# Sprawozdanie z zadania z grafowych baz danych

Magdalena Nowak

Przygotowałam aplikację korzystając z wersji embedded - wbudowanego Neo4j. Stworzyłam ją w oparciu o projekt zamieszczony na moodle. Tworzyłam zapytania w języku Cypher.

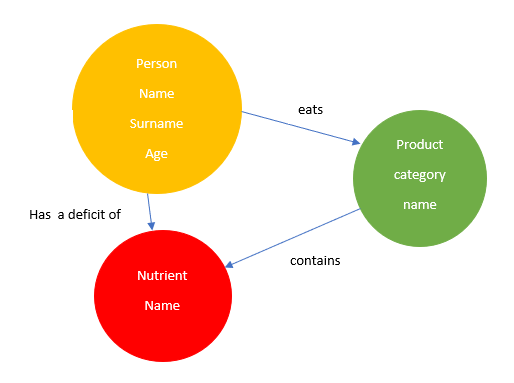
Kod dostępny tutaj ->

https://github.com/McDusia/Databases/tree/master/Neo4j

## Poniżej zamieszczono graf, który odwzorowano kolejno w bazie danych.

Baza będzie przechowywała dane osób oraz produkty, które spożywają, a także składniki odżywcze, jakich mają niedobory.

Dzięki analizie danych można np. osobie zasugerować spożywanie jakiegoś produktu, którego nie spożywa, zawierający składnik, którego ma nieobór. Oczywiście jedna osoba może jeść wiele produktów i mieć niedobory wielu składników, podobnie produkty zawierają wiele składników.



## Następnie napisałam funkcje do tworzenia obiektów i relacji.

Klasy reprezentujące poszczególne węzły:

**class** Product {  
 **public** String **category**;  
 **public** String **name**;  
  
 Product (String c, String n) {  
 **category** = c;  
 **name** = n;  
 }  
}  
  
**class** Nutrient {  
 **public** String **name**;  
 Nutrient (String n) {  
 **name** = n;  
 }  
}  
  
**class** Person {  
 **public** String **name**;  
 **public** String **surname**;  
 **public int age**;  
  
 Person(String n, String s, **int** a) {  
 **name** = n;  
 **surname** = s;  
 **age** = a;  
 }  
}

Metody dodające node węzły:

**private** String addProduct( String category, String productName) {  
  
 String query = String.*format*(**"CREATE (n:Product {category: '%s', name: '%s'}) return n"**,  
 category, productName);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String addPerson(String name, String surname, **int** age) {  
  
 String query = String.*format*(**"CREATE (n:Person { name: '%s', surname: '%s', age: %d}) return n"** , name, surname, age);  
 **return graphDatabase**.runCypher(query);  
}

**private** String addNutrient(String name) {  
  
 String query = String.*format*(**"CREATE (n:Nutrient { name: '%s' }) return n"** ,name);  
 **return graphDatabase**.runCypher(query);  
}

Metody dodające relacje:

**private** String linkProductAndNutrient(String category, String name, String nutrientName) {  
  
 String query = String.*format*(**"MATCH (p:Product),(n:Nutrient)"**+  
 **"WHERE p.category = '%s' AND p.name = '%s' AND n.name = '%s' "**+  
 **"CREATE (p)-[r:CONTAINS]->(n) RETURN r"**, category, name, nutrientName);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String linkPersonAndProduct(String name, String personName,  
 String personSurname) {  
  
 String query = String.*format*(**"MATCH (p:Person),(pr:Product)"**+  
 **"WHERE p.name = '%s' AND p.surname = '%s' "** +  
 **"AND pr.name = '%s' "**+  
 **"CREATE (p)-[r:EATS]->(pr) RETURN r"**, personName, personSurname, name);  
  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String linkPersonAndDeficit(String name, String surname, String nutrientName) {  
  
 String query = String.*format*(**"MATCH (p:Person),(n:Nutrient)"**+  
 **"WHERE p.name = '%s' AND p.surname = '%s' AND "**+  
 **"n.name = '%s' CREATE (p)-[r:HAS\_A\_DEFICIT\_OF]->(n) RETURN r"**,  
 name, surname, nutrientName);  
  
 **return graphDatabase**.runCypher(query);  
}

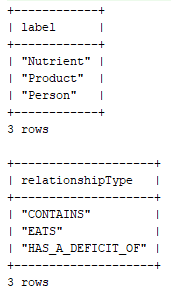
## Populator danych:

Funkcja dodająca węzły i relacje:

**public void** addData() {  
  
 List<Nutrient> nutrients = **new** ArrayList<>();  
 nutrients.add(**new** Nutrient(**"witamina A"**));  
 nutrients.add(**new** Nutrient(**"witamina C"**));  
 nutrients.add(**new** Nutrient(**"witamina D"**));  
 nutrients.add(**new** Nutrient(**"potas"**));  
 nutrients.add(**new** Nutrient(**"magnez"**));  
  
 List<Product> products = **new** ArrayList<>();  
 products.add(**new** Product(**"owoce"**,**"banan"**));  
 products.add(**new** Product(**"warzywa"**,**"pomidor"**));  
 products.add(**new** Product(**"owoce"**,**"truskawka"**));  
 products.add(**new** Product(**"owoce"**,**"cytryna"**));  
 products.add(**new** Product(**"nabiał"**,**"śmietana"**));  
 products.add(**new** Product(**"nabiał"**,**"jogurt"**));  
 products.add(**new** Product(**"nabiał"**,**"masło"**));  
 products.add(**new** Product(**"nabiał"**,**"ser żółty"**));  
 products.add(**new** Product(**"warzywa"**,**"marchewka"**));  
  
 List<Person> people = **new** ArrayList<>();  
 people.add(**new** Person(**"Jan"**,**"Kowalski"**, 50));  
 people.add(**new** Person(**"Helena"**,**"Kowalska"**, 42));  
 people.add(**new** Person(**"Jacek"**,**"Nowak"**, 13));  
 people.add(**new** Person(**"Michał"**,**"Naj"**, 24));  
 people.add(**new** Person(**"Zofia"**,**"Lis"**, 40));  
  
 **for**(Nutrient nutrient : nutrients) {  
 addNutrient(nutrient.**name**);  
 }  
  
 **for**(Product product : products) {  
 addProduct(product.**category**, product.**name**);  
 }  
  
 **for**(Person person : people) {  
 addPerson(person.**name**, person.**surname**, person.**age**);  
 }  
  
 *//add dependency between products and nutrients* linkProductAndNutrient(**"owoce"**,**"banan"**, **"potas"**);  
 linkProductAndNutrient(**"owoce"**,**"banan"**, **"magnez"**);  
 linkProductAndNutrient(**"warzywa"**,**"pomidor"**, **"potas"**);  
 linkProductAndNutrient(**"warzywa"**,**"pomidor"**, **"witamina C"**);  
 linkProductAndNutrient(**"owoce"**,**"truskawka"**,**"witamina C"**);  
 linkProductAndNutrient(**"owoce"**,**"cytryna"**,**"witamina C"**);  
 linkProductAndNutrient(**"nabiał"**,**"śmietana"**,**"witamina D"**);  
 linkProductAndNutrient(**"nabiał"**,**"jogurt"**,**"witamina D"**);  
 linkProductAndNutrient(**"nabiał"**,**"masło"**,**"witamina A"**);  
 linkProductAndNutrient(**"warzywa"**,**"marchewka"**,**"witamina A"**);  
 linkProductAndNutrient(**"nabiał"**,**"ser żółty"**,**"witamina A"**);  
  
 *//add dependency between products and people* linkPersonAndProduct(**"banan"**, **"Jan"**,**"Kowalski"**);  
 linkPersonAndProduct(**"truskawka"**, **"Jan"**,**"Kowalski"**);  
 linkPersonAndProduct(**"cytryna"**, **"Jan"**,**"Kowalski"**);  
 linkPersonAndProduct(**"cytryna"**,**"Jacek"**,**"Nowak"**);  
 linkPersonAndProduct(**"śmietana"**, **"Jacek"**,**"Nowak"**);  
 linkPersonAndProduct(**"jogurt"**, **"Jacek"**,**"Nowak"**);  
 linkPersonAndProduct(**"banan"**, **"Jacek"**,**"Nowak"**);  
 linkPersonAndProduct(**"marchewka"**, **"Zofia"**,**"Lis"**);  
 linkPersonAndProduct(**"banan"**, **"Zofia"**,**"Lis"**);  
 linkPersonAndProduct(**"masło"**, **"Jacek"**,**"Nowak"**);  
  
 *//add dependency between people and deficit of nutrient* linkPersonAndDeficit(**"Zofia"**,**"Lis"**, **"witamina D"**);  
 linkPersonAndDeficit(**"Zofia"**,**"Lis"**, **"witamina C"**);  
 linkPersonAndDeficit(**"Jacek"**,**"Nowak"**, **"potas"**);  
 linkPersonAndDeficit(**"Jacek"**,**"Nowak"**, **"magnez"**);  
 linkPersonAndDeficit(**"Jan"**,**"Kowalski"**, **"witamina D"**);  
 linkPersonAndDeficit(**"Jan"**,**"Kowalski"**, **"witamina A"**);  
 linkPersonAndDeficit(**"Jan"**,**"Kowalski"**, **"potas"**);  
 linkPersonAndDeficit(**"Michał"**,**"Naj"**, **"potas"**);  
 linkPersonAndDeficit(**"Michał"**,**"Naj"**, **"magnez"**);  
 linkPersonAndDeficit(**"Michał"**,**"Naj"**, **"witamina A"**);  
  
}

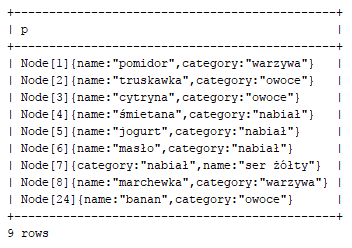
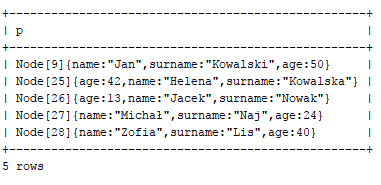
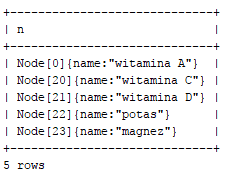
Po wywołaniu:

**public void** databaseStatistics() {  
 System.***out***.println(**graphDatabase**.runCypher(**"CALL db.labels()"**));  
 System.***out***.println(**graphDatabase**.runCypher(**"CALL db.relationshipTypes()"**));  
}

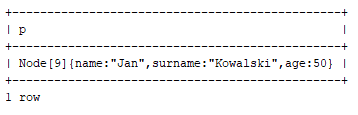


Metody dostępowe:

**private** String getPeople() {  
  
 String query = String.*format*(**"MATCH (p:Person) RETURN p"**);  
  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String getProducts() {  
  
 String query = String.*format*(**"MATCH (p:Product) RETURN p"**);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String getNutrients() {  
  
 String query = String.*format*(**"MATCH (n:Nutrient) RETURN n"**);  
 **return graphDatabase**.runCypher(query);  
}



**private** String findPerson(String name, String surname) {  
  
 String query = String.*format*(**"MATCH (p:Person)"**+  
 **"WHERE p.name = '%s' AND p.surname = '%s' RETURN p"**,  
 name, surname);  
  
 **return graphDatabase**.runCypher(query);  
}

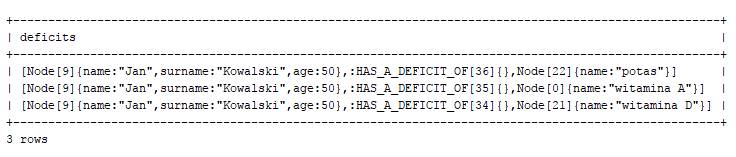


## Pobieranie wszystkich relacji dla danego węzła

Pobieranie po jednym typie relacji dla węzła (format: węzeł – relacja - węzeł):

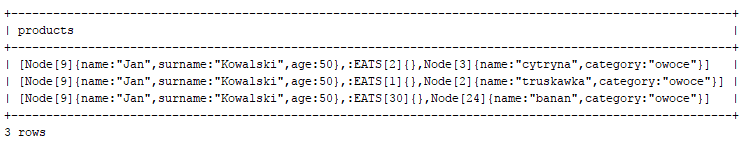
**private** String findPersonDeficits(String name, String surname) {  
  
 String query = String.*format*(**"MATCH deficits = (p:Person)-[r :HAS\_A\_DEFICIT\_OF]->(n:Nutrient) "** +  
 **"WHERE p.name = '%s' AND p.surname='%s' RETURN deficits"**, name, surname);  
 **return graphDatabase**.runCypher(query);  
}

System.***out***.println(findPersonDeficits(**"Jan"**, **"Kowalski"**));



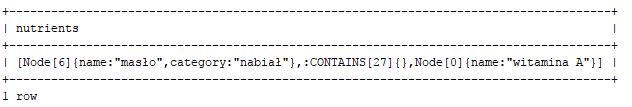
**private** String findProductsEatenByPerson(String name, String surname) {  
  
 String query = String.*format*(**"MATCH products = (p:Person)-[r :EATS]->(pr:Product) "** +  
 **"WHERE p.name = '%s' AND p.surname='%s' RETURN products"**, name, surname);  
 **return graphDatabase**.runCypher(query);  
}

System.***out***.println(findProductsEatenByPerson(**"Jan"**, **"Kowalski"**));



**private** String findNutrientsInProduct(String name) {  
  
 String query = String.*format*(**"MATCH nutrients = (p:Product)-[r :CONTAINS]->(n:Nutrient) "** +  
 **"WHERE p.name = '%s' RETURN nutrients"**, name);  
 **return graphDatabase**.runCypher(query);  
}

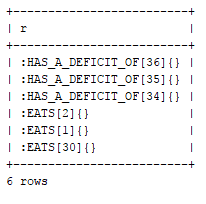
System.***out***.println(findNutrientsInProduct(**"masło"**));



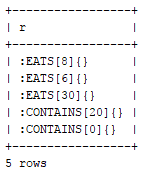
Pobieranie wszystkich relacji dla danego typu węzła (format: sama relacja):

**private** String getRelationshipsForProduct(String name) {  
  
 String query = String.*format*(**"MATCH (p:Product)-[r]-(n) "** +  
 **"WHERE p.name = '%s' RETURN r"**, name);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String getRelationshipsForPerson(String name, String surname) {  
  
 String query = String.*format*(**"MATCH (p:Person)-[r]-(n) "** +  
 **"WHERE p.name = '%s' AND p.surname = '%s' RETURN r"**, name, surname);  
 **return graphDatabase**.runCypher(query);  
}  
**private** String getRelationshipsForNutrient(String name) {  
  
 String query = String.*format*(**"MATCH (n:Nutrient)-[r]-(a) "** +  
 **"WHERE n.name = '%s' RETURN r"**, name);  
 **return graphDatabase**.runCypher(query);  
}

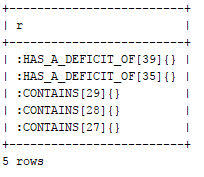
System.***out***.println(getRelationshipsForPerson(**"Jan"**, **"Kowalski"**));



System.***out***.println(getRelationshipsForProduct(**"banan"**));



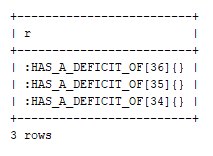
System.***out***.println(getRelationshipsForNutrient(**"witamina A"**));



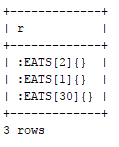
Pobieranie relacji jednego typu dla węzła:

**private** String getRelationshipsToNutrientsForProduct(String name) {  
  
 String query = String.*format*(**"MATCH (p:Product)-[r]->(n) "** +  
 **"WHERE p.name = '%s' RETURN r"**, name);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String getRelationshipsToProductsForPerson(String name, String surname) {  
  
 String query = String.*format*(**"MATCH (p:Person)-[r]->(n: Product) "** +  
 **"WHERE p.name = '%s' AND p.surname = '%s' RETURN r"**, name, surname);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String getRelationshipsToNutrientsForPerson(String name, String surname) {  
  
 String query = String.*format*(**"MATCH (p:Person)-[r]->(n: Nutrient) "** +  
 **"WHERE p.name = '%s' AND p.surname = '%s' RETURN r"**, name, surname);  
 **return graphDatabase**.runCypher(query);  
}

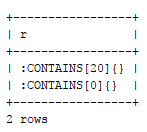
System.***out***.println(getRelationshipsToNutrientsForPerson(**"Jan"**, **"Kowalski"**));



System.***out***.println(getRelationshipsToProductsForPerson(**"Jan"**, **"Kowalski"**));



System.***out***.println(getRelationshipsToNutrientsForProduct(**"banan"**));

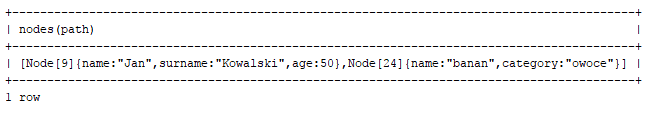


## Funkcje znajdujące ścieżki dla danych dwóch węzłów

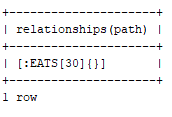
*//-------------- Find paths ----------------***private** String findPathsPersonToProducts\_Nodes(String name, String surname, String prName) {  
  
 String query = String.*format*(**"MATCH path = (p:Person)--(pr:Product) "** +  
 **"WHERE p.name = '%s' AND p.surname='%s' AND pr.name='%s' RETURN nodes(path)"**, name,surname,prName);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String findPathsPersonToProducts\_Relationships(String name, String surname, String prName) {  
  
 String query = String.*format*(**"MATCH path = (p:Person)--(pr:Product) "** +  
 **"WHERE p.name = '%s' AND p.surname='%s' AND pr.name='%s' RETURN relationships(path)"**, name,surname,prName);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String findPathsPersonToNutrient\_Nodes(String name, String surname, String nName) {  
  
 String query = String.*format*(**"MATCH path = (p:Person)--(n:Nutrient) "** +  
 **"WHERE p.name = '%s' AND p.surname='%s' AND n.name='%s' RETURN nodes(path)"**, name,surname,nName);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String findPathsPersonToNutrient\_Relationships(String name, String surname, String nName) {  
  
 String query = String.*format*(**"MATCH path = (p:Person)--(n:Nutrient) "** +  
 **"WHERE p.name = '%s' AND p.surname='%s' AND n.name='%s' RETURN relationships(path)"**,  
 name,surname,nName);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String findPathsProductToNutrient\_Nodes(String pName, String nName) {  
  
 String query = String.*format*(**"MATCH path = (p:Product)--(n:Nutrient) "** +  
 **"WHERE p.name = '%s' AND n.name='%s' RETURN nodes(path)"**, pName,nName);  
 **return graphDatabase**.runCypher(query);  
}  
  
**private** String findPathsProductToNutrient\_Relationships(String pName, String nName) {  
  
 String query = String.*format*(**"MATCH path = (p:Product)--(n:Nutrient) "** +  
 **"WHERE p.name = '%s' AND n.name='%s' RETURN relationships(path)"**,  
 pName,nName);  
 **return graphDatabase**.runCypher(query);  
}

Przykłady użycia:

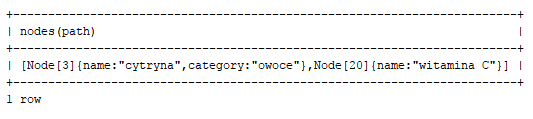
System.***out***.println(findPathsPersonToProducts\_Nodes(**"Jan"**, **"Kowalski"**,**"banan"**));



System.***out***.println(findPathsPersonToProducts\_Relationships(**"Jan"**, **"Kowalski"**,**"banan"**));



System.***out***.println(findPathsProductToNutrient\_Nodes(**"cytryna"**, **"witamina C"**));



System.***out***.println(findPathsProductToNutrient\_Relationships(**"cytryna"**, **"witamina C"**));

