

**Московский государственный технический университет им.Н.Э.Баумана
кафедра "Системы обработки информации и управления"**



Постреляционные базы данных

к лабораторной работе №6

**Лабораторная работа «Работа с графовой NOSQL БД на примере Neo4j»
по дисциплине «Постреляционные базы данных»**

Инструктор : Мария Валерьевна

Email:2623859464@qq.com

Студент: Ван Чаочао

группа ИУ5И-22М

2022/05/10

Цель работы:

1. Изучить модель представления данных и способы работы с графовыми БД NoSql.
2. Освоить методы создания графовой БД и языки запросов к ней.
3. Получить навыки работы с графовой БД Neo4j.

Время выполнения:

Время выполнения лабораторной работы 4 часа.

Исходные данные:

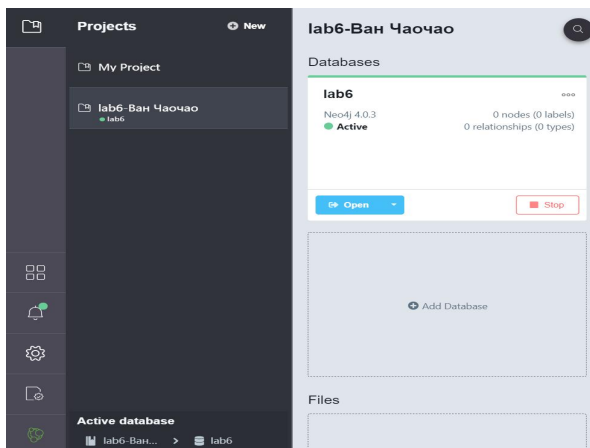
1. (локальный ПК) Дистрибутив и документация - <https://neo4j.com/>.
2. (доп., кратко) Руководство по установке и началу работы - <https://ru.bmstu.wiki/Neo4j>
3. Фаулер, Мартин, Садаладж, Прамодкумар Дж. NoSQL: новая методология разработки нереляционных баз данных. : Пер. с англ. - М.: ООО "И.Д. Вильямс", 2013г.

Пункты задания для выполнения:

Задание 1. Создание БД (базовая часть) 创建数据库 (基础部分)

Создать в Neo4j базу данных по теме своего ДЗ. Определить набор узлов, задать их свойства и метки.

Создать в Neo4j базу данных:



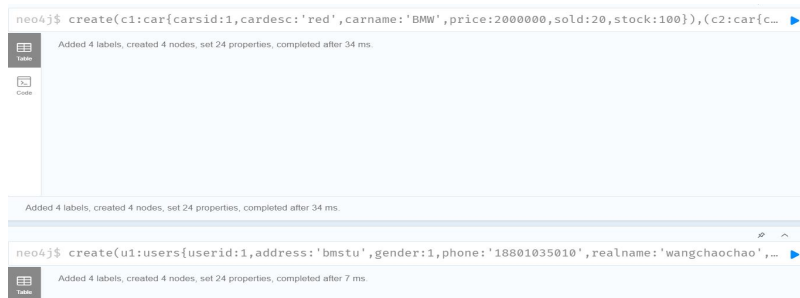
Добавление узлов:

Users:

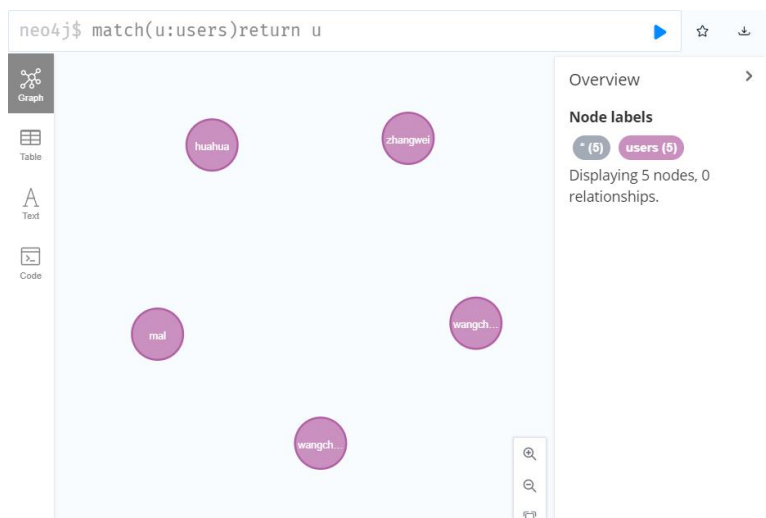
```
create (u1:users {userid:1, address:'bmstu', gender:1, phone:'18801035010', realname:'wangchaochao', username:'wcc'}), (u2:users {userid:2, address:'bmstu', gender:1, phone:'79269525693', realname:'zhangwei', username:'zw'}), (u3:users {userid:3, address:'bmstu', gender:0, phone:'321654987', realname:'huahua', username:'hh'}), (u4:users {userid:4, address:'bmstu', gender:0, phone:'987654321', realname:'mal', username:'mm'})
```

Car:

```
create(c1:car{carsid:1,cardesc:'red',carname:'BMW',price:2000000,sold:20,stock:100}), (c2:car{carsid:2,cardesc:'black',carname:'Geely',price:1800000,sold:10,stock:80}), (c3:car{carsid:3,cardesc:'wrangler',carname:'Jeep',price:429000000,sold:9,stock:600}), (c4:car{carsid:4,cardesc:'silver',carname:'bently',price:100000000,sold:8,stock:50})
```



Users:

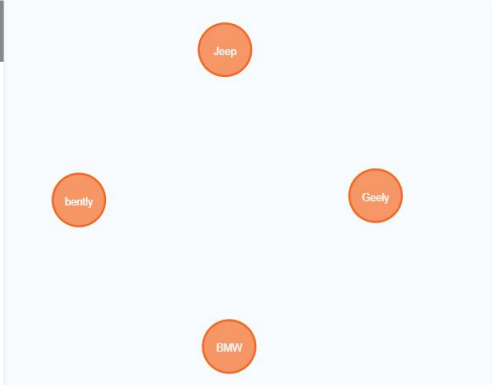


neo4j\$ match(u:users) return u

"u"
{ "address": "bmstu", "gender": 1, "phone": "18801035010", "userid": 1, "realname": "wangchaochao", "username": "wcc" }
{ "address": "bmstu", "gender": 1, "phone": "18801035010", "userid": 1, "realname": "wangchaochao", "username": "wcc" }
{ "address": "bmstu", "gender": 1, "phone": "79269525693", "userid": 2, "realname": "zhangwei", "username": "zw" }
{ "address": "bmstu", "gender": 0, "phone": "321654987", "userid": 3, "realname": "huahua", "username": "hh" }
{ "address": "bmstu", "gender": 0, "phone": "987654321", "userid": 4, "realname": "mal", "username": "mm" }

Car:

neo4j\$ match(c:car) return c



Overview

Node labels

(4) car (4)

Displaying 4 nodes, 0 relationships.

neo4j\$ match(c:car) return c

Graph

Table

Text

Code

```
{ "c"
  { "sold": 20, "cardesc": "red", "stock": 100, "carsid": 1, "price": 2000000, "carname": "BMW" }
  { "sold": 10, "cardesc": "black", "stock": 80, "carsid": 2, "price": 1800000, "carname": "Geely" }
  { "cardesc": "wrangler", "sold": 9, "stock": 600, "price": 429000000, "carsid": 3, "carname": "Jeep" }
  { "cardesc": "silver", "sold": 8, "stock": 50, "price": 100000000, "carsid": 4, "carname": "Bentley" }
```

Продемонстрировать (вывести на экран) содержимое БД (узлы и их свойства), используя команды Match/Where/Return.

```
match (a:car) where a.sold >= 10 return a.carname, a.price, a.sold
```

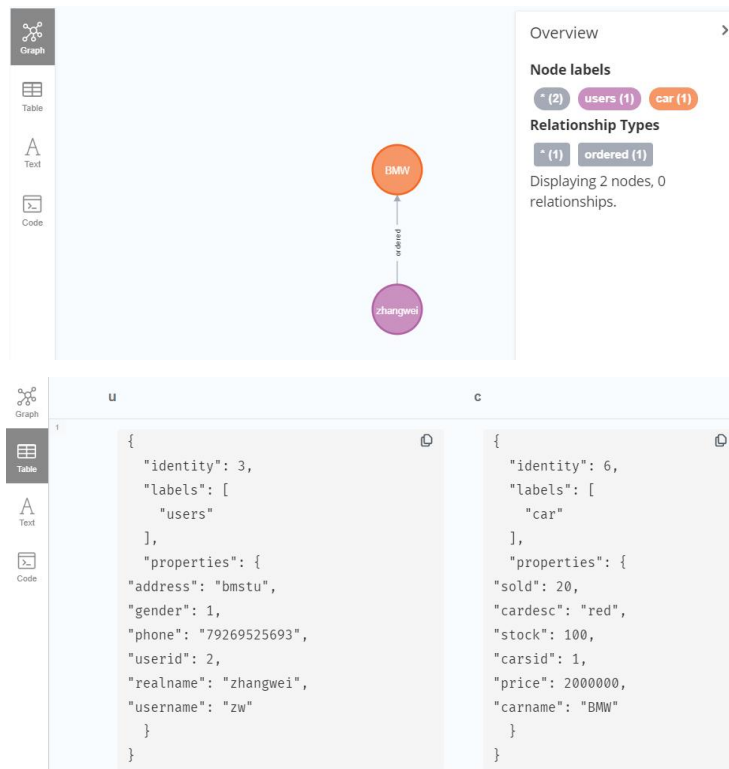
neo4j\$ match (a:car) where a.sold >= 10 return a.carname, a.price, a.sold

	a.carname	a.price	a.sold
1	"BMW"	2000000	20
2	"Geely"	1800000	10

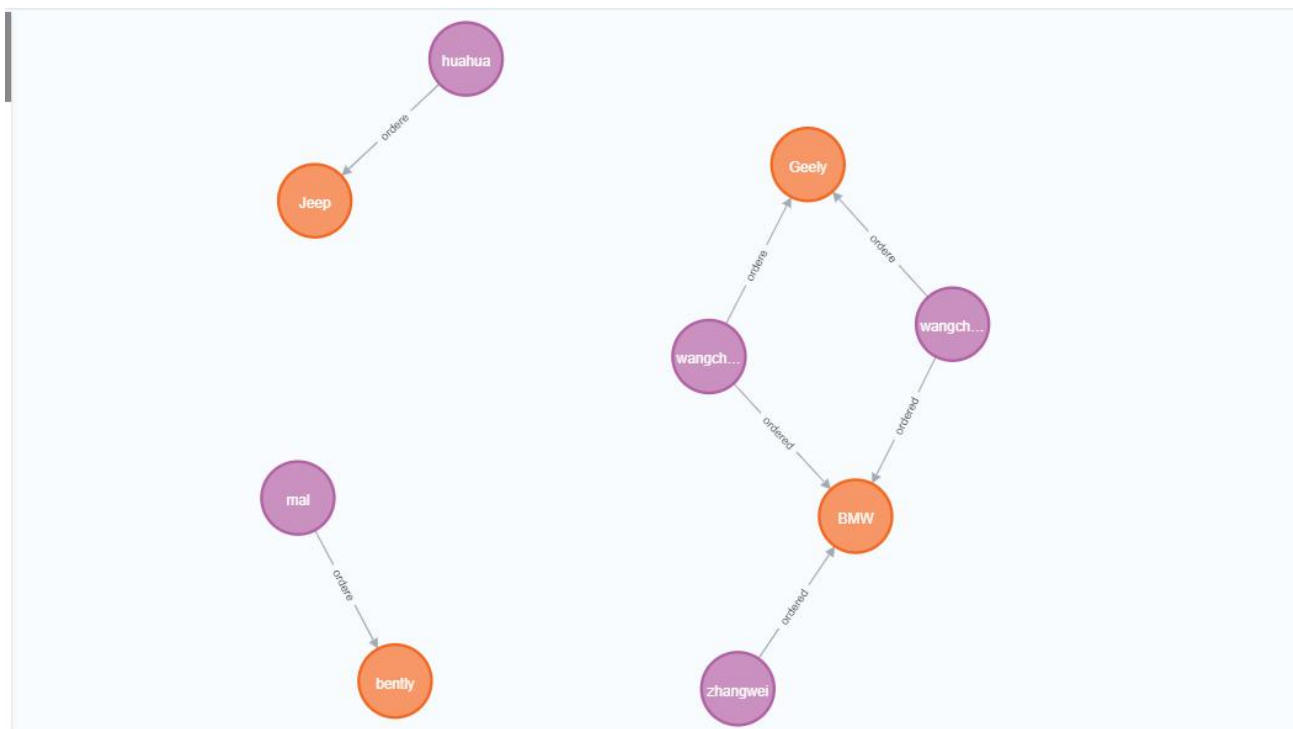
Задание 2. Отношения между узлами (базовая часть)关于节点之间的关系 (基础部分)

2.1 Создать отношения между несколькими узлами (с параметрами).

```
match (u:users{realname:"zhangwei"}), (c:car{carname:"BMW"})
merge(u)-[buy:ordered{pay:'cash'}]->(c) return u, c
```



Все relationship



2.2 Продемонстрировать содержимое БД (фильтрация по узлам, отношениям, меткам и связям).

Фильтрация по узлам:

```
match (u)-->(c) where u.realname='zhangwei' return u.address, u.phone, c.price
```



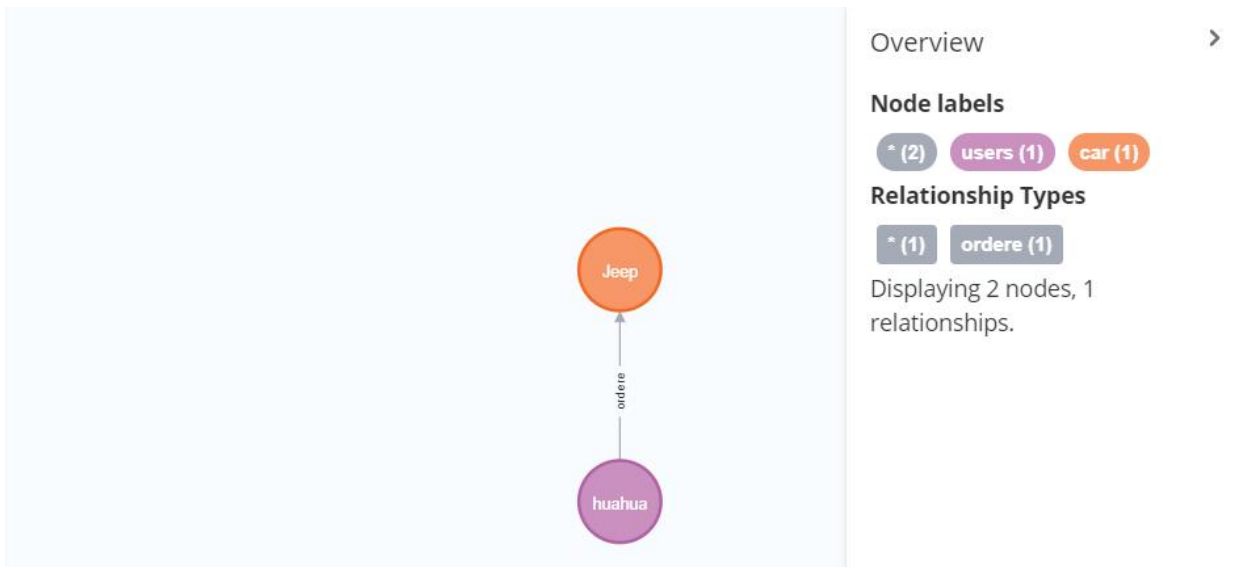
The image shows a Neo4j Cypher query interface. The query is: `match (u)-->(c) where u.realname='zhangwei' return u.address, u.phone, c.price`. The results are displayed in a table view with the following data:

	u.address	u.phone	c.price
1	"bmstu"	"79269525693"	2000000

Фильтрация по отношениям:

Клиенты просят купить JEEP

```
match(u:users)-[buy:ordere]->(c:car) where c.carname="Jeep" return u, c, buy
```



Фильтрация по метками:

```
match (u:users)-[buy]->(c) where c.carname='Jeep' return u, c
```

neo4j\$ `match (u:users)-[buy]->(c) where c.carname='Jeep'`
`return u, c`

Overview

Node labels

- * (2) users (1) car (1)

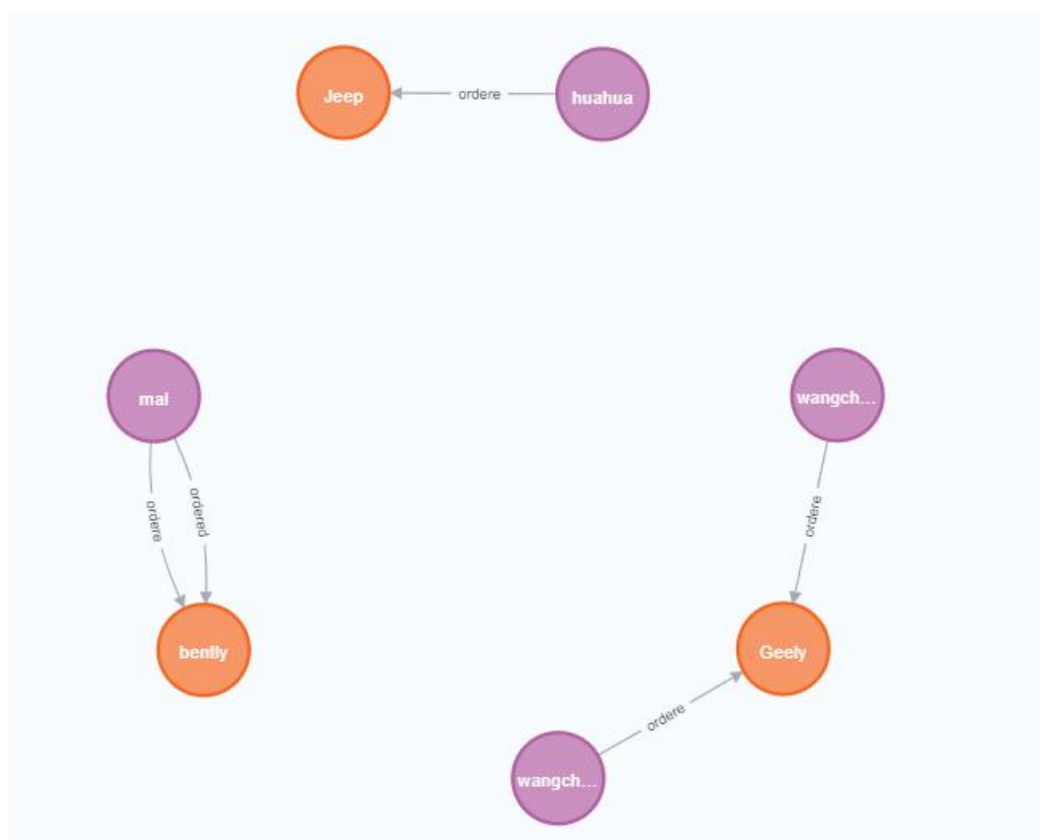
Relationship Types

- * (1) ordere (1)

Displaying 2 nodes, 0 relationships.

Фильтрация по связями:

```
match (u)-[buy:ordere]->(c) return u, c
```



Задание 3. Запросы к БД на языке Cypher (базовая часть) Cypher语言查询数据库（基础部分）

Выполнить запросы к базе данных на языке Cypher:

3.1 с условием NOT NULL

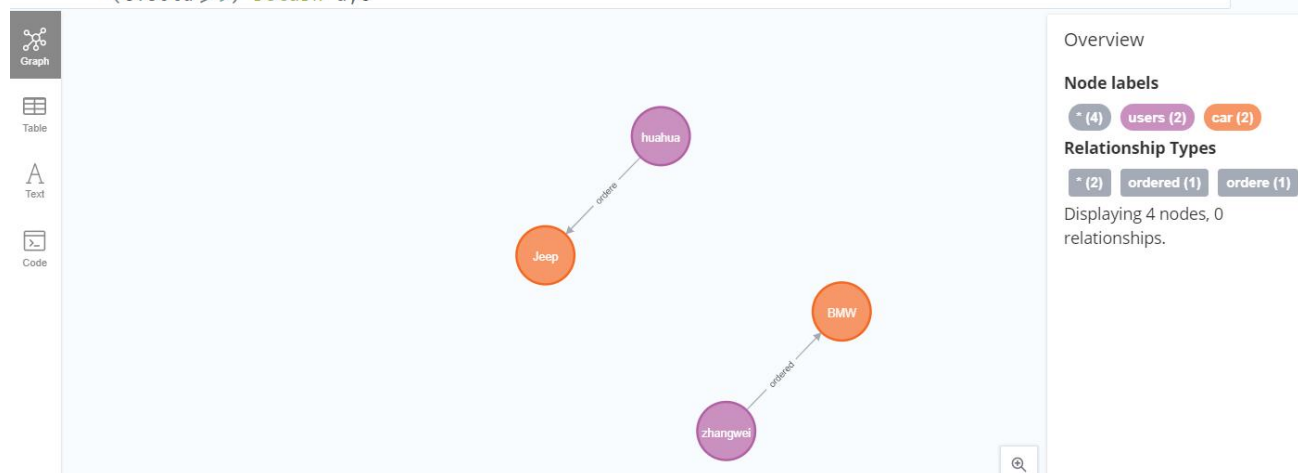
```
match (c) where c.stock is not null return c
```



3.2 операторами AND, OR

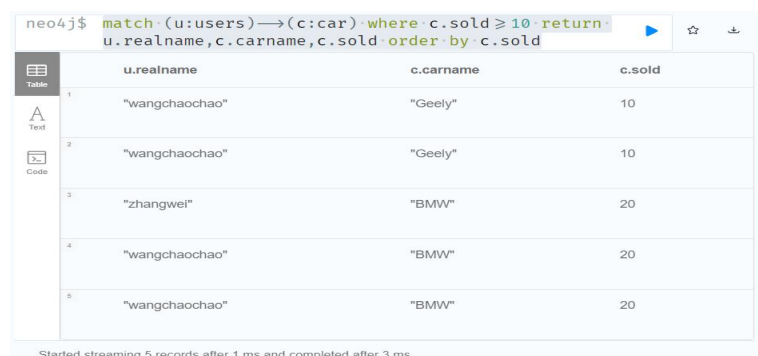
```
match (u:users)-->(c:car) where (u.realname='zhangwei' or u.realname='huahua')
and (c.sold>=9) return u, c
```

```
neo4j$ match (u:users)-->(c:car) where (u.realname='zhangwei' or u.realname='huahua') and
(c.sold >= 9) return u, c
```



3.3 с сортировкой

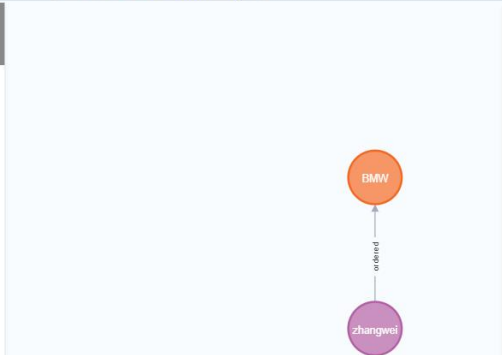
```
match (u:users)-->(c:car) where c.sold>=10 return u.realname, c.carname, c.sold
order by c.sold
```



3.4 с условием на направление отношения

```
match (u)-->(c) where u.realname='zhangwei' and c.sold>1 return u,c
```

neo4j\$ `match (u)-->(c) where u.realname='zhangwei' and c.sold>1 return u,c`



Overview

Node labels

- * (2) users (1) car (1)

Relationship Types

- * (1) ordered (1)

Displaying 2 nodes, 0 relationships.

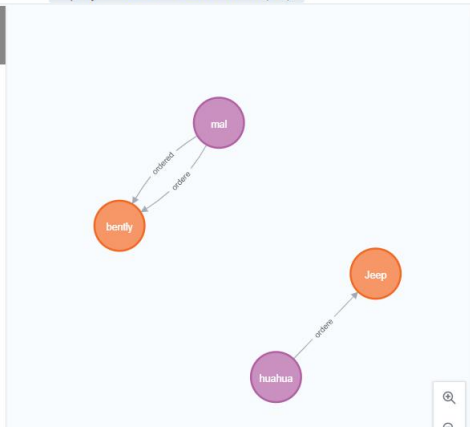
```
neo4j$ match (c)-->(u) where u.realname='zhangwei' and c.sold>1 return u,c
```

(no changes, no records)

3.5 с параметрами отношения

```
match (u:users)-[r:ordere]->(c:car) where r.pay='card' return u,c,r
```

neo4j\$ `match (u:users)-[r:ordere]->(c:car) where r.pay='card' return u,c,r`



Overview

Node labels

- * (4) users (2) car (2)

Relationship Types

- * (3) ordere (2)
- ordered (1)

Displaying 4 nodes, 2 relationships.

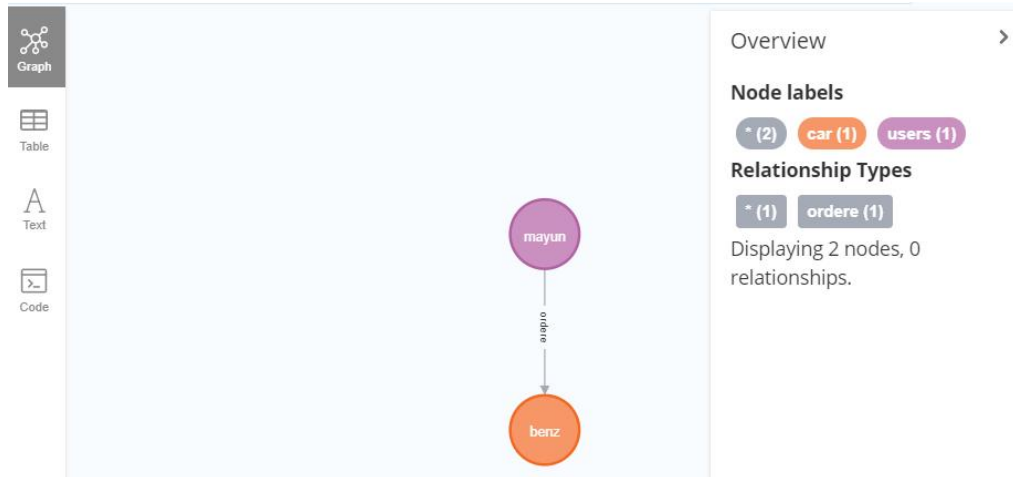
neo4j\$ `match (u:users)-[r:ordere]->(c:car) where r.pay='card' return u,c,r`

u	c	r
<pre>{ "identity": 4, "labels": ["users"], "properties": { "address": "bmstu", "gender": 0, "phone": "321654987", "userId": 3, "realname": "huahua", "username": "hh" } }</pre>	<pre>{ "identity": 8, "labels": ["car"], "properties": { "cardesc": "wrangler", "sold": 9, "stock": 600, "price": 429000000, "carsid": 3, "carname": "Jeep" } }</pre>	<pre>{ "identity": 5, "start": 4, "end": 8, "type": "ordere", "properties": { "pay": "card" } }</pre>

Задание 4. CRUD (хорошо)

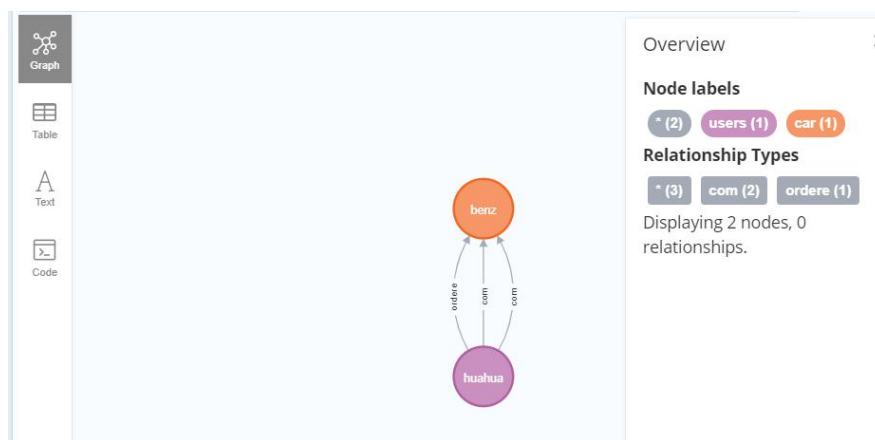
Создать отношение между новыми узлами.

```
create (u5:users{userid:5,address:'bmstu',gender:0,phone:'79269568946',realname:'mayun',username:'my'})-[buy:ordere{pay:'cash'}]->(c5:car{carsid:5,carnames:'blue',carname:'benz',price:11000000,sold:11,stock:50}) return c5,u5
```



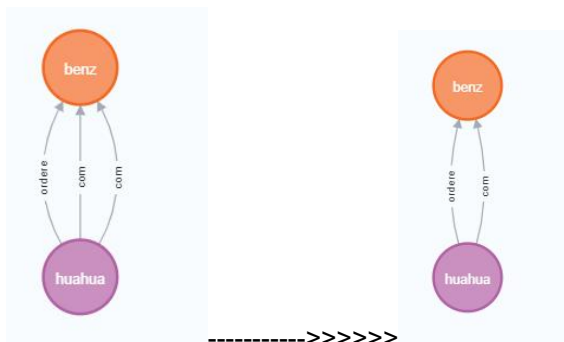
Создать отношение между существующими узлами.

```
match (u:users{realname:'huahua'}),(c:car{carname:'benz'}) merge(u)-[commend:com{likeornot:'like'}]->(c) return u,c
```



Продемонстрировать удаление узлов и связей.

```
match (u:users{realname:'huahua'})-[commend:com{likeornot:'like'}]->(c:car{carname:'benz'}) delete commend
```



Удаление одного узла и всех его связей

```
neo4j$ match(u3{realname:'huahua'}) detach delete u3
```



Table

Deleted 1 node, deleted 3 relationships, completed after 7 ms.



Продемонстрировать удаление и изменение свойств и меток.

изменение свойств .

```
neo4j$ match (c{carname:'BMW'}) return  
c.price,c.sold,c.stock
```



Table



Text

	c.price	c.sold	c.stock
1	2000000	20	100

```
neo4j$ match (c) where c.carname='BMW' set  
c.price=3000000 return  
c.carname,c.price,c.sold,c.stock
```



Table



Text

	c.carname	c.price	c.sold	c.stock
1	"BMW"	<u>3000000</u>	20	100

меток

```
4j$ match (c) where c.carname='BMW' set c:money return
```

c

```
{  
  "identity": 6,  
  "labels": [  
    "car",  
    "money"  
  ],  
}
```

удаление свойств и меток.

Свойств

```
match (c) where c.carname='BMW' return c
```

```
{
  "identity": 6,
  "labels": [
    "car",
    "money"
  ],
  "properties": {
    "sold": 20,
    "cardesc": "red",
    "stock": 100,
    "carsid": 1,
    "price": 3000000,
    "carname": "BMW"
  }
}
```

```
match (c) where c.carname='BMW' remove c.cardesc
return c
```

```
{
  "identity": 6,
  "labels": [
    "car",
    "money"
  ],
  "properties": {
    "sold": 20,
    "stock": 100,
    "carsid": 1,
    "price": 3000000,
    "carname": "BMW"
  }
}
```

меток.

```
neo4j$ match (c) where c.carname='BMW' remove c:money
return c
```

Graph	Table	Text	Code
	1		
			{ "identity": 6, "labels": ["car"], "properties": { "sold": 20, "stock": 100, "carsid": 1, "price": 3000000, "carname": "BMW" } }

Продемонстрировать работу команд UNION, MERGE.

```
UNION:match (c:car) return c union match (c:expensive) return c
```



Overview

Node labels

^ (5) car (4) expensive (1)

Displaying 5 nodes, 0 relationships.

MERGE.:

```
merge (c:car{carsid:6, cardesc:'green', carname:'lada', price:60000, sold:1, stock:1})
```

```
neo4j$ merge (c:car{carsid:6, cardesc:'green', carname:'lada', price:60000, sold:1, stock:1})
```

Added 1 label, created 1 node, set 6 properties, completed after 10 ms.

узел уже существует, то новый не создался.

(no changes, no records)

Задание 5. Расширенные запросы к БД (хорошо) 高级数据库查询 (好)

Выполнить запросы к базе данных на языке Cypher:

с агрегированием, 聚合 ,

Count

```
o4j$ match (c:car) return count(c) as car_count
```

	car_count
1	5

MAX и MIN:

```
j$ match (c:car) return max(c.price), min(c.price)
```

max(c.price)	min(c.price)
429000000	60000

SUM:

```
j$ match (c:car) return sum(c.sold)
```

sum(c.sold)
39

Avg:

```
o4j$ match (c:car) return avg(c.price)
```

avg(c.price)
108372000.0

с встроенными функциями (строковые или иные),带有内置函数 (字符串或其他) ,

ToUPPER():

```
cypher> match (c:car) return toUpper(c.carname)
```

	toUpper(c.carname)
1	"LADA"
2	"GEELY"
3	"JEEP"
4	"BENTLY"
5	"BENZ"

Started streaming 5 records after 1 ms and completed after 2 ms.

Tolower()

```
cypher> match (c:car) return toLower(c.carname)
```

	toLowerCase(c.carname)
1	"lada"
2	"geely"
3	"jeep"
4	"bently"
5	"benz"

Started streaming 5 records after 1 ms and completed after 3 ms.

Replace()

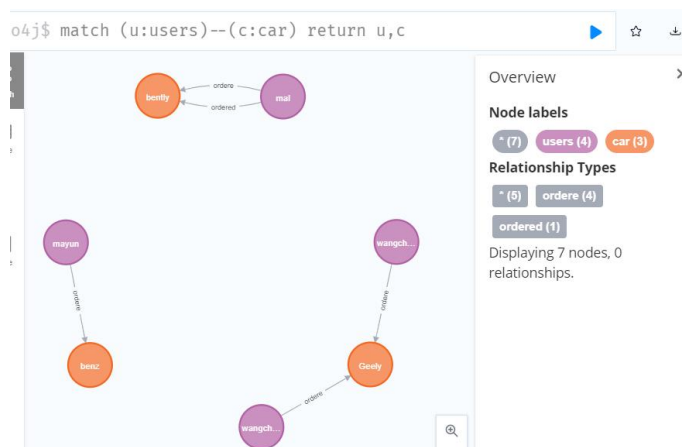
```
cypher> match (c:car) return replace(c.carname,'lada','LADA')
```

	replace(c.carname,'lada','LADA')
1	"LADA"
2	"Geely"
3	"Jeep"
4	"bently"
5	"benz"

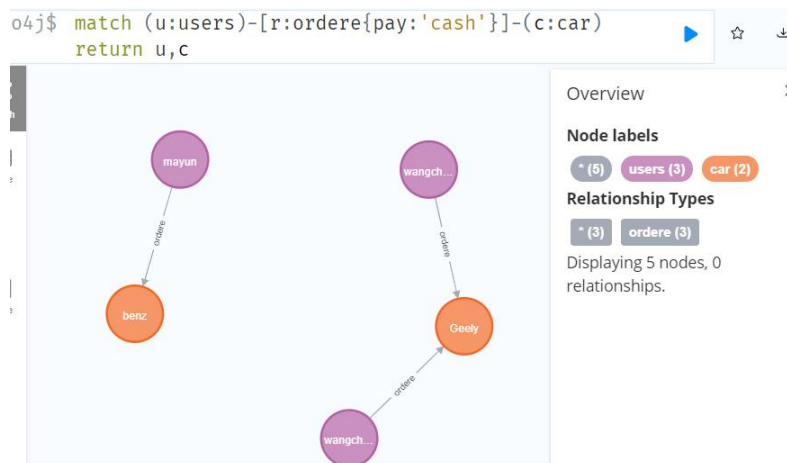
Started streaming 5 records after 1 ms and completed after 2 ms.

с шаблонами отношений,与关系模式 ,

(a) --(b)



(a)-[r:REL_TYPE]->(b)



с удалением дубликатов.删除重复项

neo4j\$ match (u:users) return u.realname

	u.realname
1	"wangchaochao"
2	"wangchaochao"
3	"zhangwei"
4	"mal"
5	"mayun"

```
o4j$ match (u:users) return distinct u.realname
```

	u.realname
1	"wangchaochao"
2	"zhangwei"
3	"mai"
4	"mayun"

Продемонстрировать работу команд LIMIT, SKIP.

LIMIT

```
neo4j$ match (c:car) return c limit 3
```

The screenshot shows the Neo4j desktop application. The main window displays a graph with three orange circular nodes labeled 'Jeep', 'Geely', and 'lada'. On the left, there is a sidebar with icons for 'Graph', 'Table', 'Text', and 'Code'. On the right, an 'Overview' sidebar shows 'Node labels' with two entries: '*' (3) and 'car' (3). Below this, it states 'Displaying 3 nodes, 0 relationships.' The command 'neo4j\$ match (c:car) return c limit 3' is entered in the top bar, with 'limit 3' underlined in red.

SKIP

```
4j$ match (c:car) return c skip 1
```

The screenshot shows the Neo4j desktop application. The main window displays a graph with four orange circular nodes labeled 'benz', 'benby', 'Jeep', and 'Geely'. On the left, there is a sidebar with icons for 'Graph', 'Table', 'Text', and 'Code'. On the right, an 'Overview' sidebar shows 'Node labels' with two entries: '*' (4) and 'car' (4). Below this, it states 'Displaying 4 nodes, 0 relationships.' The command '4j\$ match (c:car) return c skip 1' is entered in the top bar, with 'skip 1' underlined in red.

Задание 6. Индексы и ограничения (отлично) 索引和限制 (优秀)

Создать индекс. Продемонстрировать его использование.

```
neo4j$ create index russia for (c1:car) on (c1.carname)
```



Added 1 index, completed after 51 ms.

```
4j$ call db.indexes
```

	id	name	state	populationPercent	uniqueness	type	entityType	labelsOrTypes	properties
1	1	"russia"	"ONLINE"	100.0	"NONUNIQUE"	"BTREE"	"NODE"	["car"]	["carname"]

Drop index

```
4j$ drop index russia
```

Removed 1 index, completed after 6 ms.

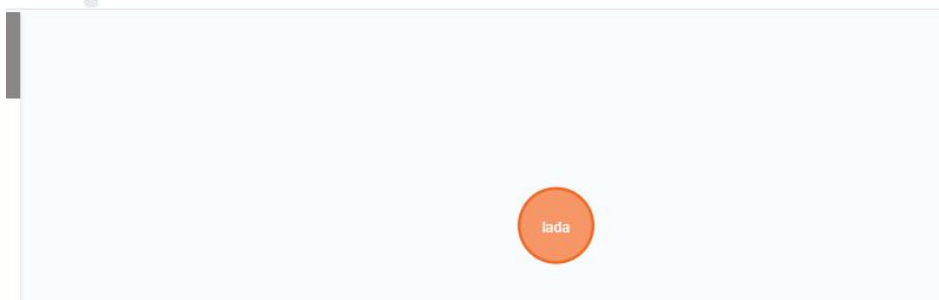
Create index

```
4j$ create index madeinrussia for(c:car) on (c.carname)
```

Added 1 index, completed after 3 ms.

Using index

```
4j$ match (c:car) using index c:car(carname) where c.carname='lada' return c
```



Overview

Node labels

^ (1) car (1)

Displaying 1 nodes, 0 relationships.

Создать ограничение. Продемонстрировать его использование.

```
o4j$ create constraint on (c:car) assert c.carsid is unique
```

Added 1 constraint, completed after 130 ms.

```
neo4j$ create (c:car{carsid:6})
```

ERROR Neo.ClientError.Schema.ConstraintValidationFailed

Node(4) already exists with label `car` and property `carsid` = 6

Не удалось создать из-за ограничений.

Создать коллекцию. Продемонстрировать запрос к ней.

```
neo4j$ match(c:car) where c.carname='lada' set c.carsid=[ 'green', '5 seats', '1987']
```

Set 1 property, completed after 15 ms.

```
neo4j$ match(c:car) where c.carname='lada' return c.carsid,c.carsdesc
```

	c.carsid	c.carsdesc
1	6	["green", "5 seats", "1987"]

```
neo4j$ match (c:car) where c.carname='lada' return c.carsid, c.carsdesc[0]
```

	c.carsid	c.carsdesc[0]
1	6	"green"

```
match (c:car) where c.carname='lada' return c.carsid, size(c.carsdesc)
```

	c.carsid	size(c.carsdesc)
1	6	3

Литература:

1. Дистрибутив - <https://neo4j.com/>.
2. Документация по языку CQL - <https://neo4j.com/docs/cypher-refcard/3.1/>.
3. Руководство по установке и началу работы - <https://ru.bmstu.wiki/Neo4j>
4. Фаулер, Мартин, Садаладж, Прамодкумар Дж. NoSQL: новая методология разработки нереляционных баз данных. : Пер. с англ. - М.: ООО "И.Д. Вильямс", 2013г.