



Міністерство освіти і науки, молоді та спорту України

Національний технічний університет України

“Київський політехнічний інститут”

Фізико-Технічний інститут

**Лабораторна робота № 4
з семестрового курсу
“Проектування високонавантажених
систем”**

Виконала:

Студентка групи ФІ-03

Швець Катерина

Київ, 2023

Task 4 - Робота з базовими функціями граф-орієнтованої БД на прикладі Neo4j

- Без установки з Neo4j можна працювати онлайн на сайті <http://neo4j.com/sandbox/> (виконувала лабораторну на сайті)

Завдання:

Змодельувати наступну предметну область:

- Є: Items, Customers, Orders
- Customer може додати Item(s) до Order (тобто купити Товар)
- У Customer може бути багато Orders
- Item може входити в багато Orders, і у Item є вартість
- Customer може переглядати (view), але при цьому не купувати Items

Структура Даних:

- Customer: Має атрибути, такі як ім'я, ідентифікатор тощо.
- Item: Має атрибути, такі як назва, вартість, ідентифікатор.
- Order: Зв'язує Customer та Item, може мати дату замовлення, ідентифікатор.

Зв'язки

- Customer -[MAKES]-> Order: Клієнт робить замовлення.
- Order -[INCLUDES]-> Item: Замовлення включає товар.
- Customer -[VIEWS]-> Item: Клієнт переглядає товар.

Далі створюю вузли та зв'язки в графі. Для цього використовую мову запитів Cypher.

The screenshot displays the Neo4j Browser interface. On the left, the 'Database Information' sidebar shows the database name 'neo4j', node labels (Customer), and relationship types. The main panel shows the Cypher query editor with the following queries and their results:

```
neo4j$ UNWIND range(2, 5) AS i CREATE (c:Customer {name: 'Customer' + toString(i), id: toString(i)})
```

Added 4 labels, created 4 nodes, set 8 properties, completed after 7 ms.

```
neo4j$ CREATE (c:Customer {name: 'Customer1', id: '1'})
```

Added 1 label, created 1 node, set 2 properties, completed after 18 ms.

Database Information

Use database

neo4j

Node labels

*() Customer Item Order

Relationship types

No relationships in database

Property keys

date id name price

Connected as

Username: google-oauth2|103315954925670918182

Roles: admin, PUBLIC

Admin: server user list server user add

Disconnect: server disconnect

DBMS

Version: 5.13.0

Edition: Enterprise

Name: neo4j

Databases: db

Information: info

Query List: queries

neo4j\$

To help make Neo4j Browser better we collect information on product usage. Review your settings at any time.

neo4j\$ UNWIND range(1, 5) AS i CREATE (:Order {id: toString(i), date: '2024-01-' + toString...)

Added 5 labels, created 5 nodes, set 10 properties, completed after 8 ms.

neo4j\$ UNWIND range(1, 5) AS i CREATE (:Item {name: 'item' + toString(i), id: toString(i), ...)

Added 5 labels, created 5 nodes, set 15 properties, completed after 15 ms.

Added 5 labels, created 5 nodes, set 15 properties, completed after 15 ms.

Database Information

Use database

neo4j

Node labels

*() Customer Item Order

Relationship types

*() MAKES

Property keys

date id name price

Connected as

Username: google-oauth2|103315954925670918182

Roles: admin, PUBLIC

Admin: server user list server user add

Disconnect: server disconnect

DBMS

Version: 5.13.0

Edition: Enterprise

Name: neo4j

Databases: db

Information: info

Query List: queries

neo4j\$

To help make Neo4j Browser better we collect information on product usage. Review your settings at any time.

neo4j\$ UNWIND range(1, 5) AS i MATCH (c:Customer {id: '1'}), (o:Order {id: toString(i)}) CR...

Created 5 relationships, completed after 11 ms.

neo4j\$ UNWIND range(1, 5) AS i MATCH (c:Customer {id: toString(i)}), (o:Order {id: '1'}) CR...

Created 5 relationships, completed after 101 ms.

Created 5 relationships, completed after 101 ms.

The image displays two screenshots of the Neo4j Browser interface, showing the database information and graph visualizations.

Top Screenshot:

- Database Information:**
 - Use database:** neo4j
 - Node labels:** * (15), Customer (5), Item (5), Order (5)
 - Relationship types:** * (20), INCLUDES (10), MAKES (10)
 - Property keys:** date, id, name, price
 - Connected as:** Username: google-oauth2|103315954925670918182, Roles: admin, PUBLIC, Admin: :server user list, :server user add, Disconnect: :server disconnect
 - DBMS:** Version: 5.13.0, Edition: Enterprise, Name: neo4j, Databases: :dbms, Information: :sysinfo, Query List: :queries
- Graph View:** neo4j\$ MATCH (n) RETURN n. The graph shows 15 nodes and 0 relationships. The nodes are labeled with their IDs and names: 2024-01-1, 2024-01-2, 2024-01-3, 2024-01-4, 2024-01-5, 2024-01-6, 2024-01-7, 2024-01-8, 2024-01-9, 2024-01-10, 2024-01-11, 2024-01-12, 2024-01-13, 2024-01-14, 2024-01-15. The nodes are connected by relationships labeled with their types: INCLUDES, MAKES, and ORDER.
- Overview:** Node labels: * (15), Customer (5), Item (5), Order (5). Relationship types: * (20), MAKES (10), INCLUDES (10). Displaying 15 nodes, 0 relationships.

Bottom Screenshot:

- Database Information:** (Same as top screenshot)
- Graph View:** neo4j\$ MATCH (n) RETURN n. The graph shows 15 nodes and 0 relationships. The nodes are labeled with their IDs and names: 2024-01-1, 2024-01-2, 2024-01-3, 2024-01-4, 2024-01-5, 2024-01-6, 2024-01-7, 2024-01-8, 2024-01-9, 2024-01-10, 2024-01-11, 2024-01-12, 2024-01-13, 2024-01-14, 2024-01-15. The nodes are connected by relationships labeled with their types: INCLUDES, MAKES, and ORDER. A tooltip message says: "Use Cmd + scroll to zoom. Don't show again."
- Overview:** Node labels: * (15), Customer (5), Item (5), Order (5). Relationship types: * (45), VIEWS (25), MAKES (10), INCLUDES (10). Displaying 15 nodes, 0 relationships.

Написати наступні види запитів:

- Знайти Items які входять в конкретний Order

Database Information

Use database

neo4j

Node labels

*(15)

Customer

Item

Order

Relationship types

*(45)

INCLUDES

MAKES

Property keys

date

id

name

price

Connected as

Username: google-oauth2|103315954925670918182

Roles: admin, PUBLIC

Admin: server user list

server user add

server disconnect

DBMS

Version: 5.13.0

Edition: Enterprise

Name: neo4j

Databases: xdb

Information: sysinfo

Query List: queries

neo4j\$

To help make Neo4j Browser better we collect information on product usage. Review your [settings](#) at any time.

neo4j\$ MATCH (o:Order {id: '4'})-[:INCLUDES]->(i:Item) RETURN o, i

Graph

Table

Text

Code

Overview

Node labels

*(6)

Order (1)

Item (5)

Relationship types

*(6)

INCLUDES (6)

Displaying 6 nodes, 0 relationships.

neo4j\$ MATCH (n) RETURN n

Graph

Table

Text

Code

Overview

Node labels

*(15)

Customer (5)

Item (5)

Order (5)

- Підрахувати вартість конкретного Order

neo4j\$ MATCH (o:Order {id: '4'})-[:INCLUDES]->(i:Item) RETURN o, sum(i.price) AS TotalP...

Graph

Table

Text

Code

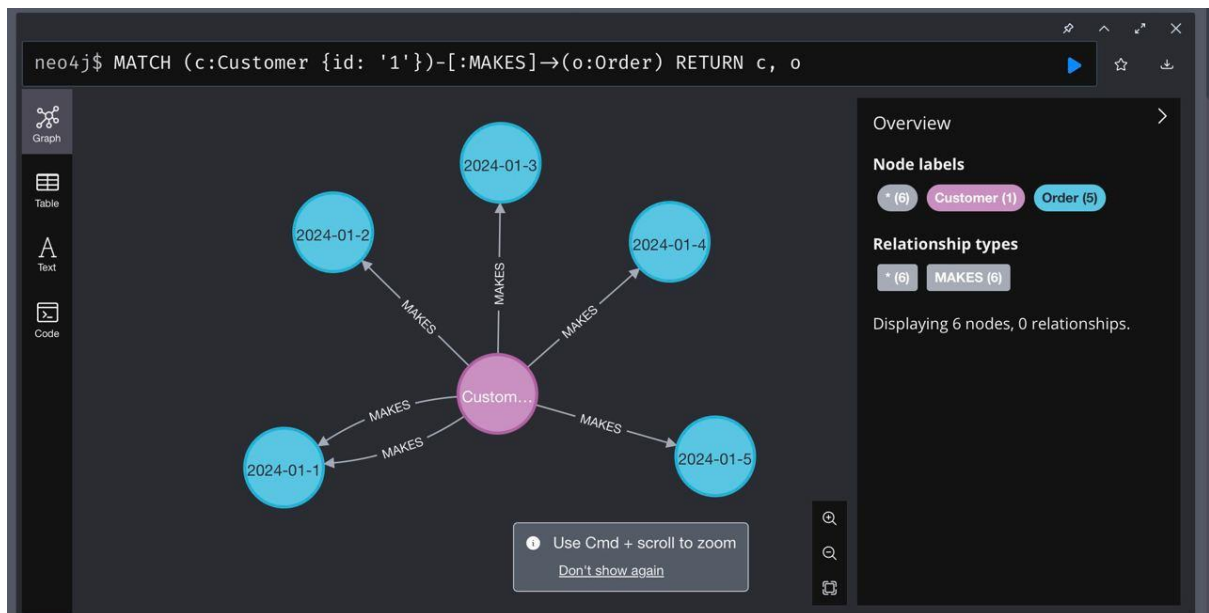
	o	TotalPrice
1	<pre>{ "identity": 13, "labels": ["Order"], "properties": { "date": "2024-01-4", "id": "4" }, "elementId": "13" }</pre>	1900

Started streaming 1 records after 2 ms and completed after 16 ms.

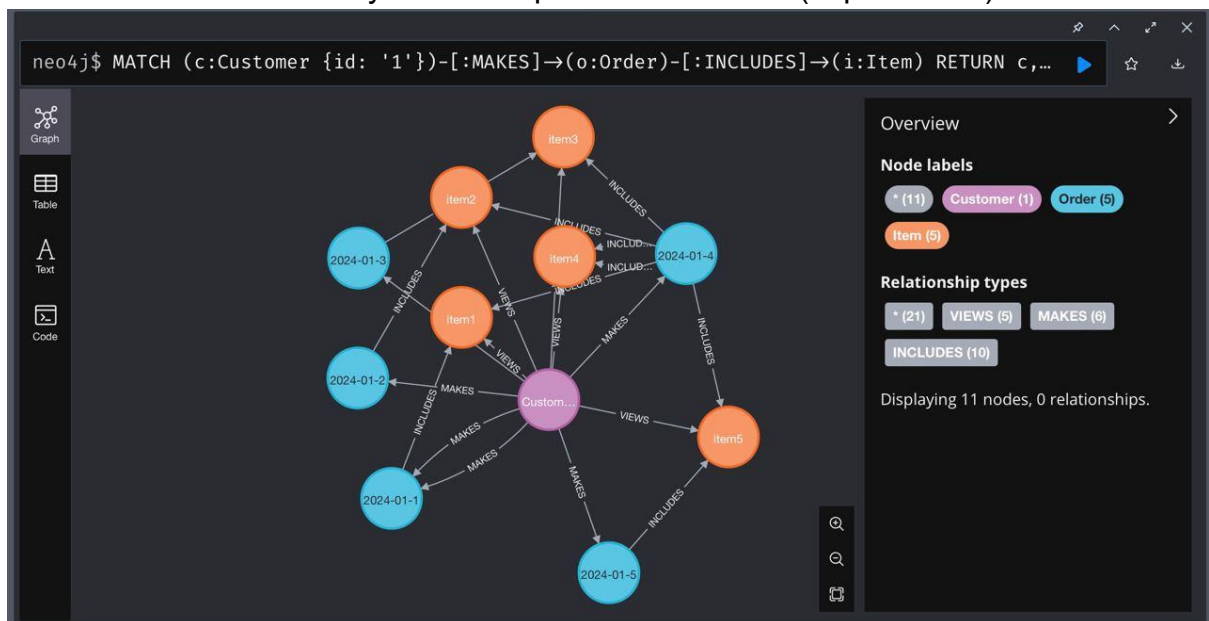
```
MATCH (o:Order {id: '4'})-[:INCLUDES]->(i:Item)
RETURN o, sum(i.price) AS TotalPrice
```

- Знайти всі Orders конкретного Customer

```
MATCH (c:Customer {id: '1'})-[:MAKES]->(o:Order)
RETURN c, o
```



- Знайти всі Items куплені конкретним Customer (через Order)



MATCH (c:Customer {id: '1'})-[:MAKES]->(o:Order)
RETURN c, o, i

- Знайти кількість Items куплені конкретним Customer (через Order)

MATCH (c:Customer {id: '1'})-[:MAKES]->(o:Order)-[:INCLUDES]->(i:Item)
RETURN c, count(i) AS NumberOfItems

```
neo4j$ MATCH (c:Customer {id: '1'})-[:MAKES]->(o:Order)-[:INCLUDES]->(i:Item) RETURN c,...
```

	c	NumberOfItems
1	<pre>{ "identity": 0, "labels": ["Customer"], "properties": { "name": "Customer1", "id": "1" }, "elementId": "0" }</pre>	11

Started streaming 1 records after 2 ms and completed after 4 ms.

- Знайти для Customer на яку суму він придбав товарів (через Order)

```
neo4j$ MATCH (c:Customer {id: '1'})-[:MAKES]->(o:Order)-[:INCLUDES]->(i:Item) RETURN c,...
```

	c	TotalSpent
1	<pre>{ "identity": 0, "labels": ["Customer"], "properties": { "name": "Customer1", "id": "1" }, "elementId": "0" }</pre>	3100

Started streaming 1 records after 2 ms and completed after 4 ms.

```
MATCH (c:Customer {id: '1'})-[:MAKES]->(o:Order)-[:INCLUDES]->(i:Item)
RETURN c, sum(i.price) AS TotalSpent
```

- Знайти скільки разів кожен товар був придбаний, відсортувати за цим значенням

```
MATCH (i:Item)<-[:INCLUDES]-(o:Order)
WITH i, count(o) AS OrdersCount
RETURN i.name, OrdersCount
ORDER BY OrdersCount DESC
```

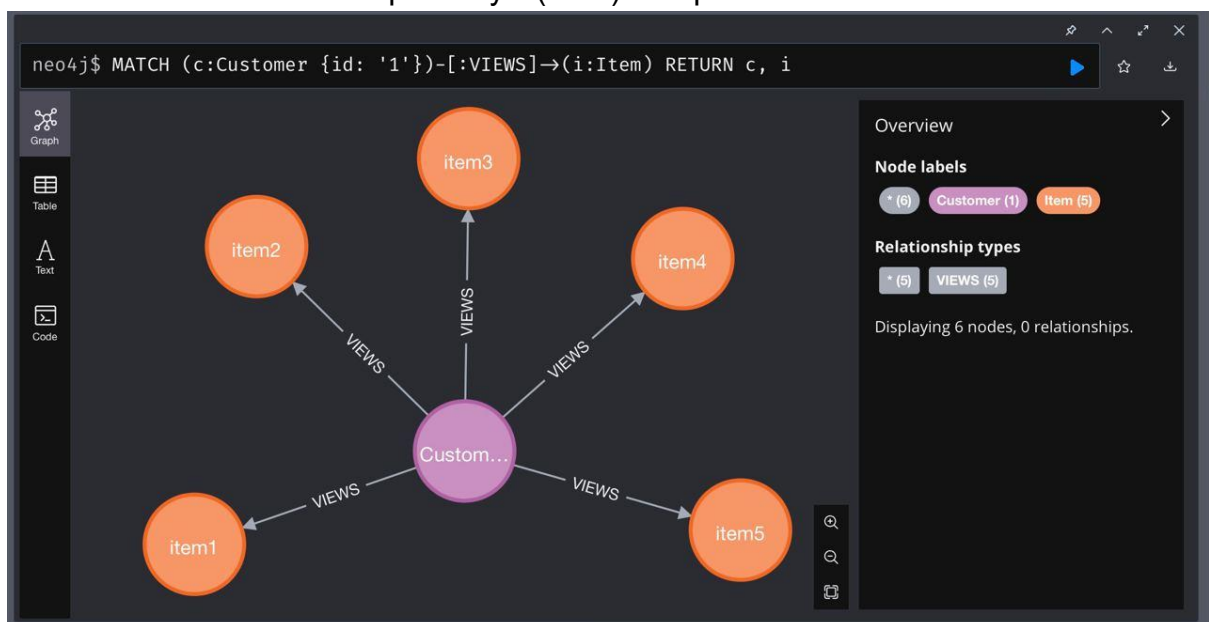


```
neo4j$ MATCH (i:Item)←[:INCLUDES]-(o:Order) WITH i, count(o) AS OrdersCount RETURN i.n...
```

	i.name	OrdersCount
1	"item1"	2
2	"item2"	2
3	"item3"	2
4	"item4"	2
5	"item5"	2

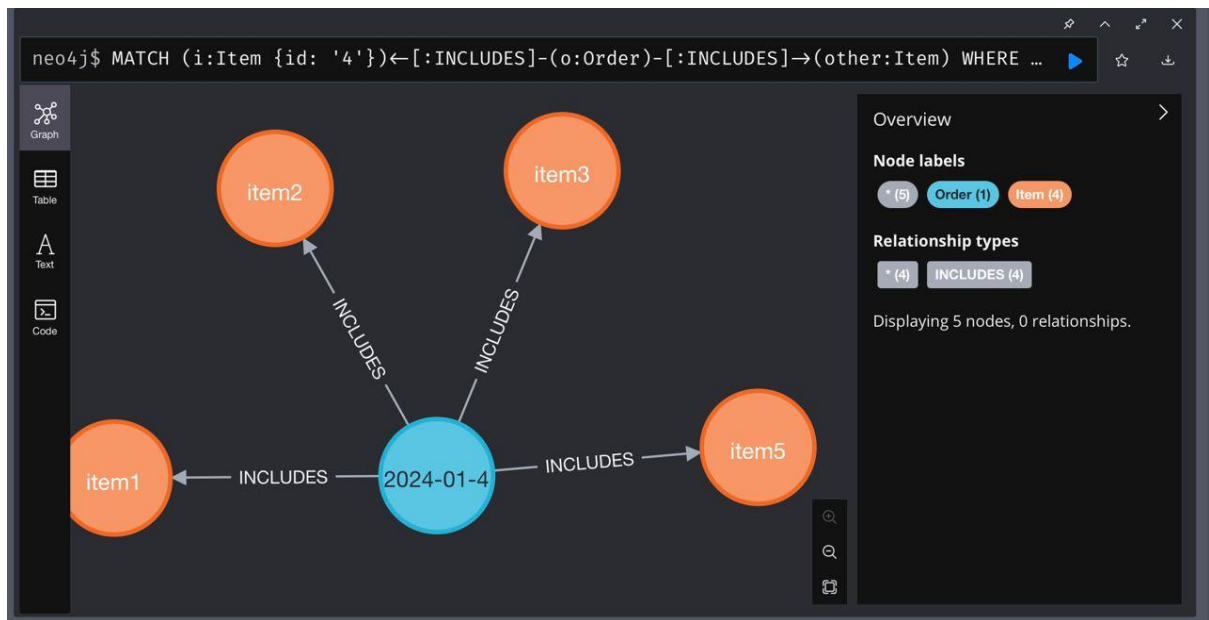
Started streaming 5 records after 17 ms and completed after 18 ms.

- Знайти всі Items переглянуті (view) конкретним Customer



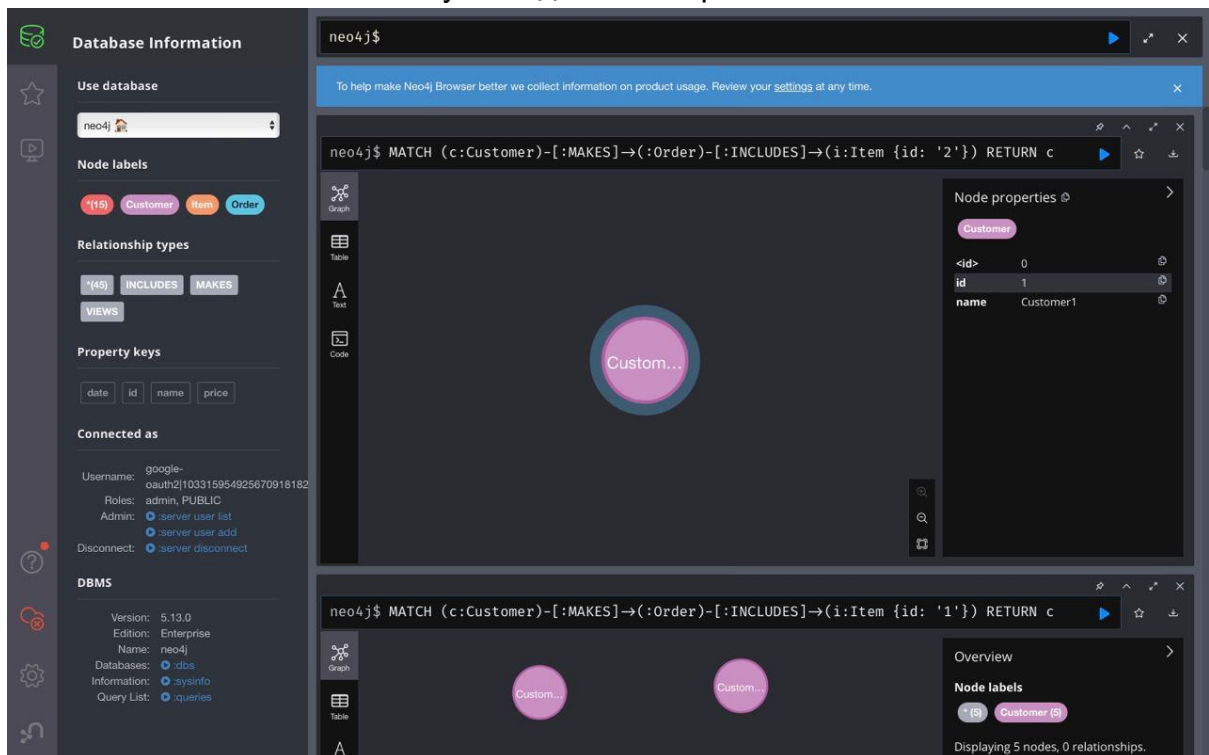
```
MATCH (c:Customer {id: '1'})-[:VIEWS]->(i:Item)
RETURN c, i
```

- Знайти інші Items що купувались разом з конкретним Item (тобто всі Items що входять до Order-s разом з даними Item)



MATCH (i:Item {id: '4'})<-[:INCLUDES]-(o:Order)-[:INCLUDES]->(other:Item)
 WHERE i <> other
 RETURN o, other

- Знайти Customers які купили даний конкретний Item



MATCH (c:Customer)-[:MAKES]->(o:Order)-[:INCLUDES]->(i:Item {id: '1'})
 RETURN c

- Знайти для певного Customer(а) товари, які він переглядав, але не купив

The screenshot displays the Neo4j Browser interface. On the left, the 'Database Information' sidebar shows the database name 'neo4j', node labels ('Customer', 'Item', 'Order'), relationship types ('INCLUDES', 'MAKES', 'VIEWS'), and property keys ('date', 'id', 'name', 'price'). The main area shows a graph with two nodes: 'item6' (orange) and 'Custom...' (purple), connected by a 'VIEWS' relationship. The right sidebar provides an overview of the graph, showing 2 nodes and 0 relationships. The command history at the bottom shows the following queries:

```
neo4j$ MATCH (c:Customer {id: '1'})-[:VIEWS]->(i:Item) WHERE NOT (c)-[:MAKES]->(:Order)...  
neo4j$ MATCH (c:Customer {id: '1'}), (i:Item {id: '6'}) CREATE (c)-[:VIEWS]->(i)  
neo4j$ CREATE (i:Item {name: 'item6', price: 999, id: '6'})
```

```
MATCH (c:Customer {id: '1'})-[:VIEWS]->(i:Item)  
WHERE NOT (c)-[:MAKES]->(:Order)-[:INCLUDES]->(i)  
RETURN c, i
```