Graph Analytics

Modeling Chat Data using a Graph Data Model

Our principal nodes are Users, Teams, Team Chat Sessions and Chat items. A User could create new chat. All the chats are part of a team chat session. A Chat item could respond to an other chat item or maybe Mention to a user. All users could join or leaves a chat session. Finally all chat sessions are owned by a Team.

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

chat_create_team_chat.csv
userid,teamid,TeamChatSessionID,timestamp
chat_item_team_chat.csv
userid,teamchatsessionid,chatitemid,timestamp
chat_join_team_chat
userid,TeamChatSessionID,teamstamp
chat_leave_team_chat.csv
userid,teamchatsessionid,timestamp
chat_mention_team_chat.csv
ChatItem,userid,timeStamp
chat_respond_team_chat.csv
chatid1,chatid2,timestamp

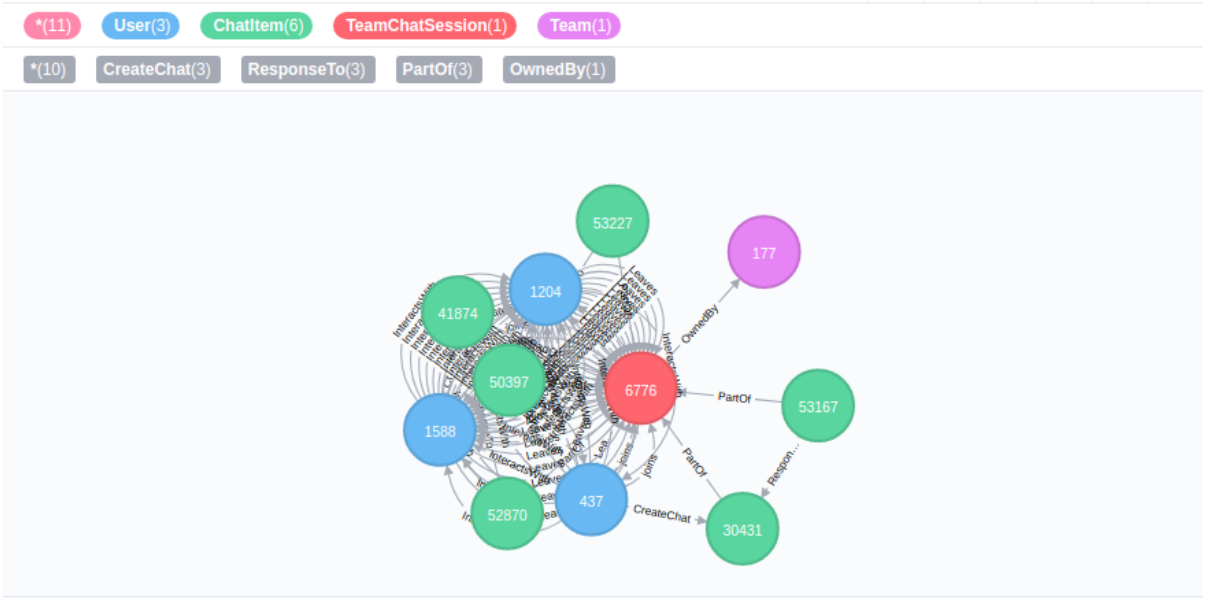
ii) Explain the loading process and include a sample LOAD command

LOAD CSV FROM "file:///chat_create_team_chat.csv" as row
MERGE (u:User {id: toInteger(row[0])})

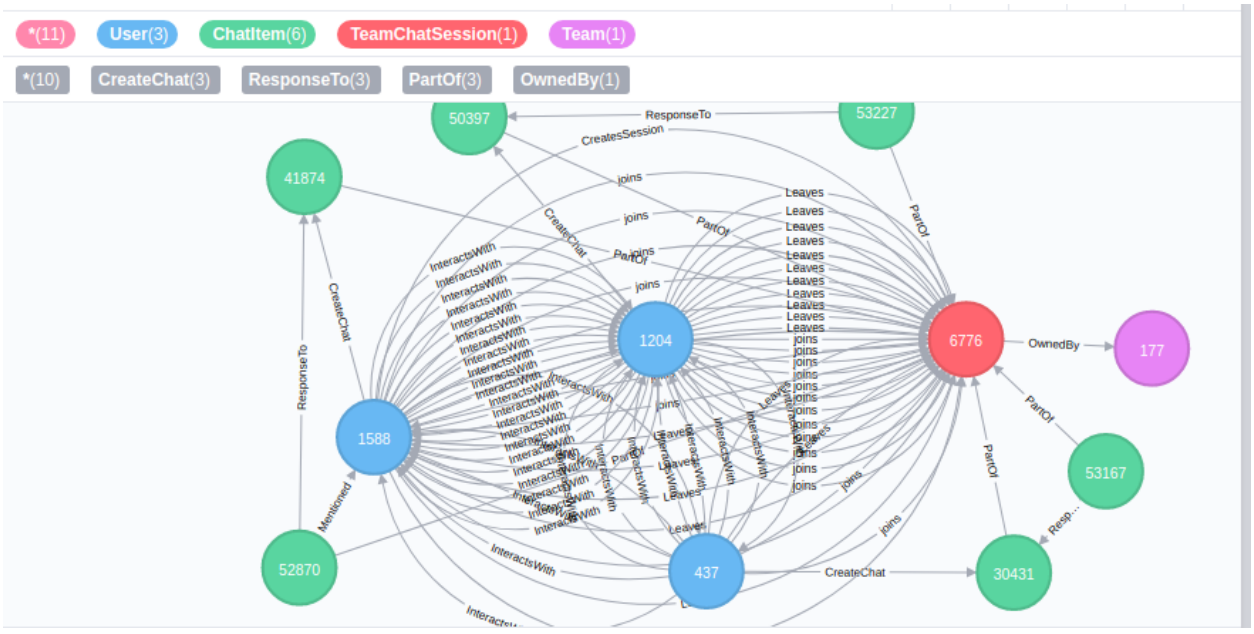
MERGE (t:Team {id: toInteger(row[1])})

```
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_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_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_mention_team_chat.csv" as row
MERGE (u:User {id: toInteger(row[1])})
MERGE (ch:ChatItem {id: toInteger(row[0])})
MERGE (ch)-[:Mentioned{timeStamp: row[2]}]->(u)
LOAD CSV FROM "file:///chat_respond_team_chat.csv" as row
MERGE (ch1:ChatItem {id: toInteger(row[0])})
MERGE (ch2:ChatItem {id: toInteger(row[1])})
    MERGE (ch1)-[:ResponseTo{timeStamp: row[2]}]→(ch2)
```

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.



Displaying 11 nodes, 92 relationships.



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.

Length of the conversation: 9

MATCH p =(a)-[:ResponseTo*]->(b)

return length(p)

order by length(p) desc

limit 1

```
$ MATCH p =(a)-[:ResponseTo*]->(b) return length(p)  order by length(p) desc limit
```

table	length(p)
	9

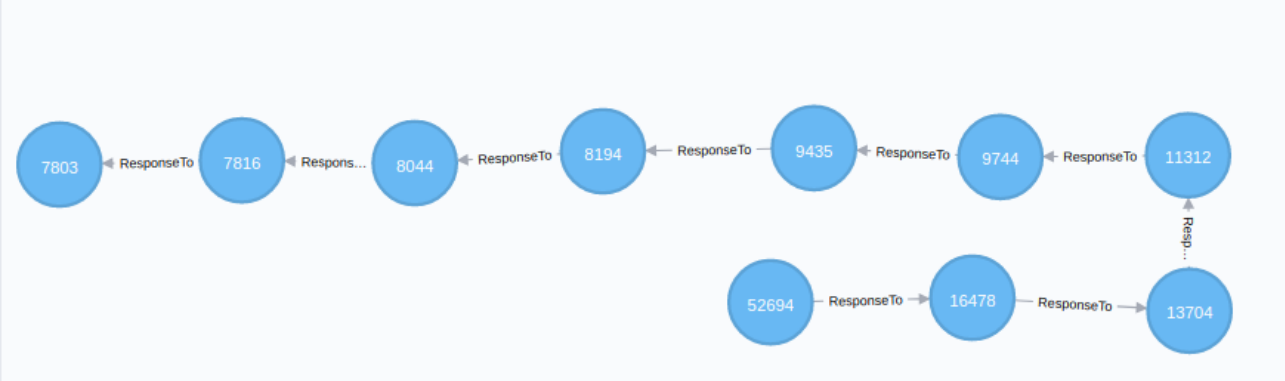
Conversation

MATCH p =(a)-[:ResponseTo*]->(b)

return p

order by length(p) desc

limit 1



Distinct Users: 5

MATCH p =(a)-[:ResponseTo*]->(b)

with p

order by length(p) desc limit 1

match p1=(x)-[:CreateChat]->(y)

where y in nodes(p)

```
$ MATCH p =(a)-[:ResponseTo*]->(b) with p  order by length(p) desc limit 1 match p1…
```

Table

count(distinct x)
5

Text

return count(distinct x)

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 User

```
$ MATCH (n:User)-[r:CreateChat]->(m:ChatItem) RETURN n.id,COUNT(n) ORDER BY COUNT(n) DESC LIMIT 10
```

Table

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

Text

Code

Chattiest Team

```
$ MATCH (n:ChatItem)-[r:PartOf]->(n1:TeamChatSession)-[r1:OwnerBy]->(n2:Team) RETURN n2.id,count(n) ORDER BY count(n) DESC LIMIT 10
```

Table | Text | Code

n2.id	count(n)
82	1324
185	1036
112	957
18	844
194	836
129	814
52	788
136	783
146	746
81	736

Chattiest Users

Users	Number of Chats
394	115
2067	111
209	109
1087	109
554	107
516	105
1627	105
999	105
668	104
461	104

Chattiest Teams

Teams	Number of Chats
82	1324
185	1036
112	957
18	844
194	836
129	814
52	788
136	783
146	746
81	736

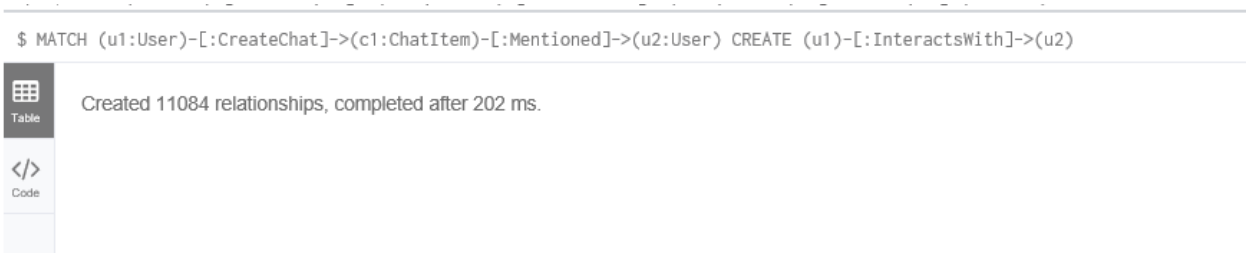
Finally, present your answer, i.e. whether or not any of the chattiest users are part of any of the chattiest teams.

The Chattiest User id 999 is part of Team id 52 which is also among top 10 Chattiest Teams.

How Active Are Groups of Users?

Describe your steps for performing this analysis. Be as clear, concise, and as brief as possible. Finally, report the top 3 most active users in the table below.

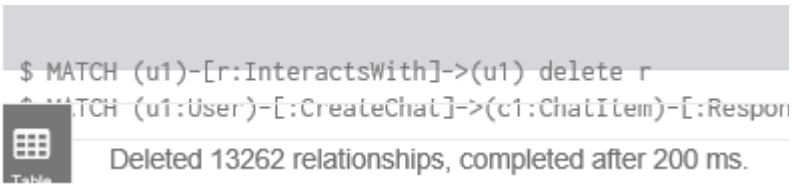
1. Created InteractWith Edge between the users based on Mentioned Edge



2.Created InteractWith Edge between the users based on ResponseTo Edge

```
MATCH (u1:User)-[:CreateChat]->(c1:ChatItem)-[:ResponseTo]->(c2:ChatItem)
WITH u1,c1,c2
MATCH (u2:User)-[:CreateChat]->(c2)
CREATE (u1)-[:InteractsWith]->(u2)
```

3. Delete the Self Loops Edges of InteractWith



Most Active Users (based on Cluster Coefficients)

User ID	Coefficient
209	0.95
554	0.90
1087	0.80