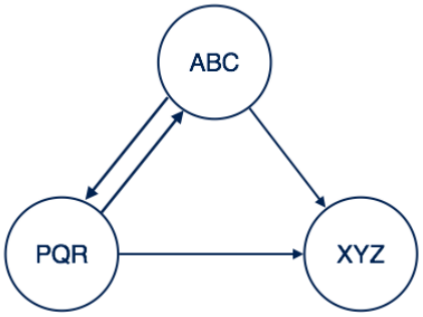
Neo4j 开源图形数据库

图形数据库数据模型主要构建块：

- 节点Nodes
- 关系Relationships
- 属性Properties
- 标签Label
- 数据浏览器Broswer（neo4j.bat console) http：// localhost：7474 / browser /



Neo4j CQL

遵循SQL语法

常用的Neo4j CQL命令/条款如下：

S.No.	CQL命令/条	用法
1	CREATE 创建	创建节点，关系和属性
2	MATCH 匹配	检索有关节点，关系和属性数据
3	RETURN 返回	返回查询结果
4	WHERE 哪里	提供条件过滤检索数据
5	DELETE 删除	删除节点和关系
6	REMOVE 移除	删除节点和关系的属性
7	ORDER BY	排序检索数据

	以...排序	
8	SET 组	添加或更新标签

创建节点

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

例:CREATE (p:Person { name:'Father', age:33 });

创建标签

CREATE (<node-name>:<label-name1>:<label-name2>.....:<label-namen>)

例：CREATE (p:Person)

查询节点

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

例：MATCH(p:Person) RETURN p

```
match (p: Person {name:'Father'}) return p
match (p: Person {name:'Father'}) return p.name,p.age
MATCH (emp:Employee)
WHERE emp.name = 'Abc' OR emp.name = 'Xyz'
RETURN emp
```

ID

```
match (m: Person {name:"ivy"}) return m,ID(m)
match (n) where ID(n) = 17846 return n
```

关系查询

Match (n:org)-[:observer]->(end:person) where n.name='juxinli' return end

Match (n:org)-[r:observer]->(p:person)
where n.name='juxinli' and p.name='maomao'
return p

创建节点、标签+关系

CREATE (<node1-name>:<label1-name>{<Property-name>:<Property-Value>})-
    [(<relationship-name>:<relationship-label-name>{<Property-name>:
<Property-Value>})]
    ->(<node2-name>:<label2-name>{<Property-name>:<Property-Value>})

例：create (f:person {name:"father"})-[r:married{status:1,time:'2017-02-23 12:12:12'}]->(m:person {name:"mother"})

创建关系(neo4j只支持方向(单/双)关系)

MATCH (<node1-label-name>:<node1-name>),(<node2-label-name>:
<node2-name>)
WHERE <condition>
CREATE (<node1-label-name>)-[<relationship-label-name>:
<relationship-name>
    {<relationship-properties>}]->(<node2-label-name>)

例：MATCH (m:Message),(c:Language)
    WHERE m.title = 'Welcome' AND c.name = 'Java'
    CREATE (m)-[:ACCESSED_FROM]->(c);

DELETE＆REMOVE

Neo4j CQL REMOVE命令用于

- 删除节点或关系的标签
- 删除节点或关系的属性

Neo4j CQL DELETE和REMOVE命令之间的主要区别 -

- DELETE操作用于删除节点和关联关系。
- REMOVE操作用于删除标签和属性。

Neo4j CQL DELETE和REMOVE命令之间的相似性 -

- 这两个命令不应单独使用。
- 两个命令都应该与MATCH命令一起使用。
-

删除节点

MATCH (e: Employee) DELETE e

start n=node(*) match (n)-[r:observer]-() delete n,r（删除所有节点、关系）

删除节点/关系

MATCH (cc: CreditCard)-[rel]-(c:Customer)
DELETE cc,c,rel
（先删除关系才能删除节点）

删除属性

MATCH (dc:DebitCard)
REMOVE dc.cvv
RETURN dc

删除标签

MATCH (m:Movie)
REMOVE m:Picture

SET

- 向现有节点或关系添加新属性
- 添加或更新属性值

```
MATCH (dc:DebitCard)
SET dc.atm_pin = 3456
RETURN dc
```

排序ORDER BY

```
MATCH (emp:Employee)
RETURN emp.empid,emp.name,emp.salary,emp.deptno
ORDER BY emp.name       #升序
```

```
MATCH (emp:Employee)
RETURN emp.empid,emp.name,emp.salary,emp.deptno
ORDER BY emp.name DESC      #降序
```

UNION

将两组结果中的公共行组合并返回到一组结果中。 不从两个节点返回重复的行。
结果列类型和来自两组结果的名称必须匹配，这意味着列名称应该相同，列的数据类型应该相同。

```
MATCH (cc:CreditCard)
RETURN cc.id as id,cc.number as number,cc.name as name,
   cc.valid_from as valid_from,cc.valid_to as valid_to
UNION
MATCH (dc:DebitCard)
RETURN dc.id as id,dc.number as number,dc.name as name,
          dc.valid_from as valid_from,dc.valid_to as valid_to
#由于节点名称前缀不同，需要使用AS语句

MATCH (cc:CreditCard)
RETURN cc.id as id,cc.number as number,cc.name as name,
cc.valid_from as valid_from,cc.valid_to as valid_to
UNION ALL          #UNION ALL不会过滤重复行
MATCH (dc:DebitCard)
RETURN dc.id as id,dc.number as number,dc.name as name,
dc.valid_from as valid_from,dc.valid_to as valid_to
```

LIMIT与SKIP

- “LIMIT”子句过滤或限制查询返回的行数。 它修剪CQL查询结果集底部的结果
- "SKIP"子句修剪CQL查询结果集顶部的结果

```
MATCH (emp:Employee)
RETURN emp
LIMIT 2    #显示顶部两条数据

MATCH (emp:Employee)
RETURN emp
SKIP 2     #显示底部两条数据
```

MERGE

MERGE = CREATE + MATCH

```
MERGE (gp2:GoogleProfile2{ Id: 201402,Name:"Nokia"})
 #检查该节点在数据库中是否可用。 如果不存在，创建新节点。
```

IN

```
MATCH (e:Employee)
WHERE e.id IN [123,124]
RETURN e.id,e.name,e.sal,e.deptno
```

字符串函数列表

S.No.	功能	描述
1。	UPPER	它用于将所有字母更改为大写字母。
2。	LOWER	它用于将所有字母改为小写字母。
3。	SUBSTRING	它用于获取给定String的子字符串。
4。	REPLACE	它用于替换一个字符串的子字符串。

```
MATCH (e:Employee)
RETURN e.id,UPPER(e.name),e.sal,e.deptno

MATCH (e:Employee)
RETURN e.id,LOWER(e.name),e.sal,e.deptno

MATCH (e:Employee)
RETURN e.id,SUBSTRING(e.name,0,2),e.sal,e.deptno  #查询name前两个字母
```

聚合函数列表

S.No.	聚集功能	描述
1。	COUNT	它返回由MATCH命令返回的行数。
2。	MAX	它从MATCH命令返回的一组行返回最大值。
3。	MIN	它返回由MATCH命令返回的一组行的最小值。
4。	SUM	它返回由MATCH命令返回的所有行的求和值。

5。	AVG	它返回由MATCH命令返回的所有行的平均值。

MATCH (e:Employee) RETURN COUNT(*) #返回employee节点数

MATCH (e:Employee)
RETURN MAX(e.sal),MIN(e.sal) #返回e.sal的最大值与最小值

MATCH (e:Employee)
RETURN SUM(e.sal),AVG(e.sal) #返回最大值与平均值

关系函数列表

S.No.	功能	描述
1。	STARTNODE	它用于知道关系的开始节点。
2。	ENDNODE	它用于知道关系的结束节点。
3。	ID	它用于知道关系的ID。
4。	TYPE	它用于知道字符串表示中的一个关系的TYPE。

STARTNODE(查找关系的开始节点from)

```
MATCH (a)-[movie:ACTION_MOVIES]->(b)
RETURN STARTNODE(movie)
```

ENDNODE(查找关系的结束节点to)

```
MATCH (a)-[movie:ACTION_MOVIES]->(b)
RETURN ENDNODE(movie)
```

ID与TYPE(查找关系的ID与TYPE)

```
MATCH (a)-[movie:ACTION_MOVIES]->(b)
RETURN ID(movie),TYPE(movie)
```

索引

CREATE INDEX ON :Customer (name)
DROP INDEX ON : :Customer (name)

UNIQUE

```
CREATE CONSTRAINT ON (cc:CreditCard)
ASSERT cc.number IS UNIQUE     #对number属性进行UNIQUE约束,使其不重复

DROP CONSTRAINT ON (cc:CreditCard)
ASSERT cc.number IS UNIQUE
```