Class 9 Neo4j

Elements of Databases

Oct 27, 2023

Reminders

- Next class: guest lecture on Zoom
- In two classes: Midterm 2

Midterm 2

- When: Fri 11/10 at 2pm
- Where: WEL 1.316
- Duration: 90 minutes
- How: On Paper
- Format:
  - T/F section (~12 questions)
  - MC section (~12 questions)
  - Coding section (~5 questions)
- Review session: Tues 11/11 from 3pm - 4pm on Zoom
- Practice Exam: Will be shared on Ed prior to the review session

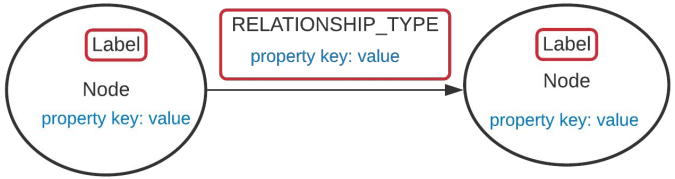
Exam Rules:

- Closed book
- No electronic devices
- 1 cheat sheet

Neo4j Overview

+ Labeled property graph database
+ Suited for highly connected data
+ Declarative, SQL-inspired query language (Cypher)
+ Open-source, sponsored by Neo4j Inc.
+ Rich plugin and extension language (similar to Postgres)
+ ACID-compliant transactions
+ Distributed architecture for scaling reads
+ Visualization tools (Neo4j Browser, Bloom)
+ Optimized for graph traversals
+ Available as a cloud offering (Aura)
- Limited scalability for writes (no sharding)

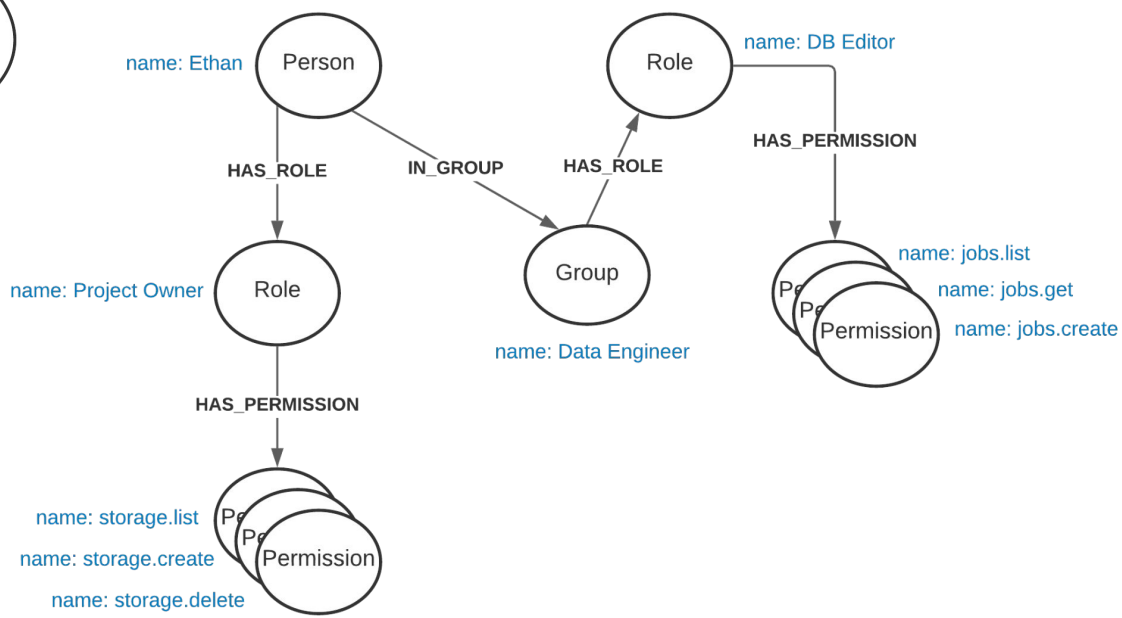
Components of a labeled property graph



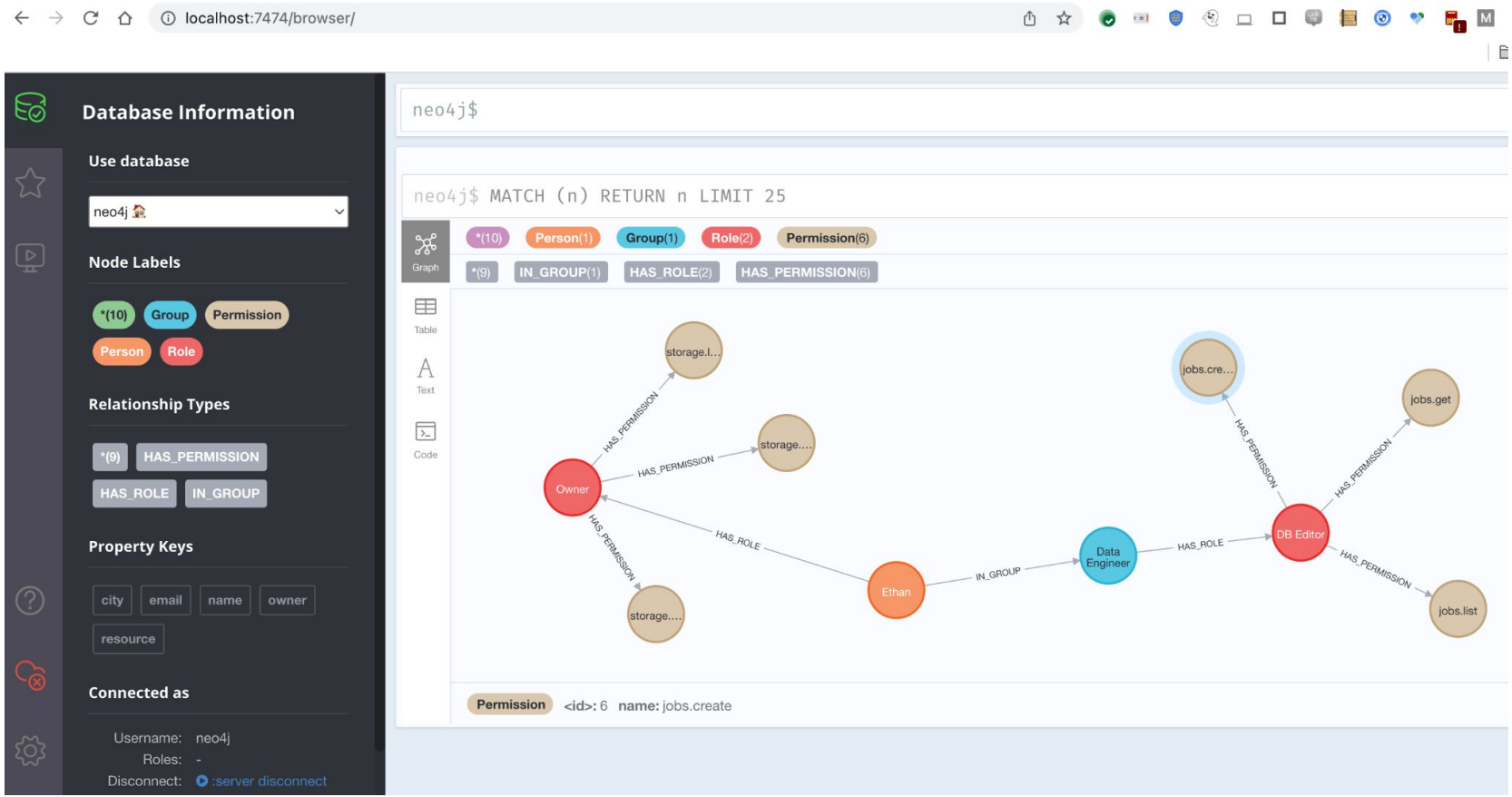
Nodes represent the objects and can be labeled.

Relationships relate nodes by type and direction.

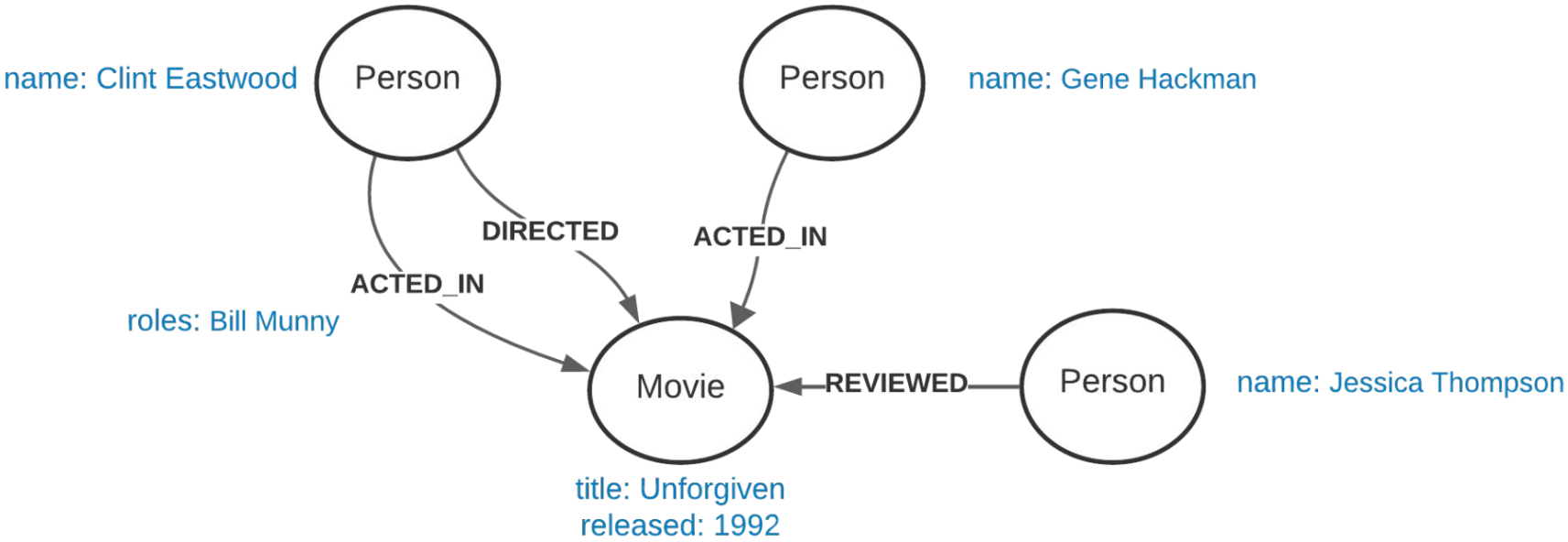
Properties are key, value pairs which can be attached to nodes and to relationships.



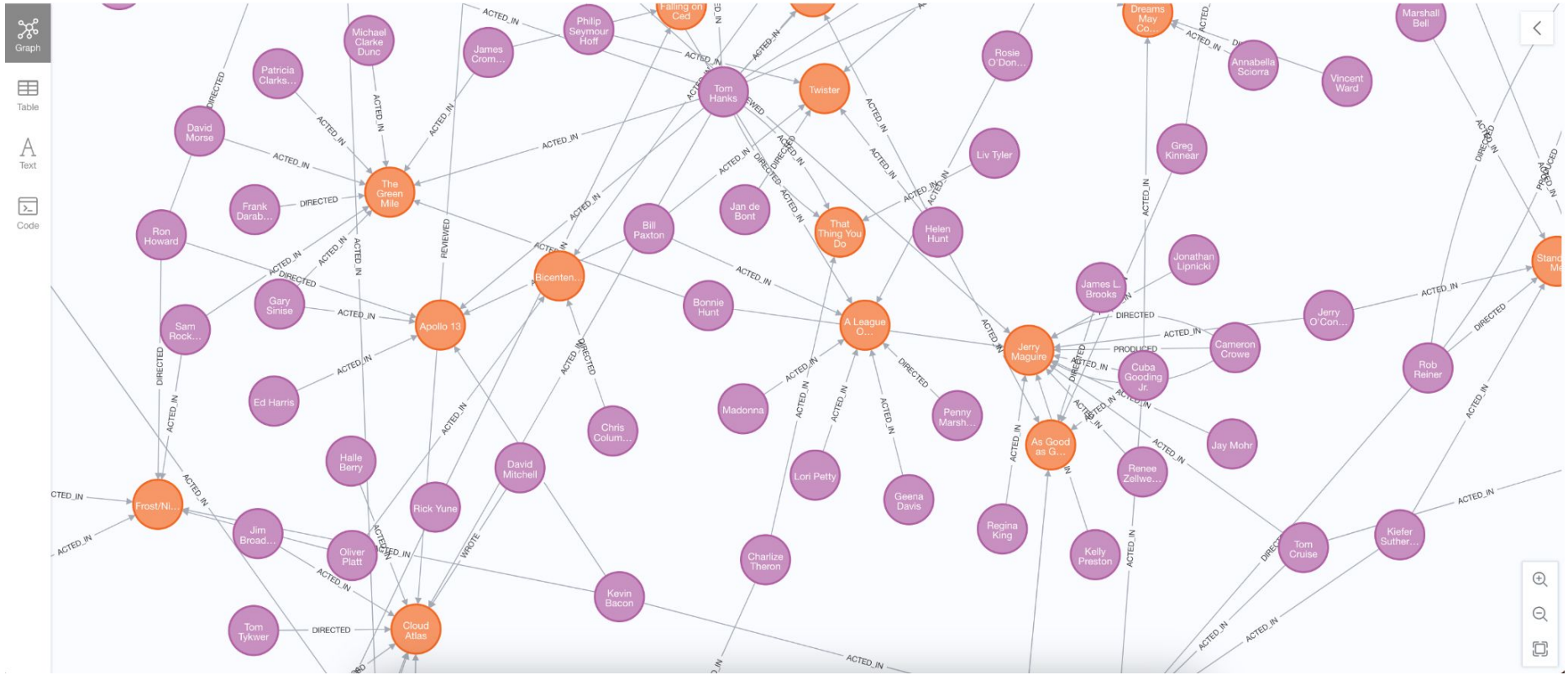
Visualizing the IAM graph



Another example label property graph



Visualizing the movie graph



Querying the graph with Cypher


```
MATCH (p:Person)-[ACTED_IN]→(m:Movie)
WHERE p.name = 'Gene Hackman'
RETURN p, m
```

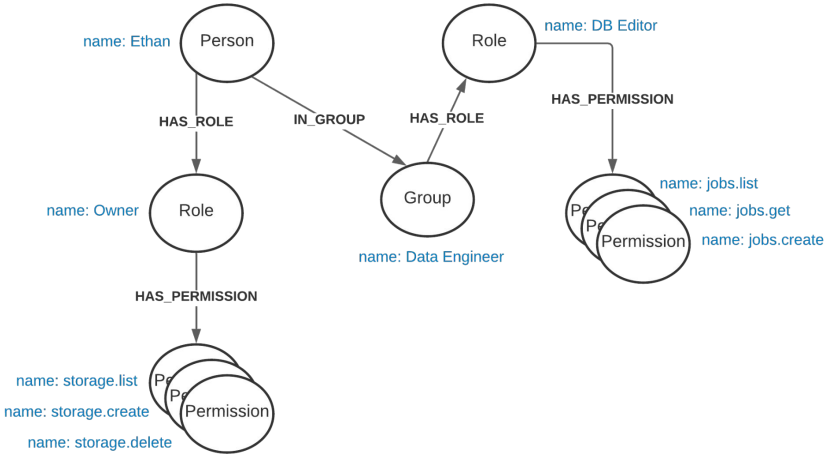
Cypher Code Lab

- Open Neo4j Browser and write some queries over the movie graph
- Switch to JupyterLab and clone the snippets repo
- Open neo4j.ipynb
- Run through the notebook cells and answer first two prompts

Variable length paths

```
12 MATCH (p:Person)-[r*1]->(m:Permission)
13 WHERE p.name = "Ethan"
14 RETURN r, m.name
15 ORDER BY m;
```

```
+------------+
| r | m.name |
+------------+
+------------+
```

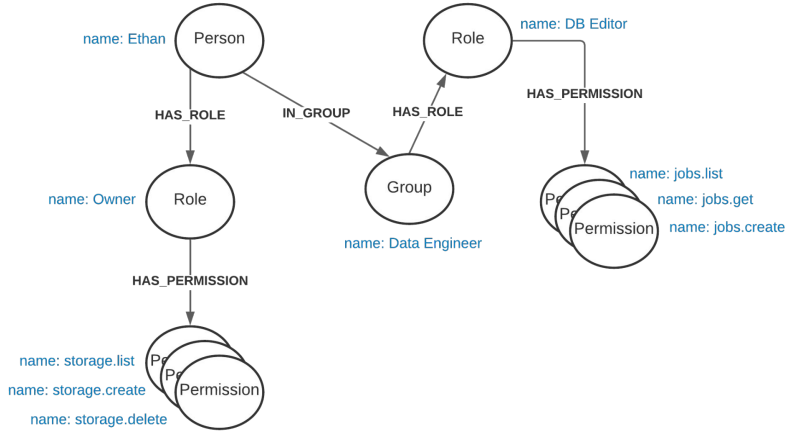



```
17 MATCH (p:Person)-[r*1..2]->(m:Permission)
18 WHERE p.name = "Ethan"
19 RETURN r, m.name
20 ORDER BY m;
```

```
+-----------------------------------------------------+
| r                                   | m.name        |
+-----------------------------------------------------+
| [[:HAS_ROLE], [:HAS_PERMISSION]] | "storage.list"   |
| [[:HAS_ROLE], [:HAS_PERMISSION]] | "storage.create" |
| [[:HAS_ROLE], [:HAS_PERMISSION]] | "storage.delete" |
+-----------------------------------------------------+
```

Variable length paths

```
1  MATCH (p:Person)-[r*]->(m:Permission)
2  WHERE p.name = "Ethan"
3  RETURN r, m.name
4  ORDER BY m;
```




```
+-------------------------------------------------------------------+
| r                                                  | m.name       |
+-------------------------------------------------------------------+
| [[:IN_GROUP], [:HAS_ROLE], [:HAS_PERMISSION]] | "jobs.list"      |
| [[:IN_GROUP], [:HAS_ROLE], [:HAS_PERMISSION]] | "jobs.get"       |
| [[:IN_GROUP], [:HAS_ROLE], [:HAS_PERMISSION]] | "jobs.create"    |
| [[:HAS_ROLE], [:HAS_PERMISSION]]              | "storage.list"   |
| [[:HAS_ROLE], [:HAS_PERMISSION]]              | "storage.create" |
| [[:HAS_ROLE], [:HAS_PERMISSION]]              | "storage.delete" |
+-------------------------------------------------------------------+
```

```
6  MATCH (p:Person)-[r*]->(m:Permission)
7  WHERE p.name = "Ethan"
8  WITH distinct m.name as distinct_perms
9  RETURN distinct_perms
10 ORDER BY distinct_perms;
```

```
+------------------+
| distinct_perms   |
+------------------+
| "jobs.create"    |
| "jobs.get"       |
| "jobs.list"      |
| "storage.create" |
| "storage.delete" |
| "storage.list"   |
+------------------+
```

Counting nodes and relationships

```
1  MATCH (n)
2  RETURN count(n);
3
4  MATCH (n)
5  RETURN distinct labels(n), count(n);
6
7  MATCH ()-[r]->()
8  RETURN count(r);
9
10 MATCH ()-[r]->()
11 RETURN type(r), count(r);
12
13 MATCH (n:Person)
14 RETURN count(n);
15
16 MATCH ()-[r:HAS_ROLE]->()
17 RETURN count(r);
```

```
+---------------------------+
| labels(n)       | count(n) |
+---------------------------+
| ["Person"]      | 1        |
| ["Group"]       | 1        |
| ["Role"]        | 2        |
| ["Permission"]  | 6        |
+---------------------------+
```

```
+---------------------------+
| type(r)          | count(r) |
+---------------------------+
| "IN_GROUP"       | 1        |
| "HAS_ROLE"       | 2        |
| "HAS_PERMISSION" | 6        |
+---------------------------+
```

Creating nodes with CREATE

```
1  CREATE (:Person {name: "Ethan", email: "ethan@utexas.edu"});
2  CREATE (:Group {name: "Data Engineer", owner: "Alex"});
3
4  CREATE (:Role {name: "Owner", resource: "Project"});
5  CREATE (:Role {name: "DB Editor", resource: "Cloud SQL"});
6
7  CREATE (:Permission {name: "jobs.list"});
8  CREATE (:Permission {name: "jobs.get"});
9  CREATE (:Permission {name: "jobs.create"});
10
11 CREATE (:Permission {name: "storage.list"});
12 CREATE (:Permission {name: "storage.create"});
13 CREATE (:Permission {name: "storage.delete"});
```

```
+-----------------------------------------------------+
| n                                                   |
+-----------------------------------------------------+
| (:Person {name: "Ethan", email: "ethan@utexas.edu"}) |
| (:Group {owner: "Alex", name: "Data Engineer"})      |
| (:Role {name: "Owner", resource: "Project"})         |
| (:Role {name: "DB Editor", resource: "Cloud SQL"})   |
| (:Permission {name: "jobs.list"})                    |
| (:Permission {name: "jobs.get"})                     |
| (:Permission {name: "jobs.create"})                  |
| (:Permission {name: "storage.list"})                 |
| (:Permission {name: "storage.create"})               |
| (:Permission {name: "storage.delete"})               |
+-----------------------------------------------------+
```

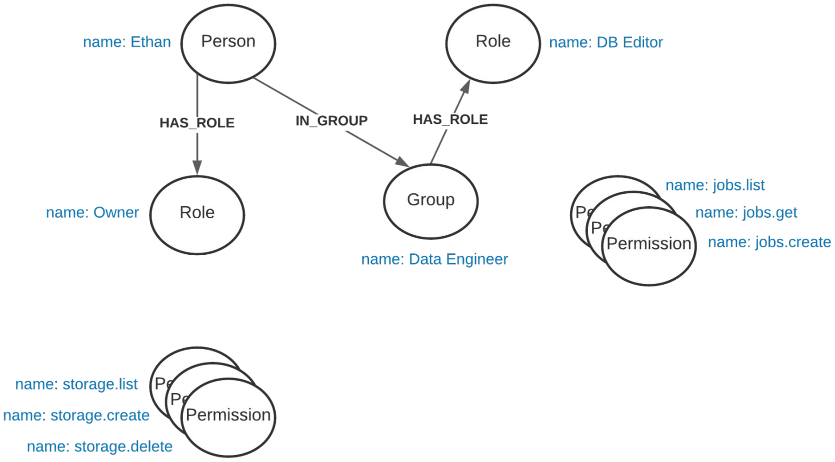
Creating relationships with MATCH

```
MATCH (p:Person {name: "Ethan"})
MATCH (r:Role {name: "Owner"})
CREATE (p)-[:HAS_ROLE]->(r);

MATCH (p:Person {name: "Ethan"})
MATCH (g:Group {name: "Data Engineer"})
CREATE (p)-[:IN_GROUP]->(g);

MATCH (g:Group {name: "Data Engineer"})
MATCH (r:Role {name: "DB Editor"})
CREATE (g)-[:HAS_ROLE]->(r);

MATCH (p)-[h]->(r) RETURN p, h, r;
```



p	h	r
(:Person {name: "Ethan", email: "ethan@utexas.edu"})	[:IN_GROUP]	(:Group {owner: "Alex", name: "Data Engineer"})
(:Person {name: "Ethan", email: "ethan@utexas.edu"})	[:HAS_ROLE]	(:Role {name: "Owner", resource: "Project"})
(:Group {owner: "Alex", name: "Data Engineer"})	[:HAS_ROLE]	(:Role {name: "DB Editor", resource: "Cloud SQL"})

Updating node properties with SET

Adding node properties:

```
1  MATCH (n:Person {name: "Ethan"})
2  SET n.current_employee = True,
3      n.start_date = "2021-06-01"
4  RETURN n.name, n.current_employee, n.start_date;
5
6  MATCH (n:Person {name: "Ethan"})
7  SET n.current_employee = False,
8      n.start_date = "2021-06-01",
9      n.end_date = "2021-08-01"
10 RETURN n.name, n.current_employee, n.start_date, n.end_date;
```

```
+-----------------------------------------+
| n.name  | n.current_employee | n.start_date |
+-----------------------------------------+
| "Ethan" | TRUE               | "2021-06-01" |
+-----------------------------------------+
```

```
+-------------------------------------------------------------+
| n.name  | n.current_employee | n.start_date | n.end_date    |
+-------------------------------------------------------------+
| "Ethan" | FALSE              | "2021-06-01" | "2021-08-01" |
+-------------------------------------------------------------+
```

Adding node labels:

```
12 MATCH (n {name: "Ethan"})
13 SET n:Principal
14 RETURN n.name, labels(n) AS labels;
```

```
+-----------------------------------+
| n.name  | labels                  |
+-----------------------------------+
| "Ethan" | ["Person", "Principal"] |
+-----------------------------------+
```

Updating relationships with MERGE

Adding and updating relationship properties:

```
16 MATCH (n:Role {name: "DB Editor"})
17 MATCH (p:Permission {name: "jobs.create"})
18 MERGE (n)-[r:HAS_PERMISSION]->(p)
19 ON MATCH SET r.start_time = "08:00", r.end_time = "17:00"
20 RETURN n.name, type(r), r.start_time, r.end_time;
```

```
+-----------------------------------------------------------------+
| n.name      | type(r)          | r.start_time | r.end_time |
+-----------------------------------------------------------------+
| "DB Editor" | "HAS_PERMISSION" | "08:00"      | "17:00"    |
+-----------------------------------------------------------------+
```

"Renaming" a relationship type:

```
22 MATCH (n:Role)-[rel:HAS_PERMISSION]->(p:Permission)
23 MERGE (n)-[:HAS_IAM_PERMISSION]->(p)
24 DELETE rel;
25
26 MATCH (r:Role)-[h:HAS_IAM_PERMISSION]->(p:Permission)
27 RETURN r, h, p;
```

```
+-------------------------------------------------------------------------------------------------------------------+
| r                                                | h                     | p                                             |
+-------------------------------------------------------------------------------------------------------------------+
| (:Role {name: "Owner", resource: "Project"})     | [:HAS_IAM_PERMISSION] | (:Permission {name: "storage.list"})          |
| (:Role {name: "Owner", resource: "Project"})     | [:HAS_IAM_PERMISSION] | (:Permission {name: "storage.create"})        |
| (:Role {name: "Owner", resource: "Project"})     | [:HAS_IAM_PERMISSION] | (:Permission {name: "storage.delete"})        |
| (:Role {name: "DB Editor", resource: "Cloud SQL"}) | [:HAS_IAM_PERMISSION] | (:Permission {name: "jobs.list"})           |
| (:Role {name: "DB Editor", resource: "Cloud SQL"}) | [:HAS_IAM_PERMISSION] | (:Permission {name: "jobs.get"})            |
| (:Role {name: "DB Editor", resource: "Cloud SQL"}) | [:HAS_IAM_PERMISSION] | (:Permission {name: "jobs.create"})         |
+-------------------------------------------------------------------------------------------------------------------+
```

Deleting relationships and nodes with DELETE

Drop the relationships connected to nodes labeled Person:

```
1 MATCH (p:Person)-[r]->()
2 DELETE r;
```

Drop nodes labeled Person:

```
4 MATCH (p:Person)
5 DELETE p;
```

Drop all the nodes and relationships in the current database:

```
7 MATCH (n)
8 DETACH DELETE n;
```

```
neo4j@neo4j> MATCH (n)
             DETACH DELETE n;
0 rows available after 7 ms, consumed after another 0 ms
Deleted 10 nodes, Deleted 9 relationships
neo4j@neo4j>
```

Project 7

http://www.cs.utexas.edu/~scohen/projects/project-7.pdf