

Exercice 1

3

a.

```
CREATE (:Person:Actor {name: 'Tom Hanks', born: 1956})
```

Cette commande crée un nœud avec les labels **Person** et **Actor** avec les propriétés **name="Tom Hanks"** et **born=1956**.

b.

```
MATCH (n) DETACH DELETE n
```

Cette commande trouve tous les nœuds et les supprime ainsi que leurs relations.

c.

```
CREATE (n:Person:Actor {name: 'Tom Hanks', born: 1956}) CREATE (n)-[:KNOWS]->(n)
```

Cette commande crée le nœud Tom Hanks puis crée une relation réflexive **KNOWS** sur ce nœud.

d.

```
CREATE (:Person:Actor {name: 'Tom Hanks', born: 1956})
```

Cette commande crée un autre nœud Tom Hanks en plus de celui qu'on avait déjà créé.

e.

```
MATCH (n:Person)-[:KNOWS]->(n) RETURN n
```

Cette commande trouve toutes les nœuds avec le label **Person** qui ont une relation réflexive **KNOWS**.

f.

```
CREATE (n:Person {name: 'Alice'}) CREATE (n)-[:KNOWS]->(n)
```

Cette commande crée la **Person** Alice puis crée une relation réflexive **KNOWS** sur Alice.

g.

```
MATCH (n:Person)-[:KNOWS]->(n) RETURN n
```

Cette commande trouve toutes les nœuds avec le label **Person** qui ont une relation réflexive **KNOWS**. On a maintenant Tom Hanks et Alice.

h.

```
MATCH (n:Person {name: 'Alice'})-[r:KNOWS]->(n) SET r.since = 2020
```

Cette commande ajoute la propriété **since = 2020** à toutes les relations réflexives **KNOWS** sur les nœuds **Person** ayant la propriété **name="Alice"**.

i.

```
MATCH (n:Person {name: 'Alice'})-[r:KNOWS]->(n) DELETE r
```

Cette commande supprime toutes les relations réflexives **KNOWS** des nœuds **Person** ayant **name="Alice"**.

j.

```
MATCH (n) DETACH DELETE n;
CREATE (:Person:Actor {name: 'Tom Hanks', born: 1956})-[:ACTED_IN {roles: ['Forrest']}]->(:Movie {title: 'Forrest Gump', released: 1994})<-[:DIRECTED]-(:Person {name: 'Robert Zemeckis', born: 1951})
```

La première commande supprime tous les nœuds et leurs relations. La deuxième commande crée le nœud Tom Hanks, le nœud **Movie** Forrest Gump, le nœud **Person** Robert Zemeckis, avec la relation **ACTED_IN** de Tom Hanks à Forrest Gump dans le rôle de Forrest, et la relation **DIRECTED** de Robert Zemeckis à Forrest Gump.

k.

```
MATCH (n) DETACH DELETE n
```

Cette commande supprime tous les nœuds et leurs relations.

l.

```
CREATE (:Person:Actor {name: 'Tom Hanks', born: 1956})-[:ACTED_IN {roles: ['Forrest']}]->(:Movie {title: 'Forrest Gump', released: 1994})<-[:DIRECTED]-(:Person {name: 'Robert Zemeckis', born: 1951})
```

Cette commande recrée les nœuds et les relations qu'on a supprimé.

m.

Création de Emil:

```
CREATE (:Person {name: "Emil"})
```

```
MATCH (ee:Person) WHERE ee.name = 'Emil' RETURN ee;
```

Cette commande trouve toutes les **Person** avec **name="Emil"**.

n.

```
MATCH (ee:Person) WHERE ee.name = 'Emil' CREATE (js:Person { name: 'Johan', from: 'Sweden', learn: 'surfing' }), (ir:Person { name: 'Ian', from: 'England', title: 'author' }), (rvb:Person { name: 'Rik', from: 'Belgium', pet: 'Orval' }), (ally:Person { name: 'Allison', from: 'California', hobby: 'surfing' }), (ee)-[:KNOWS {since: 2001}]->(js),(ee)-[:KNOWS {rating: 5}]->(ir), (js)-[:KNOWS]->(ir),(js)-[:KNOWS]->(rvb), (ir)-[:KNOWS]->(js),(ir)-[:KNOWS]->(ally), (rvb)-[:KNOWS]->(ally)
```

Cette commande crée un réseau de personne avec des nœuds et des relations.

o.

```
MATCH (ee:Person)-[:KNOWS]-(friends) WHERE ee.name = 'Emil' RETURN ee
```

Cette commande trouve les **Person** avec **name="Emil"** connaissant au moins une personne.

Exercice 2

1

Supprimer toutes les données:

```
MATCH (n) DETACH DELETE n
```

2

```
CREATE CONSTRAINT FOR (n:Movie) REQUIRE (n.title) IS UNIQUE
```

a. Quels sont les effets de cette contrainte sur les données des nœuds de type Movie dans la base de données ?

Cette contrainte requiert que tous les **Movies** aient un **title** unique.

b. Quels types de données sont affectés par cette contrainte ?

Les données affectées sont les nœuds avec le label **Movie**.

c. Que se passera-t-il si vous essayez d'ajouter deux nœuds de type Movie avec le même title ?

Le SGBD refuserait l'opération.

d. Pourquoi est-il important d'utiliser des contraintes uniques dans une base de données ?

Cela évite d'avoir des anomalies dans la base de données.

3. Donnez la requête qui permet de créer une contrainte équivalente pour le nœud Person sur la propriété name

```
CREATE CONSTRAINT FOR (n:Person) REQUIRE (n.name) IS UNIQUE
```

5. Donnez le titre de tous les films

```
match (n) where n.title is not null return n.title
```

n.title
"The Matrix Reloaded"
"The Matrix Revolutions"
"The Devil's Advocate"
"A Few Good Men"
"Top Gun"
"The Matrix"
"Jerry Maguire"
"Stand By Me"
"As Good as It Gets"
"What Dreams May Come"
"Snow Falling on Cedars"
"You've Got Mail"
"Sleepless in Seattle"
"Joe Versus the Volcano"
"When Harry Met Sally"
"That Thing You Do"
"The Replacements"
"RescueDawn"
"The Birdcage"
"Unforgiven"
"Johnny Mnemonic"
"Cloud Atlas"
"The Da Vinci Code"
"V for Vendetta"
"Speed Racer"
"Ninja Assassin"
"The Green Mile"
"Frost/Nixon"
"Hoffa"
"Apollo 13"
"Twister"
"Cast Away"
"One Flew Over the Cuckoo's Nest"
"Something's Gotta Give"
"Bicentennial Man"
"Charlie Wilson's War"
"The Polar Express"
"A League of Their Own"

6. Donnez les titres des films produits par Lilly Wachowski

```
match (p)-[:PRODUCED]->(m) where p.name = "Lilly Wachowski" return m.title
```

m.title
"V for Vendetta"
"Ninja Assassin"

7. Donnez les noms de toutes les personnes dans la base de données

```
match (p:Person) return p.name
```

p.name
"Lana Wachowski"
"Joel Silver"
"Emil Eifrem"
"Charlize Theron"
...
"Angela Scope"
"Jessica Thompson"
"James Thompson"

8. Donnez le titre du film avec le plus grand nombre de personnes associées en tant que producteurs.

```
MATCH (p:Person)-[:PRODUCED]->(m:Movie)
WITH m, COLLECT(p) as producers
RETURN m.title
ORDER BY SIZE(producers) DESC LIMIT 1
```

"V for Vendetta"

9. Donnez le titre du film avec le plus grand nombre de personnes associées en tant qu'acteurs

```
MATCH (p:Person)-[:ACTED_IN]->(m:Movie)
WITH m, COLLECT(p) as actors
RETURN m.title
ORDER BY SIZE(actors) DESC LIMIT 1
```

"A Few Good Men"

10. Donnez le titre du film avec le plus grand nombre de personnes associées, soit en tant que producteurs, soit en tant qu'acteurs.

```
MATCH (p:Person)-[:ACTED_IN|PRODUCED]->(m:Movie)
WITH m, COLLECT(p) as actors
RETURN m.title
ORDER BY SIZE(actors) DESC LIMIT 1
```

"A Few Good Men"

11. Donnez la requête qui permet d'obtenir ce résultat :

m.title	numProducers	numActors	numPeople
"A Few Good Men"	0	12	12

Requête:

```
MATCH (m:Movie)
WHERE m.title = "A Few Good Men"
OPTIONAL MATCH (m)-[:PRODUCED]->(p:Person)
```

```
OPTIONAL MATCH (a:Person)-[:ACTED_IN]->(m)
WITH m, Count(p) as numProducers, Count(a) as numActors, Count(p)+Count(a) as numPeople
RETURN m.title, numProducers, numActors, numPeople
```

12. Interprétez la requête suivante :

```
MATCH (p:Person)-[r1:PRODUCED]->(m:Movie)-[:PRODUCED]-(p2:Person)
WHERE p <> p2
RETURN p.name, p2.name, m.title;
```

Cette requête trouve les films produits par au moins deux personnes différentes.

13. Donnez les personnes qui ont produit le plus grand nombre de films

```
MATCH (p:Person)-[:PRODUCED]->(m:Movie)
WITH p, Count(m) as movies
RETURN p.name, movies
ORDER BY movies DESC LIMIT 5
```

p.name	movies
"Joel Silver"	6
"Lilly Wachowski"	2
"Lana Wachowski"	2
"Rob Reiner"	1
"Stefan Arndt"	1