Государственное учреждение образования

“БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ”

Кафедра: Интеллектуальных информационных технологий

Дисциплина: Проектирование баз знаний

Отчет по лабораторной работе №2

**“Разработка и выравнивание онтологий”**

Выполнил:

студент гр.121702

Витковская С. И.

Проверил:

Липницкая Н. Г.

Минск 2024

**Содержание**

[**Задание 3**](#_heading=h.gjdgxs)

[**Разработка онтологии 4**](#_heading=h.pw2kpt5jo7d2)

[**Запросы к собственной онтологии 6**](#_heading=h.30j0zll)

[**Отображение готовых онтологий в Neo4j 8**](#_heading=h.v9gcdl5ul5r2)

[**Вывод 10**](#_heading=h.a96hlklimqai)

#### **Задание**

Цель работы: приобрести навыки разработки онтологий предметных областей.

Задачи:

1. Изучить принципы создания онтологий (см. Материалы для ознакомления)
2. Выбрать инструментальное средство для разработки онтологии (например, Protege).
3. Тему онтологии использовать из Лабораторной работы №1 (для интеллектуальной карты).
4. Найти 2-3 онтологии совпадающих или близких к предметной области.
5. Разработать онтологию по выбранной предметной области используя инструментальное средство (например, Protege). Предусмотреть описание не менее 10 классов сущностей выбранной предметной области, у каждого класса не менее 2 слотов, у каждого класса не менее 2 экземпляров.
6. Готовые онтологии (пункт 3) загрузить в одну онтологию.
7. Создать к разработанной онтологии 5 различных запросов.
8. Сделать 3 запроса, показывающих использование информации из различных онтологий (SparQL).

По результатам работы оформить отчет: описать все выделенные классы, слоты онтологии, описать запросы и ответы, а также кратко описать функционал выбранного инструментального средства для разработки онтологии.

#### **Разработка онтологии**

Разработка онтологии производилась в инструментальном средстве – Neo4j.

Предметная область – литература

Тема интеллектуальной карты из лабораторной работы №1 – структура приложения (трекера книг).

Онтология включает в себя главные класс “Структура”. Элементами класса являются другие классы – структурные элементы, содержащие контент и обладающие некоторыми функциональными возможностями.

| Структурный элемент | Содержание | |
| --- | --- | --- |
| Контент | Функции |
| Главная страница | новости, рекомендации | поиск |
| Каталог | все произведения | фильтрация, сортировка |
| Сообщества | чаты | переписка |
| Личный кабинет | подписки, подписчики, личная информация | редактирование настроек и личной информации |
| Личная библиотека | добавленные книги | добавление новых категорий |

Некоторые функции и элементы контента имеют свои подразделения.

Новости представляют собой:

* рекомендации алгоритмов приложения
* рекомендации друзей
* информация об обновлении приложения

В Личной библиотеке хранятся:

* Читаемые
* Прочитанные
* В планах
* Брошенные

Сортировка производится по:

* алфавиту
* популярности
* оценкам

Фильтрация производится по:

* тегам
* жанрам
* авторам
* возрастным ограничениям

Теги, хранимые в онтологии: «боевик», «милитари», «герой», «детектив», «для детей», «подростковое», «игра», «история», «киберпанк», «космос», «магия», «автомобили», мистика», «музыка», «соревнования», «бизнес», «сверхъестественное», «спорт», «триллер», «полиция», «дружба», «пьеса», «драматургия», «постапокалипсис», «приключения», «повседневность».

Жанры, хранимые в онтологии: «комедия», «трагедия», «драма», «ужасы», «романтика», «фэнтези», «науч-поп», «научная литература».

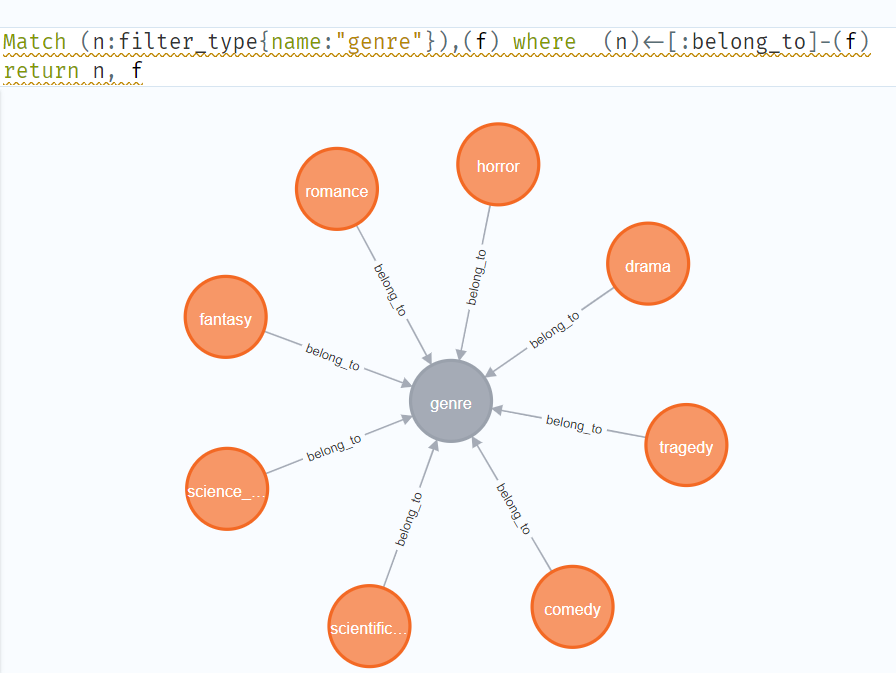
Возрастные ограничения: 0+, 6+, 12+, 16+, 18+.

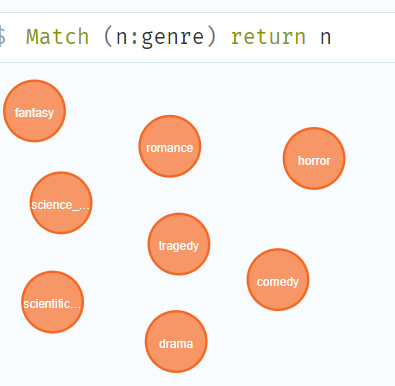
Получаем следующую онтологию:

#### 

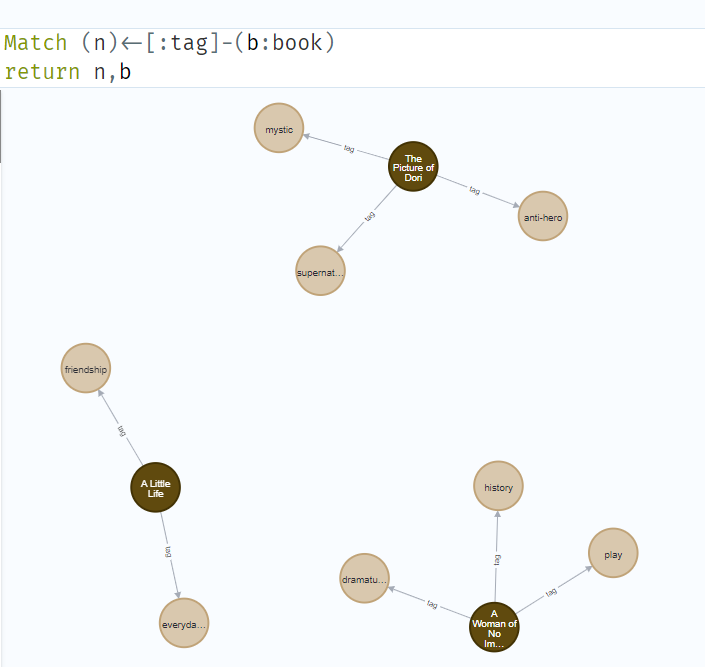
#### **Запросы к собственной онтологии**

1. Какие жанры имеются в системе:





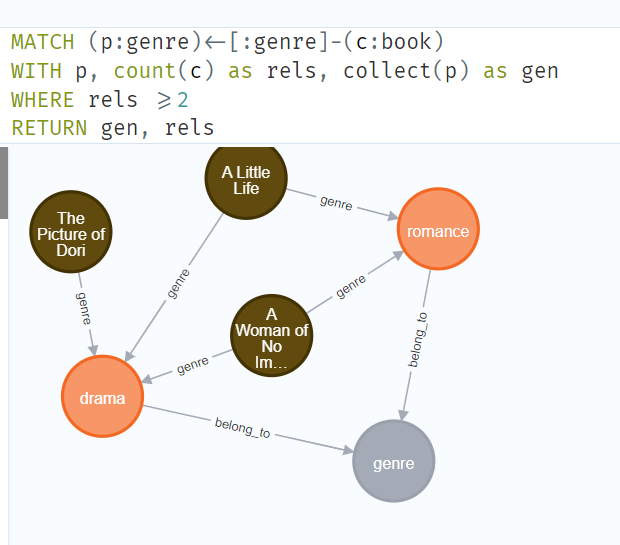
1. Теги, которые присвоены книгам



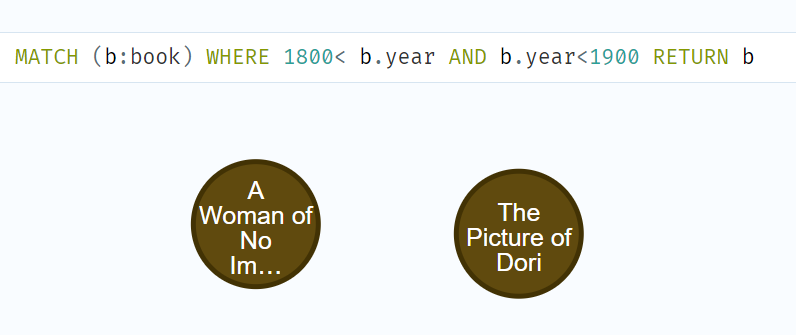
1. Жанры книг с высоким возрастным ограничением

#### 

1. Жанры, в которых не меньше 2 книг

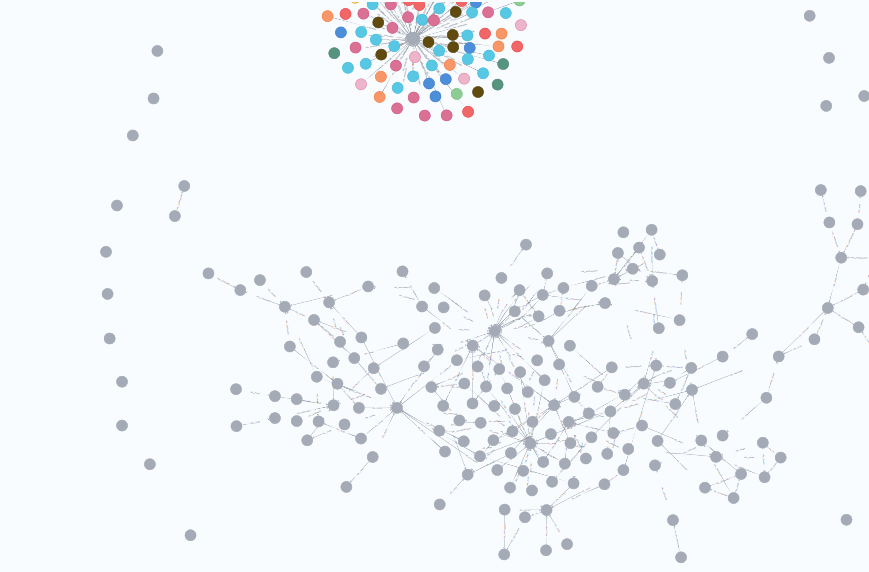


1. Книги 19 века



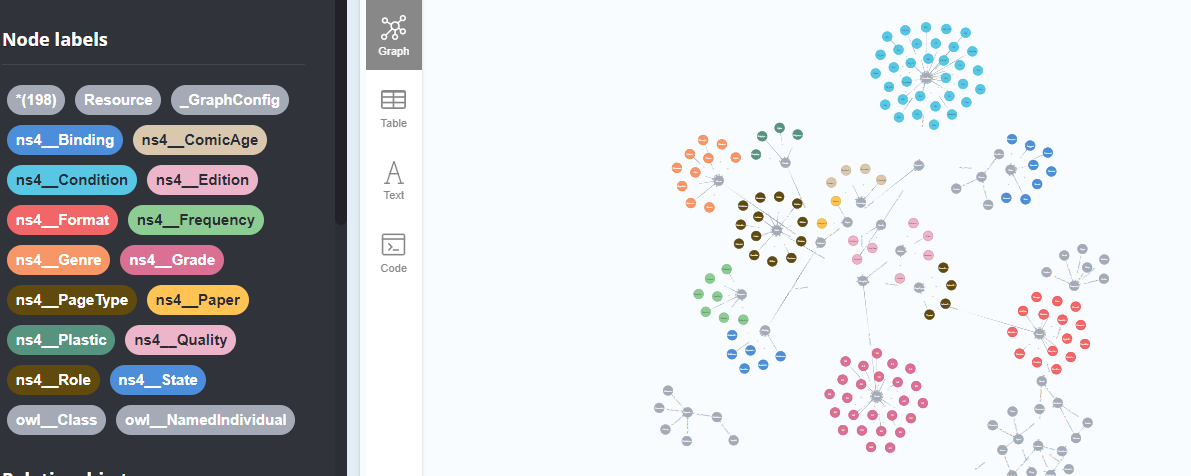
#### **Отображение готовых онтологий в Neo4j**

Для интеграции с разработанной онтологией была выбрана онотология Comic Book Ontology с сайта DBpedia.

Онтология содержит следующие классы и имеет вид следующего графа:



Т.к вершины графа не имеют понятных названий, поменяем их и уберем лишние вершины для удобной интеграции. Получили следующую структуру:



Остались классы описания комиксов, разделенные на 2 основные части: Физические и Нефизические параметры.

К физическим параметрам относятся: Метериал, Состояние, Качество, Форма.

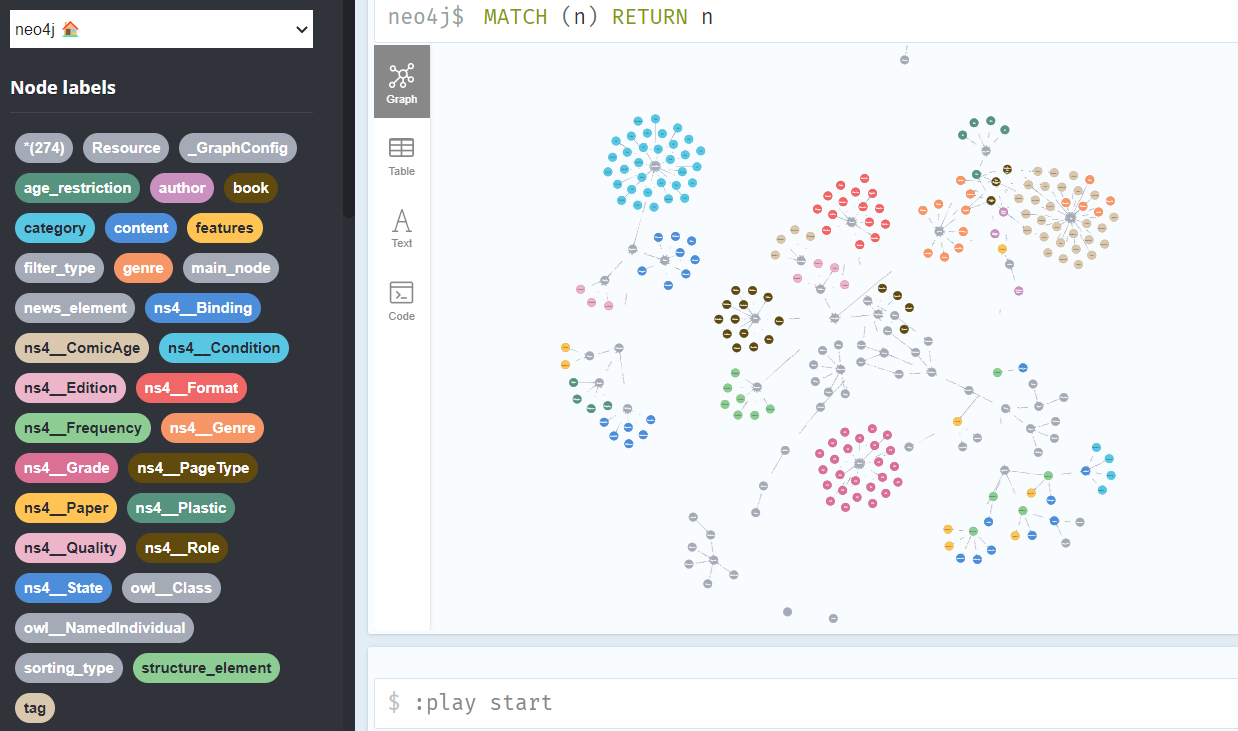
* Экземпляры класса Состояние определяют оценки состояния комикса. Основные типы: Новый, Хорошее, Плохое, Среднее и всевозможные комбинации с приставкой Очень или постфиксами + и
* Класс Материал является надклассом для классов:
  + Пластик, содержащий типы пластика;
  + Бумага с типами бумаги;
  + Переплет, с типом обложки.
* Класс Качество содержит дополнительные качества объекта например антистатический или бескислотный.
* Класс Форма определяет формат издания: Сшитый, В переплете, Необработанный и т.д.

К нефизическим параметрам относятся классы:

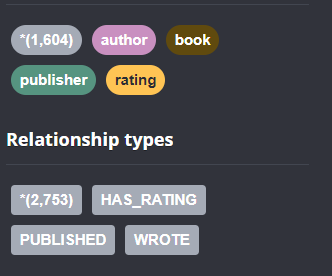
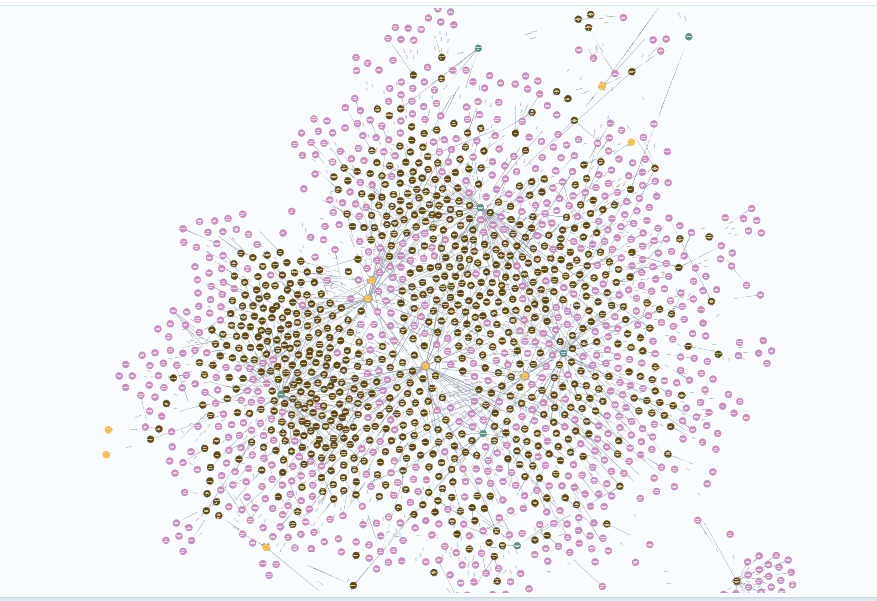
* Эра комикса
* Тип издания: Лимитированное, Поэтапное и т.д.
* Оценки от 0.5 до 10 с шагом 0.5
* Частота выхода: еженедельно, ежемесячно, несколько раз в неделю и т.д.
* Жанры
* Формат: книга, журнал, веб, манга и т.д

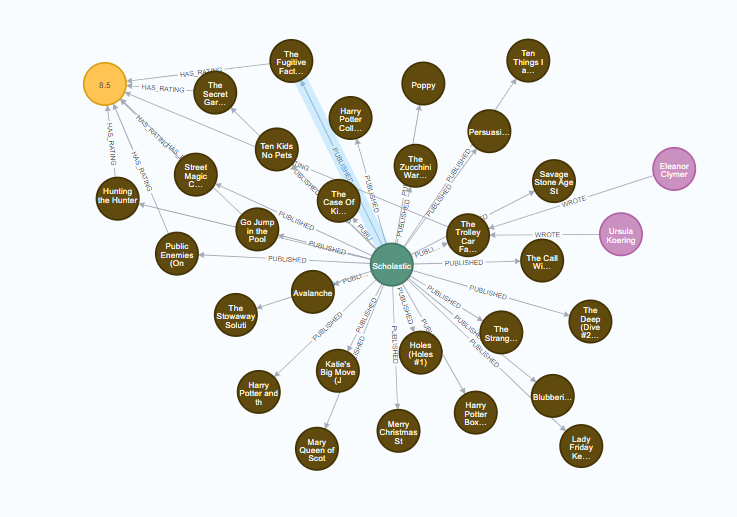
Также остались дополнительные ноды, необходимые только для объяснений некоторых понятий, т.е. содержащие в себе их определения.

После интеграции с самостоятельно разработанной онтологией получаем:

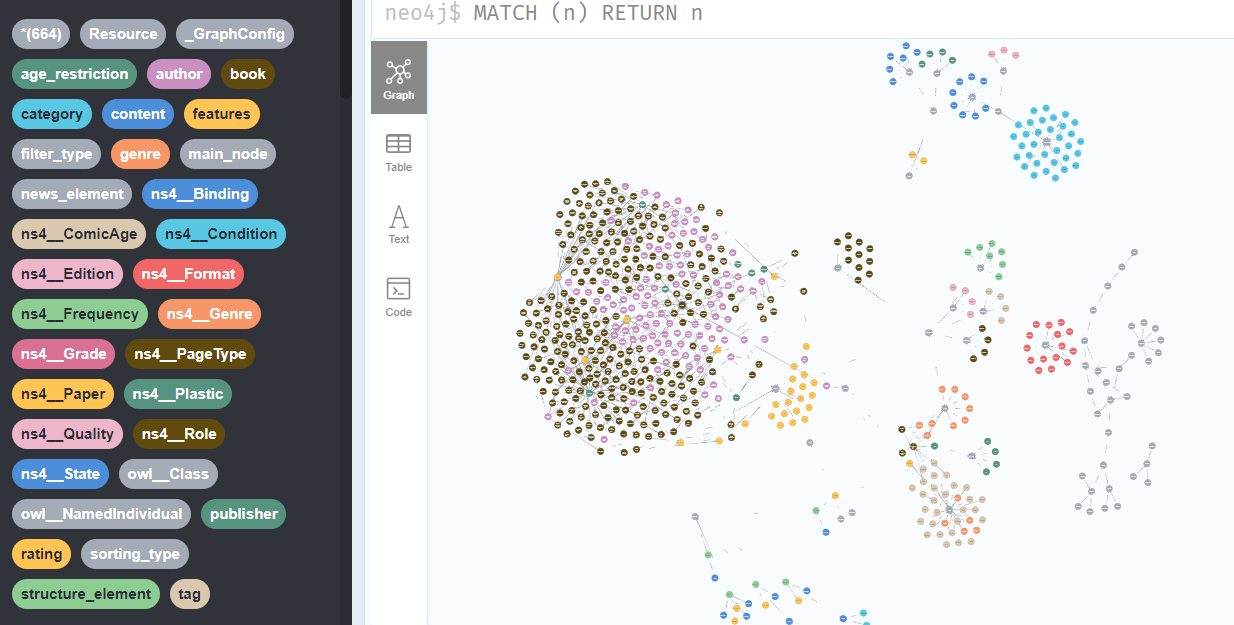


Вторая онтология Goodreads-books содержит более 1.5 тысячи вершин, но только 4 класса: Рейтинг, Книги, Авторы и Издательства.



Пример отношений:

Для интеграции были оставлены лишь авторы, написавшие несколько произведений, чтобы сократить время обработки.Результат интеграции:

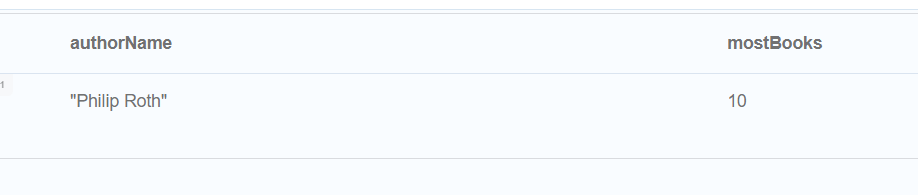


#### **Запросы к итоговой онтологии**

#### Автор написавший больше книг за конец прошлого века:

#### MATCH (a:author)-[:WROTE]->(b:book) WHERE b.year >= 1990 AND b.year <= 2000 WITH a, COUNT(\*) AS bookCount ORDER BY bookCount DESC

#### LIMIT 1 RETURN a.name AS authorName, bookCount AS mostBooks



#### Книга с наихудшим рейтингом:

MATCH (book:book)-[:HAS\_RATING]->(rating:rating)

WITH book, rating.name AS ratingValue

ORDER BY ratingValue ASC

LIMIT 1

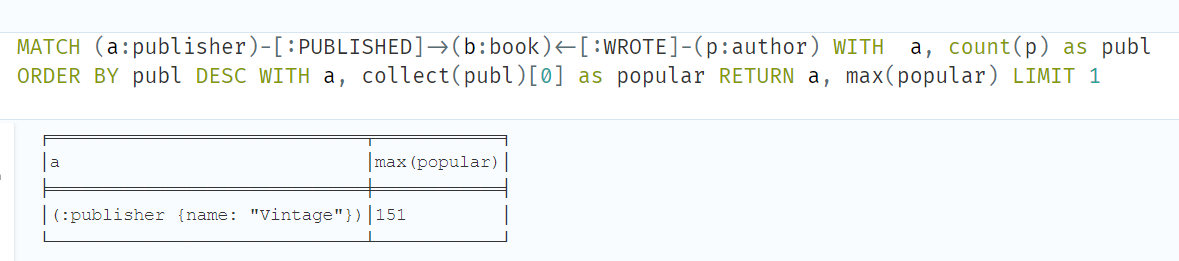
RETURN book.name AS bookName, ratingValue AS worstRating

#### 

#### 3. Самое популярное издательство (где издавалось больше всего авторов:

MATCH (a:publisher)-[:PUBLISHED]->(b:book)<-[:WROTE]-(p:author) WITH a, count(p) as publ

ORDER BY publ DESC WITH a, collect(publ)[0] as popular RETURN a, max(popular)



#### 

#### **Вывод**

Приобрели навыки разработки онтологий предметных областей и интеграции разработанных онтологий. Составили запросы для онтологии Neo4j на языке запросов Cypher.