

Graph Database Project Report

s188026 Yifei Liu

1. Assumption & Business domain


This database records the information of Word Championship (Worlds 2022) of LOL (League of Legends).

It contains all teams (24 teams) attend this tournament, 5-6 professional player in each team, and matched they performed.

It should record important data for each match. Because of time limit and there are hundreds of games totally, I only record 11 games in Quarter Finals (last weekend), 2 game in Play-in, 1 game in Groups.


2. Description

2.1 Team



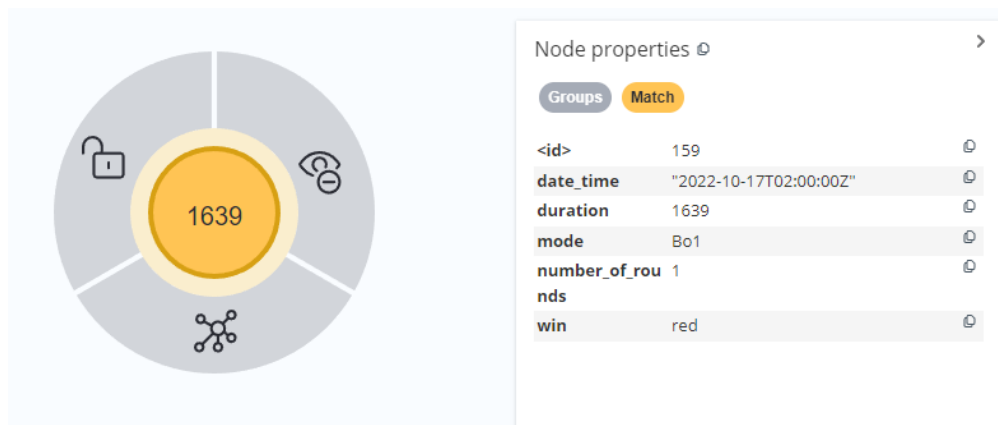
Node properties	
Team	
<id>	7
isSurvive	false
league	LEC
name	G2
region	Europe

2.2 Player

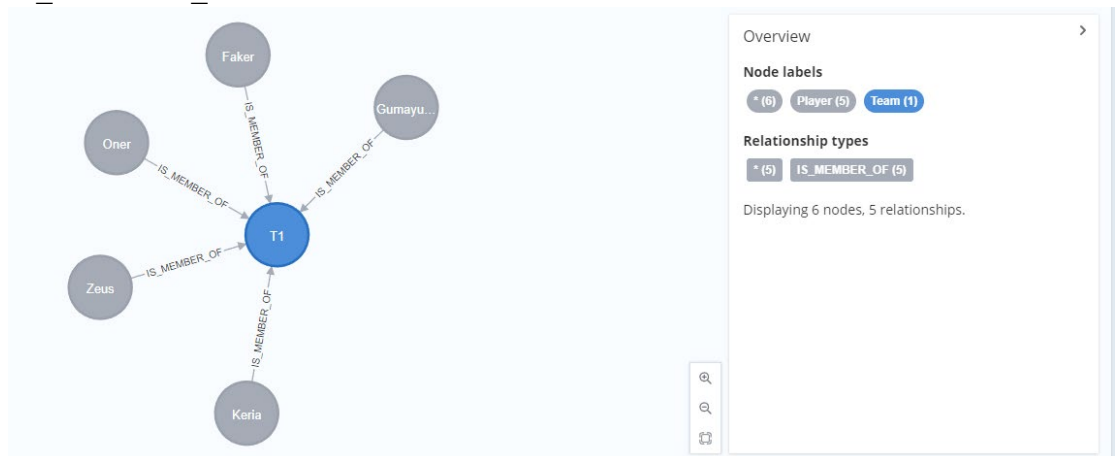


Node properties	
Player	
<id>	76
name	Jankos
role	Jungle

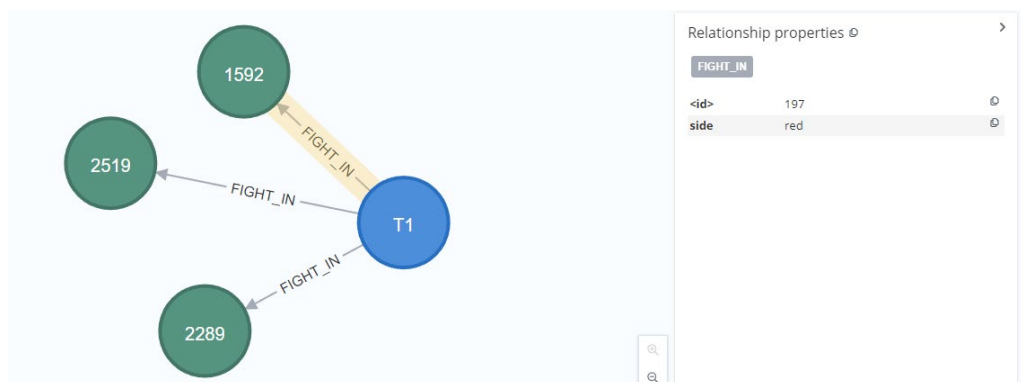
2.3 Match



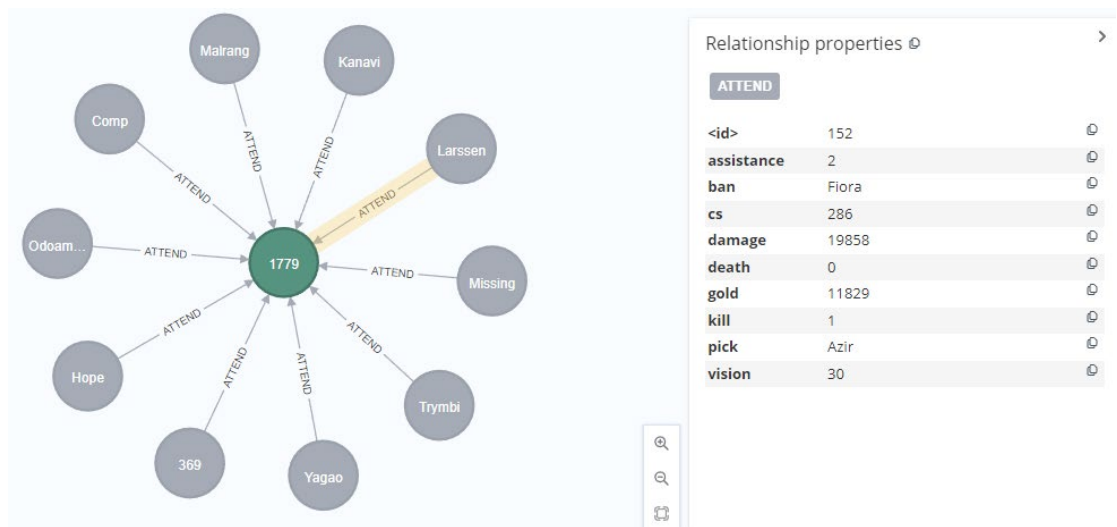
2.4 IS_MEMBER_OF



2.5 FIGHT_IN

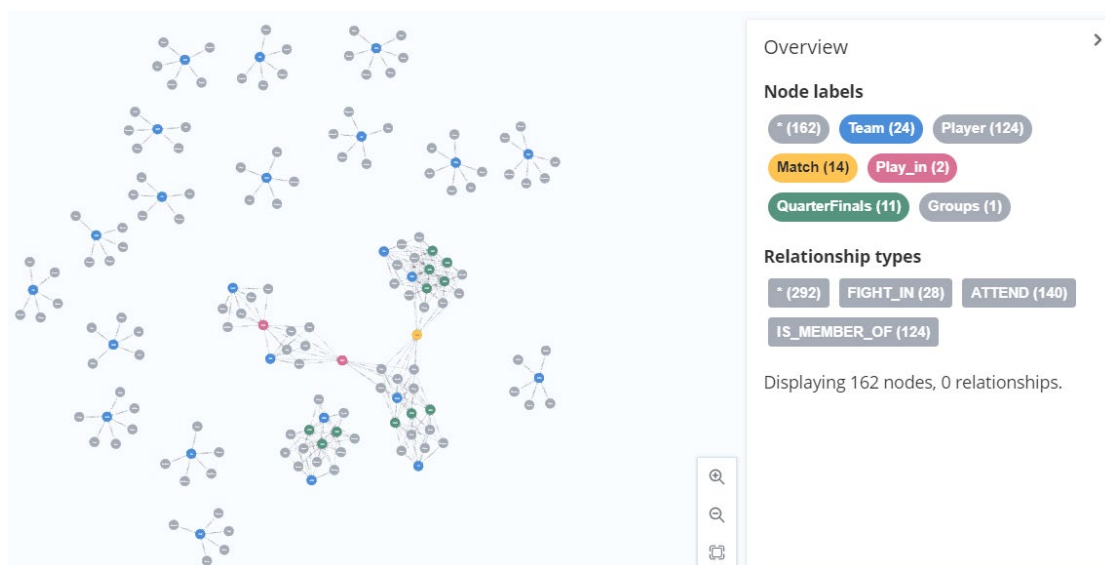
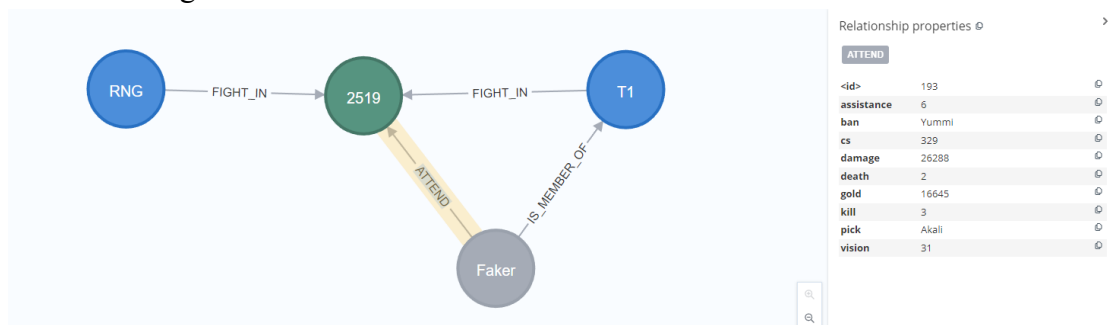


2.6 ATTEND



Information:

1. KDA (kill, death, assistance)
2. Ban & Pick (characters be banned and picked)
3. CS (minion killed)
4. Gold
5. Vision score
6. Damage



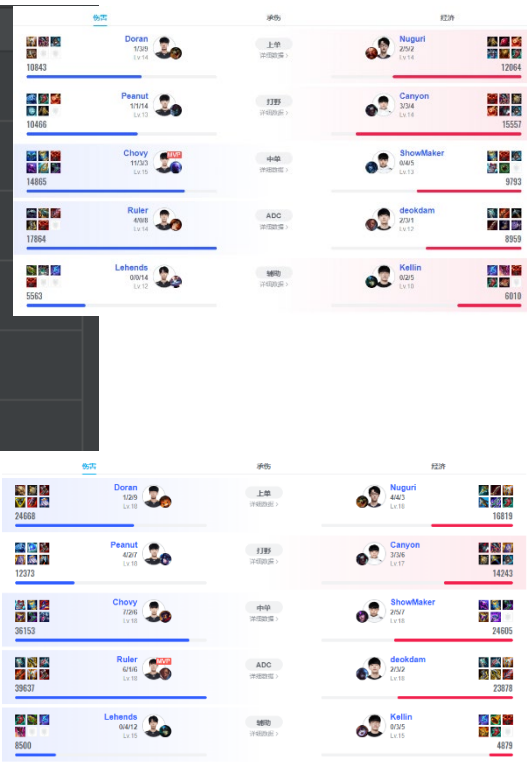
3. Competency queries

1. On Saturday night, Team GEN and Team DK showed a really wonderful performance in 5 games.

In each game, which lane of each team contribute the highest damage for their team?

```
MATCH (p:Player) - [:IS_MEMBER_OF] -> (t:Team) - [:FIGHT_IN] -> (m:Match {date_time: datetime("2022-10-22T23:00")}) <- [a:ATTEND] - (p)
WITH p, t, m, a
ORDER BY m.number_of_round, a.damage DESC
RETURN m.number_of_rounds, t.name, collect(p.name)[..1] AS topOne ORDER BY m.number_of_rounds, t.name;
```

m.number_of_rounds	t.name	topOne
1	"DK"	topOne: list 0: "Canyon"
1	"GEN"	topOne: list 0: "Ruler"
2	"DK"	topOne: list 0: "ShowMaker"
2	"GEN"	topOne: list 0: "Ruler"
3	"DK"	topOne: list 0: "ShowMaker"
3	"GEN"	topOne: list 0: "Chovy"
4	"DK"	topOne: list 0: "ShowMaker"
4	"GEN"	topOne: list 0: "Chovy"
5	"DK"	topOne: list 0: "ShowMaker"
5	"GEN"	topOne: list 0: "Ruler"



Comparing to the results (round 1 and 5) in the Internet, it is correct.

2. Next week, GEN will fight with DRX. Assume you are coach of DRX, try to find the champion which is picked most frequently by GEN?

```
MATCH (p:Player) - [:IS_MEMBER_OF] -> (t:Team {name: 'GEN'}) - [:FIGHT_IN] -> (m:Match) <- [a:ATTEND] - (p)
RETURN a.pick, count(a) AS num ORDER BY num DESC;
```

a.pick	num
"Sejuani"	3
"Renekton"	2
"Nami"	2
"Lucian"	2
"Gnar"	2
"Azir"	2
"Maokai"	2
"Yummi"	2
"MissFortune"	2
"Renata Glasc"	1
"Aphelios"	1
"Sylas"	1
"Trundle"	1
"Viktor"	1
"Lulu"	1
"Jinx"	1
"Poppy"	1
"Yone"	1
"Camille"	1
"Ryze"	1

Before next match, you want to analyze which champions are stronger in this patch. So,

3. you want to know which champions were usually be banned?

```

MATCH (m:Match) <- [a:ATTEND] - ()
RETURN a.ban, count(a) AS num ORDER BY num DESC;

```

a.ban	num
"Caitlyn"	12
"Aatrox"	12
"Yummi"	10
"Graves"	10
"Sylas"	9
"Fiora"	7
"Sejuani"	6
"Lissandra"	6
"LeBlanc"	5
"Viktor"	5
"Vi"	5
"Azir"	4
"Camille"	4
"Aphelios"	3
"Viego"	3
"Renekton"	3
"Kalista"	2

4. you want to know which champions have higher win rate (The champion should be picked more than once)

```

MATCH () <- [a:ATTEND] - ()
WITH a.pick AS aPick, count(a) AS num
WHERE num > 1
MATCH (p:Player) - [:IS_MEMBER_OF] -> (t:Team) - [:FIGHT_IN] -> (m:Match) <- [a] - (p)
WHERE f.side = m.win AND a.pick = aPick
RETURN a.pick, round(toFloatOrNull(count(a))/toFloatOrNull(num),2) AS win_rate ORDER BY win_rate DESC;

```

a.pick	win_rate
"Taliyah"	1.0
"Viktor"	1.0
"Camille"	1.0
"Renata Glasc"	1.0
"Yone"	1.0
"Yummi"	1.0
"Viego"	0.83
"Graves"	0.75
"Gnar"	0.67
"Sejuani"	0.67
"Aphelios"	0.63
"Lulu"	0.6
"Sylas"	0.6
"MissFortune"	0.5
"Ornn"	0.5
"Renekton"	0.5
"LeBlanc"	0.5
"Akali"	0.5

- Sort the players by DPM (damage per minute) in all games.

```

// 5. Sort the players by DPM (damage per minute) in all games.
MATCH (p:Player) - [:IS_MEMBER_OF] -> (t:Team) - [:FIGHT_IN] -> (m:Match) <- [a:ATTEND] - (p)
RETURN p.name, p.role, t.name, round((sum(a.damage) / toFloatOrNull(sum(m.duration)) * 60), 2) as DPM ORDER BY DPM DESC;

```

p.name	p.role	t.name	DPM
"Larssen"	"Mid"	"RGE"	690.55
"369"	"Top"	"JDG"	688.83
"Zeus"	"Top"	"T1"	682.21
"Faker"	"Mid"	"T1"	665.86
"Ruler"	"ADC"	"GEN"	641.66
"Chovy"	"Mid"	"GEN"	635.44
"Nisqy"	"Mid"	"MAD"	615.22
"ShowMaker"	"Mid"	"DK"	585.03
"Hope"	"ADC"	"JDG"	570.37
"Grell"	"Jungle"	"ISG"	567.89
"Gumayusi"	"ADC"	"T1"	565.8
"xiaohu"	"Mid"	"RNG"	521.0
"Gala"	"ADC"	"RNG"	517.56
"Breath"	"Top"	"RNG"	514.4
"Armut"	"Top"	"MAD"	513.08
"deokdam"	"ADC"	"DK"	483.31
"Yagao"	"Mid"	"JDG"	461.53
"Kanavi"	"Jungle"	"JDG"	444.14

6. Is there big gape of vision score between teams in Quarter Finals and teams which are eliminated in Play-in? Choose MAD and DK to compare here.

```
MATCH (p:Player) - [:IS_MEMBER_OF] -> (t:Team) - [:FIGHT_IN] -> (m:Match) <- [a:ATTEND] - (p)
WHERE t.name = "T1" OR t.name = "MAD"
RETURN t.name AS team, round(sum(a.vision)/toFloatOrNull(sum(m.duration)) * 60, 3) AS VSPM;
```

team	VSPM
"T1"	1.526
"MAD"	1.396

7. Who is the most important role to control vision in a team to except for support?
Mid or jungle?

```
MATCH (m:Match) <- [a:ATTEND] - (p)
RETURN p.role AS role, round(avg(a.vision), 2) AS avg_vision_score ORDER BY avg_vision_score DESC LIMIT 3;
```

role	avg_vision_score
"Sup"	79.68
"Jungle"	60.25
"ADC"	39.54

8. According to KDA ((kill + assistance) / death), who is the best player in s12 world championship?

```
MATCH (p:Player) - [:IS_MEMBER_OF] -> (t:Team) - [:FIGHT_IN] -> (m:Match) <- [a:ATTEND] - (p)
RETURN p.name, (sum(a.kill) + sum(a.assistance))/sum(a.death) as KDA ORDER BY KDA DESC;
```

p.name	KDA
"Gumayusi"	19
"UNFORGIVEN"	14
"Keria"	9
"Missing"	9
"Elyoya"	8
"Ruler"	7
"Yagao"	7
"Larssen"	6
"Armut"	6
"Oner"	5
"Chovy"	4
"Faker"	4
"Kellin"	4
"Nisqy"	4
"Hope"	4
"Kanavi"	4
"369"	4
"Gala"	3

9. Sort the teams by W-L (win-lose).

```

match (t:Team)
optional MATCH (t)-[f1:FIGHT_IN]->(m1:Match)
  where f1.side = m1.win
with t, count(m1) as win
with t, collect(win) as cwin
optional MATCH (t)-[f:FIGHT_IN]->(m:Match)
  where f.side <> m.win
unwind cwin as win
return t.name, win, count(m) as lose ORDER BY win DESC, lose ASC;

```

t.name	win	lose
"GEN"	4	2
"T1"	3	0
"JDG"	3	0
"DK"	2	3
"MAD"	1	0
"RNG"	1	4
"TES"	0	0
"EDG"	0	0
"DRX"	0	0
"G2"	0	0
"C9"	0	0
"100T"	0	0
"FNC"	0	0
"LLL"	0	0
"CFO"	0	0
"CHF"	0	0
"EG"	0	0
"SGR"	0	0

10. showed the attribute of player in radar charts.

attributes:

KDA ((Kill + Assistance)/Death)

DPM (Damage Per Minute)

GPM (Gold Per Minute)

KP% ((Personal Kill + Personal Assistance)/Team Kill)

DMG% (Personal Damage/Team Damage)

Damage Gold Conversion Rate (DMG%/GOLD% * 100%)


```
# name = "Gala"
name = "Gumayusi"
role = "ADC"

graph = Graph('bolt://localhost:1103', auth=("neo4j", ""))

query = "MATCH (p:Player) - [:IS_MEMBER_OF] -> (t:Team) - [:FIGHT_IN] -> (m:Match) <- [a:ATTEND] - (p) " \
        "WITH t, sum(a.kill) as team_ka, sum(a.damage) as team_dmg, sum(a.gold) as team_gold " \
        "MATCH (p:Player) - [:IS_MEMBER_OF] -> (t) - [:FIGHT_IN] -> (m) <- [a:ATTEND] - (p) " \
        "WHERE p.role = \"\" + role + \"\" " \
        "WITH " \
        "(sum(a.kill) + sum(a.assistance))/sum(a.death) as KDA, " \
        "sum(a.damage) / toFloatOrNull(sum(m.duration)) * 60 as DPM, " \
        "sum(a.gold) / toFloatOrNull(sum(m.duration)) * 60 as GPM, " \
        "sum(a.kill + a.assistance)/toFloatOrNull(team_ka) as KP_Percent, " \
        "sum(a.damage)/toFloatOrNull(team_dmg) as DMG_Percent, " \
        "(sum(a.damage)/toFloatOrNull(team_dmg))/(sum(a.gold)/ toFloatOrNull(team_gold)) as D6CR " \
        "RETURN " \
        "max(KDA) as KDA, max(DPM) as DPM, max(GPM) as GPM, max(KP_Percent) as KP_Percent, max(DMG_Percent) as DMG_Percent, max(D6CR) as D6CR;"

max_results = graph.run(query).data()
```

Firstly, Link to [get the maximum of each attribute in this role \(e.g. ADC here\).](#)

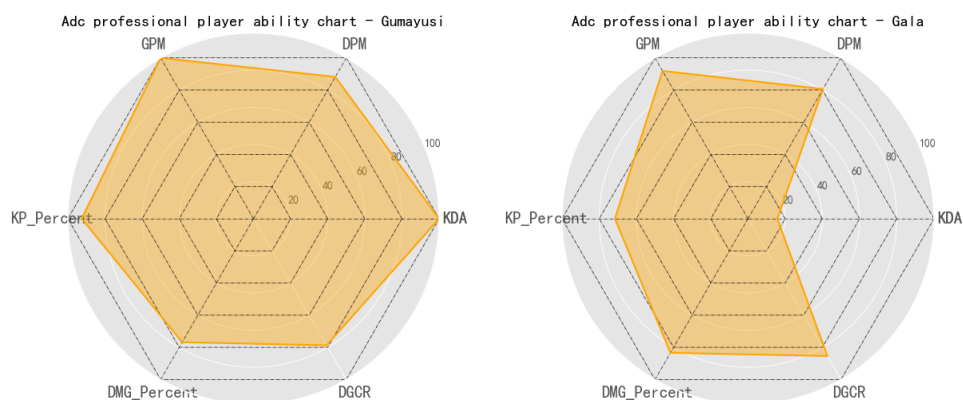
```
coefficient = [100 / max_results[0]["KDA"], 100 / max_results[0]["DPM"], 100 / max_results[0]["GPM"],
               100 / max_results[0]["KP_Percent"], 100 / max_results[0]["DMG_Percent"], 100 / max_results[0]["D6CR"]]
```

Then calculate the coefficient which can help us convert the range of score to 100 points.

```
query = "MATCH (p:Player) - [:IS_MEMBER_OF] -> (t:Team) - [:FIGHT_IN] -> (m:Match) <- [a:ATTEND] - (p) " \
        "WITH t, sum(a.kill) as team_ka, sum(a.damage) as team_dmg, sum(a.gold) as team_gold " \
        "MATCH (p:Player) - [:IS_MEMBER_OF] -> (t) - [:FIGHT_IN] -> (m) <- [a:ATTEND] - (p) " \
        "WHERE p.name = \"\" + name + \"\" " \
        "RETURN " \
        "(sum(a.kill) + sum(a.assistance))/sum(a.death) as KDA, " \
        "sum(a.damage) / toFloatOrNull(sum(m.duration)) * 60 as DPM, " \
        "sum(a.gold) / toFloatOrNull(sum(m.duration)) * 60 as GPM, " \
        "sum(a.kill + a.assistance)/toFloatOrNull(team_ka) as KP_Percent, " \
        "sum(a.damage)/toFloatOrNull(team_dmg) as DMG_Percent, " \
        "(sum(a.damage)/toFloatOrNull(team_dmg))/(sum(a.gold)/ toFloatOrNull(team_gold)) as D6CR;"

# print(graph.run(query).data())
results = graph.run(query).data()
```

Then calculate the attributes of one player, and use result of query to draw the radar chart.



11. Graph analysis

1. Community detection algorithms

Use projection to get a new graph and use **Labeled propagation algorithm** to divide them into different communities.

```
CALL gds.graph.project(
  'lbl',
  ['Team', 'Player'],
  'IS_MEMBER_OF'
)

CALL gds.labelPropagation.stream('lbl')
YIELD nodeId, communityId AS Community
RETURN gds.util.asNode(nodeId).name AS Name, Community
ORDER BY Community, Name
```

Name	Community
"JackeyLove"	0
"Knight"	0
"Mark"	0
"TES"	0
"Tian"	0
"Wayward"	0
"EDG"	1
"Flandre"	1
"JieJie"	1
"Meiko"	1
"Scout"	1
"Viper"	1
"Breath"	2
"Gala"	2
"Ming"	2
"RNG"	2
"Wei"	2
"xiaohu"	2

It is understandable that each team and their team members are in the same community.

2. Similarity algorithms

Use **Jaccard similarity** to get the most similar node in all Players to get his team members.

```

MATCH (p1:Player {name: 'Gumayusi'})-[:IS_MEMBER_OF]->(m:Team)
WITH p1, collect(id(m)) AS p1team
MATCH (p2:Player)-[:IS_MEMBER_OF]->(m2:Team) WHERE p1 <> p2
WITH p1, p1team, p2, collect(id(m2)) AS p2team
RETURN p1.name AS from,
       p2.name AS to,
       gds.similarity.jaccard(p1team, p2team) AS similarity
ORDER BY similarity DESC

```

from	to	similarity
"Gumayusi"	"Keria"	1.0
"Gumayusi"	"Faker"	1.0
"Gumayusi"	"Oner"	1.0
"Gumayusi"	"Zeus"	1.0
"Gumayusi"	"Mark"	0.0
"Gumayusi"	"JackeyLove"	0.0
"Gumayusi"	"Knight"	0.0
"Gumayusi"	"Tian"	0.0
"Gumayusi"	"Wayward"	0.0
"Gumayusi"	"Meiko"	0.0
"Gumayusi"	"Viper"	0.0
"Gumayusi"	"Scout"	0.0
"Gumayusi"	"JieJie"	0.0
"Gumayusi"	"Flandre"	0.0
"Gumayusi"	"Ming"	0.0
"Gumayusi"	"Gala"	0.0
"Gumayusi"	"xiaohu"	0.0
"Gumayusi"	"Wei"	0.0
"Gumayusi"	"Breath"	0.0

It is also obviously that their team member will have the similarity equal to 1.

3. Centrality algorithms

Use **Degree centrality** to find which team will have more matches than others.

```

▶ CALL gds.graph.project(
    'lbl2',
    ['Team', 'Player', 'Match'],
    ['IS_MEMBER_OF', 'FIGHT_IN']
)

▶ CALL gds.degree.stream('lbl2')
YIELD nodeId, score
WHERE EXISTS(gds.util.asNode(nodeId).league)
RETURN gds.util.asNode(nodeId).name AS name, score AS followers
ORDER BY followers DESC, name DESC

```

Use Exists attribute {league} to make sure return team nodes and sort the result as followers.

name	followers
"GEN"	6.0
"RNG"	5.0
"DK"	5.0
"T1"	3.0
"RGE"	3.0
"JDG"	3.0
"ISG"	2.0
"MAD"	1.0
"TES"	0.0
"SGB"	0.0
"LLL"	0.0
"IW"	0.0
"GAM"	0.0
"G2"	0.0
"FNC"	0.0
"EG"	0.0
"EDG"	0.0
"DRX"	0.0
"FCM"	0.0