

Міністерство освіти і науки України  
Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Фізико-технічний інститут

Проектування високонавантажених систем

Лабораторна робота 3  
«Робота з базовими функціями граф-орієнтованої БД на  
прикладі Neo4j»

Виконав:  
Студент групи ФБ-42мп  
Осіпчук Антон

Київ – 2024

**Змодельювати 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**

CREATE (:Customer {id: 1, name: "Alice"});

CREATE (:Customer {id: 2, name: "Bob"});

CREATE (:Item {id: 1, name: "Laptop", price: 1000});

CREATE (:Item {id: 2, name: "Phone", price: 500});

CREATE (:Item {id: 3, name: "Headphones", price: 200});

CREATE (:Order {id: 1});

CREATE (:Order {id: 2});

```
neo4j$ CREATE (:Customer {id: 1, name: "Alice"}); CREATE (:Customer {id: 2, name: "Bob"})
```

✓ CREATE (:Customer {id: 1, name: "Alice"});

✓ CREATE (:Customer {id: 2, name: "Bob"});

✓ CREATE (:Item {id: 1, name: "Laptop", price: 1000});

✓ CREATE (:Item {id: 2, name: "Phone", price: 500});

✓ CREATE (:Item {id: 3, name: "Headphones", price: 200});

✓ CREATE (:Order {id: 1});

✓ CREATE (:Order {id: 2});

```
MATCH (c:Customer {id: 1}), (o:Order {id: 1})
CREATE (c)-[:CREATED]->(o);
MATCH (o:Order {id: 1}), (i:Item {id: 1})
CREATE (o)-[:CONTAINS]->(i);
MATCH (o:Order {id: 1}), (i:Item {id: 2})
CREATE (o)-[:CONTAINS]->(i);
MATCH (c:Customer {id: 1}), (i:Item {id: 3})
CREATE (c)-[:VIEWED]->(i);
MATCH (c:Customer {id: 2}), (o:Order {id: 2})
CREATE (c)-[:CREATED]->(o);
MATCH (o:Order {id: 2}), (i:Item {id: 3})
CREATE (o)-[:CONTAINS]->(i);
```

```
neo4j$ MATCH (c:Customer {id: 1}), (o:Order {id: 1}) CREATE (c)-[:CREATED]->(o); MATC
```

✓ MATCH (c:Customer {id: 1}), (o:Order {id: 1})

✓ MATCH (o:Order {id: 1}), (i:Item {id: 1})

✓ MATCH (o:Order {id: 1}), (i:Item {id: 2})

✓ MATCH (c:Customer {id: 1}), (i:Item {id: 3})

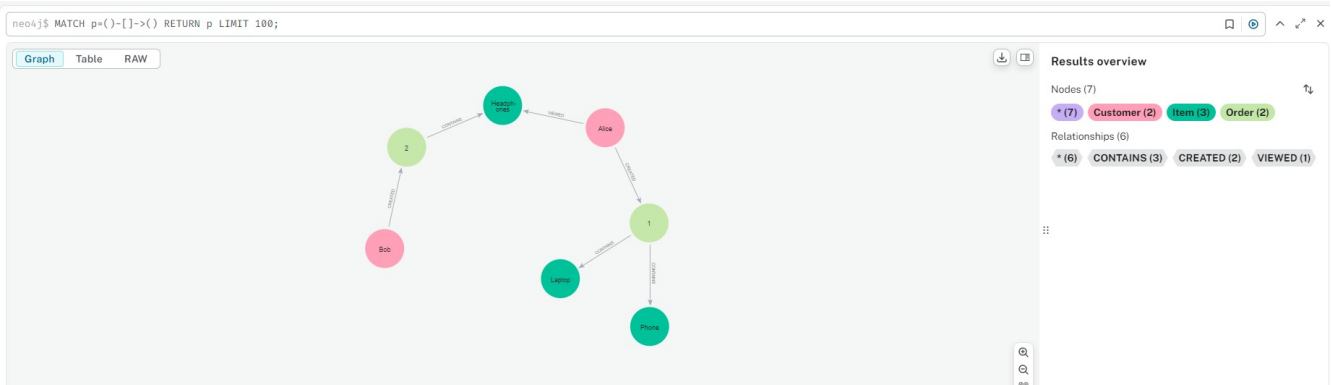
✓ MATCH (c:Customer {id: 2}), (o:Order {id: 2})

✓ MATCH (o:Order {id: 2}), (i:Item {id: 3})

**Написати наступні види запитів:**

**Знайти Items які входять в конкретний Order**

**MATCH (o:Order {id: 1})-[:CONTAINS]->(i:Item)  
RETURN i;**



**Підрахувати вартість конкретного Order**

**MATCH (o:Order {id: 1})-[:CONTAINS]->(i:Item)  
RETURN o.id AS OrderId, SUM(i.price) AS TotalPrice;**

```
neo4j$ MATCH (o:Order {id: 1})-[:CONTAINS]->(i:Item) RETURN i;
```

Graph Table RAW

i

1 (:Item {price: 1000, name: "Laptop", id: 1})

2 (:Item {price: 500, name: "Phone", id: 2})

**Знайти всі Orders конкретного Customer**

**MATCH (c:Customer {id: 1})-[:CREATED]->(o:Order)  
RETURN o;**

```
neo4j$ MATCH (c:Customer {id: 1})-[:CREATED]->(o:Order) RETURN o;
```

Graph Table RAW

o

1 (:Order {id: 1})

## Знайти всі Items куплені конкретним Customer (через Order)

MATCH (c:Customer {id: 1})-[:CREATED]->(:Order)-  
[:CONTAINS]->(i:Item)  
RETURN i;

The screenshot shows the Neo4j query results interface. At the top, the query is entered in a text box: `neo4j$ MATCH (c:Customer {id: 1})-[:CREATED]->(:Order)-[:CONTAINS]->(i:Item) RETURN i;`. Below the query box are three tabs: "Graph", "Table", and "RAW". The "Table" tab is selected. The results are displayed in a table with one column labeled "i". There are two rows of results: the first row contains `(:Item {price: 1000, name: "Laptop", id: 1})` and the second row contains `(:Item {price: 500, name: "Phone", id: 2})`. A scrollbar is visible at the bottom of the table.

i
<code>(:Item {price: 1000, name: "Laptop", id: 1})</code>
<code>(:Item {price: 500, name: "Phone", id: 2})</code>

## Знайти кількість Items куплені конкретним Customer (через Order)

MATCH (c:Customer {id: 1})-[:CREATED]->(:Order)-  
[:CONTAINS]->(i:Item)  
RETURN COUNT(i) AS TotalItems;

The screenshot shows the Neo4j query results interface. At the top, the query is entered in a text box: `neo4j$ MATCH (c:Customer {id: 1})-[:CREATED]->(:Order)-[:CONTAINS]->(i:Item) RETURN COUNT(i) AS TotalItems;`. Below the query box are two tabs: "Table" and "RAW". The "Table" tab is selected. The results are displayed in a table with one column labeled "TotalItems". There is one row of results containing the value `2`. A scrollbar is visible at the bottom of the table.

TotalItems
<code>2</code>

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

```
MATCH (c:Customer {id: 1})-[:CREATED]->(Order)-[:CONTAINS]->(i:Item)
RETURN SUM(i.price) AS TotalSpent;
```

```
neo4j$ MATCH (c:Customer {id: 1})-[:CREATED]->(Order)-[:CONTAINS]->(i:Item) RETURN SUM(i.price) AS TotalSpent;
```

TotalSpent
1 1500

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

```
MATCH (:Order)-[:CONTAINS]->(i:Item) RETURN i.name AS ItemName, COUNT(*) AS PurchaseCount ORDER BY PurchaseCount DESC;
```

```
neo4j$ MATCH (:Order)-[:CONTAINS]->(i:Item) RETURN i.name AS ItemName, COUNT(*) AS PurchaseCount ORDER BY PurchaseCount DESC;
```

ItemName	PurchaseCount
1 "Laptop"	1
2 "Phone"	1
3 "Headphones"	1

**Знайти всі Items переглянуті (view) конкретним Customer**

```
MATCH (c:Customer {id: 1})-[:VIEWED]->(i:Item)
RETURN i;
```

```
neo4j$ MATCH (c:Customer {id: 1})-[:VIEWED]->(i:Item) RETURN i;
```

i
1 (:Item {price: 200, name: "Headphones", id: 3})

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

```
MATCH (:Item {id: 1})<-[:CONTAINS]-(o:Order)-[:CONTAINS]->(i:Item)
WHERE i.id <> 1
RETURN i;
```

```
neo4j$ MATCH (:Item {id: 1})<-[:CONTAINS]-(o:Order)-[:CONTAINS]->(i:Item) WHERE i.id <> 1 RETURN i;
```

Graph Table RAW

i

```
{:Item {price: 500, name: "Phone", id: 2}}
```

**Знайти Customers які купили даний конкретний Item**

```
MATCH (c:Customer)-[:CREATED]->(o:Order)-[:CONTAINS]->(i:Item {id: 1})
RETURN c;
```

```
neo4j$ MATCH (c:Customer)-[:CREATED]->(o:Order)-[:CONTAINS]->(i:Item {id: 1}) RETURN c;
```

Graph Table RAW

c

```
{:Customer {name: "Alice", id: 1}}
```

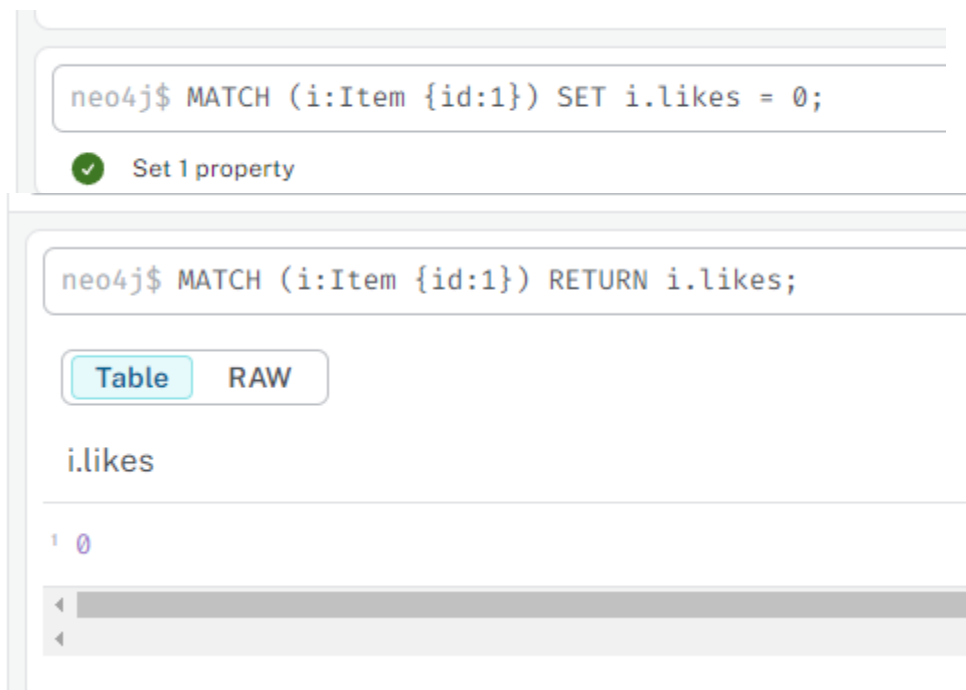
**Знайти для певного Customer(a) товари, які він переглядав, але не купив**

```
MATCH (c:Customer {id: 1})-[:VIEWED]->(i:Item)
WHERE NOT EXISTS {
  MATCH (c)-[:CREATED]->(:Order)-[:CONTAINS]->(i)
}
RETURN i;
```



**Як і в попередніх завданнях, для якогось одного обраного Item додайте поле з кількістю його лайків.**

З 10 окремих клієнтів одночасно запуснути інкрементацію каунтеру лайків по 10\_000 на кожного клієнта  
зробіть так щоб не було втрат та перевірте щоб фінальне значення було 100\_000  
заміряйте час роботи





```
NameError: name 'driver' is not defined
PS C:\Users\User\Desktop\pvs_git\lab3> py .\lab3.py
Likes: 0
Time: 1s
Likes: 100000
PS C:\Users\User\Desktop\pvs_git\lab3>
```