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

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

«Разработка базы данных для системы мониторинга на тракторном заводе»

Студент ИУ7-62Б (Группа) _____ (Подпись, дата) Руденко М.А. (И. О. Фамилия)

Руководитель курсовой работы _____ (Подпись, дата) Кудрявцев М.А. (И. О. Фамилия)

2024 г.

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

В настоящей расчетно-пояснительной записке применяют следующие сокращения и обозначения.

ПО — программное обеспечение.

БД — база данных.

СУБД — система управления базами данных.

ВВЕДЕНИЕ

Автоматизация производственных процессов на тракторном заводе имеет стратегическое значение для повышения эффективности производства и конкурентоспособности предприятия. Разработка базы данных и приложения для мониторинга поможет улучшить управление ресурсами, уменьшить затраты и увеличить производительность труда.

Разрабатываемая информационная система представляет собой комплексное решение для управления производственными процессами на тракторном заводе. Она включает в себя базу данных для хранения информации о сотрудниках, тракторах, производственных линиях, техническом обслуживании и запчастях. Приложения доступа к базе данных позволят оперативно управлять данными, отслеживать состояние оборудования, планировать и отслеживать процессы обслуживания техники.

Целью данной курсовой работы является проектирование и разработка программного обеспечения для мониторинга оборудования на тракторном заводе.

Для достижения поставленной цели необходимо решить следующие задачи:

— провести анализ предметной области и сформулировать требования и ограничения к разрабатываемой базе данных и приложению;

— сформулировать описание пользователей проектируемого приложения;

— спроектировать сущности базы данных и ограничения целостности;

— спроектировать ролевую модель на уровне базы данных;

— выбрать средства реализации базы данных и приложения;

— описать методы тестирования разработанного функционала и разработать тесты для проверки корректности работы приложения;

— исследовать зависимость времени выполнения запроса от наличия индексов.

1 Аналитический раздел

В данном разделе приводится анализ предметной области и существующих решений, формулируются требования к БД и ПО, формализуется и описывается информация, подлежащая хранению в проектируемой БД, и пользователи. На основе приведенного анализа строятся диаграмма вариантов использования и ER-диаграмма сущностей в нотации Чена.

1.1 Анализ предметной области и существующих решений

Предметная область включает в себя производственные процессы на тракторном заводе: контроль производства, техническое обслуживание, учет запасов запчастей и другие аспекты, связанные с производством сельскохозяйственной техники.

Система мониторинга позволяет следить за работой оборудования. Оператору не придётся каждый раз идти к станку, чтобы узнать, как он работает и всё ли с ним в порядке. Это повышает эффективность работы персонала, особенно при большом парке оборудования.

Собранная информация, кроме того, позволяет планировать и вовремя проводить техобслуживание и ремонты, что позволяет снизить простои и издержки.

Еще одна из задач систем мониторинга – идентификация работника и действий, производимых на станке: собирается информация о том, какая партия изделий была произведена, на каком станке, за какое время и кто был оператором оборудования. Для своевременного обслуживания важно собирать данные об износе оборудования.

Среди уже существующих проектов, решающих поставленную задачу, для сравнения были выделены три аналога: Winnum, DPA и AF Monitor. Сравнение проводилось по ряду критериев, а именно наличие возможности просмотра общей информации о производстве, контроль и регистрация времени и причин простоев и регистрация заявок на техобслуживание. Результаты сравнения приведены в таблице 1.1.

Таблица 1.1 – Сравнение функциональности систем AF, DPA и Winnum

	AF	**DPA**	**Winnum**
Просмотр информации о производстве	+	+	+
Регистрация времени и причин простоев	+	+	+
Регистрация заявок на техобслуживание	-	-	-

Как видно в таблице 1.1, рассмотренные решения не предоставляют функционал для регистрации заявок на проведение технического обслуживания производственных линий. Таким образом, данная работа отличается от существующих решений тем, что она удовлетворяет всем установленным критериям.

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

Основываясь на анализе предметной области, можно выделить следующие сущности:

- трактор;
- производственная линия;
- сотрудник;
- запчасть;
- заказ запчастей;
- заявка на обслуживание;
- отчет об обслуживании.

В таблице 1.2 приведены сведения о рассматриваемых сущностях.

На рисунке 1.1 приведена ER-диаграмма сущностей в нотации Чена.

Сущность	Сведения
трактор	модель, год выпуска, тип двигателя, модель двигателя, мощность двигателя, экологическая норма выброса, емкость топливного бака, колесная формула, размер шин задних колес, размер шин передних колес, длина, ширина, высота по кабине
производственная линия	название, длина, ширина, высота, состояние, производство (тр/мес), простой, частота техосмотров, дата последнего техосмотра, дата следующего техосмотра, процент брака
сотрудник	имя, фамилия, отчество, e-mail, пароль, дата рождения, отдел, роль, пол
запчасть	название, длина, ширина, высота, страна производитель, количество на складе, цена
заказ запчастей	статус, стоимость, дата заказа, сотрудник, заявка
заявка на обслуживание	статус, тип, дата, описание, производственная линия, сотрудник
отчет об обслуживании	производственная линия, заявка, сотрудник, дата начала обслуживания, дата завершения обслуживания, цена ремонта, описание проделанных работ

Таблица 1.2 – Описание сущностей

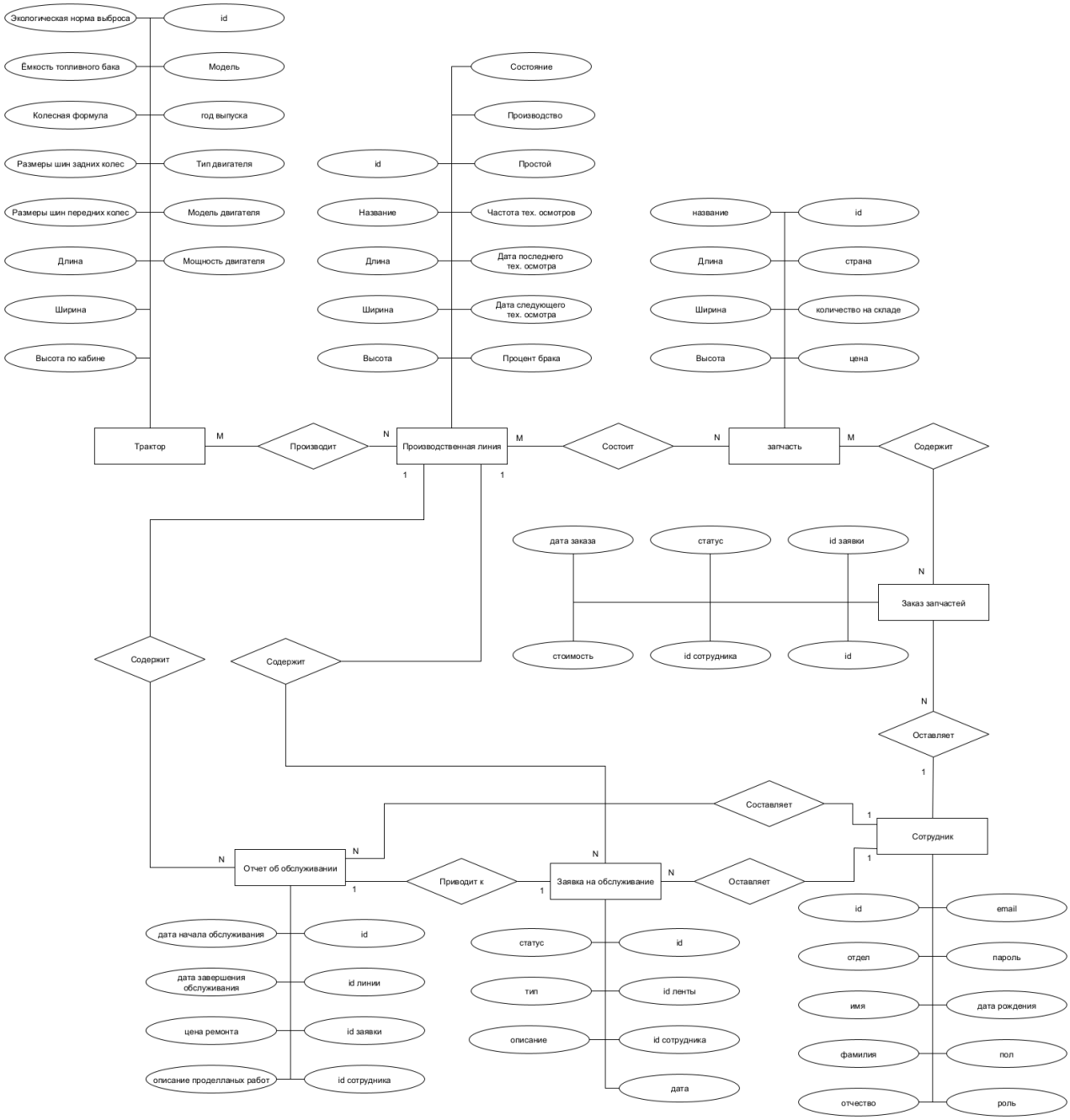

Рисунок 1.1 – ER-диаграмма сущностей в нотации Чена

1.3 Формализация и описание пользователей

В проектируемом приложении к базе данных выделим четыре типа пользователей:

– оператор производства – это сотрудник предприятия, контролирующий функционирование производственных линий. Он может просматривать информацию о тракторах и производственных линиях, создавать заявки на обслуживание и изучать отчеты об обслуживании;

– специалист по обслуживанию – это сотрудник, отвечающий за техническое обслуживание производственных линий, их техосмотры и ремонт. Он может просматривать информацию о производственных линиях и запчастях на складе, просматривать журнал обслуживания производственных линий, заказывать запчасти и регистрировать выполненное обслуживание заполняя отчет;

– администратор – это сотрудник, отвечающий за выдачу прав доступа к системе другим сотрудникам, он может просматривать информацию о зарегистрированных пользователях и менять уровень доступа отдельных сотрудников;

– неавторизованный пользователь – это пользователь не вошедший в систему, он может войти, введя свой логин и пароль, или зарегистрироваться в системе.

На рисунке 1.2 представлена диаграмма вариантов использования.

Рисунок 1.2 – Диаграмма вариантов использования

1.4 Анализ моделей баз данных

База данных – это некоторый набор перманентных (постоянно хранимых) данных, используемых прикладными программными системами какого-либо предприятия[1].

Система управления базой данных (СУБД) – совокупность программ, с помощью которых осуществляются управление структурой базы данных и контроль доступа к данным, хранящимся в ней[2].

Модель данных – это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь[1].

По модели данных СУБД делятся на дореляционные, реляционные и постреляционные.

1.4.1 Дореляционные модели

Основные типы дореляционных моделей[1]:

- инвертированные списки;
- иерархические;
- сетевые.

Инвертированные списки

Модель управления данными на основе инвертированных списков является широко распространённой в современных реляционных системах управления базами данных (СУБД)[3]. Однако, в таких системах пользователи не имеют прямого доступа к инвертированным спискам, которые являются внутренней структурой для управления доступом к данным.

База данных, построенная на основе инвертированных списков, схожа с реляционной БД, но отличается тем, что пользователи могут видеть как сами таблицы, так и пути доступа к ним.

Строки таблиц упорядочены в некоторой физической последовательности. Физический порядок строк может быть определен как для каждой

отдельной таблицы, так и для всей базы данных в целом. Для каждой таблицы можно определить произвольное количество ключей поиска, для которых автоматически создаются индексы. Эти индексы поддерживаются системой и могут быть видны пользователям.

В этой модели поддерживаются два класса операторов: операторы, устанавливающие адрес записи и операторы, находящие запись относительно предыдущей записи по некоторому пути доступа.

Иерархические модели

Иерархическая модель БД состоит из объектов с указателями от родительских объектов к потомкам, соединяя вместе связанную информацию. Иерархические БД могут быть представлены как дерево.

На самом высшем (первом) уровне иерархии находится только одна вершина, которая называется корнем дерева. Эта вершина имеет связи с вершинами второго уровня, вершины второго уровня имеют связи с вершинами третьего уровня и т.д. Связи между вершинами одного уровня отсутствуют. Следовательно, данные в иерархической структуре не равноправны – одни жестко подчинены другим. Доступ к информации возможен только по вертикальной схеме, начиная с корня, так как каждый элемент связан только с одним элементом на верхнем уровне и с одним или несколькими на низком. Примером иерархической структуры может служить книга, как иерархическая последовательность букв, которые объединяются в слова, слова – в предложения, предложения – в параграфы, затем в главы и т.д

Сетевые модели

Дальнейшим развитием иерархической модели является сетевая. Сетевая модель – это структура, у которой любой элемент может быть связан с любым другим элементом[4]. К основным понятиям сетевой модели БД относятся: элемент (узел), связь. Узел – это совокупность атрибутов данных, описывающих некоторый объект. Сетевые БД могут быть представлены в виде графа. В сетевой БД логика процедуры выборки данных зависит от физической организации этих данных. Поэтому эта модель не является полностью независимой от приложения. Другими словами, если необходимо изменить структуру данных, то нужно изменить и приложение.

1.4.2 Реляционные модели

Реляционная модель данных включает следующие компоненты:

– Структурный (данные в базе данных представляют собой набор отношений);

– Целостностный (отношения (таблицы) отвечают определенным условиям целостности);

– Манипуляционный (манипулирования отношениями осуществляется средствами реляционной алгебры и/или реляционного исчисления). Кроме того, в состав реляционной модели данных включают теорию нормализации.

Реляционная модель данных является совокупностью данных и состоит из набора двумерных таблиц. При табличной организации отсутствует иерархия элементов. Таблицы состоят из строк – записей и столбцов – полей. На пересечении строк и столбцов находятся конкретные значения. Для каждого поля определяется множество его значений.

1.4.3 Постреляционные модели

Постреляционная модель является расширением реляционной модели[5]. Она снимает ограничение неделимости данных, допуская многозначные поля, значения которых состоят из подзначений, и набор значений воспринимается как самостоятельная таблица, встроенная в главную таблицу.

Достоинством постреляционной модели является возможность представления совокупности связанных реляционных таблиц одной постреляционной таблицей. Это обеспечивает высокую наглядность представления информации и повышение эффективности ее обработки[6].

Вывод

В данном разделе приводится анализ предметной области и существующих решений, формализуется и описывается информация, подлежащая хранению в проектируемой БД, и пользователи. На основе приведенного анализа строятся диаграмма вариантов использования и ER-диаграмма сущностей в нотации Чена.

2 Конструкторский раздел

В данном разделе приведено описание сущностей и диаграмма проектируемой базы данных, описание проектируемых ограничений целостности базы данных, описание проектируемых триггеров и описание проектируемой ролевой модели.

2.1 Описание сущностей базы данных

Анализируя таблицу 1.2 и диаграмму сущностей базы данных (рисунок 1.1) формализуем сущности проектируемой базы данных. В таблицах 2.1 - 2.7 приведена формализация полей выделенных сущностей, указаны их типы и ограничения.

На рисунке 2.1 приведена ER-диаграмма проектируемой базы данных.

поле	**тип**	**ограничения**	**описание**
id	уникальный идентификатор	not null, primary key	идентификатор трактора
model	строка	not null	название модели
releaseyear	дата	not null	год создания модели
enginetype	строка	not null	тип двигателя
enginemodel	строка	not null	модель двигателя
enginepower	целое число	not null	мощность двигателя в лошадиных силах
fronttiresize	целое число	not null	диаметр передних колес в дюймах
backtiresize	целое число	not null	диаметр задних колес в дюймах
wheelsamount	целое число	not null	количество колес
tankcapacity	целое число	not null	вместимость бензобака в литрах
ecologicalstandart	строка	not null	эклолгический стандарт
length	вещественное число	not null	длина в метрах
width	вещественное число	not null	ширина в метрах
cabinheight	вещественное число	not null	высота по кабине в метрах

Таблица 2.1 – Формализация полей таблицы тракторов

поле	**тип**	**ограничения**	**описание**
id	уникальный идентификатор	not null, primary key	идентификатор производственной линии
name	строка	not null	название линии
length	целое число	not null	длина в мм
width	целое число	not null	ширина в мм
height	целое число	not null	высота в мм
status	строка	not null	статус
production	целое число	not null	количество произведенных тракторов в месяц
downtime	целое число	not null	время простоя в минутах
inspectionsamount	целое число	not null	требуемое количество техосмотров в год
lastinspection	дата	not null	дата последнего техосмотра
nextinspection	дата	not null	дата следующего техосмотра
defectrate	целое число	not null	процент брака

Таблица 2.2 – Формализация полей таблицы производственных линий

поле	**тип**	**ограничения**	**описание**
id	уникальный идентификатор	not null, primary key	идентификатор пользователя
name	строка	not null	имя
surname	строка	not null	фамилия
fathername	строка	not null	отчество
department	строка	not null	отдел
email	строка	not null	почта
password	хэшированная стрка	not null	пароль
dateofbirth	дата	not null	дата рождения
sex	строка	not null	пол
role	строка	not null	должность

Таблица 2.3 – Формализация полей таблицы сотрудников

поле	**тип**	**ограничения**	**описание**
id	уникальный идентификатор	not null, primary key	идентификатор детали
name	строка	not null	название
country	строка	not null	страна производитель
amount	целое число	not null	количество на складе
price	вещественное число	not null	цена в рублях
length	целое число	not null	длина в мм
width	целое число	not null	ширина в мм
height	целое число	not null	высота в мм

Таблица 2.4 – Формализация полей таблицы деталей

поле	**тип**	**ограничения**	**описание**
id	уникальный идентификатор	not null, primary key	идентификатор заказа деталей
userid	уникальный идентификатор	not null, foreign key	идентификатор заказчика
requestid	уникальный идентификатор	not null, foreign key	идентификатор заявки для которой заказывают детали
status	строка	not null	статус заказа
totalprice	вещественное число	not null	цена
orderdate	дата	not null	дата заказа

Таблица 2.5 – Формализация полей таблицы заказа деталей

поле	**тип**	**ограничения**	**описание**
id	уникальный идентификатор	not null, primary key	идентификатор заявки на обслуживание
lineid	уникальный идентификатор	not null, foreign key	идентификатор производственной линии
userid	уникальный идентификатор	not null, foreign key	идентификатор сотрудника
requestdate	дата	not null	дата заказа
status	строка	not null	статус
type	строка	not null	статус
description	текст	not null	описание заявки

Таблица 2.6 – Формализация полей таблицы заявки на обслуживание

поле	**тип**	**ограничения**	**описание**
id	уникальный идентификатор	not null, primary key	идентификатор отчета об обслуживании
lineid	уникальный идентификатор	not null, foreign key	идентификатор производственной линии
userid	уникальный идентификатор	not null, foreign key	идентификатор сотрудника
requestid	уникальный идентификатор	not null, foreign key	идентификатор заявки
opendate	дата и время	not null	дата начала обслуживания
closedate	дата и время	not null	дата завеншения обслуживания
totalprice	вещественное число	not null	цена ремонта
description	текст	not null	описание

Таблица 2.7 – Формализация полей таблицы отчета об обслуживании

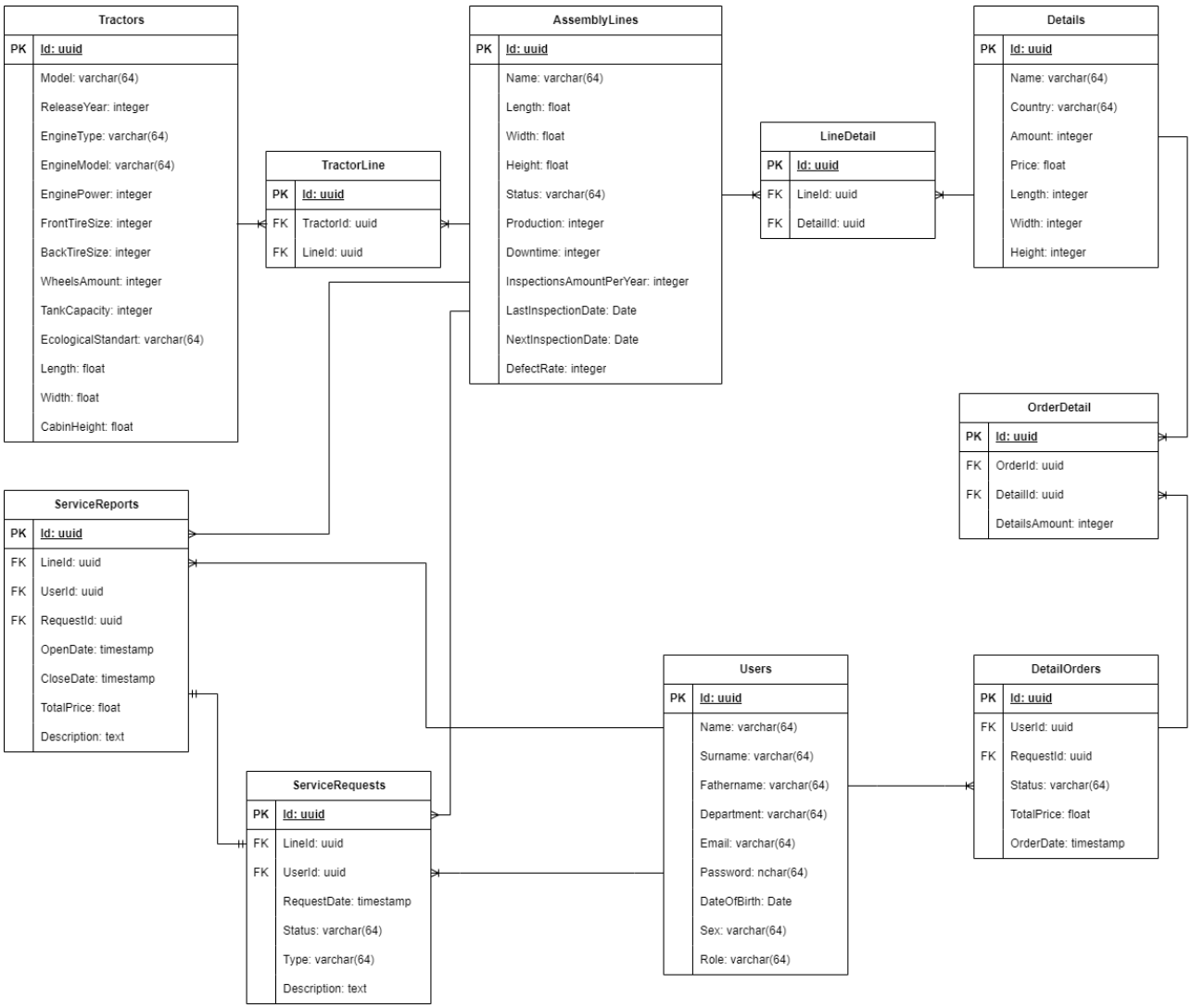

Рисунок 2.1 – ER-диаграмма проектируемой базы данных

2.2 Описание проектируемых триггеров

В проектируемой базе данных необходимо реализовать триггеры обновляющие значения некоторых полей после добавления новых записей в таблицы.

На рисунке 2.2 приведен алгоритм триггера, обновляющего статус заявки на обслуживание после добавления записи в таблицу отчетов об обслуживании. Таким образом после публикации отчета об обслуживании статус проработанной заявки автоматически изменится на «завершена».

На рисунке 2.3 приведен алгоритм триггера, обновляющего даты последнего и следующего техобслуживания, время простоя и статус производственной линии. При публикации отчета об обслуживании, будет вычислено количество часов прошедших с момента остановки работы производственной линии и это

значение будет добавлено к полю DownTime, статус линии при этом будет изменен на «работает». Если заявка на обслуживание, на основе которой был составлен отчет, имела тип «техосмотр», то публикация отчета будет означать, что техосмотр был проведен, а значит необходимо занести информацию об этом в таблицу производственной линии. Так, после завершения техосмотра, его дата будет записана в поле LastInspectionDate и на основе данных о требуемой частоте техосмотров линии будет рассчитана дата следующего техосмотра.

На рисунке 2.4 приведен алгоритм триггера, обновляющего статус производственной линии при создании заявки на ее обслуживание. После публикации заявки на обслуживание статус производственной линии автоматически изменится на «на обслуживании».

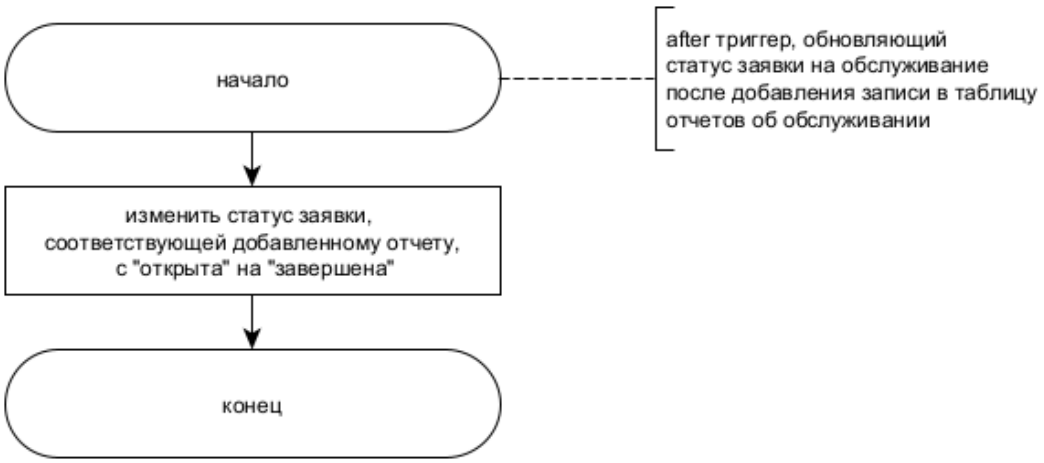

Рисунок 2.2 – Алгоритм триггера, обновляющего статус заявки на обслуживание

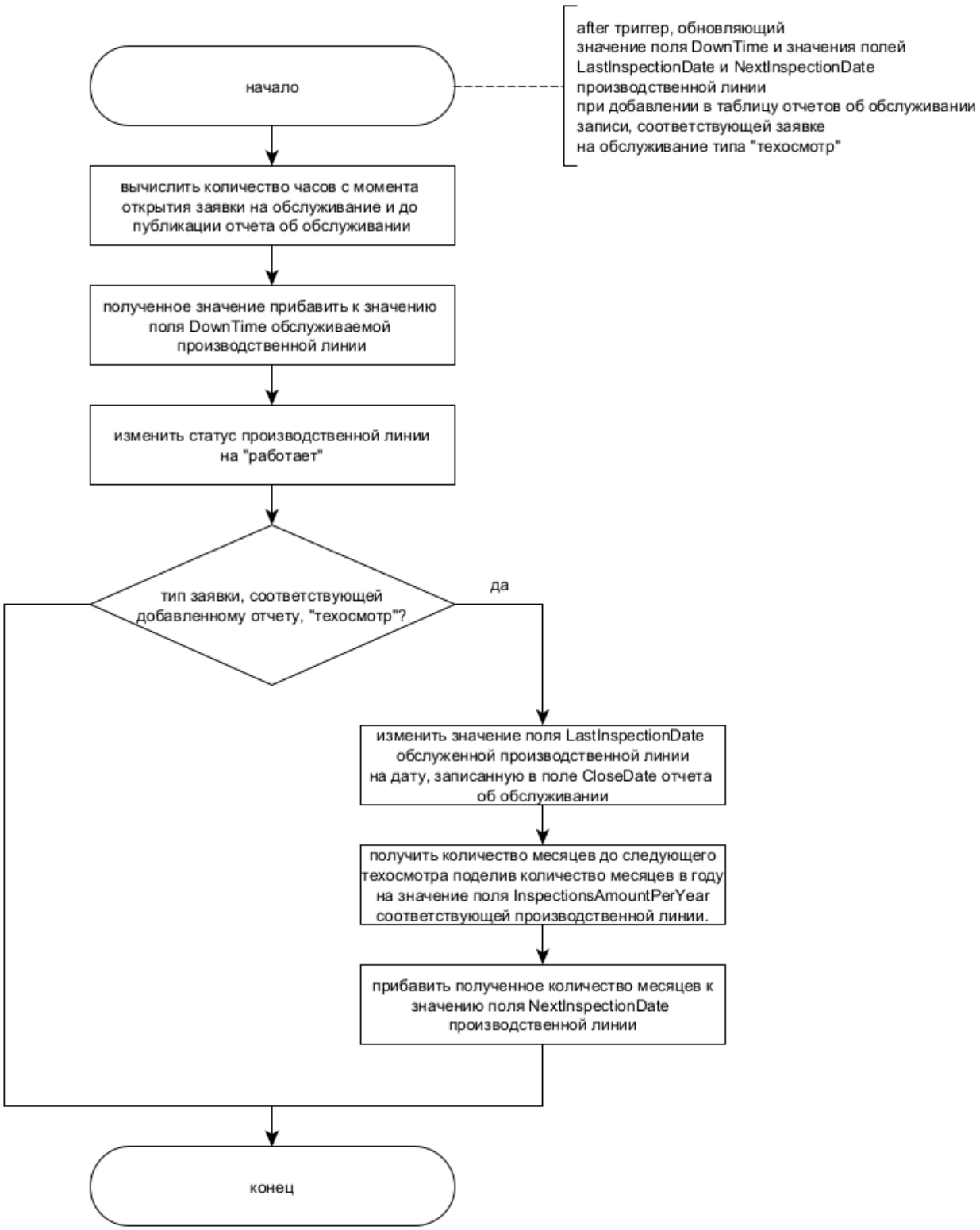

Рисунок 2.3 – Алгоритм триггера, обновляющего даты последнего и следующего техобслуживания, времени простоя и статуса производственной линии

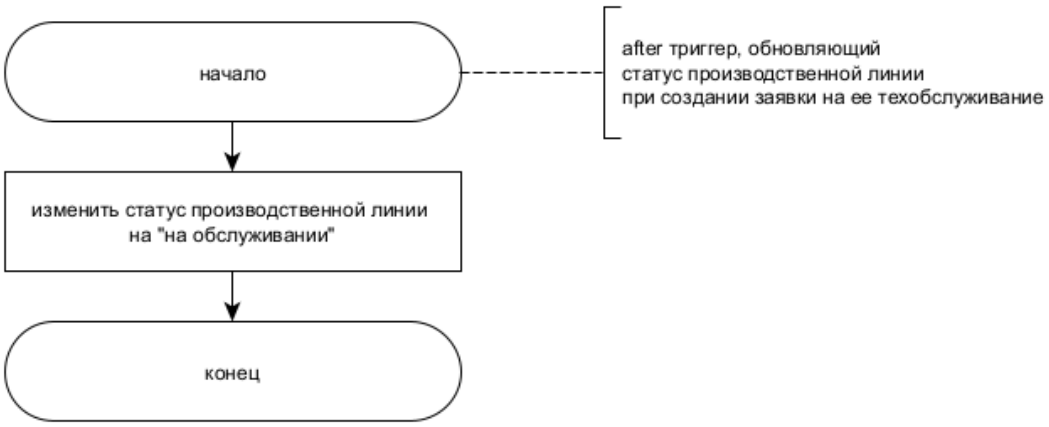

Рисунок 2.4 – Алгоритм триггера, обновляющего статус производственной линии

2.3 Описание проектируемой ролевой модели

Для эффективного и безопасного взаимодействия пользователей с базой данных была разработана ролевая модель, определяющая функционал доступный каждому типу пользователей.

В таблице 2.8 описан уровень доступа для каждого типа пользователя. Символом «(*)» обозначим все таблицы базы данных. Таким образом у администратора есть полный доступ ко всем таблицам. Символом «—» обозначим отсутствие таблиц для данного типа доступа.

	SELECT	**UPDATE**	**INSERT**
Оператор производства	Tractors, AssemblyLines, ServiceReports, ServiceRequests	AssemblyLines, ServiceReqests, Users	ServiceRequests
Специалист по обслуживанию	Tractors, AssemblyLines, ServiceRequests, ServiceReports, Details, DetailOrders	DetailOrders, ServiceReports, Users	ServiceReports, DetailOrders
Администратор	(*)	(*)	(*)
Неавторизованный пользователь	—	—	Users

Таблица 2.8 – Описание проектируемой ролевой модели

Вывод

В данном разделе была проведена формализация сущностей и построена диаграмма проектируемой базы данных, спроектирована ролевая модель и триггеры.

3 Технологический раздел

В данной части будут приведены выбор средств реализации базы данных и приложения, листинги создания базы данных, триггеров и ролевой модели, рассмотрен интерфейс и примеры работы приложения.

3.1 Выбор средств реализации

На данный момент самыми широко используемыми СУБД являются[7]:

– Oracle[8];

– MySQL[9];

– Microsoft SQL Server[10];

– PostgreSQL[11].

3.2 Создание базы данных

На листинге 3.1 показан sql скрипт для создания проектируемой базы данных. На листингах 3.2 – 3.11 показано создание таблиц базы данных.

Листинг 3.1 – Создание базы данных

```
CREATE DATABASE "ProductMonitor"
    WITH
    OWNER = m4ks0n
    ENCODING = 'UTF8'
    CONNECTION LIMIT = -1
    IS_TEMPLATE = False;
```

Листинг 3.2 – Создание таблицы тракторов

```
drop table if EXISTS tractors CASCADE;
create table if not exists tractors
(
    id uuid,
    model VARCHAR(64),
    release_year INT,
    enginetype VARCHAR(64),
    enginemodel VARCHAR(64),
    enginepower INT,
    fronttiresize INT,
    backtiresize INT,
```

```
    wheelsamount INT,
    tankcapacity INT,
    ecologicalstandart VARCHAR(64),
    length FLOAT,
    width FLOAT,
    cabinheight FLOAT
);
```

Листинг 3.3 – Создание таблицы производственныйх линий

```
drop table if exists assemblylines CASCADE;
create table if not exists assemblylines
(
    id uuid,
    name VARCHAR(64),
    length FLOAT,
    height FLOAT,
    width FLOAT,
    status VARCHAR(64),
    production INT,
    downtime INT,
    inspectionsamountperyear INT,
    lastinspectiondate date,
    nextinspectiondate date,
    defectrate int
);
```

Листинг 3.4 – Создание таблицы tractor_line

```
drop table if exists tractor_line CASCADE;
create table if not exists tractor_line
(
    tractorid uuid,
    lineid uuid
);
```

Листинг 3.5 – Создание таблицы деталей

```
drop table if exists details CASCADE;
create table if not exists details
(
    id uuid,
    name VARCHAR(64),
    country VARCHAR(64),
    amount INT,
```

```
    price FLOAT,
    length INT,
    height INT,
    width INT
);
```

Листинг 3.6 – Создание таблицы line_detail

```
drop table if exists line_detail CASCADE;
create table if not exists line_detail
(
    lineid uuid,
    detailid uuid
);
```

Листинг 3.7 – Создание таблицы пользователей

```
drop table if exists users CASCADE;
create table if not exists users
(
    id uuid,
    name VARCHAR(64),
    surname VARCHAR(64),
    fatherame VARCHAR(64),
    department VARCHAR(64),
    email VARCHAR(64),
    password nchar(64),
    dateofbirth date,
    sex VARCHAR(64),
    role VARCHAR(64)
);
```

Листинг 3.8 – Создание таблицы заказов деталей

```
drop table if exists detailorders CASCADE;
create table if not exists detailorders
(
    id uuid,
    userid uuid,
    requestid uuid,
    status VARCHAR(64),
    totalprice float,
    orderdate timestamp
);
```

Листинг 3.9 – Создание таблицы order_detail

```
drop table if exists order_detail CASCADE;
create table if not exists order_detail
(
    orderid uuid,
    detailid uuid,
    detailsamount int
);
```

Листинг 3.10 – Создание таблицы заявок на обслуживание

```
drop table if exists servicerequests CASCADE;
create table if not exists servicerequests
(
    id uuid,
    lineid uuid,
    userid uuid,
    requestdate timestamp,
    status VARCHAR(64),
    type VARCHAR(64),
    description text
);
```

Листинг 3.11 – Создание таблицы отчетов об обслуживании

```
drop table if exists servicereports CASCADE;
create table if not exists servicereports
(
    id uuid,
    lineid uuid,
    userid uuid,
    requestid uuid,
    opendate timestamp,
    closedate timestamp,
    totalprice float,
    description text
);
```

Листинг 3.12 – Ограничения таблицы тракторов

```
alter table tractors
    add CONSTRAINT pk_tractors_is PRIMARY KEY(id),
    alter column id set not null,
    alter column model set not null,
    alter column release_year set not null,
```

```
    alter column enginetype set not null,
    alter column enginetype set not null,
    alter column enginepower set not null,
    alter column fronttiresize set not null,
    alter column backtiresize set not null,
    alter column wheelsamount set not null,
    alter column tankcapacity set not null,
    alter column ecologicalstandart set not null,
    alter column length set not null,
    alter column cabinheight set not null,
    alter column width set not null;
```

Листинг 3.13 – Ограничения таблицы тракторов

```
alter table assemblylines
    add CONSTRAINT pk_assembly_line PRIMARY key(id),
    alter column id set not null,
    alter column name set not null,
    alter column length set not null,
    alter column width set not null,
    alter column height set not null,
    alter column status set not null,
    alter column production set not null,
    alter column downtime set not null,
    alter column inspectionsamountperyear set not null,
    alter column lastinspectiondate set not null,
    alter column nextinspectiondate set not null,
    alter column defectrate set not null;
```

Листинг 3.14 – Ограничения таблицы тракторов

```
alter table users
    add CONSTRAINT pk_usersid PRIMARY key(id),
    alter column id set not null,
    alter column name set not null,
    alter column surname set not null,
    alter column fatherame set not null,
    alter column department set not null,
    alter column email set not null,
    alter column password set not null,
    alter column dateofbirth set not null,
    alter column sex set not null,
    alter column role set not null;
```

Листинг 3.15 – Ограничения таблицы тракторов

```
alter table details
    add CONSTRAINT pk_details PRIMARY key(id),
    alter column id set not null,
    alter column name set not null,
    alter column country set not null,
    alter column amount set not null,
    alter column price set not null,
    alter column length set not null,
    alter column width set not null,
    alter column height set not null;
```

Листинг 3.16 – Ограничения таблицы тракторов

```
alter table servicerequests
    add CONSTRAINT pk_requestid PRIMARY key(id),
    alter column id set not null,
    add CONSTRAINT fk_lineid_servicerequests FOREIGN key
       (lineid) REFERENCES assemblylines(id) on DELETE CASCADE,
    add CONSTRAINT fk_userid_servicerequests FOREIGN key
       (userid) REFERENCES users(id) on delete cascade,
    alter column lineid set not null,
    alter column userid set not null,
    alter column requestdate set not null,
    alter column status set not null,
    alter column type set not null,
    alter column description set not null;
```

Листинг 3.17 – Ограничения таблицы тракторов

```
alter table detailorders
    add CONSTRAINT pk_detaleorders PRIMARY key(id),
    alter column id set not null,
    add CONSTRAINT fk_userid_detailorders FOREIGN key (userid)
       REFERENCES users(id) on delete cascade,
    add CONSTRAINT fk_requestid_detailorders FOREIGN key
       (requestid) REFERENCES servicerequests(id) on delete
       cascade,
    alter column userid set not null,
    alter column requestid set not null,
    alter column status set not null,
    alter column totalprice set not null,
    alter column orderdate set not null;
```

Листинг 3.18 – Ограничения таблицы тракторов

```
alter table servicereports
    add CONSTRAINT pk_reporttid PRIMARY key(id),
    alter column id set not null,
    add CONSTRAINT fk_lineid_servicereports FOREIGN key (lineid)
       REFERENCES assemblylines(id) on DELETE CASCADE,
    add CONSTRAINT fk_userid_servicereports FOREIGN key (userid)
       REFERENCES users(id) on delete cascade,
    add CONSTRAINT fk_requestid_servicerequests FOREIGN key
       (requestid) REFERENCES servicerequests(id) on delete
       cascade,
    alter column lineid set not null,
    alter column userid set not null,
    alter column requestid set not null,
    alter column opendate set not null,
    alter column closedate set not null,
    alter column totalprice set not null,
    alter column description set not null;
```

Листинг 3.19 – Ограничения таблицы тракторов

```
alter table order_detail
    add CONSTRAINT fk_orderid_order_detail FOREIGN key (orderid)
       REFERENCES detailorders(id) on DELETE CASCADE,
    add CONSTRAINT fk_detailid_order_detail FOREIGN key
       (detailid) REFERENCES details(id) on DELETE CASCADE,
    alter column orderid set not null,
    alter column detailid set not null,
    alter column detailsamount set not null,
    add constraint pk_order_detail primary key(orderid,
       detailid);
```

Листинг 3.20 – Ограничения таблицы тракторов

```
alter table tractor_line
    add CONSTRAINT fk_tractorid_tractor_line FOREIGN key
       (tractorid) REFERENCES tractors(id) on DELETE CASCADE,
    add CONSTRAINT fk_lineid_tractorline FOREIGN key (lineid)
       REFERENCES assemblylines(id) on DELETE CASCADE,
    add constraint pk_tractor_line primary key(tractorid,
       lineid);
```

Листинг 3.21 – Ограничения таблицы тракторов

```
alter table line_detail
```

```
    add CONSTRAINT fk_lineid_line_detail FOREIGN key (lineid)
       REFERENCES assemblylines(id) on DELETE CASCADE,
    add CONSTRAINT fk_detaleid_line_detail FOREIGN key
       (detailid) REFERENCES details(id) on delete cascade,
    add constraint pk_line_detail primary key(lineid, detailid);
```

3.3 Реализация триггеров

Листинг 3.22 – Триггер, обновляющий статус заявки на обслуживание

```
CREATE OR REPLACE FUNCTION update_service_request_status()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE servicerequests
    SET status = 'завершена'
    WHERE id = NEW.requestid;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER update_service_request_status_trigger
AFTER INSERT ON servicereports
FOR EACH ROW
EXECUTE FUNCTION update_service_request_status();
```

Листинг 3.23 – Триггер, обновляющий статус производственной линии при отправлении на обслуживание

```
CREATE OR REPLACE FUNCTION update_line_status()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE assemblylines
    SET status = 'на обслуживании'
    WHERE  id = NEW.lineid;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER update_line_status_trigger
AFTER INSERT ON servicerequests
FOR EACH ROW
```

```
15 EXECUTE FUNCTION update_line_status();
```

Листинг 3.24 – Триггер, обновляющий даты последнего и следующего техобслуживания, времени простоя и статуса производственной линии

```
1  CREATE OR REPLACE FUNCTION update_assemblyline_after_report()
2  RETURNS TRIGGER AS $$
3  DECLARE
4      downtime_hours INTEGER;
5      next_inspection_date DATE;
6  BEGIN
7      downtime_hours := EXTRACT(EPOCH FROM (NEW.closedate -
          (SELECT requestdate FROM servicerequests WHERE id =
          NEW.requestid))) / 3600;
8
9      UPDATE assemblylines
10     SET downtime = downtime + downtime_hours,
11         status = 'работает'
12     WHERE id = NEW.lineid;
13
14     IF (SELECT type FROM servicerequests WHERE id =
          NEW.requestid) = 'техосмотр' THEN
15         UPDATE assemblylines
16         SET lastinspectiondate = NEW.closedate,
17             nextinspectiondate = NEW.closedate + interval '1
                  month' * (12 / (SELECT inspectionsamountperyear
                  FROM assemblylines WHERE id = NEW.lineid))
18         WHERE id = NEW.lineid;
19     END IF;
20
21     RETURN NEW;
22 END;
23 $$ LANGUAGE plpgsql;
24
25 CREATE TRIGGER update_assemblyline_trigger
26 AFTER INSERT ON servicereports
27 FOR EACH ROW
28 EXECUTE FUNCTION update_assemblyline_after_report();
```

3.4 Реализация ролевой модели

3.5 Интерфейс

Вывод

4 Исследовательский раздел

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Дейт К. Д.* Введение в системы баз данных. — 2001.
2. *Роб П.*, *Коронел К.* Системы баз данных: проектирование, реализация и управление. — 2004.
3. *Кузнецов С.* Основы современных баз данных // М: Центр Информационных Технологий. — 1998.
4. Модели организации баз данных [Электронный ресурс]. — — Режим доступа: `https://intuit.ru/studies/courses/3439/681/lecture/` (дата обращения: 30.04.2024).
5. Лекция на тему МОДЕЛИ ДАННЫХ [Электронный ресурс]. — — Режим доступа: `https://bseu.by/it/tohod/` (дата обращения: 30.04.2024).
6. *Парфенов Ю. П.* Постреляционные хранилища данных. — 2016.
7. DB-Engines Ranking [Электронный ресурс]. — — Режим доступа: `https://db-engines.com/en/ranking` (дата обращения: 09.05.2024).
8. Oracle Россия и СНГ [Электронный ресурс]. — — Режим доступа: `https://www.oracle.com/cis/database/` (дата обращения: 09.05.2024).
9. MySQL [Электронный ресурс]. — — Режим доступа: `https://www.mysql.com/` (дата обращения: 09.05.2024).
10. Microsoft SQL server [Электронный ресурс]. — — Режим доступа: `https://www.microsoft.com/ru-ru/sql-server/sql-server-2019` (дата обращения: 09.05.2024).
11. PostgreSQL [Электронный ресурс]. — — Режим доступа: `https://www.postgresql.org/` (дата обращения: 09.05.2024).

ПРИЛОЖЕНИЕ А