet_count = row[4]
        for row_file in read_file:
            if text == row_file[ 'text ' ] and str(time) == row_file[ 'time ' ]:
                if row_file[ 'is-retweet ' ] == 'True':
                    og = row_file[ 'original-author ' ]
                else:
                    og = row_file[ 'handle ' ]
                cur.execute("INSERT INTO VON VALUES(%s, %s, %s)", (id, og, ))
                break
    print("Finished populating von!")

def populate_enthaelt():
    print("Populating enthaelt...")
    cur.execute("SELECT * FROM TWEET")
    rows_tweet = cur.fetchall()

```

```

cur.execute("SELECT * FROM HASHTAG")
rows_hashtag = cur.fetchall()
for r_tweet in rows_tweet:
    id = r_tweet[0]
    text = r_tweet[1]
    for r_hash in rows_hashtag:
        if "#" + r_hash[0] in text.replace("# ", "#"):
            cur.execute("INSERT INTO ENTHAELT VALUES(%s, %s)", (id,
print("Finished populating enthaelt!")

```

```

populate_hashtags()
populate_autor()
populate_tweets()
populate_von()
populate_enthaelt()
conn.commit()
cur.close()
conn.close()

```

Link zu Repository: https://github.com/ralitsadineva/DBSProjekt/blob/master/populate_db.py

2.4 Webserver

Es wurde einen Apache Tomcat installiert und cofiguriert nach diesem Tutroi-

al: [https://www.howtoforge.com/tutorial/how-to-install-apache-tomcat-8-5-on-ubuntu-](https://www.howtoforge.com/tutorial/how-to-install-apache-tomcat-8-5-on-ubuntu-14-04-lts/)