Pour créer un nœud (de type « Person », qui est un libellé

CREATE (n:Personne { nom : 'John', title : 'Dev, an : 1990 })

CREATE (n:Personne { nom : 'Joe', poste : 'Dev', an : 1991 })

CREATE (n:Personne { nom : 'Jack', poste : 'Dev', an : 1992 })

Pour retourner tous les noeuds

MATCH (a)

RETURN a *(ou a.nom)*

Pour ajouter une relation entre deux nœuds avec propriétés

MATCH (a),(b)

WHERE a.nom = 'John' AND b.nom = 'Jack'

CREATE (a)-[r:AMI\_AVEC {depuis: 1995}]->(b)

**ou mieux :**

MATCH (a:Personne {nom : 'John'}),(b:Personne {nom : 'Jack'})

CREATE (a)-[r:AMI\_AVEC]->(b)

L’idéal serait de créer un index pour faciliter les recherches selon le nom.

CREATE INDEX idx\_p FOR (p:Personne) ON (p.nom)

Ajoutons une autre relation (Jack à Joe)

MATCH (a:Personne {nom : 'Jack'}),(b:Personne {nom : 'Joe'})

CREATE (a)-[r:AMI\_AVEC]->(b)

Si l’on veut trouver un nœud :

MATCH (p)

WHERE p.nom = 'John'

RETURN p

Si l’on veut trouver le(s) nœud(s) (différent de ), ordonné:

MATCH (p)

WHERE p.nom <> 'John'

RETURN p

ORDER BY p.nom

Pour les amis immédiats

MATCH (p)–[:AMI\_AVEC]->(p2)

WHERE p.nom = 'Jack'

RETURN p2.nom

Pour les amis immédiats (**peu importe le sens**)

MATCH (p {nom : 'Jack'} )–[:AMI\_AVEC]-(p2)

RETURN p2.nom

Retrouver les amis potentiels (de John)

MATCH (p)-[:AMI\_AVEC]->(p2), (p2)-[:AMI\_AVEC]->(p3)

WHERE p.nom = 'John'

RETURN **DISTINCT** p3.nom

Toutes les personnes qui n’ont pas d’amis

MATCH (a)

WHERE NOT(()-[:AMI\_AVEC]->(a))

RETURN a.nom

Retourner les relation entre deux noeuds

MATCH (p {nom : 'Jack'} )–[rel]-(p2 {nom : 'John'} )

RETURN rel

Suppression (avec les relations)

MATCH (a {nom : 'John'})

DETACH DELETE a

Mise à jour

MATCH (n:Personne {nom:"Jack"})

SET n.nom = "Jane", n.poste = "Finance"

Les ids commencent à zero, pour les comparer …

(On n’utilise plus la fonction ID(n)…)

MATCH (n :Personne)

WHERE elementId(n) = 1

RETURN n

Aggrégation (group by)

MATCH (n)

WHERE n.an > 1900

RETURN n.poste, avg(n.an) (group by n.poste)

Ou avec having…

MATCH (n)

WHERE n.an > 1900

WITH n.poste as titre, count(\*) as total

WHERE total > 1

RETURN titre, total