

Neo4j学习笔记

https://www.w3cschool.cn/neo4j/neo4j_cql_create_node.html

一、创建语句（**create**）

```
CREATE (emp:Employee)
```

1. 这里 **emp** 是一个节点名Employee 是 **emp** 节点的标签名称 它显示在 Neo4j 数据库中创建一个标签和一个节点。它在数据库中创建一个带有标签名“Employee”的节点“emp”。
2. Neo4j CQL创建具有属性的节点，Neo4j CQL“CREATE”命令用于创建带有属性的节点。它创建一个具有一些属性（键值对）的节点来存储数据。
3. **CREATE**命令语法：

```
CREATE (  
  <node-name>:<label-name>  
  {  
    <Property1-name>:<Property1-Value>  
    .....  
    <Propertyn-name>:<Propertyn-Value>  
  }  
)
```

4. 此示例演示如何创建具有一些属性（deptno, dname, 位置）的Dept节点。

```
CREATE (dept:Dept {  
  deptno:10,dname:"Accounting",location:"Hyderabad" })
```

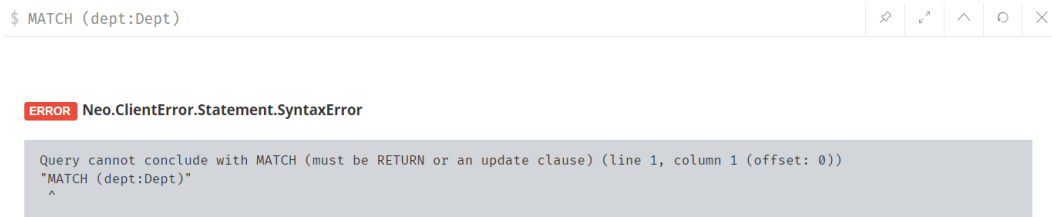
这里的属性名称是deptno, dname, location 属性值为 10, "Accounting","Hyderabad" 正如我们讨论的，属性一个名称 - 值对。Property = deptno:10 因为deptno是一个整数属性，所以我们没有使用单引号或双引号定义其值10。由于dname和location是String类型属性，因此我们使用单引号或双引号定义其值10。

注意 - 要定义字符串类型属性值，我们需要使用单引号或双引号。

创建一个标签，即“Dept” 创建一个节点，即“dept” 创建三个属性，即deptno, dname, location

二、匹配语句（**match**）

1. `MATCH (dept:Dept)`



因此MATCH 经常需要与其他的语句配合才可以使用。

如： `match (n) return n`

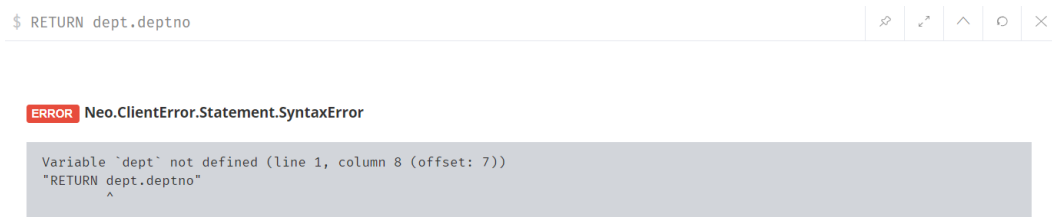
```
# 查询Dept下的内容
MATCH (dept:Dept) return dept

# 查询Employee标签下 id=123, name="Lokesh"的节点
MATCH (p:Employee {id:123,name:"Lokesh"}) RETURN p

## 查询Employee标签下name="Lokesh"的节点，使用（where命令）
MATCH (p:Employee)
WHERE p.name = "Lokesh"
RETURN p
```

三、返回语句（**return**）

`RETURN dept.deptno`



`dept`是节点名称 `deptno`是`dept`节点的属性名称

如果发现错误消息，它告诉我们，我们不能单独使用`RETURN`子句。我们应该既`MATCH`使用或`CREA`

四、**RETURN** 和 **MATCH**命令语法：

MATCH:

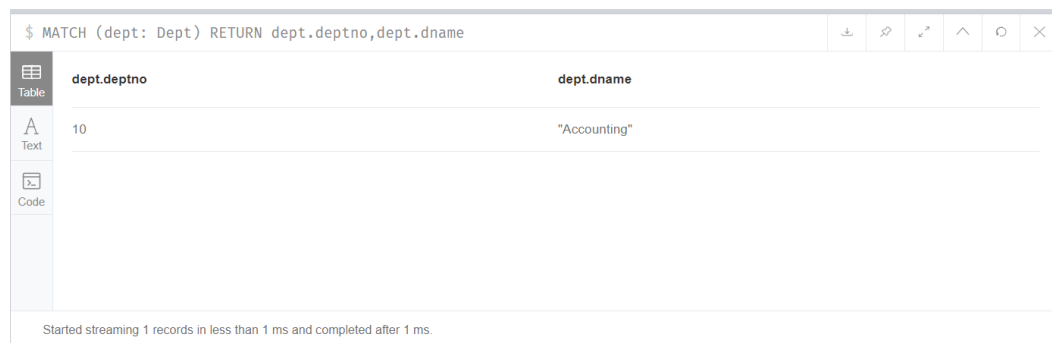
```
MATCH
(
  <node-name>:<label-name>
)
```

RETURN:

```
RETURN
  <node-name>.<property1-name>,
  ...
  <node-name>.<propertyn-name>
```

本示例演示如何从数据库检索Dept节点的一些属性（deptno, dname）数据。注-结点包含3个属性：deptno, dname, location。然而在这个例子中，我们感兴趣的是只查看两个属性数据：

```
MATCH (dept: Dept) RETURN dept.deptno,dept.dname
```



dept.deptno	dept.dname
10	"Accounting"

此示例演示如何从数据库检索Dept节点的数据，而无需指定其属性。

```
MATCH (dept: Dept)
RETURN dept
```

五、CREATE+MATCH+RETURN命令

演示如何使用属性和这两个节点之间的关系创建两个节点

5.1 注一我们将创建两个节点：客户节点(*Customer*)和信用卡节点(*CreditCard*)。

- 客户节点包含：ID，姓名，出生日期属性
- CreditCard节点包含：id, number, cvv, expiredate属性
- 客户与信用卡关系：DO_SHOPPING_WITH
- CreditCard到客户关系：ASSOCIATED_WITH

我们将在以下步骤中处理此示例：

- 创建客户Customer节点
- 创建CreditCard节点
- 观察先前创建的两个节点：Customer和CreditCard
- 创建客户和CreditCard节点之间的关系
- 查看新创建的关系详细信息
- 详细查看每个节点和关系属性

```
CREATE (e:Customer{id:"1001",name:"Abc",dob:"01/10/1982"})
```

- e是节点名称
- 在这里Customer是节点标签名称
- id, name和dob是Customer节点的属性名称

CREATE

```
(cc:CreditCard{id:"5001",number:"1234567890",cvv:"888",expiredate:"20/17"})
```

- 这里cc是一个节点名
- 这里CreditCard是节点标签名称
- id, number, cvv和expiredate是CreditCard节点的属性名称

5.2 查看客户节点和CreditCard详细节点信息

```
MATCH (e:Customer)
RETURN e.id,e.name,e.dob
```

```
MATCH (cc:CreditCard)
RETURN cc.id,cc.number,cc.cvv,cc.expiredate
```