Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ульяновский государственный технический университет»

Кафедра: «Вычислительная техника»

Дисциплина: «Системы искусственного интеллекта»

Лабораторная работа №3

«Базы знаний и онтологии»

Выполнила

Студентка группы ИВТАСбд-41

Микка Е. И.

Ульяновск, 2024

# Общее задание на лабораторную работу

В данной лабораторной работе необходимо:

1. Ознакомиться с особенностями выбранного объекта проектирования, включая его функциональные требования, ограничения и задачи автоматизации.
2. Разработать базу данных знаний (создание онтологии и формулировка правил управления объектом).
3. Проектирование логической модели управления.
4. Создание симуляции, проверяющей выполнение правил, на разных сценариях.

# Введение

Выбранная предметная область – это автоматизация управления освещением на основе времени суток, присутствия людей и уровня естественного освещения.

Автоматизация управления освещением — это система, которая позволяет оптимизировать использование освещения в помещениях на основе различных факторов, таких как время суток, присутствие людей и уровень естественного освещения.

Задача автоматизации – на основе вышеуказанных факторов необходимо определить наиболее подходящий уровень освещения.

**Модель базы знаний и правила управления**

В реляционной базе данных были созданы четыре таблицы, отвечающие на возможные варианты факторов время суток, присутствие людей и уровень естественного освещения. Последняя четвертая таблица отвечает за формулировку правил, она содержит четыре колонки. Три посвящены факторам, возможные варианты для факторов определены в ранее описанных таблицах, четвертая отвечает за уровень искусственного освещения, различные комбинации дают различные значения уровня искусственного освещения.

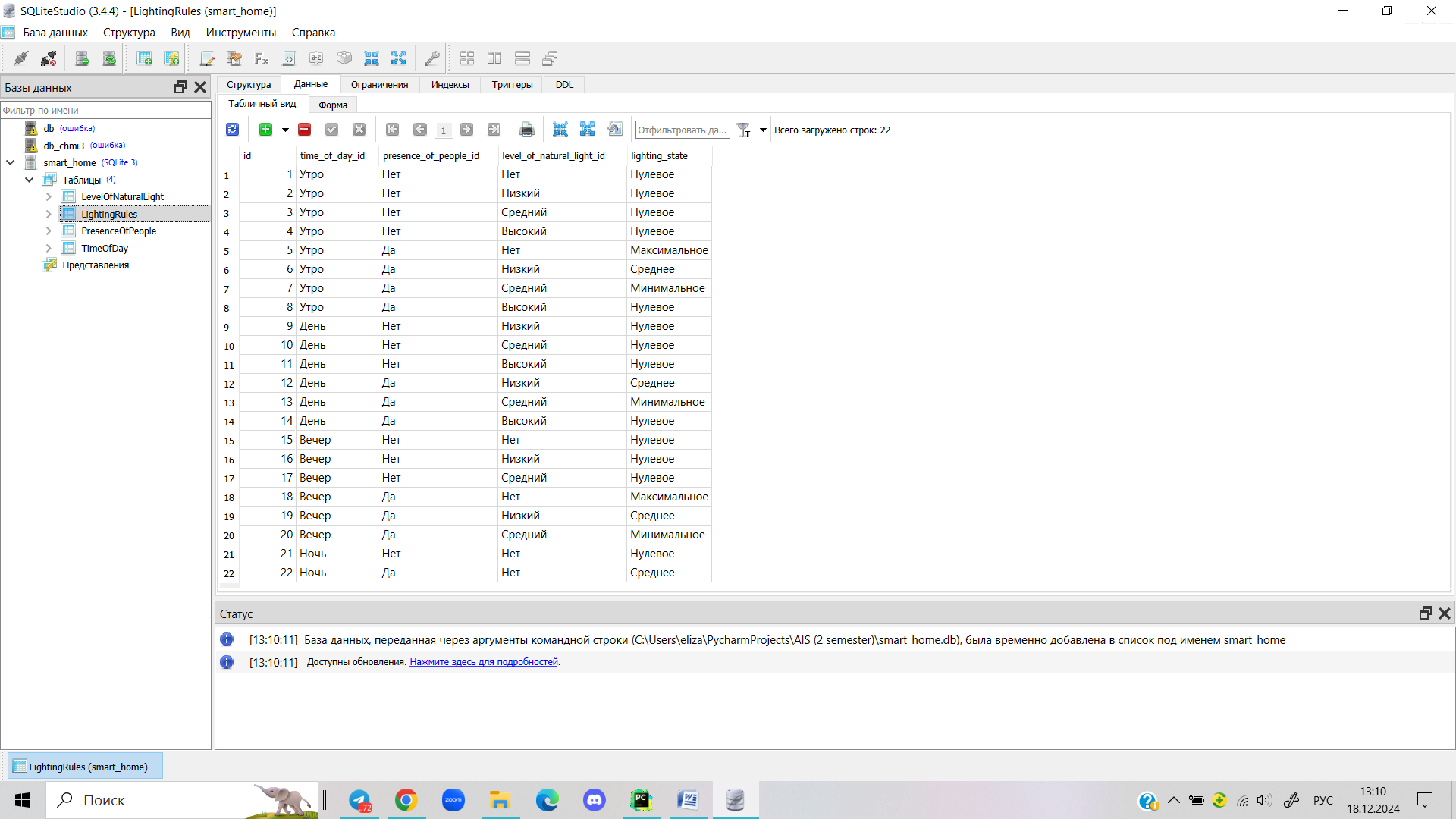


Рис 1. Формулировка правил

**Логическая модель управления**

Функция fuzzify\_conditions (фаззификация условий) принимает три параметра: hour (время суток), n\_people (количество людей) и natural\_light (уровень естественного освещения), определяет время суток (утро, день, вечер, ночь) на основе значения hour; устанавливает наличие людей: "Да", если n\_people не равно нулю, и "Нет" в противном случае; классифицирует уровень естественного освещения на "Нет", "Низкий", "Средний" или "Высокий" в зависимости от значения natural\_light и возвращает три значения: time\_of\_day, presence\_of\_people, level\_of\_natural\_light.

Листинг 1. Функция фаззификации условий

def fuzzify\_conditions(hour, n\_people, natural\_light):  
 # Определение времени суток  
 time\_of\_day = ''  
 level\_of\_natural\_light = ''  
 if 6 <= hour < 12:  
 time\_of\_day = 'Утро'  
 elif 12 <= hour < 17:  
 time\_of\_day = 'День'  
 elif 17 <= hour < 22:  
 time\_of\_day = 'Вечер'  
 else:  
 time\_of\_day = 'Ночь'  
 # Присутствие людей  
 presence\_of\_people = 'Да' if n\_people != 0 else 'Нет'  
 # Уровень естественного освещения  
 if natural\_light <= 10:  
 level\_of\_natural\_light = 'Нет'  
 elif 10 < natural\_light <= 30:  
 level\_of\_natural\_light = 'Низкий'  
 elif 30 < natural\_light <= 70:  
 level\_of\_natural\_light = 'Средний'  
 elif 70 < natural\_light <= 100:  
 level\_of\_natural\_light = 'Высокий'  
  
 return time\_of\_day, presence\_of\_people, level\_of\_natural\_light

Функция get\_lighting\_action принимает три параметра: time\_of\_day, presence\_condition и natural\_light\_level\_condition, подключается к базе данных smart\_home.db и выполняет SQL-запрос, чтобы получить состояние освещения (lighting\_state) на основе заданных условий. Функция ищет правило в БД, которое будет удовлетворять заданным параметрам.

Листинг 2. Функция, получающая действия по правилам

def get\_lighting\_action(time\_of\_day, presence\_condition, natural\_light\_level\_condition):  
 conn = sqlite3.connect('smart\_home.db')  
 cursor = conn.cursor()  
 query = '''  
 SELECT lighting\_state FROM LightingRules   
 WHERE time\_of\_day\_id = ? AND presence\_of\_people\_id = ? AND level\_of\_natural\_light\_id = ?  
 '''  
 cursor.execute(query, (time\_of\_day, presence\_condition, natural\_light\_level\_condition))  
 result = cursor.fetchone()  
 conn.close()  
  
 # Проверка возращаемого значения

if result is not None:  
 return result[0]  
 else:  
 return None

**Создание симуляции**

Данный код реализует симуляцию управления освещением в умном доме на основе случайно сгенерированных условий, таких как время суток, количество людей и уровень естественного освещения. Случайно генерируются следующие условия: случайное время суток (0-23), количество людей (0-10) и уровень естественного освещения, зависящий от времени. Определяет категории условий (время суток, наличие людей, уровень освещения) с помощью функции fuzzify\_conditions. Запрашивает действие по управлению освещением через функцию get\_lighting\_action на основе определённых категорий. Сообщает о действии управления освещением или отсутствии данных.

Листинг 3. Симуляция управления освещением

def simulate\_lighting():  
  
 hour = random.randint(0, 23) # Случайный час  
 n\_people = random.randint(0, 10) # Количество человек  
 natural\_light = 0  
  
 if 6 <= hour < 12: # Утреннее естественное освещение  
 natural\_light = random.randint(0, 90)  
 elif 12 <= hour < 17: # Дневное естественное освещение  
 natural\_light = random.randint(20, 100)  
 elif 17 <= hour < 22: # Вечернее естественное освещение  
 natural\_light = random.randint(0, 70)  
 else: # Ночное естественное освещение  
 natural\_light = random.randint(0, 10)  
  
 print(f'Время - {hour}\nКоличество людей - {n\_people}\nУровень естественного освещения - {natural\_light}')  
  
 # Фаззификация  
 time\_of\_day, presence\_condition, natural\_light\_level\_condition = fuzzify\_conditions(hour, n\_people, natural\_light)  
  
 lighting\_action = get\_lighting\_action(time\_of\_day, presence\_condition, natural\_light\_level\_condition)  
  
 if lighting\_action is not None:  
 print(f"Действие по управлению освещением: {lighting\_action} освещение.")  
 else:  
 print("Нет данных для управления освещением.")

**Тестирование**

Симуляция 10 сценариев

