En utilisant la clause Set, vous pouvez ajouter de nouvelles propriétés à un nœud ou une relation existante, et également ajouter ou mettre à jour les valeurs des propriétés existantes. Dans ce Tp, nous allons voir comment :

- Définir une propriété
- Enlever un bien
- Définir des propriétés multiples
- Apposer une étiquette sur un nœud
- Mettre plusieurs étiquettes sur un même nœud

1. La clause SET

En utilisant la clause SET, vous pouvez créer une nouvelle propriété dans un nœud.

Vous trouverez ci-dessous la syntaxe pour modifier une propriété.

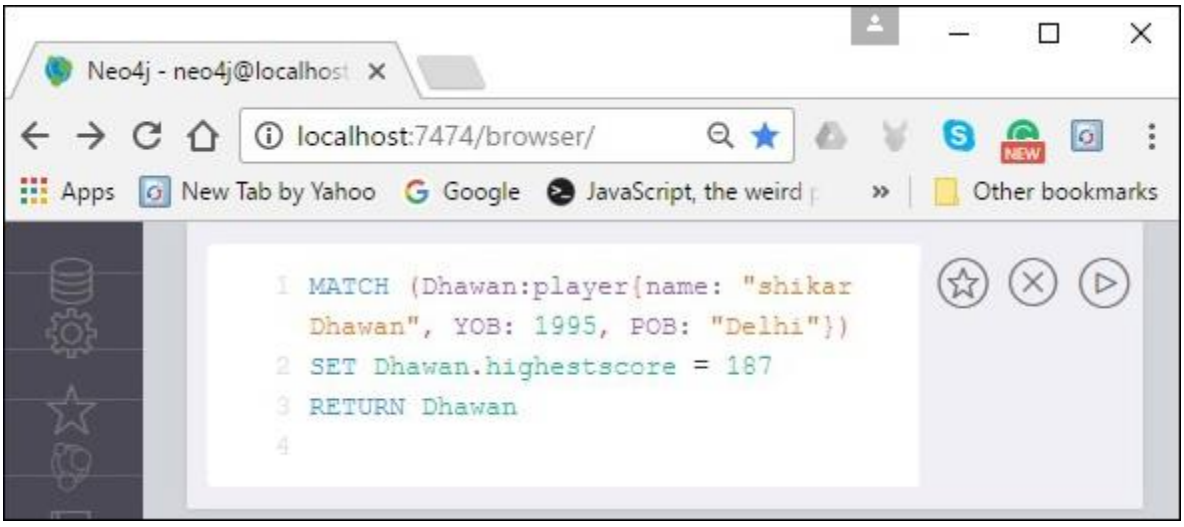
```
MATCH (node:label{properties . . . . . }) 

SET node.property = value

RETURN node
```

Avant de poursuivre l'exemple, créez d'abord un nœud nommé Dhawan comme indiqué ci-dessous.

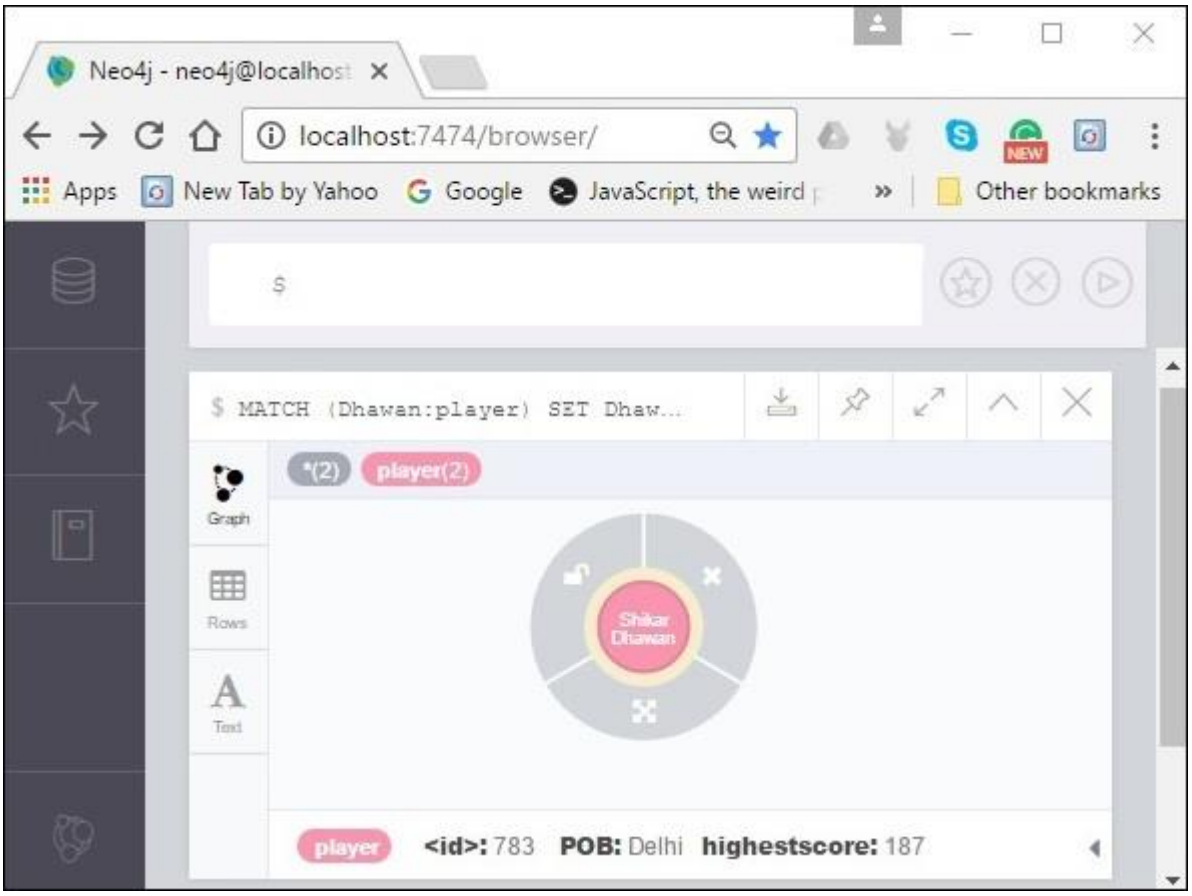
```
CREATE (Dhawan:player{name: "Shikar Dhawan", YOB: 1985, POB: "Delhi"})
```

Voici un exemple de requête chiffrée pour créer une propriété nommée "highestscore" avec la valeur "187".

```
MATCH (Dhawan:player{name: "shikar Dhawan", YOB: 1985, POB: "Delhi"})

SET Dhawan.highestscore = 187

RETURN Dhawan
```



Lors de l'exécution, vous obtiendrez le résultat suivant. Vous pouvez observer ici qu'une propriété avec une paire de valeurs clés highestscore/187 est créée dans le nœud nommé "Dhawan".



Vous pouvez supprimer un bien existant en lui attribuant la valeur **NULL**.

Voici la syntaxe de la suppression d'une propriété d'un nœud à l'aide de la clause SET.

```
MATCH (node:label{properties})

SET node.property = NULL

RETURN node
```

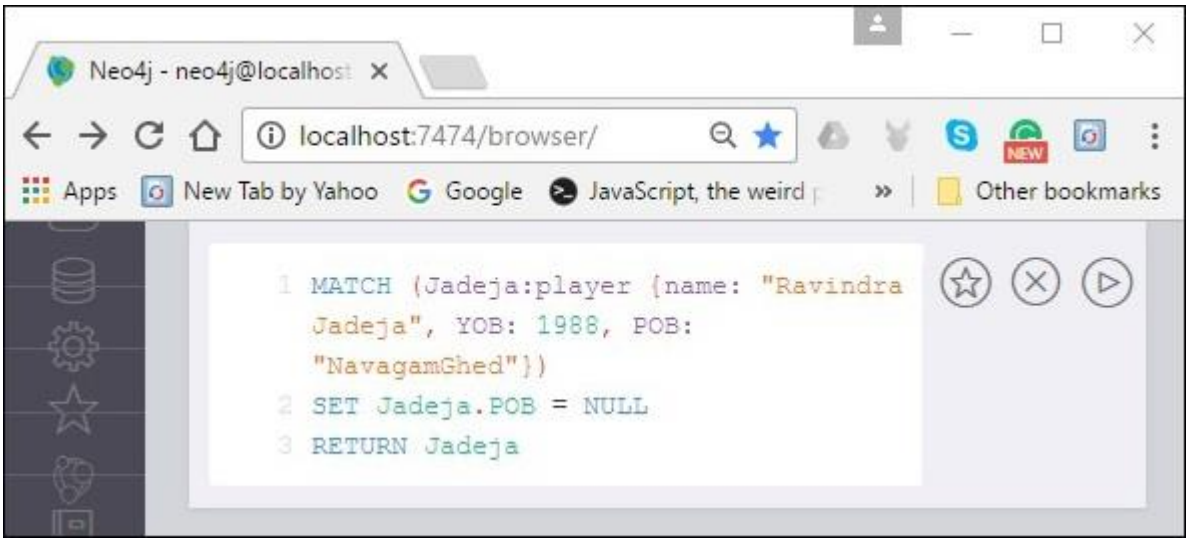
Avant de poursuivre l'exemple, créez d'abord un nœud "jadeja" comme indiqué ci-dessous.

```
Create (Jadeja:player {name: "Ravindra Jadeja", YOB: 1988, POB:
"NavagamGhed"})
```

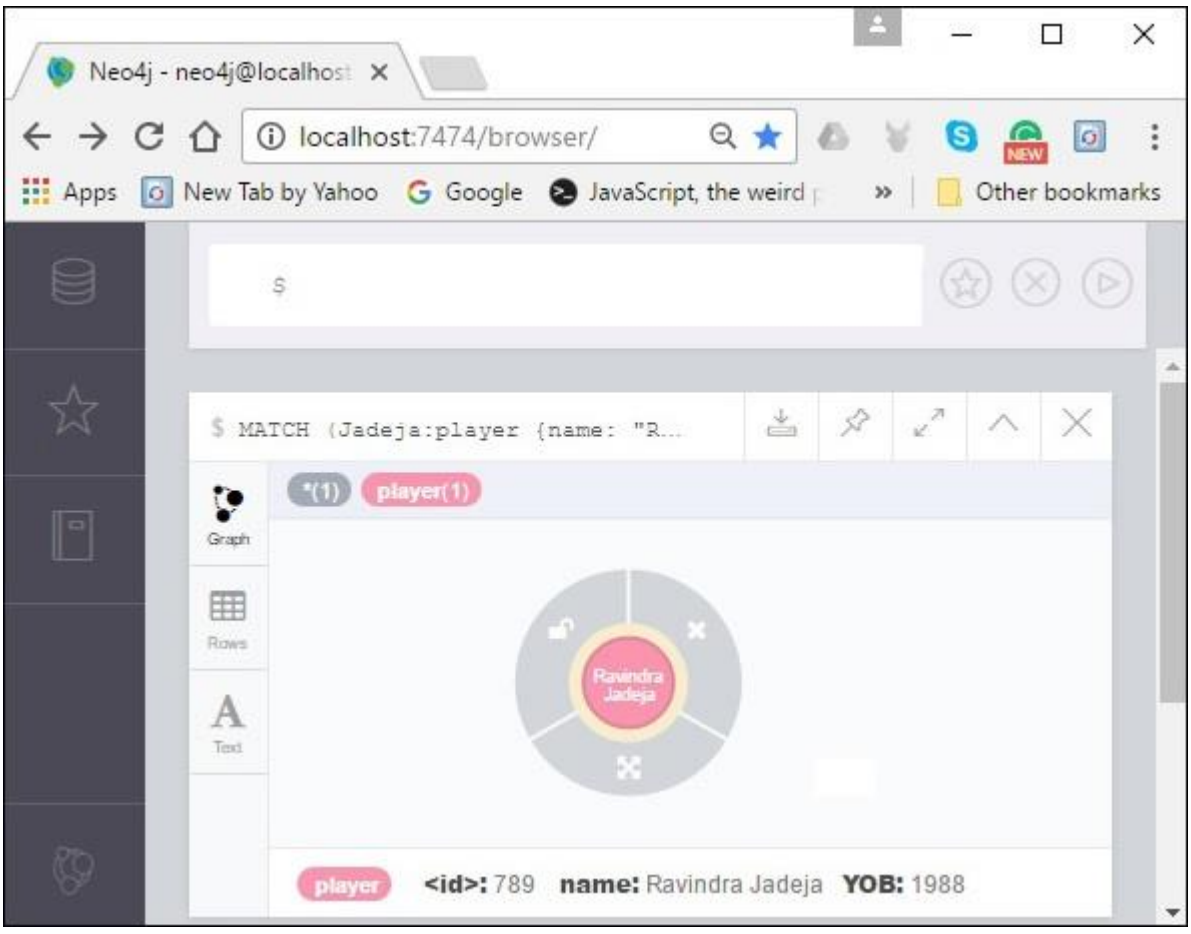
Voici un exemple de requête de qui supprime la propriété nommée POB de ce nœud en utilisant la clause SET comme indiqué ci-dessous.

```
MATCH (Jadeja:player {name: "Ravindra Jadeja", YOB: 1988, POB:
"NavagamGhed"})

SET jadeja.POB = NULL
```

```
RETURN jadeja
```



Lors de l'exécution, vous obtiendrez le résultat suivant. Ici, vous pouvez observer que la variable nommée POB a été supprimée.



Vous pouvez attribuer un label d'un nœud existant en utilisant la clause SET.

La syntaxe suivante permet de définir un label pour un nœud existant.

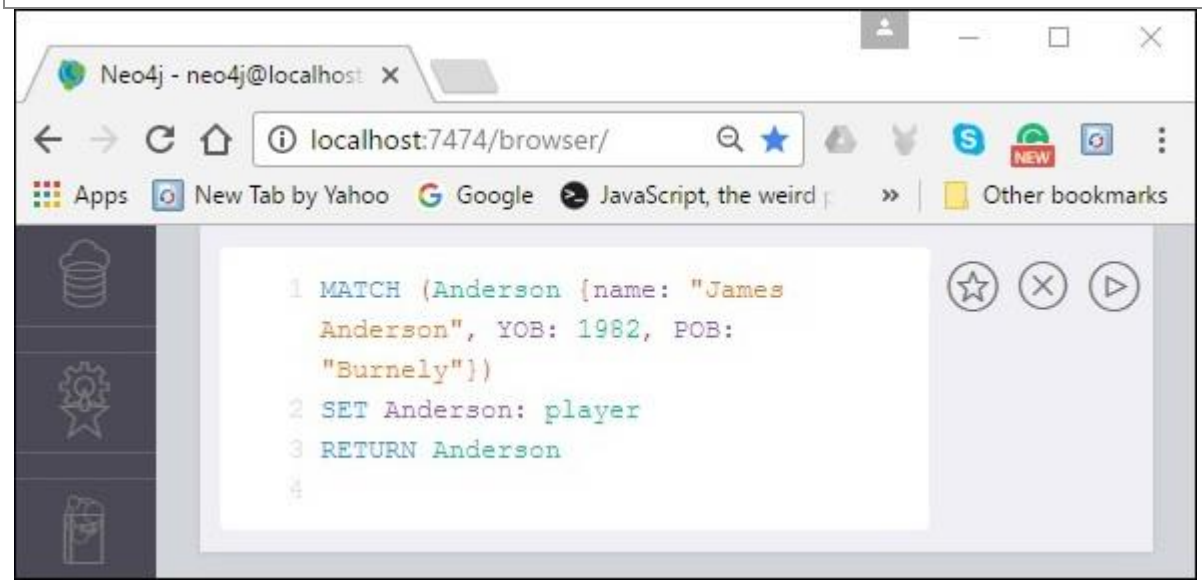
```
MATCH (n {properties . . . . . })

SET n :label

RETURN n
```

Avant de poursuivre l'exemple, créez d'abord un nœud "Anderson" comme indiqué ci-dessous.

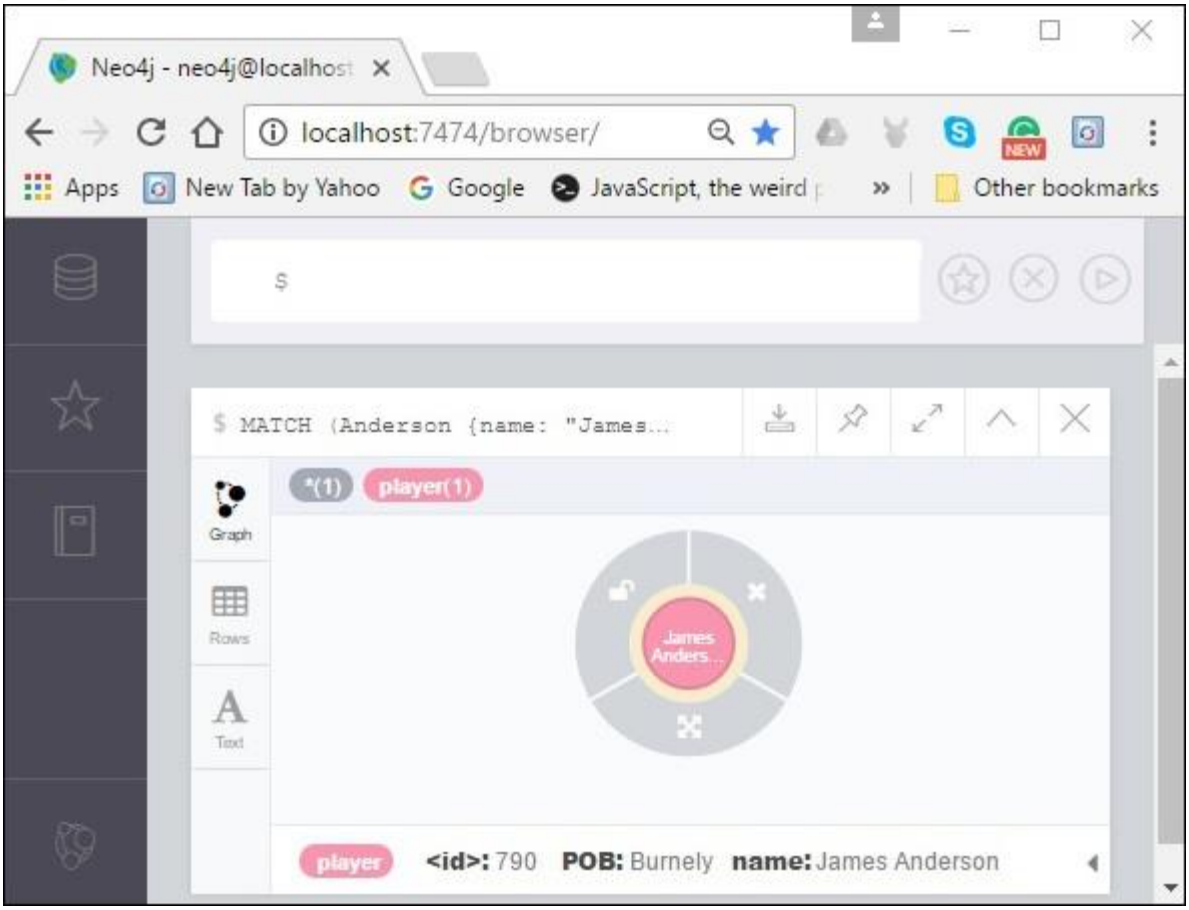
```
CREATE (Anderson {name: "James Anderson", YOB: 1982, POB: "Burnely"})
```

Vous trouverez ci-dessous un exemple de requête de CQL permettant de définir une étiquette sur un nœud à l'aide de la clause SET. Cette requête ajoute le label "player" au nœud Anderson et le renvoie.

```
MATCH (Anderson {name: "James Anderson", YOB: 1982, POB:
"Burnely"})
SET Anderson: player
RETURN Anderson
```



Lors de l'exécution, vous obtiendrez le résultat suivant. Ici, vous pouvez observer que l'étiquette nommée "player" est ajoutée au nœud.



Vous pouvez attribuer plusieurs étiquettes à un nœud existant en utilisant la clause SET. Ici, vous devez spécifier les étiquettes en les séparant par des deux-points " :".

La syntaxe suivante permet de fixer plusieurs étiquettes à un nœud existant en utilisant la clause SET.

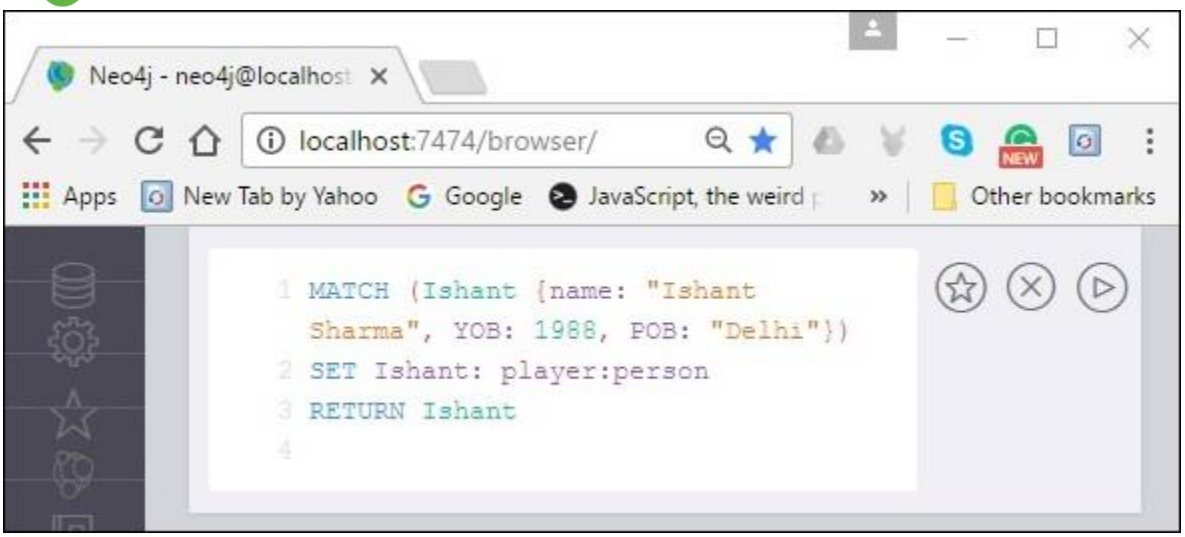
```
MATCH (n {properties . . . . . })

SET n :label1:label2

RETURN n
```

Avant de poursuivre l'exemple, créez d'abord un nœud nommé "Ishant" comme indiqué ci-dessous.

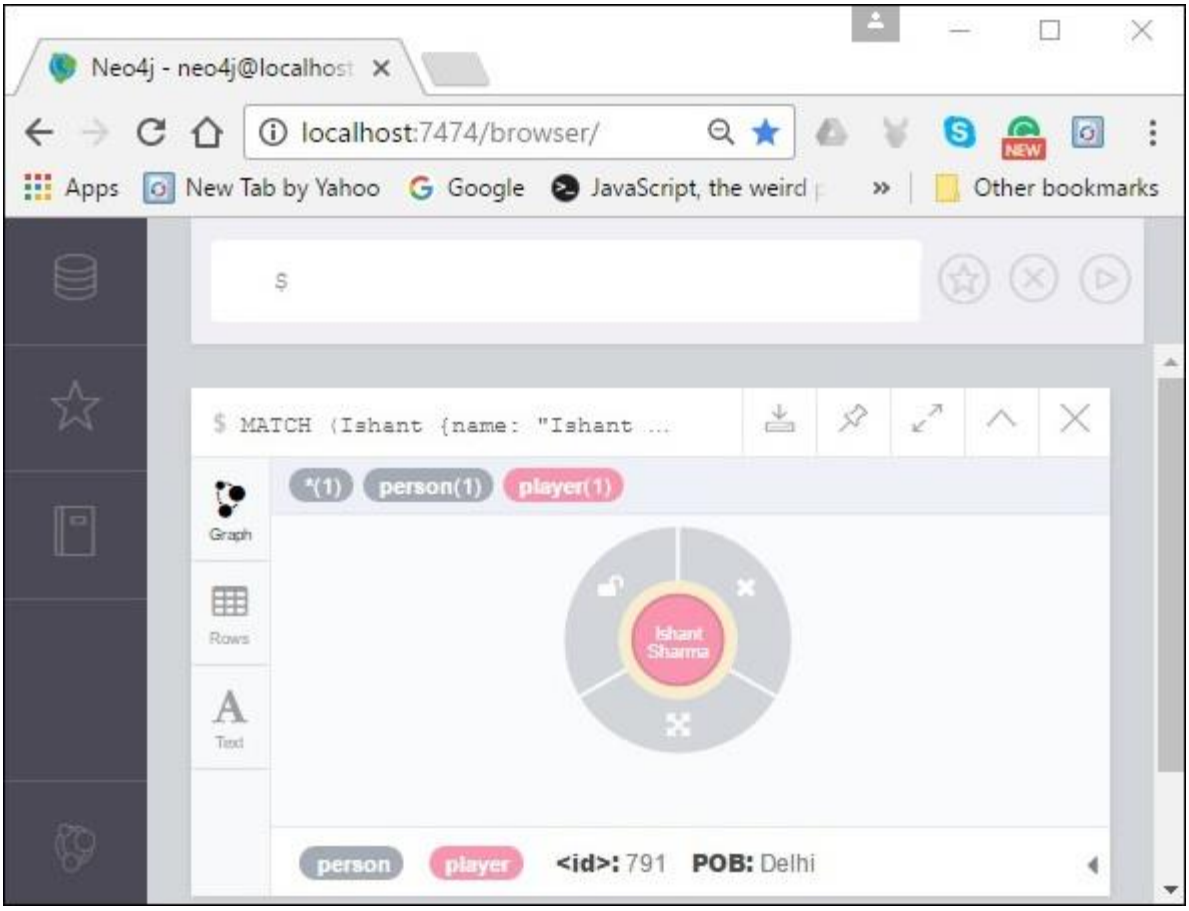
```
CREATE (Ishant {name: "Ishant Sharma", YOB: 1988, POB: "Delhi"})
```

Voici un exemple de requête de CQL utilisée pour créer plusieurs étiquettes sur un nœud en utilisant la clause SET.

```
MATCH (Ishant {name: "Ishant Sharma", YOB: 1988, POB: "Delhi"})
SET Ishant: player:person
RETURN Ishant
```



Lors de l'exécution, vous obtiendrez le résultat suivant. Ici, vous pouvez observer que deux étiquettes - personne et joueur - sont ajoutées au nœud nommé Ishant.



2. La clause DELETE

La requête suivante permet de supprimer tous les nœuds et les relations dans la base de données en utilisant la clause DELETE.

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

Pour supprimer un nœud particulier, vous devez spécifier les détails du nœud à la place de "n" dans la requête ci-dessus.

Voici la syntaxe pour supprimer un noeud particulier de Neo4j en utilisant la clause DELETE.

```
MATCH (node:label {properties . . . . . })
DETACH DELETE node
```

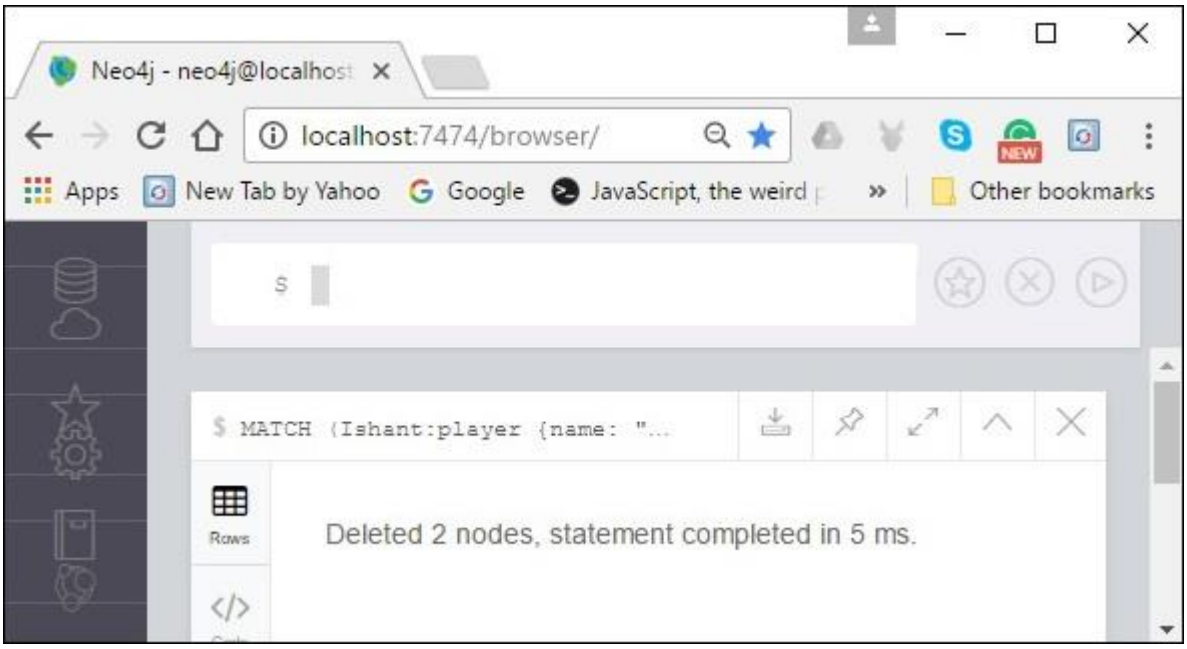
Avant de poursuivre l'exemple, créez un noeud "Ishant" dans la base de données Neo4j comme indiqué ci-dessous.

```
CREATE (Ishant:player {name: "Ishant Sharma", YOB: 1988, POB:
"Delhi"})
```

Voici un exemple de requête de chiffrement qui supprime le nœud créé ci-dessus en utilisant la clause DELETE.

```
MATCH (Ishant:player {name: "Ishant Sharma", YOB: 1988, POB:
"Delhi"})
DETACH DELETE Ishant
```

Lors de l'exécution, vous obtiendrez le résultat suivant. Ici, vous pouvez observer que le nœud spécifié est supprimé.



3. La clause REMOVE

La clause REMOVE est utilisée pour supprimer les propriétés et les étiquettes des éléments du graphique (nœuds ou relations). La principale différence entre les commandes Neo4j CQL DELETE et REMOVE est :

- L'opération DELETE est utilisée pour supprimer les nœuds et les relations associées.
- L'opération REMOVE est utilisée pour supprimer les étiquettes et les propriétés.

Vous pouvez supprimer une propriété d'un nœud en utilisant MATCH avec la clause REMOVE.

Voici la syntaxe pour supprimer une propriété d'un nœud en utilisant la clause REMOVE.

```
MATCH (node:label{properties . . . . . })
REMOVE node.property
RETURN node
```

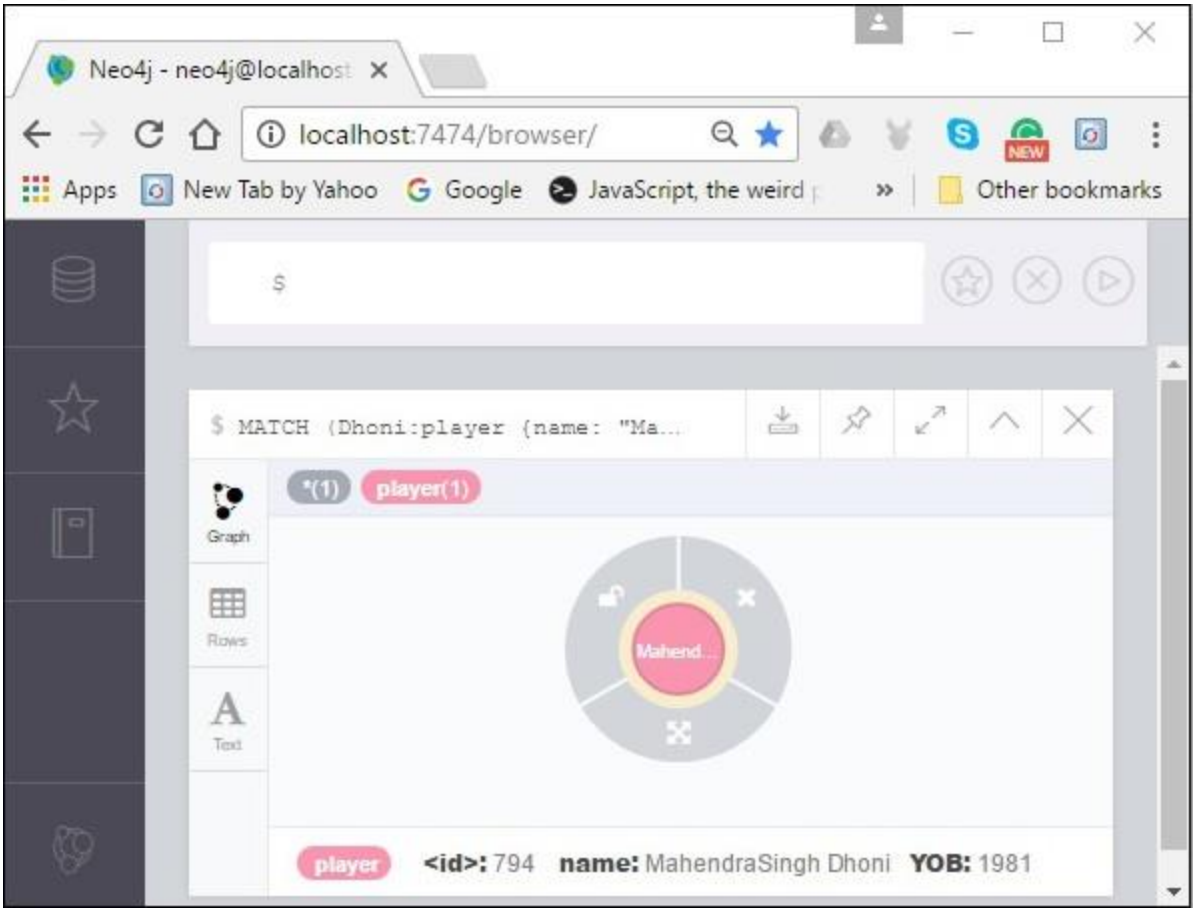
Avant de poursuivre l'exemple, créez un nœud nommé Dhoni comme indiqué ci-dessous.

```
CREATE (Dhoni:player {name: "MahendraSingh Dhoni", YOB: 1981, POB:
"Ranchi"})
```

Voici un exemple de requête de CQL pour supprimer le nœud créé ci-dessus en utilisant la clause REMOVE.

```
MATCH (Dhoni:player {name: "MahendraSingh Dhoni", YOB: 1981, POB:
"Ranchi"})
REMOVE Dhoni.POB
RETURN Dhoni
```

Lors de l'exécution, vous obtiendrez le résultat suivant. Ici, vous pouvez observer que le nœud nommé POB a été supprimé.

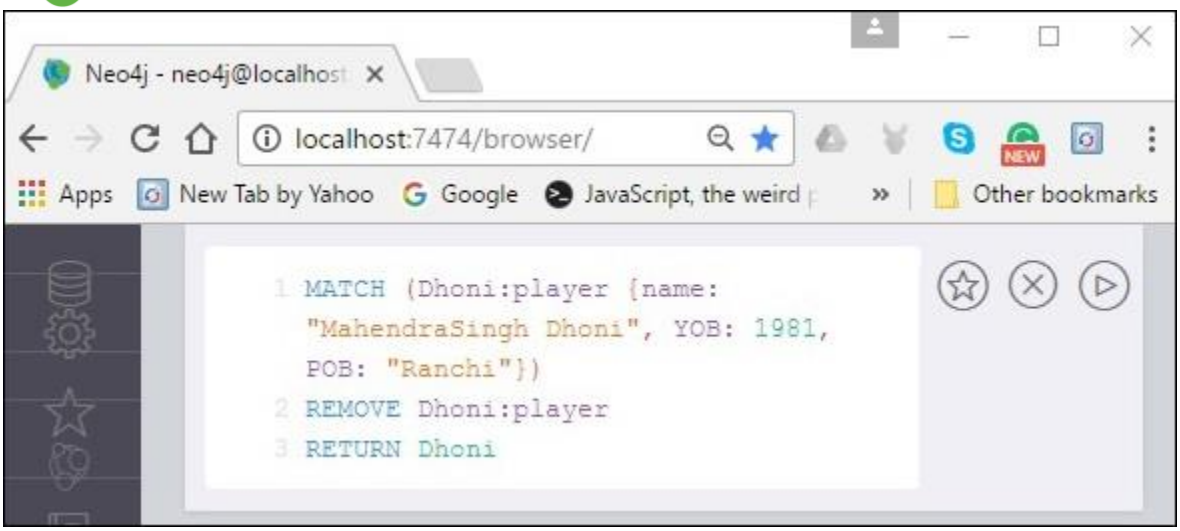


Comme pour la propriété, vous pouvez également supprimer une étiquette d'un nœud existant en utilisant la clause de suppression.

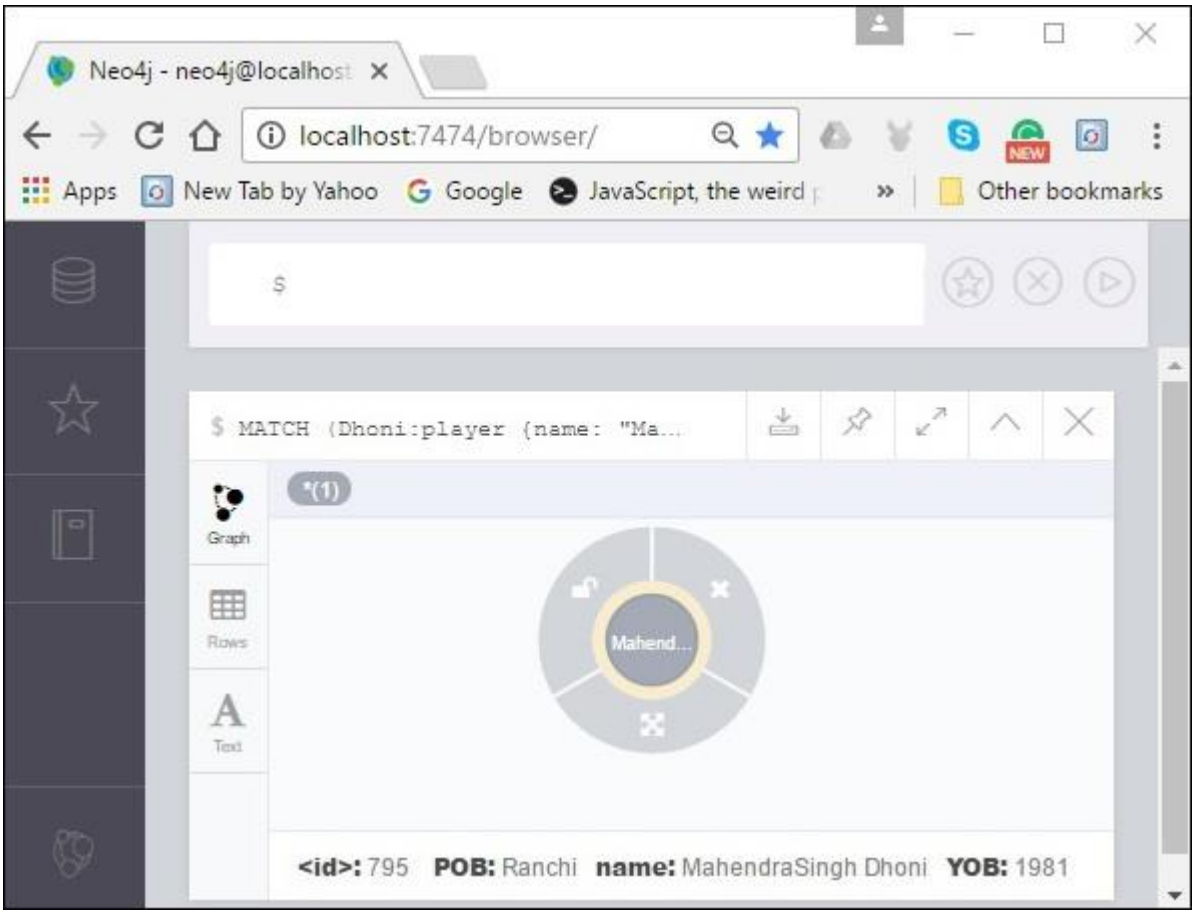
```
MATCH (node:label {properties . . . . . })
REMOVE node:label
RETURN node
```

Vous trouverez ci-dessous un exemple de requête de CQL permettant de supprimer un label d'un nœud existant en utilisant la clause de suppression.

```
MATCH (Dhoni:player {name: "MahendraSingh Dhoni", YOB: 1981, POB:
"Ranchi"})
REMOVE Dhoni:player
RETURN Dhoni
```



Lors de l'exécution, vous obtiendrez le résultat suivant. Ici, vous pouvez observer que le label a été supprimé du nœud.



You can also remove multiple labels from an existing node.

```
MATCH (node:label1:label2 {properties . . . . . })
```

```
REMOVE node:label1:label2
RETURN node
```

4. La clause FOREACH

La clause FOREACH est utilisée pour mettre à jour les données au sein d'une liste, qu'il s'agisse de composantes d'un chemin ou du résultat d'une agrégation.

```
MATCH p = (start node)-[*]->(end node)
WHERE start.node = "node_name" AND end.node = "node_name"
FOREACH (n IN nodes(p)| SET n.marked = TRUE)
```

Avant de poursuivre avec l'exemple, créez un chemin p dans la base de données Neo4j comme indiqué ci-dessous.

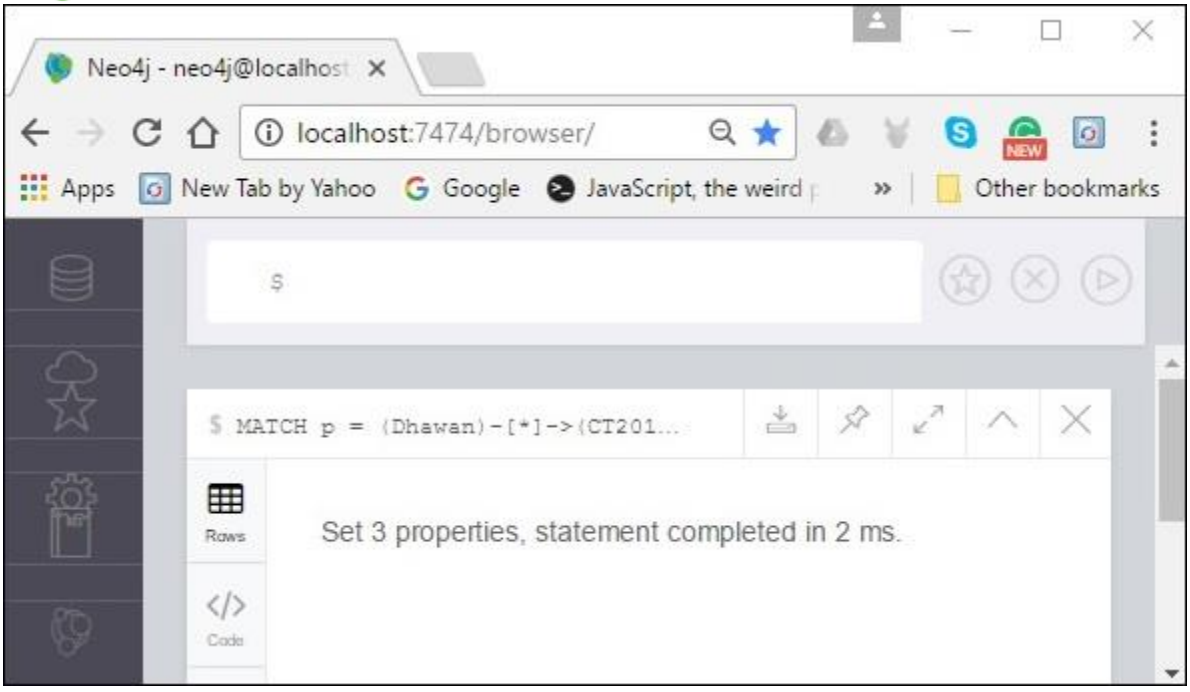
```
CREATE p = (Dhawan {name:"Shikar Dhawan"})-[:TOPSCORRER_OF]-
>(Ind{name:
   "India"})-[:WINNER_OF]->(CT2013{name: "Champions Trophy 2013"})
RETURN p
```

Voici un exemple de requête de CQL qui ajoute une propriété à tous les nœuds le long du chemin en utilisant la clause FOREACH.

```
MATCH p = (Dhawan)-[*]->(CT2013)
   WHERE Dhawan.name = "Shikar Dhawan" AND CT2013.name = "Champions
Trophy 2013"
FOREACH (n IN nodes(p)| SET n.marked = TRUE)
```

Pour exécuter la requête ci-dessus, effectuez les étapes suivantes -

Lors de l'exécution, vous obtiendrez le résultat suivant.

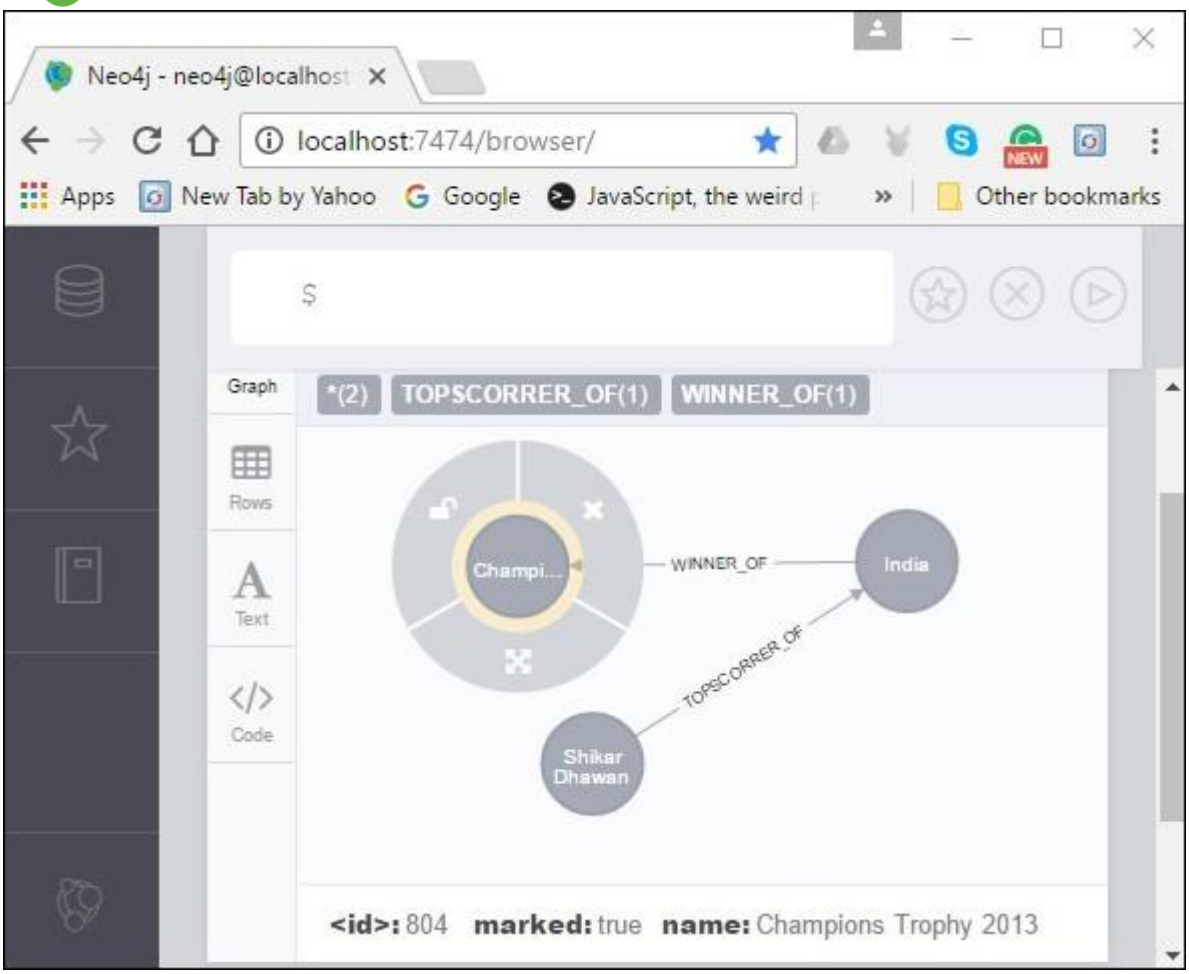


Pour vérifier la création du nœud, tapez et exécutez la requête suivante dans l'invite de commande.

```
MATCH (n) RETURN n
```

Cette requête renvoie tous les nœuds de la base de données

À l'exécution, cette requête affiche le nœud créé comme le montre la capture d'écran suivante.


Neo4j - neo4j@localhost
localhost:7474/browser/
Apps
New Tab by Yahoo
Google
JavaScript, the weird
Other bookmarks
$
Graph
Rows
Text
Code
*(2)
TOPSCORRER_OF(1)
WINNER_OF(1)
Champi...
WINNER_OF
India
TOPSCORRER_OF
Shikar Dhawan
<id>: 804 marked: true name: Champions Trophy 2013


neo4j