Begründung Cassandra

Cassandra erschien dem Team aufgrund der Vorlesungsinhalte als eine sichere Wahl. Im Nachgang kann festgestellt werden, dass Cassandra trotz aller Vorteile für ein Twitter-Szenario unangemessen ist. Viele mögliche (und auch zum Teil in der Augabenstellung geforderten) Queries sind schlichtweg ohne massives Aufblähen der Datenmenge nicht darstellbar. Im Anbetracht alternativer Technologien und deren Möglichkeiten, Interaktionen in einem sozialen Netzwerk zu speichern und zu prozessieren, ist die Wahl von Cassandra für diese Aufgabenstellung kaum rechtfertigbar.

Setup Cassandra

Da sich das Team für Cassandra mit einem (Lucene-Plugin)[https://github.com/Stratio/cassandra-lucene-index] entschieden hat konnte das offizielle Cassandra-Image nicht verwendet werden. Hier wurde, nach vielen zeitaufwändigen Iterationen dann schlussendlich von einem "FROM SCRATCH" gebautem Image abgesehen und als Grundlage auf (jeffharwell/cassandra-lucene)[https://github.com/jeffharwell/cassandra-lucene] zurückgegriffen.

Auf Basis dieses Images wurde das Team-Eigene Dockerfile sowie die Team-Eigene docker-compose.yml geschrieben. Da es in immer wieder zu Abstürzen aufgrund einer korrupten SSTable im Base-Image kam wurde ein Healthcheck angepasst um diese Datei zu löschen.

Ebenfalls kam es, bei unterdimensionierung der Speicherressourcen zu äußerst instabilen Containern. Daher wurde die verfügbare Speichermenge auf 4GB gesetzt.

Setup des Cassandra-Clusters

Im Ordner docker-stratio genügt es den Befehl docker-compose up zu verwenden um den Cluster zu starten. Da es sich hierbei um einen lokalen Testcluster handelt sind die Authentisierungsoptionen abgeschaltet. Nach der anfänglichen Start-Phase des Clusters kann mithilfe von docker psgepfüft werden, ob auch alle Nodes vorhanden sind (cass1, cass2, cass3).
Mit `docker exec -it cass1 cqlsh` sowie darauf folgend describe keyspaces kann überprüft werden, ob CQL korrekt funktioniert.

Mithilfe von `docker exec -it cass1 nodetool status` lässt sich der Cluster sowie sein Status beschreiben.

CSVs in Container kopieren

Da die generierten CSV Dateien (Generierung anhand der Skripts **/cleaning_scripts/tweets.py** und **/cleaning_scripts/follower_relations.py**) leider zum Teil die maximale git-Größe übersteigen müssen diese manuell vom jeweiligen Speicherort importiert werden.

```
cd Documents/UNI/BDEA/CA_3/BDEA3/
```

```
docker cp res_follower.csv cass1:/data/res_follower.csv
docker cp tweets_clean.csv cass1:/data/tweets_clean.csv
```

Aufsetzen der Tabellen

```
docker exec -it cass1 cqlsh

CREATE KEYSPACE twitter WITH REPLICATION = {'class': 'SimpleStrategy',
'replication_factor': 3};

USE twitter;

CREATE TABLE twitter.tweets_by_authors (author TEXT, date_time TIMESTAMP,
tweet_id BIGINT, content TEXT, language TEXT, number_of_likes INT,
number_of_shares INT, PRIMARY KEY ((author), date_time, tweet_id)) WITH
CLUSTERING ORDER BY (date_time DESC);

CREATE TABLE twitter.caches_by_users (username TEXT, date_time
TIMESTAMP, tweet_id BIGINT, author TEXT, content TEXT, language TEXT,
number_of_likes INT, number_of_shares INT, PRIMARY KEY ((username),
date_time, tweet_id)) WITH CLUSTERING ORDER BY (date_time DESC);

CREATE TABLE twitter.follower_relations_by_users (username text, rel_type text,
rel_target_id BIGINT, rel_target_username text,  PRIMARY KEY ((username),
rel_type, rel_target_id));
```

Tweets Index für Lucene anlegen

```
CREATE CUSTOM INDEX tweets_index ON twitter.tweets_by_authors () USING
'com.stratio.cassandra.lucene.Index' WITH OPTIONS = {'refresh_seconds': 1,
'schema': '{fields: {author: {type: "text", analyzer: "english"}, date_time: {type: "date",
pattern: "yyyy-MM-dd HH:MM:SS"}, tweet_id: {type: "integer"}, content: {type:
"string"}, language: {type: "string"}, number_of_likes: {type: "integer"},
number_of_shares: {type: "integer"}}}';
```

Follower Relations importieren

```
COPY twitter.follower_relations_by_users (username, rel_type,
rel_target_username, rel_target_id) FROM 'data/res_follower.csv' WITH HEADER =
TRUE;
```

Testen, ob die Daten erfolgreich importiert wurden

```
SELECT COUNT(*) FROM twitter.follower_relations_by_users WHERE username =
'katyperry';
```

Tweets importieren

```
COPY twitter.tweets_by_authors (author, content, date_time, tweet_id, language,
number_of_likes, number_of_shares) FROM 'data/tweets_clean.csv' WITH HEADER
= TRUE;
```

Testen, ob die Daten erfolgreich importiert wurden:

```
SELECT COUNT(*) FROM twitter.tweets_by_authors;
```