**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc120179548)

[1 ПОСТРОЕНИЕ ИНФОЛОГИЧЕСКОЙ КОНЦЕПТУАЛЬНОЙ МОДЕЛИ 6](#_Toc120179549)

[1.1 Анализ предметной области и выявление необходимого набора сущностей 6](#_Toc120179550)

[1.2 Обоснование требуемого набора атрибутов для каждой сущности и выделение идентифицирующих атрибутов 7](#_Toc120179551)

[1.3 Определение связей между объектами 10](#_Toc120179552)

[1.4 Описание полученной модели на языке инфологического проектирования 11](#_Toc120179553)

[2 ПОСТРОЕНИЕ СХЕМЫ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ 12](#_Toc120179554)

[2.1 Построение набора необходимых отношений базы данных 12](#_Toc120179555)

[2.2 Задание первичных и внешних ключей определенных отношений 13](#_Toc120179556)

[2.3 Третья нормальная форма 14](#_Toc120179557)

[2.4 Определение ограничений целостности для внешних ключей отношений и для отношений в целом 14](#_Toc120179558)

[2.5 Графическое представление связей между внешними ключами 15](#_Toc120179559)

[3 СОЗДАНИЕ СПРОЕКТИРОВАННОЙ БАЗЫ ДАННЫХ 16](#_Toc120179560)

[4 ЗАПИСЬ ВЫРАЖЕНИЙ, УКАЗАННЫХ В ВАРИАНТЕ ЗАДАНИЯ ТИПОВ ЗАПРОСОВ НА ЯЗЫКЕ SQL 20](#_Toc120179561)

[5 ВЫБОР И ОСНОВАНИЕ СРЕДСТВ РАЗРАБОТКИ ПРИЛОЖЕНИЯ 23](#_Toc120179562)

[6 РЕАЛИЗАЦИЯ ЗАКОНЧЕННОГО ПРИЛОЖЕНИЯ, РАБОТАЮЩЕГО С СОЗДАННОЙ БАЗОЙ ДАННЫХ 25](#_Toc120179563)

[6.1 Разработка и построение интерфейса главной и рабочих форм 25](#_Toc120179564)

[6.2 Построение главного меню и кнопок панели инструментов 25](#_Toc120179565)

[6.3 Выполнение программного кода на языке C# 26](#_Toc120179566)

[ЗАКЛЮЧЕНИЕ 28](#_Toc120179567)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 29](#_Toc120179568)

[ПРИЛОЖЕНИЕ А (обязательное) Концептуальная схема БД 30](#_Toc120179569)

[ПРИЛОЖЕНИЕ Б (обязательное) Схема реляционной базы данных 31](#_Toc120179570)

[ПРИЛОЖЕНИЕ В (обязательное) Описание задания курсовой работы 32](#_Toc120179571)

[ПРИЛОЖЕНИЕ Г (обязательное) Главная и рабочие формы приложения 34](#_Toc120179572)

# ВВЕДЕНИЕ

Любая современная организация не может обойтись без базы данных. Это учебные заведения, банки, магазины, заводы, любые предприятия и государственные учреждения. Они используют их для перевода данных в электронный вид и объединения данных, а также оперативного доступа к ним. Это позволяет экономить время и средства на затраты.

Конечно, снижение времени является лишь побочным эффектом автоматизации. Самая главная задача развития информационных технологий в совсем другом – в приобретении той или иной организацией исключительно новых качеств, придающих ей существенную конкурентоспособность. А это дорогого стоит.

К тому же, сейчас установка и управление базы данных не является таким уж и трудным процессом, как это было десятилетие назад. Когда проектировка и управление базами данных были не автоматизированы. Система управления базой данных позволяет создавать базу данных, обновляя в ней хранимую информацию, обеспечивая оперативный доступ к ней для просмотра и поиска информации.

Вся современная экономика базируется на управлении информацией. Данные решают все, и очень важно эффективно их обрабатывать. Область применения БД и СУБД для решения различных экономических задач очень обширна. Сегодня организации всё больше осознают необходимость в интегрировании всех типов информации – от офисных документов и электронных таблиц до, например, медицинских изображений или географических данных – в повседневные бизнес-процессы. Единое размещение и защита всех типов информации помогает значительно сократить затраты на устройства хранения.

В наше время огромное количество фирм используют персональные компьютеры для сохранения и обработки любого вида информации. Эта информация содержится в базах данных. Базы данных играют важную роль в развивающемся мире технологий. Всё, с чем мы каждый день взаимодействуем в жизни, по всей видимости, зафиксировано в какой-нибудь базе. Работа с базами данных является важнейшим навыком в работе с компьютером, а специалисты данной области становятся всё более востребованными. Главные идеи нынешней информационной методики базируются на представлении, в соответствии чему информация должна быть образована в базы данных с задачей отображения динамически изменяющегося мира и удовлетворения всех потребностей в информации у пользователей. Базы данных формируются и работают под управлением специальных программных средств, называемых системами управления базами данных.

База данных, которая представлена в объективной форме, это совокупность следующих материалов: статей, счетов, нормативных актов, судебных решений или иных подобных материалов, собранные вместе таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины.

База данных – это организованная структура, которая предназначена для хранения информации. В то время, когда происходило развитие термина баз данных, в них сохранялись исключительно информация, однако уже в наши дни многие системы управления базами данных позволяет размещать в своих структурах и данные, и программный код, с помощью которого совершается связь с пользователями или с другими программно-аппаратными комплексами [1]. При этом данные должны не противоречить друг другу, быть целостными и не избыточными. База данных создается для сохранения и непосредственного доступа к информации, содержащей сведения об искомой предметной области. Степень конкретизации данных обуславливается группой факторов. Прежде всего, целью использования информации из баз данных и сложностью информационных процессов, существующих в пределах предметной области в конкретных условиях.

Система управления базами данных – это программный механизм, предназначенный для записи, поиска, сортировки, обработки и печати информации, содержащейся в базе данных [2].

В компьютере данные базы данных представляется в виде таблицы, схожей на электронную таблицу. Названия столбцов, представляющих заголовки таблицы, называют именами полей, а сами столбцы – полями. Данные, которые находятся в полях, называют значениями полей.

Создание современного программного обеспечения – это весьма трудоемкий процесс, требующий от специалиста представлений о методах анализа, проектирования, реализации и тестирования программного продукта.

В данной курсовой работе поставлена задача базы данных и создания информационной системы железнодорожной пассажирской станции.

Для создания информационной базы данных будет использоваться СУДБ SQL Server. Для создания приложения – среда Visual Studio 2022.

# 1 ПОСТРОЕНИЕ ИНФОЛОГИЧЕСКОЙ КОНЦЕПТУАЛЬНОЙ

**МОДЕЛИ**

# 1.1 Анализ предметной области и выявление необходимого набора

**сущностей**

В качестве идеи для проектирования и разработки была выбрана тема «Информационная система железнодорожной пассажирской станции», которая содержит следующее описание исходя из выданного задания к курсовой работе: работников железнодорожной станции можно подразделить на водителей подвижного состава, диспетчеров, ремонтников подвижного состава, путей, кассиров, работников службы подготовки составов, справочной службы и других, которые административно относятся каждый к своему отделу. Каждая из перечисленных категорий работников имеет уникальные атрибуты-характеристики, определяемые профессиональной направленностью. В отделах существует разбиение работников на бригады. Отделы возглавляются начальниками, которые представляют собой администрацию железнодорожной станции. В функции администрации входит планирование маршрутов, составление расписаний, формирование кадрового состава железнодорожной станции. За каждым локомотивом закрепляется локомотивная бригада. За несколькими локомотивами закрепляется бригада техников-ремонтников, выполняющая рейсовый и плановый техосмотр (по определенному графику), ремонт, техническое обслуживание. Водители локомотивов обязаны проходить каждый год медосмотр, не прошедших медосмотр необходимо перевести на другую работу. Локомотив должен своевременно осматриваться техниками-ремонтниками и при необходимости ремонтироваться. Подготовка к рейсу включает в себя техническую часть (рейсовый техосмотр, мелкий ремонт) и обслуживающую часть (уборка вагонов, запас продуктов питания и тому подобное).

В расписании указывается тип поезда (скорый, пассажирский), номер поезда, дни и время отправления и прибытия, маршрут (начальный и конечный пункты назначения, основные узловые станции), стоимость билета. Билеты на поезд можно приобрести заранее или забронировать в железнодорожных кассах.

До отправления поезда, если есть необходимость, билет можно вернуть. Отправление поездов может быть задержано из-за опозданий поездов, погодных условий, технических неполадок.

Железнодорожные маршруты можно разделить на следующие категории: внутренние, международные, туристические, специальные маршруты. Пассажиры могут сдавать свои вещи в багажное отделение.

# 1.2 Обоснование требуемого набора атрибутов для каждой сущности и

**выделение идентифицирующих атрибутов**

Определим нужный набор атрибутов для каждой сущности с целью построения инфологической концептуальной модели. Концептуальная модель данных (КМД) – это общая информационная модель предметной области, охватывающая вопросы классификации, структуризации и семантической целостности (достоверности и согласованности данных).

Атрибутом является поименованная характеристика сущности. Наименования должны быть уникальными для конкретного типа сущности и не совпадать с системными словами, такими как ключевые слова запросов (select, from), но могут быть одинаковыми для различного типа сущностей. Атрибуты используются для определения того, какая информация должна быть собрана о сущности.

В таблице 1.1 представлены сущности, определенные для них атрибуты, описание атрибутов и ключи.

Таблица 1.1 – Описание сущностей

| Таблица | Поле | Ключ | Описание |
| --- | --- | --- | --- |
| Speciality | IdSpeciality | PK | Идентификационный номер таблицы Speciality |
| SalaryOneRate |  | Зарплата за одну ставку |
| Name |  | Наименование специальности |
| Description |  | Описание специальности |
| TypeDepartment | IdTypeDepartment | PK | Идентификационный номер таблицы TypeDepartment |
| NameType |  | Название типа отдела |
| Department | IdDepartment | PK | Идентификационный номер таблицы Department |
| Name |  | Название отдела |
| Description |  | Описание отдела |
| CountBrigade |  | Количество бригад в отделе |
| IdTypeDepartment |  | Идентификационный номер таблицы TypeDepartment |
| TypeEngine | IdTypeEngine | PK | Идентификационный номер таблицы TypeEngine |
| NameType |  | Название типа двигателя |
| Engine | IdEngine | PK | Идентификационный номер таблицы Engine |
| Name |  | Название двигателя |
| Efficiency |  | КПД |

Продолжение таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица | Поле | Ключ | Описание |
|  | NumberOfCylinders |  | Количество цилиндров |
| IdTypeEngine | FK | Идентификационный номер таблицы TypeEngine |
| Locomative | IdLocomative | PK | Идентификационный номер таблицы Locomative |
| Name |  | Название локоматива |
| Weight |  | Вес |
| TractionForce |  | Сила тяги |
| InspectionDate |  | Дата тех. обслуживания |
| IdEngine | FK | Идентификационный номер таблицы Engine |
| Brigade | IdBrigade | PK | Идентификационный номер таблицы Brigade |
| CountWorker |  | Количество работников в бригаде |
| NumberInDepartment |  | Номер бригады в отделе |
| IdDepartment | FK | Идентификационный номер таблицы Department |
| IdLocomative | FK | Идентификационный номер таблицы Locomative |
| Worker | IdWorker | PK | Идентификационный номер таблицы Worker |
| Name |  | Имя работника |
| Surname |  | Фамилия работника |
| Email |  | Email работника |
| Rate |  | Тарифная ставка работника |
| IdSpeciality | FK | Идентификационный номер таблицы Speciality |
| IdBrigade | FK | Идентификационный номер таблицы Brigade |
| TypeTrain | IdTypeTrain | PK | Идентификационный номер таблицы TypeTrain |
| NameType |  | Название типа поезда |
| Train | IdTrain | PK | Идентификационный номер таблицы Train |
| Name |  | Имя поезда |
| NumberOfTrainCarriage |  | Количество вагонов |

Продолжение таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица | Поле | Ключ | Описание |
|  | IdLocomative | FK | Идентификационный номер таблицы Locomative |
| IdTypeTrain | FK | Идентификационный номер таблицы TypeTrain |
| Driver | IdDriver | PK | Идентификационный номер таблицы Driver |
| DateOfMedical  Examination |  | Дата прохождения медосмотра |
| IdWorker | FK | Идентификационный номер таблицы Worker |
| TrainDriver | IdTrainDriver | PK | Идентификационный номер таблицы TrainDriver |
| IdDriver | FK | Идентификационный номер таблицы Driver |
| IdTrain | FK | Идентификационный номер таблицы Train |
| TypeTrain  Carriage | IdTypeTrainCarriage | PK | Идентификационный номер таблицы TypeTrainCarriage |
| NameType | FK | Название типа вагона |
| TrainCarriage | IdTrainCarriage | PK | Идентификационный номер таблицы TrainCarriage |
| NumberOfSeats |  | Количество мест в вагоне |
| IdTrain | FK | Идентификационный номер таблицы Train |
| IdTypeTrainCarriage | FK | Идентификационный номер таблицы TypeTrainCarriage |
| Route | IdRoute | PK | Идентификационный номер таблицы Route |
| NameRoute |  | Название маршрута |
| DepartureTime |  | Время отправления |
| ArravalTime |  | Время прибытия |
| Ticket | IdTicket | PK | Идентификационный номер таблицы Ticket |
| PlaceofNumber |  | Номер места |
| IsTaken |  | Состояние места |
| TimeofPurchase |  | Время покупки билета |
| IdRoute | FK | Идентификационный номер таблицы Route |

Продолжение таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица | Поле | Ключ | Описание |
| Schedule | IdSchedule | PK | Идентификационный номер таблицы Schedule |
| IdTrain | FK | Идентификационный номер таблицы Train |
| IdRoute | FK | Идентификационный номер таблицы Route |
| Station | IdStation | PK | Идентификационный номер таблицы Station |
| NameStation |  | Название станции |
| StationRoute | IdStationRoute | PK | Идентификационный номер таблицы StationRoute |
| IdRoute | FK | Идентификационный номер таблицы Route |
| IdStation | FK | Идентификационный номер таблицы Station |
| DepartureTime |  | Время отправления |
| ArravalTime |  | Время прибытия |

# 1.3 Определение связей между объектами

Между двумя сущностями может быть установлена связь. Отношения между сущностями характеризуются глаголом, который можно применить для взаимодействия между ними. Связь – это некое отношение между двумя типами сущностей[1].

Внешний ключ (FK) — это столбец или сочетание столбцов, которое применяется для принудительного установления связи между данными в двух таблицах с целью контроля данных, которые могут храниться в таблице внешнего ключа[2]. Другими словами, это указатель на строку другой таблицы.

Связи позволяют по одной сущности находить другие сущности, связанные с ней.

Связи делятся на:

* Однозначная связь (One-to-One) – когда одна из таблиц ссылается на другую, но не наоборот. Например, таблица «Заказы» имеет внешний ключ, связанный с таблицей «Клиенты», что позволяет определить, какой клиент сделал заказ.
* Одноправленная связь (One-to-Many)  – когда обе таблицы имеют внешние ключи, связанные друг с другом. Например, таблица «Авторы» имеет внешний ключ, связанный с таблицей «Книги», и таблица «Книги» также имеет внешний ключ, связанный с таблицей «Авторы». Это позволяет найти авторов для конкретной книги и книги для конкретного автора.
* Множественные (Many-to-Many) связи — каждая запись в одной таблице может иметь несколько соответствующих записей в другой таблице, и наоборот. Например, множество студентов может быть зарегистрировано на множество курсов, и каждый курс может иметь множество студентов[3].

Для реализации информационной системы станции необходимо установить все связи между объектами. Для этого нужно рассмотреть всю информационную систему в совокупности и определить отношения объектов, составляющих систему.

Проследить отношения, в которых состоят таблицы базы данных можно по схеме, изображенной на рисунке A.1 приложения A.

# 1.4 Описание полученной модели на языке инфологического

**проектирования**

Инфологическая модель применяется на втором этапе проектирования БД, т.е. после словесного описания предметной области. При создании информационных систем проект базы данных является фундаментом построения всей системы в целом. Следовательно, инфологическая модель должна включать формализованное описание предметной области, одинаково доступное для пользователей и специалистов по созданию БД[4].

Проектирование инфологической модели предметной области – частично формализованное описание объектов предметной области в терминах некоторой семантической модели, например, в терминах ER-модели (*англ.* entity-relationship model)[5].

Концептуальное проектирование БД - это процесс создания модели используемой информации, не зависящей от любых физических аспектов ее представления. Эта модель данных создается на основе информации, записанной в спецификациях требований пользователей. Концептуальное проектирование БД абсолютно не зависит от таких подробностей ее реализации, как тип выбранной целевой СУБД, набор создаваемых прикладных программ, используемые языки программирования, тип выбранной вычислительной платформы, а также от любых других особенностей физической реализации[6].

Чаще всего [концептуальная модель базы данных](https://intellect.icu/4-kontseptualnaya-model-bazy-dannykh-5342#term-kontseptualnaya-model-bazy-dannykh) включает в себя: описание информационных объектов или понятий предметной области и связей между ними; описание ограничений целостности, т.е. требований к допустимым значениям данных и к связям между ними.

Концептуальная модель проектируемой базы данных представлена на рисунке А.1 приложения А.

# 2 ПОСТРОЕНИЕ СХЕМЫ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ

# 2.1 Построение набора необходимых отношений базы данных

Проектирование реляционной базы данных проходит в том же порядке, что и проектирование БД других моделей данных, но имеет свои особенности.

Проектирование схемы БД должно решать задачи минимизации дублирования данных и упрощения процедур их обработки и обновления[7].

Для разработки схемы реляционной базы данных необходимо определить набор таблиц, составляющих эту базу данных. [Эти таблицы должны содержать всю информацию, хранящуюся в базе данных](https://habr.com/ru/articles/193136/).

На основе полученной концептуальной модели следует определить набор необходимых отношений базы данных. На рисунке 2.1 представлены отношения базы данных.

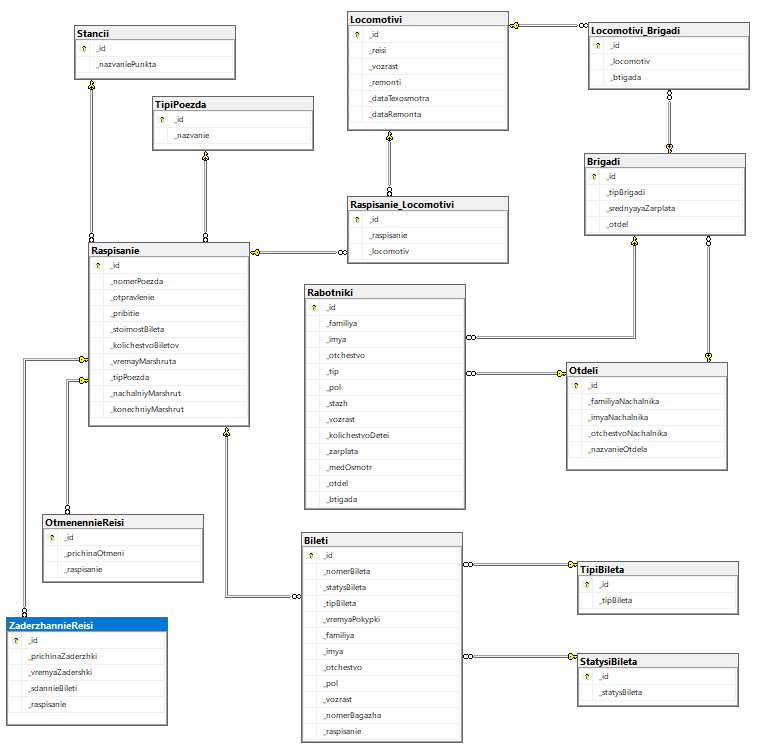


Рисунок 2.1 – Набор необходимых отношений базы данных

# 2.2 Задание первичных и внешних ключей определенных отношений

Ключи представляют способ идентификации строк в таблице. С помощью ключей мы также можем связывать строки между различными таблицами в отношения.

Первичный ключ (primary key) непосредственно применяется для идентификации строк в таблице. Он должен соответствовать следующим ограничениям:

* Первичный ключ должен быть уникальным все время
* Он должен постоянно присутствовать в таблице и иметь значение
* Он не должен часто менять свое значение. В идеале он вообще не должен изменять значение.

Как правило, первичный ключ представляет один столбец таблицы, но также может быть составным и состоять из нескольких столбцов[8].

Первичный ключ (англ. primary key) – в реляционной модели данных один из потенциальных ключей отношения, выбранный в качестве основного ключа (или ключа по умолчанию).

Если в отношении имеется единственный потенциальный ключ, он является и первичным ключом. Если потенциальных ключей несколько, один из них выбирается в качестве первичного, а другие называют «альтернативными».

С точки зрения теории все потенциальные ключи отношения эквивалентны, то есть обладают одинаковыми свойствами уникальности и минимальности. Однако в качестве первичного обычно выбирается тот из потенциальных ключей, который наиболее удобен для тех или иных практических целей, например, для создания внешних ключей в других отношениях либо для создания кластерного индекса. Поэтому в качестве первичного ключа, как правило, выбирают тот, который имеет наименьший размер (физического хранения) или включает наименьшее количество атрибутов[9].

Внешний ключ (FK) — это столбец или сочетание столбцов, которое применяется для принудительного установления связи между данными в двух таблицах с целью контроля данных, которые могут храниться в таблице внешнего ключа. Если один или несколько столбцов, в которых находится первичный ключ для одной таблицы, упоминается в одном или нескольких столбцах другой таблицы, то в ссылке внешнего ключа создается связь между двумя таблицами. Этот столбец становится внешним ключом во второй таблице[10].

Поддержка внешних ключей также называется соблюдением ссылочной целостности. Реляционные СУБД поддерживают автоматический контроль ссылочной целостности.

Первичные ключи имеют постфикс PK, вторичные – FK. Следует упомянуть, что в контексте данной работы первичные ключи будут являться единственным полем в таблице.

Первичные и вторичные ключи представлены в таблице 1.1.

# 2.3 Третья нормальная форма

Нормализация — это процесс организации данных в базе данных, Она включает в себя создание таблиц и установление связей между ними в соответствии с правилами, разработанными как для защиты данных, так и для повышения гибкости базы данных, устраняя избыточность и несогласованную зависимость.

Избыточность данных приводит к непродуктивному расходованию свободного места на диске и затрудняет обслуживание баз данных. Например, если данные, хранящиеся в нескольких местах, потребуется изменить, в них придется внести одни и те же изменения во всех этих местах. Изменение адреса клиента проще реализовать, если эти данные хранятся только в таблице Customers и нигде в базе данных.

Первая нормальная форма

* Устраните повторяющиеся группы в отдельных таблицах.
* Создайте отдельную таблицу для каждого набора связанных данных.
* Идентифицируйте каждый набор связанных данных с помощью первичного ключа.

Не используйте несколько полей в одной таблице для хранения похожих данных. Например, для слежения за товаром, который закупается у двух разных поставщиков, можно создать запись с полями, определяющими код первого поставщика и код второго поставщика.

Что произойдет при добавлении третьего поставщика? Добавление поля не является ответом; он требует изменений в программе и таблице и не обеспечивает плавное размещение динамического числа поставщиков. Вместо этого можно поместить все сведения о поставщиках в отдельную таблицу Vendors (поставщики) и связать товары с поставщиками с помощью кодов товаров или поставщиков с товарами с помощью кодов поставщиков.

Вторая нормальная форма

* Создайте отдельные таблицы для наборов значений, относящихся к нескольким записям.
* Свяжите эти таблицы с помощью внешнего ключа.
* Записи не должны зависеть от чего-либо, кроме первичного ключа таблицы (составного ключа, если это необходимо). Возьмем для примера адрес клиента в системе бухгалтерского учета. Этот адрес необходим не только таблице Customers, но и таблицам Orders, Shipping, Invoices, Accounts Receivable и Collections. Вместо того чтобы хранить адрес клиента как отдельный элемент в каждой из этих таблиц, храните его в одном месте: или в таблице Customers, или в отдельной таблице Addresses.

Третья нормальная форма

* Исключите поля, которые не зависят от ключа.

Значения в записи, которые не являются частью ключа этой записи, не принадлежат в таблице. Если содержимое группы полей может относиться более чем к одной записи в таблице, попробуйте поместить эти поля в отдельную таблицу.

Например, в таблицу Employee Recruitment (наем сотрудников) можно включить адрес кандидата и название университета, в котором он получил образование. Однако для организации групповой почтовой рассылки необходим полный список университетов. Если сведения об университетах будут храниться в таблице Candidates, составить список университетов при отсутствии кандидатов не получится. Таким образом, создайте вместо этого отдельную таблицу Universities и свяжите ее с таблицей Candidates при помощи ключа — кода университета[11].

# 2.4 Определение ограничений целостности для внешних ключей

**отношений и для отношений в целом**

Целостность базы данных — соответствие имеющейся в [базе данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) информации её внутренней логике, структуре и всем явно заданным правилам. Каждое правило, налагающее некоторое ограничение на возможное состояние базы данных, называется [ограничением целостности](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%B3%D1%80%D0%B0%D0%BD%D0%B8%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D1%86%D0%B5%D0%BB%D0%BE%D1%81%D1%82%D0%BD%D0%BE%D1%81%D1%82%D0%B8&action=edit&redlink=1).

Очевидно, что ограничения должны быть формально объявлены для [СУБД](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), после чего СУБД должна предписывать их выполнение. Объявление ограничений сводится просто к использованию соответствующих средств языка базы данных, а соблюдение ограничений осуществляется с помощью контроля со стороны СУБД над операциями обновления, которые могут нарушить эти ограничения, и запрещения тех операций, которые их действительно нарушают. При первоначальном объявлении ограничения система должна проверить, удовлетворяет ли ему в настоящий момент база данных. Если это условие не соблюдается, ограничение должно быть отвергнуто; в противном случае оно принимается (то есть записывается в каталог системы) и начиная с этого момента соблюдается[12].

Ограничения целостности можно определить как специальные средства в базах данных, главное назначение которых – не дать попасть в базу недопустимым данным.

Ограничение целостности отношений заключается в том, что в любом отношении должны отсутствовать записи с одним и тем же значением первичного ключа. Конкретно требование состоит в том, что любая запись любого отношения должна быть отличной от любой другой записи этого отношения. Это требование автоматически удовлетворяется, если в системе не нарушаются базовые свойства отношений.

Ограничение целостности подразумевает, что в любом отношении не должны появляться записи с одним и тем же значением первичного ключа. То есть любая запись отношения должна быть отлична от любой другой записи этого же отношения. В проектируемой таблице все сущности будут иметь первичный ключ. Он необходим для её однозначной идентификации. Например, на клиентской стороне, также будет использоваться при удалении, добавлении или обновлении записи.

Условиями целостности называется набор правил, используемых для поддержания допустимых межтабличных связей и запрета на случайное изменение или удаление связанных данных.

Для автоматического обновления связанных полей (удаления записей) при обновлении (удалении) в главной таблице, следует устанавливать обеспечение целостности данных и каскадное обновление связанных полей (каскадное удаление связанных записей).

Для удовлетворения требования ограничения целостности для внешних ключей отношений и для отношений в целом необходимо, чтобы выполнялось соответствие между типами вводимых данных и типами столбцов в таблицах, чтобы были заполнены все обязательные поля в таблицах, то есть те поля, которые не могут содержать значения NULL.

Система управления базами данных не может контролировать правильность каждого отдельного значения, вводимого в базу данных. Для этого существует ряд средств, помогающих разработчику минимизировать возможность нарушения целостности данных базы: триггеры, проверки, уникальность и другое[13].

# 2.5 Графическое представление связей между внешними

**первичными ключами**

По результатам нормализации и определении первичных ключей, внешних ключей и связей между сущностями, была разработана схема реляционной базы данных, которая представлена на рисунке Б.1 приложения Б. На данной схеме изображаются все отношения базы данных, а также связи между внешними и первичными ключами.

# 3 СОЗДАНИЕ СПРОЕКТИРОВАННОЙ БАЗЫ ДАННЫХ

В системе управления базами данных SQL Server 2019 была реализована спроектированная ранее база данных станции. Данная система была выбрана по ряду весомых причин: широкое распространение, наличие свободно распространяемых сборок, наличие высококачественных программных средств разработки, позволяющих создавать разного вида приложения, которые, в свою очередь, смогут использовать базы данных Microsoft SQL Server.

В свою очередь стоит упомянуть о преимуществах использования SQL Server 2019 для разработки и управления базами данных. Например, SQL Server обладает мощными инструментами для обеспечения безопасности данных, такими как механизмы шифрования, аудита и управления доступом. Кроме того, SQL Server предлагает широкий спектр возможностей для оптимизации производительности баз данных, включая индексацию, хранение данных в памяти и оптимизацию запросов.

Также стоит отметить, что SQL Server 2019 поддерживает различные типы данных и функциональности, что делает его универсальным инструментом для создания разнообразных приложений. Благодаря активной поддержке и развитию со стороны Microsoft, SQL Server постоянно обновляется и совершенствуется, что обеспечивает надежность и эффективность работы с базами данных на долгосрочной основе.

Описание структур каждой из таблиц базы данных с описанием типа полей представлено в таблицах ниже.

Таблица 3.1 – Характеристики атрибутов таблицы Speciality

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdSpeciality | INT | Идентификационный номер таблицы Speciality |
| SalaryOneRate | MONEY | Зарплата за одну ставку |
| Name | VARCHAR(30) | Наименование специальности |
| Description | VARCHAR(100) | Описание специальности |

Таблица 3.2 – Характеристики атрибутов таблицы TypeDepartment

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdTypeDepartment | INT | Идентификационный номер таблицы TypeDepartment |
| NameType | VARCHAR(50) | Название типа отдела |

Таблица 3.3 – Характеристики атрибутов таблицы Department

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdDepartment | INT | Идентификационный номер таблицы Department |
| Name | VARCHAR(50) | Название отдела |

Продолжение таблицы 3.3

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| Description | VARCHAR(150) | Описание отдела |
| CountBrigade | INT | Количество бригад в отделе |
| IdTypeDepartment | INT | Идентификационный номер таблицы TypeDepartment |

Таблица 3.4 – Характеристики атрибутов таблицы TypeEngine

| Поле | Тип данных | Описание |
| --- | --- | --- |
| IdTypeEngine | INT | Идентификационный номер таблицы TypeEngine |
| NameType | VARCHAR(50) | Название типа двигателя |

Таблица 3.5 – Характеристики атрибутов таблицы Engine

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdEngine | INT | Идентификационный номер таблицы Engine |
| Name | VARCHAR(20) | Название двигателя |
| Efficiency | FLOAT | КПД |
| NumberOfCylinders | INT | Количество цилиндров |
| IdTypeEngine | INT | Идентификационный номер таблицы TypeEngine |

Таблица 3.6 – Характеристики атрибутов таблицы Locomative

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdLocomative | INT | Идентификационный номер таблицы Locomative |
| Name | VARCHAR(20) | Название локоматива |
| Weight | FLOAT | Вес |
| TractionForce | FLOAT | Сила тяги |
| InspectionDate | DATE | Дата тех. обслуживания |
| IdEngine | INT | Идентификационный номер таблицы Engine |

Таблица 3.7 – Характеристики атрибутов таблицы Brigade

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdBrigade | INT | Идентификационный номер таблицы Brigade |
| CountWorker | INT | Количество работников в бригаде |
| NumberInDepartment | INT | Номер бригады в отделе |
| IdDepartment | INT | Идентификационный номер таблицы Department |
| IdLocomative | INT | Идентификационный номер таблицы Locomative |

Таблица 3.8 – Характеристики атрибутов таблицы Worker

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdWorker | INT | Идентификационный номер таблицы Worker |
| Name | VARCHAR(50) | Имя работника |
| Surname | VARCHAR(50) | Фамилия работника |
| Email | VARCHAR(100) | Email работника |
| Rate | FLOAT | Тарифная ставка работника |
| IdSpeciality | INT | Идентификационный номер таблицы Speciality |
| IdBrigade | INT | Идентификационный номер таблицы Brigade |

Таблица 3.9 – Характеристики атрибутов таблицы TypeTrain

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdTypeTrain | INT | Идентификационный номер таблицы TypeTrain |
| NameType | VARCHAR(10) | Название типа поезда |

Таблица 3.10 – Характеристики атрибутов таблицы Train

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdTrain | INT | Идентификационный номер таблицы Train |
| Name | VARCHAR(20) | Имя поезда |
| NumberOfTrainCarriage | INT | Количество вагонов |
| IdLocomative | INT | Идентификационный номер таблицы Locomative |
| IdTypeTrain | INT | Идентификационный номер таблицы TypeTrain |

Таблица 3.11 – Характеристики атрибутов таблицы Driver

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdDriver | INT | Идентификационный номер таблицы Driver |
| DateOfMedical  Examination | DATE | Дата прохождения медосмотра |
| IdWorker | INT | Идентификационный номер таблицы Worker |

Таблица 3.12 – Характеристики атрибутов таблицы TrainDriver

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdTrainDriver | INT | Идентификационный номер таблицы TrainDriver |

Продолжение таблицы 3.12

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdDriver | INT | Идентификационный номер таблицы Driver |
| IdTrain | INT | Идентификационный номер таблицы Train |

Таблица 3.13 – Характеристики атрибутов таблицы TypeTrainCarriage

| Поле | Тип данных | Описание |
| --- | --- | --- |
| IdTypeTrainCarriage | INT | Идентификационный номер таблицы TypeTrainCarriage |
| NameType | VARCHAR(20) | Название типа вагона |

Таблица 3.14 – Характеристики атрибутов таблицы TrainCarriage

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdTrainCarriage | INT | Идентификационный номер таблицы TrainCarriage |
| NumberOfSeats | INT | Количество мест в вагоне |
| IdTrain | INT | Идентификационный номер таблицы Train |
| IdTypeTrainCarriage | INT | Идентификационный номер таблицы TypeTrainCarriage |

Таблица 3.15 – Характеристики атрибутов таблицы Route

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdRoute | INT | Идентификационный номер таблицы Route |
| NameRoute | VARCHAR(20) | Название маршрута |
| DepartureTime | DATETIME | Время отправления |
| ArravalTime | DATETIME | Время прибытия |

Таблица 3.16 – Характеристики атрибутов таблицы Ticket

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdTicket | INT | Идентификационный номер таблицы Ticket |
| PlaceofNumber | INT | Номер места |
| IsTaken | BIT | Состояние места |
| TimeofPurchase | DATETIME | Время покупки билета |
| IdRoute | INT | Идентификационный номер таблицы Route |

Таблица 3.17 – Характеристики атрибутов таблицы Schedule

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdSchedule | INT | Идентификационный номер таблицы Schedule |

Продолжение таблицы 3.17

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdTrain | INT | Идентификационный номер таблицы Train |
| IdRoute | INT | Идентификационный номер таблицы Route |

Таблица 3.18 – Характеристики атрибутов таблицы Station

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdStation | INT | Идентификационный номер таблицы Station |
| NameStation | VARCHAR(60) | Название станции |

Таблица 3.19 – Характеристики атрибутов таблицы StationRoute

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdStationRoute | INT | Идентификационный номер таблицы StationRoute |
| IdRoute | INT | Идентификационный номер таблицы Route |
| IdStation | INT | Идентификационный номер таблицы Station |
| DepartureTime | DATETIME | Время отправления |
| ArravalTime | DATETIME | Время прибытия |

# 4 ЗАПИСЬ ВЫРАЖЕНИЙ, УКАЗАННЫХ В ВАРИАНТЕ ЗАДАНИЯ

**ТИПОВ ЗАПРОСОВ НА ЯЗЫКЕ SQL**

Описание запросов и их реализация указаны в названии листингов и их содержании ниже.

Листинг 4.1 – Получить перечень и общее число всех работников железнодорожной станции, начальников отделов, работников указанного отдела, по стажу работы на станции, половому признаку, возрасту, признаку наличия и количества детей, размеру заработной платы

SELECT \* FROM Rabotniki

SELECT \* FROM Otdeli

SELECT \* FROM Rabotniki

WHERE Rabotniki.\_otdel = 1

SELECT \* FROM Rabotniki

WHERE \_stazh > 20

SELECT \* FROM Rabotniki

WHERE \_pol = 'жен'

SELECT \* FROM Rabotniki

WHERE \_vozrast < 45

SELECT \* FROM Rabotniki

WHERE \_kolichestvoDetei IS NOT NULL AND \_kolichestvoDetei > 1

SELECT \* FROM Rabotniki

WHERE \_zarplata > 800.5001

Листинг 4.2 – Получить перечень и общее число работников в бригаде, по всем отделам, в указанном отделе, обслуживающих некоторый локомотив, по возрасту, суммарной (средней) зарплате в бригаде

SELECT COUNT(Rabotniki.\_id) AS 'Количество рабочих', Otdeli.\_nazvanieOtdela AS 'Отдел' FROM Rabotniki, Otdeli

WHERE Otdeli.\_id = Rabotniki.\_otdel

GROUP BY Otdeli.\_nazvanieOtdela

SELECT COUNT(\*) AS 'Количество рабочих по всем отделам' FROM Rabotniki

WHERE \_otdel IS NOT NULL

SELECT \* FROM Rabotniki, Brigadi

WHERE Rabotniki.\_btigada = Brigadi.\_id AND Brigadi.\_srednyayaZarplata <2000

Листинг 4.3 – Получить перечень и общее число водителей локомотивов, прошедших медосмотр либо не прошедших медосмотр в указанный год, по половому признаку, возрасту, размеру заработной платы

SELECT \* FROM Rabotniki

WHERE Rabotniki.\_tip = 'водитель' AND Rabotniki.\_medOsmotr IS NOT NULL

Листинг 4.4 – Получить перечень и общее число локомотивов, приписанных к железнодорожной станции, находящихся на ней в указанное время, по времени прибытия на станции, по количеству совершенных маршрутов

Продолжение листинга 4.4

SELECT Raspisanie.\_nachalniyMarshrut, Raspisanie.\_konechniyMarshrut, Locomotivi.\_id FROM Locomotivi, Raspisanie, Raspisanie\_Locomotivi

WHERE \_pribitie >= '2019-02-01 15:30:00.000' AND \_pribitie<='2019-02-03 15:30:00.000' AND

Raspisanie.\_id = Raspisanie\_Locomotivi.\_raspisanie AND Locomotivi.\_id = Raspisanie\_Locomotivi.\_locomotiv

Листинг 4.5 – Получить перечень и общее число локомотивов, прошедших плановый техосмотр за определенный период времени, отправленных в ремонт в обозначенное время, ремонтированных указанное число раз, по количеству совершенных рейсов до ремонта, по возрасту локомотива

SELECT \* FROM Locomotivi

WHERE \_dataTexosmotra>='2022-10-06' AND \_dataTexosmotra <='2022-10-8'

SELECT \* FROM Locomotivi

WHERE \_dataRemonta = '2022-10-06'

Листинг 4.6 – Получить перечень и общее число поездов на указанном маршруте, по длительности маршрута, по цене билета и по всем этим критериям сразу

SELECT \* FROM Raspisanie

WHERE \_nachalniyMarshrut = '' AND \_konechniyMarshrut = ''

Листинг 4.7 – Получить перечень и общее число отмененных рейсов полностью, в указанном направлении, по указанному маршруту

SELECT \* FROM OtmenennieReisi, Raspisanie, Stancii

WHERE Raspisanie.\_id = OtmenennieReisi.\_raspisanie AND Stancii.\_nazvaniePunkta = 'Туда' AND Raspisanie.\_konechniyMarshrut = Stancii.\_id

Листинг 4.8 – Получить перечень и общее число задержанных рейсов полностью, по указанной причине, по указанному маршруту, и количество сданных билетов за время задержки

SELECT \* FROM ZaderzhannieReisi

WHERE \_prichinaZaderzhki = 'Погодные условия'

Листинг 4.9 – Получить перечень и среднее количество проданных билетов за указанный интервал времени на определённые маршруты, по длительности маршрута, по цене билета

SELECT Bileti.\_nomerBileta FROM Bileti, Raspisanie

WHERE \_vremyaPokypki IS NULL AND Bileti.\_raspisanie = Raspisanie.\_id AND Raspisanie.\_konechniyMarshrut =3

Листинг 4.10 – Получить перечень и общее число маршрутов указанной категории, следующих в определенном напpавлении

SELECT \* FROM Raspisanie

WHERE \_tipPoezda = 1 AND \_konechniyMarshrut = 3

Листинг 4.11 – Получить перечень и общее число пассажиров на указанном рейсе, уехавших в указанный день, уехавших за границу в указанный день, по признаку сдачи вещей в багажное отделение, по половому признаку, по возрасту

SELECT \* FROM Raspisanie, Bileti

Продолжение листинга 4.11

WHERE Raspisanie.\_otpravlenie<='2022-10-06 03:00:00.000' AND Raspisanie.\_pribitie >='2022-10-06 04:00:00.000' AND Bileti.\_raspisanie = Raspisanie.\_id AND Bileti.\_statysBileta != 1

Листинг 4.12 – Получить перечень и общее число невыкупленных билетов на указанном рейс, день, некоторый маршрут

SELECT FROM Bileti

WHERE \_statysBileta != ''

Листинг 4.13 – Получить общее число сданных билетов на указанный рейс, день, маршрут

SELECT COUNT(\*) AS 'Count' FROM Bileti

WHERE \_statysBileta = 1 AND \_id = 1)

# 5 ВЫБОР И ОСНОВАНИЕ СРЕДСТВ РАЗРАБОТКИ

**ПРИЛОЖЕНИЯ**

Для реализации приложения была выбрана среда разработки Microsoft Visual Studio 2022, в качестве языка программирования – C#.

Был выбран именно этот язык по следующим причинам:

* Претендует на подлинную объектную ориентированность (а всякая языковая сущность претендует на то, чтобы быть объектом).
* Призван практически реализовать компонентно-ориентированный подход к программированию, который способствует меньшей машинно-архитектурной зависимости результирующего программного кода, большей гибкости, переносимости и легкости повторного использования как программ целиком, так и ее фрагментов.
* Совместимость с базами данных на SQL Server 2018.

Для объединения базы данных с интерфейсом приложения использовался SqlClient.

SqlClient – это ресурс для администрирования базы данных с использованием языка структурированных запросов (SQL). SqlClient также является пространством имен, предназначенным для этих ресурсов, чтобы помочь реализовать операции SQL [6].

Специальный инструмент под названием SQuirreL SQL Client работает на драйвере подключения к базе данных Java и помогает пользователям взаимодействовать с базой данных, а также предлагает независимые от базы данных функции. SQuirreL SQL Client – это программа с открытым исходным кодом, которая работает с другой инфраструктурой Java.

В общем, технология SqlClient поможет различными способами управлять базой данных или несколькими базами данных. У некоторых инструментов SQL есть деревья объектов или другие визуальные отображения. Они также могут содержать различные изменения в файлах с элементами SQL. Эти ресурсы помогают менеджерам баз данных больше работать с информацией, хранящейся в определенном наборе таблиц.

Данная технология была выбрана для подключения к базе данных из-за своей скорости работы с простыми запросами к базе. Entity Framework безусловно более гибок и быстр в освоении людьми, не знающих язык SQL, и позволяет ускорить кодировку из-за своей автоматической генерации кода, но в данной работе обращение к базе происходит банальными CRUD-операциями, из-за чего фреймворк отошел на второй план.

Выбор СУБД – важный этап при реализации БД. Существующие критерии, по которым можно оценивать СУБД условно делятся на три большие области:

* функциональность;
* особенности разработки приложений;
* смешанные критерии.

Условность такого деления определяется тем, что некоторые особенности конкретной СУБД проявляются не только в одной области, в которой производится оценка, но и влияют на другие.

Существуют еще две области оценок СУБД:

* поддерживаемые платформы;
* производительность.

Рассмотрим критерии оценки объектных СУБД с точки зрения разработки высокоэффективных приложений.

Проведенное исследование позволяет разбить эту область на пять групп, в которых можно с большой степенью независимости анализировать характеристики СУБД и оценивать их влияние на возможные свойства разрабатываемых приложений:

* особенности объектной модели;
* архитектура СУБД;
* механизмы СУБД;
* программирование интерфейса приложений;
* запросы к объектам.

На основе вышесказанного была выбрана система SQL Server. Microsoft SQL Server – система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов – Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL и является реализацией стандарта ANSI/ISO по структурированному языку запросов с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия, а также конкурирует с другими СУБД в этом сегменте рынка.

Данная СУБД полностью реализовывает все аспекты реляционной модели данных и позволяет нам с помощью встроенных функций поддерживать ссылочную целостность данных. Так же MS SQL Server стал де факто стандартом источника данных для всех решений Microsoft, что облегчает его интеграцию с различными программными пакетами.

Расширенные функции безопасности, в сочетании со встроенными, удобными для использования инструментами и управляемым доступом к данным, позволяют организации выполнить требования строгих политик соответствия нормам.

# 6 РЕАЛИЗАЦИЯ ЗАКОНЧЕННОГО ПРИЛОЖЕНИЯ,

**РАБОТАЮЩЕГО С СОЗДАННОЙ БАЗОЙ ДАННЫХ**

# 6.1 Разработка и построение интерфейса главной и рабочих форм

Главная страница представлена в виде страницы с возможностью перехода по представленным пунктам системы, с помощью которых можно просматривать всю информацию, касающейся станций, локомотивов, работников и так далее.

При запуске программы пользователь попадает на главную страницу программы, на которой расположены все пункты управления данными железнодорожной станции.

Все основные формы и виды выполнены в отдельных страницах программы с собственным функционалом и с использованием интерактивных полей ввода для взаимодействия с пользователем.

При проектировании программы были учтены все возможные случаи некорректной работы программы, поэтому большинство нештатных ситуаций сопровождается оповещениями с описанием проблемы.

Скриншоты, а также описание работы главной и рабочих окон представлены в приложении Г.

Для создания интерфейса мы используем Windows Forms, которая значительно облегчает разработку десктопного приложения, благодаря своему встроенному графическому редактору.

Графический редактор позволяет в графическом виде представить создаваемую форму и в принципе упрощает работу с графическими компонентами.

Для открытия формы в режиме графического дизайнера необходимо в структуре проекта нажать на файл формы, либо левой кнопкой мыши двойным кликом, либо правой кнопкой мыши. В появившемся контекстном меню необходимо выбрать View Designer.

Visual Studio имеет еще одну связанную функциональность. Она обладает панелью графических инструментов. И мы можем, вместо создания элементов управления в коде C#, просто переносить их на форму с панели инструментов с помощь мыши.

# 6.2 Построение главного меню и кнопок панели инструментов

Навигационная панель программы представлена пунктами: Меню, Таблицы, Запрос на получение перечня, Справка. Данные пункты выполнены в виде выпадающего меню. Подпунктами меню «Таблицы» являются названия таблиц используемой базы данных [4].

# 6.3 Выполнение программного кода на языке C#

Далее следует описание работы программы с базой данных. Все необходимые методы для работы с базами данных описаны в классе DBMethods и связываются с базой данных при помощи класса SqlConnection. Подключение к базе данных начинается с формирования строки подключения. Код класса DBMethods представлен в листинге 6.1.

Листинг 6.1 – Методы для работы с базой данных

using System;

using System.Data;

using System.Data.SqlClient;

using System.IO;

using System.Reflection;

using System.Windows.Forms;

namespace TrainsApp

{

class DBMethods

{

SqlConnection connection = new SqlConnection(@"Data Source=DESKTOP-JPAQV1A;Initial Catalog=TrainS;Integrated Security=True");

public void OpenConnection()

{

if (connection.State == System.Data.ConnectionState.Closed)

{

connection.Open();

}

}

public void CloseConnectiom()

{

if (connection.State == System.Data.ConnectionState.Open)

{

connection.Close();

}

}

public SqlConnection GetConnection()

{

return connection;

}

public void DefaultUpdateGrid(string query, DataGridView show)

{

SqlDataAdapter adapter = new SqlDataAdapter();

DataTable dt = new DataTable();

try

{

SqlCommand Comand = new SqlCommand(query, GetConnection());

adapter.SelectCommand = Comand;

adapter.Fill(dt);

show.DataSource = dt;

Продолжение листинга 6.1

show.Columns[0].Visible = false;

}

catch

{

MessageBox.Show("Что-то пошло не так, повторите попытку ил обратитесь в службу поддержки", "", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

public void CUOperations(string query)

{

try

{

SqlCommand Comand = new SqlCommand(query, GetConnection());

OpenConnection();

Comand.ExecuteNonQuery();

CloseConnectiom();

}

catch

{

MessageBox.Show("Не удалось внести данные, проверьте правильность и повторите попытку", "", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

public void DeleteOperation(string tableName, int elementId)

{

try

{

SqlCommand Comand = new SqlCommand($"DELETE FROM {tableName} WHERE \_id = {elementId}", GetConnection());

OpenConnection();

Comand.ExecuteNonQuery();

CloseConnectiom();

}

catch

{

MessageBox.Show("Не удалось внести данные, проверьте правильность и повторите попытку", "", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

# ЗАКЛЮЧЕНИЕ

В процессе выполнения данной курсовой работы были закреплены навыки проектировании баз данных и реализации их в MS SQL Server 2018.

Были определены основные цели системы в соответствии с выбранным вариантом и выделены требования, которым должна удовлетворять система.

Спроектированная база данных соответствует всем требованиям, которые предъявляются в задании. Данная база позволяет без проблем хранить и извлекать нужную информацию. Разработанная система реагирует на ошибочный ввод данных, а также способна определять возникающие ошибки и уведомлять об этом пользователя, чтобы в любой момент он знал из-за чего или почему произошла ошибка, и устранил её.

Разработанная информационная система железнодорожной пассажирской станции удовлетворяет всем требованиям комфортного использования и обеспечивает беспроблемное хранение и изменение информации.

В результате выполнения курсовой работы были спроектированы и реализованы: база данных информационной системы железнодорожной пассажирской станции, программа, которая эффективно взаимодействует с базой данных. Программное средство реализовано с помощью языка программирования C#.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* 1. Информатика [Электронный ресурс]. Режим доступа: <http://flash-library.narod.ru/Ch-Informatics/lektion/lektion7.html>. Дата доступа 21.11.2022.
  2. СУБД [Электронный ресурс]. Режим доступа: <https://clck.ru/NnKES>. Дата доступа: 22.11.2022.
  3. Понятие первичного и вторичного ключа [Электронный ресурс]. Режим доступа: <http://informatic.ugatu.ac.ru/lib/office/Proekt.htm>. Дата доступа 22.11.2022.
  4. Пример оформления курсовой работы [Электронный ресурс]. Режим доступа: <https://studbooks.net/2156021/informatika/zaklyuchenie>. Дата доступа 22.11.2022.
  5. Нормализация [Электронный ресурс]. Режим доступа: https://metanit.com/sql/tutorial/2.1.php/. Дата доступа 19.11.2022.
  6. Изучаем C#, 3-е издание. Эндрю Стиллмен, Дженнифер Грин. [Электронный ресурс]. Режим доступа: https://codernet.ru/books/c\_sharp/endryu\_stillmen\_-\_izuchaem\_c\_3-e\_izdanie/. Дата доступа 20.11.2022.

# ПРИЛОЖЕНИЕ А

(обязательное)

**Концептуальная схема БД**

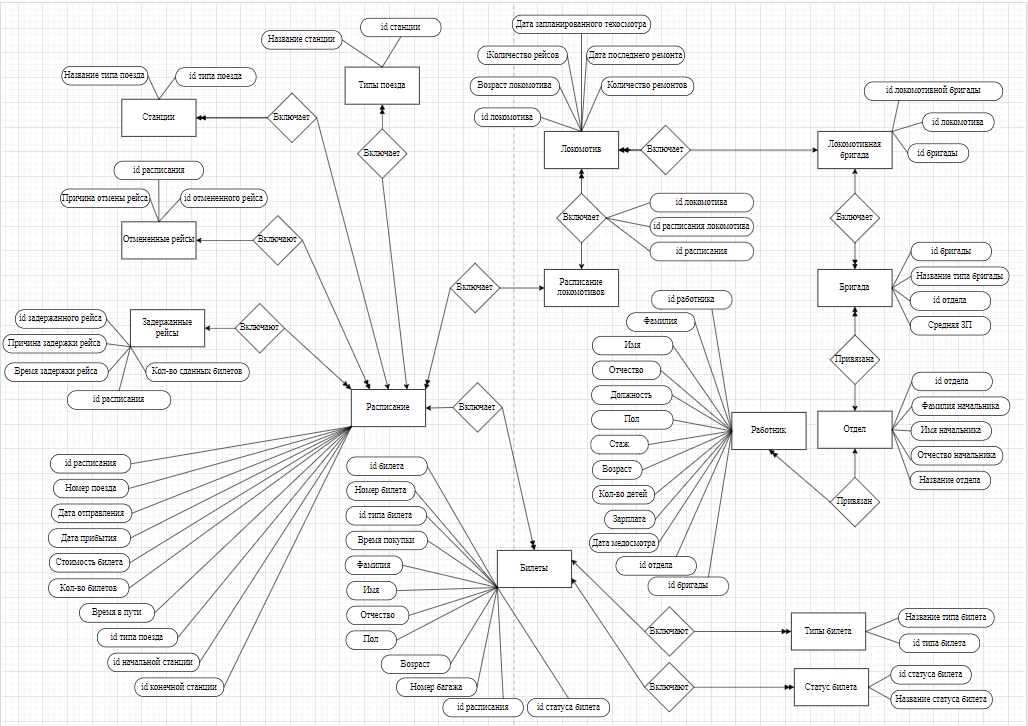


Рисунок А.1 – Концептуальная модель проектируемой базы данных

# ПРИЛОЖЕНИЕ Б

(обязательное)

**Схема реляционной базы данных**

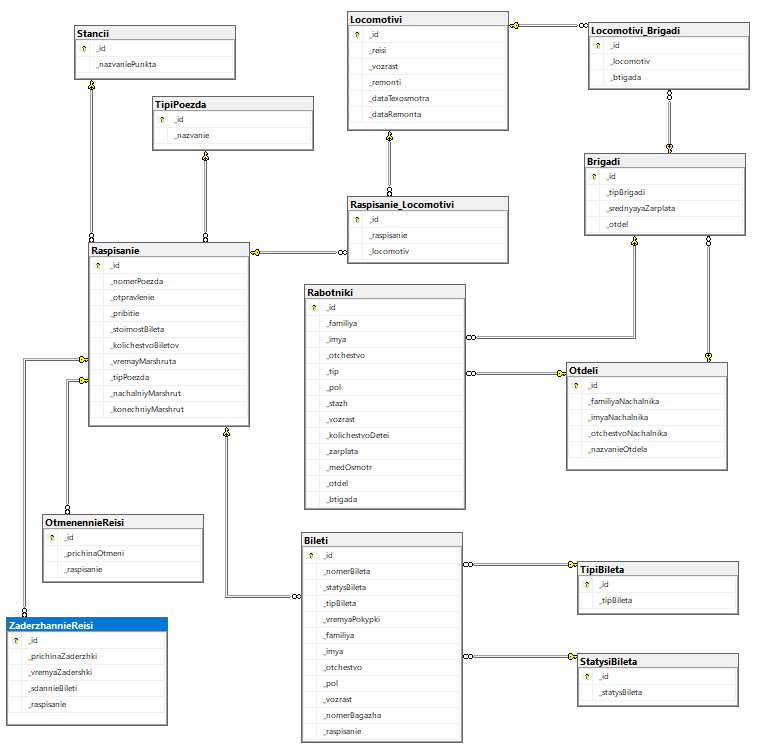


Рисунок Б.1 – Схема реляционной базы данных

# ПРИЛОЖЕНИЕ В

(обязательное)

**Описание задания курсовой работы**

Реализованная база данных должна состоять не менее чем из 15-и таблиц, а также выполнять все указанные в варианте запросы. Кроме того, в программе, при выводе данных, пользователь не должен видеть уникальные идентификаторы. Добавление, удаление, редактирование и поиск данных также не должно осуществляться с помощью уникальных идентификаторов. (ДАННЫЙ АБЗАЦ ДОЛЖНО БЫТЬ У КАЖДОГО СТУДЕНТА).

Работников железнодорожной станции можно подразделить на водителей подвижного состава, диспетчеров, ремонтников подвижного состава, путей, кассиров, работников службы подготовки составов, справочной службы и других, которые административно относятся каждый к своему отделу. Каждая из перечисленных категорий работников имеет уникальные атрибуты-характеристики, определяемые профессиональной направленностью. В отделах существует разбиение работников на бригады. Отделы возглавляются начальниками, которые представляют собой администрацию железнодорожной станции. В функции администрации входит планирование маршрутов, составление расписаний, формирование кадрового состава железнодорожной станции. За каждым локомотивом закрепляется локомотивная бригада. За несколькими локомотивами закрепляется бригада техников-ремонтников, выполняющая рейсовый и плановый техосмотр (по определенному графику), ремонт, техническое обслуживание. Водители локомотивов обязаны проходить каждый год медосмотр, не прошедших медосмотр необходимо перевести на другую работу. Локомотив должен своевременно осматриваться техниками-ремонтниками и про необходимости ремонтироваться. Подготовка к рейсу включает в себя техническую часть (рейсовый техосмотр, мелкий ремонт) и обслуживающую часть (уборка вагонов, запас продуктов питания и т.п.).

В расписании указывается тип поезда (скорый, пассажирский . . .), номер поезда, дни и время отправления и прибытия, маршрут (начальный и конечный пункты назначения, основные узловые станции), стоимость билета. Билеты на поезд можно приобрести заранее или забронировать в железнодорожных кассах. До отправления поезда, если есть необходимость, билет можно вернуть. Отправление поездов может быть задержано из-за опозданий поездов, погодных условий, технических неполадок.

Железнодорожные маршруты можно разделить на следующие категории: внутренние, международные, туристические, специальные маршруты. Пассажиры могут сдавать свои вещи в багажное отделение.

Виды запросов в информационной системе:

* Получить перечень и общее число всех работников железнодорожной станции, начальников отделов, работников указанного отдела, по стажу работы на станции, половому признаку, возрасту, признаку наличия и количества детей, размеру заработной платы.
* Получить перечень и общее число работников в бригаде, по всем отделам, в указанном отделе, обслуживающих некоторый локомотив, по возрасту, суммарной (средней) зарплате в бригаде.
* Получить перечень и общее число водителей локомотивов, прошедших медосмотр либо не прошедших медосмотр в указанный год, по половому признаку, возрасту, размеру заработной платы.
* Получить перечень и общее число локомотивов, приписанных к железнодорожной станции, находящихся на ней в указанное время, по времени прибытия на станции, по количеству совершенных маршрутов.
* Получить перечень и общее число локомотивов, прошедших плановый техосмотр за определенный период времени, отправленных в ремонт в обозначенное время, ремонтированных указанное число раз, по количеству совершенных рейсов до ремонта, по возрасту локомотива.
* Получить перечень и общее число поездов на указанном маршруте, по длительности маршрута, по цене билета и по всем этим критериям сразу.
* Получить перечень и общее число отмененных рейсов полностью, в указанном направлении, по указанному маршруту.
* Получить перечень и общее число задержанных рейсов полностью, по указанной причине, по указанному маршруту, и количество сданных билетов за время задержки.
* Получить перечень и среднее количество проданных билетов за указанный интервал времени на определённые маршруты, по длительности маршрута, по цене билета.
* Получить перечень и общее число маршрутов указанной категории, следующих в определенном направлении.
* Получить перечень и общее число пассажиров на указанном рейсе, уехавших в указанный день, уехавших за границу в указанный день, по признаку сдачи вещей в багажное отделение, по половому признаку, по возрасту.
* Получить перечень и общее число невыкупленных билетов на указанном рейс, день, некоторый маршрут.
* Получить общее число сданных билетов на указанный рейс, день, маршрут.

Необходимо предусмотреть возможность выдачи пассажиру билета, в котором указано локомотив, маршрут, дата и время отправления, а также наличия багажа.

# ПРИЛОЖЕНИЕ Г

(обязательное)

**Главная и рабочие формы приложения**

В ходе проектирования было определено, что при запуске приложения будет открываться главное окно с общим элементом для вывода содержимого таблиц. Окно входа в систему представлено на рисунке Г.1.

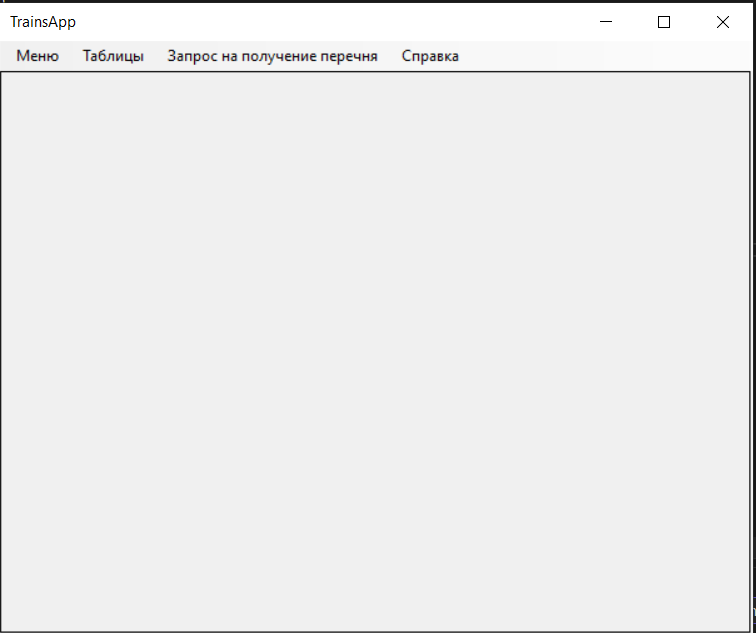


Рисунок Г.1 – Главное окно приложения

Главное окно при нажатии на пункт «Отделы» выпадающего меню «Таблицы» представлено на рисунке Г.2.

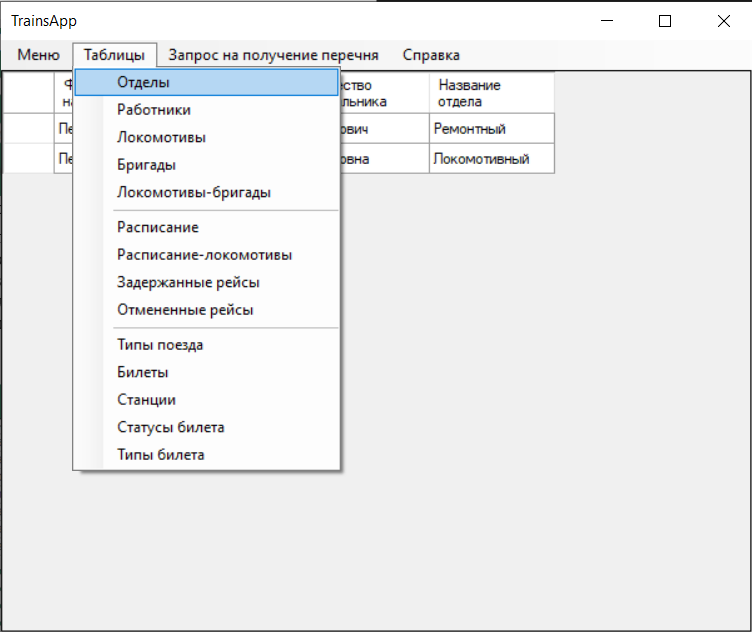


Рисунок Г.2 – Выбор пункта «Отделы»

При выборе из какой таблицы произвести общую выборку приложение определяет какое окно добавления или редактирования открывать после нажатия на пункты «Добавить» и «Редактировать» выпадающего меню «Меню».

На рисунке Г.3 представлено окно добавления новых отделов, так как до этого мы запросили отобразить все отделы базы данных, обращаясь к таблице «Отдели».

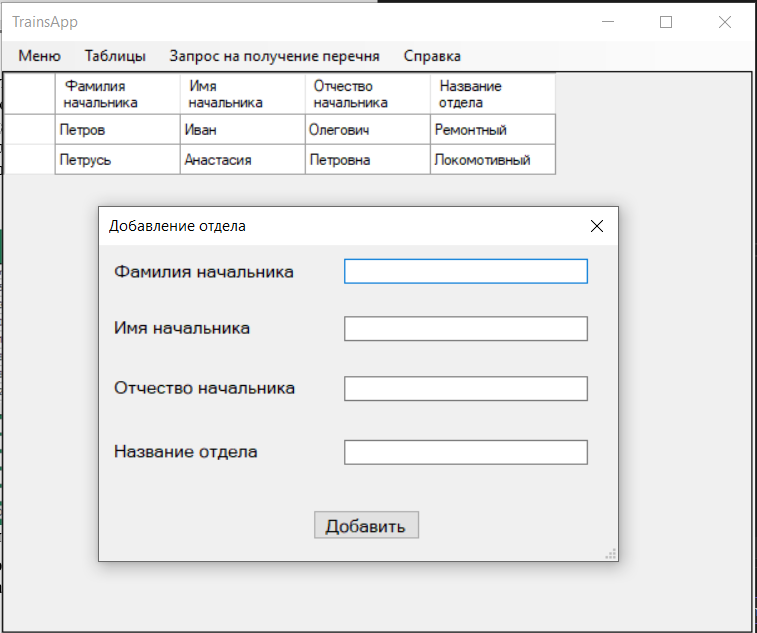


Рисунок Г.3 – Окно добавления записи

При редактировании записей мы обязаны указать какую именно запись мы хотим отредактировать, после этого нам отобразиться меню редактирования с выгруженными полями выбранной записи. Окно редактирования представлено на рисунке Г.4.

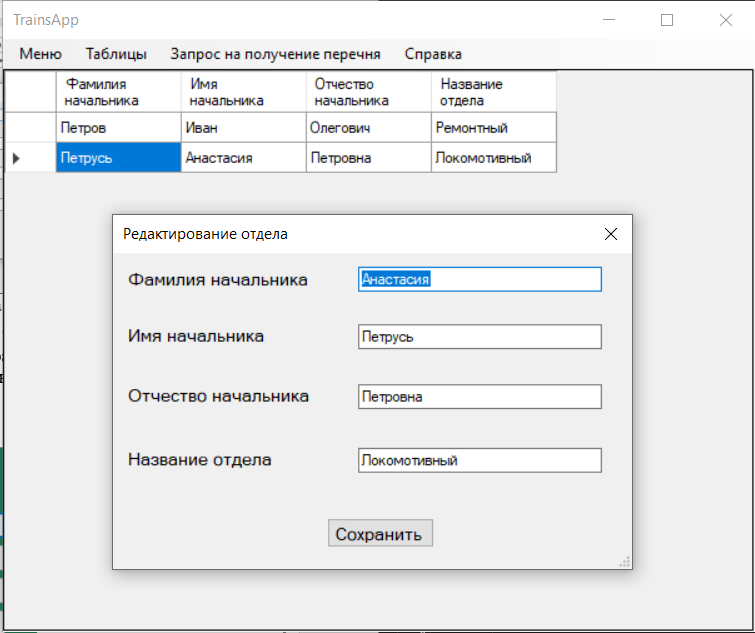


Рисунок Г.4 – Окно редактирования записи

После выбора пункта меню «Удалить» нам будет отображено окно подтверждения, если перед этим мы указали какую запись мы собираемся удалять. После чего, в зависимости от выбора, будет произведено удаление или же закрытие окна подтверждения. Подтверждающее окно представлено на рисунке Г.5.

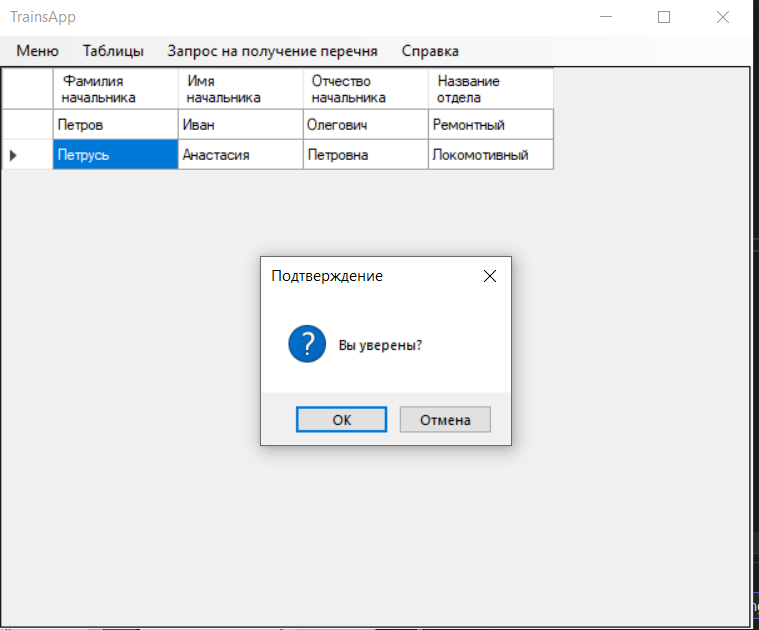


Рисунок Г.5 – Окно подтверждения удаления

Мы имеем возможность произвести выборку по конкретным полям таблицы. На рисунке Г.6 представлено окно, появлению которого предшествует нажатие на пункт «Работников» выпадающего меню «Запрос на получение перечня». На рисунке Г.7 представлен результат ввода данных в текстовое поле и нажатие кнопки «Работники по полу»

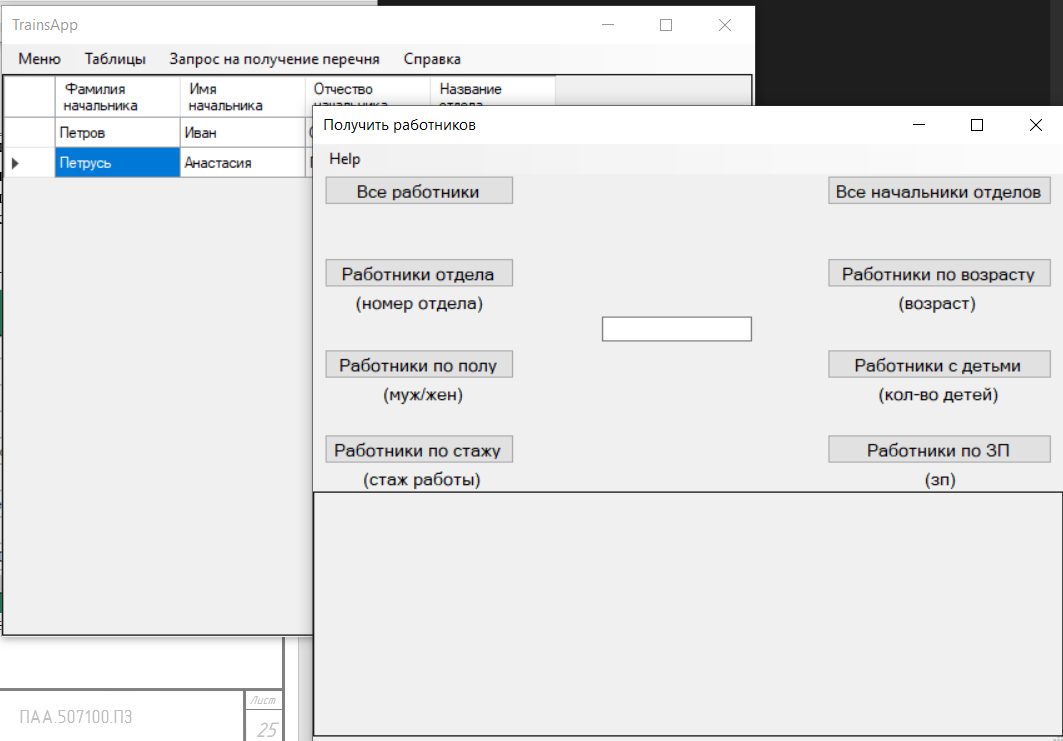


Рисунок Г.6 – Окно для запросов к таблице «Работники»

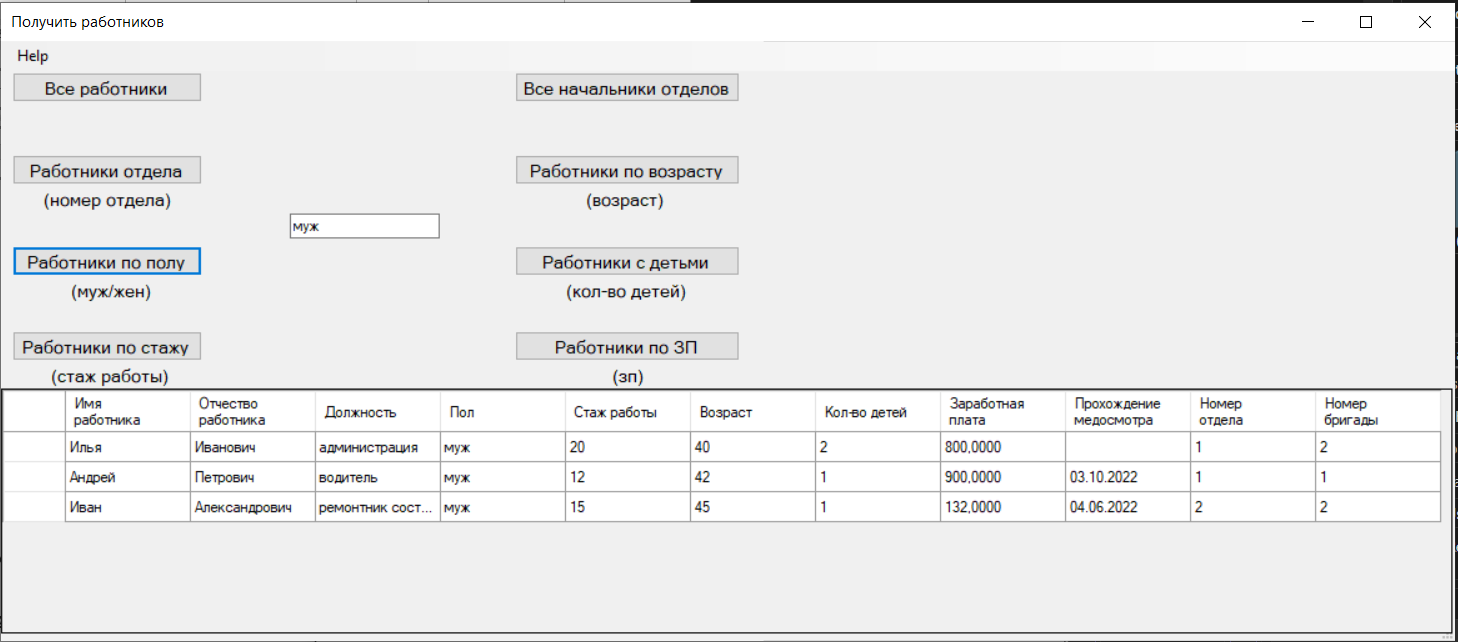


Рисунок Г.7 – Результат выборки по полу «Пол»

Любые изменения будут сохранятся в используемой базе данных.