neo4j学习

简介

开源图形数据库

- no-Schema
- no-SQL

图形

一组节点和连接这些节点的关系

- 属性用来表示数据，属性的格式是健值对
- 属性存在于节点和关系中

属性在节点中

属性在关系中

图形数据库

以图形结构的形式存储数据的数据库，以节点、关系和属性的形式存储应用程序的数据

- RDBMS以表的行、列形式存储数据
- GDBMS以图形的形式存储数据

图形数据库主要用于存储更多的连接数据，如果用RDMMS数据库存储连接数据，它不能提供用于遍历大量数据的性能

数据模型（属性图模型）

- 节点和关系都包含属性
- **属性**是健值对
- **关系**
  - 关系连接节点
  - 关系具有方向：单向、双向
  - 每个关系包含开始节点和结束节点

环境搭建

操作系统：macOS Mojave 10.14.6

- 下载neo4j-community-3.4.15-unix.tar
- jdk版本jdk-8u221-macosx-x64

> mac jdk版本切换

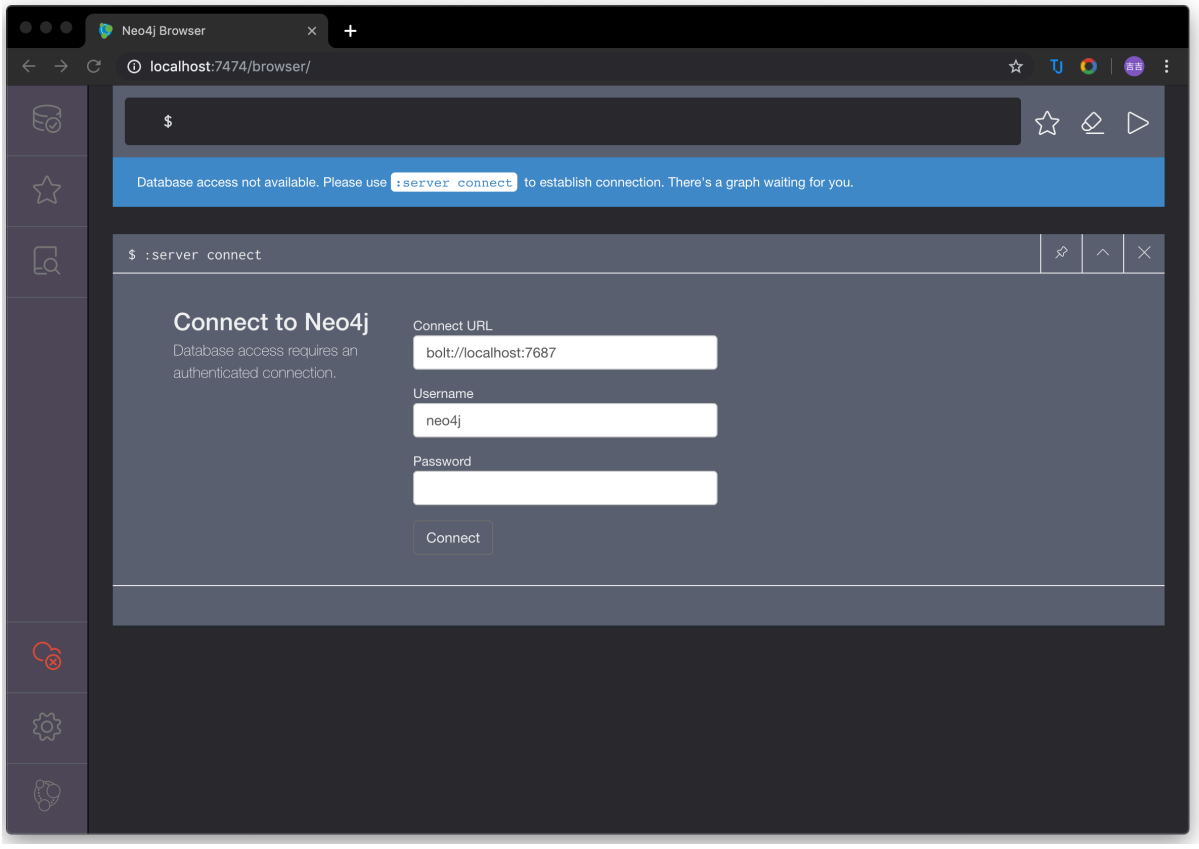
```
# 创建.bash_profile配置文件
vim ~/.bash_profile

# 设置jdk版本
export
JAVA8_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_221.jdk/Content
s/Home
export JAVA_HOME=$JAVA8_HOME

# 动态切换JAVA_HOME的配置
alias jdk8='export JAVA_HOME=$JAVA8_HOME'

# 重新执行.bash_profile文件
source .bash_profile
```

- cd neo4j-community-3.4.15/
- cd bin/
- ./neo4j start
- 在浏览器中访问url: localhost:7474/



- ./neo4j stop

> 通过shell脚本快捷启动neo4j

```
vim .bash_aliases
alias neo4j='~/neo4j-community-3.4.15/bin/neo4j'
source .bash_aliases
```

构建模块

节点

- 图表的基本单位
- 包含属性（键值对）

Node Name = "Employee"

关系

- 连接两个节点
- 包含起始节点和结束节点
- 也可以包含属性（键值对）

1. Emp是起始节点，Dept是结束节点
2. WORKS_FOR是节点之间的关系
3. 这个关系有一个属性ID=123
4. 箭头表示从Emp到Dept的关系

属性

- 键值对
- 用来描述节点和关系
- 格式：`Key = value`
  - Key是字符串
  - value可以是任何Neo4j数据类型
- **Neo4j将数据存储在属性中**

标签

- 将公共名称和一组节点或关系相关联
- 节点或关系可以包含一个或多个标签

CQL

命令

命令	作用
CREATE	创建节点、关系、属性
MATCH	检索节点、关系、属性数据
RETURN	返回查询结果
WHERE	条件过滤
DELETE	删除节点和关系
REMOVE	删除（节点和关系的）属性
ORDER BY	排序检索数据
SET	添加/更新标签

清空Neo4j数据库

match (n) detach delete n

函数

函数	作用
String	使用String字面量
Aggregation	对CQL查询结果执行聚合操作
Relationship	获取关系的细节，如startnode，endnode等

数据类型

数据类型	描述
boolean	布尔：true，false
byte	8位整数
short/int/long	16/32/64位整数
float/double	32/64位浮点数
char	16位字符
String	字符串

CREATE

创建节点

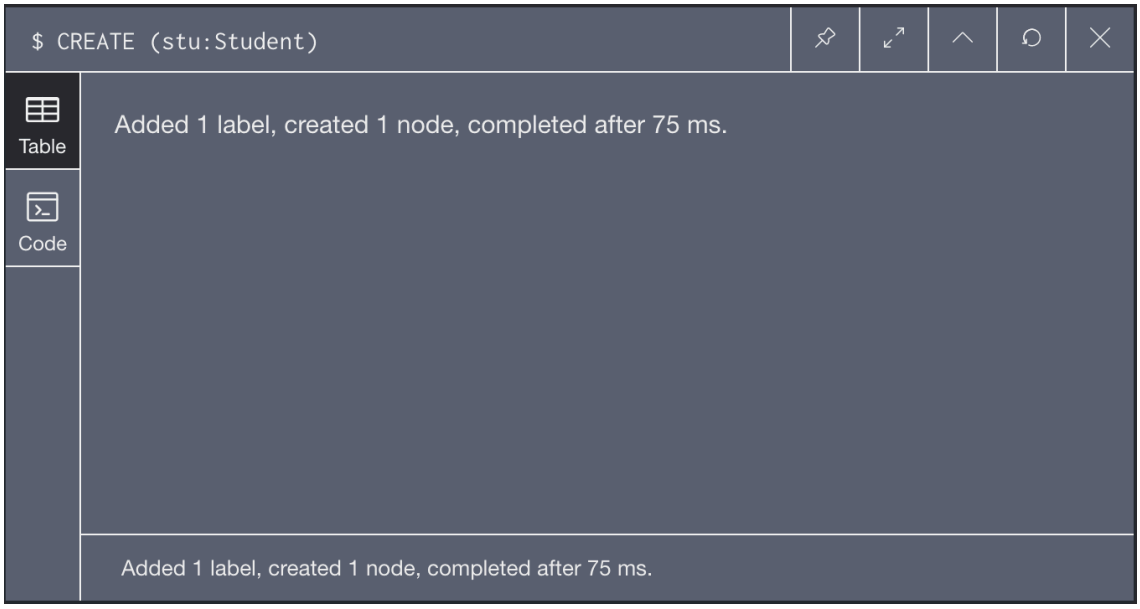
```
CREATE (
   <node-name>:<label-name>
   {
      <Property1-name>:<Property1-Value>   //属性是键值对
      ........
      <Propertyn-name>:<Propertyn-Value>
   }
)

//node-name: 节点，将节点详细信息存储在Database.As中，不能使用它访问节点的详细信息
//label-name: 标签，作为节点名称的别名，应该使用标签名访问节点的详细信息
```

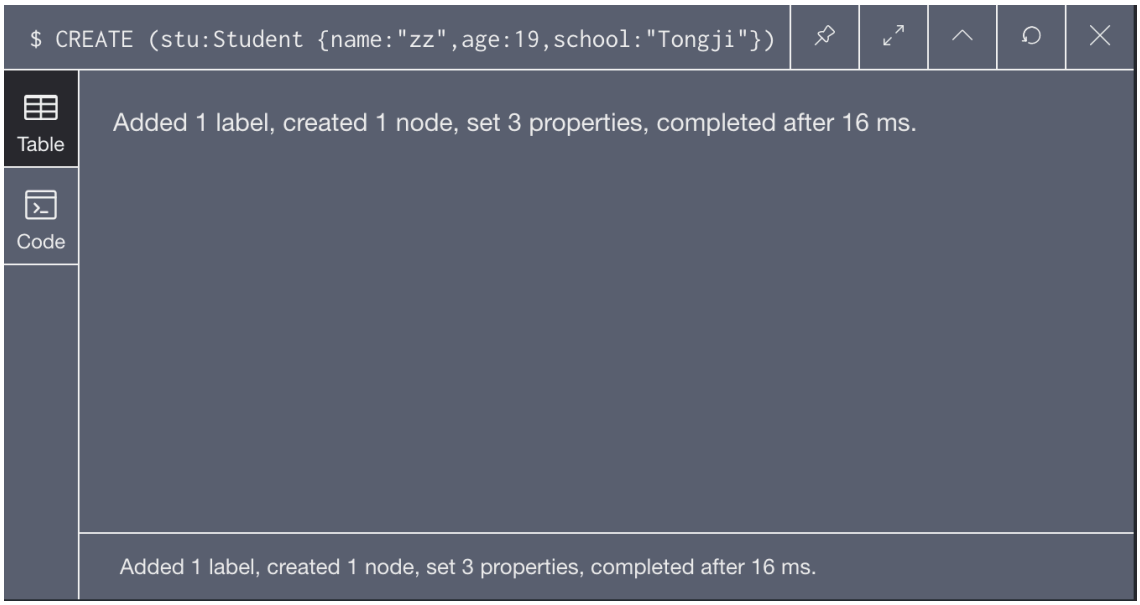
创建关系

```
MATCH (<node1-label-name>:<node1-name>),(<node2-label-name>:<node2-name>)
CREATE
  (<node1-label-name>)-[<relationship-label-name>:<relationship-name>{<define-properties-list>}]->(<node2-label-name>)
RETURN <relationship-label-name>
```

- **创建节点**
  - 没有属性的节点



  - 有属性的节点



- **创建关系:** 关系必须是定向的
  - 两个现有节点
  - 两个新节点

Graph
Table
Text
Code

yt

```
{
  "name": "ZhangYiteng"
}
```

Started streaming 1 records after 1 ms and completed after 1 ms.

```
$ match (zz:Boy),(yt:Girl) create (zz)-[bf:IS_BOYFRIEND]…
```

Table
Code

Created 2 relationships, completed after 3 ms.

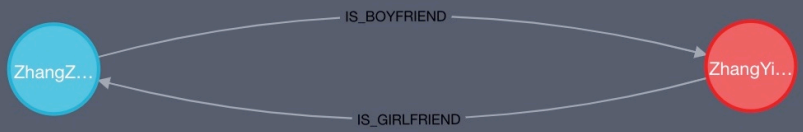
Created 2 relationships, completed after 3 ms.

```
$ match p=(zz)-[bf:IS_BOYFRIEND]->(yt) return p
```

Graph
Table
Text
Code

*(2) Boy(1) Girl(1)

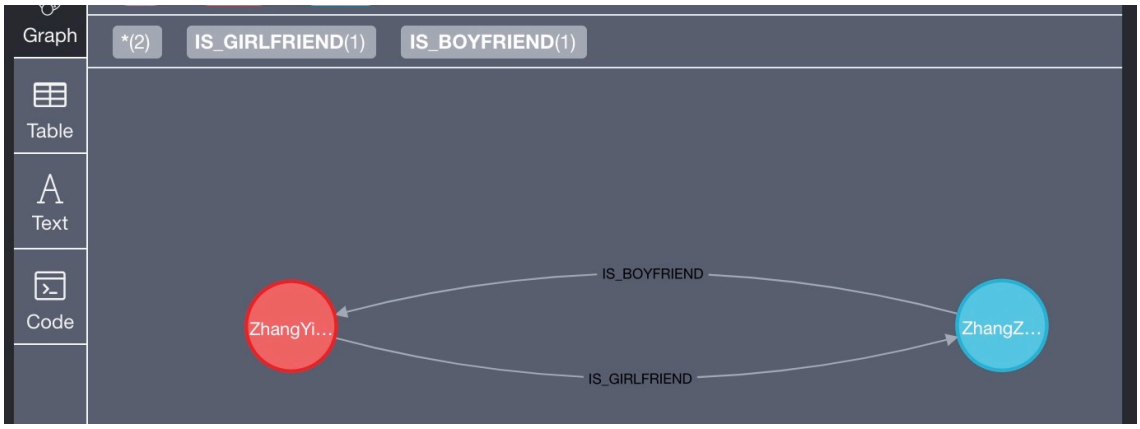
*(2) IS_BOYFRIEND(1) IS_GIRLFRIEND(1)



Displaying 2 nodes, 2 relationships.

```
$ match p=(yt)-[gf:IS_GIRLFRIEND]->(zz) return p
```

*(2) Girl(1) Boy(1)

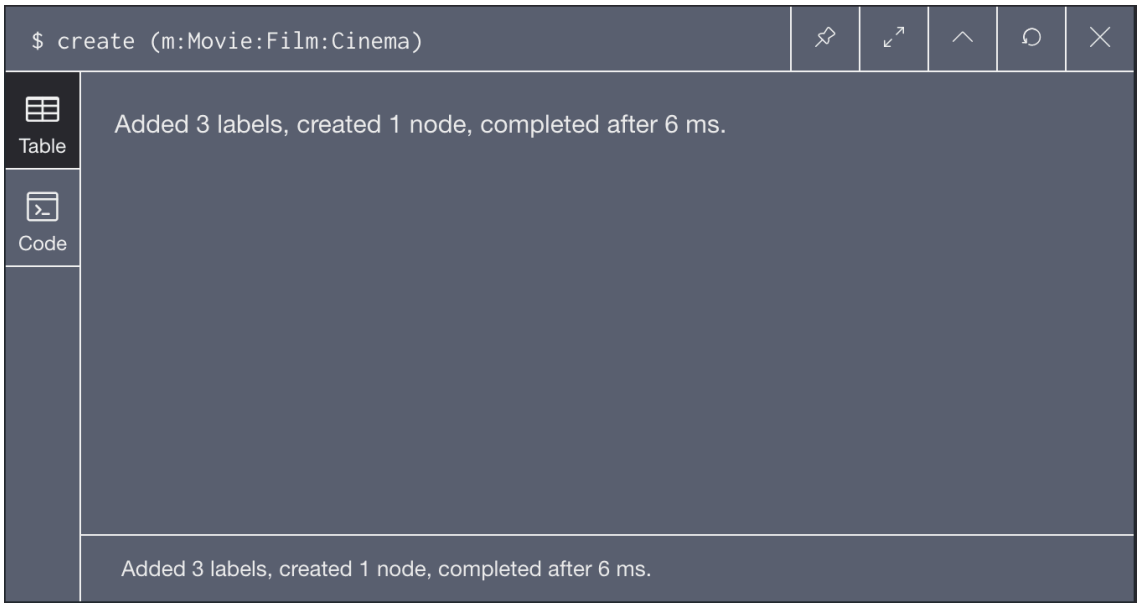


  - WHERE子句的两个退出节点

  - 无属性的关系
  - 有属性的关系

- **创建标签**
  - 为节点创建
  - 为关系创建

  - 单个标签
  - 多个标签

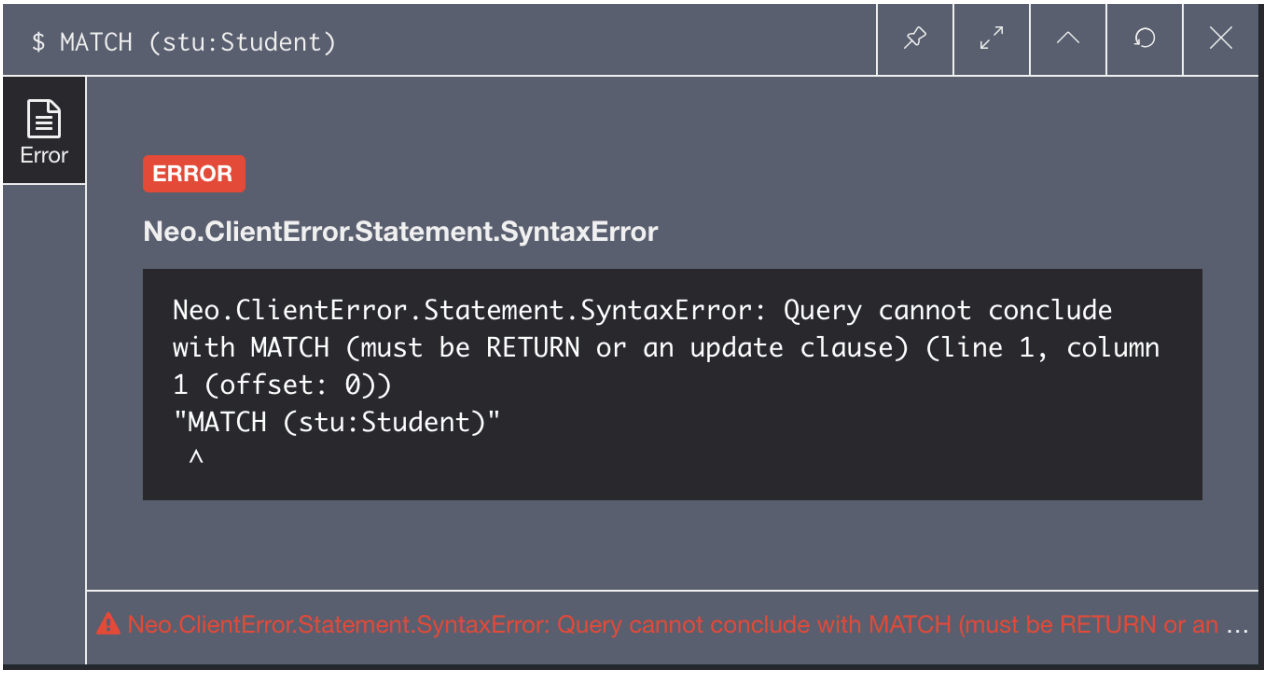


MATCH

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

- 获取有关节点和属性的数据
- 获取有关节点、关系和属性的数据

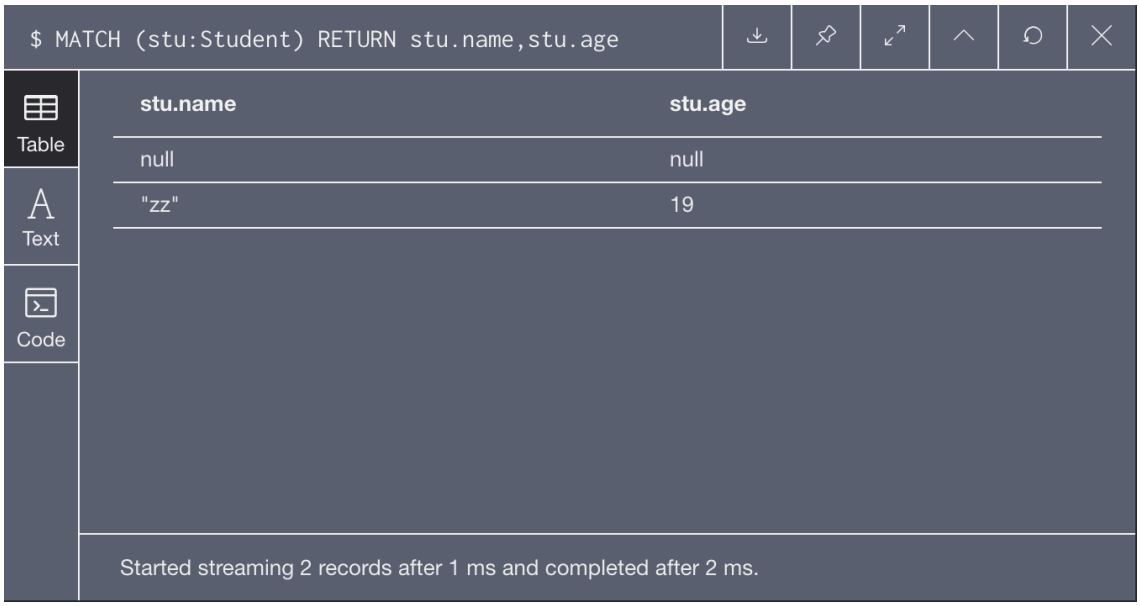
MATCH单独使用会引发InvalidSyntax错误



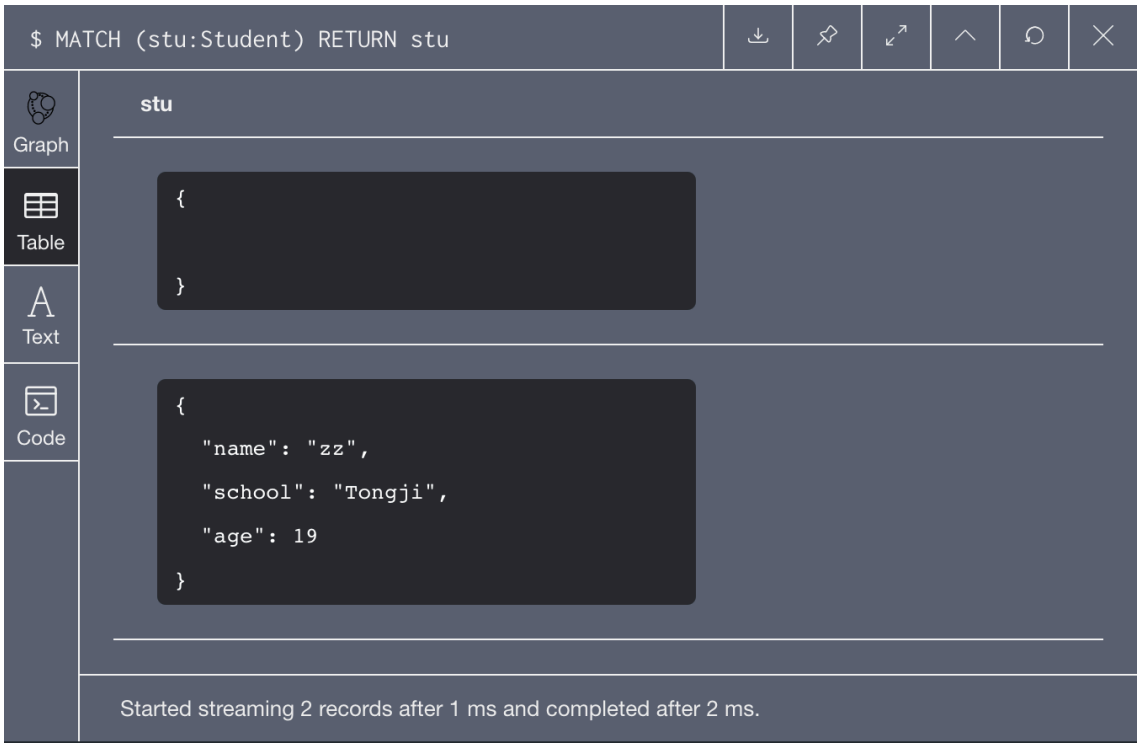
RETURN

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

- 检索属性
  - 节点的某些属性



  - 节点的所有属性

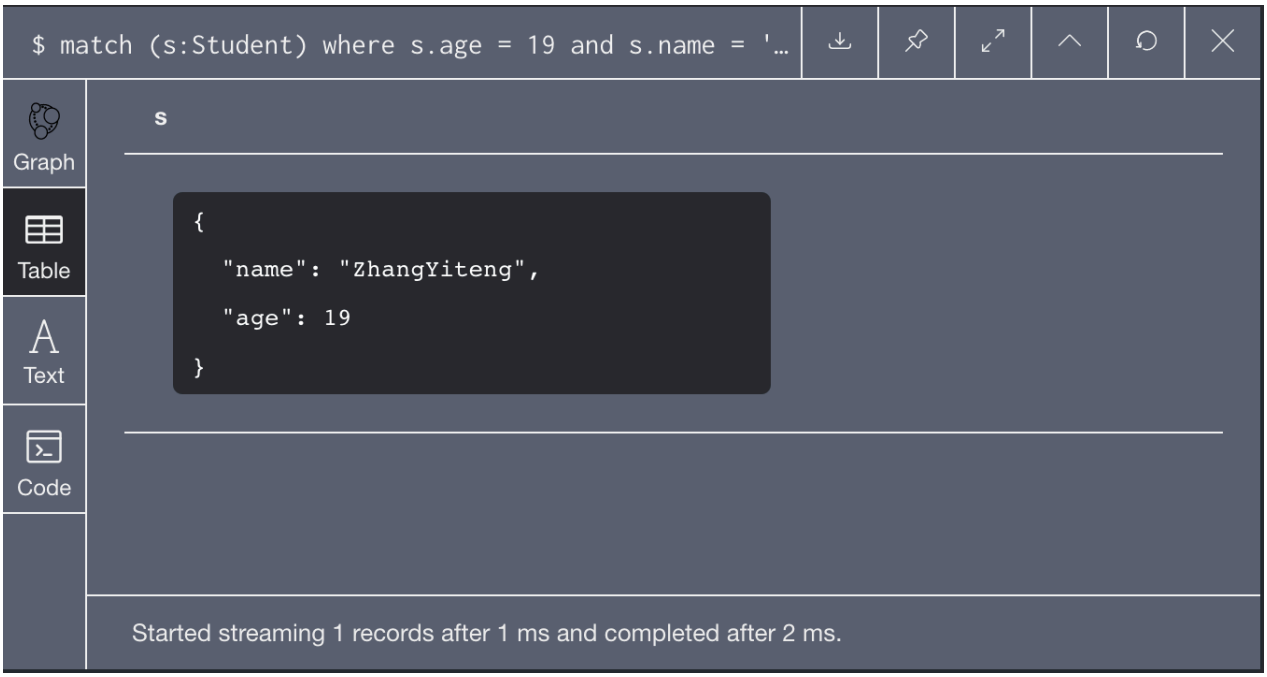


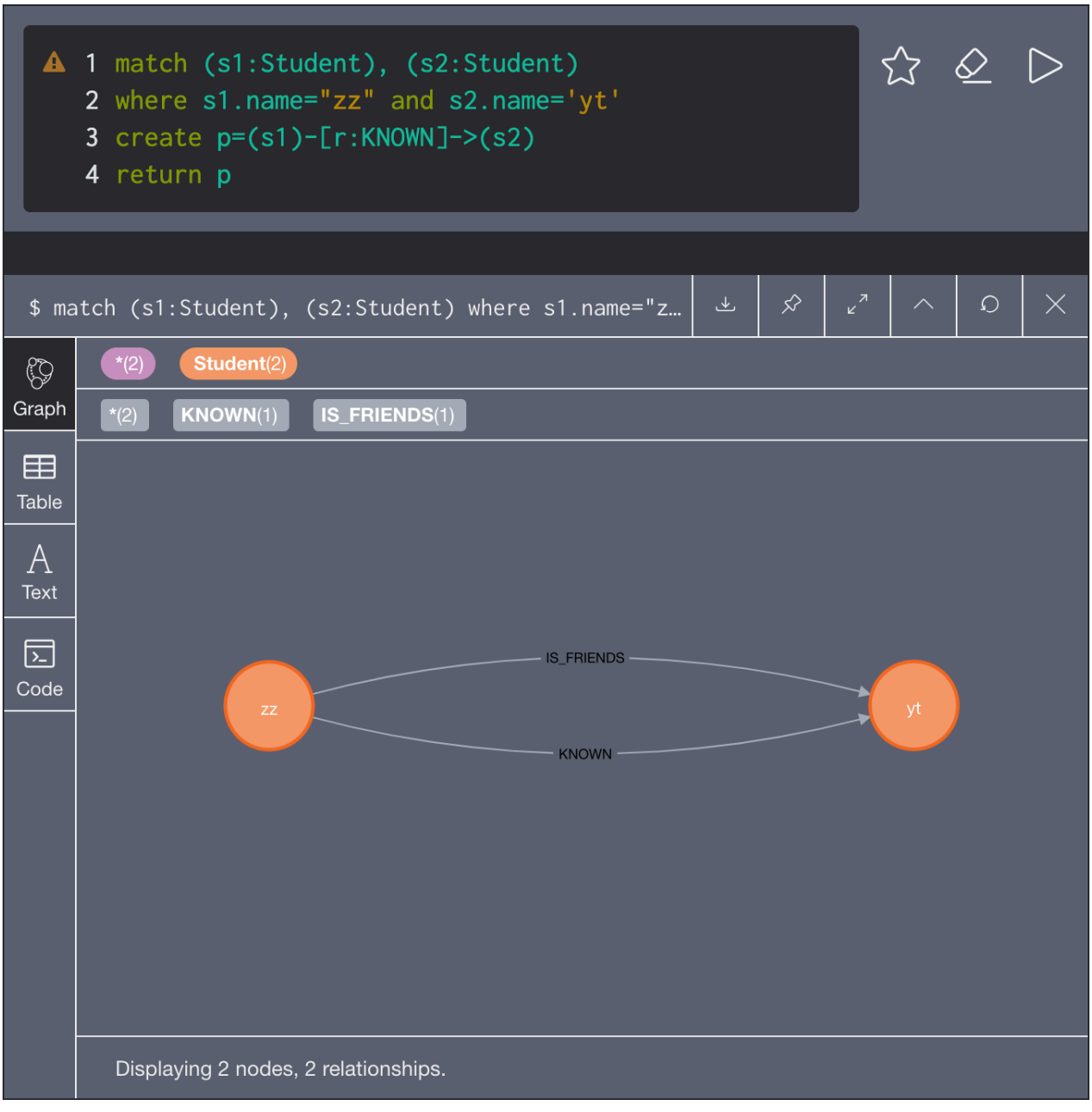
- 节点和关联关系的某些关系
- 节点和关联关系的所有关系

WHERE

```
WHERE <condition> <boolean-operator> <condition>
WHERE <property-name> <comparison-operator> <value>
```

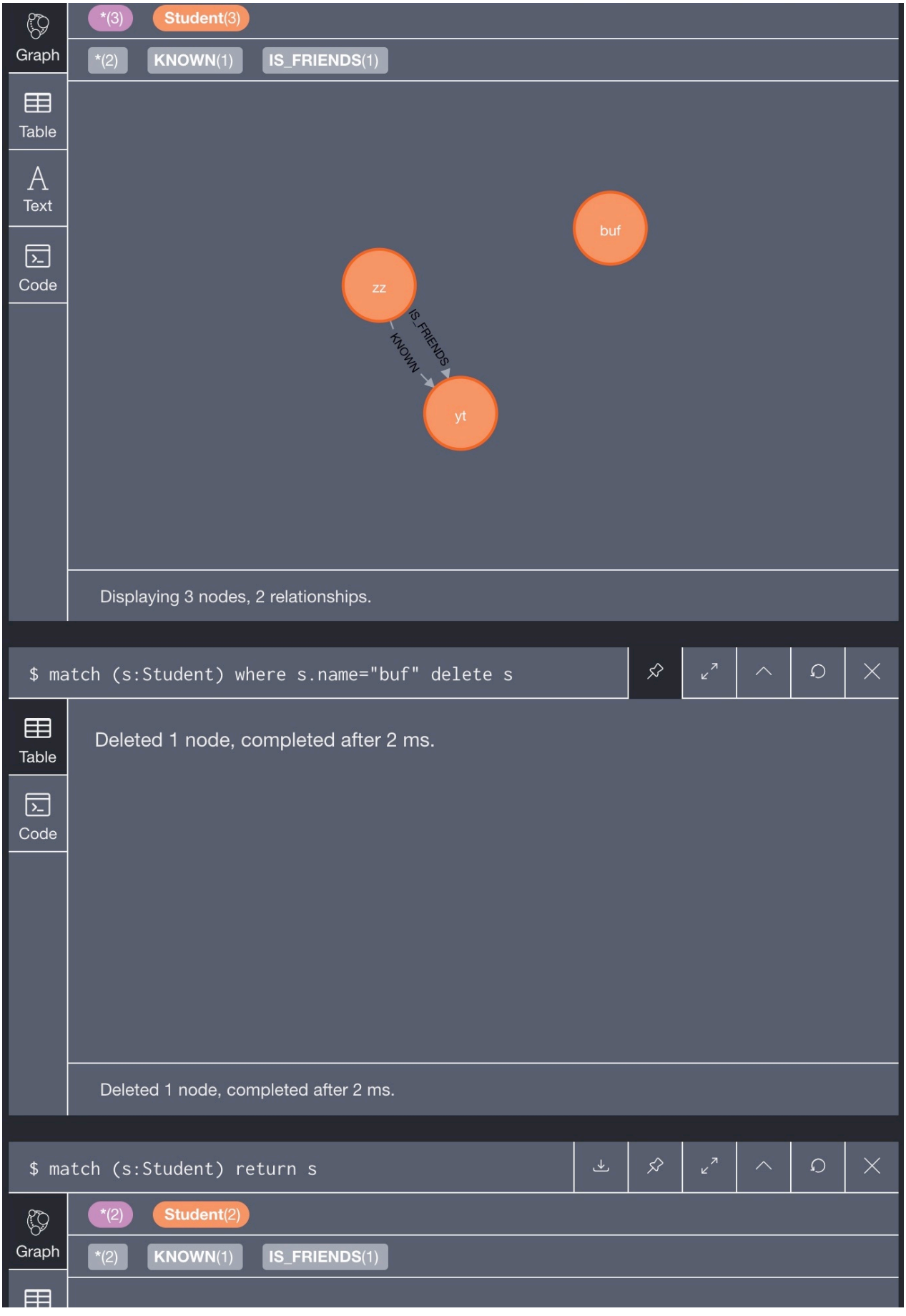
逻辑运算符	比较运算符
AND	=
OR	<>
NOT	<
XOR	>
	<=
	>=


$ match (s:Student) where s.age = 19 and s.name = '...
Graph
Table
Text
Code
s
{
"name": "ZhangYiteng",
"age": 19
}
Started streaming 1 records after 1 ms and completed after 2 ms.


1 match (s1:Student), (s2:Student)
2 where s1.name="zz" and s2.name='yt'
3 create p=(s1)-[r:KNOWN]->(s2)
4 return p
$ match (s1:Student), (s2:Student) where s1.name="z...
Graph
Table
Text
Code
*(2)
Student(2)
*(2)
KNOWN(1)
IS_FRIENDS(1)
zz
IS_FRIENDS
KNOWN
yt
Displaying 2 nodes, 2 relationships.

DELETE

- 删除节点
- 删除节点及相关节点和关系



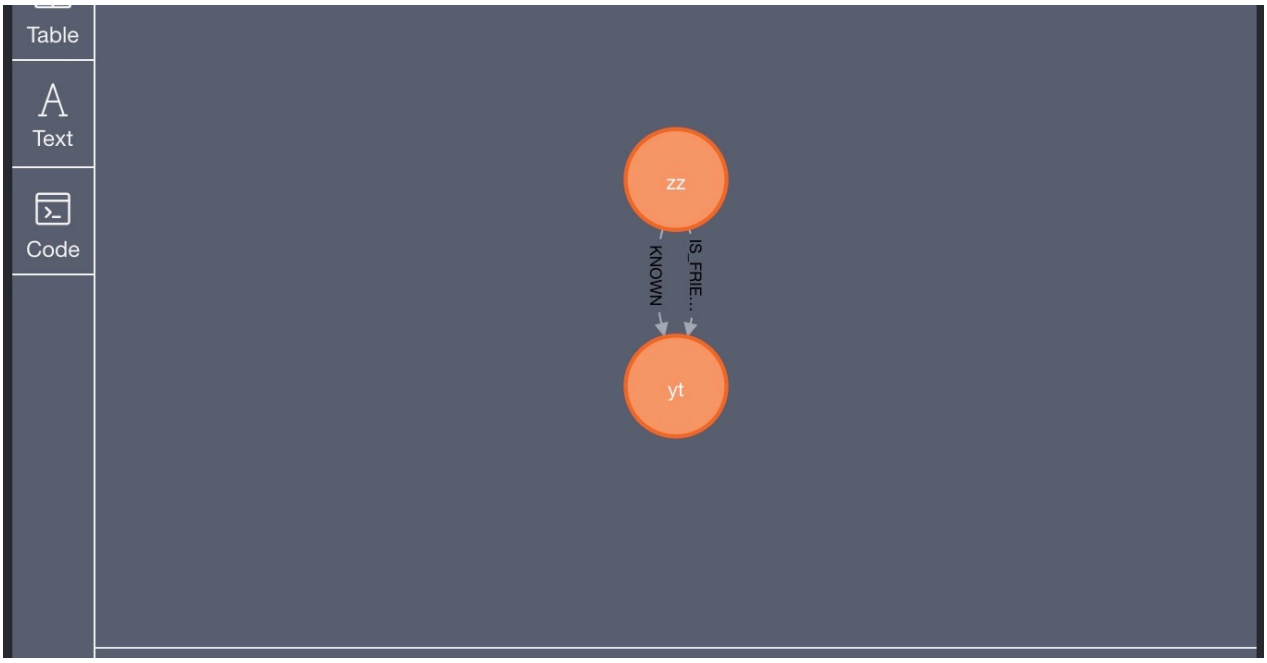
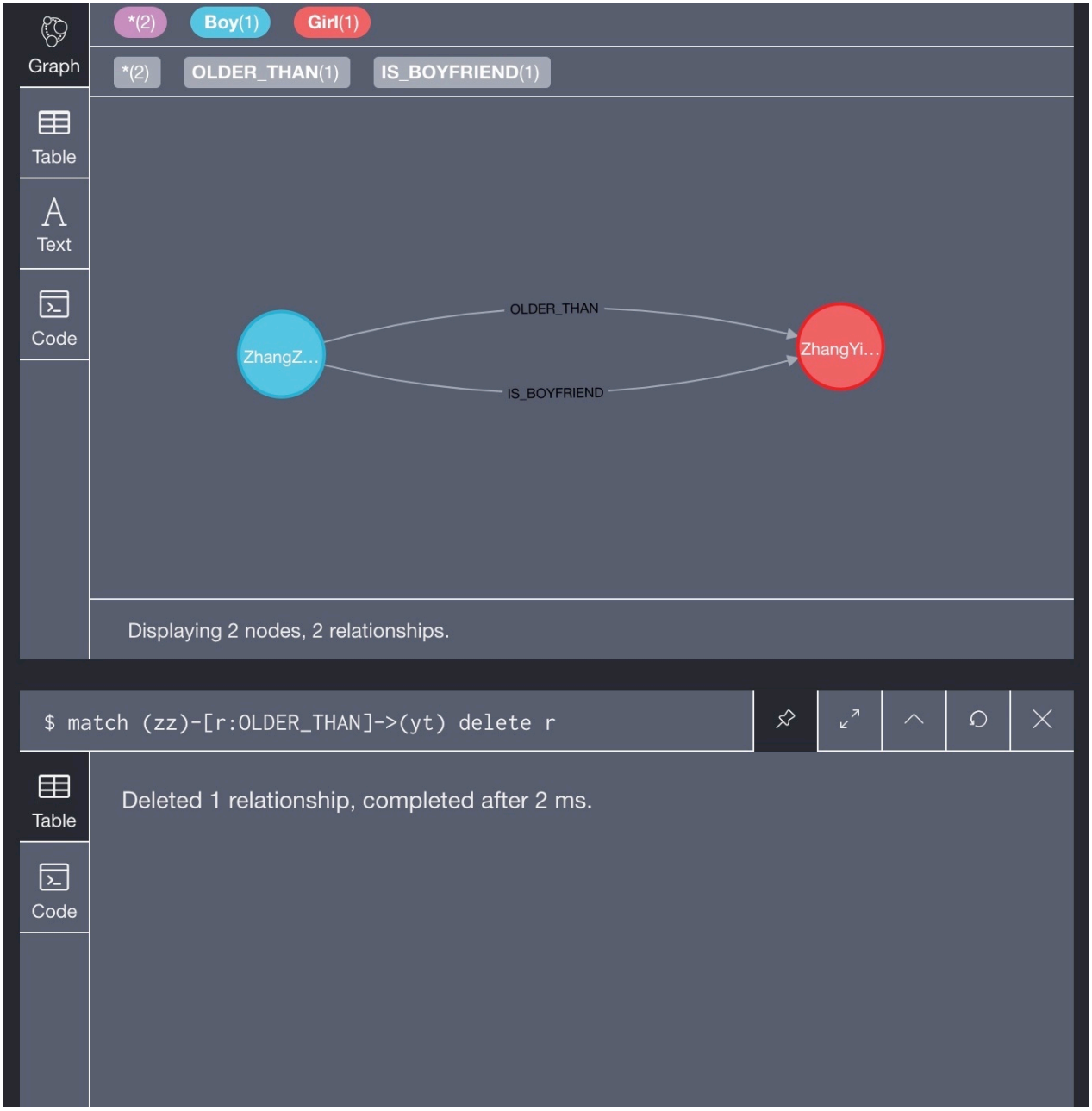
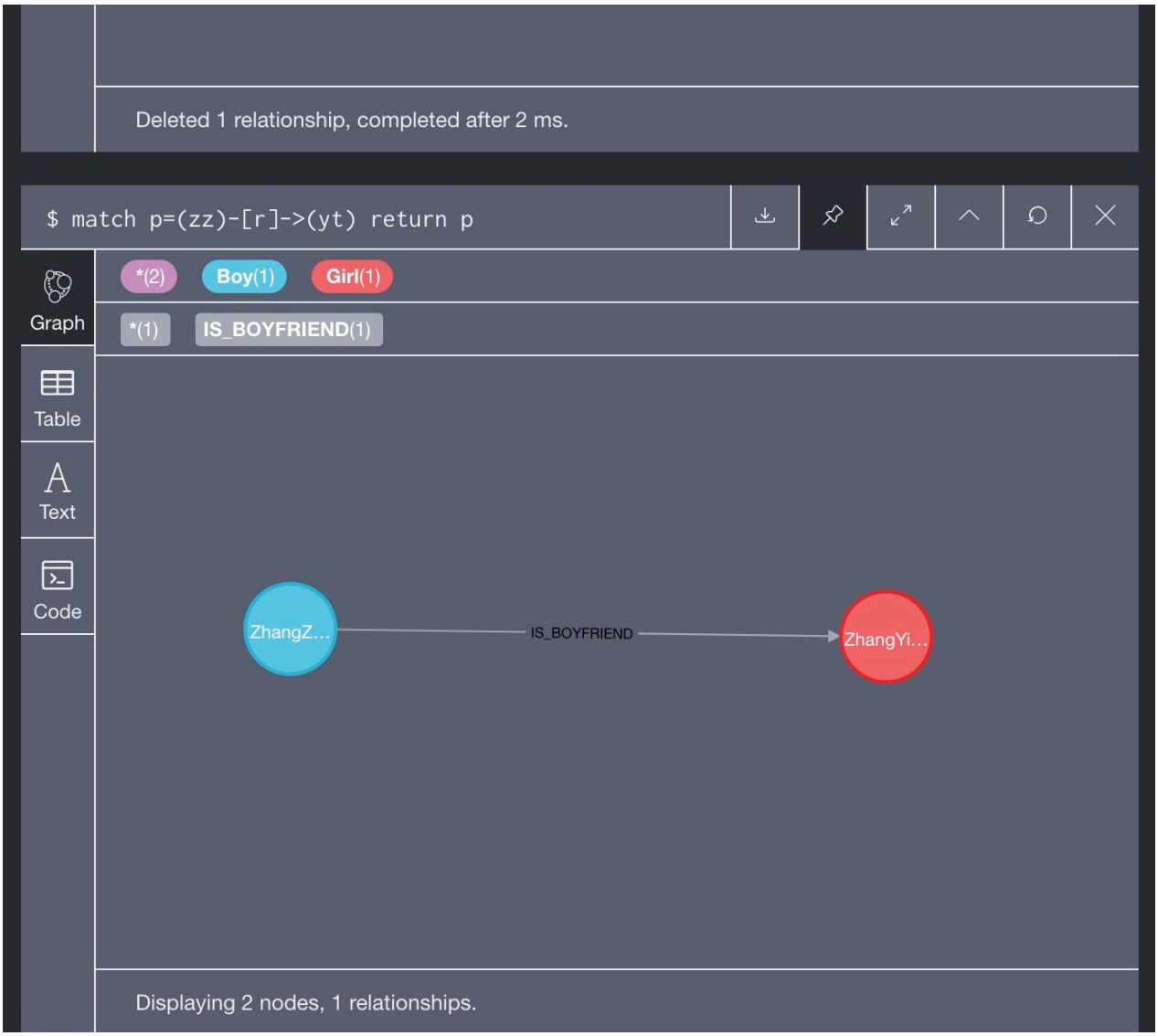


Table
Text
Code
zz
KNOWN
IS_FRIE...
yt

删除某个节点

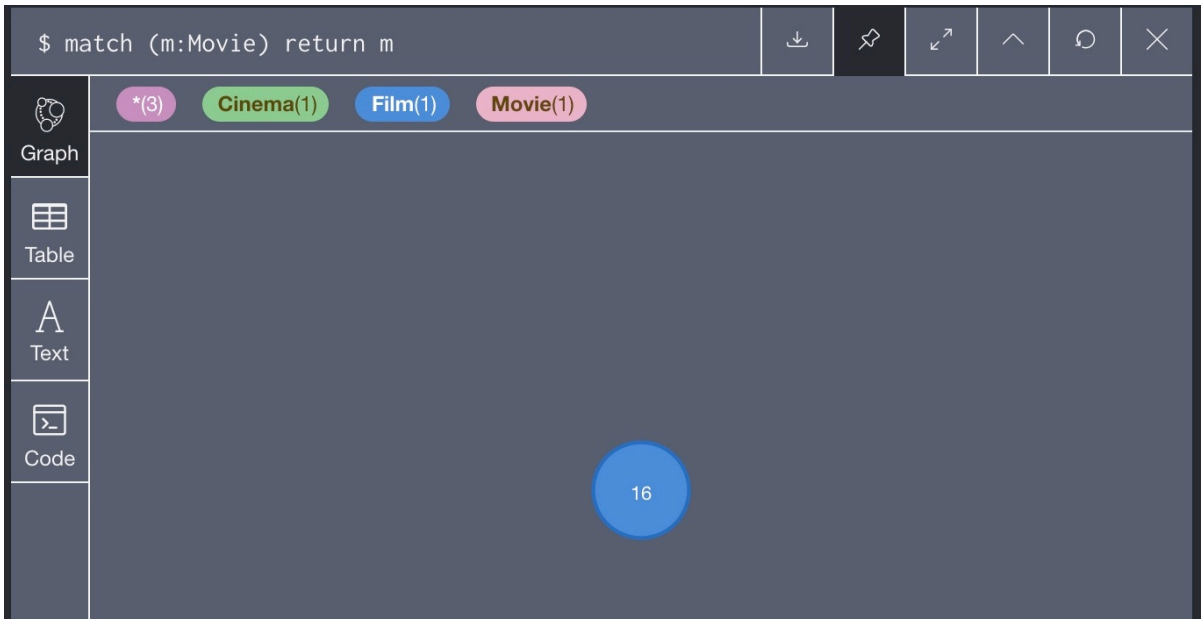

Graph
Table
Text
Code
*(2)
Boy(1)
Girl(1)
*(2)
OLDER_THAN(1)
IS_BOYFRIEND(1)
ZhangZ...
OLDER_THAN
IS_BOYFRIEND
ZhangYi...
Displaying 2 nodes, 2 relationships.
$ match (zz)-[r:OLDER_THAN]->(yt) delete r
Table
Code
Deleted 1 relationship, completed after 2 ms.

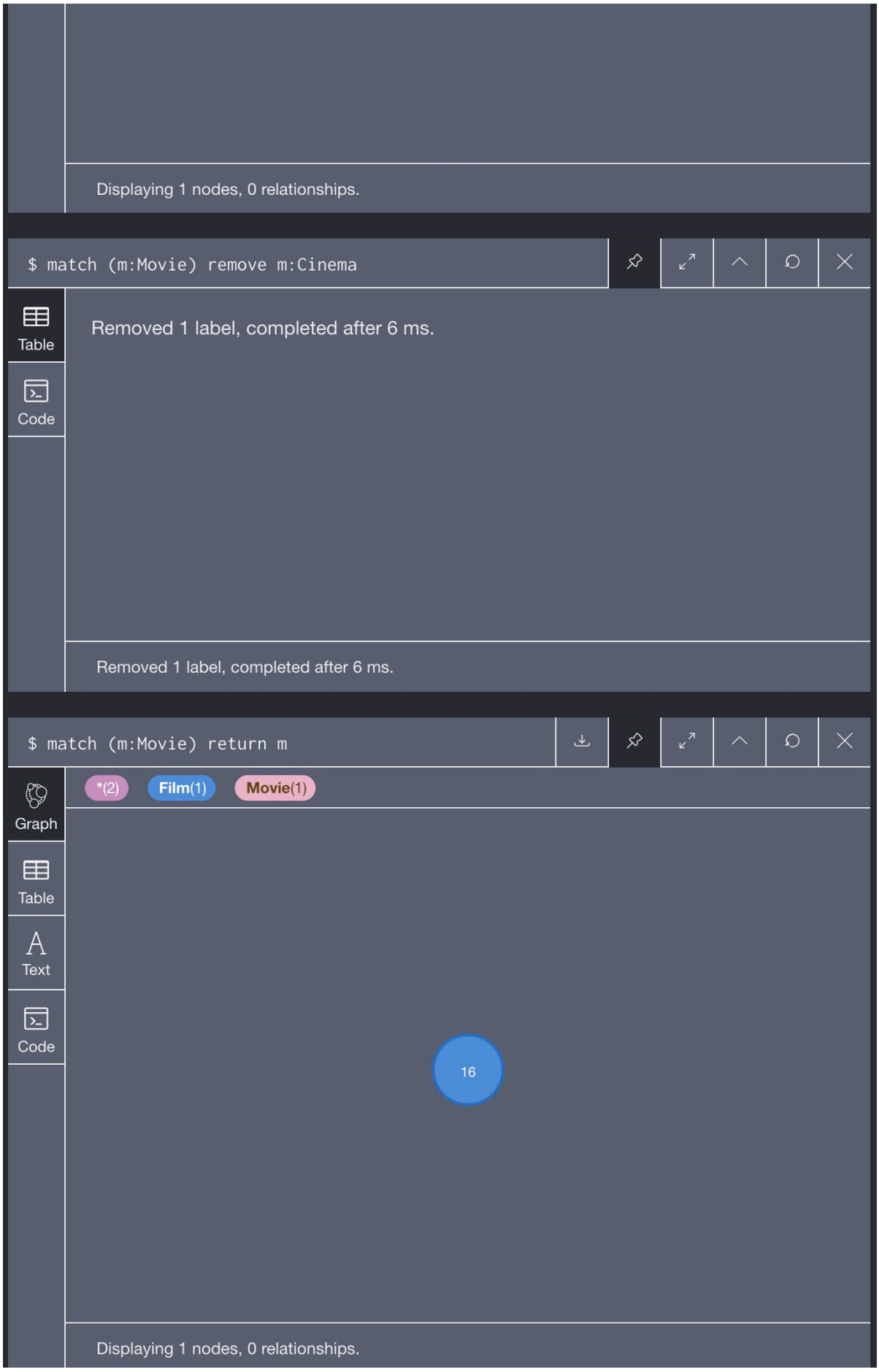


删除节点间的某个属性

REMOVE

- 删除标签




Displaying 1 nodes, 0 relationships.
$ match (m:Movie) remove m:Cinema
Table
Code
Removed 1 label, completed after 6 ms.
Removed 1 label, completed after 6 ms.
$ match (m:Movie) return m
Graph
Table
Text
Code
*(2)
Film(1)
Movie(1)
16
Displaying 1 nodes, 0 relationships.

- 删除属性

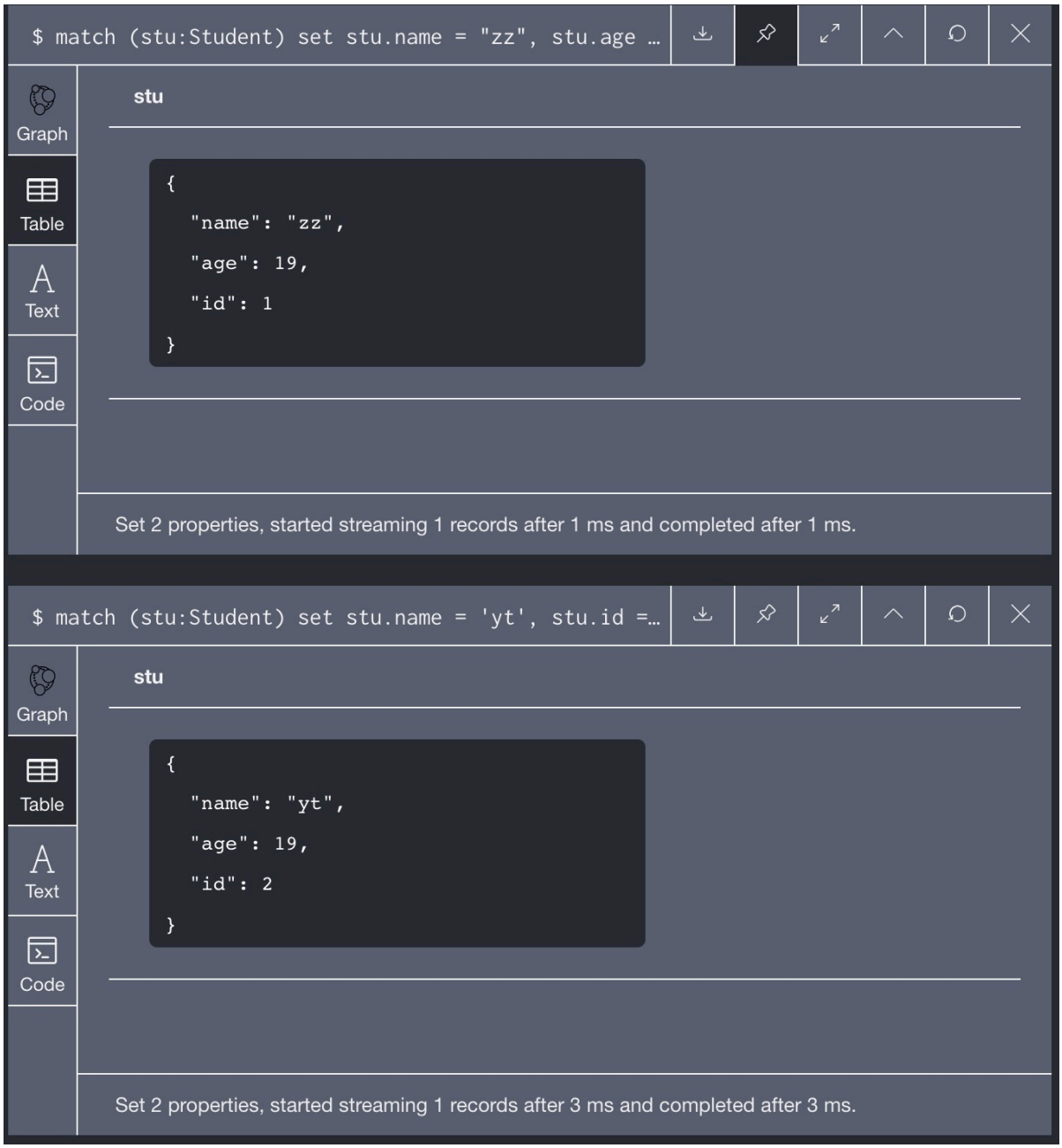

$ match (book:Book) return book
Graph
Table
Text
Code
book
{
"title": "Python从入门到入土",
"price": 35.2
}
Started streaming 1 records after 1 ms and completed after 1 ms.
$ match (book {title:"Python从入门到入土"}) remove …
Graph
Table
Text
Code
book
{
"title": "Python从入门到入土"
}
Set 1 property, started streaming 1 records after 5 ms and completed after 5 ms.

SET

- 添加属性


$ match (stu:Student) set stu.name = "zz", stu.age …
Graph
Table
Text
Code
stu
{
"name": "zz",
"age": 19,
"id": 1
}
Set 2 properties, started streaming 1 records after 1 ms and completed after 1 ms.
$ match (stu:Student) return stu
Graph
Table
Text
Code
stu
{
"id": 1
}
Started streaming 1 records after 1 ms and completed after 1 ms.

- 更新属性

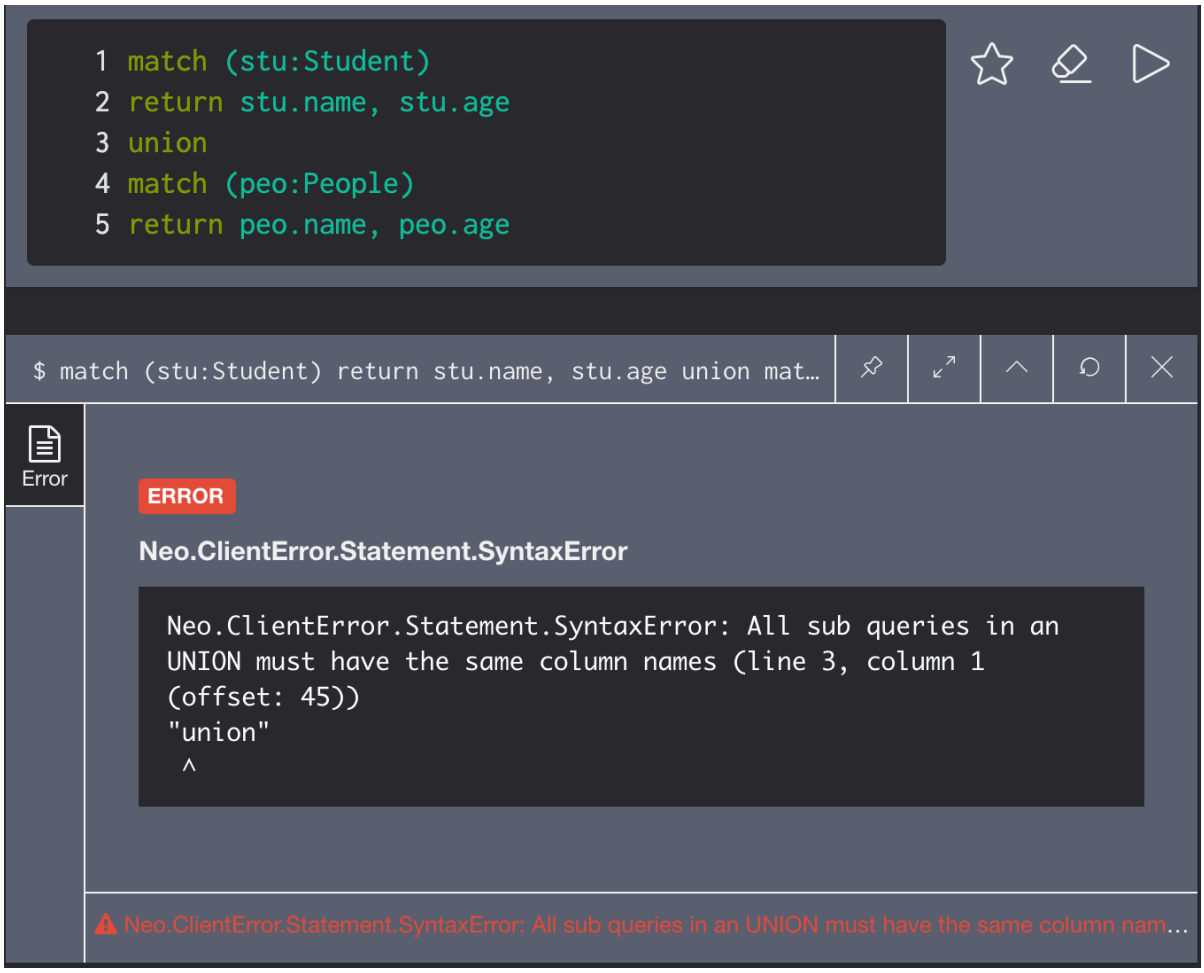


ORDER BY

```
MATCH...
RETURN...
ORDER BY XXX          //默认升序排列
ORDER BY XXX DESC     //降序
```

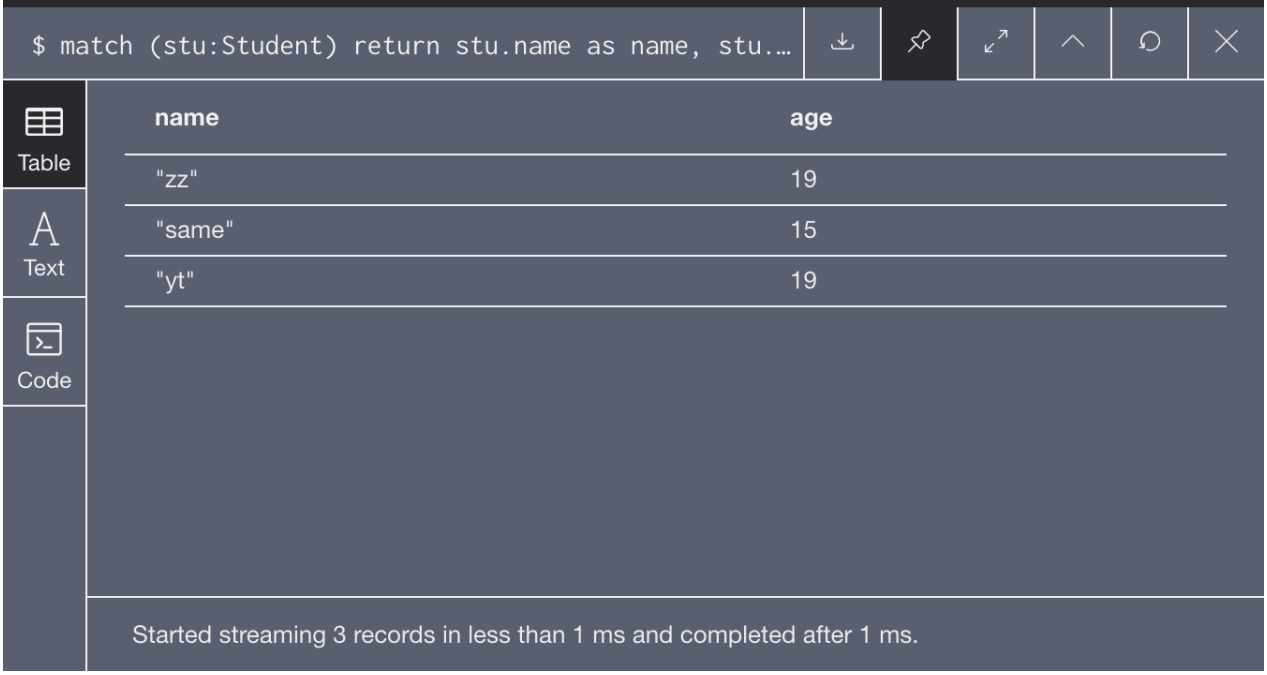
UNION

- 将结果中的公共行组合并返回（不返回重复的行）
  - 想要保留重复行：UNION ALL
- 列名称和列数据类型必须相同

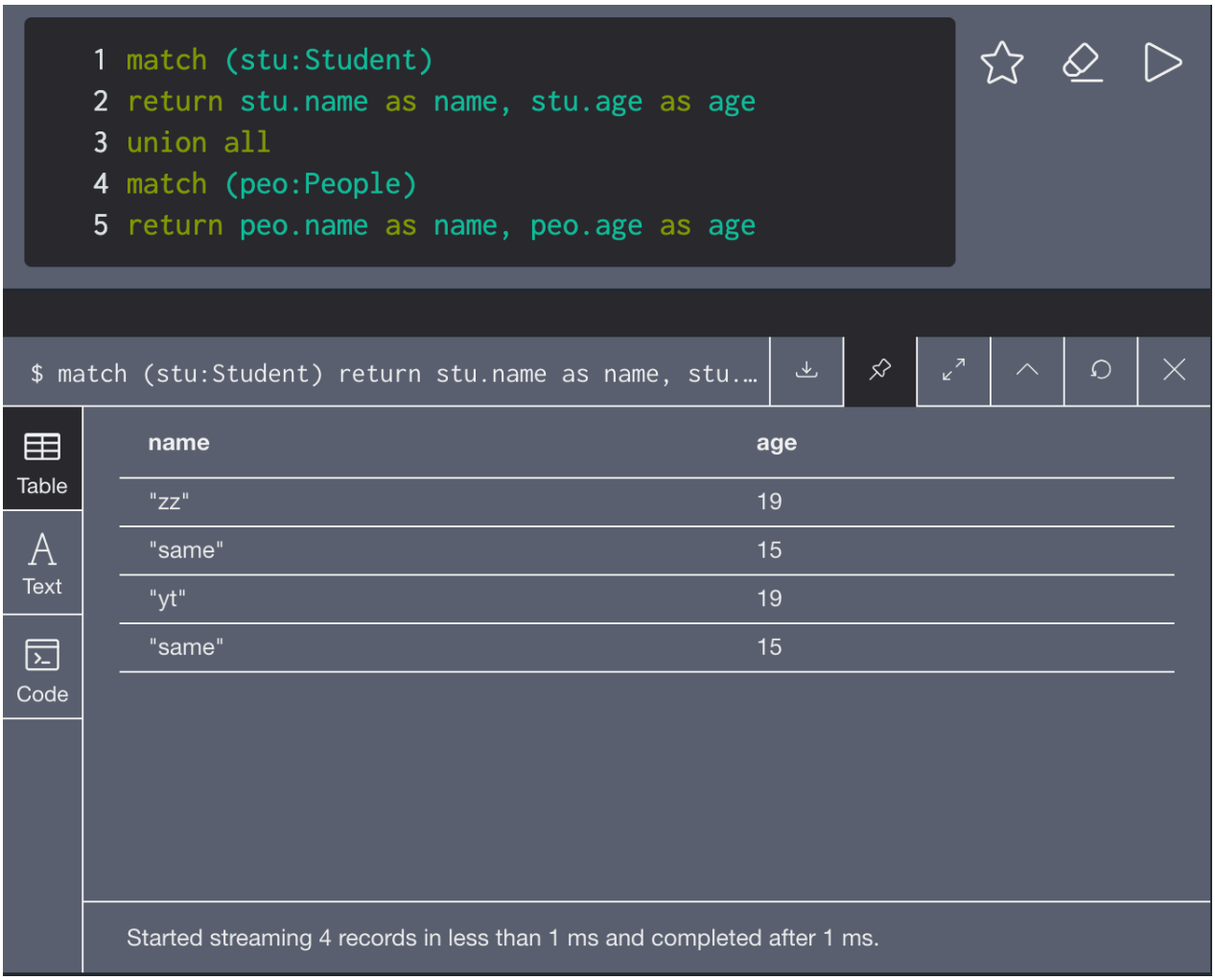


虽然有相同的属性名，但是节点名不同

```
match (stu:Student)
return stu.name as name, stu.age as age
union
match (peo:People)
return peo.name as name, peo.age as age
```



union



union all

LIMIT

- 过滤或限制查询返回的行数
- 修建结果集底部的结果（把底部多的去掉）

SKIP

- 修建结果集顶部的结果

```
MATCH...
RETURN
LIMIT 25
SKIP 25
```

MERGE

- MERGE = CREATE + MATCH
  - 在图中搜索，如果存在，则返回结果
  - 不存在，创建新的节点/关系并返回

> CREATE 总是向数据库中添加新的节点，如果信息一模一样也会插入

NULL值

- 创建现有节点标签名单未指定属性值的节点时，将创建一个具有NULL值的新节点

```
//过滤NULL值
MATCH (e:E)
WHERE e IS NOT NULL
```

IN

```
//过滤在范围中的节点
WHERE e.value in [5,10]
```