Project 3: Neo4j

Data Science and Engineering

Data Bases

Group 3

Alejandro Leonardo García Navarro

Melania Guerra Ulloa

EXERCISE 1 **3**

Insert data 3

Display schema/graph 7

Query/Update/Remove 8

EXERCISE 2 **11**

Create 11

Display Schema/Graph 13

Query 14

EXERCISE 1

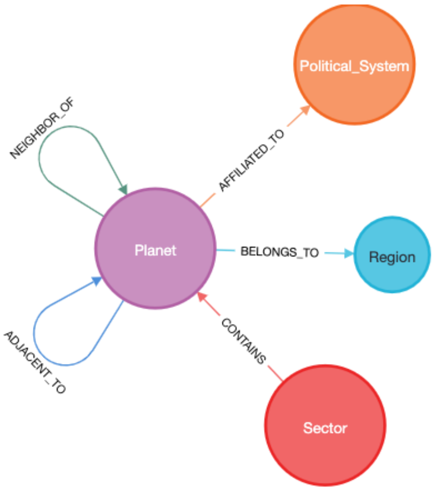
Insert data

```
CREATE
  (Core:Region {name: 'Core'}),
  (Colonies:Region {name: 'Colonies'}),
  (Outer_rim:Region {name: 'Outer Rim'}),
  (Expansion_region:Region {name: 'Expansion Region'}),
  (Inner_rim:Region {name: 'Inner Rim'}),
  (Mid_rim:Region {name: 'Mid Rim'}),
  (Deep_core:Region {name: 'Deep Core'}),
  (Arkanis_sector:Sector {name: 'Arkanis'}),
  (TransNebular_sector:Sector {name: 'Trans-Nebular'}),
  (Karthakk_sector:Sector {name: 'Karthakk'}),
  (Lambda_sector:Sector {name: 'Lambda'}),
  (Doldur_sector:Sector {name: 'Doldur'}),
  (Abrion_sector:Sector {name: 'Abrion'}),
  (Tynna_sector:Sector {name: 'Tynna'}),
  (Manda_sector:Sector {name: 'Manda'}),
  (Herdessa_sector:Sector {name: 'Herdessa'}),
  (BothanSpace_sector:Sector {name: 'Bothan Space'}),
  (Daalang_sector:Sector {name: 'Daalang'}),
  (Chaykin_sector:Sector {name: 'Chaykin'}),
  (Narvath_sector:Sector {name: 'Narvath'}),
  (tatooine:Planet{name: 'Tatooine'}),
  (piroket:Planet{name: 'Piroket'}),
  (arkanis:Planet{name: 'Arkanis'}),
  (pii:Planet{name: 'Pii'}),
  (herdessa:Planet{name: 'Herdessa'}),
  (llanic:Planet{name: 'Llanic'}),
  (molavar:Planet{name: 'Molavar'}),
  (manda:Planet{name: 'Manda'}),
  (daalang:Planet{name:'Daalang'}),
  (bothawui:Planet{name: 'Bothawui'}),
  (aridus:Planet{name: 'Aridus'}),
  (milagro:Planet{name: 'Milagro'}),
  (iktotchon:Planet{name: 'Iktotchon'}),
  (mon_gazza:Planet{name: 'Mon Gazza'}),
  (druckenwell:Planet{name: 'Druckenwell'}),
  (denon:Planet{name: 'Denon'}),
  (corellia:Planet{name: 'Corellia'}),
  (coruscant:Planet{name: 'Coruscant'}),
  (brentaal:Planet{name: 'Brentaal'}),
  (alderaan:Planet{name: 'Alderaan'}),
  (fedalle:Planet{name: 'Fedalle'}),
  (commenor:Planet{name: 'Commenor'}),
  (trellen:Planet{name: 'Trellen'}),
  (exodeen:Planet{name: 'Exodeen'}),
  (allanteen:Planet{name: 'Allanteen'}),
  (chardaan:Planet{name: 'Chardaan'}),
  (foless:Planet{name: 'Foless'}),
  (empress_teta:Planet{name: 'Empress Teta'}),
  (byss:Planet{name: 'Byss'}),
  (quellor:Planet{name: 'Quellor'}),
  (goroth:Planet{name: 'Goroth'}),
  (habassa:Planet{name: 'Habassa'}),
  (monor:Planet{name: 'Monor'}),
  (gendrah_narvin:Planet{name: 'Gendrah-Narvin'}),
  (devaron:Planet{name: 'Devaron'}),
  (duro:Planet{name: 'Duro'}),
  (sacorria:Planet{name: 'Sacorria'}),
  (samaria:Planet{name: 'Samaria'}),
  (puul:Planet{name: 'Puul'}),
  (perma:Planet{name: 'Perma'}),
  (goorla:Planet{name: 'Goorla'}),
  (pria:Planet{name: 'Pria'}),
  (xorth:Planet{name: 'Xorth'}),
  (palawa:Planet{name: 'Palawa'}),
  (aldraig:Planet{name: 'Aldraig'}),
  (vanjervalis:Planet{name: 'Vanjervalis'}),
(galactic_empire:Political_System{system: 'Galactic Empire'}),
(hutt_clan:Political_System{system: 'Hutt Clan'}),
(jedi_order:Political_System{system: 'Jedi Order'}),
(galactic_republic:Political_System{system: 'Galactic Republic'}),
(confederacy:Political_System{system: 'Confederacy of Independent Systems'}),
(neutral:Political_System{system: 'Neutral'}),
(Arkanis_sector)-[:CONTAINS]->(tatooine)-[:BELONGS_TO]->(Outer_rim),
(Arkanis_sector)-[:CONTAINS]->(piroket)-[:BELONGS_TO]->(Outer_rim),
(Arkanis_sector)-[:CONTAINS]->(arkanis)-[:BELONGS_TO]->(Outer_rim),
(Arkanis_sector)-[:CONTAINS]->(pii)-[:BELONGS_TO]->(Outer_rim),
(Herdessa_sector)-[:CONTAINS]->(herdessa)-[:BELONGS_TO]->(Mid_rim),
(Karthakk_sector)-[:CONTAINS]->(llanic)-[:BELONGS_TO]->(Outer_rim),
(Lambda_sector)-[:CONTAINS]->(mon_gazza)-[:BELONGS_TO]->(Mid_rim),
(Abrion_sector)-[:CONTAINS]->(molavar)-[:BELONGS_TO]->(Mid_rim),
(Manda_sector)-[:CONTAINS]->(manda)-[:BELONGS_TO]->(Mid_rim),
(Doldur_sector)-[:CONTAINS]->(druckenwell)-[:BELONGS_TO]->(Mid_rim),
(Doldur_sector)-[:CONTAINS]->(monor)-[:BELONGS_TO]->(Mid_rim),
(BothanSpace_sector)-[:CONTAINS]->(bothawui)-[:BELONGS_TO]->(Mid_rim),
(Daalang_sector)-[:CONTAINS]->(daalang)-[:BELONGS_TO]->(Mid_rim),
(Narvath_sector)-[:CONTAINS]->(aridus)-[:BELONGS_TO]->(Expansion_region),
(Narvath_sector)-[:CONTAINS]->(iktotchon)-[:BELONGS_TO]->(Expansion_region),
(Tynna_sector)-[:CONTAINS]->(allanteen)-[:BELONGS_TO]->(Expansion_region),
(Chaykin_sector)-[:CONTAINS]->(milagro)-[:BELONGS_TO]->(Expansion_region),
(TransNebular_sector)-[:CONTAINS]->(goroth)-[:BELONGS_TO]->(Mid_rim),
(Herdessa_sector)-[:CONTAINS]->(habassa)-[:BELONGS_TO]->(Mid_rim),
(chardaan)-[:BELONGS_TO]->(Inner_rim),
(foless)-[:BELONGS_TO]->(Inner_rim),
(denon)-[:BELONGS_TO]->(Inner_rim),
(gendrah_narvin)-[:BELONGS_TO]->(Inner_rim),
(corellia)-[:BELONGS_TO]->(Inner_rim),
(exodeen)-[:BELONGS_TO]->(Colonies),
(vanjervalis)-[:BELONGS_TO]->(Colonies),
(devaron)-[:BELONGS_TO]->(Colonies),
(commenor)-[:BELONGS_TO]->(Colonies),
(quellor)-[:BELONGS_TO]->(Colonies),
(duro)-[:BELONGS_TO]->(Core),
(sacorria)-[:BELONGS_TO]->(Core),
(goorla)-[:BELONGS_TO]->(Core),
(coruscant)-[:BELONGS_TO]->(Core),
(trellen)-[:BELONGS_TO]->(Core),
(pria)-[:BELONGS_TO]->(Core),
(fedalle)-[:BELONGS_TO]->(Core),
(brentaal)-[:BELONGS_TO]->(Core),
(alderaan)-[:BELONGS_TO]->(Core),
(puul)-[:BELONGS_TO]->(Core),
(perma)-[:BELONGS_TO]->(Core),
(samaria)-[:BELONGS_TO]->(Core),
(xorth)-[:BELONGS_TO]->(Core),
(palawa)-[:BELONGS_TO]->(Core),
(aldraig)-[:BELONGS_TO]->(Core),
(empress_teta)-[:BELONGS_TO]->(Deep_core),
(byss)-[:BELONGS_TO]->(Deep_core),
(goroth)-[:NEIGHBOR_OF]->(pii),
(habassa)-[:NEIGHBOR_OF]->(herdessa),
(monor)-[:NEIGHBOR_OF]->(druckenwell),
(gendrah_narvin)-[:NEIGHBOR_OF]->(denon),
(devaron)-[:NEIGHBOR_OF]->(foless),
(duro)-[:NEIGHBOR_OF]->(corellia),
(sacorria)-[:NEIGHBOR_OF]->(corellia),
(goorla)-[:NEIGHBOR_OF]->(corellia),
(vanjervalis)-[:NEIGHBOR_OF]->(exodeen),
(vanjervalis)-[:NEIGHBOR_OF]->(quellor),
(pria)-[:NEIGHBOR_OF]->(trellen),
(pria)-[:NEIGHBOR_OF]->(fedalle),
(aridus)-[:NEIGHBOR_OF]->(milagro),
(iktotchon)-[:NEIGHBOR_OF]->(allanteen),
(iktotchon)-[:NEIGHBOR_OF]->(aridus),
(palawa)-[:NEIGHBOR_OF]->(xorth),
(palawa)-[:NEIGHBOR_OF]->(aldraig),
(aldraig)-[:NEIGHBOR_OF]->(alderaan),
(samaria)-[:NEIGHBOR_OF]->(corellia),
(samaria)-[:NEIGHBOR_OF]->(exodeen),
(perma)-[:NEIGHBOR_OF]->(puul),
(perma)-[:NEIGHBOR_OF]->(trellen),
(tatooine)-[:AFFILIATED_TO]->(galactic_empire),
(tatooine)-[:AFFILIATED_TO]->(hutt_clan),
(piroket)-[:AFFILIATED_TO]->(galactic_empire),
(piroket)-[:AFFILIATED_TO]->(hutt_clan),
(pii)-[:AFFILIATED_TO]->(galactic_empire),
(pii)-[:AFFILIATED_TO]->(galactic_republic),
(goroth)-[:AFFILIATED_TO]->(galactic_empire),
(goroth)-[:AFFILIATED_TO]->(galactic_republic),
(arkanis)-[:AFFILIATED_TO]->(galactic_empire),
(arkanis)-[:AFFILIATED_TO]->(hutt_clan),
(herdessa)-[:AFFILIATED_TO]->(galactic_empire),
(herdessa)-[:AFFILIATED_TO]->(galactic_republic),
(habassa)-[:AFFILIATED_TO]->(neutral),
(druckenwell)-[:AFFILIATED_TO]->(galactic_empire),
(druckenwell)-[:AFFILIATED_TO]->(galactic_republic),
(druckenwell)-[:AFFILIATED_TO]->(confederacy),
(daalang)-[:AFFILIATED_TO]->(confederacy),
(monor)-[:AFFILIATED_TO]->(galactic_empire),
(molavar)-[:AFFILIATED_TO]->(hutt_clan),
(molavar)-[:AFFILIATED_TO]->(confederacy),
(manda)-[:AFFILIATED_TO]->(galactic_empire),
(manda)-[:AFFILIATED_TO]->(galactic_republic),
(denon)-[:AFFILIATED_TO]->(galactic_empire),
(gendrah_narvin)-[:AFFILIATED_TO]->(galactic_republic),
(corellia)-[:AFFILIATED_TO]->(galactic_empire),
(samaria)-[:AFFILIATED_TO]->(galactic_empire),
(duro)-[:AFFILIATED_TO]->(galactic_republic),
(duro)-[:AFFILIATED_TO]->(galactic_empire),
(sacorria)-[:AFFILIATED_TO]->(galactic_republic),
(sacorria)-[:AFFILIATED_TO]->(galactic_empire),
(goorla)-[:AFFILIATED_TO]->(galactic_republic),
(goorla)-[:AFFILIATED_TO]->(galactic_empire),
(coruscant)-[:AFFILIATED_TO]->(galactic_empire),
(coruscant)-[:AFFILIATED_TO]->(galactic_republic),
(coruscant)-[:AFFILIATED_TO]->(jedi_order),
(devaron)-[:AFFILIATED_TO]->(galactic_empire),
(devaron)-[:AFFILIATED_TO]->(galactic_republic),
(devaron)-[:AFFILIATED_TO]->(jedi_order),
(devaron)-[:AFFILIATED_TO]->(confederacy),
(exodeen)-[:AFFILIATED_TO]->(galactic_empire),
(vanjervalis)-[:AFFILIATED_TO]->(galactic_empire),
(vanjervalis)-[:AFFILIATED_TO]->(galactic_republic),
(milagro)-[:AFFILIATED_TO]->(galactic_empire),
(milagro)-[:AFFILIATED_TO]->(galactic_republic),
(iktotchon)-[:AFFILIATED_TO]->(galactic_empire),
(iktotchon)-[:AFFILIATED_TO]->(galactic_republic),
(aridus)-[:AFFILIATED_TO]->(galactic_empire),
(aridus)-[:AFFILIATED_TO]->(galactic_republic),
(trellen)-[:AFFILIATED_TO]->(galactic_empire),
(trellen)-[:AFFILIATED_TO]->(galactic_republic),
(aldraig)-[:AFFILIATED_TO]->(galactic_empire),
(aldraig)-[:AFFILIATED_TO]->(galactic_republic),
(puul)-[:AFFILIATED_TO]->(galactic_republic),
(palawa)-[:AFFILIATED_TO]->(galactic_empire),
(palawa)-[:AFFILIATED_TO]->(galactic_republic),
(xorth)-[:AFFILIATED_TO]->(galactic_empire),
(xorth)-[:AFFILIATED_TO]->(galactic_republic),
(perma)-[:AFFILIATED_TO]->(galactic_republic),
(pria)-[:AFFILIATED_TO]->(galactic_empire),
(pria)-[:AFFILIATED_TO]->(galactic_republic),
(fedalle)-[:AFFILIATED_TO]->(galactic_empire),
(fedalle)-[:AFFILIATED_TO]->(galactic_republic),
(brentaal)-[:AFFILIATED_TO]->(galactic_empire),
(brentaal)-[:AFFILIATED_TO]->(galactic_republic),
(brentaal)-[:AFFILIATED_TO]->(confederacy),
(quellor)-[:AFFILIATED_TO]->(galactic_empire),
(quellor)-[:AFFILIATED_TO]->(galactic_republic),
(commenor)-[:AFFILIATED_TO]->(galactic_empire),
(alderaan)-[:AFFILIATED_TO]->(galactic_empire),
(alderaan)-[:AFFILIATED_TO]->(galactic_republic),
(allanteen)-[:AFFILIATED_TO]->(galactic_empire),
(allanteen)-[:AFFILIATED_TO]->(galactic_republic),
(foless)-[:AFFILIATED_TO]->(galactic_republic),
(empress_teta)-[:AFFILIATED_TO]->(galactic_empire),
(empress_teta)-[:AFFILIATED_TO]->(galactic_republic),
(byss)-[:AFFILIATED_TO]->(galactic_empire),
(tatooine)-[:ADJACENT_TO {hyperspaceroute: 'Triellus Trade Route'}]->(arkanis),
(tatooine)-[:ADJACENT_TO {hyperspaceroute: 'Triellus Trade Route'}]->(piroket),
(tatooine)-[:ADJACENT_TO {hyperspaceroute: 'Old Corellian Run'}]->(pii),
(pii)-[:ADJACENT_TO {hyperspaceroute: 'Old Corellian Run'}]->(herdessa),
(piroket)-[:ADJACENT_TO {hyperspaceroute: 'Triellus Trade Route'}]->(molavar),
(arkanis)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(herdessa),
(arkanis)-[:ADJACENT_TO {hyperspaceroute: 'Triellus Trade Route'}]->(llanic),
(llanic)-[:ADJACENT_TO {hyperspaceroute: 'Llanic Spice Run'}]->(mon_gazza),
(herdessa)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(mon_gazza),
(molavar)-[:ADJACENT_TO {hyperspaceroute: 'Manda Merchant Route'}]->(manda),
(manda)-[:ADJACENT_TO {hyperspaceroute: 'Manda Merchant Route'}]->(mon_gazza),
(manda)-[:ADJACENT_TO {hyperspaceroute: 'Bothan Run'}]->(bothawui),
(bothawui)-[:ADJACENT_TO {hyperspaceroute: 'Bothan Run'}]->(daalang),
(daalang)-[:ADJACENT_TO {hyperspaceroute: 'Gamor Run'}]->(aridus),
(aridus)-[:ADJACENT_TO {hyperspaceroute: 'Gamor Run'}]->(milagro),
(bothawui)-[:ADJACENT_TO {hyperspaceroute: 'Reena Trade Route'}]->(druckenwell),
(mon_gazza)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(druckenwell),
(druckenwell)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(milagro),
(milagro)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(allanteen),
(allanteen)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(denon),
(allanteen)-[:ADJACENT_TO {hyperspaceroute: 'Shipwrights Trace'}]->(chardaan),
(chardaan)-[:ADJACENT_TO {hyperspaceroute: 'Shipwrights Trace'}]->(foless),
(foless)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Trade Spine'}]->(corellia),
(denon)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(corellia),
(corellia)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(perma),
(perma)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(xorth),
(xorth)-[:ADJACENT_TO {hyperspaceroute: 'Corellian Run'}]->(coruscant),
(denon)-[:ADJACENT_TO {hyperspaceroute: 'Hydian Way'}]->(exodeen),
(exodeen)-[:ADJACENT_TO {hyperspaceroute: 'Nanth ri Route'}]->(quellor),
(exodeen)-[:ADJACENT_TO {hyperspaceroute: 'Hydian Way'}]->(trellen),
(quellor)-[:ADJACENT_TO {hyperspaceroute: 'Quellor Run'}]->(commenor),
(trellen)-[:ADJACENT_TO {hyperspaceroute: 'Hydian Way'}]->(fedalle),
(fedalle)-[:ADJACENT_TO {hyperspaceroute: 'Hydian Way'}]->(aldraig),
(aldraig)-[:ADJACENT_TO {hyperspaceroute: 'Hydian Way'}]->(brentaal),
(coruscant)-[:ADJACENT_TO {hyperspaceroute: 'Perlimian Trade Route'}]->(brentaal),
(trellen)-[:ADJACENT_TO {hyperspaceroute: 'Trellen Trade Route'}]->(commenor),
(fedalle)-[:ADJACENT_TO {hyperspaceroute: 'Fedalle Run'}]->(commenor),
(commenor)-[:ADJACENT_TO {hyperspaceroute: 'Commenor Run'}]->(alderaan),
(brentaal)-[:ADJACENT_TO {hyperspaceroute: 'Commenor Run'}]->(alderaan),
(coruscant)-[:ADJACENT_TO {hyperspaceroute: 'Byss Run'}]->(empress_teta),
(empress_teta)-[:ADJACENT_TO {hyperspaceroute: 'Byss Run'}]->(byss)
```

Display schema/graph

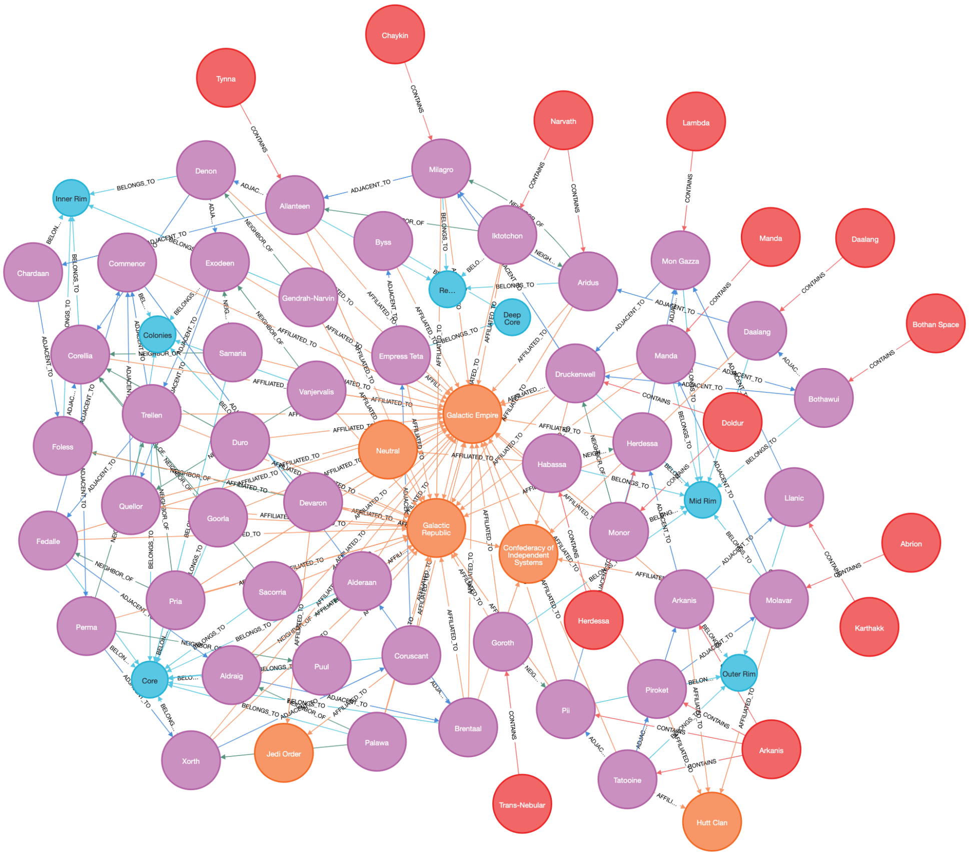
2.1. Write a query to display the schema of your database.

```
call db.schema.visualization();
```



2.2. Recover all nodes and relationships (display graph).

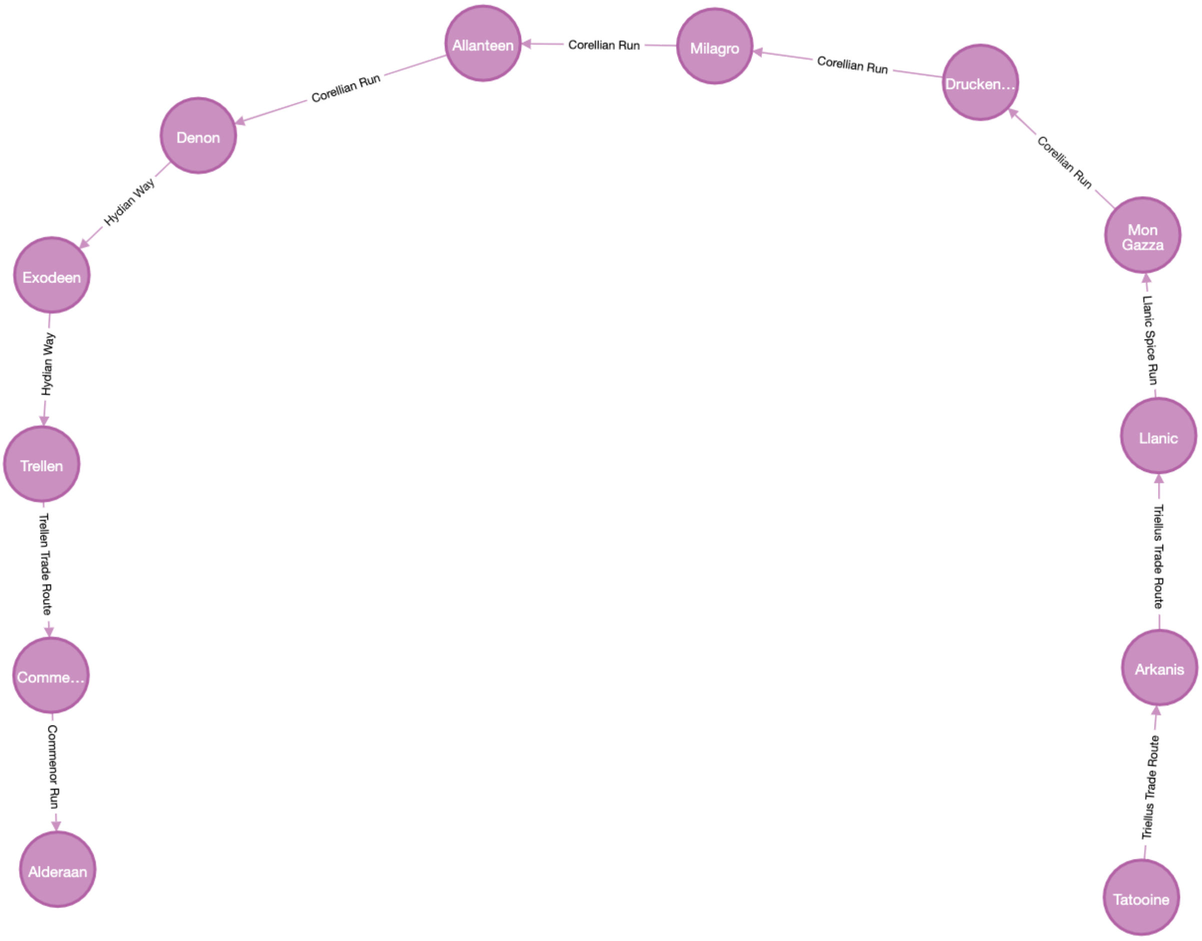
```
match(n) return n;
```



Query/Update/Remove

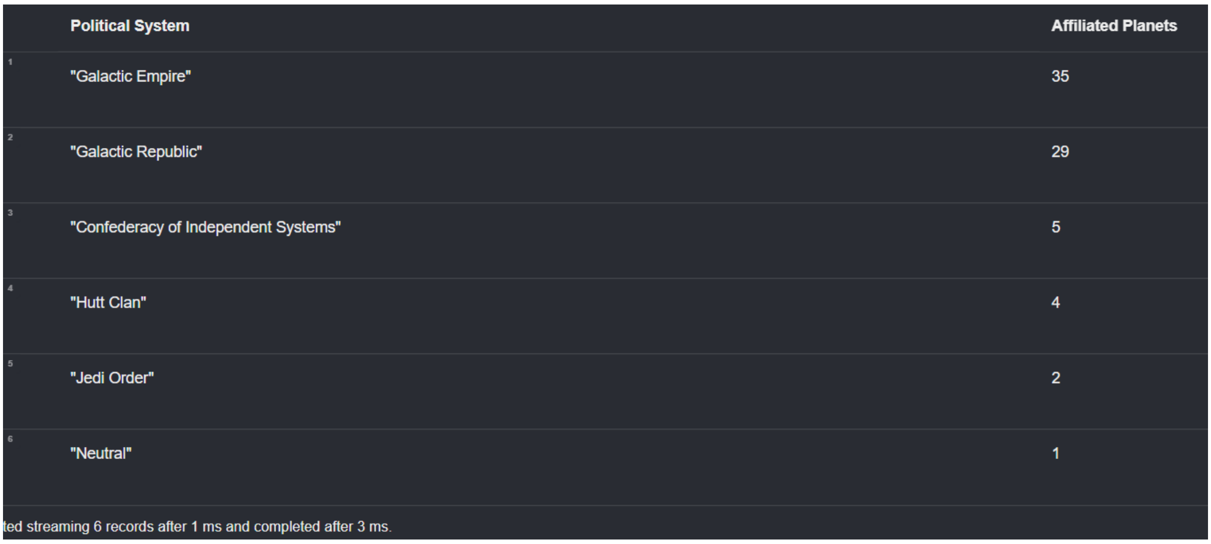
2.3 Find the way to "Alderaan" (return the graph with the best route), starting from the planet "Tatooine".

```
MATCH (p1:Planet {name: 'Alderaan'}),
(p2:Planet {name: 'Tatooine'}),
p = shortestPath((p2)-[:ADJACENT_TO*]-(p1))
WHERE all(r IN relationships(p) WHERE r.hyperspaceroute IS NOT NULL)
RETURN p;
```



2.4. Returns a list indicating which are the dominant political forces in the galaxy and how many affiliated planets each of them has, in descending order.

```
MATCH (p:Planet)-[:AFFILIATED_TO]->(ps:Political_System)
RETURN distinct ps.system AS `Political System`, count(p) AS `Affiliated Planets` ORDER BY
`Affiliated Planets` DESC;
```

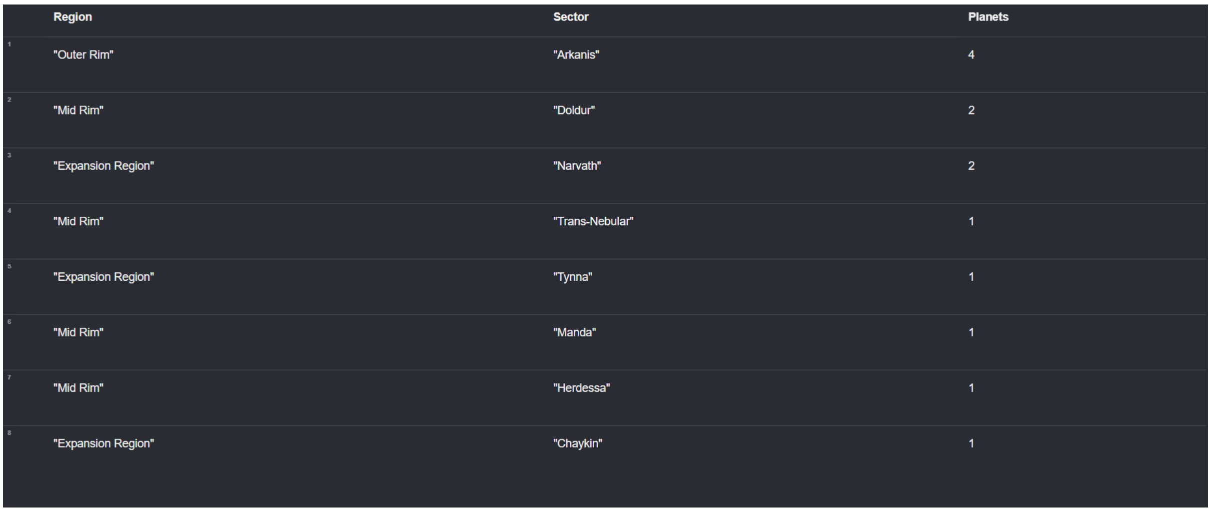


	Political System	Affiliated Planets
1	"Galactic Empire"	35
2	"Galactic Republic"	29
3	"Confederacy of Independent Systems"	5
4	"Hutt Clan"	4
5	"Jedi Order"	2
6	"Neutral"	1

ted streaming 6 records after 1 ms and completed after 3 ms.

2.5. Returns a list where indicate which sectors, regions and planets is the Galactic Empire better stablished, in descending order.

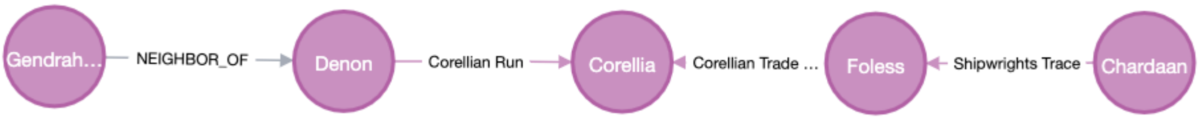
```
MATCH(r:Region)<-[:BELONGS_TO]-(p:Planet)<-[:CONTAINS]-(s:Sector) WHERE
(p)-[:AFFILIATED_TO]->(:Political_System{system:"Galactic Empire"}) RETURN
DISTINCT(r.name) AS Region, s.name AS Sector, count(DISTINCT p) as Planets ORDER BY
Planets DESC;
```



	Region	Sector	Planets
1	"Outer Rim"	"Arkanis"	4
2	"Mid Rim"	"Doldur"	2
3	"Expansion Region"	"Narvath"	2
4	"Mid Rim"	"Trans-Nebular"	1
5	"Expansion Region"	"Tynna"	1
6	"Mid Rim"	"Manda"	1
7	"Mid Rim"	"Herdessa"	1
8	"Expansion Region"	"Chaykin"	1

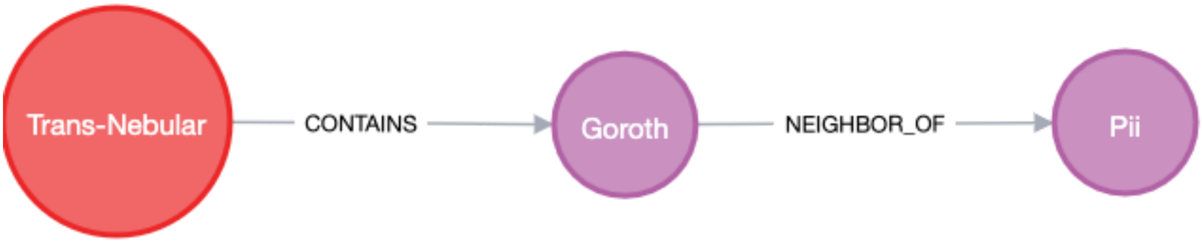
2.6. Find all the planets that belong to the region "Inner Rim", return the graph.

```
MATCH (p:Planet),(m:Region {name: "Inner Rim"}) WHERE (p)-[:BELONGS_TO*] ->(m)
RETURN p;
```



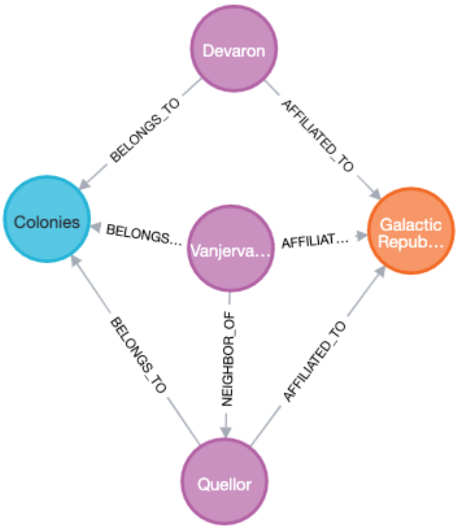
2.7. Return the graph that represents the trajectory from the "Trans-Nebular" sector, to reach the planet "Pii".

```
MATCH (s1:Sector {name: 'Trans-Nebular'}),
(p1:Planet {name: 'Pii'}),
p = allShortestPaths((s1)-[*]-(p1))
RETURN p;
```



2.8. Return the graph that represents the planets that are part of the "Colonies" sector that are under the power of the "Galactic Republic' political system.

```
MATCH (r:Region{name: "Colonies"})<-[:BELONGS_TO]-(p:Planet)-[:AFFILIATED_TO]->
(s:Political_System {system:"Galactic Republic"}) RETURN *;
```



EXERCISE 2

Create

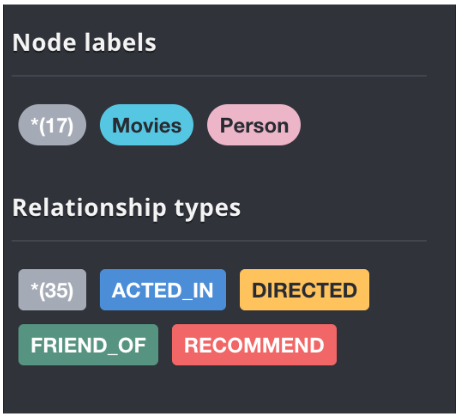
2.1. Write the script, insert data (nodes, relationships, properties) into the graph, display the screenshot of the right menu (Database Information) and output (Table).

→ Insert data:

```
CREATE
(keanu: Person{name: 'Keanu', lastname: 'Reeves', nick: 'Kea', user_account: '@keareeves#'}),
(james: Person{name: 'James', lastname: 'Miller', nick: 'JJ', user_account: '@jjmiller45'}),
(johnson: Person{name: 'James', lastname: 'Johnson', nick: 'Jhonny', user_account: '@jjohnson88'}),
(melania: Person{name: 'Melania', lastname: 'Gutierrez', nick: 'Mel', user_account: '@melgutierrez$'}),
(monica: Person{name: 'Monica', lastname: 'Hernandez', nick: 'Mo', user_account: '@mohernandez%'}),
(daniel: Person{name: 'Daniel', lastname: 'Jimenez', nick: 'Dani', user_account: '@danijimenez*'}),
(sara: Person{name: 'Sara', lastname: 'Carvajal', nick: 'Sarita', user_account: '@saritacarvajal1'}),
(alejandro: Person{name: 'Alejandro', lastname: 'Monge', nick: 'Alex', user_account: '@alexmonge2003'}),
(teresa: Person{name: 'Teresa', lastname: 'Ulloa', nick: 'Tere', user_account: '@tereulloa28'}),
(jorge: Person{name: 'Jorge', lastname: 'Garcia', nick: 'Jorgito', user_account: '@jorgitogarcia__'}),
(mery: Person{name: 'Mery', lastname: 'Perez', nick: 'Mery', user_account: '@meryperez$89'});

CREATE
(loverosie: Movies{title: 'Love Rosie', date: '24/2/2019', budget: '$113,426'}),
(theimpossible: Movies{title: 'The Impossible', date: '16/4/1995', budget: '$279,646,5'}),
(wonder: Movies{title: 'Wonder', date: '18/4/1991', budget: '$347,411,7'}),
(thematrix: Movies{title: 'The Matrix', date: '14/12/2012', budget: '$121,181,9'}),
(prideandprejudice: Movies{title: 'Pride and Prejudice', date: '2/11/2008', budget: '$422,084,0'}),
(nowyouseeme: Movies{title: 'Now You See Me', date: '16/2/1994', budget: '$118,019,3'});

MATCH(a:Person), (b:Person) WHERE a.name =  "Melania" AND b.name =  "Alejandro" CREATE (a)
-[r:FRIEND_OF]->(b) SET r.Friend_since = 2021;
MATCH(a:Person), (b:Person) WHERE a.name = "Alejandro" AND b.name = "James" AND b.lastname =  "Miller"
CREATE (a) -[r:FRIEND_OF]->(b)  SET r.Friend_since = 2018;
MATCH(a:Person), (b:Person) WHERE a.name = "Daniel" AND b.name = "Teresa" CREATE (a) -[r:FRIEND_OF]->(b) SET
r.Friend_since = 2020;
MATCH(a:Person), (b:Person) WHERE a.name = "James" AND a.lastname = "Miller"  AND b.name = "Melania" CREATE
(a) -[r:FRIEND_OF]->(b) SET r.Friend_since = 2015;
MATCH(a:Person), (b:Person) WHERE a.name = "Sara" AND b.name = "Jorge" CREATE (a) -[r:FRIEND_OF]->(b) SET
r.Friend_since = 2022;
MATCH(a:Person), (b:Person) WHERE a.name = "Keanu" AND b.name = "James" AND b.lastname = "Miller" CREATE (a)
-[r:FRIEND_OF]->(b)  SET r.Friend_since = 1970;
MATCH(a:Person), (b:Person) WHERE a.name = "Monica" AND b.name = "Alejandro" CREATE (a) -[r:FRIEND_OF]->(b)
SET r.Friend_since = 2019;
MATCH(a:Person), (b:Person) WHERE a.name = "Jorge" AND b.name = "Daniel" CREATE (a) -[r:FRIEND_OF]->(b) SET
r.Friend_since = 1996;
MATCH(a:Person), (b:Person) WHERE a.name = "Alejandro" AND b.name = "Teresa" CREATE (a) -[r:FRIEND_OF]->(b)
SET r.Friend_since = 2020;
MATCH(a:Person), (b:Person) WHERE a.name = "Melania" AND b.name = "Sara" CREATE (a) -[r:FRIEND_OF]->(b) SET
r.Friend_since = 2022;
MATCH(a:Person), (b:Person) WHERE a.name = "Teresa" AND b.name = "Sara" CREATE (a) -[r:FRIEND_OF]->(b) SET
r.Friend_since = 2000;
MATCH(a:Person), (b:Person) WHERE a.name = "Mery" AND b.name = "Daniel" CREATE (a) -[r:FRIEND_OF]->(b) SET
r.Friend_since = 2003;
MATCH(a:Person), (b:Person) WHERE a.name = "James" AND a.lastname = "Johnson" AND b.name = "Daniel" CREATE
(a) -[r:FRIEND_OF]->(b) SET r.Friend_since = 2009;

MATCH(a:Person), (b:Movies) WHERE a.name = "Keanu" AND b.title = "Love Rosie" CREATE (a) -[p:ACTED_IN]->(b)
SET p.Roles = "Main character";
MATCH(a:Person), (b:Movies) WHERE a.name = "Keanu" AND b.title = "The Impossible" CREATE (a)
-[p:ACTED_IN]->(b) SET p.Roles = "Main character" ;
MATCH(a:Person), (b:Movies) WHERE a.name = "Keanu" AND b.title = "Wonder" CREATE (a) -[p:ACTED_IN]->(b) SET
p.Roles = "Secondary Character";
MATCH(a:Person), (b:Movies) WHERE a.name = "James" AND a.lastname = "Miller" AND b.title = "The Matrix"
CREATE (a) -[p:ACTED_IN]->(b) SET p.Roles = "Extra";
MATCH(a:Person), (b:Movies) WHERE a.name = "James" AND a.lastname = "Johnson" AND b.title = "Love Rosie"
CREATE (a) -[p:ACTED_IN]->(b) SET p.Roles = "Side Character";
MATCH(a:Person), (b:Movies) WHERE a.name = "Melania" AND b.title = "Pride and Prejudice" CREATE (a)
-[p:ACTED_IN]->(b) SET p.Roles = "Main Character";
```

```
MATCH(a:Person), (b:Movies) WHERE a.name = "Mery" AND b.title = "Now You See Me" CREATE (a)
-[p:ACTED_IN]->(b) SET p.Roles = "Side Character";

MATCH(a:Person), (b:Movies) WHERE a.name = "Alejandro" AND b.title = "The Matrix" CREATE (a)
-[:DIRECTED]->(b);
MATCH(a:Person), (b:Movies) WHERE a.name = "Monica" AND b.title = "Pride and Prejudice" CREATE (a)
-[:DIRECTED]->(b);
MATCH(a:Person), (b:Movies) WHERE a.name = "Jorge" AND b.title = "Now You See Me" CREATE (a)
-[:DIRECTED]->(b);
MATCH(a:Person), (b:Movies) WHERE a.name = "Daniel" AND b.title = "Wonder" CREATE (a) -[:DIRECTED]->(b);
MATCH(a:Person), (b:Movies) WHERE a.name = "Alejandro" AND b.title = "Love Rosie" CREATE (a)
-[:DIRECTED]->(b);
MATCH(a:Person), (b:Movies) WHERE a.name = "Teresa" AND b.title = "The Impossible" CREATE (a)
-[:DIRECTED]->(b);

MATCH(a:Person), (b:Movies) WHERE a.name = "Alejandro" AND b.title = "The Impossible" CREATE (a)
-[v:RECOMMEND]->(b) SET v.Recommend_to = "James Miller";
MATCH(a:Person), (b:Movies) WHERE a.name = "Alejandro" AND b.title = "Wonder" CREATE (a) -[v:RECOMMEND]->(b)
SET v.Recommend_to = "James Miller";
MATCH(a:Person), (b:Movies) WHERE a.name = "Alejandro" AND b.title = "Wonder" CREATE (a) -[v:RECOMMEND]->(b)
SET v.Recommend_to = "Teresa Ulloa";
MATCH(a:Person), (b:Movies) WHERE a.name = "Alejandro" AND b.title = "Pride and Prejudice" CREATE (a)
-[v:RECOMMEND]->(b) SET v.Recommend_to = "Teresa Ulloa";
MATCH(a:Person), (b:Movies) WHERE a.name = "Melania" AND b.title = "Love Rosie" CREATE (a)
-[v:RECOMMEND]->(b) SET v.Recommend_to = "Alejandro Monge";
MATCH(a:Person), (b:Movies) WHERE a.name = "Melania" AND b.title = "The Matrix" CREATE (a)
-[v:RECOMMEND]->(b) SET v.Recommend_to = "Sara Carvajal";
MATCH(a:Person), (b:Movies) WHERE a.name = "Daniel" AND b.title = "The Matrix" CREATE (a)
-[v:RECOMMEND]->(b) SET v.Recommend_to = "Teresa Ulloa";
MATCH(a:Person), (b:Movies) WHERE a.name = "Sara" AND b.title = "Now You See Me" CREATE (a)
-[v:RECOMMEND]->(b) SET v.Recommend_to = "Jorge Garcia";
MATCH(a:Person), (b:Movies) WHERE a.name = "Teresa" AND b.title = "Pride and Prejudice" CREATE (a)
-[v:RECOMMEND]->(b) SET v.Recommend_to = "Sara Carvajal";
```

→ Database information:



Node labels

*(17) Movies Person

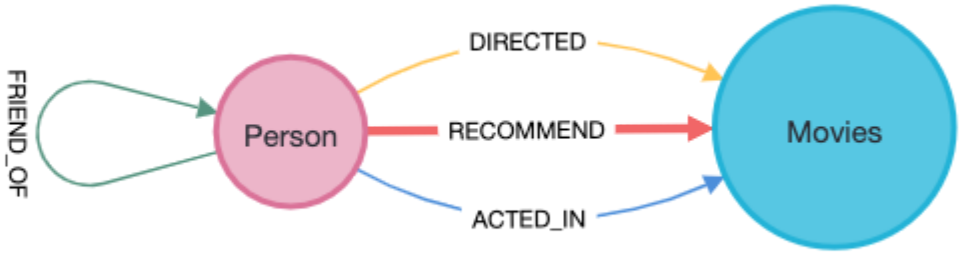
Relationship types

*(35) ACTED_IN DIRECTED FRIEND_OF RECOMMEND

Display Schema/Graph

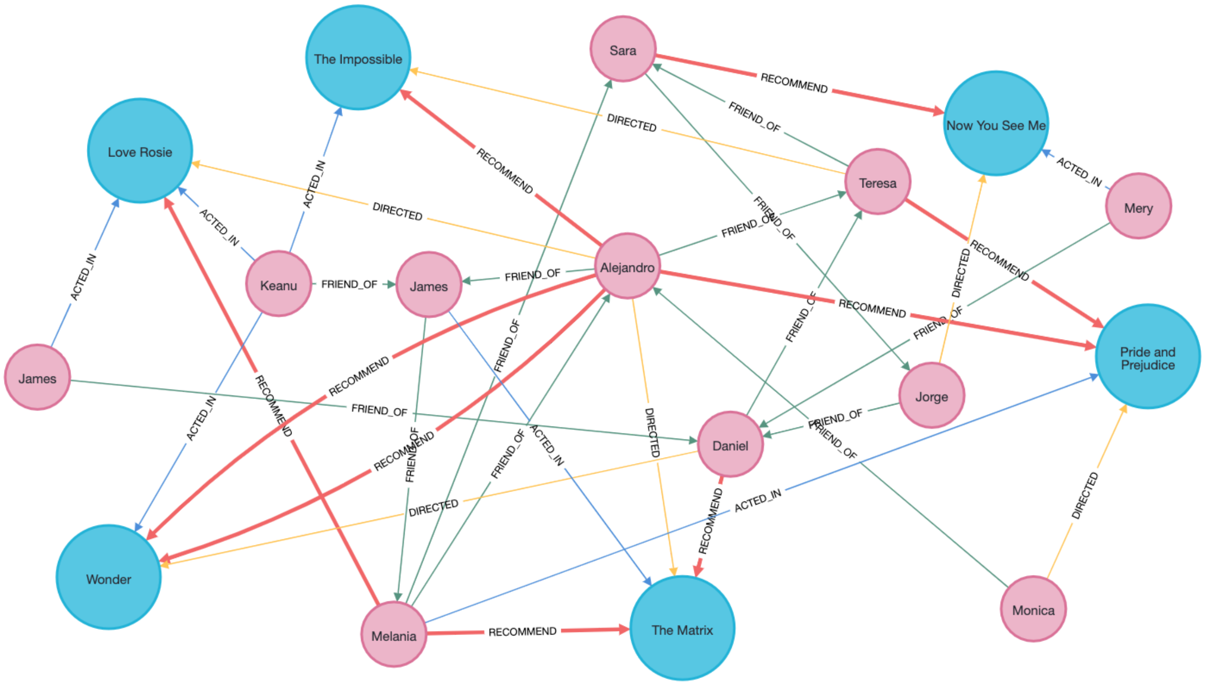
2.2. Write a query to display the schema of your database.

```
CALL db.schema.visualization();
```



2.3. Recover all nodes and relationships (display graph)

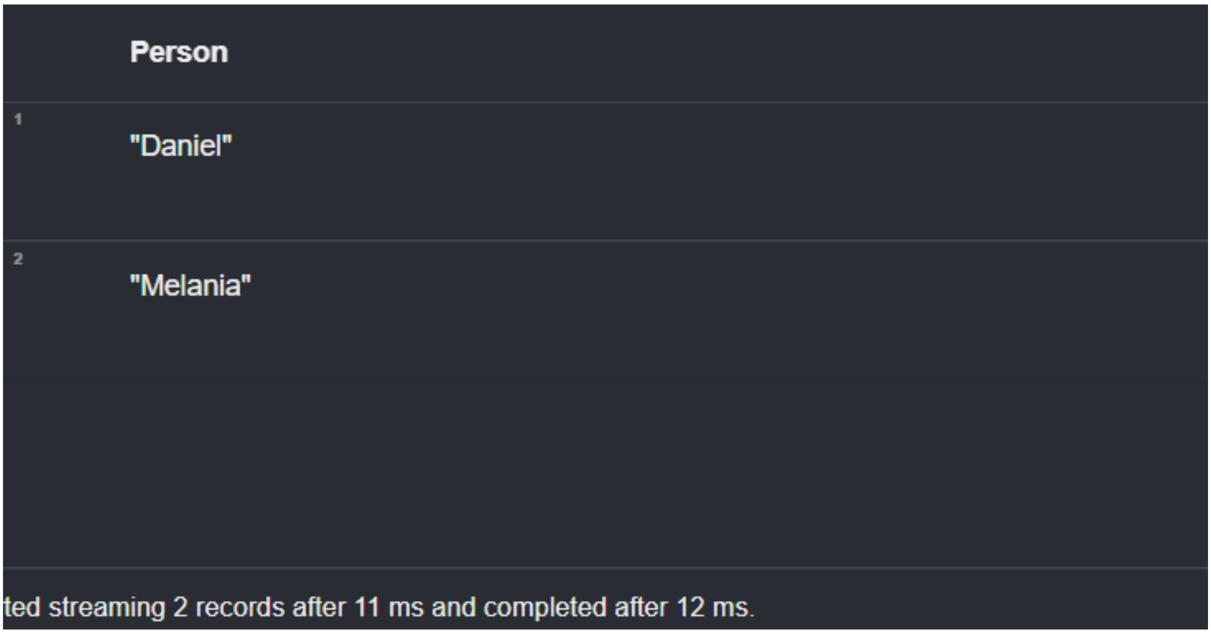
```
MATCH(n) RETURN n;
```



Query

2.4. Recover all the people on the network who recommend the writer "The Matrix".

```
MATCH(p:Person)-[:RECOMMEND]->(m:Movies) WHERE m.title = "The Matrix" RETURN
p.name AS Person;
```

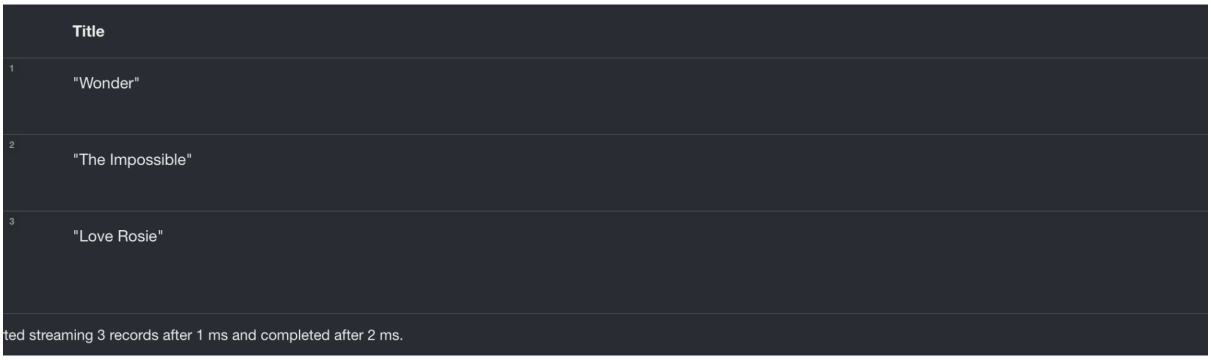


	Person
1	"Daniel"
2	"Melania"

ted streaming 2 records after 11 ms and completed after 12 ms.

2.5. Retrieve all movies that "Keanu Reeves" acted in and return their titles.

```
MATCH (p:Person {name: "Keanu", lastname: "Reeves"})-[:ACTED_IN]-(m:Movies)
RETURN m.title AS Title;
```

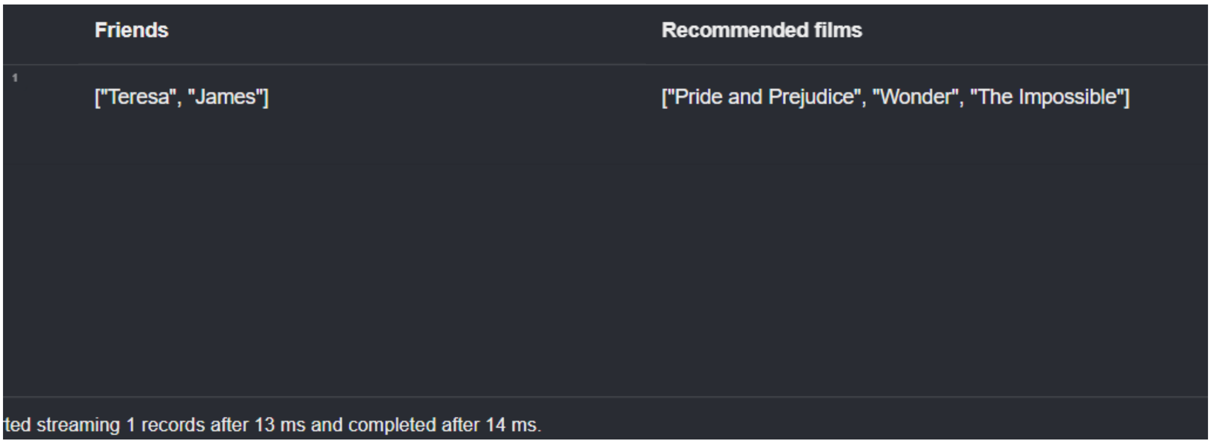


	Title
1	"Wonder"
2	"The Impossible"
3	"Love Rosie"

ted streaming 3 records after 1 ms and completed after 2 ms.

2.6. Return all "Alejandro" friends and what movies he has recommended.

```
MATCH
(a:Person)<-[:FRIEND_OF]-(b:Person{name:"Alejandro"})-[:RECOMMEND]->(m:Movies)
RETURN collect(DISTINCT a.name) AS `Friends`, collect(DISTINCT m.title) AS
`Recommended films`;
```

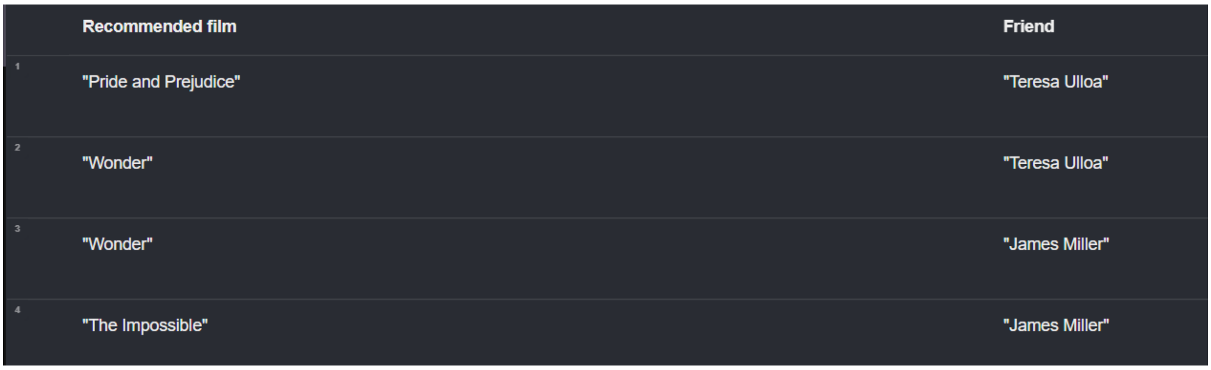


	Friends	Recommended films
1	["Teresa", "James"]	["Pride and Prejudice", "Wonder", "The Impossible"]

ted streaming 1 records after 13 ms and completed after 14 ms.

- In case it was also desired to know specifically what movie was recommended to each of Alejandro's friends, the following query would be done:

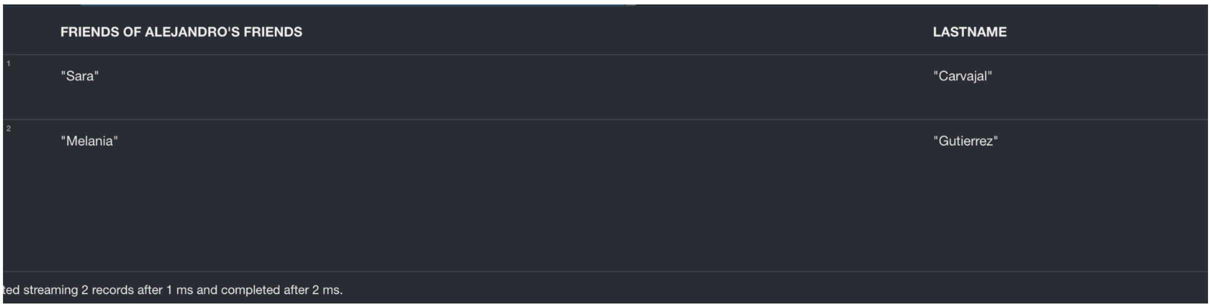
```
MATCH (a:Person{name: "Alejandro"})-[r:RECOMMEND]->(m:Movies) return m.title AS
`Recommended film`,r.Recommend_to AS `Friend`;
```



	Recommended film	Friend
1	"Pride and Prejudice"	"Teresa Ulloa"
2	"Wonder"	"Teresa Ulloa"
3	"Wonder"	"James Miller"
4	"The Impossible"	"James Miller"

2.7. List the friends of "Alejandro's" friends.

```
MATCH (p:Person{name:
"Alejandro"})-[:FRIEND_OF]->(f:Person)-[:FRIEND_OF]->(a:Person) RETURN a.name AS
`FRIENDS OF ALEJANDRO'S FRIENDS`, a.lastname AS LASTNAME;
```

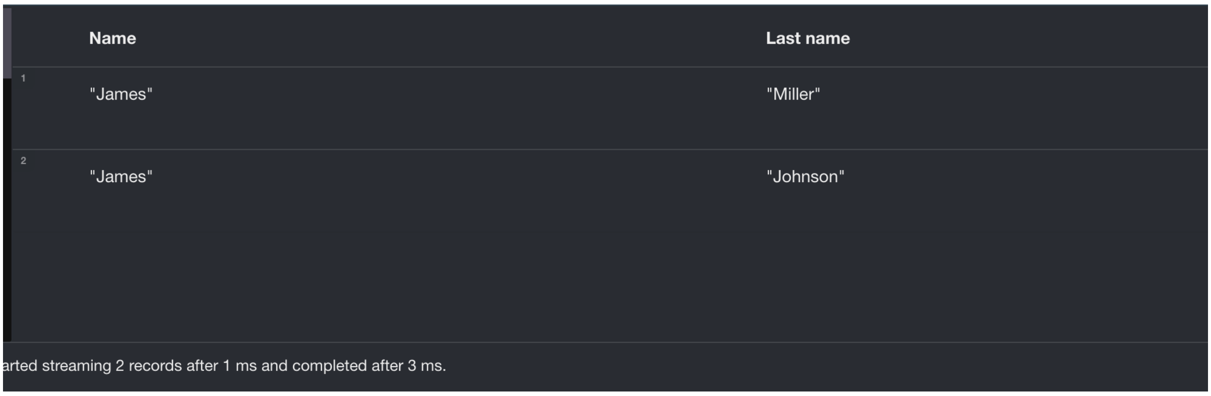


	FRIENDS OF ALEJANDRO'S FRIENDS	LASTNAME
1	"Sara"	"Carvajal"
2	"Melania"	"Gutierrez"

ted streaming 2 records after 1 ms and completed after 2 ms.

2.8. Retrieve all actors whose name begins with "James", returning their names.

```
MATCH(p:Person {name: "James"})-[:ACTED_IN]->(m:Movies) RETURN p.name AS Name,
p.lastname AS `Last name`;
```

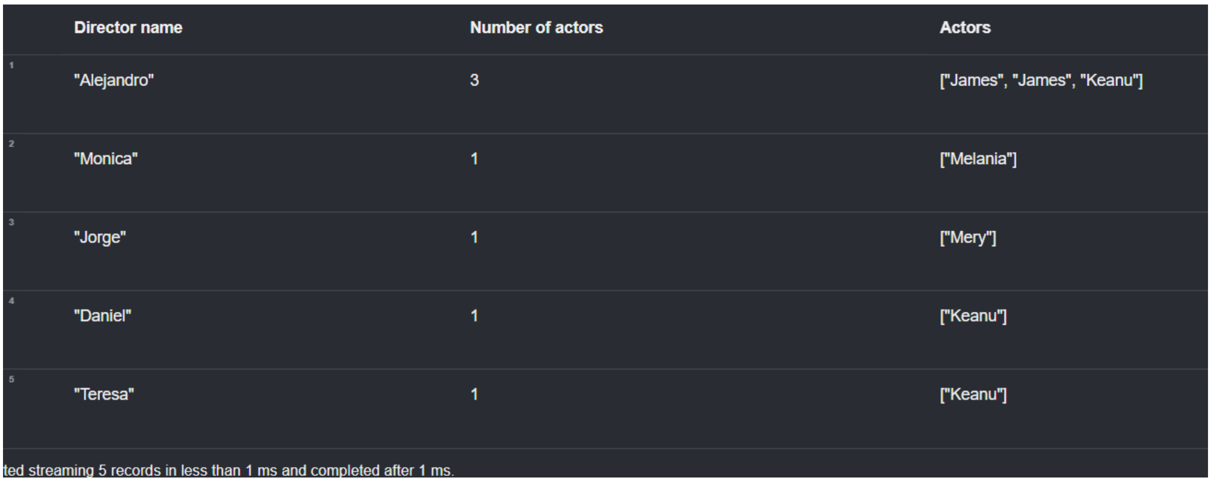


	Name	Last name
1	"James"	"Miller"
2	"James"	"Johnson"

arted streaming 2 records after 1 ms and completed after 3 ms.

2.9. Retrieve all directors, their movies, and people who acted in the movies, returning the name of the director, the number of actors the director has worked with, and the list of actors.

```
MATCH(p:Person)-[:DIRECTED]->(m:Movies)<-[:ACTED_IN]-(actor:Person) RETURN
p.name AS `Director name`, count(actor.name) AS `Number of actors`, collect(actor.name) AS
Actors;
```



	Director name	Number of actors	Actors
1	"Alejandro"	3	["James", "James", "Keanu"]
2	"Monica"	1	["Melania"]
3	"Jorge"	1	["Mery"]
4	"Daniel"	1	["Keanu"]
5	"Teresa"	1	["Keanu"]

ted streaming 5 records in less than 1 ms and completed after 1 ms.