Graph Analytics

Modeling Chat Data using a Graph Data Model

A graph model is used to illustrate the interactions between users. A user(node) can interact(creating edges) with others by chatting with others in a session.

Creation of the Graph Database for Chats

Describe the steps you took for creating the graph database. As part of these steps

i) Write the schema of the 6 CSV files

File: chat_create_team_chat.csv
A line is added to this file when a player creates a new chat with their team.
Example:
userid, teamid, TeamChatSessionID, timestamp
559,48,6288,14567

File: chat_item_team_chat.csv
Creates nodes labeled ChatItems. Column 0 is User id, column 1 is the TeamChatSession id, column 2 is the ChatItem id (i.e., the id property of the ChatItem node), column 3 is the timestamp for an edge labeled "CreateChat". Also create an edge labeled "PartOf" from the ChatItem node to the TeamChatSession node. This edge should also have a timeStamp property using the value from Column 3.
Example:
userid, teamchatsessionid, chatitemid, timestamp
1956,6299,6305,1464235803

File: chat_join_team_chat.csv
Creates an edge labeled "Joins" from User to TeamChatSession. The columns are the User id, TeamChatSession id and the timestamp of the Joins edge.
Example:
userid, TeamChatSessionID, teamstamp
559,6288,12345

File: chat_leave_team_chat.csv
ERD table: chat_leave_team_chat
Creates an edge labeled "Leaves" from User to TeamChatSession. The columns are the User id, TeamChatSession id and the timestamp of the Leaves edge.

Example:
userid, teamchatsessionid, timestamp
1244,6821,1464241204.0

File: chat_mention_team_chat.csv
Creates an edge labeled "Mentioned". Column 0 is the id of the ChatItem, column 1 is the id of the User, and column 2 is the timeStamp of the edge going from the chatItem to the User.
Example:

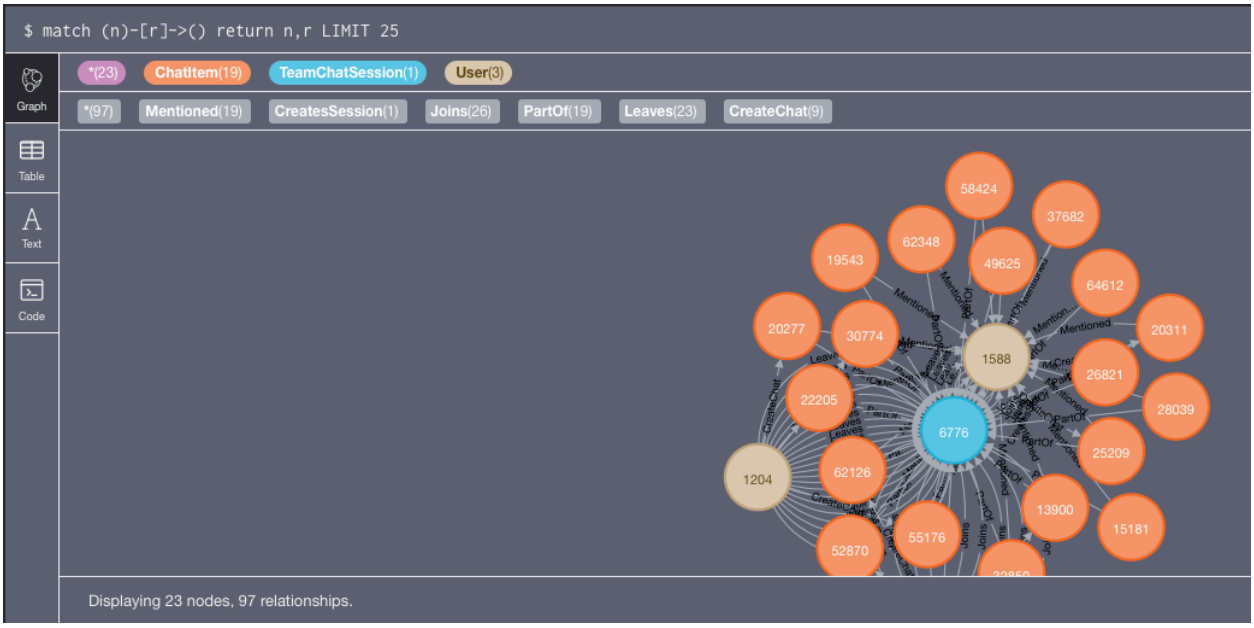
ChatItem, userid, timeStamp
6349,2508

File: chat_respond_team_chat.csv
A line is added to this file when player with chatid2 responds to a chat post by another player with chatid1.
Example:
chatid1, chatid2,timestamp
6326,6305,21564

ii) Explain the loading process and include a sample LOAD command

```
CREATE CONSTRAINT ON (u:User) ASSERT u.id IS UNIQUE;
CREATE CONSTRAINT ON (t:Team) ASSERT t.id IS UNIQUE;
CREATE CONSTRAINT ON (c:TeamChatSession) ASSERT c.id IS UNIQUE;
CREATE CONSTRAINT ON (i:ChatItem) ASSERT i.id IS UNIQUE;

LOAD CSV FROM "file:///chat-data/chat_create_team_chat.csv" AS row
MERGE (u:User {id: toInteger(row[0])})
MERGE (t:Team {id: toInteger(row[1])})
MERGE (c:TeamChatSession {id: toInteger(row[2])})
MERGE (u)-[:CreatesSession{timeStamp: row[3]}]->(c)
MERGE (c)-[:OwnedBy{timeStamp: row[3]}]->(t)

LOAD CSV FROM "file:///chat-data/chat_join_team_chat.csv" AS row
MERGE (u:User {id: toInteger(row[0])})
MERGE (c:TeamChatSession {id: toInteger(row[1])})
MERGE (u)-[:Joins{timeStamp: row[2]}]->(c)

LOAD CSV FROM "file:///chat-data/chat_leave_team_chat.csv" AS row
MERGE (u:User {id: toInteger(row[0])}) MERGE (c:TeamChatSession {id: toInteger(row[1])})
MERGE (u)-[:Leaves{timeStamp: row[2]}]->(c)

LOAD CSV FROM "file:///chat-data/chat_item_team_chat.csv" AS row
MERGE (u:User {id: toInteger(row[0])})
MERGE (c:TeamChatSession {id: toInteger(row[1])})
MERGE (i:ChatItem {id: toInteger(row[2])})
MERGE (u)-[:CreateChat{timeStamp: row[3]}]->(i)
MERGE (i)-[:PartOf{timeStamp: row[3]}]->(c)

LOAD CSV FROM "file:///chat-data/chat_mention_team_chat.csv" AS row
MERGE (i:ChatItem {id: toInteger(row[0])})
MERGE (u:User {id: toInteger(row[1])})
MERGE (i)-[:Mentioned{timeStamp: row[2]}]->(u)

LOAD CSV FROM "file:///chat-data/chat_respond_team_chat.csv" AS row
MERGE (i:ChatItem {id: toInteger(row[0])})
MERGE (w:ChatItem {id: toInteger(row[1])})
MERGE (i)-[:ResponseTo{timeStamp: row[2]}]->(w)
```

iii) Present a screenshot of some part of the graph you have generated. The graphs must include clearly visible examples of most node and edge types. Below are two acceptable examples. The first example is a rendered in the default Neo4j distribution, the second has had some nodes moved to expose the edges more clearly. Both include examples of most node and edge types.



Finding the longest conversation chain and its participants

Report the results including the length of the conversation (path length) and how many unique users were part of the conversation chain. Describe your steps. Write the query that produces the correct answer.

a. There are 9 chats involved.

b. 5 users participated in this chain

```
$ match p = (u)-[:ResponseTo*]->(v) return length(p) order by length(p) desc limit 1
```

Table

length(p)
9

Text

```
$ match p = (u)-[:ResponseTo*]->(v) where length(p) = 9 with p match (u)-[:CreateChat]->(i) where i in nodes(p) return count(distinct u)
```

Table

count(distinct u)
5

Text

Code

Analyzing the relationship between top 10 chattiest users and top 10 chattiest teams

Describe your steps from Question 2. In the process, create the following two tables. You only need to include the top 3 for each table. Identify and report whether any of the chattiest users were part of any of the chattiest teams.

Chattiest Users

We count the number of edges/chats user created.

Chattiest Teams

We count the total number of chats the team made.

```
$ match (u)-[:CreateChat*]->(i) return u.id, count(i) order by count(i) desc limit 10
```

u.id	count(i)
394	115
2067	111
1087	109
209	109
554	107
1627	105
999	105
516	105
461	104
668	104

```
$ match (i)-[:PartOf*]->(c)-[:OwnedBy*]->(t) return t.id, count(c) order by count(c) desc limit 10
```

t.id	count(c)
null	3850163
82	1324
185	1036
112	957
18	844
194	836
129	814
52	788
136	783
146	746

Only team 52 is part of the top 10 chattiest teams. Hence, most of the chattiest users are not in the chattiest teams.

How Active Are Groups of Users?

Firstly, we create the InteractsWith relationship.

Second, we get the neighbours of each of the chattiest users

Finally, we query each user in the neighbourhood and sum up the results and calculate the coefficient by dividing the sum with the number of possible interactions.

Most Active Users (based on Cluster Coefficients)

User ID	Coefficient
209	0.95
394	1.00
2067	0.93