



Master of Science in Data Analytics

Dublin Business School

CA

Data Storage Solutions for Data Analytics

Course Code: B9DA111_2122_TMD3

Submitted By: Saima Saleem 10603707

Individual Report

Submitted To: Bernie Lydon

Table of Contents

Group Task	1
Database Description	1
Database Creation	1
Importing The Tables	1
Loading Nodes	1
Indexes	16
Relationships	17
Employee and Store	17
Store and Order	18
Order and Customer (Order_Sold_To)	19
Order and Order_Item (Order_Include_Items)	20
Order_Items and Product (OrderedItems_Includes)	21
Product and Category (PART_OF)	22
Product and Brand (PART_OF_Brand)	23
Database Schema Visualization	24
Queries: Cypher Versus SQL	25
Query1	25
Query 2	26
Query 3	27
Query 4	28
Query 6	30
Query 7	31
Conclusion	32
References	34

Group Task

My group task is to work on Neo4J and to compare graph and relational databases with the consultation with other group member who is working on SQL part. Rest was to work on presentation collectively. I enjoyed my group task.

Database Description

The database “Bike Store” has the following tables:

- brand: brand_id, brand_name. There are 9 brands.
- category: category_id, category_name. There are 7 categories.
- customer: customer_id, first_name, last_name, phone, email, street, city, state, zip_code. There are 1445 customers based in 3 states i.e. NY, TX, CA
- employee: staff_id, first_name, last_name, email, phone, active, store_id, manager_id. Total number of employees is 10.
- order: order_id, customer_id, order_status, order_date, required_date, shipped_date, store_id, staff_id. Total number of orders are 1615. Ordered years are 2016, 2017, 2018
- product:product_id, product_name, brand_id, category_id, model_year, list_price. There are 321 products.
- store: store_id, store_name, phone. Email, street, city, state, zip_code. There are 3 stores with the names Santa Cruz Bikes, Baldwin Bikes, Rowlett Bikes.
- order_items: order_id, item_id, product_id, quantity, list_price, discount

Database Creation

Local Database “Graph DBMS_BikeStore” was created.

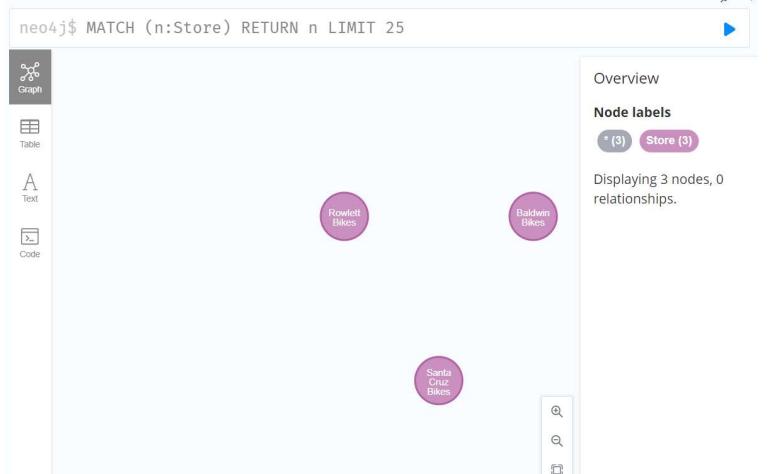
Importing The Tables

The tables csv files were placed in the import folder.

Loading Nodes

Store Node

```
neo4j$ LOAD CSV WITH HEADERS FROM 'file:///store.csv' AS row CREATE  
  (n:Store)SET n=row
```



```
neo4j$ MATCH (n:Store) RETURN n LIMIT 25
```

The screenshot shows the results of the query `MATCH (n:Store) RETURN n LIMIT 25`. The results are displayed in a JSON format. The first result, indexed at 1, is shown in a light gray background. It contains a single node object with the label "Store". The properties of the node are:

```

1
{
  "identity": 0,
  "labels": [
    "Store"
  ],
  "properties": {
    "store_id": "1",
    "phone": "(831) 476-4321",
    "city": "Santa Cruz",
    "street": "3700 Portola Drive",
    "store_name": "Santa Cruz Bikes",
    "state": "CA",
    "email": "santacruz@bikes.shop",
    "zip_code": "95060"
  }
}

```

Product Node

```
neo4j$ LOAD CSV WITH HEADERS FROM 'file:///product.csv' AS row CREATE  
(n:Product)SET n=row
```

The screenshot shows the Neo4j browser interface. At the top, a command line window displays the import command: `neo4j$ LOAD CSV WITH HEADERS FROM 'file:///product.csv' AS row CREATE (n:Product)SET n=row`. Below it, a message window shows the result: "Added 321 labels, created 321 nodes, set 1926 properties, completed after 66 ms." On the left, a sidebar menu includes "Table", "Code", and "Graph". The main area is a graph visualization showing 25 nodes as orange circles, each labeled with a product name like "Pure Cycles", "Electra Townie Org", "Electra 1...", "Surly Straggler", etc.

The screenshot shows the Neo4j browser interface. At the top, a command line window displays the query: `neo4j$ MATCH (n:Product) RETURN n LIMIT 25`. Below it, a message window shows the result: "Displaying 25 nodes, 0 relationships." On the left, a sidebar menu includes "Table", "Text", and "Code". The main area is a graph visualization showing 25 nodes as orange circles, each labeled with a product name like "Pure Cycles", "Electra Townie Org", "Electra 1...", "Surly Straggler", etc. A sidebar on the right titled "Overview" shows statistics: "Node labels" with "(25)" and "Product (25)".

```
neo4j$ MATCH (n:Product) RETURN n LIMIT 25
```

The screenshot shows the Neo4j browser interface with a single node highlighted. The node details are as follows:

```
n
{
    "identity": 3,
    "labels": [
        "product"
    ],
    "properties": {
        "category_id": "6",
        "product_id": "1",
        "model_year": "2016",
        "list_price": "379.99",
        "product_name": "Trek 820 - 2016",
        "brand_id": "9"
    }
}
```

```
neo4j$ MATCH (n:Product) RETURN n LIMIT 25
```

The screenshot shows the Neo4j browser interface displaying a list of product nodes. The results are as follows:

```
| {"category_id": "6", "product_id": "1", "model_year": "2016", "list_price": "379.99", "product_name": "Trek 820 - 2016", "brand_id": "9"} |
| {"category_id": "6", "product_id": "2", "model_year": "2016", "list_price": "749.99", "product_name": "Ritchey Timberwolf Frameset - 2016", "brand_id": "5"} |
| {"category_id": "6", "product_id": "3", "model_year": "2016", "list_price": "999.99", "product_name": "Surly Wednesday Frameset - 2016", "brand_id": "8"} |
| {"category_id": "6", "product_id": "4", "model_year": "2016", "list_price": "2899.99", "product_name": "Trek Fuel EX 8 29 - 2016", "brand_id": "9"} |
| {"category_id": "6", "product_id": "5", "model_year": "2016", "list_price": "1320.99", "product_name": "Heller Shagamaw Frame - 2016", "brand_id": "3"} |
| {"category_id": "6", "product_id": "6", "model_year": "2016", "list_price": "469.99", "product_name": "Surly Ice Cream Truck Frameset - 2016", "brand_id": "8"} |
| {"category_id": "6", "product_id": "7", "model_year": "2016", "list_price": "1999.99", "product_name": "Trek Superfly 9.8 2016", "brand_id": "9"} |
```

Order Node

```
neo4j$ LOAD CSV WITH HEADERS FROM 'file:///order.csv' AS row
      CREATE (n:Order) SET n=row
```

```
neo4j$
```

```
neo4j$ LOAD CSV WITH HEADERS FROM 'file:///order.csv' AS row CREATE (n...
```



Added 1615 labels, created 1615 nodes, set 12920 properties, completed after 90 ms.



```
neo4j$
```

```
neo4j$ MATCH (n:Order) RETURN n LIMIT 25
```

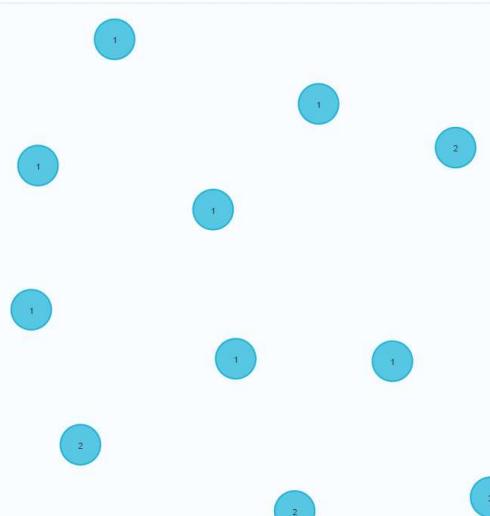


Overview

Node labels

* (25) Order (25)

Displaying 25 nodes, 0 relationships.



```
neo4j$
```

```
neo4j$ MATCH (n:Order) RETURN n LIMIT 25
```



n

```
{
  "identity": 324,
  "labels": [
    "Order"
  ],
  "properties": {
    "store_id": "1",
    "order_status": "4",
    "order_date": "2016-01-01",
    "required_date": "2016-01-03",
    "shipped_date": "2016-01-03",
    "staff_id": "2",
    "customer_id": "259",
    "order_id": "1"
  }
}
```

```
neo4j$ MATCH (n:Order) RETURN n LIMIT 25
```

Graph

Table

A Text

Code

```
| "n"  
|  
| {"store_id": "1", "order_status": "4", "order_date": "2016-01-01", "required_date": "2016-01-03", "shipped_date": "2016-01-03", "staff_id": "2", "custom_order_id": "259", "order_id": "1"}  
|  
| {"store_id": "2", "order_status": "4", "order_date": "2016-01-01", "required_date": "2016-01-04", "shipped_date": "2016-01-03", "staff_id": "6", "custom_order_id": "1212", "order_id": "2"}  
|  
| {"store_id": "2", "order_status": "4", "order_date": "2016-01-02", "required_date": "2016-01-05", "shipped_date": "2016-01-03", "staff_id": "7", "custom_order_id": "523", "order_id": "3"}  
|  
| {"store_id": "1", "order_status": "4", "order_date": "2016-01-03", "required_date": "2016-01-04", "shipped_date": "2016-01-05", "staff_id": "3", "custom_order_id": "175", "order_id": "4"}  
|  
| {"store_id": "2", "order_status": "4", "order_date": "2016-01-03", "required_date": "2016-01-06", "shipped_date": "2016-01-06", "staff_id": "6", "custom_order_id": "1324", "order_id": "5"}  
|
```

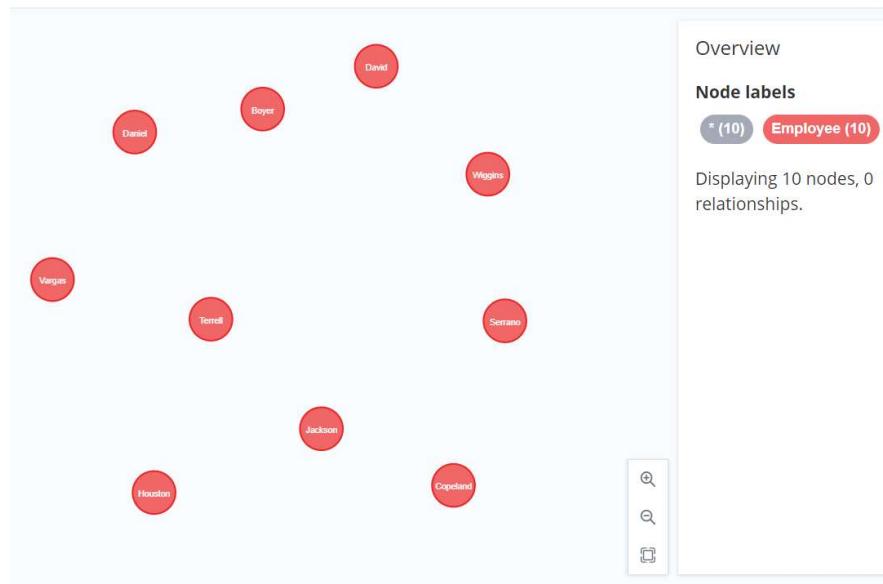
Employee Node

```
neo4j$ LOAD CSV WITH HEADERS FROM 'file:///employee.csv' AS row  
CREATE (n:Employee){row}
```



Added 10 labels, created 10 nodes, set 79 properties, completed after 24 ms.

```
$ MATCH (n:Employee) RETURN n LIMIT 25
```



```
neo4j$ MATCH (n:Employee) RETURN n LIMIT 25
```

The screenshot shows the Neo4j browser interface with a single result row. The left sidebar has tabs for Graph, Table (selected), and Text. The right panel displays the JSON representation of an employee node.

```
n
{
    "identity": 1939,
    "labels": [
        "Employee"
    ],
    "properties": {
        "store_id": "1",
        "phone": "(831) 555-5554",
        "staff_id": "1",
        "last_name": "Jackson",
        "active": "1",
        "first_name": "Fabiola",
        "email": "fabiola.jackson@bikes.shop"
    }
}
```

```
neo4j$ MATCH (n:Employee) RETURN n LIMIT 25
```

The screenshot shows the Neo4j browser interface with multiple results. The left sidebar has tabs for Graph, Table (selected), and Text. The right panel displays a list of employee nodes, each represented as a JSON object.

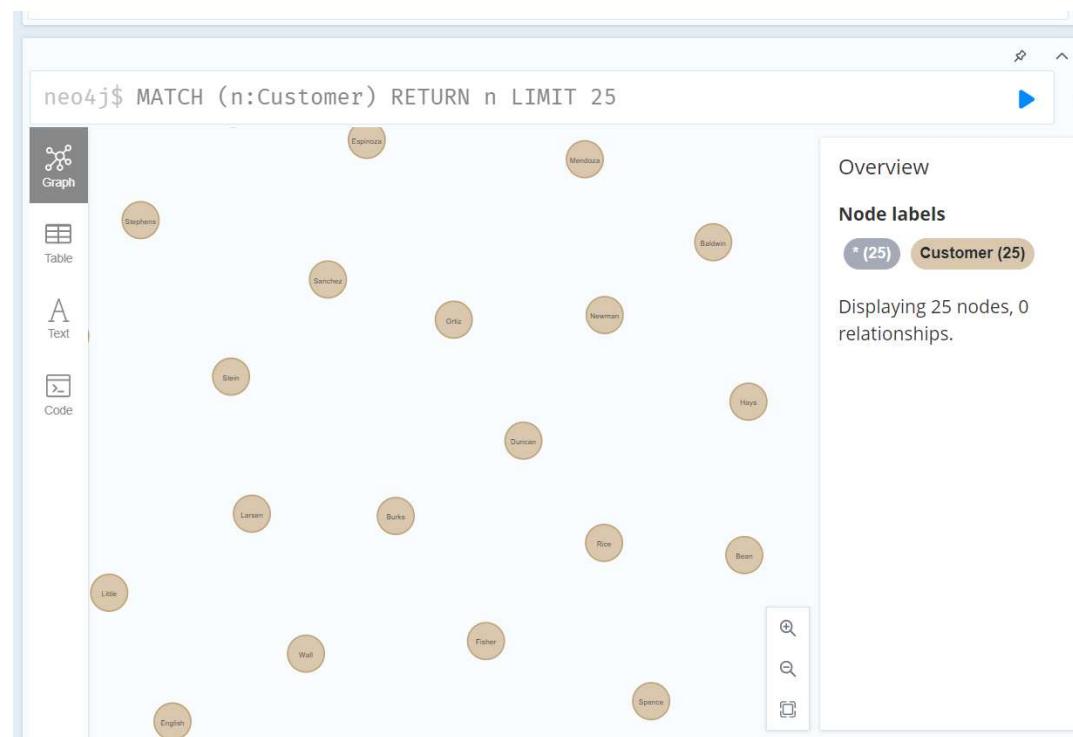
```
{
    "store_id": "1",
    "phone": "(831) 555-5554",
    "staff_id": "1",
    "last_name": "Jackson",
    "active": "1",
    "first_name": "Fabiola",
    "email": "fabiola.jackson@bikes.shop"
}
{
    "store_id": "1",
    "manager_id": "1",
    "phone": "(831) 555-5555",
    "staff_id": "2",
    "last_name": "Copeland",
    "active": "1",
    "first_name": "Mireya",
    "email": "mireya.copeland@bikes.shop"
}
{
    "store_id": "1",
    "manager_id": "2",
    "phone": "(831) 555-5556",
    "staff_id": "3",
    "last_name": "Serrano",
    "active": "1",
    "first_name": "Genna",
    "email": "genna.serrano@bikes.shop"
}
{
    "store_id": "1",
    "manager_id": "2",
    "phone": "(831) 555-5557",
    "staff_id": "4",
    "last_name": "Wiggins",
    "active": "1",
    "first_name": "Virgie",
    "email": "virgie.wiggins@bikes.shop"
}
{
    "store_id": "2",
    "manager_id": "1",
    "phone": "(516) 379-4444",
    "staff_id": "5",
    "last_name": "David",
    "active": "1",
    "first_name": "Jannette",
    "email": "jannette.david@bikes.shop"
}
{
    "store_id": "2",
    "manager_id": "5",
    "phone": "(516) 379-4445",
    "staff_id": "6",
    "last_name": "Boyer",
    "active": "1",
    "first_name": "Marcelene",
    "email": "marcelene.boyer@bikes.shop"
}
```

Customer Node

```
neo4j$ LOAD CSV WITH HEADERS FROM 'file:///customer.csv' AS row  
CREATE (n:Customer) SET n=row
```



Added 1445 labels, created 1445 nodes, set 13005 properties, completed after 90 ms.



```
neo4j$ MATCH (n:Customer) RETURN n LIMIT 25
```

Graph

Table

A Text

Code

```
n
1
{
  "identity": 1949,
  "labels": [
    "Customer"
  ],
  "properties": {
    "city": "Orchard Park",
    "street": "9273 Thorne Ave. ",
    "last_name": "Burks",
    "state": "NY",
    "customer_id": "1",
    "first_name": "Debra",
    "email": "debra.burks@yahoo.com",
    "zip_code": "14127"
  }
}
```

Graph

Table

A Text

Code

```
{"city":"Orchard Park","street":"9273 Thorne Ave. ","last_name":"Burks","state":"NY","customer_id":"1","first_name":"Debra","email":"debra.burks@yahoo.com","zip_code":"14127"} {"city":"Campbell","street":"910 Vine Street ","last_name":"Todd","state":"CA","customer_id":"2","first_name":"Kasha","email":"kasha.todd@yahoo.com","zip_code":"95008"} {"city":"Redondo Beach","street":"769C Honey Creek St. ","last_name":"Fisher","state":"CA","customer_id":"3","first_name":"Tameka","email":"tameka.fisher@aol.com","zip_code":"90278"} {"city":"Uniondale","street":"988 Pearl Lane ","last_name":"Spence","state":"NY","customer_id":"4","first_name":"Daryl","email":"daryl.spence@aol.com","zip_code":"11553"} {"phone": "(916) 381-6003", "city": "Sacramento", "street": "107 River Dr.", "last_name": "Rice", "state": "CA", "customer_id": "5", "first_name": "Charolette", "email": "charolette.rice@msn.com", "zip_code": "95820"}
```

Category Node

```
neo4j$ LOAD CSV WITH HEADERS FROM 'file:///category.csv' AS row  
CREATE (n:Category) SET n=row
```



Added 7 labels, created 7 nodes, set 14 properties, completed after 16 ms.



n

Graph

Table

Text

Code

```
1 {
    "identity": 3394,
    "labels": [
        "Category"
    ],
    "properties": {
        "category_name": "Children Bicycles",
        "category_id": "1"
    }
}

2 {
    "identity": 3395,
    "labels": [
        "Category"
    ]
}
```

```
eo4j$ MATCH (n:Category) RETURN n LIMIT 25
```

Graph

Table

Text

Code

```
"n"
|{"category_name":"Children Bicycles","category_id":"1"}
|{"category_name":"Comfort Bicycles","category_id":"2"}
| {"category_name":"Cruisers Bicycles","category_id":"3"}
| {"category_name":"Cyclocross Bicycles","category_id":"4"}
| {"category_name":"Electric Bikes","category_id":"5"}
| {"category_name":"Mountain Bikes","category_id":"6"}
| {"category_name":"Road Bikes","category_id":"7"}
```

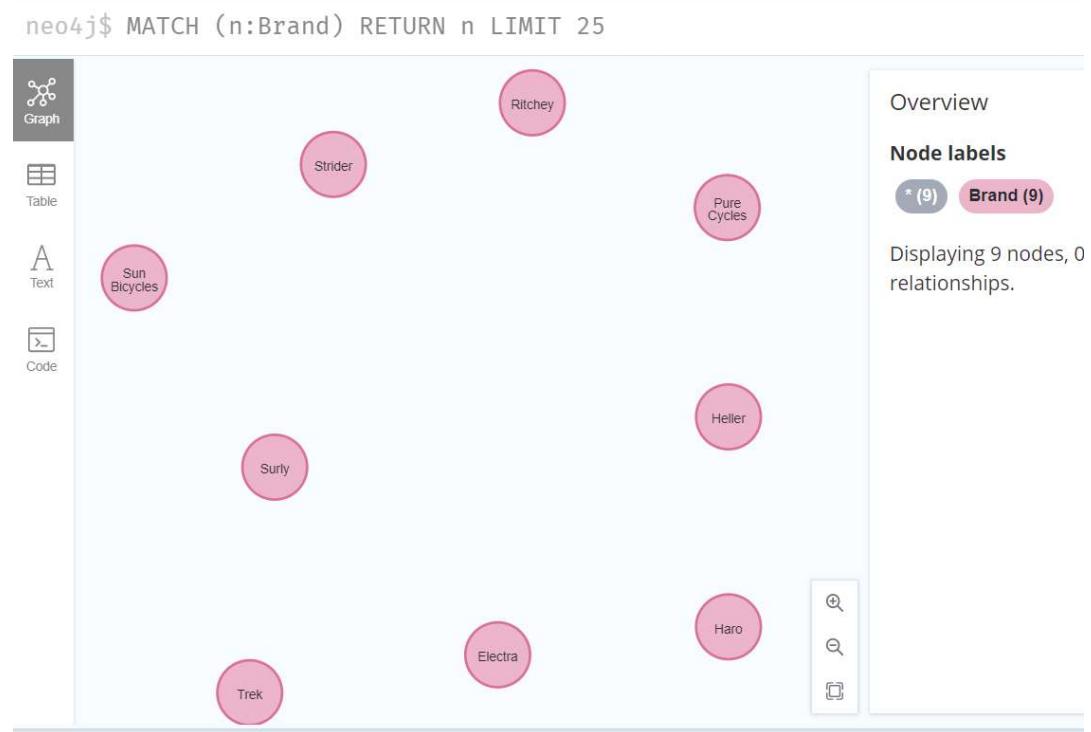
MAX COLUMN WIDTH:

Brand Node

```
neo4j$ LOAD CSV WITH HEADERS FROM 'file:///brand.csv' AS row CREATE  
(n:Brand)SET n=row
```



Added 9 labels, created 9 nodes, set 18 properties, completed after 16 ms.



```
neo4j$ MATCH (n:Brand) RETURN n LIMIT 25
```

Graph

Table

A
Text

Code

n

1

```
{  
    "identity": 3401,  
    "labels": [  
        "Brand"  
    ],  
    "properties": {  
        "brand_name": "Electra",  
        "brand_id": "1"  
    }  
}
```

2

```
{  
    "identity": 3402,  
    "labels": [  
        "Brand"  
    ],
```

```
neo4j$ MATCH (n:Brand) RETURN n LIMIT 25
```

Graph

Table

A
Text

Code

```
"n"  
{"brand_name":"Electra","brand_id":"1"}  
{"brand_name":"Haro","brand_id":"2"}  
{"brand_name":"Heller","brand_id":"3"}  
{"brand_name":"Pure Cycles","brand_id":"4"}  
{"brand_name":"Ritchey","brand_id":"5"}  
{"brand_name":"Strider","brand_id":"6"}  
{"brand_name":"Sun Bicycles","brand_id":"7"}  
{"brand_name":"Surly","brand_id":"8"}  
{"brand_name":"Trek","brand_id":"9"}
```

Order_Items Node

```
LOAD CSV WITH HEADERS FROM 'file:///order_items.csv' AS row  
CREATE (n:Order_Items) SET n=row
```

Added 4722 labels, created 4722 nodes, set 28332 properties, completed after 938 ms.

```
WITH order_items AS (SELECT * FROM order_items LIMIT 20)  
UNWIND order_items AS item  
MERGE (n:Order_Items {id: item.id})  
SET n.quantity = item.quantity,  
n.item_id = item.item_id,  
n.product_id = item.product_id,  
n.discount = item.discount,  
n.list_price = item.list_price,  
n.order_id = item.order_id
```

Now all the eight nodes have been loaded as shown in the fig. below:



Indexes

Product (product_id)

```
neo4j$ CREATE INDEX ON : Product(product_id)
```



Added 1 index, completed after 89 ms.

Store(store_id)

```
eo4j$ CREATE INDEX ON : Store(store_id)
```



Added 1 index, completed after 8 ms.

Order(order_id)

```
$ CREATE INDEX ON : Order(order_id)
```

Added 1 index, completed after 8 ms.

Employee(staff_id)

```
CREATE INDEX ON : Employee(staff_id)
```

Added 1 index, completed in less than 1 ms.

Customer(customer_id)

```
i CREATE INDEX ON : Customer(customer_id)
```

Added 1 index, completed in less than 1 ms.

Category(category_id)

```
j$ CREATE INDEX ON :Category(category_id)
```

Added 1 index, completed in less than 1 ms.

Brand(brand_id)

```
j$ CREATE INDEX ON :Brand(brand_id)
```

Added 1 index, completed after 16 ms.

Order_Items(item_id)

```
$ CREATE INDEX ON :Order_Items(item_id)
```

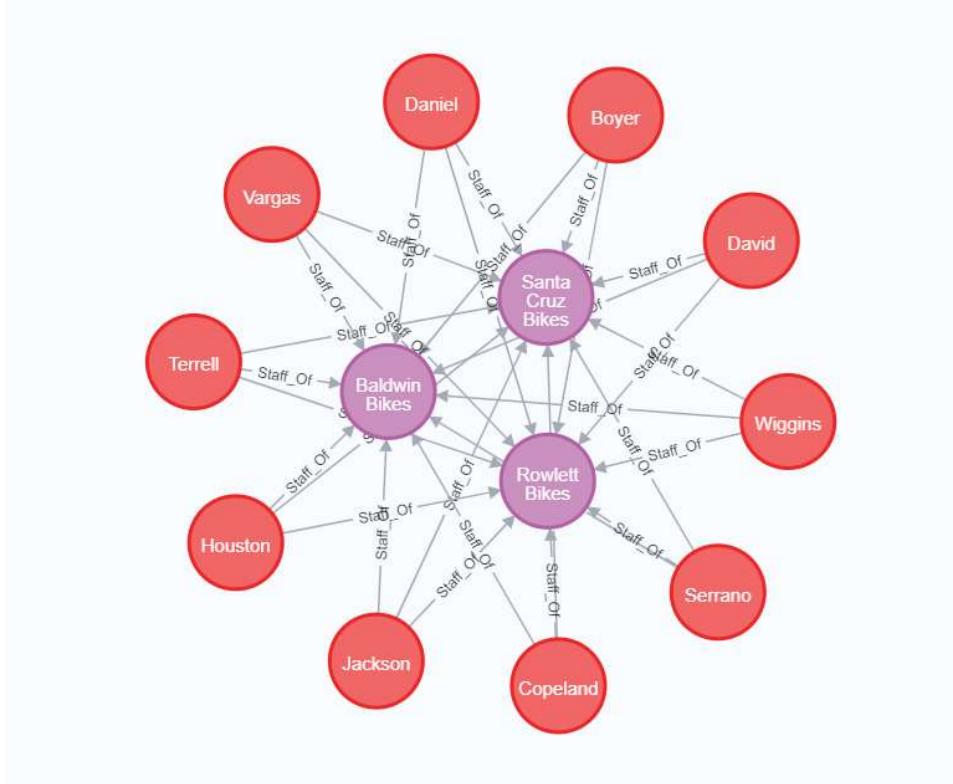
Added 1 index, completed after 112 ms.

Relationships

Employee and Store

```
> MATCH(e:Employee),(s:Store) WHERE e:store_id=s:store_id  
CREATE (e)-[:Staff_Of]→(s)
```

Created 30 relationships, completed after 16 ms.

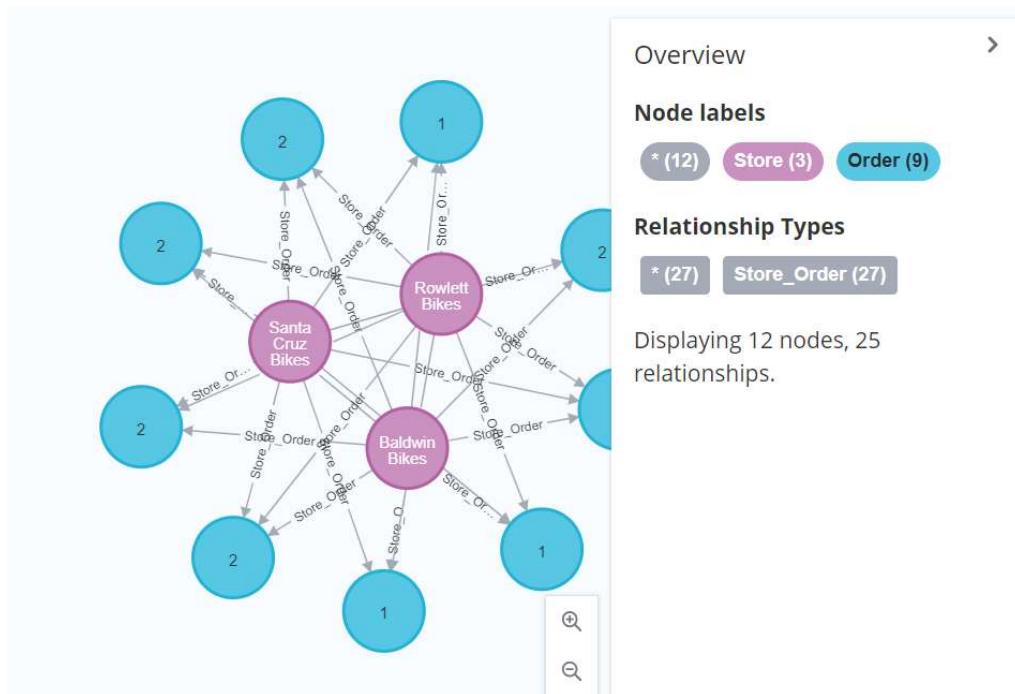


Store and Order

```

MATCH(s:Store),(o:Order) WHERE s: store_id=o:store_id CREATE
(s)-[:Store_Order]->(o)
  
```

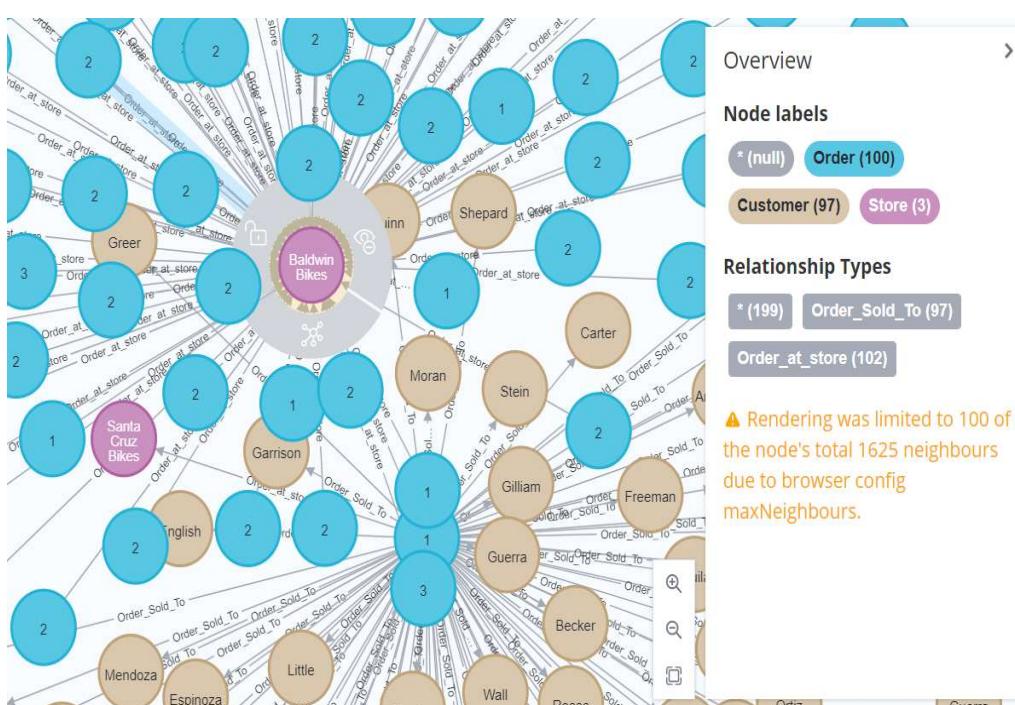
Created 4845 relationships, completed after 54 ms.



Order and Customer (Order_Sold_To)

```
MATCH(c:Customer),(o:Order) WHERE o:customer_id=c:customer_id
CREATE (o) - [ :Order_Sold_To]→(c)
```

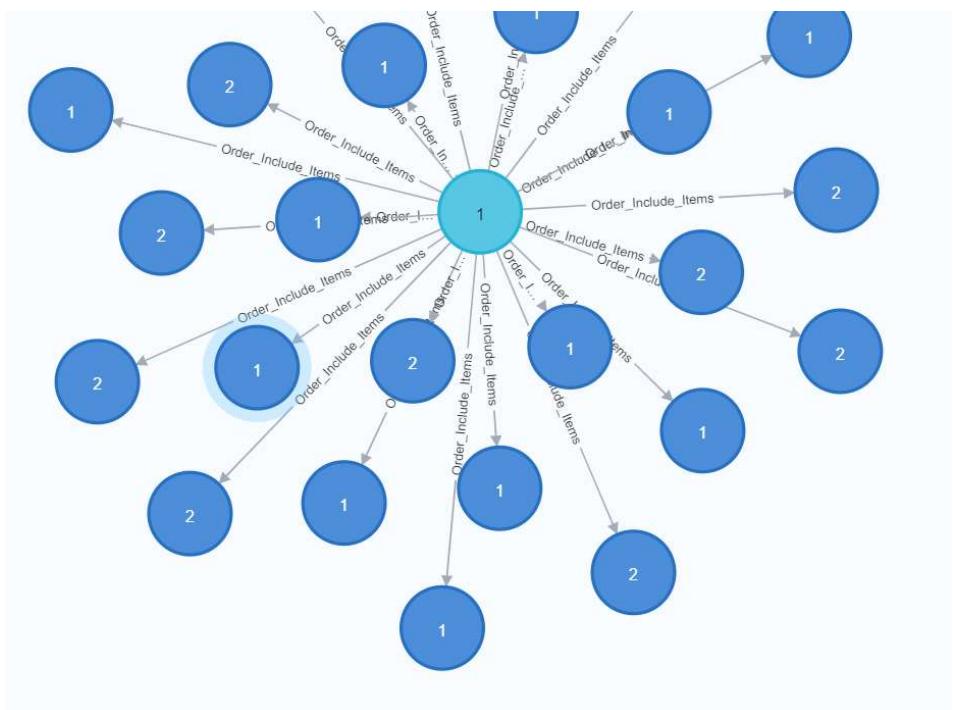
reated 2333675 relationships, completed after 8531 ms.



Order and Order_Item (Order_Include_Items)

```
MATCH(f:Order),(d:Order_Items) WHERE d: order_id=f:order_id CREATE (f)-[:Order_Include_Items]->(d)
```

Created 7626030 relationships, completed after 14646 ms.



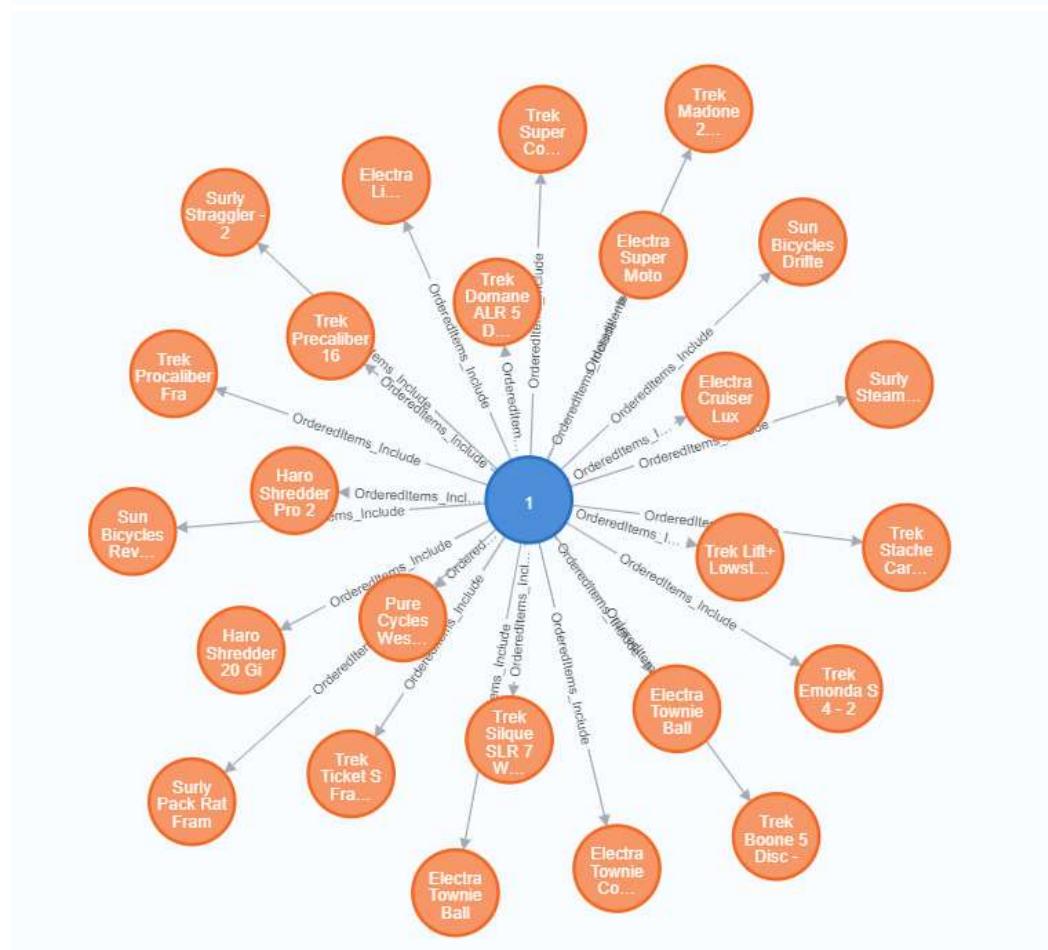
```
j$ MATCH p=()-[r:Order_Include_Items]->() RETURN p LIMIT 25
```

```
| "p"
| [{"store_id": "1", "order_status": "4", "order_date": "2016-01-01", "require| | d_date": "2016-01-03", "shipped_date": "2016-01-03", "staff_id": "2", "custo| | mer_id": "259", "order_id": "1"}, {}, {"quantity": "1", "item_id": "2", "produ| | ct_id": "16", "discount": ".20", "list_price": "599.99", "order_id": "172"}]
|
| [{"store_id": "1", "order_status": "4", "order_date": "2016-01-01", "require| | d_date": "2016-01-03", "shipped_date": "2016-01-03", "staff_id": "2", "custo| | mer_id": "259", "order_id": "1"}, {}, {"quantity": "1", "item_id": "2", "produ| | ct_id": "44", "discount": ".07", "list_price": "539.99", "order_id": "936"}]
|
| [{"store_id": "1", "order_status": "4", "order_date": "2016-01-01", "require| | d_date": "2016-01-03", "shipped_date": "2016-01-03", "staff_id": "2", "custo| | mer_id": "259", "order_id": "1"}, {}, {"quantity": "1", "item_id": "2", "produ| | ct_id": "2", "discount": ".07", "list_price": "749.99", "order_id": "97"}]
|
| [{"store_id": "1", "order_status": "4", "order_date": "2016-01-01", "require| | d_date": "2016-01-03", "shipped_date": "2016-01-03", "staff_id": "2", "custo| | mer_id": "259", "order_id": "1"}, {}, {"quantity": "1", "item_id": "2", "produ| | ct_id": "54", "discount": ".07", "list_price": "3199.99", "order_id": "800"}]
```

Order_Items and Product (OrderedItems_Includes)

```
MATCH(d:Order_Items),(p:Product) WHERE  
d:product_id=p:product_id CREATE (d)-[  
:OrderedItems_Include]->(p)
```

Created 1515762 relationships, completed after 3455 ms.



```

| "p"
|
| [{"quantity": "1", "item_id": "1", "product_id": "20", "discount": ".20", "list_price": "599.99", "order_id": "1"}, {}, {"category_id": "3", "product_id": "67", "model_year": "2017", "list_price": "250.99", "product_name": "Sun Bicycles Revolutions 24 - Girl's - 2017", "brand_id": "7"}]
|
| [{"quantity": "1", "item_id": "1", "product_id": "20", "discount": ".20", "list_price": "599.99", "order_id": "1"}, {}, {"category_id": "4", "product_id": "210", "model_year": "2018", "list_price": "1549.00", "product_name": "Surly Straggler - 2018", "brand_id": "8"}]
|
| [{"quantity": "1", "item_id": "1", "product_id": "20", "discount": ".20", "list_price": "599.99", "order_id": "1"}, {}, {"category_id": "5", "product_id": "189", "model_year": "2018", "list_price": "2799.99", "product_name": "Trek Lift+ Lowstep - 2018", "brand_id": "9"}]
|
| [{"quantity": "1", "item_id": "1", "product_id": "20", "discount": ".20", "list_price": "599.99", "order_id": "1"}, {}, {"category_id": "6", "product_id": "117", "model_year": "2018", "list_price": "1469.99", "product_name": "Trek Ticket S Frame - 2018", "brand_id": "9"}]
|

```

Product and Category (PART_OF)

```

neo4j$ MATCH(p:Product),(z:Category) WHERE p:category_id=z:category_id CREATE (p)-[:PART_OF]-(z)

```

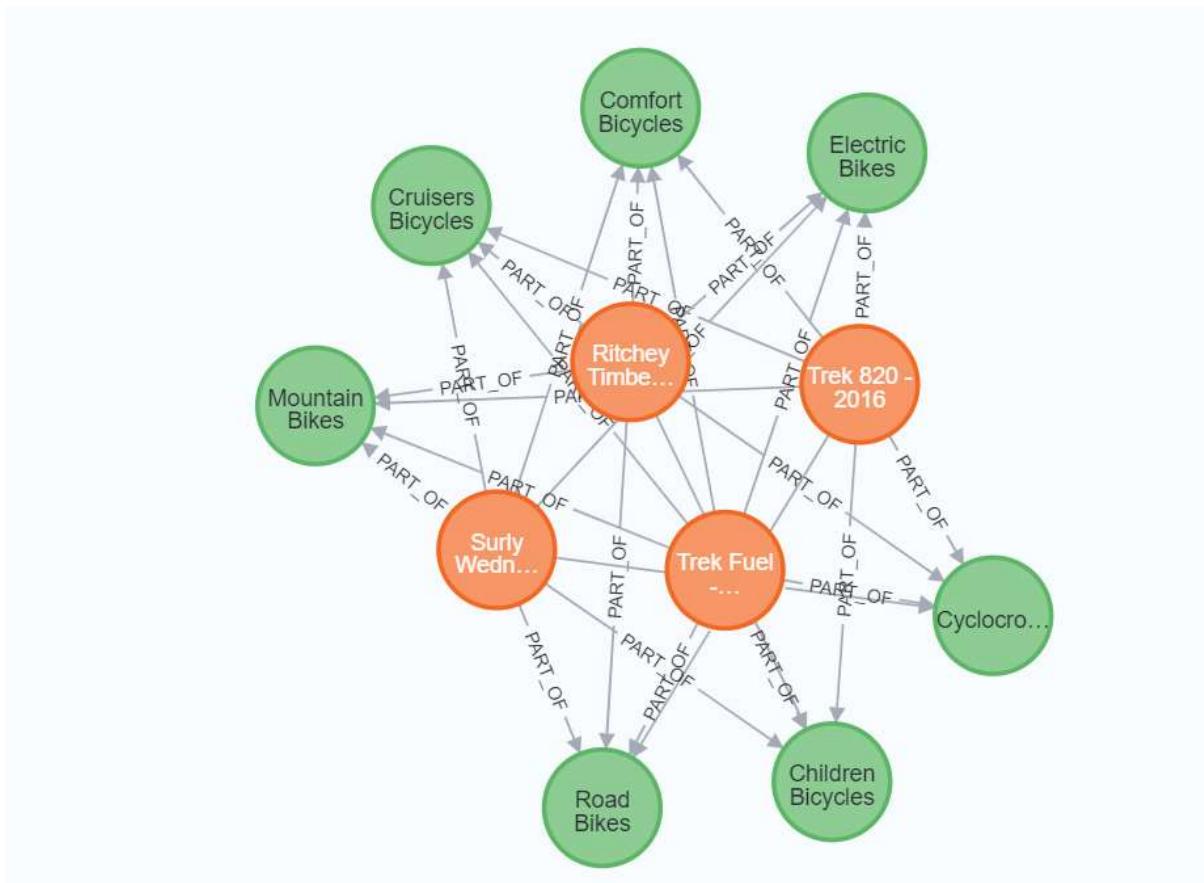
```

neo4j$ MATCH(p: Product),(z:Category) WHERE p:category_id=z:category_i...

```



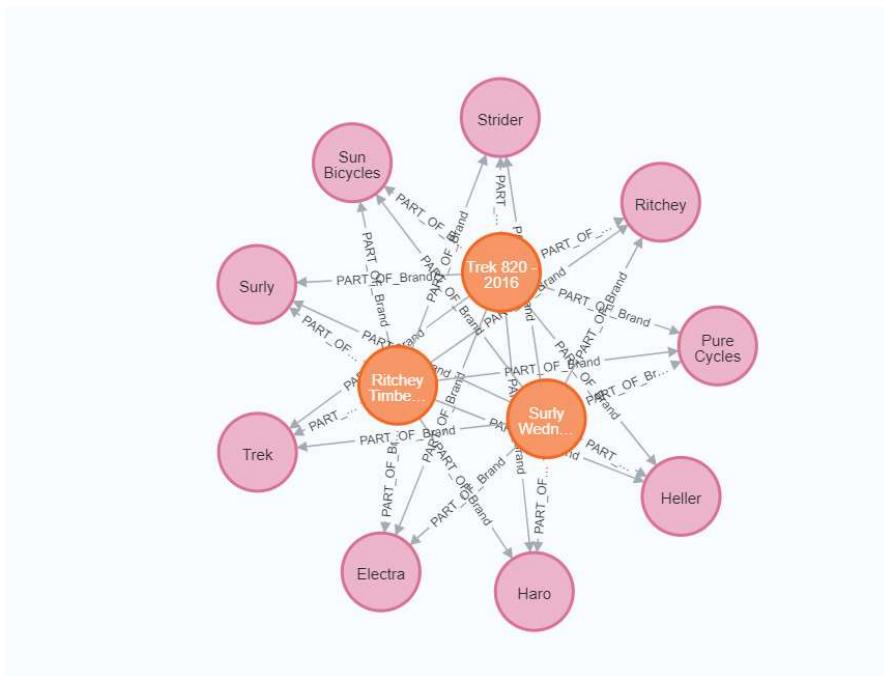
Created 2247 relationships, completed after 32 ms.



Product and Brand (PART_OF_Brand)

```
; MATCH(p:Product),(b:Brand) WHERE p: brand_id=b:brand_id
CREATE (p)-[ :PART_OF_Brand]→(b)
```

Created 2889 relationships, completed after 8 ms.



Overview

Node labels

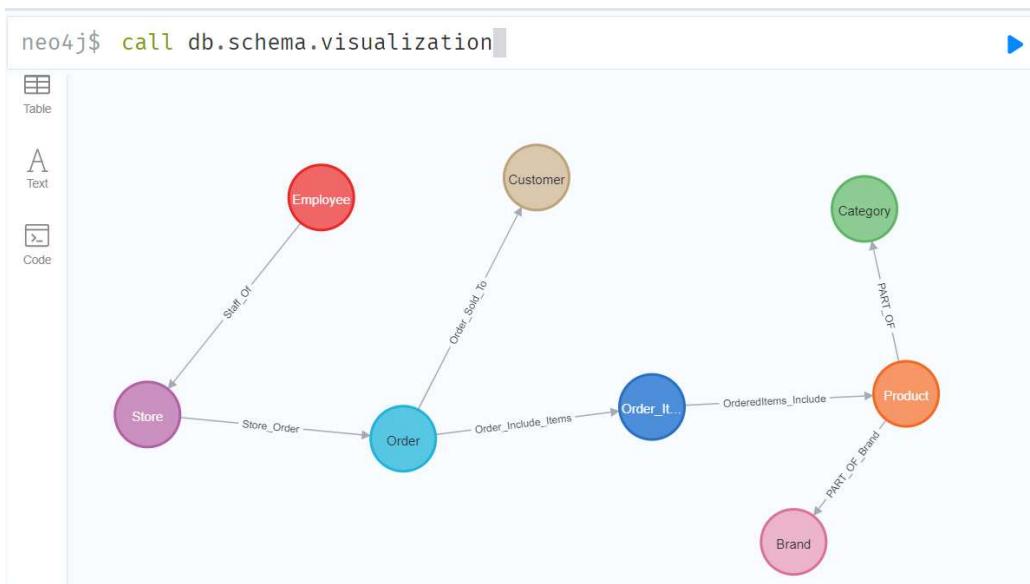
* (12) Product (3) Brand (9)

Relationship Types

* (27) PART_OF_Brand (27)

Displaying 12 nodes, 25 relationships.

Database Schema Visualization



Node 1	Node 2	Relationship
Employee	Store	Staff Of
Store	Order	Store Order
Order	Customer	Order Sold To
Order	Order Items	Order Include Items
Order Items	Product	OrderedItems Include
Product	Category	PART OF
Product	Brand	PART OF Brand

Queries: Cypher Versus SQL

Query1

To know the category of a product

```
neo4j$
```

```
neo4j$ MATCH(p:Product{product_name:'Surly Big Fat Dummy Frameset - 2018'})-(c:Category) WHERE
  EXISTS{MATCH(p)-[PART_OF]-(c) WHERE p.category_id=c.category_id} RETURN p.product_name AS
  ProductName, c.category_name AS CATEGORIES
```

	ProductName	CATEGORIES
A Text	"Surly Big Fat Dummy Frameset - 2018"	"Mountain Bikes"

Started streaming 1 records in less than 1 ms and completed in less than 1 ms.

SQLQuery1.sql - DE...O1T8J2I\seekr (65) X

```

select p.product_name ,c.category_name [category]
from
[production].[products] p  INNER JOIN [production].[categories] c ON p.category_id=c.category_id
where p.product_name ='Surly Big Fat Dummy Frameset - 2018'

```

100 % ▾

Results Messages Client Statistics

product_name	category
Surly Big Fat Dummy Frameset - 2018	Mountain Bikes

Client processing time 2 → 2.0000
Total execution time 9 → 9.0000
Wait time on server replies 7 → 7.0000

Query 2

To retrieve the information about the staff working at one of the stores

```

neo4j$ MATCH (e:Employee)-[:Staff_Of]->(s:Store) WHERE s.store_name='Baldwin Bikes' RETURN
s.store_name AS Store, s.email AS StoreEmail, e.email AS EmployeeEmail, e.first_name AS
FirstName, e.last_name AS LastName

```

Table

A text

Code

Store	StoreEmail	EmployeeEmail	FirstName	LastName
"Baldwin Bikes"	"baldwin@bikes.shop"	"bernardine.houston@bikes.shop"	"Bernardine"	"Houston"
"Baldwin Bikes"	"baldwin@bikes.shop"	"layla.terrell@bikes.shop"	"Layla"	"Terrell"
"Baldwin Bikes"	"baldwin@bikes.shop"	"kali.vargas@bikes.shop"	"Kali"	"Vargas"
"Baldwin Bikes"	"baldwin@bikes.shop"	"venita.daniel@bikes.shop"	"Venita"	"Daniel"
"Baldwin Bikes"	"baldwin@bikes.shop"	"marcelene.boyer@bikes.shop"	"Marcelene"	"Boyer"
"Baldwin Bikes"	"baldwin@bikes.shop"	"jannette.david@bikes.shop"	"Jannette"	"David"
"Baldwin Bikes"	"baldwin@bikes.shop"	"virgie.wiggins@bikes.shop"	"Virgie"	"Wiggins"
"Baldwin Bikes"	"baldwin@bikes.shop"	"genna.serrano@bikes.shop"	"Genna"	"Serrano"

neo4j\$ MATCH (e:Employee)-[:Staff_Of]->(s:Store) WHERE s.store_name='Baldwin Bikes' RETURN s.store_name AS Store, s.email AS StoreEmail, e.email AS EmployeeEmail, e.first_name AS FirstName, e.last_name AS LastName

Table

Server version Neo4j/4.4.5
Server address localhost:7687
Query MATCH (e:Employee)-[:Staff_Of]->(s:Store) WHERE s.store_name='Baldwin Bikes' RETURN s.store_name AS Store, s.email AS StoreEmail, e.email AS EmployeeEmail, e.first_name AS FirstName, e.last_name AS LastName
Summary { "query": { "text": "MATCH (e:Employee)-[:Staff_Of]->(s:Store) WHERE s.store_name='Baldwin Bikes' RETURN s.store_name AS Store, s.email AS StoreEmail, e.email AS EmployeeEmail, e.first_name AS FirstName, e.last_name AS LastName", ... } }
Response [{ "keys": [...] }

Started streaming 10 records in less than 1 ms and completed in less than 1 ms.

SQLQuery9.sql - DE...O1T8J2\seekr (56)*

```
SELECT
    st.store_name, st.email [store_email], st.email [employee_email], sf.first_name, sf.last_name
FROM
    [sales].[staffs] sf JOIN [sales].[stores] st ON sf.store_id=st.store_id
WHERE st.store_name='Baldwin Bikes'
```

100 %

Results Messages Client Statistics

	store_name	store_email	employee_email	first_name	last_name
1	Baldwin Bikes	baldwin@bikes.shop	baldwin@bikes.shop	Jannette	David
2	Baldwin Bikes	baldwin@bikes.shop	baldwin@bikes.shop	Marcelene	Boyer
3	Baldwin Bikes	baldwin@bikes.shop	baldwin@bikes.shop	Venita	Daniel

Time Statistics

	Client processing time	17	↑ 0	→ 8.5000
Total execution time		23	↑ 0	→ 11.5000
Wait time on server replies		6	↑ 0	→ 3.0000

Query 3

To retrieve the information about employee with last_name working at a store

```

1 MATCH (e:Employee{store_id:'1'})-->(s:Store{store_id:'1'})
2 WHERE EXISTS {
3   MATCH (e)-[:Staff_Of]-->(s)
4   WHERE e.last_name = 'Jackson'
5 }
6 RETURN e.last_name AS LastName, e.first_name AS FirstName,s.store_name AS StoreName, s.state AS State
7

```

Table

	LastName	FirstName	StoreName	State
1	"Jackson"	"Fabiola"	"Santa Cruz Bikes"	"CA"

Text

Code

Started streaming 1 records in less than 1 ms and completed in less than 1 ms.

SQLQuery3.sql - DE...O1T8J2\seekr (63) ▾ × SQLQuery2.sql - DE...O1T8J2\seekr (56)

```

SELECT
    sf.first_name,sf.last_name,st.store_name,st.state [state]
    FROM
        [sales].[staffs] sf JOIN [sales].[stores] st ON sf.store_id=st.store_id
    WHERE st.store_id=1 and sf.last_name='Jackson'

```

100 %

Results Messages Client Statistics

first_name	last_name	store_name	state
Fabiola	Jackson	Santa Cruz Bikes	CA

Time Statistics

Client processing time	1	→ 1	↑ 0	→ 0.6667
Total execution time	11	↑ 6	↓ 20	→ 12.3333
Wait time on server replies	10	↑ 5	↓ 20	→ 11.6667

Query 4

To retrieve the information about an order_id with product_id

```

1 MATCH (o:Order{order_id:'2'})-->(i:Order_Items{product_id:'20',order_id:'2'})-->
(p:Product{product_id:'20'})-->(b:Brand)
2 WHERE EXISTS {
3   MATCH (o)-[:Order_Include_Items]-->(i)-[:OrderedItems_Include]-->(p)-[:PART_OF_Brand]-->(b)
4   WHERE p.brand_id = b.brand_id
5 }
6 RETURN o.order_id AS OrderNum, p.product_id AS ProductNum,p.product_name AS ProductName,i.quantity AS QTY,b.brand_id AS BRAND,b.brand_name AS BrandName
7

```

Table

	OrderNum	ProductNum	ProductName	QTY	BRAND	BrandName
1	"2"	"20"	"Electra Townie Original 7D EQ - Women's - 2016"	"1"	"1"	"Electra"

A Text

Code

Started streaming 1 records in less than 1 ms and completed after 8 ms.

SQLQuery4.sql - DE...O1T8J2\seekr (56) X

```

SELECT
    oi.order_id, p.product_id, p.product_name, oi.quantity, b.brand_id, b.brand_name
FROM
    [sales].[order_items] oi INNER JOIN [production].[products] p ON oi.product_id=p.product_id
    INNER JOIN [production].[brands] b ON p.brand_id=b.brand_id
WHERE
    oi.order_id=2 and p.product_id=20

```

100 %

Results Messages Client Statistics

order_id	product_id	product_name	quantity	brand_id	brand_name
2	20	Electra Townie Original 7D EQ - Women's - 2016	1	1	Electra

Time Statistics

Client processing time	2	↑ 1	→ 1.5000
Total execution time	11	↓ 17	→ 14.0000
Wait time on server replies	9	↓ 16	→ 12.5000

Query 5

To retrieve the information including category, about order_id and product_id

```

1 MATCH (o:Order{order_id:'2'})-->(i:Order_Items{item_id:'1',order_id:'2'})-->(p:Product)-->
(c:Category)
2 WHERE EXISTS {
3   | MATCH (o)-[:Order_Include_Items]-->(i)-[:OrderedItems_Include]-->(p)-[:PART_OF]-->(c)
4 WHERE p.category_id = c.category_id AND p.product_id=i.product_id
5 }
6 RETURN o.order_id AS OrderNum,p.product_id AS ProductNum,p.product_name AS
ProductName,i.quantity AS QTY,c.category_id AS CATEGORY, c.category_name AS CategoryName
7

```

Table

	OrderNum	ProductNum	ProductName	QTY	CATEGORY	CategoryName
1	"2"	"20"	"Electra Townie Original 7D EQ - Women's - 2016"	"1"	"3"	"Cruisers Bicycles"

Code

Started streaming 1 records after 8 ms and completed after 24 ms.

```

SELECT
    oi.order_id, p.product_id, p.product_name, oi.quantity, c.category_id, c.category_name
FROM
    [sales].[order_items] oi INNER JOIN [production].[products] p ON oi.product_id=p.product_id
    INNER JOIN [production].[categories] c ON p.category_id=c.category_id
WHERE
    oi.order_id=2 and p.product_id=20

```

Results

order_id	product_id	product_name	quantity	category_id	category_name
1	2	Electra Townie Original 7D EQ - Women's - 2016	1	3	Cruisers Bicycles

Time Statistics

	2	12	10	49	38	34	14.2500	27.5000	13.2500
Client processing time	→ 2	↑ 11	↑ 9	↓ 49	↑ 38	↓ 34	→ 14.2500	→ 27.5000	→ 13.2500
Total execution time									
Wait time on server replies									

Query 6

To retrieve the information about a store_id e.g. '2', order_date e.g. '2018-04-28' about store name, order date, and customer's email.

```

1 MATCH (s:Store{store_id:'2'})-->(o:Order{order_date:'2018-04-28'})-->(c:Customer)
2 WHERE EXISTS {
3   | MATCH (s)-[:Store_Order]-->(o)-[:Order_Sold_To]-->(c)
4 WHERE s.store_id = o.store_id AND o.customer_id=c.customer_id
5 }
6 RETURN s.store_id AS StoreNum, s.store_name AS StoreName, o.order_date AS
OrderDate, c.customer_id AS CustomerNum, c.email as CustomerEmail
7

```

Table	StoreNum	StoreName	OrderDate	CustomerNum	CustomerEmail	
A	1	"2"	"Baldwin Bikes"	"2018-04-28"	"71"	"takako.casey@aol.com"
Code	2	"2"	"Baldwin Bikes"	"2018-04-28"	"45"	"bennett.armstrong@aol.com"

Started streaming 2 records in less than 1 ms and completed after 24 ms.

SQLQuery13.sql - D...O1T8J2\seekr (66)* × SQLQuery5.sql - DE...O1T8J2\seekr (63))

```
select s.store_id,s.store_name,o.order_date,o.customer_id,c.email
from [sales].[orders] o
inner join [sales].[stores] s on o.store_id=s.store_id
inner join [sales].[customers] c on o.customer_id=c.customer_id
Where o.order_date ='2018-04-28' and s.store_id=2
```

100 %

Results Messages Client Statistics

store_id	store_name	order_date	customer_id	email
1	Baldwin Bikes	2018-04-28	45	bennett.armstrong@aol.com
2	Baldwin Bikes	2018-04-28	71	takako.casey@aol.com

Time Statistics

Client processing time	1	→ 1.0000
Total execution time	4	→ 4.0000
Wait time on server replies	3	→ 3.0000

Query 7

To retrieve the information about order_id e.g. '4' when the order date is e.g. '2016-01-03' about ordered quantity, product name and list price.

```
1 MATCH (o:Order{order_date:'2016-01-03'})-->(i:Order_Items{order_id:'4'})-->(p:Product)
2 WHERE EXISTS {
3   | MATCH (o)-[:Order_Include_Items]-->(i)-[:OrderedItems_Include]-->(p)
4   WHERE i.order_id =o.order_id AND i.product_id=p.product_id
5 }
6 RETURN o.order_date AS OrderDate, i.order_id AS OrderNum,i.quantity AS QTY,p.product_name AS
ProductName,i.list_price AS ListPrice
7
```

Table

	OrderDate	OrderNum	QTY	ProductName	ListPrice
1	"2016-01-03"	"4"	"2"	"Ritche Timbervolf Frameset - 2016"	"749.99"

A Text

Code

Started streaming 1 records in less than 1 ms and completed after 16 ms.

```

SQLQuery17.Sql - D...O11621\seekr (67) - SQLQuery17.Sql - D...O11621\seekr (53) ~ ^

select o.order_date,o.order_id,oi.quantity,p.product_name,oi.list_price
from [sales].[orders] o
INNER JOIN [sales].[order_items] oi ON oi.order_id=o.order_id
INNER JOIN [production].[products] p ON oi.product_id=p.product_id
Where o.order_date ='2016-01-03' and o.order_id=4

100 % ▾

Results Messages Client Statistics
order_date order_id quantity product_name list_price
1 2016-01-03 4 2 Ritchey Timberwolf Frameset - 2016 749.99

Time Statistics
Client processing time 2 → 2.0000
Total execution time 10 → 10.0000
Wait time on server replies 8 → 8.0000

```

Conclusion

My Learning

MS SQL vs Cypher		
Queries	Execution Time (ms)	
	MS SQL Server	Cypher
Query 1	9	After 1
Query 2	11.5	After 1
Query 3	12.3333	After 1
Query 4	14	After 8
Query 5	27.5	After 24
Query 6	4	After 24
Query 7	10	After 16

Table 1

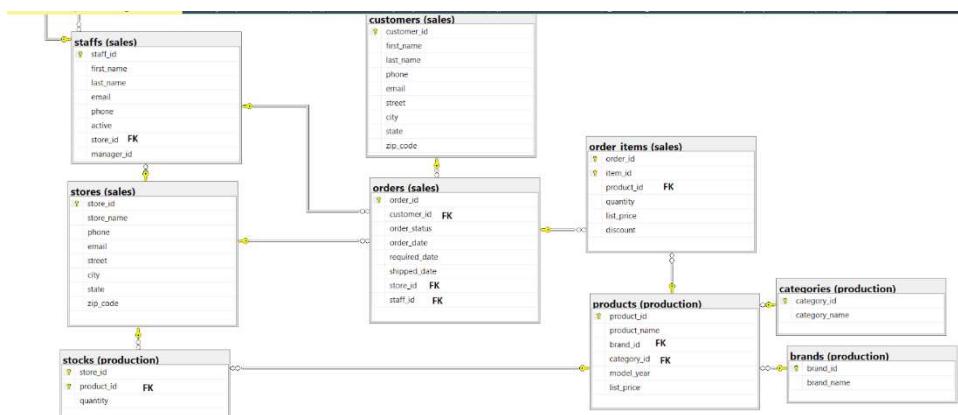
SQL queries for deep hierarchies are complex and slow while Neo4J queries are concise and fast. Table 1 shows that cypher is faster as compared to SQL.

Relational databases do not store the relationship between data. Graph Databases store relationship between data as entities.

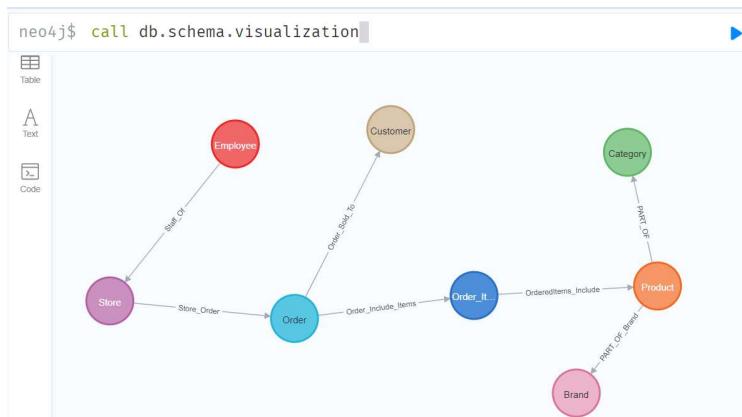
Relational Database	Graph Databases Neo4J
Tables	Graphs
Rows	Nodes
Constraints	Relationships
Columns	Properties
Use of joins	No use of joins
Use of Schema	Schema free
Structured Data	All
Secure	No security
Less or no scalable	High scalability
ACID compliant	ACID compliant

(Hunger, Boyd and Lyon, 2016)

ERD of the operational database



As shown in ERD, it is difficult to match in-memory data structures to the relational tables. While in graph database, as in figure below, it is easy to match the data structures to the relational tables.



References

'Comparing SQL with Cypher' *Neo4J Developer*, Available at: [Comparing SQL with Cypher - Developer Guides \(neo4j.com\)](#)

Hunger, M., Boyd, R. and Lyon, W. (2016)'RDMS & Graphs: SQL vs. Cypher Query Languages', *neo4j*, Available at: [RDBMS & Graphs: SQL vs. Cypher Query Languages \(neo4j.com\)](#)

Lydon, B. (2022), 'On Demand Lesson 5 -NoSQL Databases: The Emergence of NoSQL', *B9DA111_2122_TMD3: Data Storage Solutions for Data Analytics*. Dublin Business School. Available at:<https://study.dbs.ie/2122/msc-data/B9DA111/Unit5/index.html#/lessons/KConnDRWyCDPTnj-mSGDRz2THNIFP4ab>