Основы теории систем

**Лабораторная работа №3**

**«Формирование онтологической модели с помощью инструментов онтологического инжиниринга Protégé и Neo4j»**

**Вариант 21**

Выполнил студент

группы ИВТИИбд-12

Алиуллов И.Т.

Ульяновск, 2025

**1. Описание моделируемой ситуации (Рабочий Процесс в Ресторане):**

Есть классы: Покупатель: Петя (200), Вася (150); Колбаса: Cервелат (100), Пепперони (180); Хлеб: Чёрный (30), Белый (170). В скобках у колбас и хлеба – цена, у покупателей – бюджет.

Аксиомы: Если покупатель может купить товар, то есть его бюджет превышает цену товара, то образуется связь “Может купить”, иначе образуется связь “Не может купить”.

**2. Особенности реализации в Protege:**

Сначала создаются классы, их свойства, а потом уже представители этих классов. Правила логического вывода пишутся на языке SWRL. Связи отображаются в таблице для каждого из покупателей.











**3. 3. Особенности реализации в Neo4j:**

Класс и свойства привязаны к конкретным представителям класса и носителям свойств и появляются только после добавления представителей. Запросы пишутся на языке Cypher. Связи и классы отображаются в виде графов.

create(n:Покупатель {budget: 150, name: "Вася"})

create(n:Покупатель {budget: 200, name: "Коля"})

create(n:Колбаса {price: 160, name: "Пепперони"})

create(n:Колбаса {price: 100, name: "Сервелат"})

create(n:Хлеб {price: 30, name: "Чёрный"})

create(n:Хлеб {price: 170, name: "Белый"})

match(x:Покупатель), (y:Колбаса)

where x.budget >= y.price

create (x)-[r:Может\_купить]->(y)

match(x:Покупатель), (y:Хлеб)

where x.budget >= y.price

create (x)-[r:Может\_купить]->(y)

match(x:Покупатель), (y:Колбаса)

where x.budget < y.price

create (x)-[r:Не\_может\_купить]->(y)

match(x:Покупатель), (y:Хлеб)

where x.budget < y.price

create (x)-[r:Не\_может\_купить]->(y)

1

4. **Сравнительный анализ инструментов**:

В Protégé все классы, связи, свойства, объекты расположены в соответствующих вкладках, в отличии от Neo4j, где чтобы добавить связи или класс, нужно прописывать соответствующую конструкцию из кода на языке Cypher. Но в Neo4j не нужно отдельно создавать классы и свойства, они создаются вместе с конкретными объектами.