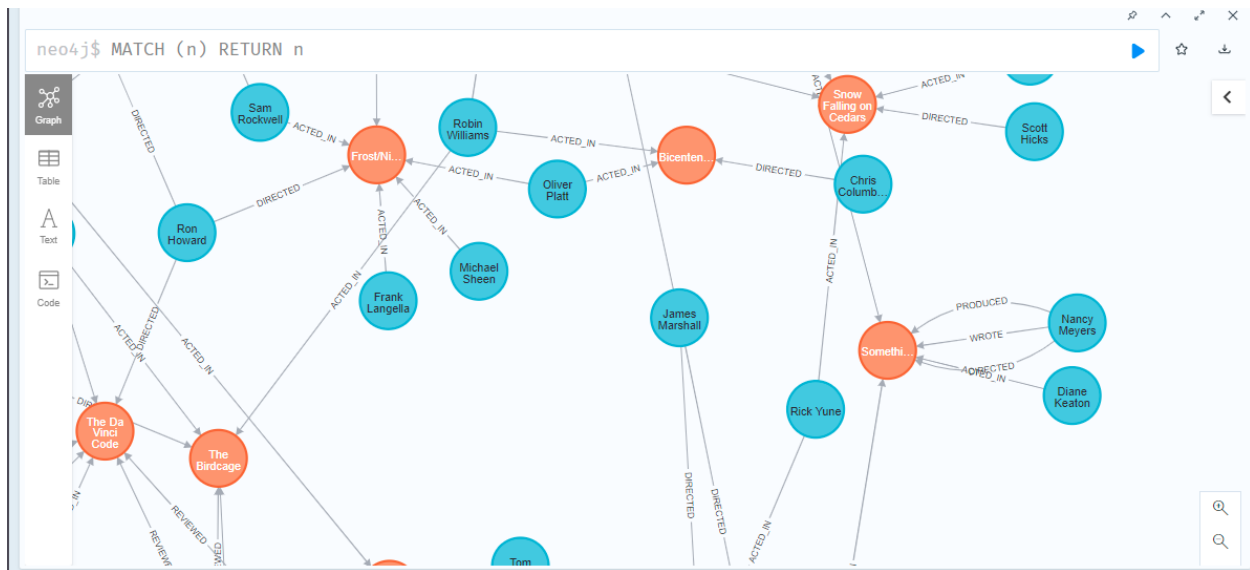




The #1 Database for Connected Data

## 2 Prise en main de la base Movies

### 1. Executer la requete suivante : MATCH (n) RETURN n



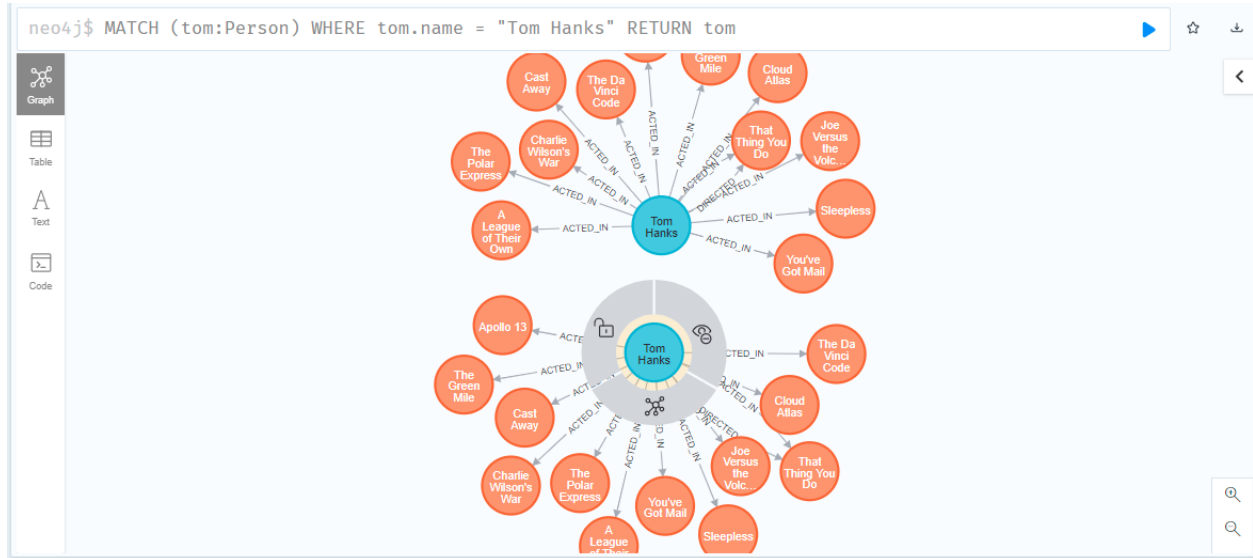
### 4. Extraire le modèle de données. MATCH (start)-[r]->(end) RETURN distinct type(r), labels(start) as startNode, labels(end) as endtNode.

neo4j\$ MATCH (start)-[r]->(end) RETURN distinct type(r), labels(start) as startNode, labels(end) as endtNode

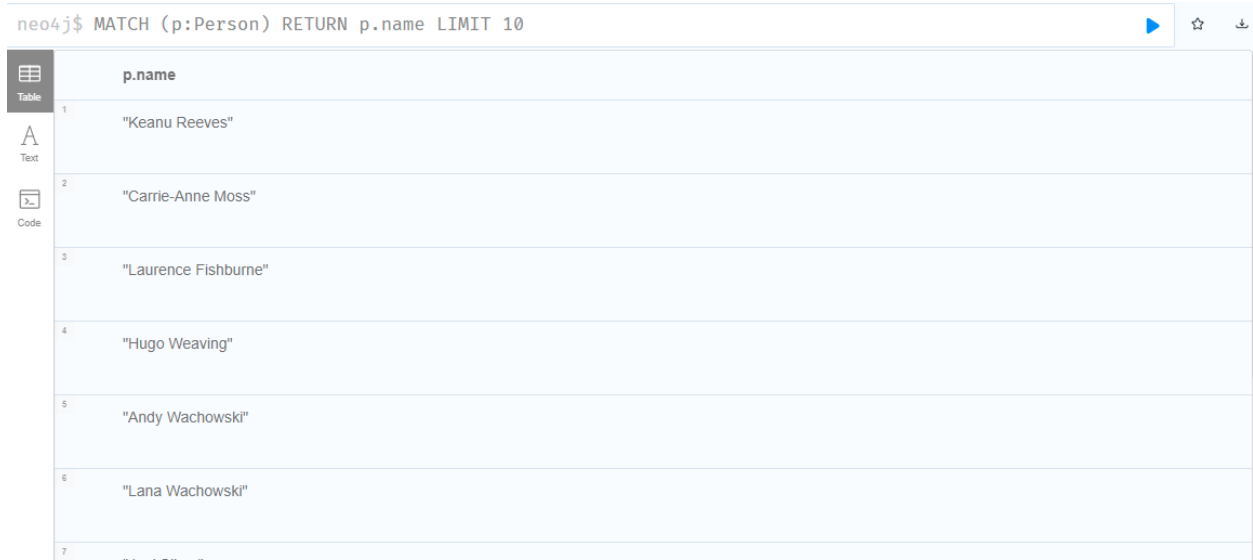
	type(r)	startNode	endtNode
1	"ACTED_IN"	["Person"]	["Movie"]
2	"WROTE"	["Person"]	["Movie"]
3	"DIRECTED"	["Person"]	["Movie"]
4	"PRODUCED"	["Person"]	["Movie"]
5	"FOLLOWS"	["Person"]	["Person"]
6	"REVIEWED"	["Person"]	["Movie"]

Started streaming 6 records after 19 ms and completed after 39 ms.

5. Recuperer le nœud de l'acteur Tom Hanks MATCH (tom:Person) WHERE tom.name = "Tom Hanks"  
RETURN tom



6. Afficher les noms de 10 personnes quelconques



The image shows the Neo4j Cypher query interface. The query entered is: `neo4j$ MATCH (p:Person) RETURN p.name LIMIT 10`. The interface displays a table with 10 rows of random person names. The table has a header row with the column name "p.name". The rows are numbered 1 through 10.

	p.name
1	"Keanu Reeves"
2	"Carrie-Anne Moss"
3	"Laurence Fishburne"
4	"Hugo Weaving"
5	"Andy Wachowski"
6	"Lana Wachowski"
7	"Lana Wachowski"
8	"Lana Wachowski"
9	"Lana Wachowski"
10	"Lana Wachowski"

## 7. Afficher les titres des films sortis dans les années 90

```
neo4j$ MATCH (nineties:Movie) WHERE nineties.released ≥ 1990 AND nineties.released < 2000 RETURN n...
```

	nineties.title
1	"The Matrix"
2	"The Devil's Advocate"
3	"A Few Good Men"
4	"As Good as It Gets"
5	"What Dreams May Come"
6	"Snow Falling on Cedars"
7	"The Green Mile"

Started streaming 40 records after 17 ms and completed after 23 ms.

## 8. Afficher les titres des films de Tom Hanks ainsi que les rôles qu'il a joué

```
neo4j$ MATCH (tom:Person {name : "Tom Hanks"})-[r:ACTED_IN]→(movie) RETURN movie.title AS Movie, r...
```

	Movie	Roles
1	"You've Got Mail"	["Joe Fox"]
2	"Apollo 13"	["Jim Lovell"]
3	"Joe Versus the Volcano"	["Joe Banks"]
4	"That Thing You Do"	["Mr. White"]
5	"Cloud Atlas"	["Zachry", "Dr. Henry Goose", "Isaac Sachs", "Dermot Hoggins"]
6	"The Da Vinci Code"	["Dr. Robert Langdon"]
7	"The Green Mile"	["Paul Edgecomb"]

## 9. Qui a réalisé le film "Cloud Atlas"?

```
neo4j$ MATCH (m:Movie {title : "Cloud Atlas"})←[:DIRECTED]-(director:Person) RETURN director.name
```

	director.name
1	"Tom Tykwer"
2	"Lana Wachowski"
3	"Andy Wachowski"
4	"Andy Wachowski"
5	"Tom Tykwer"
6	"Lana Wachowski"

Started streaming 6 records after 10 ms and completed after 10 ms.

## 10. Comment les gens sont-ils reliés au film "Cloud Atlas" ?

```
neo4j$ MATCH (p:Person)-[relatedTo]-(:Movie {title: "Cloud Atlas"}) RETURN p.name, Type(relatedTo),...
```

	p.name	Type(relatedTo)	related To
1	"Tom Hanks"	"ACTED_IN"	<pre>{   "identity": 137,   "start": 71,   "end": 105,   "type": "ACTED_IN",   "properties": {     "roles": [       "Zachry",       "Dr. Henry Goose",       "Isaac Sachs",       "Dermot Hoggins"     ]   } }</pre>

## 3 Premières manipulations

1. Supprimer toutes les données existantes via la requête ci-dessous

```
neo4j$ MATCH (n) DETACH DELETE n
```

Deleted 348 nodes, deleted 508 relationships, completed after 180 ms.

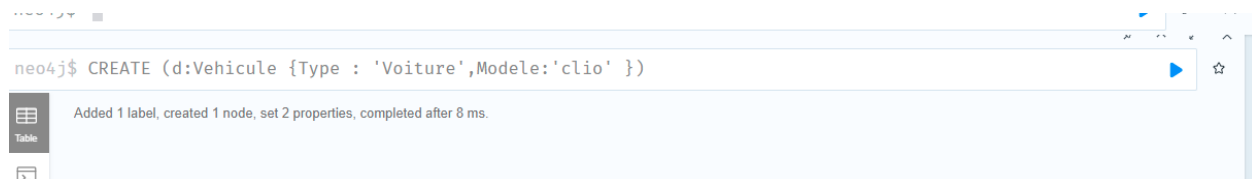
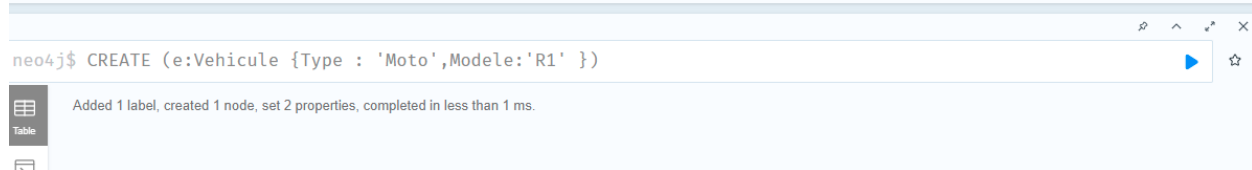
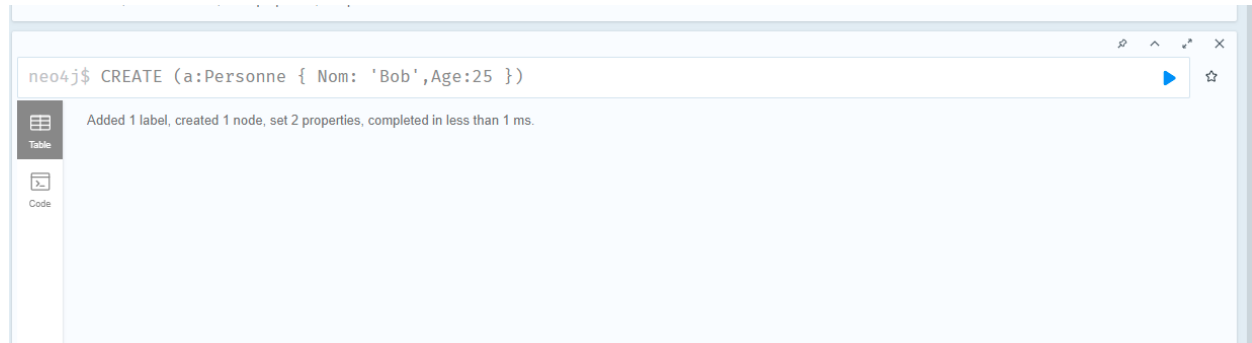
2. Création de nœuds ayant les caractéristiques spécifiées dans le tableau 1

```
neo4j$ CREATE (c:Personne { Nom: 'Ben',Age:25 })
```

Added 1 label, created 1 node, set 2 properties, completed after 8 ms.

```
neo4j$ CREATE (b:Personne { Nom: 'Alice',Age:25 })
```

Added 1 label, created 1 node, set 2 properties, completed after 8 ms.



3. Création de relations ayant les caractéristiques spécifiées dans le tableau 2.

```
1 MATCH(a:Personne),(b:Vehicule)
2 WHERE a.Nom = 'Bob' AND b.Modele = 'Clio'
3 CREATE (a)-[p:Posséder{Depuis:2014}]->(b)
```

Set 1 property, created 1 relationship, completed after 16 ms.

```
1 MATCH(b:Personne),(e:Vehicule)
2 WHERE b.Nom = 'Alice' AND e.Modele = 'R1'
3 CREATE (b)-[p:Posséder{Depuis:2007}]->(e)
```

Set 1 property, created 1 relationship, completed after 8 ms.

```
1 MATCH(c:Personne),(f:Vehicule)
2 WHERE c.Nom = 'Ben' AND f.Modele = 'A4'
3 CREATE (c)-[p:Posséder{Depuis:2010}]->(f)
```

Set 1 property, created 1 relationship, completed after 8 ms.

```
1 MATCH(c:Personne),(f:Vehicule)
2 WHERE c.Nom = 'Ben' AND f.Modele = 'A4'
3 CREATE (c)-[p:Conduire]->(f)
```

Created 1 relationship, completed after 24 ms.

```
1 MATCH(a:Personne),(d:Vehicule)
2 WHERE a.Nom = 'Bob' AND d.Modele = 'Clio'
3 CREATE (a)-[p:Conduire]->(d)
```

Created 1 relationship, completed after 16 ms.

```
1 MATCH(b:Personne),(e:Vehicule)
2 WHERE b.Nom = 'Alice' AND e.Modele = 'R1'
3 CREATE (b)-[p:Conduire]->(e)
```

Created 1 relationship, completed after 16 ms.

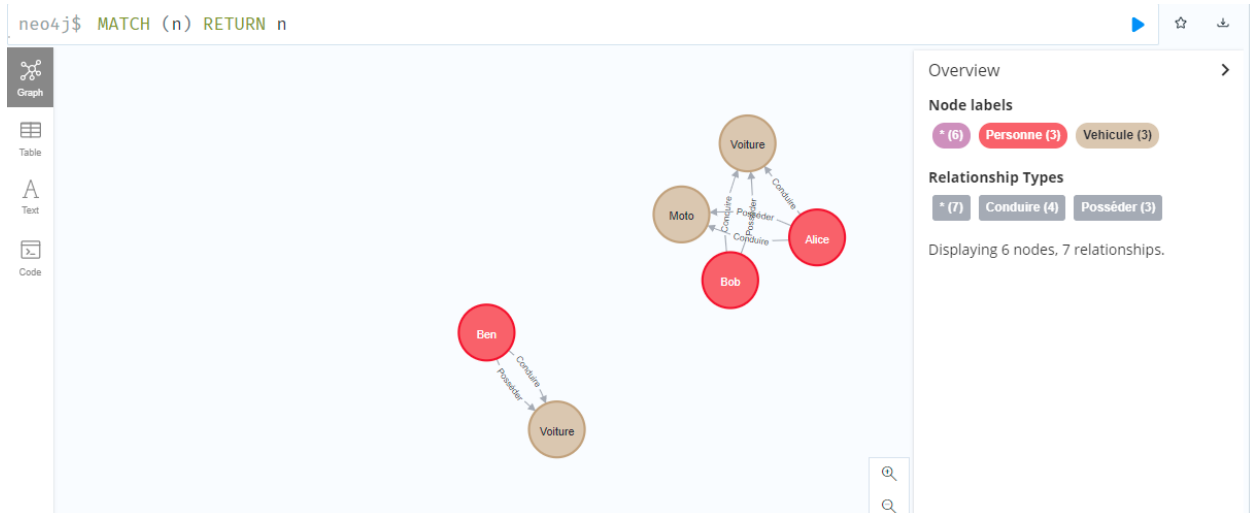


```

1 MATCH(b:Personne),(a:Vehicule)
2 WHERE b.Nom = 'Alice' AND a.Modele = 'Clio'
3 CREATE (b)-[p:Conduire]-(a)

```

Created 1 relationship, completed after 8 ms.



## 3.2 Quelques mises à jour et premières requêtes

Maintenant que quelques données sont présentes dans la base, nous allons pouvoir effectuer quelques manipulations et interrogations.

### 3.2.1 Manipulation de données

Effectuez les mises à jour suivantes :

1. Modifiez le nom de Bob par Jack et augmenter son Age de 10 ans;

```

neo4j$ MATCH (Bob:Personne) WHERE Bob.Age=33 SET Bob.Age=43

```

(no changes, no records)

neo4j\$ MATCH (n:Personne) RETURN n LIMIT 25

Graph

Table

Text

Code

n

1

```
{
  "identity": 6,
  "labels": [
    "Personne"
  ],
  "properties": {
    "Nom": "Bob",
    "Age": 43
  }
}
```

2

neo4j\$ MATCH (Bob:Personne) WHERE Bob.Nom='Bob' SET Bob.Nom='Jack'

Set 1 property, completed after 6 ms.

neo4j\$ MATCH (n:Personne) RETURN n LIMIT 25

Graph

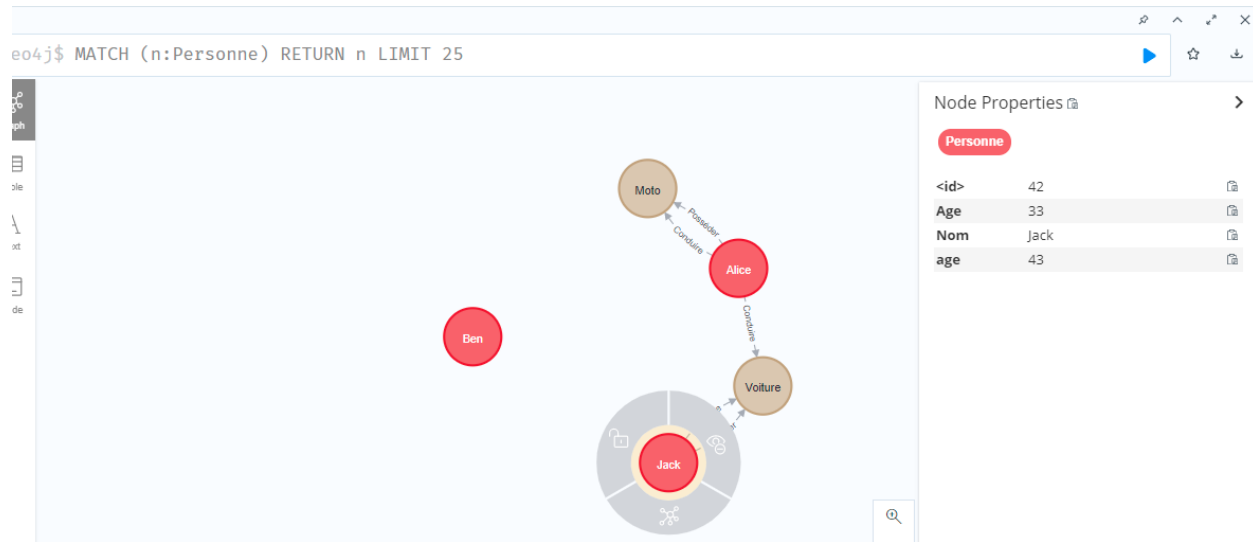
Table

Text

Code

n

```
"identity": 6,
"labels": [
  "Personne"
],
"properties": {
  "Nom": "Jack",
  "Age": 43
}
```



2. Ajouter un attribut PermisMoto valant OUI aux personnes possédant une moto ;



3. Ajouter un nouvel attribut NbRoues aux nœuds de type Véhicule et remplissez selon le type de véhicule.

neo4j\$ `MATCH (v:Vehicule{Type:"Voiture"}) SET v.NbRoues=4 Return v`

Graph

Table

Text

Code

```
{
  "identity": 45,
  "labels": [
    "Vehicule"
  ],
  "properties": {
    "Type": "Voiture",
    "NbRoues": 4,
    "Modele": "Clio"
  }
}
```

neo4j\$ `MATCH (x:Vehicule{Type:"Moto"}) SET x.NbRoues=2 Return x`

Graph

Table

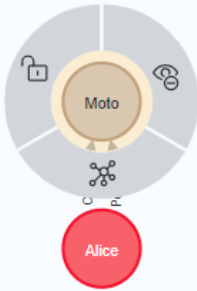
Text

Code

Node Properties

Vehicule

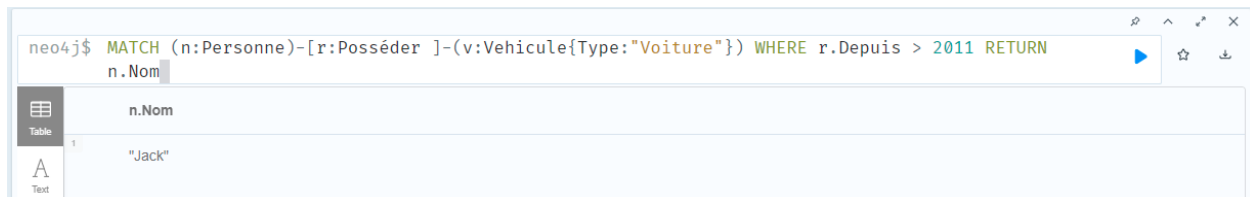
<id>	46	
Modele	R1	
NbRoues	2	
Type	Moto	



### 3.3 Interrogation

Ecrivez et testez les requêtes suivantes :

1. Qui possède une voiture depuis au moins 2012 ? Vous ne renverrez que l'attribut Nom de la ou des personnes concernées.



```
neo4j$ MATCH (n:Personne)-[r:Posséder ]-(v:Vehicule{Type:"Voiture"}) WHERE r.Depuis > 2011 RETURN n.Nom
```

n.Nom
"Jack"

2. Quels sont les propriétaires de moto ? Vous ne renverrez que l'attribut Nom de la ou des Personnes concernées.



```
neo4j$ MATCH (personne:Personne)-[r:Posséder ]-(v:Vehicule{Type:"Moto"...
```

personne.Nom
"Alice"

3. Qui conduit une voiture conduite également par Alice ?