Cypher Query

Dr. Sirasit Lochanachit

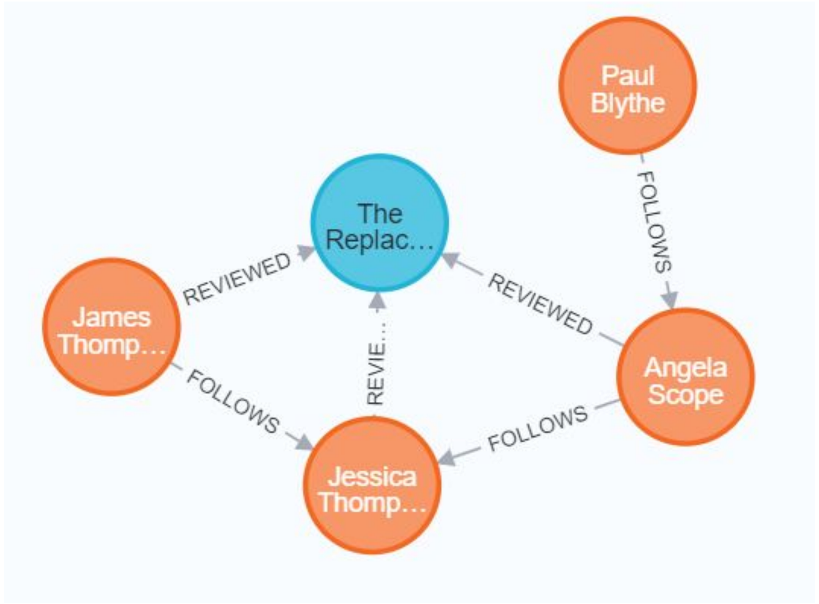
Today's Outline

- Filtering queries
    - Patterns in queries
- Aggregations
- Other data types [Lists, maps, dates]
- Intermediate processing using WITH
- Controlling results returned

Traversal in a MATCH clause

Example

- Find all of the followers of people who reviewed the movie, *The Replacements*. Then return names of follower and reviewer.

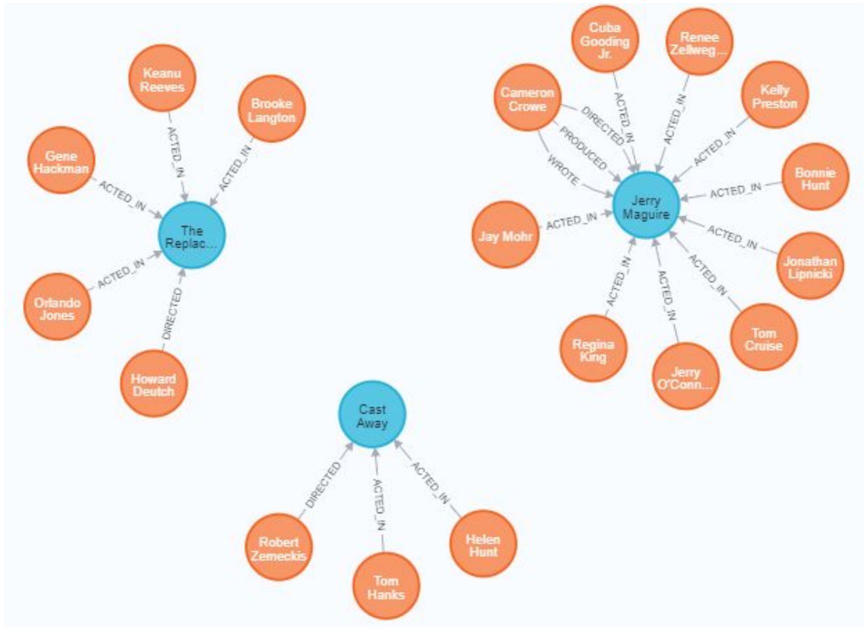


```
MATCH (follower:Person) -[:FOLLOWS]-> (reviewer:Person) -[:REVIEWED]-> (m:Movie)
WHERE m.title = 'The Replacements'
RETURN follower.name, reviewer.name
```

Multiple Patterns in a MATCH

Example

- Find all people who acted in movies released in year 2000 and also retrieve the name of the director.



```
MATCH (a:Person) -[:ACTED_IN]-> (m:Movie), (m) <-[:DIRECTED]- (d:Person)
WHERE m.released = 2000
RETURN a.name, m.title, d.name
```

use m if the arrow is more than 2
because 3 arrows cant point m thats in the middle

Multiple Patterns in a MATCH

Example

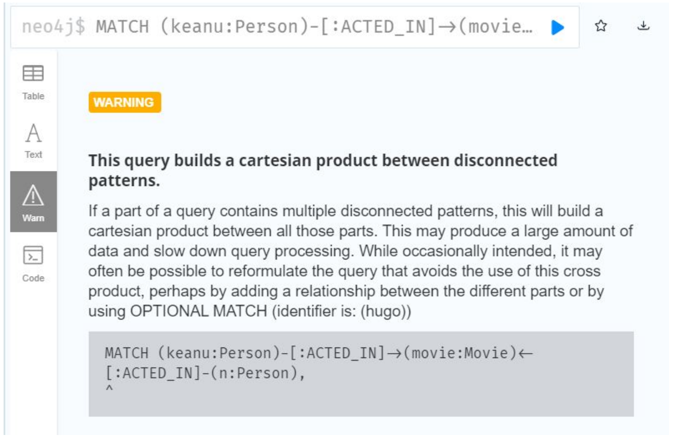
- Find actors that worked with *Keanu Reeves*, but not when *Hugo Weaving* acted in the same movie.

```
MATCH (keanu:Person) -[:ACTED_IN]-> (movie:Movie) <-[:ACTED_IN]- (n:Person),
      (hugo:Person)
WHERE keanu.name='Keanu Reeves' AND hugo.name='Hugo Weaving' AND NOT
      (hugo) -[:ACTED_IN]-> (movie)
RETURN n.name
```

Multiple Patterns in a MATCH

Example

- Find actors that worked with *Keanu Reeves*, but not when *Hugo Weaving* acted in the same movie.



```
neo4j$ MATCH (keanu:Person)-[:ACTED_IN]→(movie…
```

WARNING

This query builds a cartesian product between disconnected patterns.

If a part of a query contains multiple disconnected patterns, this will build a cartesian product between all those parts. This may produce a large amount of data and slow down query processing. While occasionally intended, it may often be possible to reformulate the query that avoids the use of this cross product, perhaps by adding a relationship between the different parts or by using OPTIONAL MATCH (identifier is: (hugo))

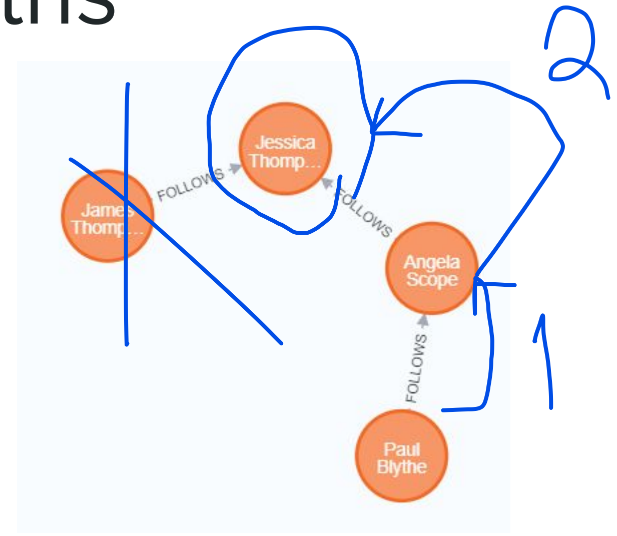
```
MATCH (keanu:Person)-[:ACTED_IN]→(movie:Movie)←
[:ACTED_IN]-(n:Person),
^
```

```
MATCH (keanu:Person) -[:ACTED_IN]-> (movie:Movie) <-[:ACTED_IN]- (n:Person),
      (hugo:Person)
WHERE keanu.name='Keanu Reeves' AND hugo.name='Hugo Weaving' AND NOT
      (hugo) -[:ACTED_IN]-> (movie)
RETURN n.name
```

Specifying Varying Length Paths

Example

- Retrieve all *Person* nodes that are 2 hops away in :FOLLOWS path from *Paul Blythe*



```
MATCH (follower:Person) -[:FOLLOWS*2]-> (p:Person)
WHERE follower.name = 'Paul Blythe'
RETURN p.name
```

Specifying Varying Length Paths

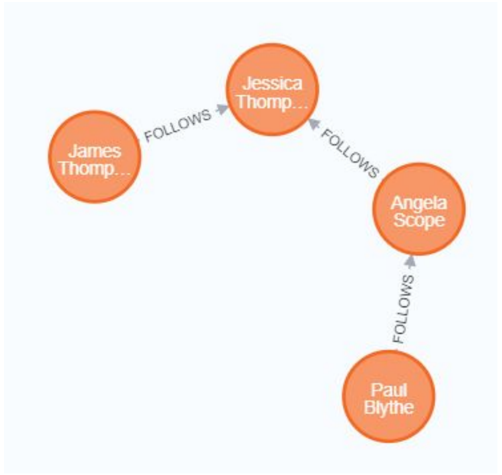
(nodeA) - [:RELTYPE*] -> (nodeB)

means "all"

(nodeA) - [:RELTYPE*] - (nodeB)

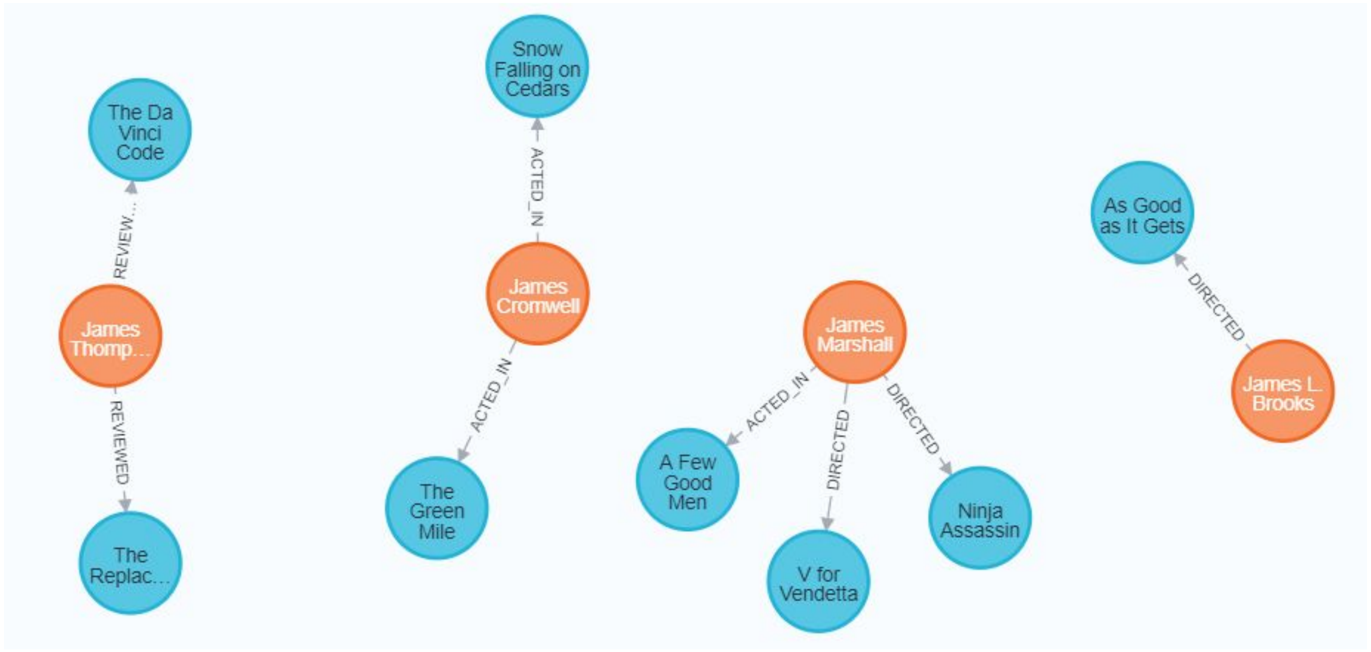
(nodeA) - [:RELTYPE*3] -> (nodeB)

(nodeA) - [:RELTYPE*1..3] -> (nodeB)



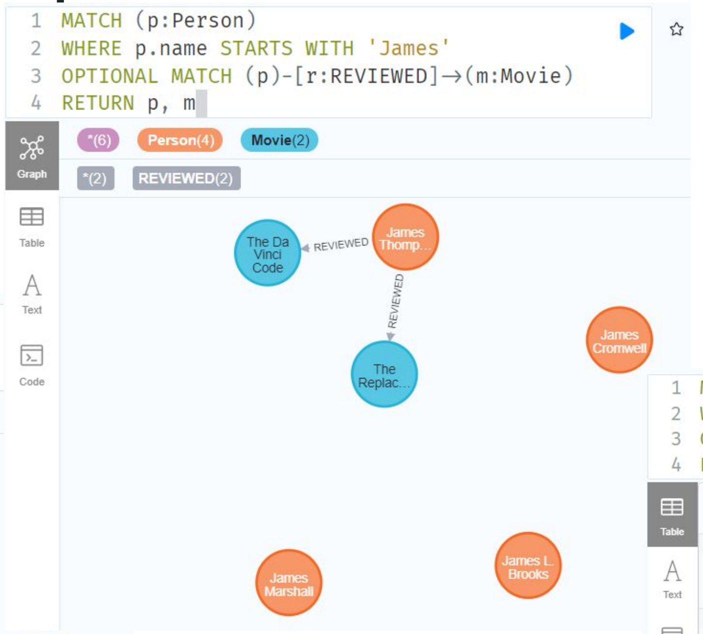
Specifying Optional Pattern Matching

OPTIONAL MATCH

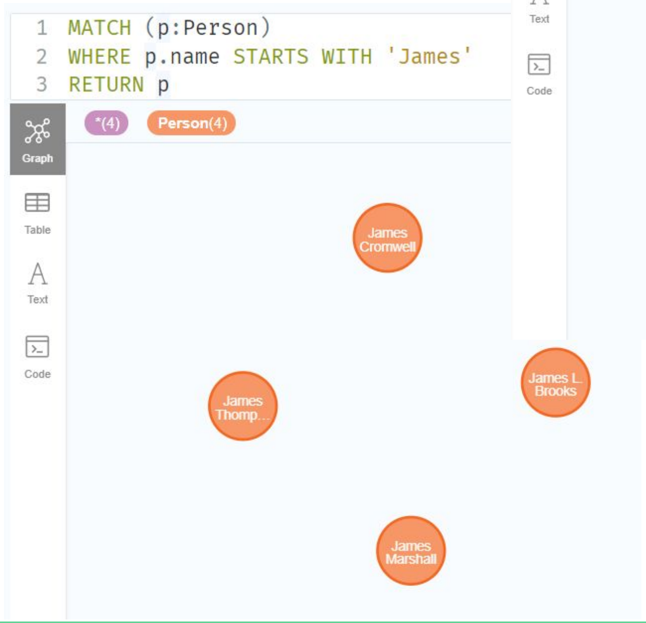


Specifying Optional Pattern Matching

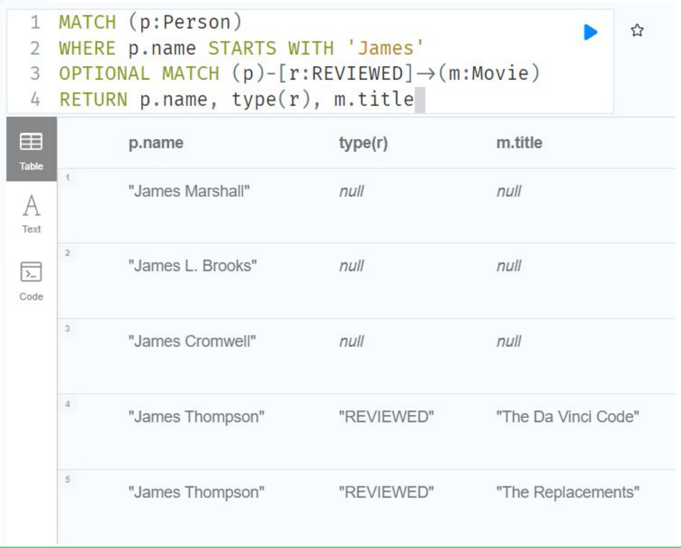
OPTIONAL MATCH



```
MATCH (p:Person)
WHERE p.name STARTS WITH 'James'
OPTIONAL MATCH (p)-[r:REVIEWED]→(m:Movie)
RETURN p, m
```



```
MATCH (p:Person)
WHERE p.name STARTS WITH 'James'
RETURN p
```

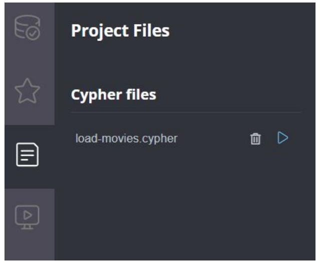


```
MATCH (p:Person)
WHERE p.name STARTS WITH 'James'
OPTIONAL MATCH (p)-[r:REVIEWED]→(m:Movie)
RETURN p.name, type(r), m.title
```

	p.name	type(r)	m.title
1	"James Marshall"	*null*	*null*
2	"James L. Brooks"	*null*	*null*
3	"James Cromwell"	*null*	*null*
4	"James Thompson"	"REVIEWED"	"The Da Vinci Code"
5	"James Thompson"	"REVIEWED"	"The Replacements"

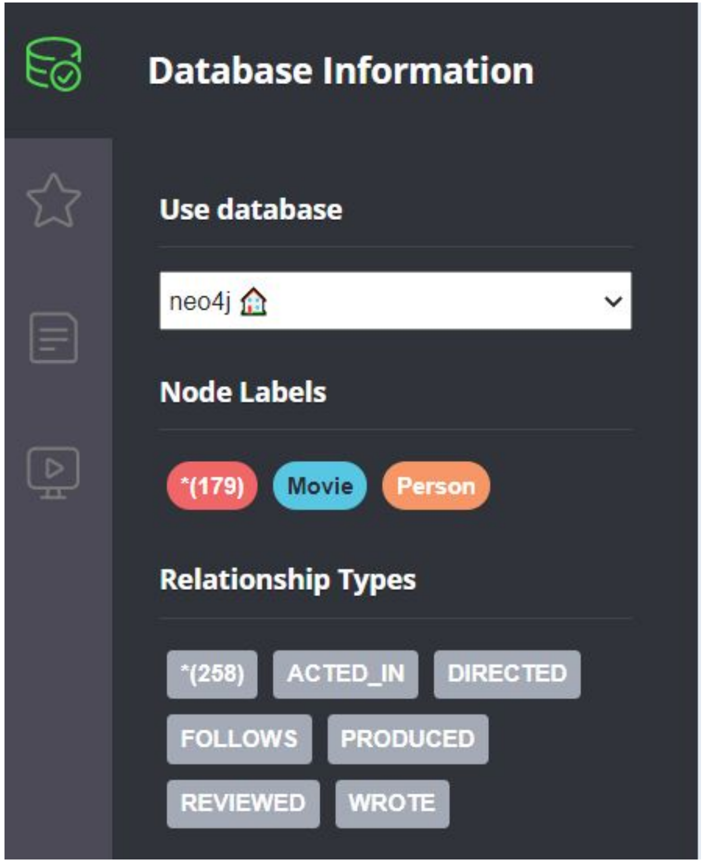
Exercise: Preparations

- Movie DB
  - 171 nodes (169 in Lab)
  - 253 relationships (250 in Lab)
- If not, try
  - MATCH (n) WHERE (n:Person OR n:Movie) DETACH DELETE n
  - Then run load-movies.cypher


Project Files

Cypher files

load-movies.cypher


Database Information

Use database

neo4j

Node Labels

*(179) Movie Person

Relationship Types

*(258) ACTED_IN DIRECTED FOLLOWS PRODUCED REVIEWED WROTE

Exercise 5: Patterns in Queries

I) Write a query to retrieve all movies that *Gene Hackman* has acted in, along with the directors of the movies. In addition, retrieve the actors that acted in the same movies as *Gene Hackman*.

- Return the name of the movie as ‘movie’, the name of the director as ‘director’, and the names of actors that worked with *Gene Hackman* as ‘co-actors’.
- *Gene Hackman should not appear in ‘co-actors’.*

Exercise 5: Patterns in Queries

Note: 2) to 5) should show results as a graph with *James Thompson* node.

2) Retrieve all nodes that the person named *James Thompson* directly has the *FOLLOWS* relationship in either direction.

3) Modify 2) query to retrieve nodes that are exactly 3 hops away.

4) Modify 3) query to retrieve nodes that are 1 and 2 hops away.

5) Modify 4) query to retrieve all nodes that are connected to *James* by a *Follows* relationship no matter how many hops.

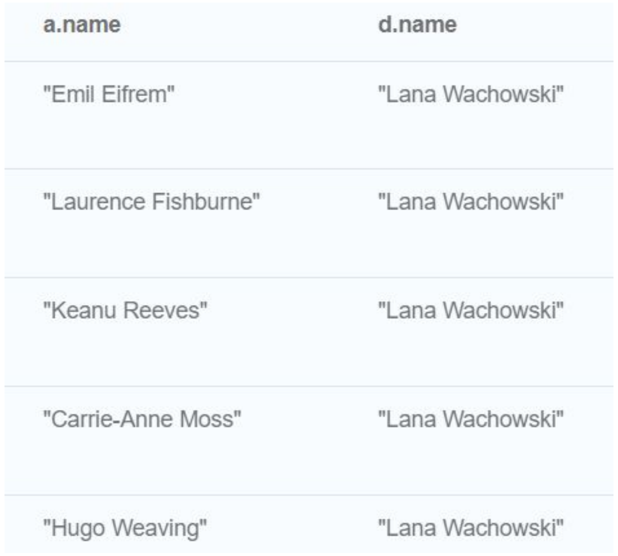
6) Return all persons whose name begins with *Tom* and optionally return the name of a movie that this person directed.

count()

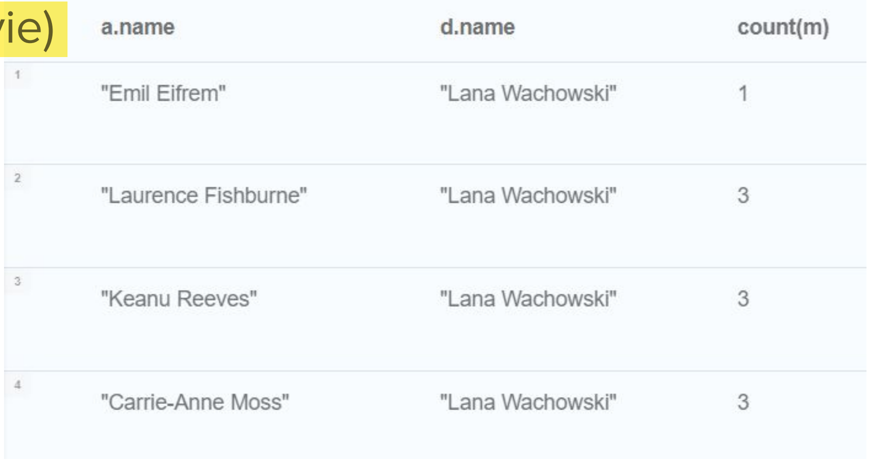
Example

- Count of all movies for a particular actor/director pair

```
MATCH (a:Person) -[:ACTED_IN]-> (m:Movie)
      <-[:DIRECTED]- (d:Person)
RETURN a.name, d.name, count(m)
```



a.name	d.name
"Emil Eifrem"	"Lana Wachowski"
"Laurence Fishburne"	"Lana Wachowski"
"Keanu Reeves"	"Lana Wachowski"
"Carrie-Anne Moss"	"Lana Wachowski"
"Hugo Weaving"	"Lana Wachowski"



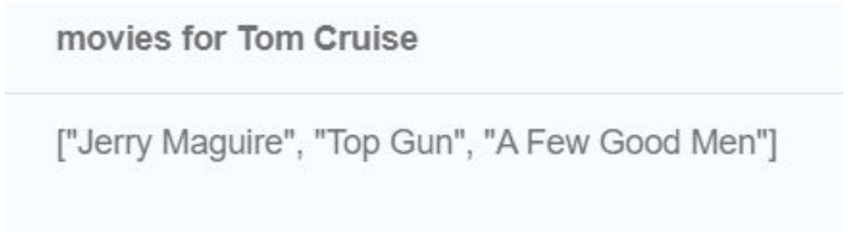
	a.name	d.name	count(m)
1	"Emil Eifrem"	"Lana Wachowski"	1
2	"Laurence Fishburne"	"Lana Wachowski"	3
3	"Keanu Reeves"	"Lana Wachowski"	3
4	"Carrie-Anne Moss"	"Lana Wachowski"	3

collect()

Example

- Collect the list of movie titles that *Tom Cruise* acted in

```
MATCH (p:Person) -[:ACTED_IN]-> (m:Movie)
WHERE p.name = 'Tom Cruise'
RETURN collect(m.title) AS `movies for Tom Cruise`
```



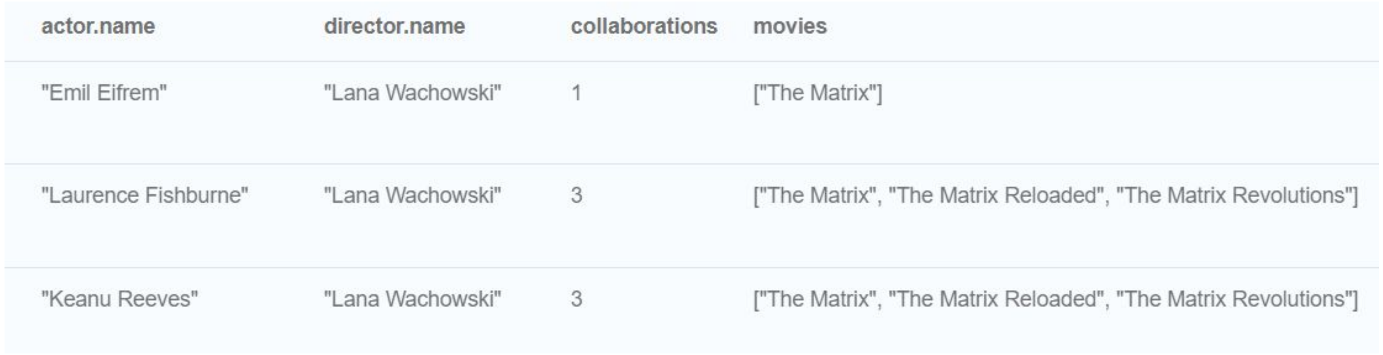
```
movies for Tom Cruise

["Jerry Maguire", "Top Gun", "A Few Good Men"]
```

count() and collect()

count(n)

count(*)



actor.name	director.name	collaborations	movies
"Emil Eifrem"	"Lana Wachowski"	1	["The Matrix"]
"Laurence Fishburne"	"Lana Wachowski"	3	["The Matrix", "The Matrix Reloaded", "The Matrix Revolutions"]
"Keanu Reeves"	"Lana Wachowski"	3	["The Matrix", "The Matrix Reloaded", "The Matrix Revolutions"]

Example

- Count the number of paths retrieved where an actor and director collaborated in a movie.

```
MATCH (actor:Person) -[:ACTED_IN]-> (m:Movie) <-[:DIRECTED]- (director:Person)
RETURN actor.name, director.name, count(m) AS collaborations,
    collect(m.title) AS movies
```

Aggregating functions

avg() - Numeric values

avg() - Durations

collect()

count()

max()

min()

percentileCont()

percentileDisc()

stDev()

stDevP()

sum() - Numeric values

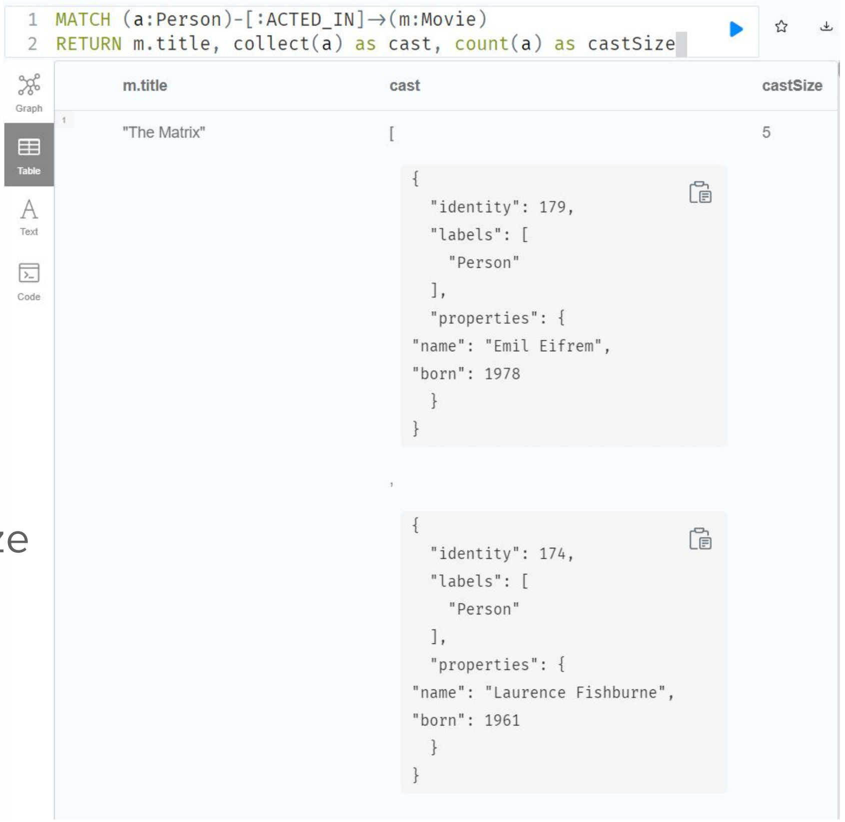
sum() - Durations

Working with Cypher Data

- Lists
- Maps
- Dates

MATCH (a:Person) -[:ACTED_IN]-> (m:Movie)

RETURN m.title, collect(a) as cast, count(a) as castSize



```
MATCH (a:Person)-[:ACTED_IN]→(m:Movie)
RETURN m.title, collect(a) as cast, count(a) as castSize
```

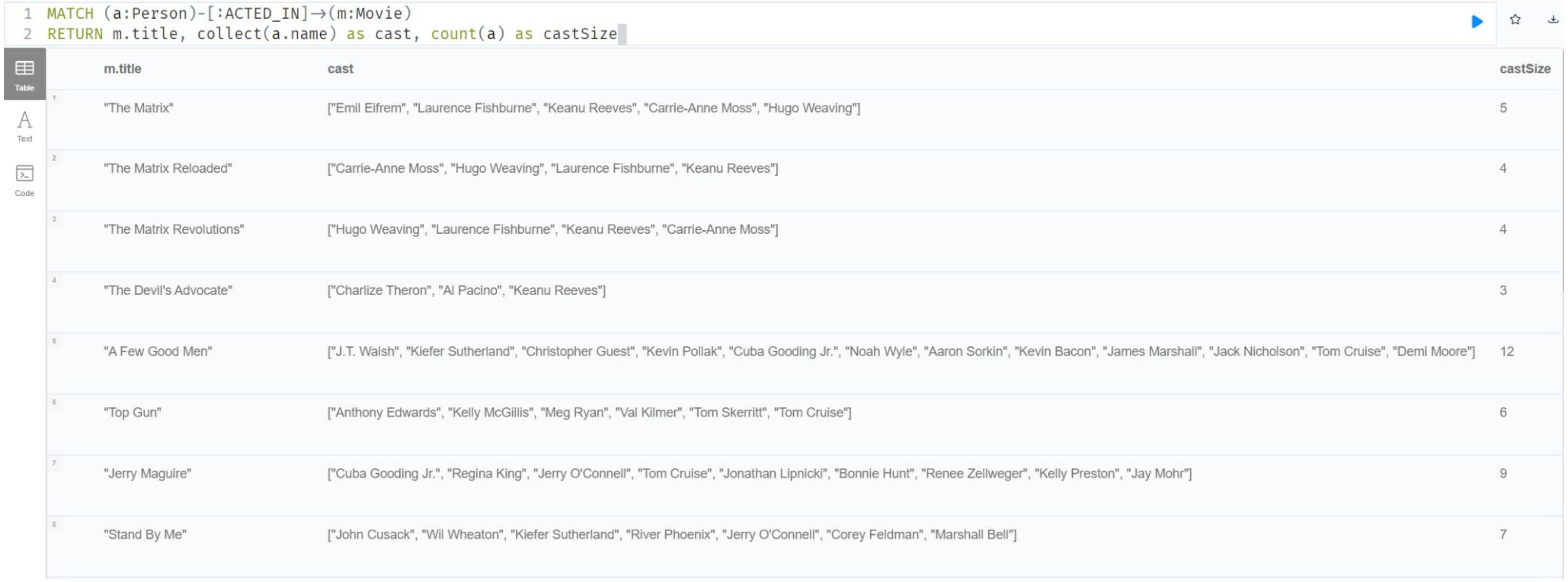
m.title	cast	castSize
"The Matrix"	[{ "identity": 179, "labels": ["Person"], "properties": { "name": "Emil Eifrem", "born": 1978 } } , { "identity": 174, "labels": ["Person"], "properties": { "name": "Laurence Fishburne", "born": 1961 } }	5

Dr. Sirasit Lochanachit

Lists

MATCH (a:Person) -[:ACTED_IN]-> (m:Movie)

RETURN m.title, collect(a.name) as cast, count(a) as castSize



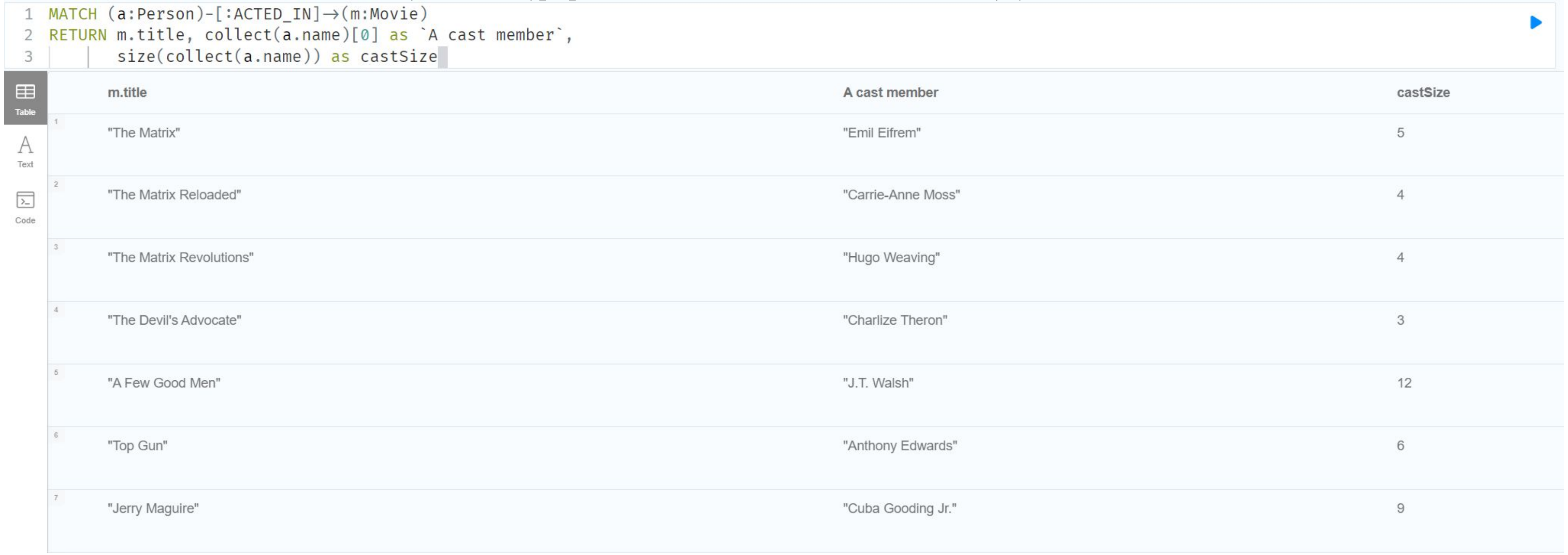
```
MATCH (a:Person)-[:ACTED_IN]→(m:Movie)
RETURN m.title, collect(a.name) as cast, count(a) as castSize
```

	m.title	cast	castSize
1	"The Matrix"	["Emil Eifrem", "Laurence Fishburne", "Keanu Reeves", "Carrie-Anne Moss", "Hugo Weaving"]	5
2	"The Matrix Reloaded"	["Carrie-Anne Moss", "Hugo Weaving", "Laurence Fishburne", "Keanu Reeves"]	4
3	"The Matrix Revolutions"	["Hugo Weaving", "Laurence Fishburne", "Keanu Reeves", "Carrie-Anne Moss"]	4
4	"The Devil's Advocate"	["Charlize Theron", "Al Pacino", "Keanu Reeves"]	3
5	"A Few Good Men"	["J.T. Walsh", "Kiefer Sutherland", "Christopher Guest", "Kevin Pollak", "Cuba Gooding Jr.", "Noah Wyle", "Aaron Sorkin", "Kevin Bacon", "James Marshall", "Jack Nicholson", "Tom Cruise", "Demi Moore"]	12
6	"Top Gun"	["Anthony Edwards", "Kelly McGillis", "Meg Ryan", "Val Kilmer", "Tom Skerritt", "Tom Cruise"]	6
7	"Jerry Maguire"	["Cuba Gooding Jr.", "Regina King", "Jerry O'Connell", "Tom Cruise", "Jonathan Lipnicki", "Bonnie Hunt", "Renee Zellweger", "Kelly Preston", "Jay Mohr"]	9
8	"Stand By Me"	["John Cusack", "Wil Wheaton", "Kiefer Sutherland", "River Phoenix", "Jerry O'Connell", "Corey Feldman", "Marshall Bell"]	7

Accessing elements of the list

MATCH (a:Person) -[:ACTED_IN]-> (m:Movie)

RETURN m.title, collect(a.name)[0] as `A cast member`, count(a) as castSize



```
MATCH (a:Person)-[:ACTED_IN]→(m:Movie)
RETURN m.title, collect(a.name)[0] as `A cast member`,
       size(collect(a.name)) as castSize
```

	m.title	A cast member	castSize
1	"The Matrix"	"Emil Eifrem"	5
2	"The Matrix Reloaded"	"Carrie-Anne Moss"	4
3	"The Matrix Revolutions"	"Hugo Weaving"	4
4	"The Devil's Advocate"	"Charlize Theron"	3
5	"A Few Good Men"	"J.T. Walsh"	12
6	"Top Gun"	"Anthony Edwards"	6
7	"Jerry Maguire"	"Cuba Gooding Jr."	9

Maps

Example

- A map of months and the number of days per month

{Jan: 31, Feb: 28, Mar: 31, Apr: 30 , May: 31, Jun: 30 , Jul: 31, Aug: 31, Sep: 30, Oct: 31, Nov: 30, Dec: 31}

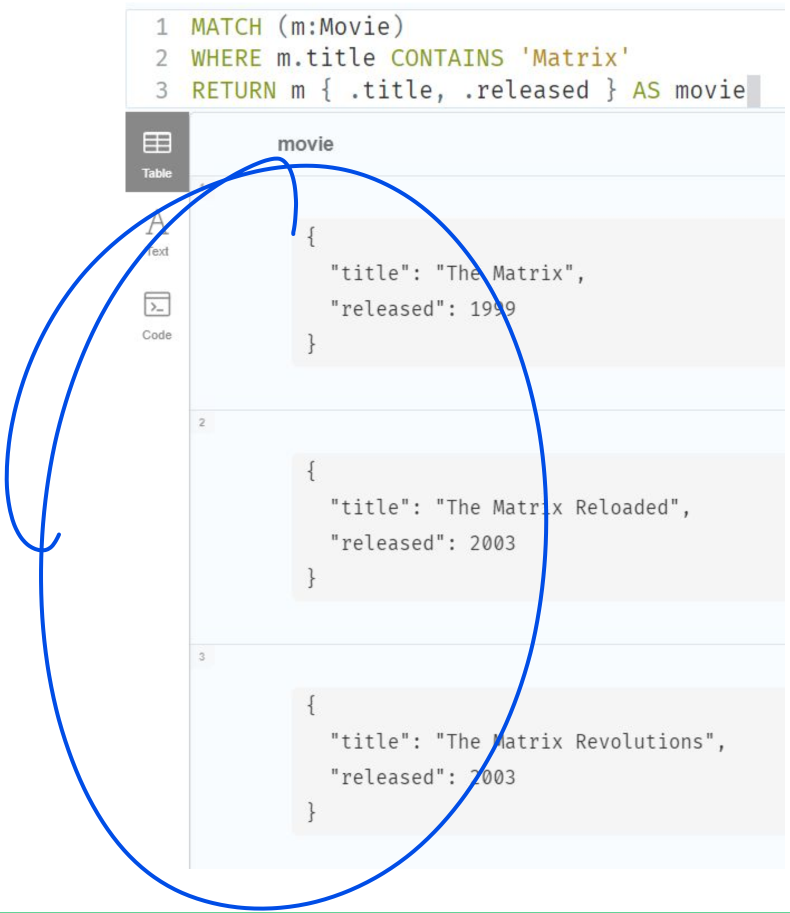
```
RETURN {Jan: 31, Feb: 28, Mar: 31, Apr: 30 , May: 31, Jun: 30 ,
    Jul: 31, Aug: 31, Sep: 30, Oct: 31, Nov: 30, Dec: 31}['Nov'] AS DaysInNov
```

Map Projections

MATCH (m:Movie)

WHERE m.title CONTAINS 'Matrix'

RETURN m { .title, .released } AS movie

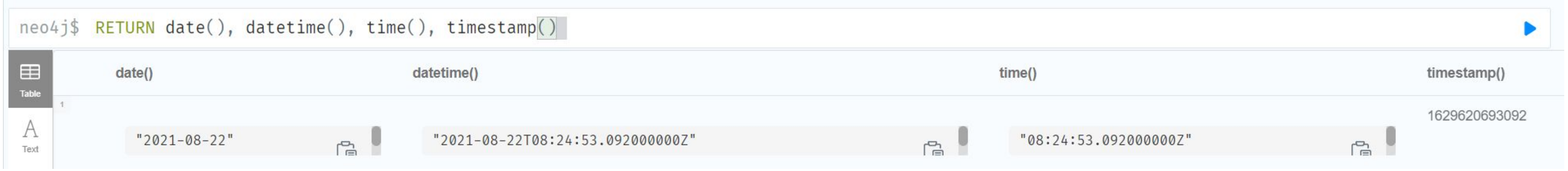


```
MATCH (m:Movie)
WHERE m.title CONTAINS 'Matrix'
RETURN m { .title, .released } AS movie
```

```
1
{
  "title": "The Matrix",
  "released": 1999
}
2
{
  "title": "The Matrix Reloaded",
  "released": 2003
}
3
{
  "title": "The Matrix Revolutions",
  "released": 2003
}
```

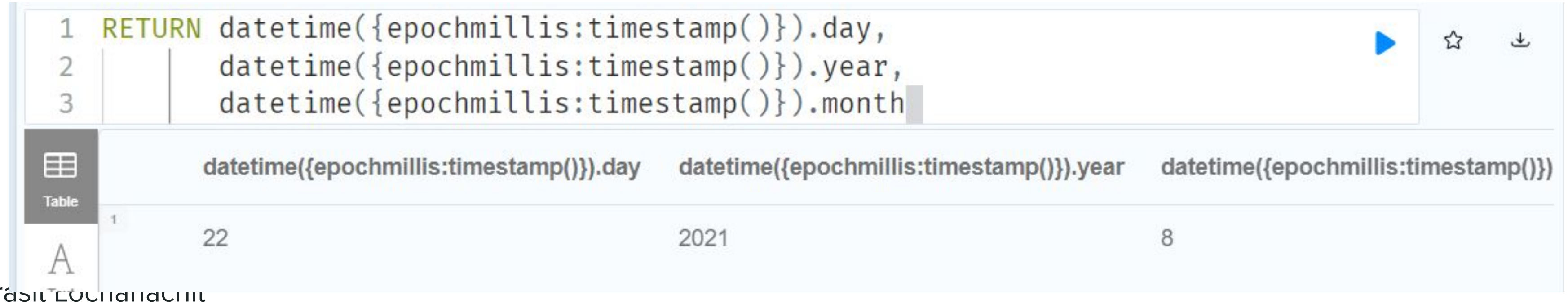
Dates

RETURN date(), datetime(), time(), timestamp()



```
neo4j$ RETURN date(), datetime(), time(), timestamp()
```

date()	datetime()	time()	timestamp()
"2021-08-22"	"2021-08-22T08:24:53.092000000Z"	"08:24:53.092000000Z"	1629620693092



```
RETURN datetime({epochmillis:timestamp()}).day,
       datetime({epochmillis:timestamp()}).year,
       datetime({epochmillis:timestamp()}).month
```

datetime({epochmillis:timestamp()}).day	datetime({epochmillis:timestamp()}).year	datetime({epochmillis:timestamp()})
22	2021	8

Exercise 6: Working with Cypher Data

* Every results must have alias name (e.g. column names)

1) Retrieve actors and the movies they have acted in, returning each actor’s name as ‘actor’ and the **list** of movies they acted in as ‘ movie list’.

2) Retrieve all movies that *Tom Cruise* has acted in and the co-actors that acted in the same movie, returning the movie title as ‘movie’ and the **list** of co-actors that *Tom Cruise* worked with as ‘co-actors’.

3) Retrieve all people who reviewed a movie, returning the name of the movie as ‘movie’, the list of reviewers as ‘reviewers list’, and how many reviewers reviewed the movie as ‘Number of reviews’.

Exercise 6: Working with Cypher Data

4) Retrieve all directors, their movies, and people who acted in the movies, returning the name of the director as ‘director’ , the number of actors the director has worked with as ‘number of actors’, and the list of actors as ‘collab actors’.

5) Return a map of all movies that *Tom Hanks* acted in. The map returned only contains the title of the movie and the year released for the movie as `Tom Hanks’ movie` .

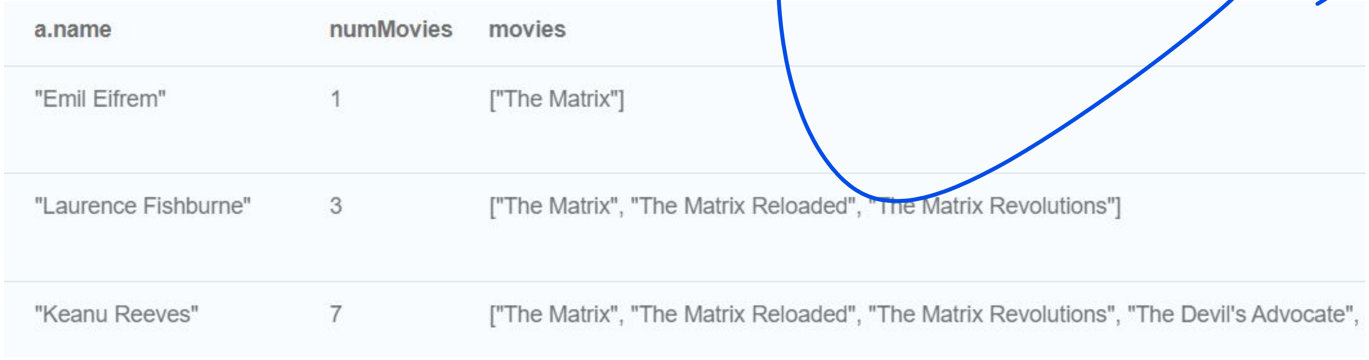
6) Retrieves all movies that *Tom Hanks* acted in, returning the title of the movie, the year the movie was released, the number of years ago that the movie was released as ‘movieAge’, and the age of Tom when the movie was released as `Tom’s Age`.

Intermediate processing using WITH

Example

- Return all actors, the number of movies they acted in, and the titles of the movies.

```
MATCH (a:Person) -[:ACTED_IN]-> (m:Movie)
RETURN a.name, count(m) AS numMovies, collect(m.title) as movies
```



a.name	numMovies	movies
"Emil Eifrem"	1	["The Matrix"]
"Laurence Fishburne"	3	["The Matrix", "The Matrix Reloaded", "The Matrix Revolutions"]
"Keanu Reeves"	7	["The Matrix", "The Matrix Reloaded", "The Matrix Revolutions", "The Devil's Advocate",

Dr. Sirasit Lochanachit

Intermediate processing using WITH

Example

- Return all actors, the number of movies they acted in, and the titles of the movies.
  - Return only actors that have 2 or 3 movies.

```
MATCH (a:Person) -[:ACTED_IN]-> (m:Movie)
WITH  a, count(m) AS numMovies, collect(m.title) as movies
WHERE 1 < numMovies < 4
RETURN a.name, numMovies, movies
```

Example: Subqueries with WITH

- Retrieve all movies reviewed by a person, then return the movie's title, its rating score, reviewer's name and director's name.
    - Director is the second query.

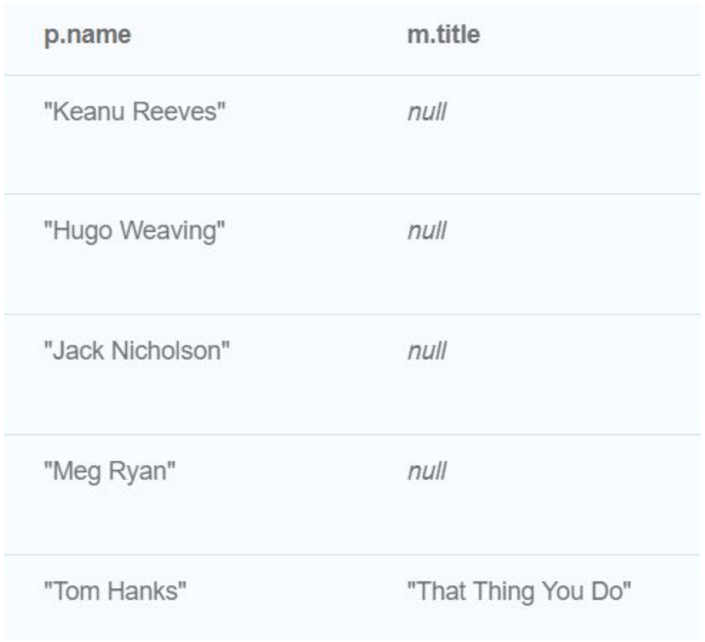
```
MATCH (m:Movie) <-[rv:REVIEWED]- (r:Person)
WITH m, rv, r
MATCH (m) <-[:DIRECTED]- (d:Person)
RETURN m.title, rv.rating, r.name, collect(d.name)
```

Example: Subqueries with WITH

- Find all actors who have acted in at least 5 movies, and find (optionally) the movies they directed and return the person and those movies.

```
MATCH (p:Person)
WITH p, size((p) -[:ACTED_IN]-> ()) AS movies
WHERE movies >= 5
OPTIONAL MATCH (p) -[:DIRECTED]-> (m:Movie)
RETURN p.name, m.title
```

Note: In DBMS v5, writing a query within the **size()** function is not permitted, whereas it works as expected in v4.



p.name	m.title
"Keanu Reeves"	*null*
"Hugo Weaving"	*null*
"Jack Nicholson"	*null*
"Meg Ryan"	*null*
"Tom Hanks"	"That Thing You Do"

Exercise 7: Intermediate processing using WITH

1) Write a Cypher query to list directors and the number of movies they directed. Use the WITH clause to aggregate movie counts per director.

2) Write a Cypher query to find all movies that were directed by more than one person. Use the WITH clause to the number of directors.

3) Retrieve the actors who have acted in exactly 5 movies, returning the name of the actor, and the list of movies (as movies) for that actor.

4) Retrieve all actors that have not appeared in more than 3 movies. Return their names and list of movies (as movies).

Exercise 7: Intermediate processing using WITH

5) Find actors who played more than one role in at least one movie, including the movie title and roles.

6) Find actors who appeared in at least two movies directed by the same director.
    - Return actor's name, director's name, and movie names.

Exercise 7: Subqueries with WITH

7) For each movie, list its title, the names of its actors, and the names of reviewers who reviewed the movie.

8) Retrieve movies with a rating greater than 80, including their directors and actors.

Exercise 7: Subqueries with WITH

9) Retrieve the movies that have at least 2 directors, and optionally the names of people who reviewed the movies.

- Return movie title, number of directors, number of actors, and reviewer names
- In DBMS v5, writing a query within the **size()** function is not permitted, whereas it works as expected in v4.

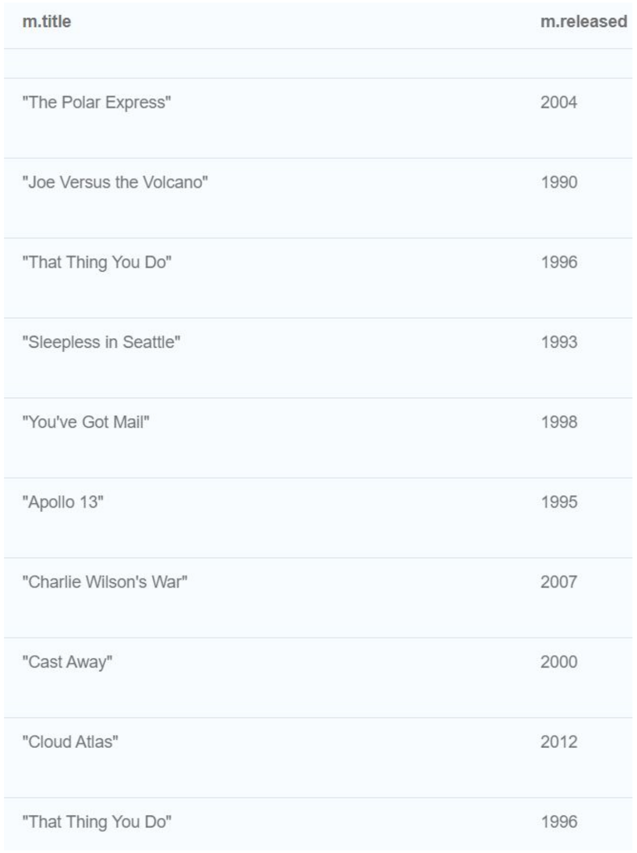
10) Modify 9) to include number of actors

Controlling Results Returned

```
MATCH (p:Person) -[:DIRECTED | ACTED_IN]-> (m:Movie)
WHERE p.name = 'Tom Hanks'
RETURN m.title, m.released
```

Eliminate Duplicate results:

RETURN

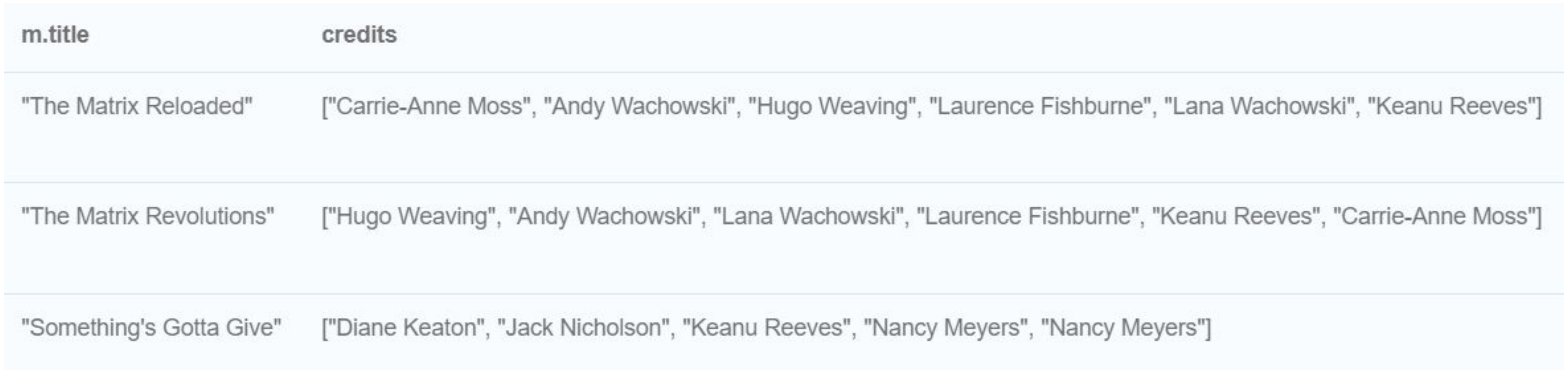


m.title	m.released
"The Polar Express"	2004
"Joe Versus the Volcano"	1990
"That Thing You Do"	1996
"Sleepless in Seattle"	1993
"You've Got Mail"	1998
"Apollo 13"	1995
"Charlie Wilson's War"	2007
"Cast Away"	2000
"Cloud Atlas"	2012
"That Thing You Do"	1996

Duplication in lists

```
MATCH (p:Person) -[:ACTED_IN | DIRECTED | WROTE]-> (m:Movie)
WHERE m.released = 2003
RETURN m.title, collect(p.name) AS credits
```

Eliminate Duplicate results:



m.title	credits
"The Matrix Reloaded"	["Carrie-Anne Moss", "Andy Wachowski", "Hugo Weaving", "Laurence Fishburne", "Lana Wachowski", "Keanu Reeves"]
"The Matrix Revolutions"	["Hugo Weaving", "Andy Wachowski", "Lana Wachowski", "Laurence Fishburne", "Keanu Reeves", "Carrie-Anne Moss"]
"Something's Gotta Give"	["Diane Keaton", "Jack Nicholson", "Keanu Reeves", "Nancy Meyers", "Nancy Meyers"]

WITH and DISTINCT Example

```
MATCH (p:Person) -[:DIRECTED | ACTED_IN]-> (m:Movie)

WHERE p.name = 'Tom Hanks'

WITH DISTINCT m

RETURN m.released, m.title
```

Ordering Results

Example

- Sorting Tom Hanks and Keanu Reeves movies by year released.

Ordering Single Results:

```
MATCH (p:Person) -[:DIRECTED | ACTED_IN]-> (m:Movie)
WHERE p.name = 'Tom Hanks' OR p.name = 'Keanu Reeves'
RETURN DISTINCT m.title, m.released ORDER BY m.released DESC
```

Ordering Multiple Results:

```
MATCH (p:Person) -[:DIRECTED | ACTED_IN]-> (m:Movie)
WHERE p.name = 'Tom Hanks' OR p.name = 'Keanu Reeves'
RETURN DISTINCT m.title, m.released ORDER BY m.released DESC , m.title
```

Limiting the Number of Results

Example

- Return 10 most recently released movies.

```
MATCH (m:Movie)
RETURN m.title as title, m.released as year ORDER BY m.released DESC LIMIT 10
```

- Limit the number of actors collected by 6.

```
MATCH (p:Person) -[:ACTED_IN]-> (m:Movie)
WITH m, p LIMIT 6
RETURN collect(p.name), m.title
```

Exercise 8: Controlling Results Returned

1) Retrieve all actors that acted in movies during the 1990s, where you return the released date, the movie title, and the collected actor names for the movie.

2) Modify 1) query so that the **released date** records returned are not duplicated.

no duplicate

3) Modify 2) query so that there is no duplication of the movies listed for a year.

4) Modify 3) query to order the results returned so that the more recent years are displayed first. arrange year first

5) Retrieve the top 5 ratings and their associated movies, returning the movie title and the rating.

Exercise 8: Controlling Results Returned

calculate average rating

6) Retrieve the top 5 movies based on average rating, returning the movie title and the average rating. Order the results returned so that the higher rating movies are displayed first.