Graph Analytics

Modeling Chat Data using a Graph Data Model

The graph can be described as follow: a user (User) can create a chat (CreateChat), this chat (ChatItem) is a part of (PartOf) a team chat session (TeamChatSession) owned by (OwnedBy) a team (Team).

Also a user (User) can create (CreatesSession), join (Joins) or leave (Leaves) a team chat session (TeamChatSession).

Inside a chat (ChatItem) a user (User) can be mentioned (Mentioned). And finally to answer (ResponseTo) to a chat (ChatItem) is also a chat (ChatItem).

Convention:
User : all names painted in yellow are Nodes of the graph.
CreatesSession : all names painted in green are Edges of the graph.

Creation of the Graph Database for Chats

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

i) The schema of the 6 CSV files:

File: chat_create_team_chat.csv
userid : Id of the user creating a new ChatSession node
teamid : Id of the team which the user belongs to
teamchatsessionid : id of the newly chat session created
timestamp : timestamp of the creation of this new node.

File: chat_item_team_chat.csv
userid : Id of the user creating a new ChatItem node
teamchatsessionid : Id of the team chat session rattached to this ChatItem Node
chatItemid : id of the newly chat item created
timestamp : timestamp of the creation of this new node. Will serve to generate 2 edges: CreateChat and PartOf respectively coming from a User node and TeamSessionChat node.

File: chat_join_team_chat.csv
userid : : Id of the user joinng a new ChatSession
TeamChatSessionID : Chatsession id that is joined by the user
timestamp : timestamp of the creation of this new edge.

File: chat_leave_team_chat.csv
userid : Id of the user leaving a new ChatSession
TeamChatSessionID : Chatsession id that is left by the user
timestamp : timestamp of the creation of this new edge.

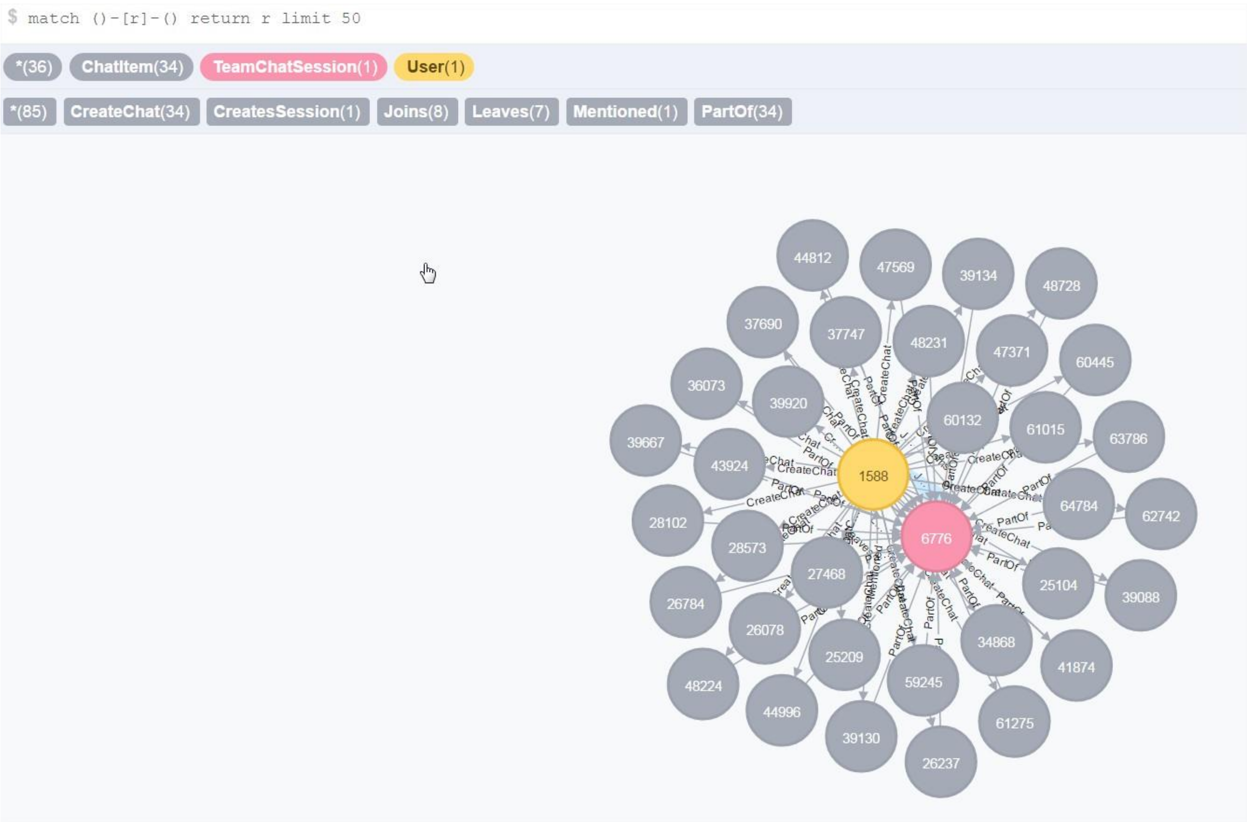
File: chat_mention_team_chat.csv
ChatItem : id from the chat item that mentioned an user
userid : user id mentioned by the Chat Item
timestamp : used to make the edge between the chatitem and the user node.

File: chat_respond_team_chat.csv
userid1 : user that create a new chatitem to respond to another second user
userid2 : that second user
timestamp : used to make the edge between the chatitems.

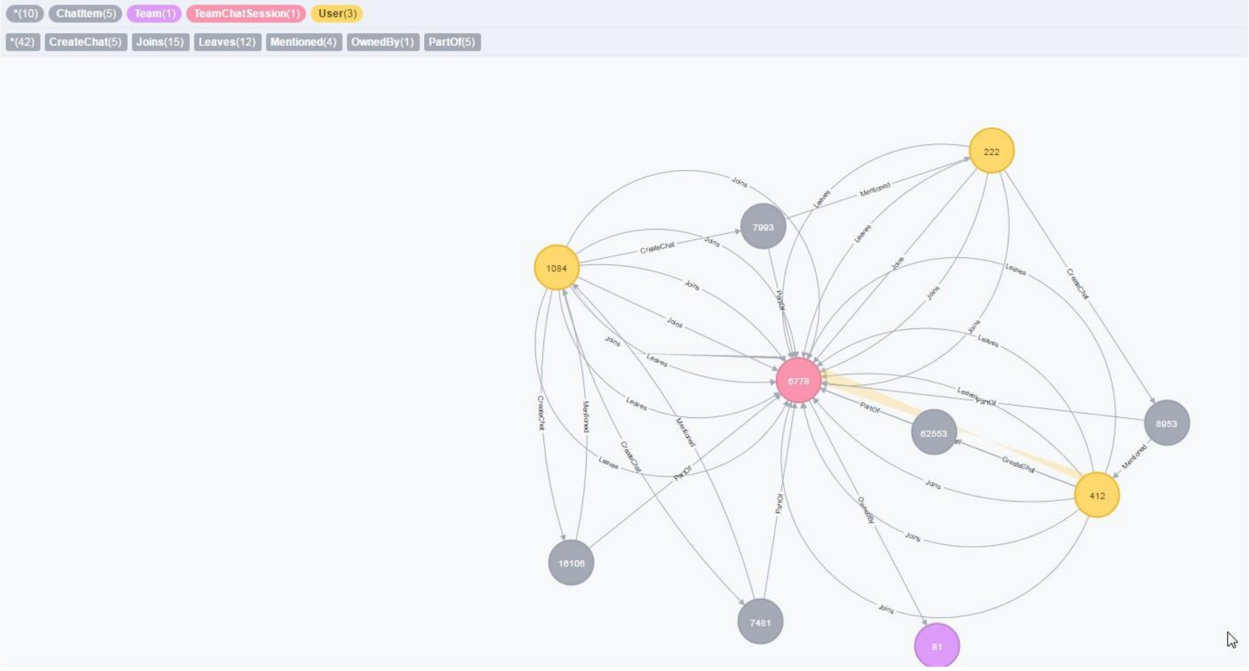
ii) The loading process consists of three main steps:
1/Naming the source of the data through a CSV file
2/Adding the Nodes using the MERGE command
3/Adding the Edges using also the MERGE command.
LOAD Command for the chat_join_team_chat.csv : :

```
LOAD CSV FROM
"file:///c:/Users/jean/Desktop/Big%20Data%20Cours/Course%206%20Capstone%20Project/
w4/chat%20data/chat_join_team_chat.csv" AS row
MERGE (u:User {id: toInt(row[0])})
MERGE (c:TeamChatSession {id: toInt(row[1])})
MERGE (u)-[:Joins{timeStamp: row[2]}]->(c)
```

iii) Screenshots of some part of the graph generated:



```
$ match ()-[r]-() return r limit 50
```



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.

The longest conversation chain based on the “ResponseTo” edge is composed of 10 nodes and 9 edges. The query is composed by a MATCH including all nodes connected by the ResponseTo edge then it’s ordered by the length of all paths founds in descending order.

The participants of the longest conversation chain can be found by using the first query and adding the command “With P” and an another MATCH to count the distinct users who create a chat using only the edges [CreateChat]. They are 5 users. Their ID are the following: 1192, 853, 1514, 1978, 1153.

The query to the longest conversation chain:

```
match p=()-[:ResponseTo*]-() return p order by length(p) desc limit 1.
```

And its participants:

```
match p=()-[:ResponseTo*]-() where length(p)=9

with p

match (u:User)-[c:CreateChat*]-(i:ChatItem)

where i.id in EXTRACT(n IN NODES(p)| n.id)

return u as Users, p, c

limit 25.
```

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

The first query is simple: we match any user who have any number of edges called CreateChat and then ordered them by the numbers of these edges in the descending order to get the chattiest first in the list of results.

The query to get the 10 chattiest users:

```
match p2=(u)-[r:CreateChat]-(i2)

return u as UserS, count(distinct r) as rel

order by rel desc limit 10
```

Chattiest Users

Users	Number of Chats
394	115

2067	111
1087	109

The query to get the 10 chattiest teams :

```
match p=(i)-[r:PartOf]-(:TeamChatSession)-[r2:OwnedBy]-(t:Team)

return t as Team, count(distinct i) as chatitems

order by chatitems desc limit 10
```

Chattiest Teams

Teams	Number of Chats
82	1324
185	1036
112	957

Is there any chattiest user in the chattiest team?

Let's execute the following query:

```
Match (u:User)-[:Joins]-(:TeamChatSession)-[o:OwnedBy]-(t:Team)

where t.id in ([82,185,112,18,194,129,52,136,146,81])

AND u.id in ([394,2067,1087,209,554,1627,516,999,461,668])

return t as Team, u as Users, o limit 1000
```

Finally, just one user, user.id = 999 belongs to the team, team.id = 52.

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.
First we will find the neighbors of the 10th chattiest users thanks to this query:

Ex. Done with user 209:

```
match (u)-[r:InteractsWith]-(u2)
where u.id=209
return u, u2,r
```

To get only one relationship InteractsWith between two nodes, the following query is passed :

```
match (s)-[r:InteractsWith]->(e)
with s,e,type(r) as typ, tail(collect(r)) as coll
foreach(x in coll | delete x)
```

From that we get the number of relationship between the neighbors.
Now as we have both numbers of neighbors and relationships between them we can calculate we cluster coefficient of each.

Most Active Users (based on Cluster Coefficients)

User ID	Coefficient
209 (8 users in the cluster and 54 relationships)	0,9642857142857143
554 (8 users in the cluster and 48 relationships)	0,8571428571428571
1087 (7 users in the cluster and 35 relationships)	0.8333333333333333