Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА ИУ7 «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»

КУРСОВАЯ РАБОТА

НА ТЕМУ:

Разработка базы данных для системы мониторинга оборудования на тракторном заводе

Студент	**ИУ7-66Б**		**Александрова**
	(группа)	(подпись, дата)	(И.О. Фамилия)
Руководитель курсового проекта			**Дроздецкая П.В.**
		(подпись, дата)	(И.О. Фамилия)
Консультант			
		(подпись, дата)	(И.О. Фамилия)

2025 г.

РЕФЕРАТ

Расчетно-пояснительная записка 42 с., 7 рис., 2 табл., 10 источн., 1 прил.

В данной курсовой работе была разработана база данных для системы мониторинга оборудования на тракторном заводе. Рассмотрены основные этапы проектирования базы данных, включая анализ предметной области, определение ключевых сущностей и их взаимосвязей, разработку структуры таблиц с ограничениями целостности и ролевой модели доступа. Реализован программный интерфейс для взаимодействия с данными, проведено тестирование корректности работы базы данных и триггеров, а также исследование зависимости времени выполнения запросов от количества записей в базе данных. Полученные результаты подтверждают эффективность и надежность разработанной системы для мониторинга оборудования и анализа его состояния.

Ключевые слова: мониторинг оборудования, база данных, проектирование базы данных, реляционная модель, тестирование, производительность запросов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современные производственные предприятия, такие как тракторные заводы, зачастую сталкиваются с растущими требованиями к эффективности и надежности оборудования. На современных тракторных заводах важно следить за состоянием оборудования, чтобы обеспечить его эффективную работу и избежать поломок. Регулярный мониторинг позволяет заранее выявлять проблемы и снижать затраты на ремонт. В условиях жесткой конкуренции заводы стремятся повысить свою эффективность, и именно здесь мониторинг техники играет ключевую роль.

Целью данной курсовой работы является разработка базы данных для системы мониторинга оборудования на тракторном заводе, которая позволит централизованно собирать, хранить и анализировать данные о состоянии техники.

Для достижения данной цели были поставлены следующие задачи.

1. Провести анализ предметной области с целью выявления основных объектов и процессов, связанных с мониторингом оборудования на тракторном заводе.

2. Определить ключевые сущности базы данных и их взаимосвязи, необходимые для эффективного представления данных в рамках предметной области.

3. Сформулировать требования и ограничения, предъявляемые к проектируемой базе данных, включая функциональные и нефункциональные аспекты.

4. Спроектировать структуру базы данных, включая описание сущностей, атрибутов, связей между сущностями и ограничений целостности данных.

5. Разработать ролевую модель безопасности на уровне базы данных для разграничения прав доступа к данным и обеспечения их защиты.

6. Выбрать подходящие средства реализации базы данных и приложения, исходя из специфики задачи и доступных технологий.

7. Реализовать спроектированную базу данных, включая создание таблиц, ограничений и других объектов базы данных.

8. Разработать методы тестирования и реализовать тесты, направленные на проверку корректности функционирования базы данных и приложения.

9. Провести исследование производительности, изучив зависимость времени выполнения запросов от объема обрабатываемых данных, и сделать выводы по результатам анализа.

1 Аналитический раздел

В данном разделе проводится анализ предметной области и сравнительный анализ существующих на рынке аналогов, формализация требований к базе данных и информации, которая будет в ней храниться. Опираясь на произведенную формализацию задачи и данных, была построена диаграмма прецедентов с учетом ролевой модели и диаграмма сущность-связь.

1.1 Анализ предметной области

Предметная область охватывает процессы эксплуатации, технического обслуживания и мониторинга оборудования, задействованного в производственном цикле тракторного завода. Особое внимание уделяется учету технического состояния оборудования, своевременному выявлению неисправностей и предотвращению сбоев в производстве. Мониторинг осуществляется как вручную, так и автоматически, с помощью установленных датчиков.

1.2 Сравнительный анализ аналогов и проектируемого приложения

Перед началом разработки информационной системы мониторинга оборудования необходимо рассмотреть существующие решения, используемые в промышленности. Это позволяет выявить сильные и слабые стороны аналогов, определить ключевые требования к собственной системе и обосновать необходимость её разработки.

Для сравнительного анализа выбраны решения, охватывающие различные подходы к мониторингу оборудования:

1. 1С:ТОиР [1] — представляет собой учетно-аналитическую систему, ориентированную на документооборот, планирование технического обслуживания и ремонта оборудования;

2. Siemens MindSphere [2] — мощная облачная IoT-платформа, предназначенная для сбора, хранения и анализа данных с производственного оборудования в реальном времени;

3. АСКОН–Вертикаль [3] — отечественное решение для управления производственными активами, ориентированное на технический учёт и контроль

состояния оборудования.

Для сравнения были выделены следующие критерии.

— Мониторинг в реальном времени — способность системы отслеживать технические параметры оборудования (температура, вибрация, давление и др.) с минимальными задержками, что критически важно для предупреждения аварийных ситуаций и своевременного обслуживания.

— Гибкость проектирования базы данных — возможность адаптировать структуру БД под конкретные особенности производства, включая добавление новых сущностей, изменение связей, формирование индивидуальных отчётов.

— Учёт персонала и технического обслуживания — поддержка механизмов для ведения журнала ремонтов, регистрации ответственных работников, планирования и контроля выполнения операций по техобслуживанию.

— Стоимость внедрения — совокупные затраты на лицензирование, настройку, обучение персонала и сопровождение системы. Для небольших предприятий этот критерий может стать определяющим при выборе решения.

Сравнение представленных систем по указанным критериям представлено в таблице 1.1.

Таблица 1.1 – Сравнение аналогичных решений

Критерий	**Собственная система**	**1С:ТОиР**	**Siemens MindSphere**	**АСКОН – Вертикаль**
Мониторинг в реальном времени	+	—	+	—
Гибкость проектирования БД	+	—	—	±
Учёт персонала и ТО	+	+	±	+
Стоимость внедрения	Низкая	Высокая	Очень высокая	Средняя

Анализ существующих решений показал, что промышленные платформы, такие как Siemens MindSphere, ориентированы на крупные предприятия и требуют значительных финансовых и технических ресурсов для внедрения. Несмотря на широкие возможности мониторинга, такие системы сложно адаптировать под конкретные нужды небольшого производства. Отечественные решения, такие как 1С:ТОиР и АСКОН–Вертикаль, в первую очередь направлены на документооборот и планирование технического обслуживания, однако не всегда обеспечивают гибкость в проектировании базы данных и не поддерживают мониторинг в реальном времени. В этих условиях разработка собственной базы данных позволяет учесть особенности конкретного производственного процесса, реализовать необходимые функции мониторинга и учёта с минимальными затратами, а также обеспечить полную кастомизацию структуры данных и прав доступа.

1.3 Анализ моделей данных

В процессе проектирования системы мониторинга оборудования необходимо определить наиболее подходящую модель базы данных, которая будет соответствовать сформулированным целям, задачам и структуре предметной области. Будут рассмотрены три основных типа моделей: дореляционные, реляционные и постреляционные.

1.3.1 Дореляционные модели баз данных

Дореляционные модели включают иерархическую и сетевую модели.

Иерархические базы данных [4] — самая ранняя модель представления сложной структуры данных. Информация в иерархической базе организована по принципу древовидной структуры, в виде отношений "предок-потомок".

Сетевая модель данных [5] определяется в тех же терминах, что и иерархическая. Она состоит из множества записей, которые могут быть владельцами или членами групповых отношений. Связь между записью-владельцем и записью-членом также имеет вид 1:N. Основное различие этих моделей состоит в том, что в сетевой модели запись может быть членом более чем одного группового отношения.

Эти подходы строятся на жёстко заданных связях между сущностями. Данные хранятся в виде древовидных или графовых структур, что затрудняет реализацию сложных взаимосвязей между сущностями.

1.3.2 Реляционные модели баз данных

Реляционные системы [6] баз данных основаны на формальной теории, которая называется реляционной моделью данных. В рамках этой модели выполняются три основных условия.

1. Данные представляются пользователю исключительно в виде таблиц, что составляет структурную основу модели.

2. Данные таблицы подчиняются определённым условиям целостности, которые гарантируют корректность и непротиворечивость информации.

3. Пользователь может работать с таблицами с помощью набора операций, позволяющих находить, фильтровать и объединять данные. При этом результатами данных операций будут также таблицы.

1.3.3 Постреляционные модели баз данных

Постреляционные модели [7] (расширенные реляционные или объектно-реляционные модели) развивают принципы реляционной модели, добавляя поддержку хранения сложных структур данных — вложенных объектов, JSON-документов, массивов и т.д. Они представляют собой логическое развитие классической реляционной парадигмы, направленное на устранение её ограничений при работе со сложноструктурированной информацией. В отличие от традиционных реляционных систем, требующих строгого соблюдения нормальных форм, постреляционный подход допускает контролируемое нарушение 1NF, что обеспечивает возможность хранения составных и иерархических данных непосредственно в рамках табличной модели.

1.3.4 Вывод

С учётом структуры предметной области, необходимости строгой формализации сущностей, чётких связей между ними, а также поддержки разграничения прав доступа и мониторинга в реальном времени, оптимальным выбором является реляционная модель базы данных в связи с тем,что она обеспечивает прозрачную структуру хранения информации о сущностях, строгую целостность данных и контроль связей, возможность использования проверенных решений (PostgreSQL, MySQL) и упрощённую реализацию пользовательских ролей и прав.

1.4 Требования к разрабатываемой базе данных и приложению

Разрабатываемая система мониторинга оборудования для тракторного завода должна соответствовать следующим формальным требованиям, определённым на основе анализа предметной области и сравнительного анализа существующих решений.

1. База данных должна обеспечивать хранение и обработку информации о следующем перечне объектов: оборудование, детали оборудования, работники, роли, журнал технического обслуживания, датчики и их показатели.

2. Структура базы данных должна соответствовать принципам реляционной модели с нормализацией данных не ниже третьей нормальной формы (3НФ), что обеспечит устранение избыточности и целостность данных.

3. В системе должны быть реализованы механизмы обеспечения целостности данных: первичные и внешние ключи, ограничения типа NOT NULL, UNIQUE, CHECK.

4. Система должна поддерживать разграничение прав доступа на уровне ролей, с соответствующими уровнями доступа к данным (чтение, редактирование, удаление).

5. Приложение должно обеспечивать удобный интерфейс для работы с базой данных: добавление, редактирование и удаление записей, а также просмотр журналов обслуживания и показаний датчиков в реальном времени.

6. База данных должна позволять хранение временных рядов измерений от датчиков (например, температура, давление, вибрация) с возможностью последующего анализа.

7. Система должна быть ориентирована на небольшое или среднее предприятие и не требовать значительных затрат на внедрение и обучение персонала.

1.5 Формализация и описание информации, подлежащей хранению в проектируемой базе данных

База данных должна содержать информацию о следующих сущностях.

1. Оборудование — основные производственные машины и установки, подлежащие мониторингу.
2. Типы оборудования — классификация оборудования по модели, хранящая информацию о назначении и сроке эксплуатации.
3. Детали оборудования — составные элементы оборудования, которые могут выходить из строя и требуют индивидуального контроля.
4. Типы деталей оборудования — классификация деталей по модели, хранящая информацию о сроке эксплуатации и наличии запасных деталей на складе.
5. Работники — сотрудники предприятия, выполняющие функции обслуживания, ремонта или эксплуатации оборудования.
6. Роли — сущности, представляющие собой различные уровни прав доступа и хранящие соответствующие логины и пароли.
7. Журнал технического обслуживания — записи о проведённых регламентных и внеплановых работах по обслуживанию оборудования.
8. Датчики — устройства, установленные на оборудование и передающие параметры его состояния в систему мониторинга.
9. Показатели датчиков — конкретные измерения (температура, давление, вибрация и т.п.), фиксируемые датчиками во времени.

Была разработана диаграмма сущность-связь в нотации Чена, учитывающая все вышеперечисленные сущности. Она предоставлена на рисунке 1.1.

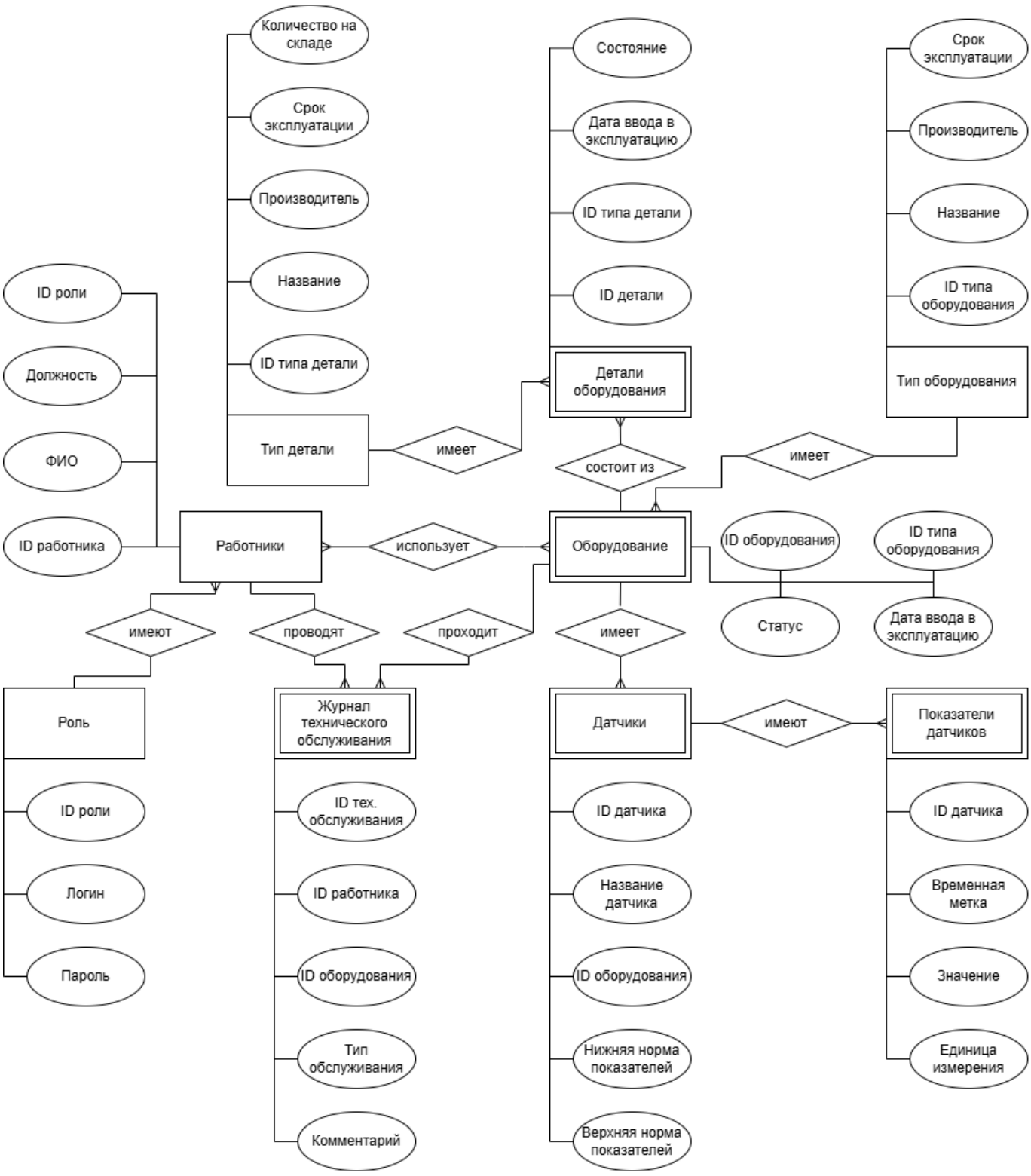

Рисунок 1.1 – Диаграмма сущность-связь базы данных

1.6 Формализация ролей

Для обеспечения разграничения прав доступа в разрабатываемой базе данных предусмотрена ролевая модель, включающая три основные категории пользователей:

— Технический директор — пользователь с полным доступом ко всем разделам базы данных. Имеет право добавлять, редактировать и удалять записи в таблицах оборудования, типов оборудования, деталей, датчиков, показателей, журнала обслуживания, а также в таблицах работников и ролей.

— Инженер — пользователь с ограниченными правами редактирования. Может добавлять, изменять и удалять записи, касающиеся оборудования, его деталей, типов, датчиков, показателей и журнала технического обслуживания. Не имеет доступа к редактированию информации о персонале.

— Оператор — пользователь с правами только на просмотр данных. Имеет доступ к информации об оборудовании, его деталях и типах, а также показаниям датчиков и журналу обслуживания. Также может просматривать информацию о сотрудниках и ролях, но не имеет права вносить изменения.

Построенная ролевая модель и способы использования приложения в соответствии с ролями представлены на диаграмме прецедентов на рисунке 1.2.

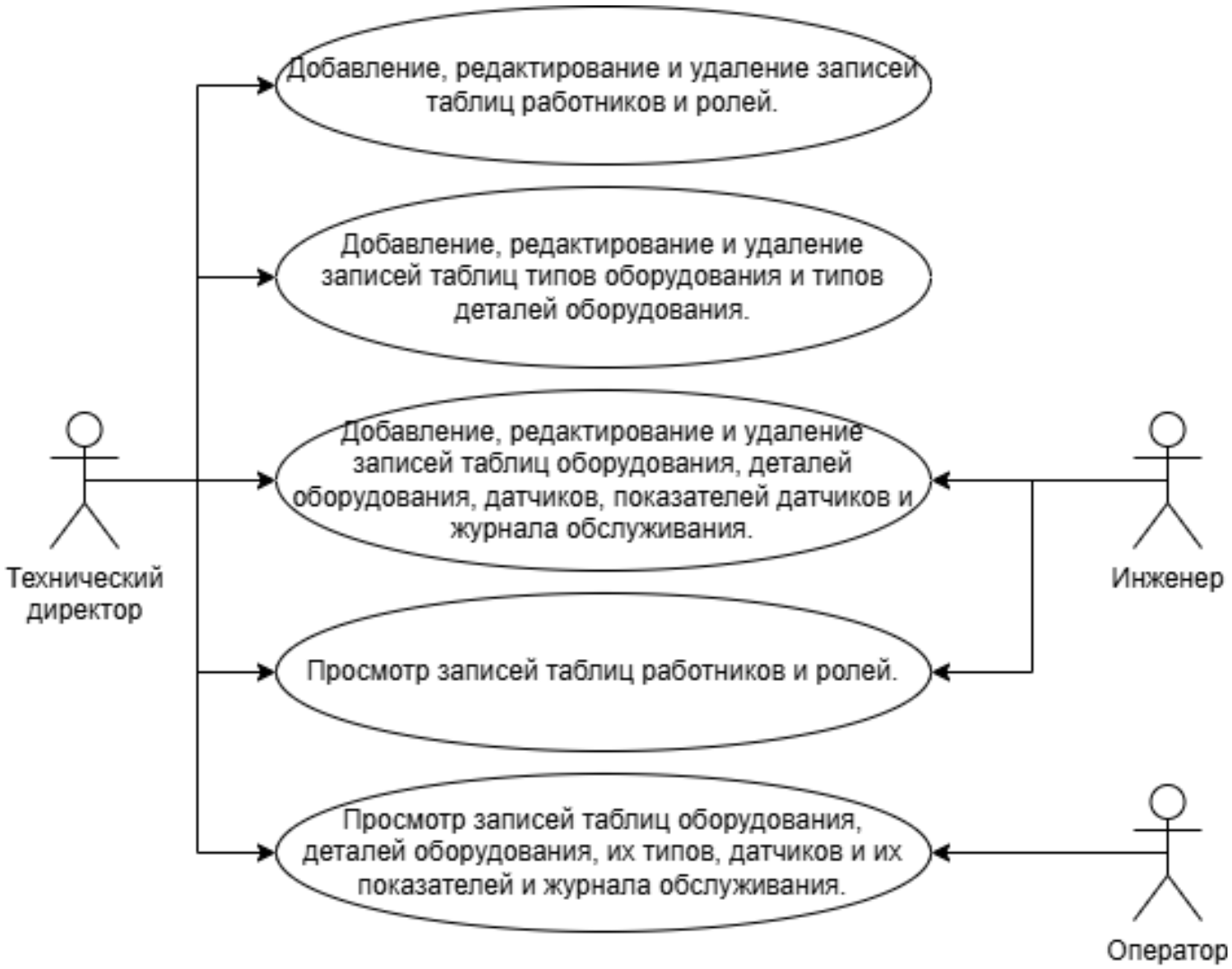

Рисунок 1.2 – Диаграмма прецедентов с учетом ролевой модели.

Вывод

В результате проведенного анализа предметной области, изучения существующих аналогов и формализации требований к базе данных была построена диаграмма сущность-связь, отражающая все основные сущности и их взаимосвязи, а также диаграмма прецедента, учитывающая ролевую модель пользователей и ключевые варианты использования системы.

2 Конструкторский раздел

В данном разделе проводится проектирование базы данных для системы мониторинга оборудования на тракторном заводе. Представлены диаграмма связей между сущностями, описание самих сущностей и реализуемых ограничений целостности данных. Также разработаны схемы функций используемых при работе с базой данных, и определена ролевая модель, включающая разграничение прав доступа к объектам базы данных в соответствии с пользовательскими ролями.

2.1 Описание сущностей и ограничений целостности проектируемой базы данных

1. ComponentTypes. Типы деталей оборудования. Классификация составных компонентов оборудования.
   - component_type_id [первичный ключ], уникальный идентификатор типа (целое число);
   - name, наименование компонента (строка);
   - manufacturer, производитель (строка);
   - service_life, срок службы, в днях (целое число), ограничение, больше 0;
   - quantity, доступное количество на складе (целое число), ограничение, больше или равно 0.
2. Components. Детали оборудования. Фактические экземпляры компонентов, установленные на оборудование.
   - component_id [первичный ключ], уникальный идентификатор компонента (целое число);
   - component_type_id [внешний ключ], ссылка на тип компонента (целое число);
   - equipment_id [внешний ключ], идентификатор оборудования, где установлен компонент (целое число);
   - commissioning_date, дата установки (дата);
   - status, текущий статус (строка).

3. EquipmentTypes. Типы оборудования. Модели и технические параметры оборудования.
    - equipment_type_id [первичный ключ], уникальный идентификатор типа оборудования (целое число);
    - name, наименование (строка);
    - manufacturer, производитель (строка);
    - service_life, срок службы, в днях (целое число).
4. Equipments. Оборудование. Конкретные производственные установки.
    - equipment_id [первичный ключ], уникальный идентификатор (целое число);
    - equipment_type_id [внешний ключ], ссылка на тип оборудования (целое число);
    - commissioning_date, дата ввода в эксплуатацию (дата);
    - status, текущий статус (строка).
5. Employees. Работники. Сотрудники, осуществляющие эксплуатацию и обслуживание.
    - employee_id [первичный ключ], уникальный идентификатор (целое число);
    - full_name, полное имя (строка);
    - job_title, должность (строка);
    - role_id [внешний ключ], ссылка на роль доступа (целое число).
6. Assignment. Назначения работников. Соответствие между сотрудниками и закреплённым за ними оборудованием.
    - equipment_id [первичный ключ] [внешний ключ], идентификатор оборудования (целое число);
    - employee_id [первичный ключ] [внешний ключ], идентификатор сотрудника (целое число).
7. Roles. Роли пользователей. Уровни доступа и авторизации.

- role_id [первичный ключ], уникальный идентификатор роли (целое число);
- login, логин пользователя (строка);
- password, зашифрованный пароль (строка).

8. MaintenanceJournal. Журнал обслуживания. Учёт проведённых технических работ.
   - maintenance_id [первичный ключ], уникальный идентификатор записи (целое число);
   - equipment_id [внешний ключ], идентификатор оборудования (целое число);
   - employee_id [внешний ключ], сотрудник, выполнивший работу (целое число);
   - type, тип работы (строка);
   - timestamp, дата и время выполнения (дата/время);
   - comment, дополнительный комментарий (строка).
9. Sensors. Датчики. Физические датчики, установленные на оборудовании.
   - sensor_id [первичный ключ], уникальный идентификатор датчика (целое число);
   - name, название датчика (строка);
   - equipment_id [внешний ключ], идентификатор оборудования (целое число);
   - lower_limit, нижняя граница допустимого значения (число);
   - upper_limit, верхняя граница допустимого значения (число).
10. SensorReadings. Показания датчиков. Записи о значениях, снятых с датчиков.
    - sensor_id [первичный ключ] [внешний ключ], идентификатор датчика (целое число);
    - timestamp [первичный ключ], момент снятия показаний (дата/время);
    - value, зафиксированное значение (число);

— measurement_unit, единица измерения (строка).

Согласно сформулированному выше описанию сущностей и их ограничений целостности, была также построена диаграмма сущность-связь, изображенная на рисунке 2.1

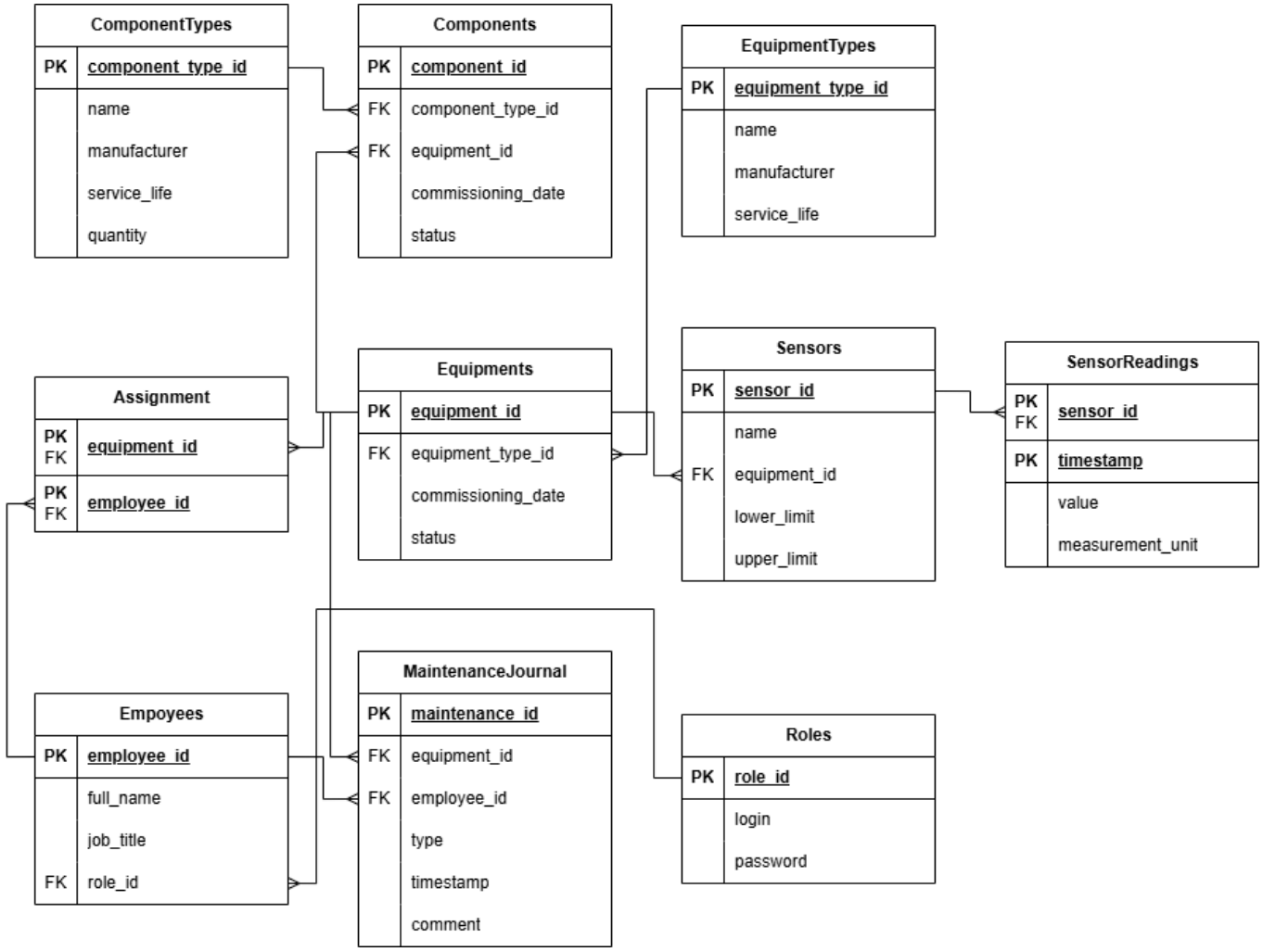

Рисунок 2.1 – Диаграмма сущность-связь проектируемой базы данных

2.2 Описание ролевой модели на уровне базы данных

В проектируемой базе данных реализуются следующие роли пользователей.

1. Оператор. Роль с правами на выполнение операций SELECT из таблиц Equipments, EquipmentParts, EquipmentTypes, Sensors, SensorReadings и MaintenanceJournal. Ограничивается только чтением данных без возможности их модификации.

2. Инженер. Роль с правами на выполнение операций SELECT, INSERT, UPDATE и DELETE для таблиц Equipments, EquipmentParts, Sensors,

SensorReadings и MaintenanceJournal. Таким образом, инженер может как анализировать данные, так и регистрировать новые показания или записи об обслуживании.

3. Технический директор. Роль с максимальными привилегиями, включающими полный доступ (SELECT, INSERT, UPDATE, DELETE) ко всем таблицам базы данных, включая Employees и Roles. Данная роль обладает административными функциями, позволяющими управлять пользователями и их доступом.

2.3 Описание проектируемой функции базы данных

В рамках разработанной базы данных реализована функция log_sensor_warning() и соответствующий триггер trg_log_sensor_warning, предназначенные для автоматического контроля значений, снимаемых с датчиков, и ведения учёта возможных отклонений от допустимых параметров оборудования.

Функция log_sensor_warning() выполняет проверку нового показания датчика на соответствие установленным допустимым границам (lower_limit и upper_limit). Если зафиксированное значение превышает верхнюю границу или ниже нижней, функция автоматически создаёт запись в журнале обслуживания (MaintenanceJournal) с типом работы 'предупреждение'. В комментарии к записи фиксируется идентификатор датчика и зафиксированное значение, вышедшее за пределы нормы. При этом поле employee_id оставляется пустым, так как запись формируется системой автоматически.

Схема алгоритма работы данного триггера представлена на рисунке 2.2.

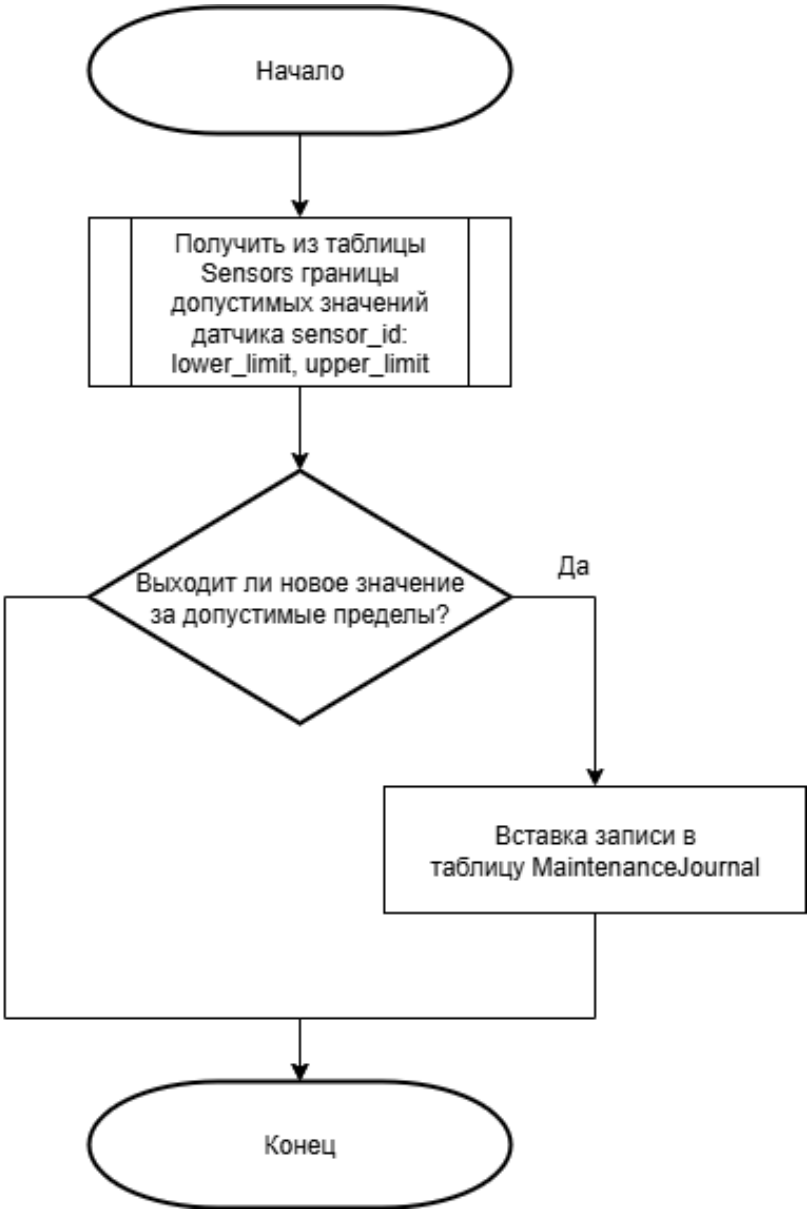

Рисунок 2.2 – Схема работы триггера

Вывод

В результате проектирования базы данных для системы мониторинга оборудования на тракторном заводе были созданы диаграмма связей между сущностями и подробное описание каждой сущности с реализуемыми ограничениями целостности данных. Кроме того, разработана схема функции, обеспечивающей работу с базой данных, и определена ролевая модель с разграничением прав доступа в соответствии с пользовательскими ролями.

3 Технологический раздел

В данном разделе рассматриваются средства реализации базы данных и приложения, приводится реализация структуры таблиц с ограничениями целостности, ролевого доступа пользователей, триггера для контроля данных, а также программный интерфейс для создания, чтения, обновления и удаления записей, включая работу с взаимосвязанными сущностями.

3.1 Средства реализации

В качестве системы управления базами данных выбрана PostgreSQL [8]. Данный выбор обусловлен тем, что PostgreSQL полностью соответствует принципам реляционной модели и поддерживает нормализацию данных не ниже третьей нормальной формы, что позволяет устранить избыточность и обеспечить целостность информации. СУБД предоставляет механизмы реализации первичных и внешних ключей, ограничений NOT NULL, CHECK, а также разграничение прав доступа на уровне ролей, что отвечает требованиям к защите данных. PostgreSQL поддерживает работу с временными рядами, что делает её подходящей для хранения и анализа показаний датчиков. Кроме того, использование данной СУБД не требует значительных затрат на внедрение и администрирование, что важно для предприятия среднего уровня. Также PostgreSQL предоставляет развитые средства для создания функций и триггеров, что необходимо для реализации автоматизированного контроля данных. Таким образом данная СУБД удовлетворяет всем требованиям, представленным к работе проектируемой базы данных.

Для прикладной логики выбран Python [9], так как он обеспечивает удобное взаимодействие с PostgreSQL через специализированные библиотеки (SQLAlchemy [10]), поддерживает обработку всех спроектированных структур данных и отличается простотой разработки и сопровождения.

3.2 Реализация

В данном разделе представлена реализация таблиц, ролевой модели и триггера базы данных.

3.2.1 Создание таблиц базы данных и их ограничений целостности

На листинге 3.1 представлено создание всех спроектированных таблиц базы данных и описанных ограничений целостности данных.

Листинг 3.1 – Реализация создания таблиц базы данных и ограничений целостности

```
CREATE TABLE IF NOT EXISTS component_types (
    component_type_id BIGSERIAL PRIMARY KEY,
    name TEXT NOT NULL,
    manufacturer TEXT,
    service_life INTEGER NOT NULL CHECK (service_life > 0),
    quantity INTEGER NOT NULL CHECK (quantity >= 0)
);

CREATE TABLE IF NOT EXISTS equipment_types (
    equipment_type_id BIGSERIAL PRIMARY KEY,
    name TEXT NOT NULL,
    manufacturer TEXT,
    service_life INTEGER CHECK (service_life > 0)
);

CREATE TABLE IF NOT EXISTS equipments (
    equipment_id BIGSERIAL PRIMARY KEY,
    equipment_type_id BIGINT NOT NULL REFERENCES
       equipment_types(equipment_type_id) ON DELETE RESTRICT,
    commissioning_date DATE,
    status TEXT
);

CREATE TABLE IF NOT EXISTS components (
    component_id BIGSERIAL PRIMARY KEY,
    component_type_id BIGINT NOT NULL REFERENCES
       component_types(component_type_id) ON DELETE RESTRICT,
    equipment_id BIGINT REFERENCES equipments(equipment_id) ON
       DELETE SET NULL,
    commissioning_date DATE,
    status TEXT
);

CREATE TABLE IF NOT EXISTS roles (
    role_id BIGSERIAL PRIMARY KEY,
```

```
    login TEXT NOT NULL UNIQUE,
    password TEXT NOT NULL
);

INSERT INTO roles (login, password) VALUES
    ('operator', 'operator_pass'),
    ('engineer', 'engineer_pass'),
    ('tech_director', 'director_pass');

CREATE TABLE IF NOT EXISTS employees (
    employee_id BIGSERIAL PRIMARY KEY,
    full_name TEXT NOT NULL,
    job_title TEXT,
    role_id BIGINT REFERENCES roles(role_id) ON DELETE SET NULL
);

CREATE TABLE IF NOT EXISTS assignment (
    equipment_id BIGINT NOT NULL REFERENCES equipments(equipment_id)
       ON DELETE CASCADE,
    employee_id BIGINT NOT NULL REFERENCES employees(employee_id) ON
       DELETE CASCADE,
    PRIMARY KEY (equipment_id, employee_id)
);

CREATE TABLE IF NOT EXISTS maintenance_journal (
    maintenance_id BIGSERIAL PRIMARY KEY,
    equipment_id BIGINT NOT NULL REFERENCES equipments(equipment_id)
       ON DELETE CASCADE,
    employee_id BIGINT NOT NULL REFERENCES employees(employee_id) ON
       DELETE SET NULL,
    type TEXT NOT NULL CHECK (type IN ('ремонт', 'техническое
       обслуживание', 'предупреждение')),
    timestamp TIMESTAMP WITH TIME ZONE NOT NULL DEFAULT now(),
    comment TEXT
);

CREATE TABLE IF NOT EXISTS sensors (
    sensor_id BIGSERIAL PRIMARY KEY,
    name TEXT NOT NULL,
    equipment_id BIGINT NOT NULL REFERENCES equipments(equipment_id)
       ON DELETE CASCADE,
```

```
    lower_limit DOUBLE PRECISION,
    upper_limit DOUBLE PRECISION,
    CHECK (lower_limit IS NULL OR upper_limit IS NULL OR lower_limit
       < upper_limit)
);

CREATE TABLE IF NOT EXISTS sensor_readings (
    sensor_id BIGINT NOT NULL REFERENCES sensors(sensor_id) ON
       DELETE CASCADE,
    timestamp TIMESTAMP WITH TIME ZONE NOT NULL,
    value DOUBLE PRECISION NOT NULL,
    measurement_unit TEXT NOT NULL,
    PRIMARY KEY (sensor_id, timestamp)
);
```

3.2.2 Реализация ролевой модели базы данных

На листинге 3.2 представлено создание всех формализованных ролей и выдача им соответствующих привилегий и прав доступа к данных.

Листинг 3.2 – Реализация создания ролей и назначения соответствующих прав доступа

```
CREATE ROLE operator LOGIN PASSWORD 'operator_pass';
GRANT CONNECT ON DATABASE tractor_factory TO operator;
GRANT SELECT ON
    Equipments,
    Components,
    EquipmentTypes,
    Sensors,
    SensorReadings,
    MaintenanceJournal
TO operator;

CREATE ROLE engineer LOGIN PASSWORD 'engineer_pass';
GRANT CONNECT ON DATABASE tractor_factory TO engineer;
GRANT SELECT, INSERT, UPDATE, DELETE ON
    Equipments,
    Components,
    Sensors,
    SensorReadings,
    MaintenanceJournal
TO engineer;
```

```
CREATE ROLE tech_director LOGIN PASSWORD 'director_pass';
GRANT CONNECT ON DATABASE tractor_factory TO tech_director;
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA public
   TO tech_director;
```

3.2.3 Реализация и тестирование триггера

На листинге 3.3 представлена реализация триггера trg_log_sensor_warning для проверки измерений датчиков при добавлении данных в таблицу SensorReadings.

Листинг 3.3 – Реализация триггера проверки новых измерений датчиков

```
CREATE OR REPLACE FUNCTION log_sensor_warning()
RETURNS TRIGGER AS $$
DECLARE
    lower_val NUMERIC;
    upper_val NUMERIC;
BEGIN
    SELECT lower_limit, upper_limit
    INTO lower_val, upper_val
    FROM Sensors
    WHERE sensor_id = NEW.sensor_id;

    IF NEW.value < lower_val OR NEW.value > upper_val THEN
        INSERT INTO MaintenanceJournal(equipment_id, employee_id,
           type, timestamp, comment)
        VALUES (
            (SELECT equipment_id FROM Sensors WHERE sensor_id =
               NEW.sensor_id),
            NULL,
            'warning',
            NEW.timestamp,
            CONCAT('Показатель датчика ', NEW.sensor_id, ' вышел за
               пределы: ', NEW.value)
        );
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trg_log_sensor_warning
AFTER INSERT ON SensorReadings
FOR EACH ROW
EXECUTE FUNCTION log_sensor_warning();
```

Для тестирования данного триггера были выделены следующие классы эквивалентности:

— значение в пределах допустимого диапазона, триггер не добавляет запись в журнал;

— значение ниже нижней границы, запись добавляется в журнал;

— значение выше верхней границы, запись добавляется в журнал;

— значение ровно на границе, триггер не добавляет запись в журнал;

— граничные значения не заданы, триггер не добавляет запись в журнал;

— некорректная ссылка на датчик, возникновение ошибки.

Все тесты, реализованные согласно данным классам эквивалентности, были пройдены успешно.

3.3 Интерфейс взаимодействия с базой данных

Для обеспечения удобного доступа к данным был реализован функционал взаимодействия с базой данных с использованием библиотеки SAFRS. Дополнительно применён интерфейс Swagger, позволяющий в удобной форме выполнять операции с данными и проверять корректность работы системы.

Методы интерфейса, обеспечивающие возможность взаимодействия с таблицей типов компонентов оборудования.

— Получение списка всех типов компонентов.

— Создание нового типа компонента.

— Получение информации о конкретном типе компонента по идентификатору.

— Обновление данных конкретного типа компонента.

— Удаление конкретного типа компонента.

— Получение списка компонентов, относящихся к определённому типу.

— Добавление компонентов к определённому типу.

— Обновление информации о компонентах, связанных с определённым типом.

— Удаление компонентов, связанных с определённым типом.

Методы интерфейса, обеспечивающие возможность взаимодействия с таблицей компонентов оборудования.

— Получение списка всех компонентов.

— Создание нового компонента.

— Получение информации о конкретном компоненте по идентификатору.

— Обновление данных конкретного компонента.

— Удаление конкретного компонента.

— Получение информации о типе компонента, к которому он относится.

— Обновление информации о связи компонента с его типом.

— Удаление связи компонента с его типом.

— Получение информации об оборудовании, к которому относится компонент.

— Обновление информации о связи компонента с оборудованием.

— Удаление связи компонента с оборудованием.

Методы интерфейса, обеспечивающие возможность взаимодействия с таблицей типов оборудования.

— Получение списка всех типов оборудования.

— Создание нового типа оборудования.

— Получение информации о конкретном типе оборудования по идентификатору.

— Обновление данных конкретного типа оборудования.

— Удаление конкретного типа оборудования.

— Получение списка оборудования, относящегося к определённому типу.

— Добавление оборудования к определённому типу.

— Обновление информации о связи оборудования с определённым типом.

— Удаление оборудования, связанного с определённым типом.

Методы интерфейса, обеспечивающие возможность взаимодействия с таблицей оборудования.

— Получение списка всего оборудования.

— Создание нового оборудования.

— Получение информации о конкретном оборудовании по идентификатору.

— Обновление данных конкретного оборудования.

— Удаление конкретного оборудования.

— Получение списка компонентов, относящихся к оборудованию.

— Добавление компонентов к оборудованию.

— Обновление информации о компонентах, связанных с оборудованием.

— Удаление компонентов, связанных с оборудованием.

— Получение списка сотрудников, закреплённых за оборудованием.

— Добавление сотрудников к оборудованию.

— Обновление информации о сотрудниках, связанных с оборудованием.

— Удаление сотрудников, связанных с оборудованием.

— Получение информации о типе оборудования.

— Обновление информации о связи оборудования с его типом.

— Удаление связи оборудования с его типом.

— Получение записей журнала обслуживания, связанных с оборудованием.

— Добавление записей в журнал обслуживания оборудования.

— Обновление информации о записях журнала обслуживания оборудования.

— Удаление записей журнала обслуживания оборудования.

— Получение списка датчиков, установленных на оборудовании.

— Добавление датчиков к оборудованию.

— Обновление информации о датчиках, связанных с оборудованием.

— Удаление датчиков, связанных с оборудованием.

Методы интерфейса, обеспечивающие возможность взаимодействия с таблицей сотрудников.

— Получение списка всех сотрудников.

— Создание нового сотрудника.

— Получение информации о конкретном сотруднике по идентификатору.

— Обновление данных конкретного сотрудника.

— Удаление конкретного сотрудника.

— Получение списка оборудования, закреплённого за сотрудником.

— Добавление оборудования сотруднику.

— Обновление информации о закреплённом оборудовании сотрудника.

— Удаление оборудования, связанного с сотрудником.

— Получение записей журнала обслуживания, связанных с сотрудником.

— Добавление записей в журнал обслуживания сотрудника.

— Обновление информации о записях журнала обслуживания сотрудника.

— Удаление записей журнала обслуживания сотрудника.

— Получение информации о роли сотрудника.

— Обновление информации о роли сотрудника.

— Удаление роли, связанной с сотрудником.

Методы интерфейса, обеспечивающие возможность взаимодействия с таблицей журнала обслуживания.

— Получение списка всех записей журнала обслуживания.

— Создание новой записи журнала обслуживания.

— Получение информации о конкретной записи журнала обслуживания по идентификатору.

— Обновление данных конкретной записи журнала обслуживания.

— Удаление конкретной записи журнала обслуживания.

— Получение информации о сотруднике, связанном с записью журнала обслуживания.

— Обновление информации о сотруднике, связанном с записью журнала обслуживания.

— Удаление сотрудника, связанного с записью журнала обслуживания.

— Получение информации об оборудовании, связанном с записью журнала обслуживания.

— Обновление информации об оборудовании, связанном с записью журнала обслуживания.

— Удаление оборудования, связанного с записью журнала обслуживания.

Методы интерфейса, обеспечивающие возможность взаимодействия с таблицей датчиков.

— Получение списка всех датчиков.

— Создание нового датчика.

— Получение информации о конкретном датчике по идентификатору.

— Обновление данных конкретного датчика.

— Удаление конкретного датчика.

— Получение информации об оборудовании, связанном с датчиком.

— Обновление информации об оборудовании, связанном с датчиком.

— Удаление оборудования, связанного с датчиком.

— Получение показаний датчика.

— Добавление новых показаний к датчику.

— Обновление показаний датчика.

— Удаление показаний датчика.

Методы интерфейса, обеспечивающие возможность взаимодействия с таблицей показаний датчиков.

— Получение списка всех показаний датчиков.

— Создание нового показания датчика.

— Получение информации о конкретном показании датчика по идентификатору.

— Обновление данных конкретного показания датчика.

— Удаление конкретного показания датчика.

— Получение информации о датчике, связанном с показанием.

— Обновление информации о датчике, связанном с показанием.

— Удаление датчика, связанного с показанием.

На рисунках 3.1–3.2 представлена часть методов реализованного программного интерфейса.

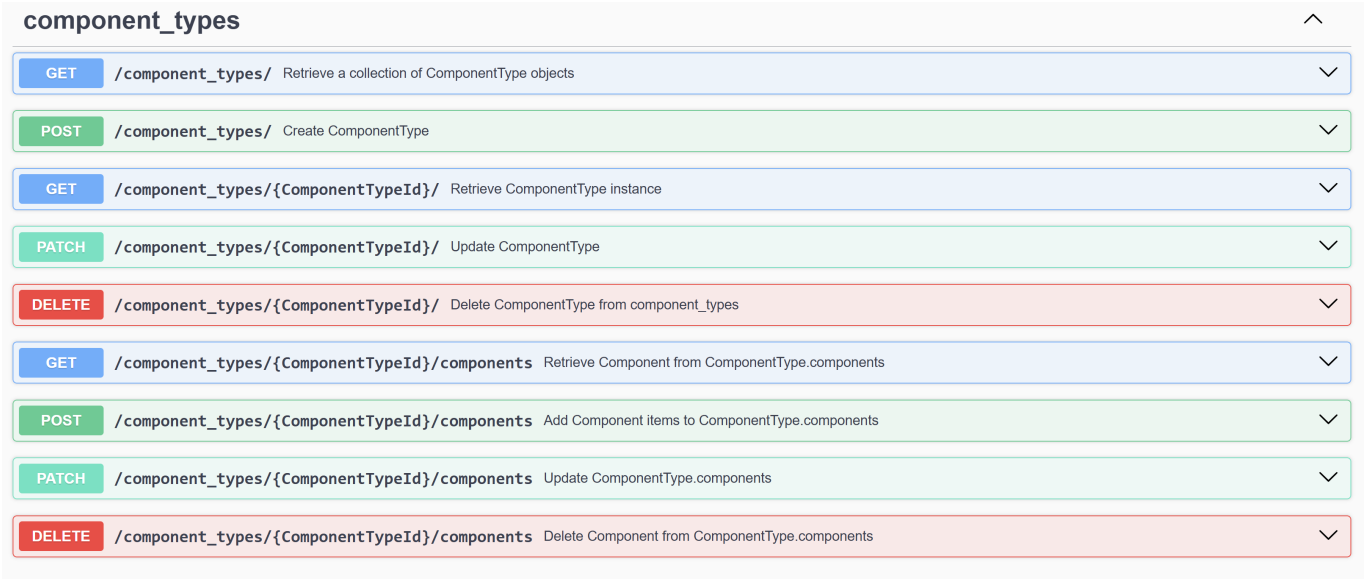

Рисунок 3.1 – Пример реализации программного интерфейса (часть 1)

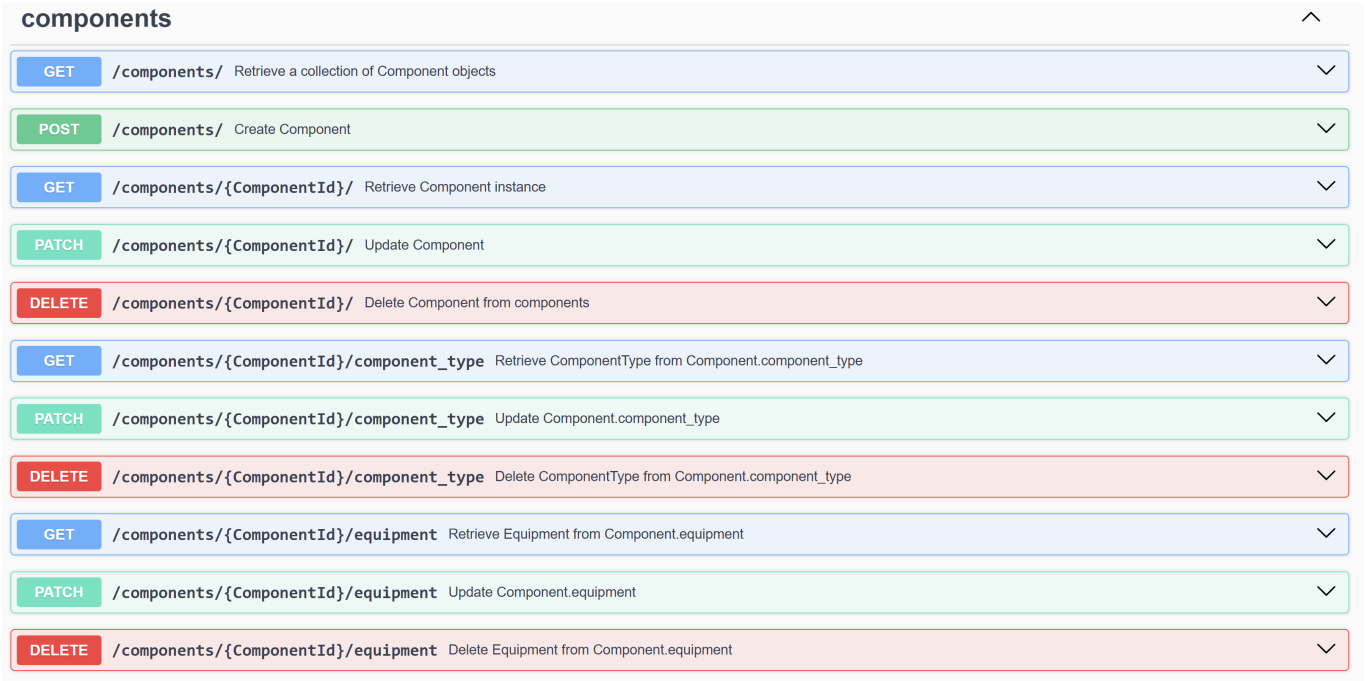

Рисунок 3.2 – Пример реализации программного интерфейса (часть 2)

Вывод

В результате реализации была создана полнофункциональная база данных с обеспечением целостности и корректности данных, реализован механизм ролевого доступа пользователей, а также триггер для автоматического контроля показаний датчиков. Программный интерфейс позволяет удобно управлять записями и

взаимодействовать с взаимосвязанными сущностями, обеспечивая полноту и надёжность работы системы. Все разработанные компоненты успешно протестированы и готовы к использованию в приложении.

4 Исследовательский раздел

В данном разделе представлена постановка исследования, технические характеристики устройства, используемого для замеров, а также результаты проведенного исследования.

4.1 Постановка исследования

Цель исследования заключается в изучении влияния объема обрабатываемых данных на производительность базы данных. Основная задача состоит в анализе зависимости времени выполнения запросов от количества записей в системе. Полученные результаты позволят оценить эффективность структуры базы данных и выявить закономерности изменения производительности при увеличении объема данных.

Исследование будет проводиться на таблице «Типы компонентов», так как эта таблица является независимой. Также были определены основные виды действий над данными для которых будут проводиться замеры — добавление, чтение, обновление и удаление.

4.2 Технические характеристики

Технические характеристики устройства, на котором выполнялось исследование:

— процессор 11th Gen Intel(R) Core(TM) i5-1135G7

— оперативная память 8Гб

— операционная система Windows 11 Домашняя версии 24H2

4.3 Результаты исследования

Результаты проведения исследования приведены в таблице 4.1. Для каждого выделенного запроса приведено среднее время его выполнения в миллисекундах. Для определения среднего времени замеры были проведены 500 раз. Во время выполнения исследования машина была нагружена только системой тестирования.

Таблица 4.1 – Результаты замеров времени выполнения запросов в зависимости от количества записей в базе данных

Количество записей	**INSERT (мс)**	**SELECT (мс)**	**UPDATE (мс)**	**DELETE (мс)**
0	14.990	8.546	15.231	10.694
1000	15.861	9.136	15.852	11.328
5000	17.087	9.061	16.258	11.450
15000	15.604	9.159	15.797	11.179
25000	16.861	10.757	17.812	13.003
50000	17.785	10.329	18.189	12.742
75000	16.194	9.217	16.339	11.505
100000	15.885	8.933	16.877	12.200

Полученная зависимость времени выполнения запросов от количества записей в базе данных представлена на рисунке 4.1.

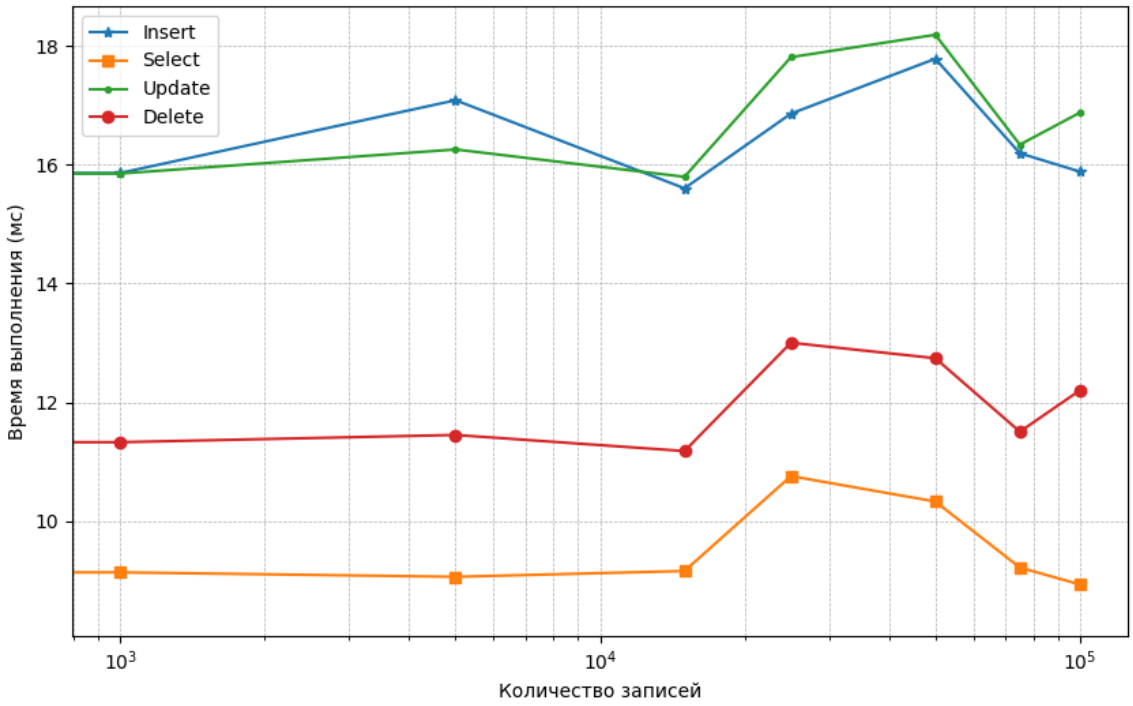

Рисунок 4.1 – Зависимость времени выполнения запросов от количества записей в базе данных

Вывод

По результатам исследования можно сделать вывод, что время выполнения операций не растет экспоненциально с увеличением объема данных. При масштабировании от 0 до 100000 записей наблюдается сохранение того же порядка времени исполнения.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы была разработана база данных для системы мониторинга оборудования на тракторном заводе, обеспечивающая централизованный сбор, хранение и анализ данных о состоянии техники. Были определены ключевые сущности и их взаимосвязи, спроектирована структура таблиц с ограничениями целостности, реализована ролевой модель доступа, а также создан программный интерфейс для взаимодействия с данными.

Проведено тестирование функциональности базы данных и триггеров, а также исследование производительности, показавшее зависимость времени выполнения запросов от объема обрабатываемых данных. Полученные результаты подтверждают корректность работы системы и её способность эффективно обрабатывать информацию в условиях увеличения нагрузки.

Список литературы

1. Обзорная презентация продукта 1С:ТОИР Управление ремонтами и обслуживанием оборудования [Электронный ресурс]. — Режим доступа: `https://solutions.1c.ru/catalog/eam/materials` (Дата обращения: 13.05.2025).
2. Siemens MindSphere – Capabilities [Электронный ресурс]. — Режим доступа: `https://plm.sw.siemens.com/en-US/insights-hub/capabilities/?c=198495` (Дата обращения: 13.05.2025).
3. Система автоматизированного проектирования технологических процессов ВЕРТИКАЛЬ [Электронный ресурс]. — Режим доступа: `https://ascon.ru/products/vertikal/` (Дата обращения: 13.05.2025).
4. *Б.Я.Советов.* Базы данных : учебник для вузов / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской. - 4-е изд., перераб. и доп. //. — Издательство Юрайт, 2025.
5. *В.В.Бойко.* Проектирование баз данных информационных систем / Бойко В.В., Савинков В.М. - Изд. 2, перер. и доп. //. — Финансы и статистика, 1989.
6. *К.Дж.Дейт.* Введение в системы баз данных, 8-е издание.: Пер. с англ //. — Издательский дом Вильяме, 2005.
7. *П.Ю.Парфенов.* Постреляционные хранилища данных: учебное пособие. //. — Центр Информационных Технологий, 2016.
8. Документация PostgreSQL [Электронный ресурс]. — Режим доступа: `https://www.postgresql.org/` (Дата обращения: 13.05.2025).
9. Документация языка Python [Электронный ресурс]. — Режим доступа: `https://docs.python.org/3/` (Дата обращения: 13.05.2025).
10. Документация модуля SQLAlchemy [Электронный ресурс]. — Режим доступа: `https://www.sqlalchemy.org/` (Дата обращения: 13.05.2025).

ПРИЛОЖЕНИЕ А

Презентация к курсовой работе состоит из 11 слайдов.