**Отчет по лабораторной работе №3**

**Введение**

**Цель работы:** разработать и сравнить два подхода к моделированию знаний — онтологический (в редакторе Protégé) и графовый (в базе данных Neo4j) — на примере предметной области «Библиотечный каталог».

Задачи:

1. Создать формальную онтологию в редакторе Protégé, определив классы, свойства и правила логического вывода (аксиомы).
2. Проверить работоспособность онтологии с помощью механизма логического вывода (Reasoner).
3. Адаптировать разработанную онтологическую модель в графовую базу данных Neo4j.
4. Реализовать логику аксиом с помощью языка запросов Cypher и проверить результаты.
5. Провести сравнительный анализ двух подходов, выявив их сильные и слабые стороны, а также области применения.

**Описание моделируемой предметной области**

Предметная область представляет собой библиотечный каталог, содержащий информацию о книгах, их авторах, издательствах и читателях. В системе отражаются связи между объектами: кто написал книгу, кто её издал и кто её взял.

Основные сущности:

* Книга — литературное произведение, которое может быть написано автором, издано издательством и взято читателем.
* Автор — человек, написавший одну или несколько книг.
* Издательство — организация, выпускающая книги.
* Читатель — человек, который может брать книги из библиотеки.

Ключевые правила логического вывода:

* Читатель считается активным, если он взял хотя бы одну книгу.
* Книга считается опубликованной, если у неё есть автор и издательство.

**Реализация в Protégé**

Здесь мы описываем знания в виде фактов и правил, а механизм Reasoner автоматически делает логические выводы, не изменяя исходные данные.

Структура онтологии:

Классы:

* Книга (Book)
* Автор (Author)
* Читатель (Reader)
* Издательство (Publisher)
* АктивныйЧитатель (ActiveReader) - выведенный класс
* ОпубликованнаяКнига (PublishedBook) - выведенный класс

Объектные свойства (отношения):

* написанаАвтором (writtenBy)
* изданаИздательством (publishedBy)
* взятаЧитателем (borrowedBy)

Свойства данных (атрибуты):

* имеетНазвание (hasTitle)
* имеетИмя (hasName)
* номерБилета (hasCardNumber)
* названиеИздательства (hasPublisherName)

Созданные экземпляры (Individuals):

| Класс | Индивиды | Свойства |
| --- | --- | --- |
| Книга | Война и мир, Преступление и наказание, Мастер и Маргарита | имеетНазвание: "Война и мир", "Преступление и наказание", "Мастер и Маргарита" |
| Автор | Лев Толстой, Фёдор Достоевский, Михаил Булгаков | имеетИмя: "Лев Толстой", "Фёдор Достоевский", "Михаил Булгаков" |
| Издательство | Азбука, Эксмо, АСТ | названиеИздательства: "Азбука", "Эксмо", "АСТ" |
| Читатель | Иван Иванов, Михаил Клюкин, Максим Орлов | номерБилета: 101, 102, 103 |

Связи между экземплярами:

* Война и мир → написанаАвтором → Лев Толстой
* Преступление и наказание → написанаАвтором → Фёдор Достоевский
* Мастер и Маргарита → написанаАвтором → Михаил Булгаков
* Война и мир → изданаИздательством → Азбука
* Преступление и наказание → изданаИздательством → Эксмо
* Мастер и Маргарита → изданаИздательством → АСТ
* Война и мир → взятаЧитателем → Иван Иванов
* Преступление и наказание → взятаЧитателем → Михаил Клюкин
* Преступление и наказание → взятаЧитателем → Максим Орлов

**Правила логического вывода (SWRL):**

Правило 1: Book(?b) ^ Reader(?r) ^ borrowedBy(?b, ?r) → ActiveReader(?r)

Правило 2: Book(?b) ^ Author(?a) ^ writtenBy(?b, ?a) ^ Publisher(?p) ^ publishedBy(?b, ?p) → PublishedBook(?b)

Результаты логического вывода:  
После запуска Reasoner (Drools) было выведено 79 аксиом, включая:

* Иван Иванов, Михаил Клюкин, Максим Орлов → АктивныйЧитатель
* Война и мир, Преступление и наказание, Мастер и Маргарита → ОпубликованнаяКнига

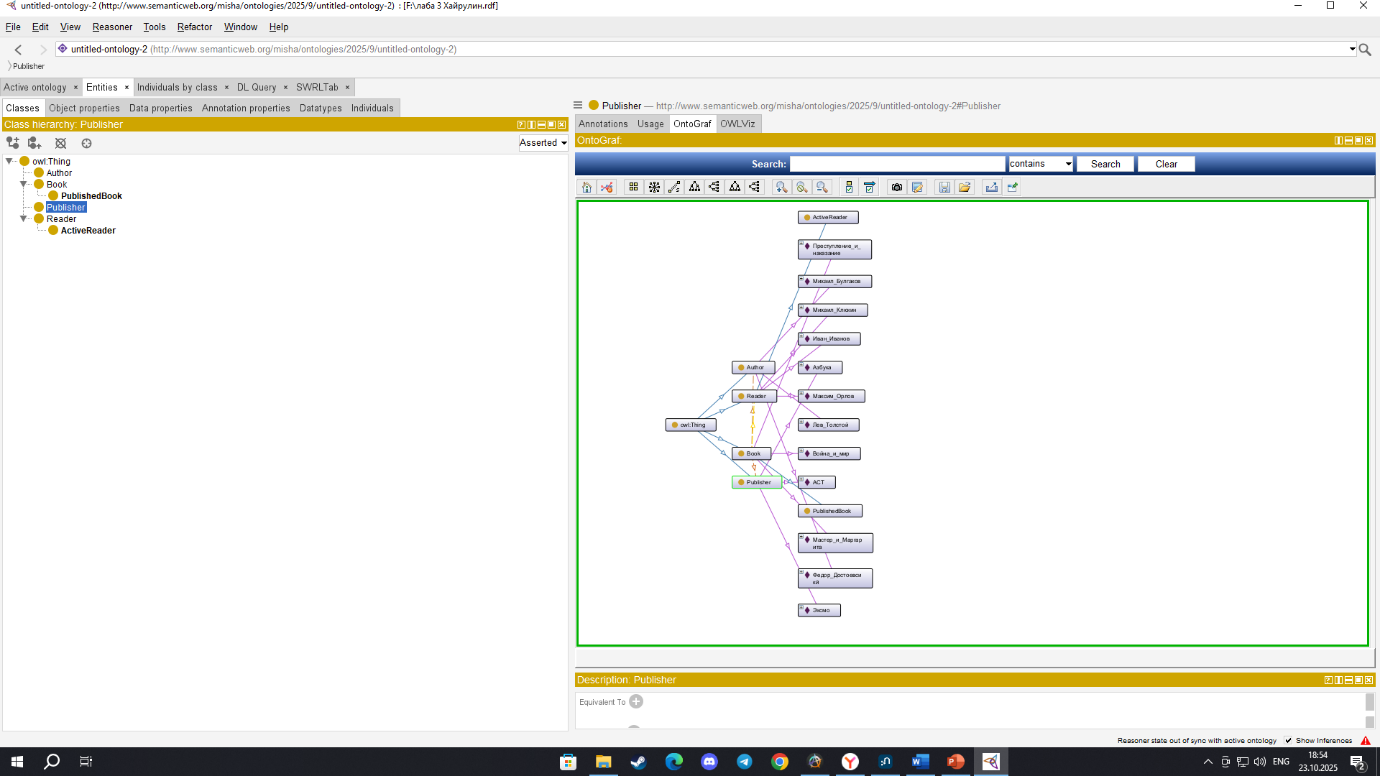


Рисунок 1 – Классы и OntoGraf

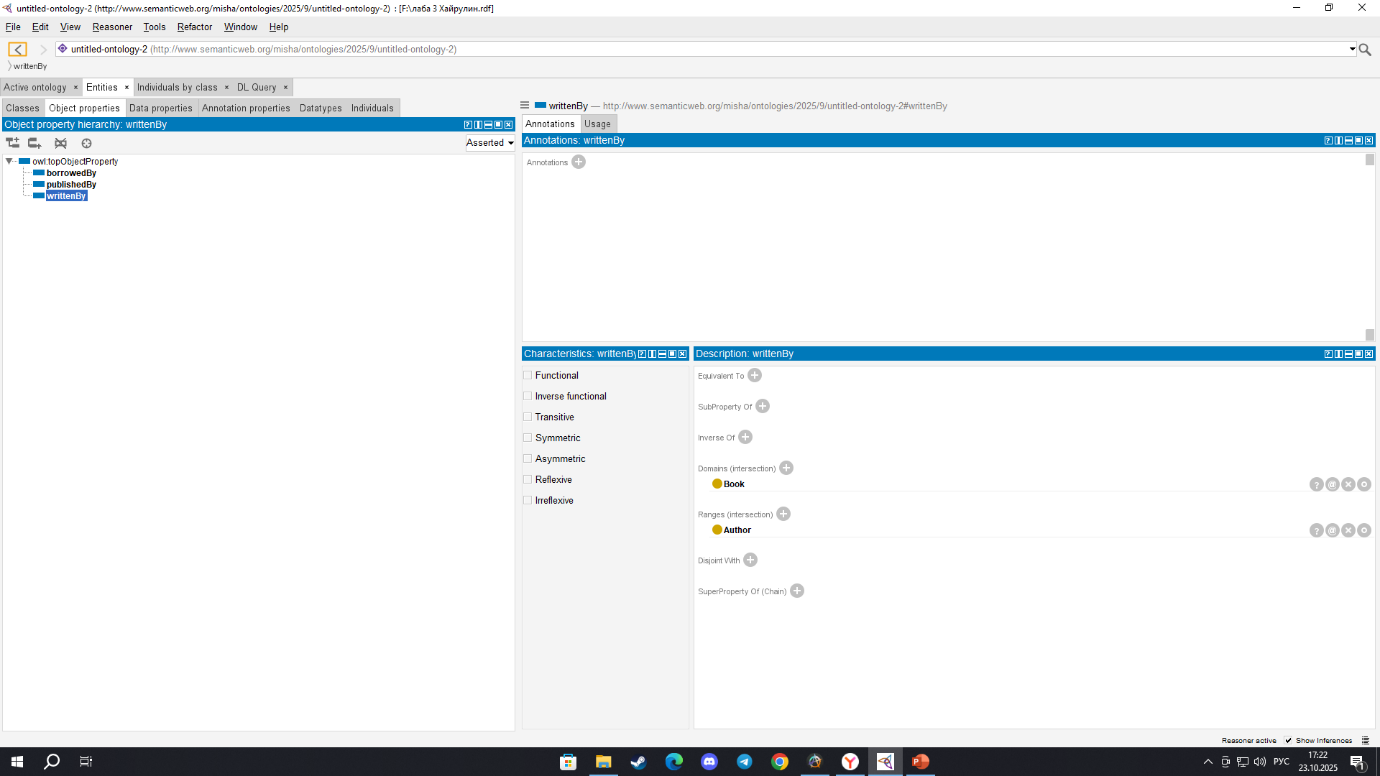


Рисунок 2 – Объектные свойства (отношения)

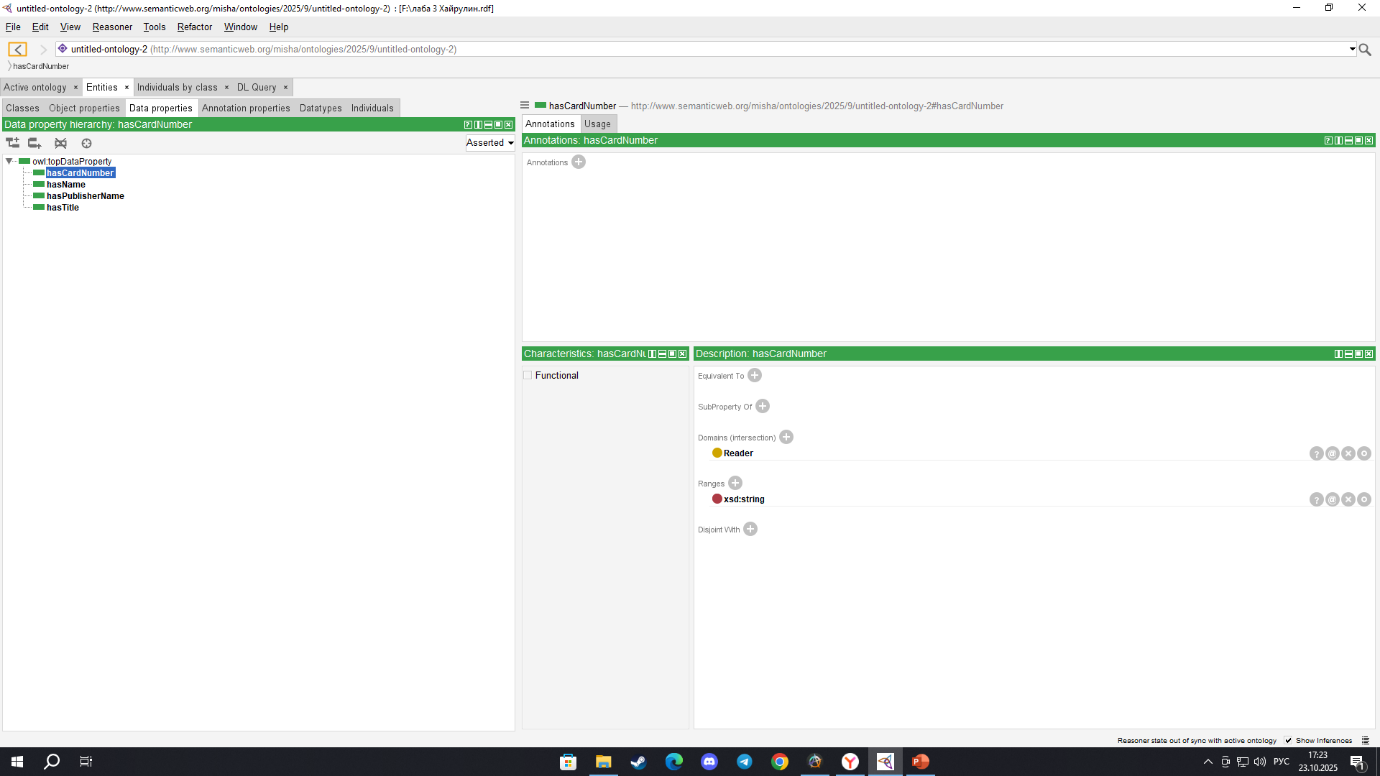


Рисунок 3 – Свойства данных (атрибуты)

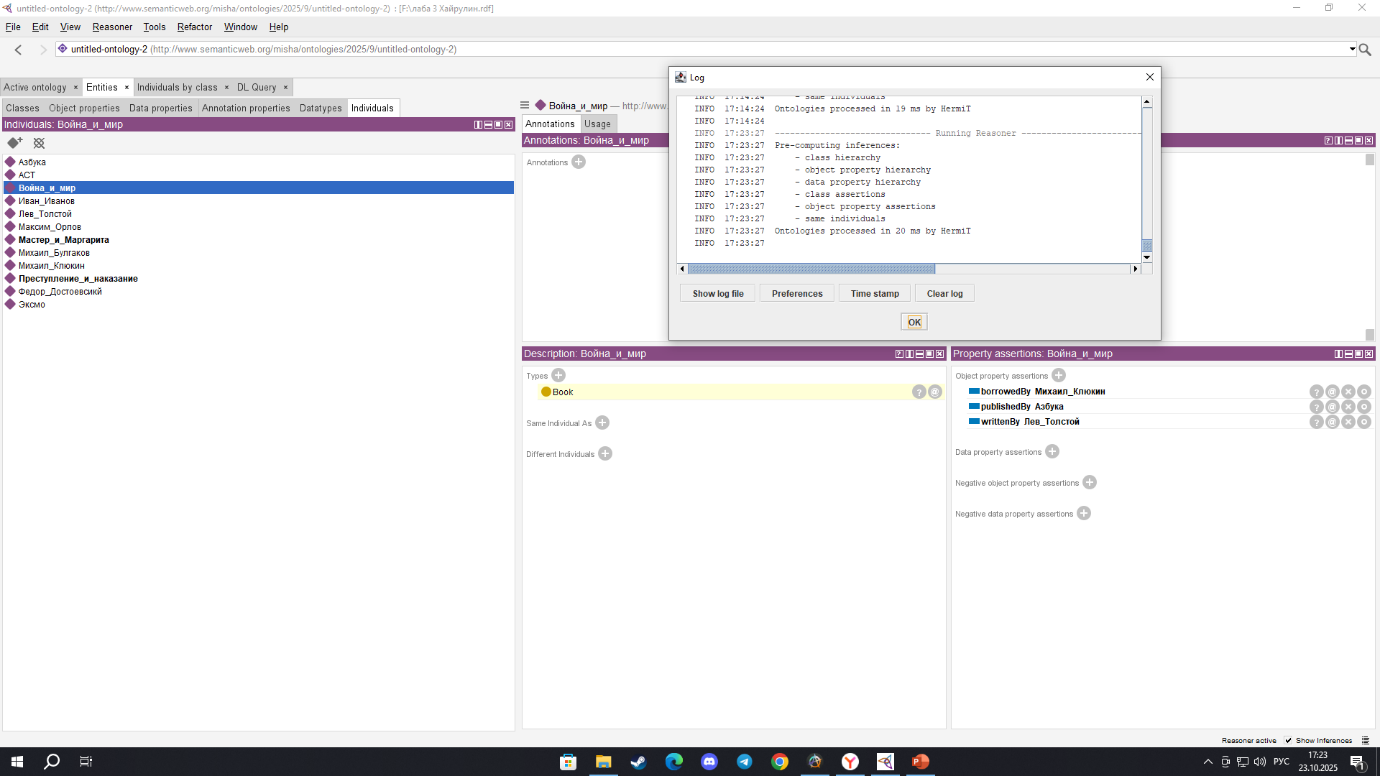


Рисунок 4 – Созданные экземпляры (Individuals) и запуск Reasoner

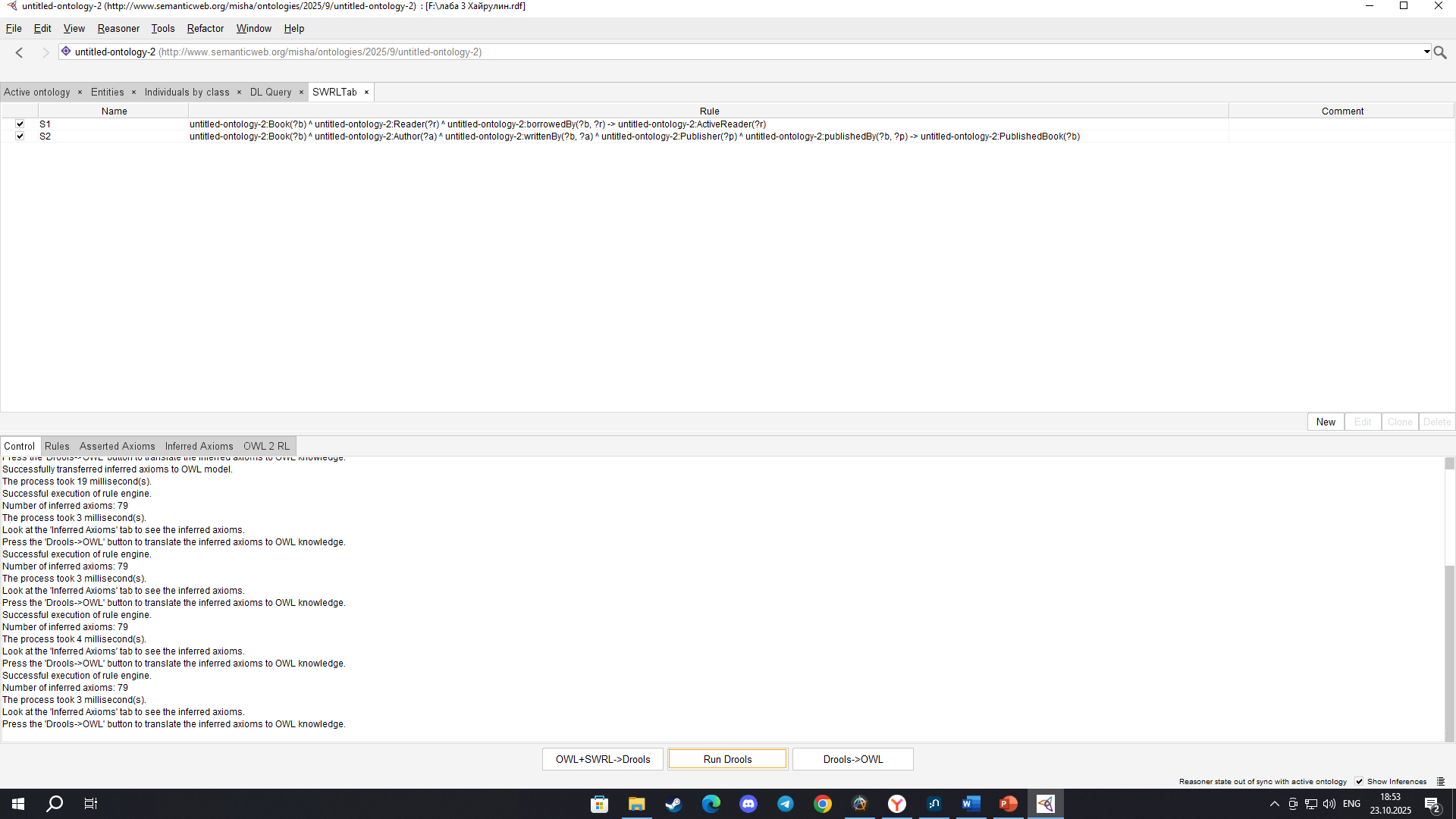


Рисунок 5 – Правила логического вывода (SWRL)

**Реализация в Neo4j**

Подход к моделированию: В Neo4j используется модель маркированного графа свойств (Labeled Property Graph) и императивный подход. Мы не просто описываем правила, а даем прямые команды: «Найди вот такой шаблон в данных и измени его вот так». Логический вывод — это не фоновый процесс, а явный запрос на модификацию данных.

Структура графа:

Метки узлов (Классы):

* :Читатель, :АктивныйЧитатель, :Книга, :ОпубликованнаяКнига, :Автор, :Издательство

Отношения (Связи):

* [:ВЗЯЛ\_КНИГУ], [:НАПИСАНА\_АВТОРОМ], [:ИЗДАНА\_ИЗДАТЕЛЬСТВОМ]

Свойства узлов (Атрибуты):

* имя, название, номерБилета, названиеИздательства

Реализация логического вывода (Аксиомы):  
Логика была реализована с помощью запросов Cypher, которые физически изменяют (материализуют) граф.

Правило 1 (Активный читатель):

cypher

MATCH (r:Читатель)-[:ВЗЯЛ\_КНИГУ]->(:Книга)

SET r:АктивныйЧитатель

Правило 2 (Опубликованная книга):

cypher

MATCH (k:Книга)-[:НАПИСАНА\_АВТОРОМ]->(:Автор),

(k)-[:ИЗДАНА\_ИЗДАТЕЛЬСТВОМ]->(:Издательство)

SET k:ОпубликованнаяКнига

Результат: после выполнения этих запросов граф был изменен. Проверочные запросы показали, что узлы читателей получили метку: АктивныйЧитатель, а узлы книг — метку: ОпубликованнаяКнига.

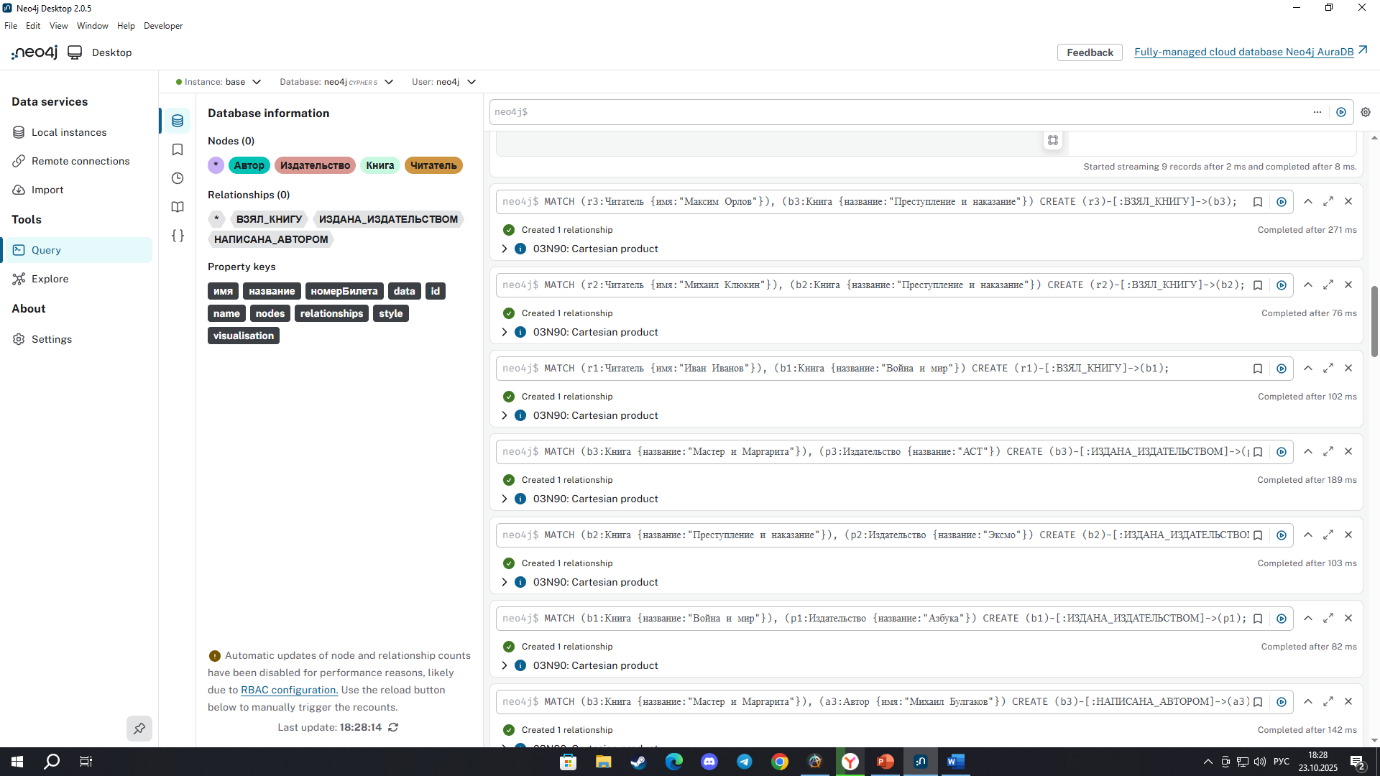


Рисунок 6 – Заполнение базы данных

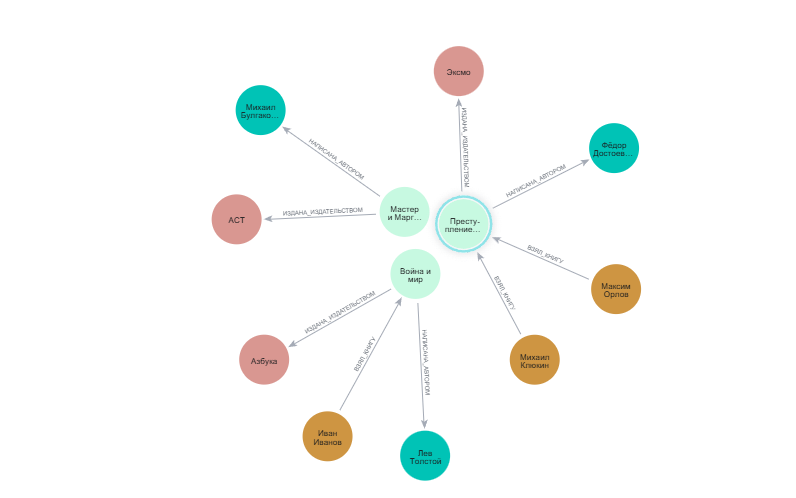
****

Рисунок 7 – Визуализация графовой модели библиотечного каталога

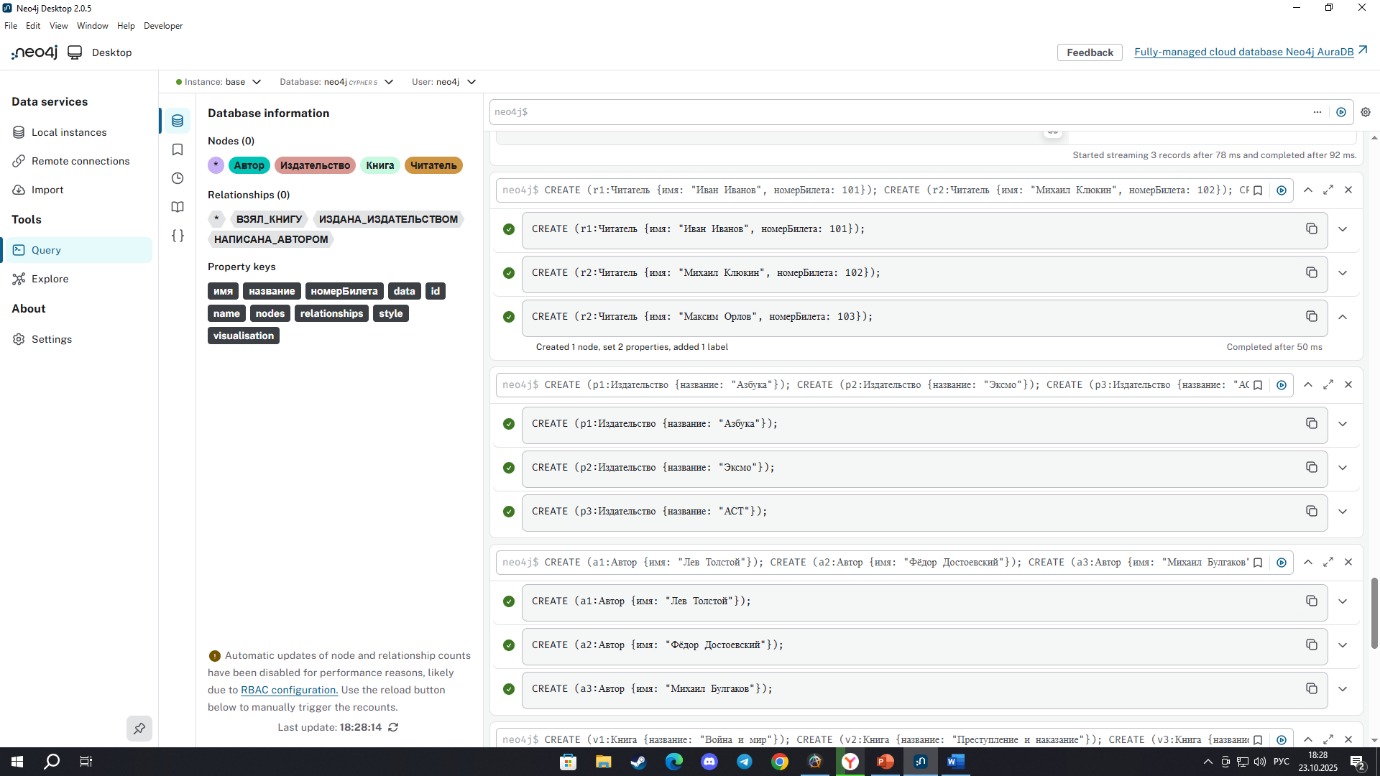


Рисунок 8 – Реализация правил и вывод результата

Сравнительный анализ

| Критерий | Protégé | Neo4j |
| --- | --- | --- |
| Подход | Декларативный (описание "что") | Императивный (описание "как") |
| Логический вывод | Автоматический, виртуальный | Ручной, материализованный |
| Изменение данных | Исходные данные не изменяются | Данные физически изменяются |
| Гибкость | Высокая для сложных логических систем | Высокая для операционных запросов |
| Производительность | Медленнее при сложных выводах | Быстрее для поиска паттернов |
| Область применения | Семантические веб-системы, экспертные системы | Операционные системы, рекомендации |

**Заключение**

В ходе работы было установлено, что Protégé и Neo4j, хотя и могут решать схожие задачи, являются инструментами для разных этапов жизненного цикла модели знаний.

Protégé — это про «что есть истина». Он создает «чертеж» знаний и проверяет, что из этого следует по логике. Подход основан на формальной логике и автоматическом выводе следствий.

Neo4j — это про «что нужно сделать». Он хранит реальные данные и изменяет их по вашим командам. Подход ориентирован на эффективное хранение и манипуляцию графовыми структурами.

Итог:

| Критерии | Protégé | Neo4j |
| --- | --- | --- |
| Что вы делаете? | Описываете правила мира | Даете пошаговые команды |
| Как он думает? | Делает логические выводы "в уме" | Выполняет ваши приказы |
| Что получается? | Виртуальное знание. Данные не меняются | Физическое изменение. Данные меняются навсегда |

Оба инструмента имеют свои преимущества и должны выбираться в зависимости от конкретных требований проекта: Protégé для сложных онтологий и семантического анализа, Neo4j для высокопроизводительных операций с графовыми данными.