Technical Appendix
Catch the Pink Flamingo Analysis

Produced by: TrungUng

Acquiring, Exploring and Preparing the Data

Data Exploration

Data Set Overview

The table below lists each of the files available for analysis with a short description of what is found in each one.

File Name	Description	Fields
ad-clicks.csv	A line is added to this file when a player clicks on an advertisement in the Flamingo app.	• timestamp: when the click occurred. • txId: a unique id (within ad-clicks.log) for the click • userSessionid: the id of the user session for the user who made the click • teamid: the current team id of the user who made the click • userid: the user id of the user who made the click • adId: the id of the ad clicked on • adCategory: the category/type of ad clicked on
buy-clicks.csv	A line is added to this file when a player makes an in-app purchase in the Flamingo app.	• timestamp: when the purchase was made. • txId: a unique id (within buy-clicks.log) for the purchase • userSessionId: the id of the user session for the user who made the purchase • team: the current team id of the user who made the purchase • userId: the user id of the user who made the purchase • buyId: the id of the item purchased • price: the price of the item purchase

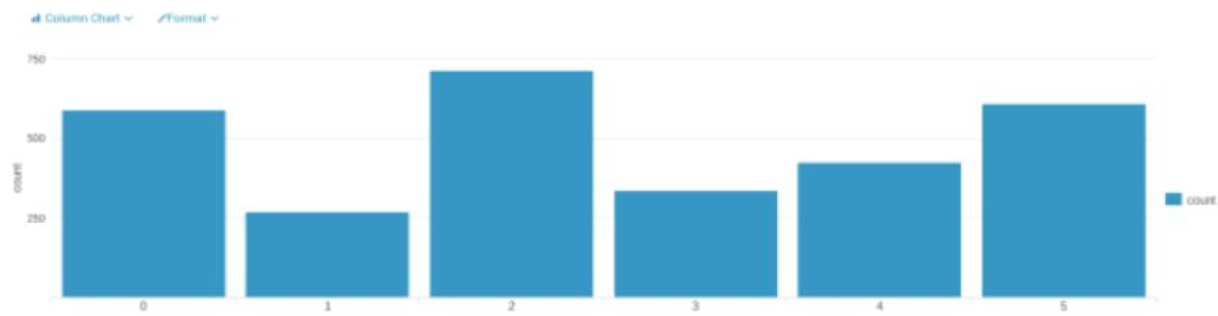
users.csv	This file contains a line for each user playing the game	• timestamp: when user first played the game. • userId: the user id assigned to the user. • nick: the nickname chosen by the user. • twitter: the twitter handle of the user. • dob: the date of birth of the user. • country: the two-letter country code where the user lives.
team.csv	This file contains a line for each team terminated in the game	• teamId: the id of the team • name: the name of the team • teamCreationTime: the timestamp when the team was created • teamEndTime: the timestamp when the last member left the team • strength: a measure of team strength, roughly corresponding to the success of a team • currentLevel: the current level of the team
team-assignments.csv	A line is added to this file each time a user joins a team. A user can be in at most a single team at a time.	• timestamp: when the user joined the team. • team: the id of the team • userId: the id of the user • assignmentId: a unique id for this assignment
level-events.csv	A line is added to this file each time a team starts or finishes a level in the game	• timestamp: when the event occurred. • eventId: a unique id for the event • teamId: the id of the team • teamLevel: the level started or completed • eventType: the type of event, either start or end
user-session.csv	Each line in this file describes a user session, which denotes when a user starts and stops playing the game.	• timestamp: a timestamp denoting when the event occurred. • userSessionId: a unique id for the session. • userId: the current user's ID.

	Additionally, when a team goes to the next level in the game, the session is ended for each user in the team and a new one started.	• teamId: the current user's team. • assignmentId: the team assignment id for the user to the team. • sessionType: whether the event is the start or end of a session. • teamLevel: the level of the team during this session. • platformType: the type of platform of the user during this session
game-clicks.csv	A line is added to this file each time a user performs a click in the game	• timestamp: when the click occurred. • clickId: a unique id for the click. • userId: the id of the user performing the click. • userSessionId: the id of the session of the user when the click is performed. • isHit: denotes if the click was on a flamingo (value is 1) or missed the flamingo (value is 0) • teamId: the id of the team of the user • teamLevel: the current level of the team of the user

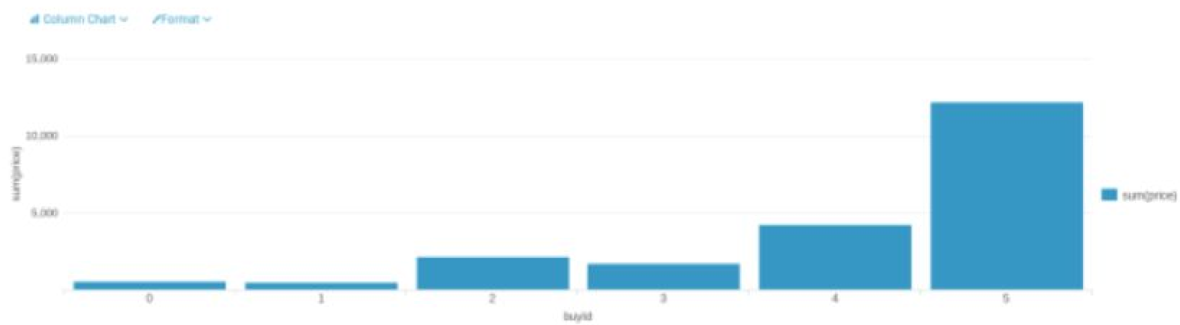
Aggregation

Amount spent buying items	21407
Number of unique items available to be purchased	6

A histogram showing how many times each item is purchased:

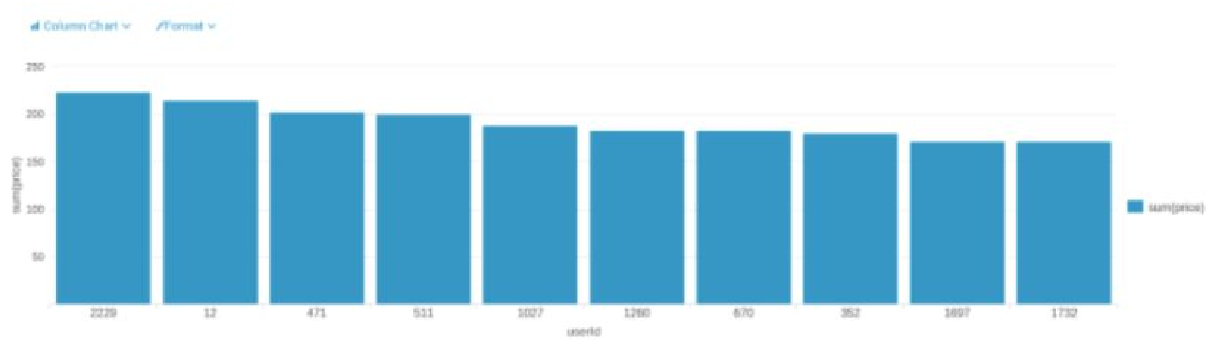


A histogram showing how much money was made from each item:



Filtering

A histogram showing total amount of money spent by the top ten users (ranked by how much money they spent).



The following table shows the user id, platform, and hit-ratio percentage for the top three buying users:

Rank	User Id	Platform	Hit-Ratio (%)
1	2229	iphone	11.60
2	12	iphone	13.07
3	471	iphone	14.50

According to the histogram above, we know the userId of top three users are “2229”, “12” and “471”. In order to check their platform, we can use the file “user-session.csv”. Then with the file “game-clicks.csv”, we can calculate the Hit-Ratio by sum(isHit)/count(isHit) for each user.

Data Classification Analysis

Data Preparation

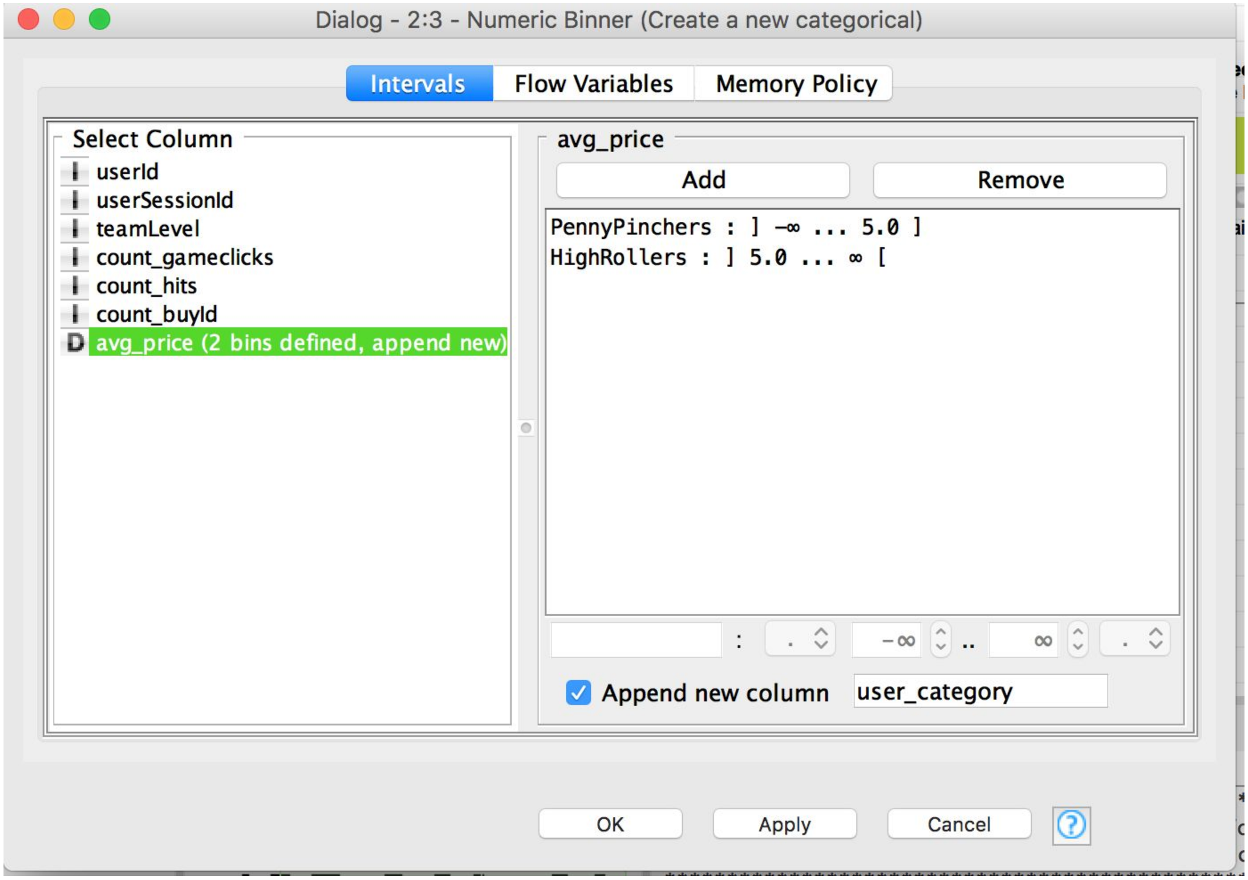
Analysis of combined_data.csv

Sample Selection

Item	Amount
# of Samples	*4619*
# of Samples with Purchases	*1411*

Attribute Creation

A new categorical attribute was created to enable analysis of players as broken into 2 categories (HighRollers and PennyPinchers). A screenshot of the attribute follows:



A new categorical attribute, named "user_category", is created by the Numeric Binner node. As presented in the instruction, we need to define two categories for price which we will use to distinguish between HighRollers(buyers of items that cost more than $5.00) and PennyPinchers (buyers of items that cost $5.00 or less), so as we see in the screenshot above, the user who costs $5.00 or less is defined as "PennyPinchers", the user who costs more than $5.00 is defined as "HighRollers".

The creation of this new categorical attribute was necessary because *it can facilitate the classification of users and contribute to the following steps.*

Attribute Selection

The following attributes were filtered from the dataset for the following reasons:

Attribute	**Rationale for Filtering**

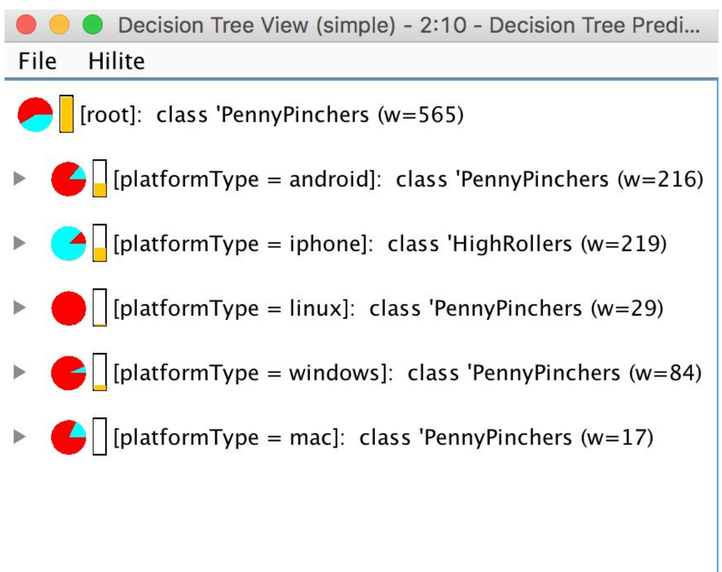
userId	*Since the objective is to predict which user is likely to purchase big-ticket items, and the attribute "userId" has no effect on it, so it's removed.*
userSessionId	*Since the objective is to predict which user is likely to purchase big-ticket items, and the attribute "userSessionId" has no effect on it, so it's removed.*
avg_price	*Since a new attribute "user_category" has been created, which was generated from the attribute "avg_price", so we can remove it.*
userId	*Since the objective is to predict which user is likely to purchase big-ticket items, and the attribute "userId" has no effect on it, so it's removed.*

Data Partitioning and Modeling

The data was partitioned into train and test datasets.
The **training** data set was used to create the decision tree model.
The trained model was then applied to the **test** dataset.
This is important because **train data set is used in creating the decision tree model, the apply the model to the test data set, which is not used to train the mode then we can see the accuracy of the model.**

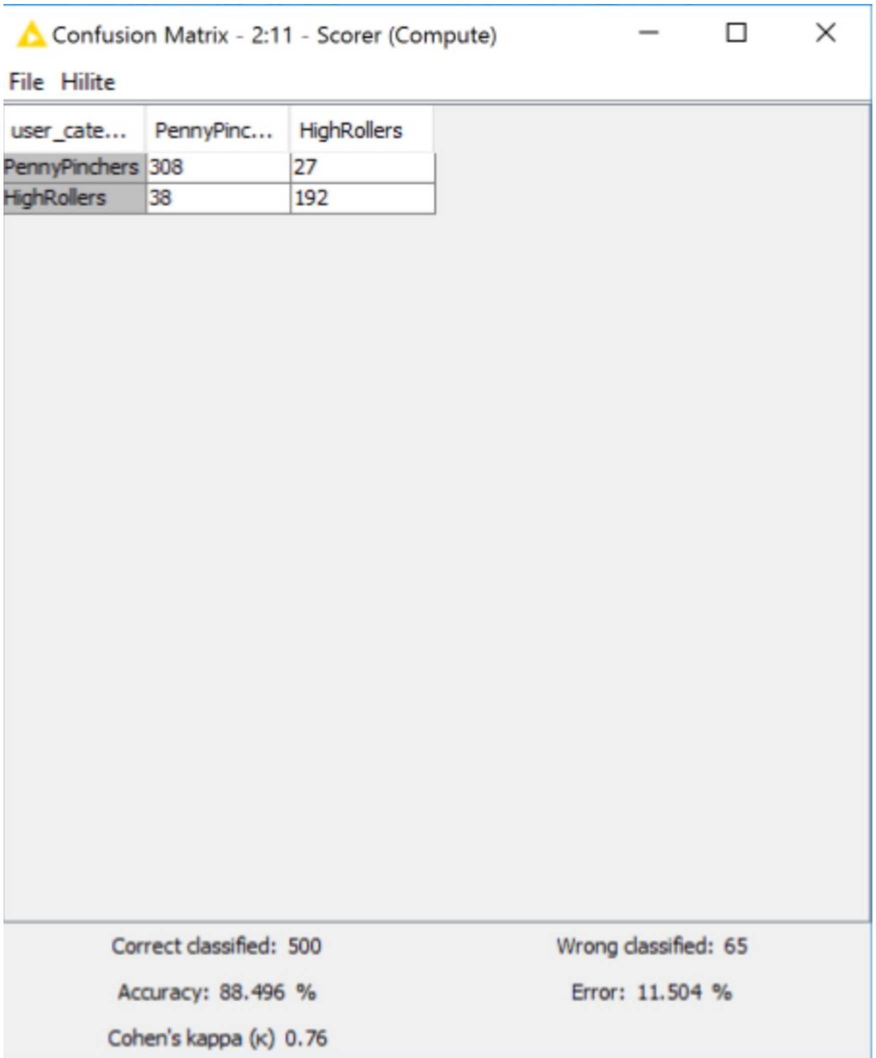
When partitioning the data using sampling, it is important to set the random seed because **it can get the same data partitions every time the node is executed.**

A screenshot of the resulting decision tree can be seen below:



Evaluation

A screenshot of the confusion matrix can be seen below:

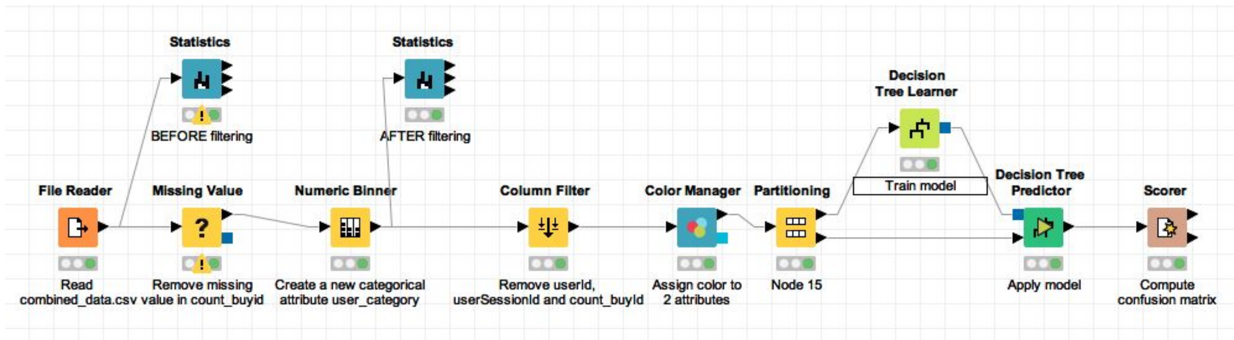


As seen in the screenshot above, the overall accuracy of the model is **88.496%.**

- **"308" & "38"**: according to the model, we predict that **348(308+38)** users are PennyPinchers, but among them **308** users are truly predicted, which means among these **348** users, **308** users are exactly **PennyPinchers**, **38 HighRollers** are incorrectly predicted as PennyPinchers.
- **"192" & "27"**: according to the model, we predict that **219(192+27)** users are **HighRollers**, but among them **192** users are truly predicted, which means among these **219** users, **192** users are exactly **HighRollers**, **27 PennyPinchers** are incorrectly predicted as **HighRollers**.

Analysis Conclusions

The final KNIME workflow is shown below:



What makes a **HighRoller** vs. a **PennyPincher**?

Following the resulting decision tree, it obviously shows that the predicted user_category is different in various platforms, the users on the platform android, linux, windows and mac are almost PennyPincher, however, most users which on the platform iphone are HighRoller

Specific Recommendations to Increase Revenue
1. Offer more products to iPhone users.
2. Offer some promotions to PennyPinchers for attracting their consommation.

Clustering Analysis

Attribute Selection

Attribute	**Rationale for Selection**
amount of ad-clicking per user	according to this attribute, we can capture users' behavior on clicking ad amount of game-clicking per user
amount of game-clicking per user	according to this attribute, we can capture users' behavior on clicking game total price spent by each user
total price spent by each user	total cost of each user can capture preference degree of each user

Training Data Set Creation

The training data set used for this analysis is shown below (first 5 lines):

Create the final training dataset

Our training data set is almost ready. At this stage we can remove the 'userId' from each row, since 'userId' is a computer generated random number assigned to each user. It does not capture any behavioral aspect of a user. One way to drop the 'userId', is to select the other two columns.

In [25]:
```
cluster_df = combined_df[['totalAdClicks', 'totalGameClicks','revenue']]
cluster_df.head(5)
```

Out[25]:

	totalAdClicks	totalGameClicks	revenue
0	44	716	21.0
1	10	380	53.0
2	37	508	80.0
3	19	3107	11.0
4	46	704	215.0

Display the dimensions of the training dataset

Display the dimension of the training data set. To display the dimensions of the training_df, simply add `.shape` as a suffix and hit enter.

In [26]:
```
cluster_df.shape
```

Out[26]: (543, 3)

Dimensions of the training data set (rows x columns) : **543 * 3**
# of clusters created: **3**

Cluster Centers

Cluster #	Cluster Center
1	array([25.12037037, 362.50308642, 35.35802469])
2	array([32.05, 2393.95, 41.2])
3	array([36.47486034, 953.82122905, 46.16201117])

These clusters can be differentiated from each other as follows:

- *The first number (field 1) in each array refers to the number of ads per user click, the second number (field 2) in each array refers to amount of game-clicking per user and the third number (field 3) is the cost on this game of each user.*

- ***Cluster 1*** *is different from the others in that the users' ad-clicks, game-clicks and cost are all less than others, this kind of users can be called "low level spending user".*

- ***Cluster 2*** *is different from the others in that the ad-clicks is not the least, game-clicks is the most but their cost is not the most, this kind of users can be called "neutral user".*

- ***Cluster 3*** *is different from the others in that the users' ad-clicks, game-clicks and cost are all more than others, this kind of users can be called "high level spending user".*

Recommended Actions

Action Recommended	**Rationale for the action**
provide more products to "high level spending user"	since they clicked less but buy more than others, we can provide more products to them for increasing the revenue
provide some fixed pay packages or promotion to users, especially to "low level spending user"	this action can stimulate consumption of users, and since the paying probability of "low level spending user" is low, the promotion can encourage them to purchase

Graph Analytics Analysis

Modeling Chat Data using a Graph Data Model

Graph data model using to illustrate the chatting interaction among users with Chat Data. A user can create a chat session and create chat in the chat session. A user could be mentioned by a chat item and a chat item can response to another chat item. A user can join in an existed team chat session or leave it.

Creation of the Graph Database for Chats

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

1. **Detail for 6 CSV Files**

File Name	**Description**	**Fields**
chat_create_team_chat.csv	userid	the user id assigned to the user

	teamid	the id of the team
	teamChatSessionID	a unique id for the chat session
	timestamp	a timestamp denoting when the chat session created
chat_item_team_chat.csv	userid	the user id assigned to the user
	teamchatsessionid	a unique id for the chat session
	chatitemid	a unique id for the chat item
	timestamp	a timestamp denoting when the chat item created
chat_join_team_chat.csv	userid	the user id assigned to the user
	teamChatSessionID	a unique id for the chat session
	timestamp	a timestamp denoting when the user join in a chat session
chat_leave_team_chat.csv	userid	the user id assigned to the user
	teamchatsessionid	a unique id for the chat session
	timestamp	a timestamp denoting when the user leave a chat session

chat_mention_team_chat.csv	ChatItemId	the id of the ChatItem
	userid	the user id assigned to the user
	timeStamp	a timestamp denoting when the user mentioned by a chat item
chat_respond_team_chat.csv	chatid1	the id of the chat post 1
	chatid2	the id of the chat post 2
	timestamp	a timestamp denoting when the chat post 1 responds to the chat post 2

2. Explain the loading process and include a sample LOAD command

Using Cypher Query Language to load the CSV data into neo4j, each row of script is parsed for refine the nodes, the edges and its timestamp. Let’s consult the following script as an example:

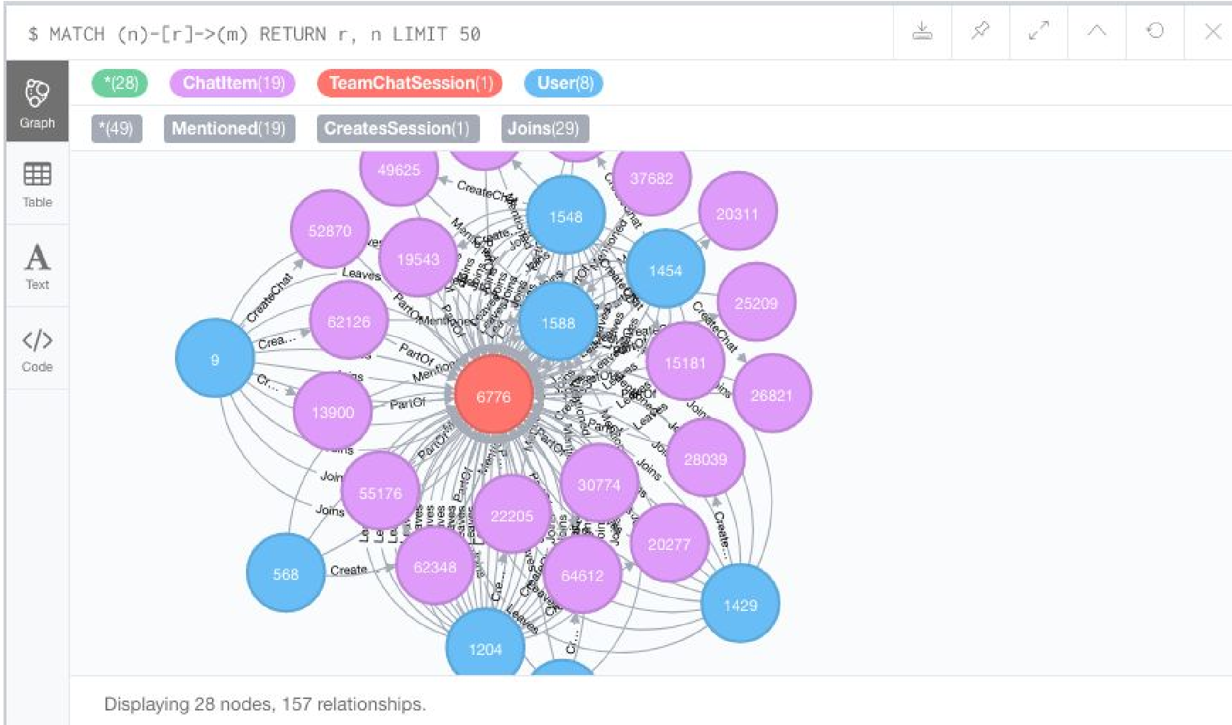
1. *LOAD CSV FROM "file:///chat_item_team_chat.csv" AS row*
2. *MERGE (u:User {id: toInt(row[0])})*
3. *MERGE (c:TeamChatSession {id: toInt(row[1])})*
4. *MERGE (i:ChatItem {id: toInt(row[2])})*
5. *MERGE (u)-[:CreateChat{timeStamp: row[3]}]->(i)*
6. *MERGE (i)-[:PartOf{timeStamp: row[3]}]->(c)*

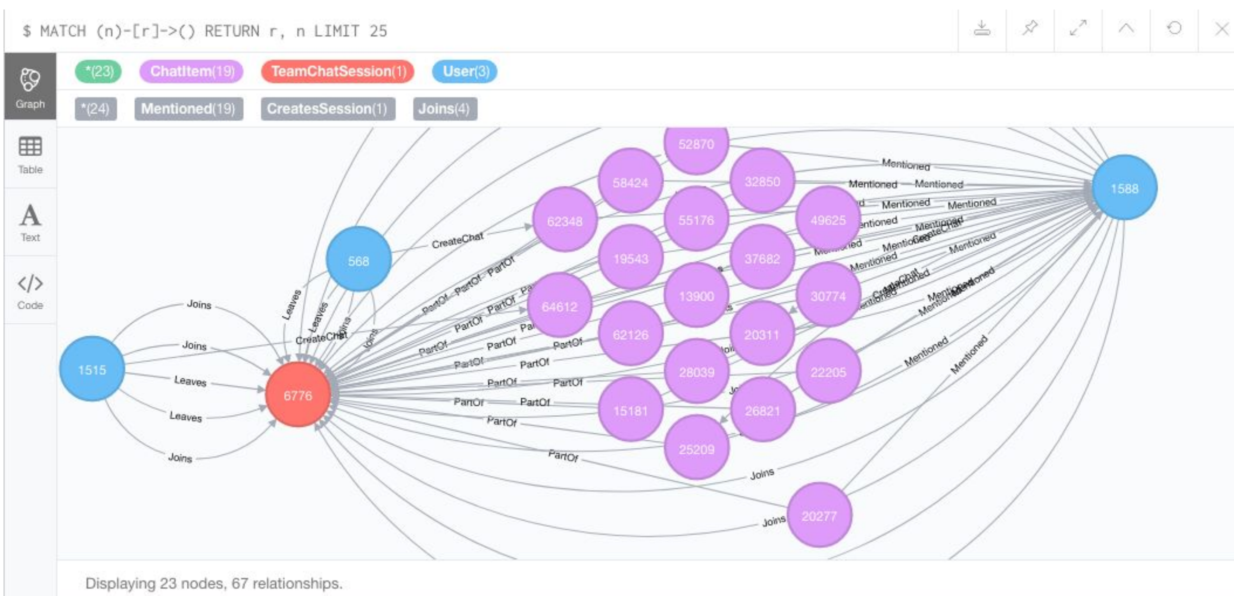
The first line gives the path of the file, this command reads the chat_item_team_chat.csv at a time and create user nodes. The 0_{th} column value is converted to an integer and is used to populate the id attribute. Similarly the other nodes are created.

Line 5, MERGE (u)-[:CreateChat{*timeStamp*: row[**3**]}]->(i) creates an edge labeled “CreateChat” between the User node u and the ChatItem node i. This edge has a property called timeStamp. This property is filled by the content of column 3 of the same row.

Line 6, MERGE (i)-[:PartOf{*timeStamp*: row[**3**]}]->(c) creates an edge labeled “PartOf” between the ChatItem node i and the TeamChatSession node c. This edge has a property called timeStamp. This property is filled by the content of column 3 of the same row.

3. A screenshot of some part of the graph





Finding the longest conversation chain and its participants

Find the longest conversation chain in the chat data using the "ResponseTo" edge label. This question has two parts

a. How many chats are involved in it?

match p = (i1)-[:ResponseTo]->(i2)*

return length(p)

order by length(p) desc limit 1

Result: 9

```
$ match p = (i1)-[:ResponseTo*]->(i2) return length(p) order by length(p) desc limit 1
```

Table

length(p)
9

b. How many users participated in this chain?

match p = (i1)-[:ResponseTo]->(i2)*

where length(p) = 9

with p

match (u)-[:CreateChat]->(i)

where i in nodes(p)

return count(distinct u)

Result: 5

```
$ match p = (i1)-[:ResponseTo*]->(i2) where length(p) = 9 with p match (u)-[:Creat
```

Table

count(distinct u)
5

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

- Match the **CreateChat** edge from User node to **ChatItem** node, then return the **ChatItem** amount per user, and order by the amount in descending order.

  match (u)-[:CreateChat]->(i)*

  return u.id, count(i)

  order by count(i) desc limit 10

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

Table | Text | Code

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

Started streaming 10 records after 745 ms and completed after 745 ms.

Users	Number of Chats
394	115
2067	111
209	109
1087	109

Chattiest Teams

- Match the PartOf edge from ChatItem node to TeamChatSession node, match the OwnedBy edge from TeamChatSession node to Team node, then return the TeamChatSession amount per team, and order by the amount in descending order.

  match (i)-[:PartOf]->(c)-[:OwnedBy*]->(t)*

  return t.id, count(c)

  order by count(c) desc limit 10

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

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

Teams	Number of Chats
82	1324
185	1036
112	957

- **Final result**

  match (u)-[:CreateChat]->(i)-[:PartOf*]->(c)-[:OwnedBy*]->(t)*

  return u.id, t.id, count(c)

  order by count(c) desc limit 10

```
$ match (u)-[:CreateChat*]->(i)-[:PartOf*]->(c)-[:OwnedBy*]->(t) return u.id, t.id, count(c) order b…
```

u.id	t.id	count(c)
394	63	115
2067	7	111
209	7	109
1087	77	109
554	181	107
516	7	105
1627	7	105
999	52	105
461	104	104
668	89	104

User 999, which in the team 52 is part of the top 10 chattiest teams, but other 9 users are not part of the top 10 chattiest teams. This states that most of the chattiest users are not in the 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.

Most Active Users (based on Cluster Coefficients)

User ID	Coefficient
209	0.95238095238095
554	0.9047619047619048
1087	0.8

Recommended Actions

- Offer more products to iPhone users.
- Offer some promotions to PennyPinchers for attracting their consommation.
- Provide more products to “high level spending user”
    - Since they clicked less but buy more than others, we can provide more products to them for increasing the revenue
- Provide some fixed pay packages or promotion to users, especially to “low level spending user”
    - This action can stimulate consumption of users, and since the paying probability of “low level spending user” is low, the promotion can encourage them to purchase