ZAGREB Graph Data

Game of Graphs

Graph analytics on a GoT dataset


THIS
PRESENTATION IS
DARK AND FULL
OF SPOILERS

ZAGREB Graph Data

Memgraph and GQLAlchemy

How to connect to Memgraph?

Memgraph Platform

Memgraph Platform contains:

- MemgraphDB - the database that holds the data
- Memgraph Lab - visual user interface for running queries and visualizing graph data
- mgconsole - command-line interface for running queries
- MAGE - graph algorithms and modules library

GQLAlchemy

- An **open-source Python** library
- **Object Graph Mapper** (OGM) - a link between graph database objects and Python objects
- It offers **query builder**, **streams and triggers management**, **table data importer** from different sources, **Memgraph instance management**, and **on-disk storage**

Run Memgraph Platform

```
katelatte@Katarinas-MBP ~ % docker run -it -p 7687:7687 -p 3000:3000 memgraph/memgraph-platform
Memgraph Lab is running at localhost:3000

mgconsole 1.1
Connected to 'memgraph://127.0.0.1:7687'
Type :help for shell usage
Quit the shell by typing Ctrl-D(eof) or :quit
memgraph> ›
```

- Port 7687 is for the communication with Memgraph via **Bolt protocol**
- Port 3000 is for **Memgraph Lab**
- Once the container has successfully started, **mgconsole** will open

Connect to Memgraph with GQLAlchemy

```
from gqlalchemy import Memgraph
memgraph = Memgraph("127.0.0.1", 7687)
```

Test the connection to Memgraph

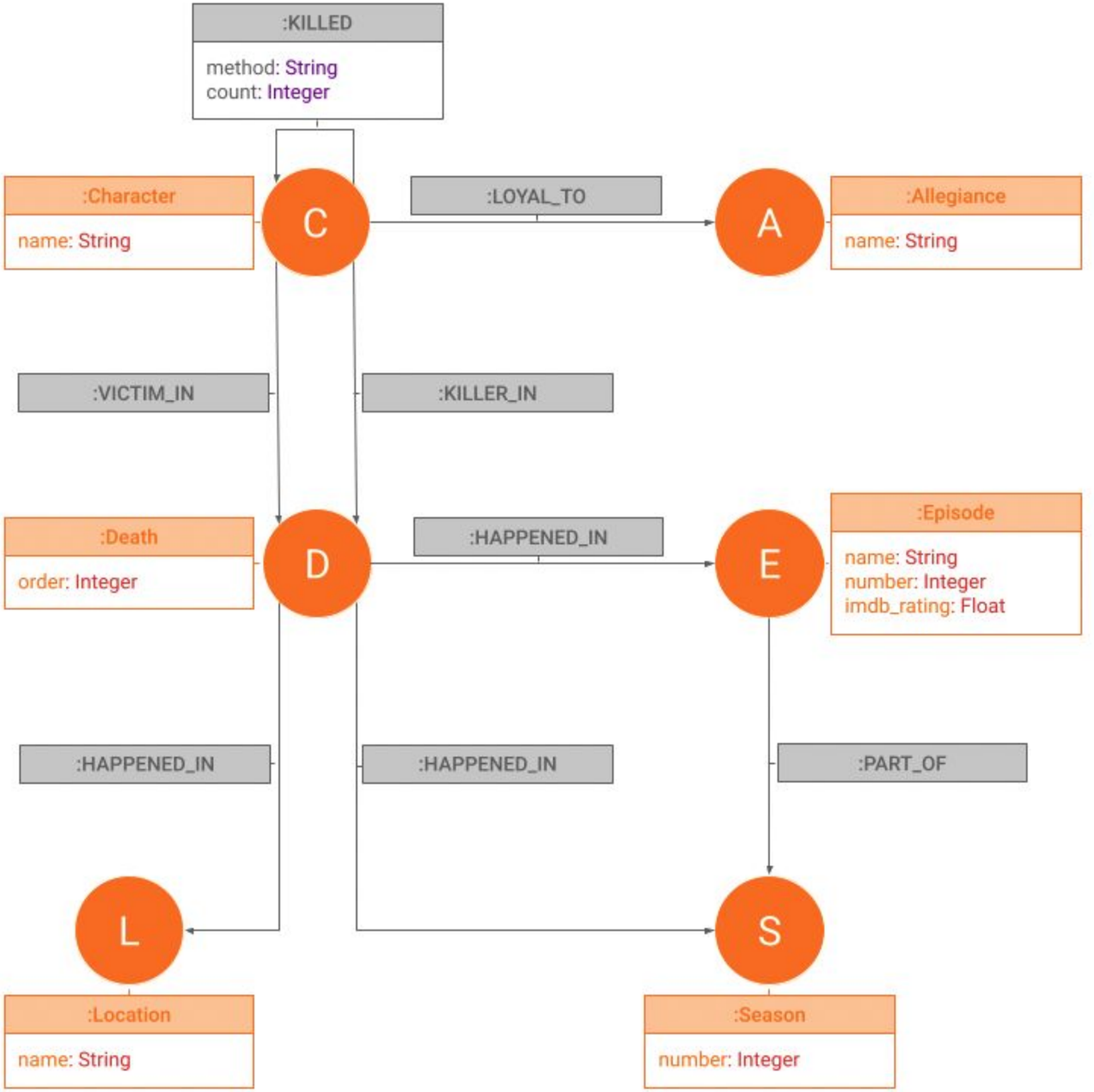
```
results = memgraph.execute_and_fetch(
    """
    MATCH (n) RETURN count(n) AS number_of_nodes ;
    """
)
print(next(results))
```

```
{'number_of_nodes': 0}
```

```
from gqlalchemy import match
results = match()
        .node(variable="n")
        .return_({"count(n)": "number_of_nodes"})
        .execute()
```

Game of Thrones dataset

What does the dataset look like?

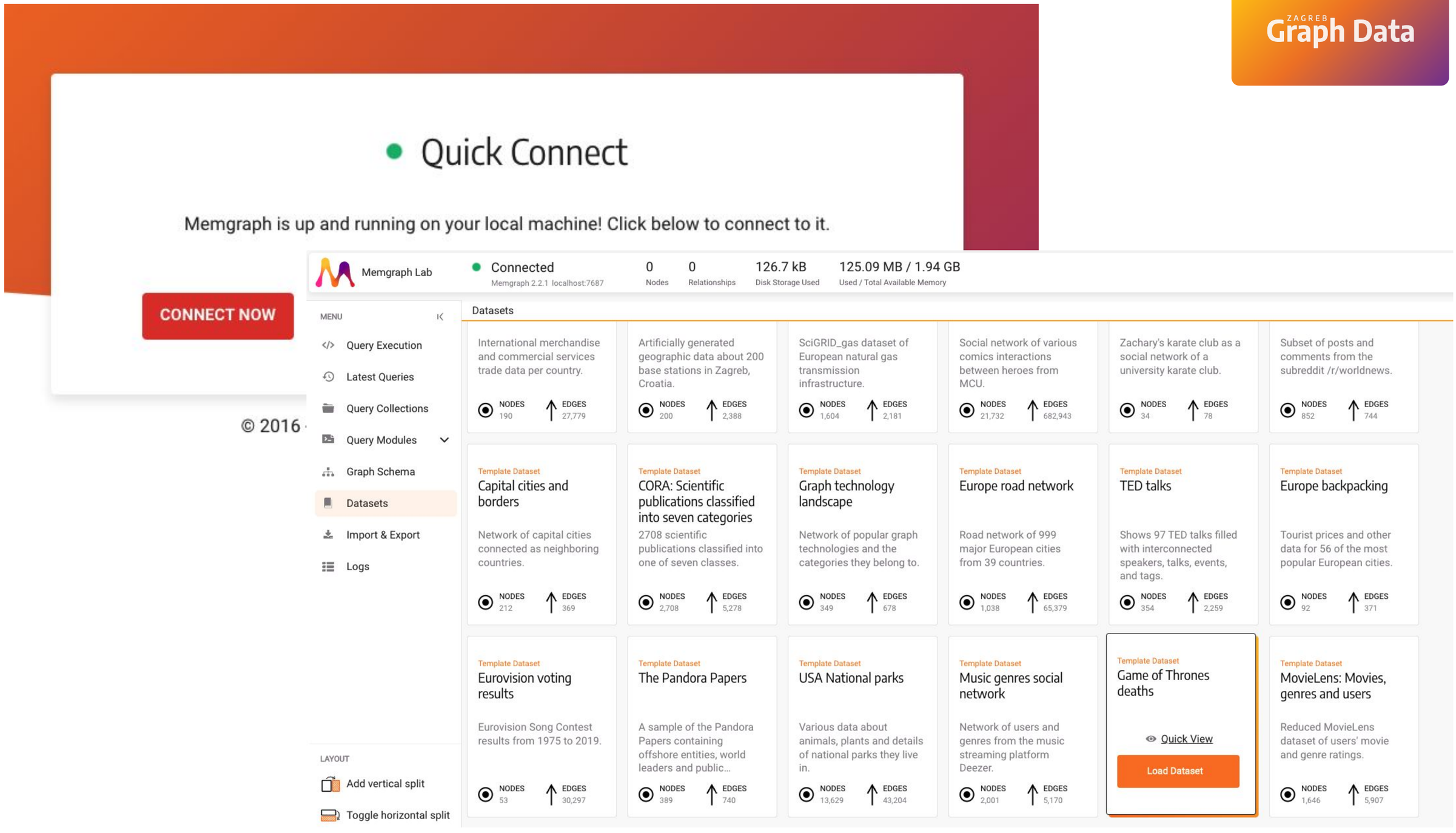

:KILLED
method: String
count: Integer
:Character
name: String
C
:LOYAL_TO
A
:Allegiance
name: String
:VICTIM_IN
:KILLER_IN
:Death
order: Integer
D
:HAPPENED_IN
E
:Episode
name: String
number: Integer
imdb_rating: Float
:HAPPENED_IN
:HAPPENED_IN
:PART_OF
L
S
:Location
name: String
:Season
number: Integer

ZAGREB Graph Data

Memgraph Lab

What is the simplest way to load the GoT dataset?

ZAGREB Graph Data



Connected
Memgraph 2.2.1 localhost:7687

2,677	11,967	2.57 MB	128.34 MB / 1.94 GB
Nodes	Relationships	Disk Storage Used	Used / Total Available Memory

Query execution

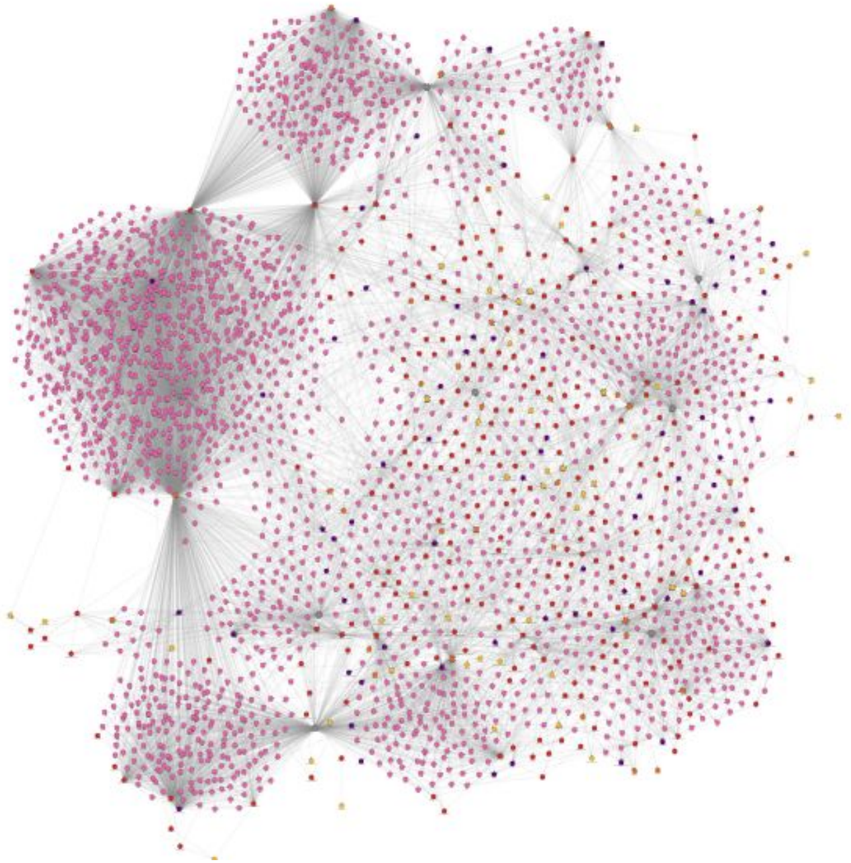
Cypher Editor | Graph Style Editor

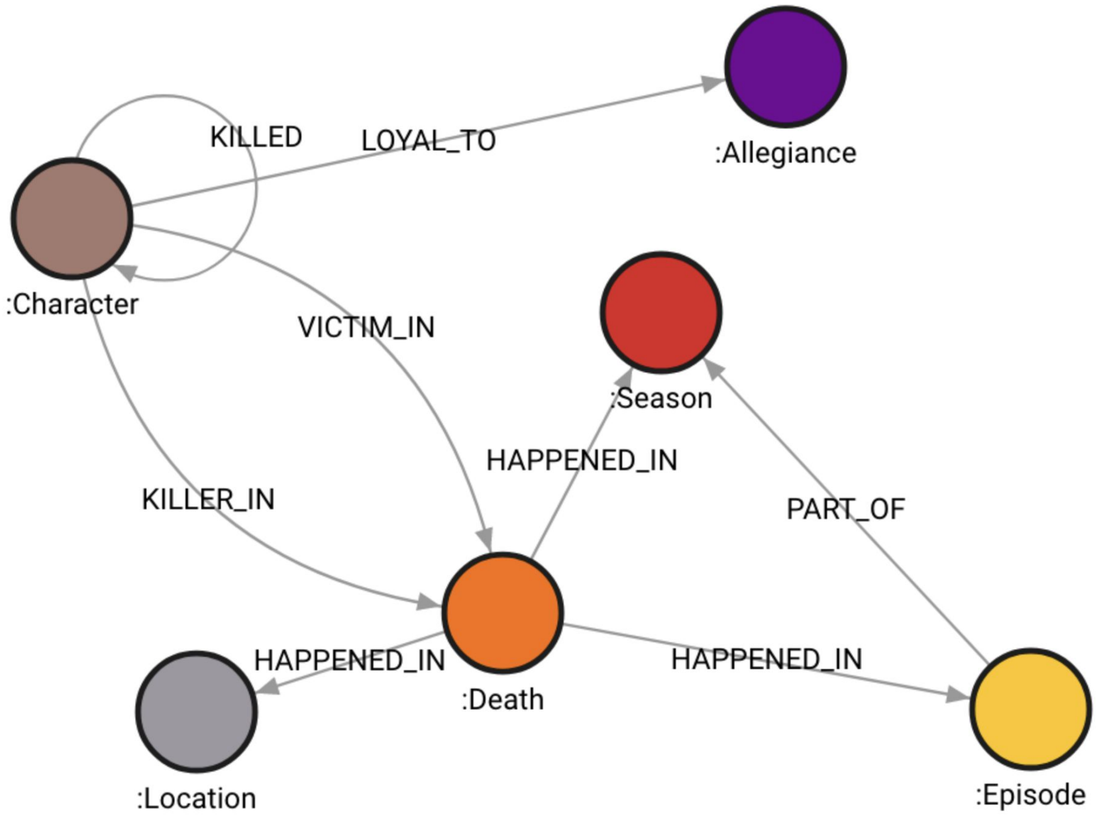
```
1  MATCH (u)-[r]->(m)
2  RETURN u, r, m;
```

Query Successful

31.41 ms	11,967	2,677	11,967
Query Time	Rows	Nodes	Relationships

Data results | Graph results






```python
from gqlalchemy import Node, Relationship

class Character(Node):
    name: str

class Allegiance(Node):
    name: str

class Death(Node):
    order: int

class Episode(Node):
    name: str
    number: int
    imdb_rating: float

class Season(Node):
    number: int

class Location(Node):
    name: str

class Killed(Relationship, type="KILLED"):
    method: str
    count: int

class LoyalTo(Relationship, type="LOYAL_TO"):
    pass

class VictimIn(Relationship, type="VICTIM_IN"):
    pass

class KillerIn(Relationship, type="KILLER_IN"):
    pass

class HappenedIn(Relationship, type="HAPPENED_IN"):
    pass

class PartOf(Relationship, type="PART_OF"):
    pass
```

ZAGREB Graph Data

Exploring the database

Simple analytics on GoT dataset

Simple analytics with GQLAlchemy

```
results = memgraph.execute_and_fetch(
    """
    MATCH (d:Death)
    RETURN count(d) AS total_number_of_deaths;
    """
)

print(next(results))
```

```
{'total_number_of_deaths': 2224}
```

```
results = memgraph.execute_and_fetch(
    """
    MATCH (c:Character)
    RETURN count(c) AS total_number_of_characters;
    """
)

print(next(results))
```

```
{'total_number_of_characters': 289}
```


VIA 9GAG.COM
'IT'S HARD KILLING OFF SO MANY CHARACTERS.'
GEORGE R.R. MARTIN
A DANCE WITH DRAGONS
YOU'RE ADORABLE

Simple analytics with GQLAlchemy

```
results = memgraph.execute_and_fetch(
    """
    MATCH (s:Season)
    RETURN count(s) AS total_number_of_seasons;
    """
)

print(next(results))
```

```
{'total_number_of_seasons': 8}
```

```
results = memgraph.execute_and_fetch(
    """
    MATCH (e:Episode)-[:PART_OF]->(s:Season)
    RETURN s, collect(e) AS episodes
    ORDER BY s.number;
    """
)

for result in results:
    print("Season:", result["s"].number, "Number of episodes:", len(result["episodes"]))
```

```
Season: 1 Number of episodes: 9
Season: 2 Number of episodes: 9
Season: 3 Number of episodes: 9
Season: 4 Number of episodes: 10
Season: 5 Number of episodes: 10
Season: 6 Number of episodes: 9
Season: 7 Number of episodes: 7
Season: 8 Number of episodes: 6
```

Season	Episodes
1	10
2	10
3	10
4	10
5	10
6	10
7	7
8	6

No deaths episodes

Which GoT episodes did not have any killings?

Seasons 1, 2, 3 and 6 - no deaths episodes

- Season 1 - Episode 3 - Lord Snow
- Season 2 - Episode 8 - The Prince of Winterfell
- Season 3 - Episode 7 - The Bear and the Maiden Fair
- Season 6 - Episode 6 - Blood of My Blood

```
season_1 = {'Winter Is Coming ', 'The Kingsroad ', 'Lord Snow',
            'Cripples Bastards and Broken Things ', 'The Wolf and the Lion ',
            'A Golden Crown ', 'You Win or You Die', 'The Pointy End ',
            'Baelor ', 'Fire and Blood'}

results = memgraph.execute_and_fetch(
    """
    MATCH (e:Episode)-[p:PART_OF]->(s:Season {number: 1})
    RETURN e;
    """
)

season_1_deaths = set()

for result in results:
    season_1_deaths.add(result["e"].name)

print(season_1 - season_1_deaths)
```

```
{'Lord Snow'}
```

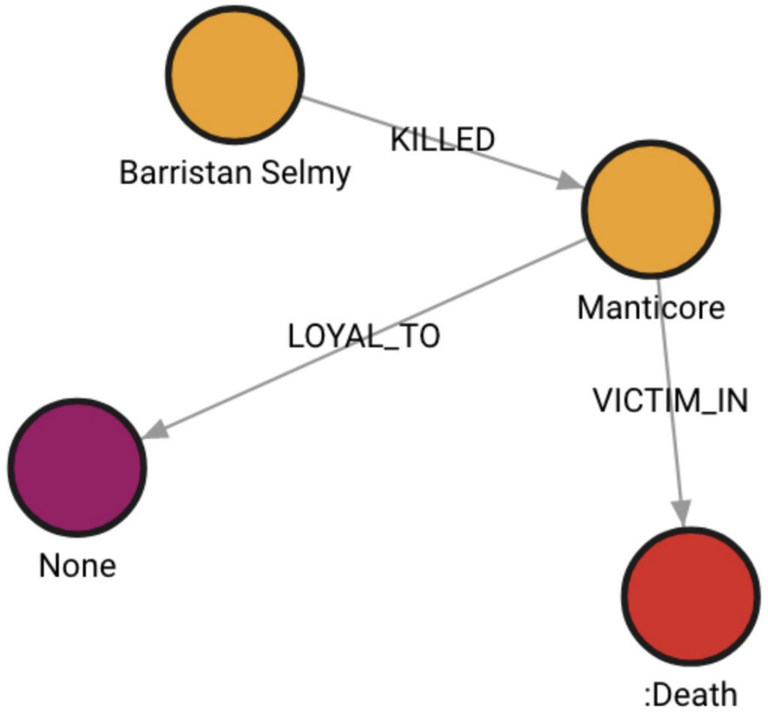
Episodes without deaths by some sources[1]

- Season 3 - Episode 1 - **Valar Dohaeris**

```
results = memgraph.execute_and_fetch(
    """
    MATCH (c:Character)-[v:VICTIM_IN]->(d:Death)
    -[:HAPPENED_IN]->(e:Episode {number: 1})
    -[p:PART_OF]->(s:Season {number: 3})
    RETURN c;
    """
)

print(next(results)["c"].name)
```

```
Manticore
```




POISONOUS MANTICORE

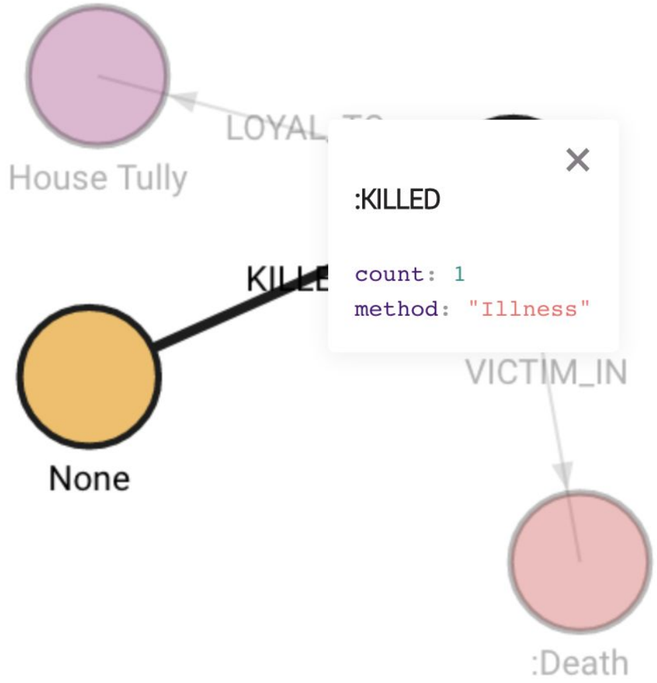
Episodes without deaths by some sources[1]

- Season 3 - Episode 2 - **Dark Wings, Dark Words**

```
results = memgraph.execute_and_fetch(
    """
    MATCH (c:Character)-[v:VICTIM_IN]->(d:Death)
    -[:HAPPENED_IN]->(e:Episode {number: 2})
    -[p:PART_OF]->(s:Season {number: 3})
    RETURN c;
    """
)

print(next(results)["c"].name)
```

```
Hoster Tully
```



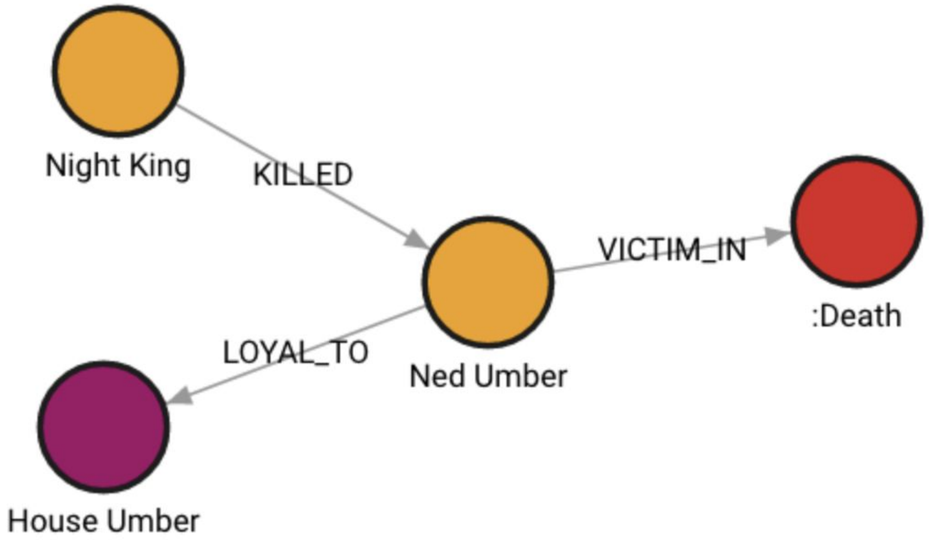
Episodes without deaths by some sources[1]

- Season 8 - Episode 2 - A Knight of the Seven Kingdoms

```
results = memgraph.execute_and_fetch(
    """
    MATCH (c:Character)-[v:VICTIM_IN]->(d:Death)
    -[:HAPPENED_IN]->(e:Episode {number: 2})
    -[p:PART_OF]->(s:Season {number: 8})
    RETURN c;
    """
)

print(next(results)["c"].name)
```

```
Ned Umber
```



Characters that killed themselves

Who managed to run from other killers, but still ended up dead?

Characters who killed themselves

```
results = memgraph.execute_and_fetch(
    """
    MATCH (c:Character)-[k:KILLED]->(d:Character)
    WHERE c = d
    RETURN c, k;
    """
)

for result in results:
    print("Character:", result["c"].name, "| Method:", result["k"].method)
```

```
Character: Dothraki man | Method: Arakh
Character: Sandor the Hound Clegane | Method: Dragonfire (Dragon)
Character: Melisandre the Red Woman of Asshai | Method: Old Age
Character: Bolton soldier | Method: Arrow
Character: Jaqen Hghar | Method: Poison
Character: Peasant | Method: Poison
Character: Greyjoy Soldier | Method: Sword
Character: Greyjoy Soldier | Method: Axe
Character: Greyjoy Soldier | Method: Mace
Character: Greyjoy Soldier | Method: Knife
Character: Tommen Baratheon | Method: Falling
Character: Sons of the Harpy agent | Method: Sword
Character: Aemon Targaryen | Method: Old Age
Character: Gladiator | Method: Sword
Character: Gladiator | Method: Spear
Character: Selyse Florent | Method: Rope
```

```
results = memgraph.execute_and_fetch(
    """
    MATCH (c:Character)-[k:KILLED {method: 'Old Age'}]->(d:Character)
    WHERE c = d
    RETURN c;
    """
)

for result in results:
    print("Character:", result["c"].name)
```

```
Character: Melisandre the Red Woman of Asshai
Character: Aemon Targaryen
```


BADASS LEVEL:
DIED OF NATURAL CAUSES IN GAME OF THRONES

ZAGREB Graph Data

Graph traversals and PageRank

Is the importance of episodes and locations really measured by the number of deaths?

The locations where the most deaths occurred

```
results = memgraph.execute_and_fetch(
    """
    MATCH (l:Location)<-[:HAPPENED_IN]-(d:Death)
    RETURN l AS location, count(d) AS death_count
    ORDER BY death_count DESC
    LIMIT 10;
    """
)

for result in results:
    print(result["location"].name, result["death_count"])
```

```
Kings Landing 1170
Roseroad 207
Winterfell 169
Meereen 136
The Twins 89
Beyond the Wall 78
Castle Black 66
The Narrow Sea 35
Riverlands 28
The Wall 26
```

The most ‘important’ locations with PageRank

```
results = memgraph.execute_and_fetch(
    """
    CALL pagerank.get()
    YIELD *
    WITH node, rank
    WHERE node:Location
    RETURN node, rank
    ORDER BY rank DESC
    LIMIT 10;
    """
)

for result in results:
    print("Episode:", result["node"].name, "| Rank:", result["rank"])
```

```
Episode: Kings Landing | Rank: 0.06106011381918924
Episode: Roseroad | Rank: 0.010786290904476399
Episode: Winterfell | Rank: 0.009650604216610618
Episode: Meereen | Rank: 0.007653449136907163
Episode: The Twins | Rank: 0.005029859049634092
Episode: Beyond the Wall | Rank: 0.0048574484597048535
Episode: Castle Black | Rank: 0.003955983097311168
Episode: The Narrow Sea | Rank: 0.0021040262793516795
Episode: Riverlands | Rank: 0.0019043870463720024
Episode: The Wall | Rank: 0.001583596860766157
```

The episodes with the most deaths

```
results = memgraph.execute_and_fetch(
    """
    MATCH (d:Death)-[:HAPPENED_IN]->(e:Episode)
    RETURN e AS episode, count(d) AS death_count
    ORDER BY death_count DESC
    LIMIT 10;
    """
)

for result in results:
    print(result["episode"].name, result["death_count"])
```

```
The Bells  844
The Spoils of War  205
The Winds of Winter 203
Battle of the Bastards 133
The Watchers on the Wall  86
Blackwater  72
The Dance of Dragons 58
Dragonstone  54
Sons of the Harpy 48
The Rains of Castamere 45
```

The most ‘important’ episodes with PageRank

```
results = memgraph.execute_and_fetch(
    """
    CALL pagerank.get()
    YIELD *
    WITH node, rank
    WHERE node:Episode
    RETURN node, rank
    ORDER BY rank DESC
    LIMIT 10;
    """
)

for result in results:
    print("Episode:", result["node"].name, "| Rank:", result["rank"])
```

```
Episode: The Bells  | Rank: 0.043164724601018695
Episode: The Winds of Winter | Rank: 0.010801371244279762
Episode: The Spoils of War  | Rank: 0.010652824071617746
Episode: Battle of the Bastards | Rank: 0.007326424536668109
Episode: The Watchers on the Wall  | Rank: 0.004888653411156207
Episode: Blackwater  | Rank: 0.0040451560244256475
Episode: The Dance of Dragons | Rank: 0.003304325314698726
Episode: Dragonstone  | Rank: 0.002979021268992005
Episode: Sons of the Harpy | Rank: 0.002835536547706026
Episode: The Rains of Castamere | Rank: 0.002811794141977363
```

ZAGREB Graph Data

Seasons and allegiances

Kills, ratings and house fights

Number of kills per season

```
results = memgraph.execute_and_fetch(
    """
    MATCH (d:Death)-[:HAPPENED_IN]->(s:Season)
    RETURN s AS season, count(d) AS death_count
    ORDER BY death_count DESC;
    """
)

for result in results:
    print(result["season"].number, result["death_count"])
```

```
8 892
6 396
7 334
4 170
5 158
2 130
3 86
1 58
```

Top seasons by their IMDB rating

```
results = memgraph.execute_and_fetch(
    """
    MATCH (e:Episode)-[:PART_OF]->(s:Season)
    RETURN s AS season, round(100 * avg(e.imdb_rating))/100 AS rating
    ORDER BY rating DESC;
    """
)

for result in results:
    print("Season:", result["season"].number, "| Rating:", result["rating"])
```

```
Season: 4 | Rating: 9.31
Season: 1 | Rating: 9.14
Season: 6 | Rating: 9.13
Season: 7 | Rating: 9.1
Season: 3 | Rating: 9.09
Season: 2 | Rating: 8.98
Season: 5 | Rating: 8.83
Season: 8 | Rating: 6.38
```

Top 10 allegiances by the kill/death ratio (KDR)

```
results = memgraph.execute_and_fetch(
    """
    MATCH (:Character)-[death:KILLED]->(:Character)-[:LOYAL_TO]->(a:Allegiance)
    WITH a, sum(death.count) AS deaths
    MATCH (:Character)<-[kill:KILLED]-(:Character)-[:LOYAL_TO]->(a)
    RETURN a AS allegiance,
           sum(kill.count) AS kills,
           deaths,
           round(100 *(tofloat(sum(kill.count))/deaths))/100 AS KDR
    ORDER BY KDR DESC
    LIMIT 10;
    """
)

for result in results:
    print("Allegiance:", result["allegiance"].name, "| Kills:", result["kills"],
          "| Deaths:", result["deaths"], "| KDR:", result["KDR"])
```

```
Allegiance: White Walkers | Kills: 205 | Deaths: 5 | KDR: 41.0
Allegiance: Warlocks of Qarth | Kills: 11 | Deaths: 1 | KDR: 11.0
Allegiance: House Clegane | Kills: 54 | Deaths: 5 | KDR: 10.8
Allegiance: House Targaryen | Kills: 1329 | Deaths: 151 | KDR: 8.8
Allegiance: Lord of Light | Kills: 7 | Deaths: 1 | KDR: 7.0
Allegiance: House Umber | Kills: 7 | Deaths: 2 | KDR: 3.5
Allegiance: Faceless Men | Kills: 3 | Deaths: 1 | KDR: 3.0
Allegiance: House Baratheon of Kings Landing | Kills: 194 | Deaths: 66 | KDR: 2.94
Allegiance: Nights Watch | Kills: 394 | Deaths: 143 | KDR: 2.76
Allegiance: Sand Snakes | Kills: 8 | Deaths: 3 | KDR: 2.67
```

Who had more casualties in the Battle of Bastards - Starks or Boltons?

```
results = memgraph.execute_and_fetch(
    """
    MATCH (c:Character)-[:LOYAL_TO]->(a:Allegiance)
    MATCH (c)-[:VICTIM_IN]-(d:Death)-[:HAPPENED_IN]-(:Episode {name: 'Battle of the Bastards'})
    RETURN a AS house, count(d) AS death_count
    ORDER BY death_count DESC
    LIMIT 2;
    """
)

for result in results:
    print("Allegiance:", result["house"].name, "| Deaths:", result["death_count"])
```

```
Allegiance: House Bolton | Deaths: 49
Allegiance: House Stark | Deaths: 42
```

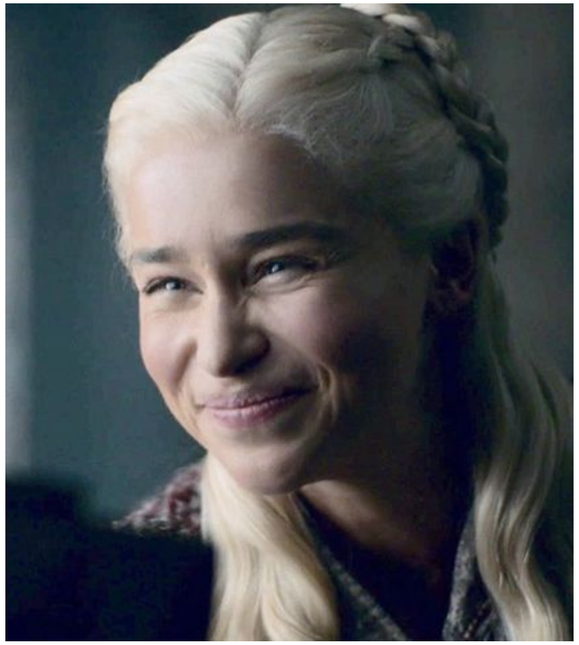
Is Daenerys that bad?

How many people did the Mother of Dragons slay?

```
results = memgraph.execute_and_fetch(
    """
    MATCH (daenerys:Character {name: 'Daenerys Targaryen'})-[:KILLED]->(victim:Character)
    MATCH (daenerys)-[:KILLER_IN]->(d:Death)<-[:VICTIM_IN]-(victim)
    MATCH (d)-[:HAPPENED_IN]-(e:Episode)
    RETURN DISTINCT victim, count(d) AS kill_count, e AS episode
    ORDER BY kill_count DESC;
    """
)

for result in results:
    print("Victim:", result["victim"].name, "| Kill Count:", result["kill_count"]
          , "| Episode name:", result["episode"].name)
```

```
Victim: Golden Company soldier | Kill Count: 374 | Episode name: The Bells
Victim: Kings Landing Citizen | Kill Count: 227 | Episode name: The Bells
Victim: Lannister soldier | Kill Count: 178 | Episode name: The Spoils of War
Victim: Lannister soldier | Kill Count: 166 | Episode name: The Bells
Victim: Sons of the Harpy agent | Kill Count: 54 | Episode name: The Dance of Dragons
Victim: Dothraki Khal | Kill Count: 14 | Episode name: Book of the Stranger
Victim: The Masters Soldier | Kill Count: 12 | Episode name: Battle of the Bastards
Victim: Greyjoy Soldier | Kill Count: 2 | Episode name: The Bells
Victim: Horse | Kill Count: 2 | Episode name: The Spoils of War
Victim: Xaro Xhoan Daxos | Kill Count: 1 | Episode name: Valar Morghulis
Victim: Khal Moro | Kill Count: 1 | Episode name: Book of the Stranger
Victim: Golden Company horse | Kill Count: 1 | Episode name: The Bells
Victim: Pyat Pree | Kill Count: 1 | Episode name: Valar Morghulis
Victim: Zalla | Kill Count: 1 | Episode name: The Children
Victim: Kraznys mo Nakloz | Kill Count: 1 | Episode name: And Now His Watch Is Ended
Victim: Varys | Kill Count: 1 | Episode name: The Bells
Victim: Doreah | Kill Count: 1 | Episode name: Valar Morghulis
Victim: Cersei Lannister | Kill Count: 1 | Episode name: The Bells
Victim: Goat | Kill Count: 1 | Episode name: The Laws of Gods and Men
Victim: Mirri Maz Duur | Kill Count: 1 | Episode name: Fire and Blood
Victim: Khal Drogo | Kill Count: 1 | Episode name: Fire and Blood
Victim: Dickon Tarly | Kill Count: 1 | Episode name: Eastwatch
Victim: Randyll Tarly | Kill Count: 1 | Episode name: Eastwatch
Victim: Jaime Lannister | Kill Count: 1 | Episode name: The Bells
```



Who is the killer influencer in the GoT world?

```
results = memgraph.execute_and_fetch(
    """
    CALL betweenness_centrality.get(False)
    YIELD node, betweenness_centrality
    WITH node, betweenness_centrality
    WHERE 'Character' IN labels(node)
    RETURN node, betweenness_centrality
    ORDER BY betweenness_centrality DESC
    LIMIT 10;
    """
)

for result in results:
    print(result["node"].name, "| BC:", result["betweenness_centrality"])
```

```
Daenerys Targaryen | BC: 0.26416716859707723
Lannister soldier | BC: 0.11163138590238252
Sons of the Harpy agent | BC: 0.04326120746568803
Jon Snow | BC: 0.04152733973021268
Wildling | BC: 0.03782396045716543
Stark Soldier | BC: 0.028609456095599388
Baratheon of Dragonstone soldier | BC: 0.027533816622595363
Cersei Lannister | BC: 0.026445092829209992
Stark soldier | BC: 0.024328184592014475
Nights Watch brother | BC: 0.022719177729567427
```

ZAGREB Graph Data

You Know Nothing, Jon Snow

Who would survive if Jon Snow stayed dead?

```
results = memgraph.execute_and_fetch(
    """
    MATCH (jon:Character {name: 'Jon Snow'})-[:KILLED]->(victim:Character)
    MATCH (jon)-[:VICTIM_IN]->(jon_death:Death)
    MATCH (jon)-[:KILLER_IN]->(victim_death:Death)<-[:VICTIM_IN]-(victim)
    WHERE victim_death.order > jon_death.order
    RETURN DISTINCT victim, count(victim_death) AS kill_count
    ORDER BY kill_count DESC;
    """
)

for result in results:
    print("Victim:", result["victim"].name, "| Kill count:", result["kill_count"])
```

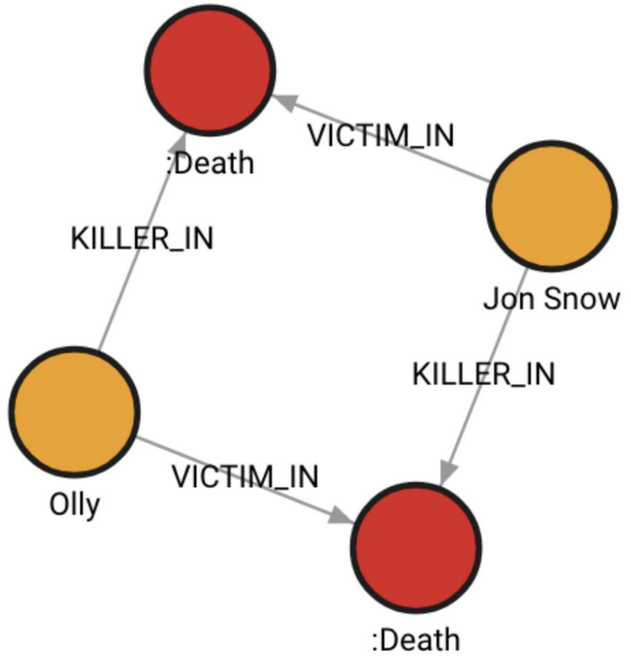
```
Victim: Bolton soldier | Kill count: 20
Victim: Lannister soldier | Kill count: 7
Victim: Daenerys Targaryen | Kill count: 1
Victim: Olly | Kill count: 1
Victim: Stark Soldier | Kill count: 1
Victim: Alliser Thorne | Kill count: 1
Victim: Othell Yarwyck | Kill count: 1
Victim: Bowen Marsh | Kill count: 1
```

Who killed Jon Snow?

```
results = memgraph.execute_and_fetch(
    """
    MATCH (jon:Character {name: 'Jon Snow'})-[v:VICTIM_IN]
    ->(d:Death)<-[k:KILLER_IN]-(c:Character)
    WITH jon, v, d, k, c
    MATCH (c)-[v2:VICTIM_IN]->(d2:Death)<-[k2:KILLER_IN]-(c2:Character)
    RETURN c, jon;
    """
)

for result in results:
    print(result["c"].name, "killed", result["jon"].name, "and",
          result["jon"].name, "killed", result["c"].name)
```

```
Olly killed Jon Snow and Jon Snow killed Olly
```



Who is the biggest traitor?

```
results = memgraph.execute_and_fetch(
    """
    MATCH (killer:Character)-[:KILLED]->(victim:Character)
    MATCH (killer)-[:LOYAL_TO]->(a:Allegiance)<-[:LOYAL_TO]-(victim)
    RETURN killer AS traitor, count(victim) AS kill_count
    ORDER BY kill_count DESC
    LIMIT 10;
    """
)

for result in results:
    print("Traitor:", result["traitor"].name, "| Kill count:", result["kill_count"])
```

```
Traitor: Jon Snow | Kill count: 9
Traitor: Ramsay Bolton | Kill count: 4
Traitor: Theon Greyjoy | Kill count: 4
Traitor: Sandor the Hound Clegane | Kill count: 3
Traitor: Reek | Kill count: 2
Traitor: Euron Greyjoy | Kill count: 2
Traitor: Daario Naharis | Kill count: 2
Traitor: Gregor the Mountain Clegane | Kill count: 2
Traitor: Daenerys Targaryen | Kill count: 2
Traitor: Tyrion Lannister | Kill count: 2
```

Dijkstra killing it

The shortest path of killings with highest kill count

```
results = memgraph.execute_and_fetch(
    """
    MATCH p = (:Character)-[:KILLED * wShortest (e,v | e.count) kill_count]->(:Character)
    RETURN nodes(p) AS kill_list, kill_count
    ORDER BY kill_count DESC
    LIMIT 1;
    """
)

for result in results:
    for kill in result["kill_list"]:
        print(kill.properties["name"])
```

```
Olly
Jon Snow
Daenerys Targaryen
Golden Company soldier
```


MAY THE
FOURTH
BE WITH YOU


MY SWORD IS ON FIRE
THAT'S CUTE

ZAGREB Graph Data

Boris Agatić

Unchained: How to transform Bitcoin Blockchain into a Graph

WESPA Spaces
Wednesday, June 8th, 6 PM

Jupyter Notebook

https://github.com/memgraph/jupyter-memgraph-tutorials/blob/main/got-analysis/game-of-graphs.ipynb

Memgraph Playground

https://playground.memgraph.com/sandbox/game-of-thrones-deaths

Memgraph Discord Community

memgr.ph/discord

Channel #graph-data-zagreb

Tell us what you like and what you don’t like about Graph Data Zagreb meetups: