минобрнауки россии

федеральное государственное бюджетное образовательное учреждение

высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

|  |  |
| --- | --- |
| Институт (факультет) | Институт Информационных Технологий |
| Кафедра | Математического и Программного Обеспечения ЭВМ |

**КУРСОВАЯ РАБОТА**

|  |
| --- |
| по дисциплине Проектирование баз данных |

|  |  |
| --- | --- |
| на тему | Проектирование базы данных общественного транспорта  г. Череповца |

|  |
| --- |
| Выполнил студент группы |
| 1ПИб-02-3оп-22 |
| направление подготовки (специальности) |
| 09.03.04, Программная инженерия |
| *шифр, наименование* |
| Маркелов Сергей Александрович |
| *фамилия, имя, отчество* |

|  |
| --- |
| Руководитель |
| Селяничев Олег Леонидович |
| *фамилия, имя, отчество* |
| доцент, кандидат технических наук |
| *должность* |

|  |
| --- |
| Дата представления работы |
| «\_\_22\_\_»\_\_\_\_\_мая\_\_\_\_\_\_\_\_\_\_\_2025 г. |
|  |
| Заключение о допуске к защите |
|  |
|  |
|  |
|  |
| Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| количество баллов |
| Подпись преподавателя\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Череповец, 2025

*Год*

Оглавление

[Введение 3](#_Toc198779997)

[1. Описание предметной области 4](#_Toc198779998)

[2. Инфологическое проектирование 7](#_Toc198779999)

[3. Даталогическое проектирование 11](#_Toc198780000)

[3.1. Нормализация 11](#_Toc198780001)

[3.1.1. Ненормализованная таблица 12](#_Toc198780002)

[3.1.2. Первая нормальная форма 15](#_Toc198780003)

[3.1.3. Вторая нормальная форма 18](#_Toc198780004)

[3.1.4. Третья нормальная форма 25](#_Toc198780005)

[3.2. Логическая модель 34](#_Toc198780006)

[3.3. Физическая модель 40](#_Toc198780007)

[4. Описание программы 49](#_Toc198780008)

[Заключение 63](#_Toc198780009)

[Список литературы 64](#_Toc198780010)

[Приложение 1. Техническое задание 67](#_Toc198780011)

[Приложение 2. Листинг 72](#_Toc198780012)

[Приложение 3. Руководство пользователя 94](#_Toc198780013)

# Введение

В настоящее время общественный транспорт играет важную роль в обеспечении мобильности населения и развитии городской инфраструктуры. Он позволяет людям быстро и удобно добираться до работы, учебных заведений, учреждений здравоохранения и других объектов городской среды. Качественная организация работы общественного транспорта напрямую влияет на уровень комфорта жизни в городе.

Для эффективной организации транспортной системы необходимы современные информационные решения, позволяющие хранить и обрабатывать данные о маршрутах, расписаниях и остановках. Наличие удобного доступа к этой информации особенно важно для планирования поездок пассажирами.

На данный момент в городе Череповце отсутствуют доступные для пассажиров базы данных общественного транспорта. Информация о маршрутах и расписаниях общественного транспорта доступна в основном через интернет-сервисы и мобильные приложения других видов (карты маршрутов, онлайн-табло прибытия транспорта). Однако такие средства получения данных содержат лишь часть необходимой информации и не всегда удобны для всех категорий пользователей, особенно в случаях отсутствия доступа к интернету.

Решением данной проблемы может стать установка терминалов на остановках и в салонах автобусов и трамваев. Программное обеспечение этих терминалов будет предоставлять доступ к базе данных, в которой в одном месте будет собрана актуальная информация о маршрутах, расписаниях, тарифах. Это обеспечит удобный доступ к данным для всех пассажиров, независимо от их технических возможностей.

Целью данной курсовой работы является создание базы данных, которая содержит информацию о маршрутах, остановках и расписаниях общественного транспорта г. Череповца и разработка программы, позволяющей работать с этой базой данных. Программа предназначена для установки на терминалах, которые можно размещать на остановках и в самих транспортных средствах. Это позволит обеспечить широкий доступ к необходимой информации и повысить удобство пользования общественным транспортом для всех жителей и гостей города.

# Описание предметной области

Общественный транспорт – это организованная система регулярных перевозок пассажиров по утверждённым маршрутам с использованием специально предназначенных транспортных средств (автобусы, троллейбусы, трамваи, метро, пригородные поезда, речные и морские паромы и др.) [14].

Он работает по фиксированным расписаниям и маршрутам, обеспечивая доступ к услугам для всех слоёв населения вне зависимости от их социального положения или уровня дохода. Цель общественного транспорта – обеспечить доступность перемещения для населения, снизить нагрузку на дорожную сеть, сократить выбросы загрязняющих веществ за счёт уменьшения использования личного автотранспорта, а также способствовать развитию городской инфраструктуры. Общественный транспорт может быть полностью или частично финансируемым государством (муниципальным или региональным бюджетом) либо частными компаниями, иногда сочетая оба источника.

Система городского общественного транспорта города Череповца включает в себя автобусные и трамвайные маршруты.

Автобусный транспорт в Череповце начал развиваться в середине XX века, когда возникла необходимость соединить быстрорастущие жилые районы с промышленными предприятиями и другими частями города. Первый регулярный автобусный маршрут был запущен в 1951 году, и с тех пор сеть маршрутов активно расширялась, охватывая всё больше микрорайонов [3]. На данный момент в городе функционирует разветвлённая сеть автобусных маршрутов, охватывающая все районы города. Основным перевозчиком является муниципальное предприятие «Автоколонна №1456», которое эксплуатирует автобусы большой вместимости марок НефАЗ, ЛиАЗ и МАЗ. Также услуги по пассажирским перевозкам предоставляет частный перевозчик «Новотранс». Помимо этого, в городе имеется автовокзал, который обслуживает пригородные, междугородние и межрегиональные маршруты [24].

Трамвайное движение в Череповце открылось 19 октября 1956 года и сохраняется до сих пор [4]. Трамвайная сеть играет важную роль в обслуживании работников промышленных предприятий, в первую очередь ПАО «Северсталь», а также соединяет густонаселённые жилые микрорайоны с центром города. Обслуживает трамвайную сеть предприятие МУП «Электротранс». Подвижной состав представлен современными вагонами моделей «Львёнок» и «Богатырь». Также до сих пор эксплуатируются несколько трамваев модели КТМ-5 1984-1992 годов выпуска. Несмотря на относительно небольшой размер сети, трамвай остаётся востребованным видом транспорта среди горожан [25].

В 1990-х годах также велось строительство троллейбусной сети, однако из-за экономических трудностей проект был заброшен. Остатки инфраструктуры были окончательно демонтированы в 2008 году, и в настоящее время развитие троллейбусного движения в городе не планируется [23].

Маршрутная сеть г. Череповца включает около 30 направлений, связывающих жилые кварталы друг с другом, а также с промышленными зонами и пригородными поселками, что позволяет жителям города комфортно добираться до центра города, рабочих мест, учебных заведений и других важных объектов инфраструктуры.

Для контроля за движением транспорта применяются спутниковые системы слежения GPS и ГЛОНАСС, благодаря чему обеспечивается контроль соблюдения расписания и оптимизация работы маршрутов. Диспетчеры в режиме реального времени могут видеть местоположение подвижного состава, что позволяет более эффективно реагировать на отклонения от графика.

Для повышения удобства пользования общественным транспортом в Череповце активно внедряются цифровые технологии. На многих остановках города установлены электронные табло, которые информируют пассажиров о времени прибытия следующего автобуса или трамвая. Также пассажиры могут воспользоваться сервисами Яндекс.Карты [15] и 35bus.ru [8] для отслеживания положения автобусов и трамваев в реальном времени, просмотра их траектории движения и расписания, а также планирования маршрутов своих поездок.

Однако не у всех пассажиров есть постоянный доступ к интернету, и не все могут использовать мобильные сервисы для отслеживания маршрутов и времени прибытия транспорта. Это создаёт трудности для части пользователей, которым необходима актуальная информация о маршрутах и расписании общественного транспорта. В связи с этим возникает необходимость разработки базы данных общественного транспорта, доступ к которой осуществлялся бы через терминалы на остановках и в транспортных средствах, не требуя доступа к интернету.

В базе данных для каждого маршрута необходимо отразить:

* номер маршрута;
* тип транспорта (автобус или трамвай);
* тип маршрута (городской, пригородный, междугородний, межрегиональный, служебный, заказной или экскурсионный);
* наименование начальной и конечной остановки;
* наименование улиц, на которых находится начальная/конечная остановка;
* наименование населенных пунктов, в которых находится начальная/конечная остановка;
* наименование перевозчика;
* класс транспорта (особо малый, малый, средний, большой или особо большой);
* тип пола (низкопольный, полунизкопольный или высокопольный);
* тип топлива (бензин, дизель, газ или электричество);
* наличие в транспортных средствах дополнительных опций, таких как кондиционер, пандус для маломобильных пассажиров, точка доступа Wi-Fi, USB-зарядки для телефонов, автоинформатор и информационные экраны;
* тарифы – наличный, безналичный и льготный, а также возможность поездки по проездному и бесплатной пересадки с одного маршрута на другой;
* способы оплаты проезда (наличные, банковская, транспортная карты или QR-код);
* места оплаты проезда (у кондуктора, через валидатор, у водителя, через автомат, на автовокзале или онлайн);
* расписание: в какие месяца и дни недели работает маршрут, по какому расписанию ходит (интервальному или точному), интервалы движения либо время отправления от конечной (в зависимости от вида расписания).

# Инфологическое проектирование

Моделирование данных – это процесс создания структурированного представления данных, который определяет, как данные организованы, как они хранятся и как взаимодействуют [9]. Существуют следующие типы моделей данных:

* концептуальная модель – описание сущностей и их связей;
* логическая модель – детализация концептуальной модели (добавляются атрибуты, типы данных, ключи), но без привязки к конкретной системе управления базами данных (СУБД);
* физическая модель – реализация логической модели в конкретной СУБД.

Инфологическое проектирование базы данных (БД) – это этап проектирования, на котором создается концептуальная модель данных, отражающая предметную область без привязки к конкретной системе СУБД [18].

Цель инфологического проектирования – создать непротиворечивое, полное и понятное представление данных предметной области в виде концептуальной модели, которая точно отражает сущности, их атрибуты, взаимосвязи и бизнес-правила, обеспечивая основу для последующего даталогического и физического проектирования БД.

Этот этап позволяет формализовать требования к данным, исключить двусмысленности и избыточность и согласовать понимание структуры информации между заказчиками и разработчиками.

Перед началом проектирования необходимо выбрать, какая модель данных будет использоваться.

Модели данных – это способы организации, хранения и управления данными в информационных системах. Они определяют структуру данных, методы доступа к ним и правила их обработки [10]. К основным моделям данных относятся:

* модель, основанная на инвертированных списках (рис. 1), организует данные через специальные индексные структуры, где для каждого уникального значения атрибута хранится список указателей на соответствующие записи, что обеспечивает быстрый поиск, но требует постоянного обновления индексов и уступает реляционным системам в обработке сложных запросов с соединениями таблиц [17];

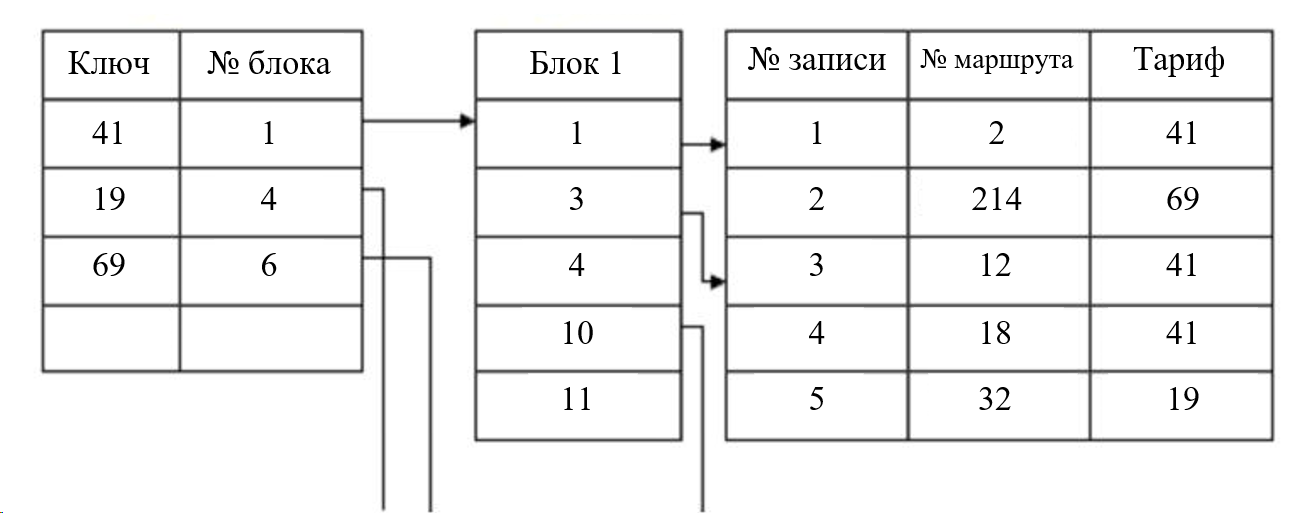


Рис. 1. Пример фрагмента инвертированного списка

* иерархическая модель (рис. 2) строится по принципу дерева, где каждый узел, за исключением корневого, имеет строго одного родителя, что делает её подходящей для файловых систем и XML-хранилищ, обеспечивая быстрый доступ к данным внутри иерархии, но усложняя работу с информацией вне её структуры [10];



Рис. 2. Иерархическая модель

* сетевая модель (рис. 3), являясь развитием иерархической, допускает наличие нескольких родителей у одного узла, что позволяло ей использоваться в ранних СУБД, однако сложность администрирования привела к её вытеснению реляционными системами [10];



Рис. 3. Сетевая модель

* реляционная модель (рис. 4) организует данные в виде таблиц, связанных между собой через ключи, и доминирует в современных СУБД благодаря своей гибкости и стандартизированному языку запросов SQL, хотя может сталкиваться с проблемами производительности при обработке сверхбольших объёмов данных [10];



Рис. 4. Реляционная модель

* объектно-ориентированная модель представляет данные в виде объектов, сочетающих атрибуты и методы, что делает её удобной для работы со сложными структурами, но ограниченная поддержка инструментария снижает её популярность [10].

Для реализации базы данных общественного транспорта г. Череповца была выбрана реляционная модель, так как она обеспечивает надежность, структурированность и гибкость. Данные хранятся в таблицах с четкой схемой, что предотвращает ошибки и дублирование. Язык структурированных запросов SQL позволяет выполнять сложные запросы.

Основными элементами инфологической модели данных являются сущности, их свойства (атрибуты), а также связи между ними.

Сущность – это самостоятельный элемент предметной области, который существует как целостная единица и требует отражения в информационной системе. Она представляет собой обобщенное представление о классе объектов реального мира или абстрактных понятий, значимых для решаемой задачи. Сущность проявляется через свои конкретные экземпляры, каждый из которых обладает уникальностью и самостоятельностью [21].

Атрибут – это характеристика или свойство, которое описывает и уточняет сущность в модели данных. Он представляет конкретный аспект сущности, позволяя выделять и различать её экземпляры. Атрибут имеет определённый тип (число, текст, дата и т. д.) и может быть обязательным или опциональным [21].

Связь – взаимодействие между сущностями, фиксирующее логическую зависимость между ними. Она определяет, как экземпляры одной сущности соотносятся с экземплярами другой [21]. Связи классифицируются по количеству соотносящихся экземпляров:

* один к одному (1:1) – уникальное соответствие, одному объекту первой сущности соответствует один объект второй сущности [21];
* один ко многим (1:M) – иерархическая зависимость, одному объекту первой сущности соответствует множество объектов второй сущности [21];
* многие ко многим (M:M) – множественное соответствие, многим объектам первой сущности соответствует множество объектов второй сущности [21].

При создании концептуальной модели необходимо выделить сущность, которая содержит в себе максимальное количество информации о рассматриваемой предметной области.

Для создания концептуальной модели по общественному транспорту можно выделить сущность «Маршруты» со следующими атрибутами:

* «Номер маршрута»;
* «Тип транспорта»;
* «Тип маршрута»;
* «Начальная и конечная остановка»;
* «Перевозчик»;
* «Работает»;
* «Характеристики транспортных средств»;
* «Тарифы»;
* «Способы оплаты»;
* «График движения»;
* «Расписание».

На этом этап инфологического проектирования завершен. После выбора модели данных и построения концептуальной модели можно переходить к следующему этапу – даталогическому проектированию.

# Даталогическое проектирование

Даталогическое проектирование – это этап проектирования БД, на котором определяется структура данных и связи между ними [5]. К основным задачам даталогического проектирования относятся:

* преобразование концептуальной модели в логическую модель данных;
* определение таблиц, атрибутов, ключей и связей;
* нормализация структуры данных.

Для выбранной в ходе инфологического проектирования реляционной модели данных даталогическая модель будет состоять из схем отношений, а также связей между ними через ключи.

## Нормализация

Нормализация базы данных – это процесс организации структуры базы данных с целью уменьшения избыточности данных и обеспечения их логической целостности [11]. Она включает разбиение таблиц на более мелкие и установление между ними связей так, чтобы:

* каждая таблица хранила данные об одном типе объектов;
* минимизировались дублирующиеся данные;
* можно было вносить изменения без нарушения целостности.

Нормализация проводится по нормальным формам. Нормальные формы – это формальные правила, которым должна соответствовать структура таблиц в базе данных, чтобы избежать избыточности и аномалий при обновлении, удалении и вставке данных. Каждая нормальная форма – это уровень нормализации. Всего выделяют 6 нормальных форм [12]:

* первая нормальная форма (1НФ) – таблица находится в 1НФ, если все ее атрибуты атомарны (т. е. неделимы);
* вторая нормальная форма (2НФ) – таблица находится в 2НФ, если она уже находится в 1НФ и все неключевые атрибуты зависят от всего первичного ключа, а не его части;
* третья нормальная форма (3НФ) – таблица находится в 3НФ, если она уже находится в 2НФ и у нее нет транзитивных зависимостей между неключевыми атрибутами;
* нормальная форма Бойса-Кодда (НФБК);
* четвертая нормальная форма (4НФ);
* пятая нормальная форма (5НФ).

Каждая следующая нормальная форма в некотором смысле лучше, чем предыдущая. При переходе к следующей нормальной форме свойства предыдущих нормальных форм должны сохранятся.

### Ненормализованная таблица

Ненормализованная таблица – это таблица, которая не удовлетворяет ни одной из нормальных форм. В одну таблицу собираются все данные предметной области, независимо от их логической природы. В ней имеются повторяющиеся группы, возможна избыточность и отсутствует структурное разделение на сущности и связи.

Любые изменения, например, названия перевозчика или тарифов, потребуют правки в каждой строке, где они используются. При изменении значений, встречающихся в нескольких строках, отсутствует гарантия того, что оно будет изменено везде одинаково. Это может привести к возникновению ошибок. Кроме того, при большом объёме данных таблица становится громоздкой и плохо масштабируемой. Она затрудняет построение запросов, поскольку не позволяет использовать средства реляционной алгебры полноценно.

В табл. 1 представлена ненормализованная таблица для сущности «Маршруты», которая была выделена на этапе инфологического проектирования. Атрибуты сущности также были выделены в ходе инфологического проектирования.

Таблица 1

Маршруты

13

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № марш. | Тип транс-порта | Тип маршрута | Начальная и конечная остановка | Перевозчик | Рабо-тает | Характеристики транспортных средств | Тарифы |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1 | Автобус | Городской | Боршодская улица (Боршодская улица, Череповец) – ЧЛМЗ (Улица Стройиндустрии, Череповец) | Автоколонна №1456 | + | Большого класса вместимости, низкопольные, на газомоторном топливе. Доп. опции: кондиционер, пандус, автоинформатор | Наличный – 41, безналичный – 38, льготный – 30, проездные, бесплатная пересадка |
| 1к | Автобус | Городской | Кладбище №5 (Боршодская улица, Череповец) – ЧЛМЗ (Улица Стройиндустрии, Череповец) | Автоколонна №1456 | + | Большого класса вместимости, низкопольные, на газомоторном топливе. Доп. опции: кондиционер, пандус, автоинформатор | Наличный – 41, безналичный – 38, льготный – 30, проездные, бесплатная пересадка |
| 2 | Автобус | Городской | Олимпийская улица (проспект Победы, Череповец) – ЧЛМЗ (Улица Стройиндустрии, Череповец) | Автоколонна №1456 | + | Большого класса вместимости, низкопольные, на газомоторном топливе. Доп. опции: кондиционер, пандус, автоинформатор | Наличный – 41, безналичный – 38, льготный – 30, проездные, бесплатная пересадка |
| 3 | Автобус | Городской | Улица Гоголя (улица Гоголя, Череповец) – ЧЛМЗ (Улица Стройиндустрии, Череповец) | Автоколонна №1456 | + | Малого класса вместимости, высокопольные, на бензине. Доп. опции: нет | Наличный – 41, безналичный – 38, льготный – 30, проездные, бесплатная пересадка |
| 4 | Автобус | Городской | Санаторий Адонис (Олимпийская улица, Череповец) – Ленинградская улица (Ленинградская улица, Череповец) | Автоколонна №1456 | + | Большого класса вместимости, низкопольные, на газомоторном топливе. Доп. опции: кондиционер, пандус, автоинформатор | Наличный – 41, безналичный – 38, льготный – 30, проездные, бесплатная пересадка |
| 4п | Автобус | Городской | Восточная улица (Восточная улица, Череповец) – Ленинградская улица (Ленинградская улица, Череповец) | Автоколонна №1456 | + | Большого класса вместимости, низкопольные, на газомоторном топливе. Доп. опции: кондиционер, пандус, автоинформатор | Наличный – 41, безналичный – 38, льготный – 30, проездные, бесплатная пересадка |
| 5 | Автобус | Городской | Солнечная улица (улица Краснодонцев, Череповец) – Азотный комплекс (Северное шоссе, Череповец) | Автоколонна №1456 | + | Большого класса вместимости, низкопольные, на газомоторном топливе. Доп. опции: кондиционер, пандус, автоинформатор | Наличный – 41, безналичный – 38, льготный – 30, проездные, бесплатная пересадка |

Продолжение табл. 1

|  |  |  |
| --- | --- | --- |
| Способы оплаты | График движения | Расписание |
| 9 | 10 | 11 |
| Наличными, банковской картой, транспортной картой, по QR-коду. Оплата через валидатор или у водителя | Круглогодично (Янв, Фев, Мар, Апр, Май, Июн, Июл, Авг, Сен, Окт, Нояб, Дек), ежедневно (Пн, Вт, Ср, Чт, Пт, Сб, Вс) | Интервальное расписание. Ежедневно с 06:00 до 19:37, в утренний час пик (05:00-10:00) – 25 мин, в вечерний час пик (15:00-19:00) – 25 мин, в вечерний межпик (19:00-00:00) – 25 мин |
| Наличными, банковской картой, транспортной картой, по QR-коду. Оплата через валидатор или у водителя | Круглогодично (Янв, Фев, Мар, Апр, Май, Июн, Июл, Авг, Сен, Окт, Нояб, Дек), ежедневно (Пн, Вт, Ср, Чт, Пт, Сб, Вс) | Интервальное расписание. Ежедневно с 07:03 до 15:38, в утренний час пик (05:00-10:00) – 25 мин, в дневной межпик (10:00-15:00) – 25 мин |
| Наличными, банковской картой, транспортной картой, по QR-коду. Оплата через валидатор или у водителя | Круглогодично (Янв, Фев, Мар, Апр, Май, Июн, Июл, Авг, Сен, Окт, Нояб, Дек), ежедневно (Пн, Вт, Ср, Чт, Пт, Сб, Вс) | Интервальное расписание. По будням с 05:35 до 23:28, в утренний час пик (05:00-10:00) – 10 минут, в дневной межпик (10:00-15:00) – 12 минут, в вечерний час пик (15:00-19:00) – 10 минут, в вечерний межпик (19:00-00:00) – 20 минут. По выходным с 05:35 до 23:23, в утренний час пик (05:00-10:00) – 20 минут, в дневной межпик (10:00-15:00) – 12 минут, в вечерний час пик (15:00-19:00) – 15 минут, в вечерний межпик (19:00-00:00) – 20 минут  14 |
| Наличными, банковской картой, транспортной картой, по QR-коду. Оплата у кондуктора | Круглогодично (Янв, Фев, Мар, Апр, Май, Июн, Июл, Авг, Сен, Окт, Нояб, Дек), ежедневно (Пн, Вт, Ср, Чт, Пт, Сб, Вс) | Интервальное расписание. По будням с 05:45 до 20:21, в утренний час пик (05:00-10:00) – 15 минут, в дневной межпик (10:00-15:00) – 30 минут, в вечерний час пик (15:00-19:00) – 15 минут, в вечерний межпик (19:00-00:00) – 15 минут. По выходным с 06:00 до 20:20, в утренний час пик (05:00-10:00) – 20 минут, в дневной межпик (10:00-15:00) – 40 минут, в вечерний час пик (15:00-19:00) – 20 минут, в вечерний межпик (19:00-00:00) – 20 минут |
| Наличными, банковской картой, транспортной картой, по QR-коду. Оплата через валидатор или у водителя | Круглогодично (Янв, Фев, Мар, Апр, Май, Июн, Июл, Авг, Сен, Окт, Нояб, Дек), ежедневно (Пн, Вт, Ср, Чт, Пт, Сб, Вс) | Интервальное расписание. По будням с 05:25 до 22:54, весь день (05:00-00:00) – 20 минут. По выходным с 05:25 до 22:46, в утренний час пик (05:00-10:00) – 20 минут, в дневной межпик (10:00-15:00) – 20 минут, в вечерний час пик (15:00-19:00) – 20 минут, в вечерний межпик (19:00-00:00) – 25 минут |
| Наличными, банковской картой, транспортной картой, по QR-коду. Оплата через валидатор или у водителя | Круглогодично (Янв, Фев, Мар, Апр, Май, Июн, Июл, Авг, Сен, Окт, Нояб, Дек), ежедневно (Пн, Вт, Ср, Чт, Пт, Сб, Вс) | Точное расписание. По будням с 05:37 до 18:38, в прямом направлении: 05:37, 06:09, 06:57, 08:16, 10:14,11:18,13:48, 16:10, 17:48, 18:38, в обратном направлении: 05:53, 07:06, 12:34, 15:04, 16:29, 17:50. По выходным с 05:48 до 19:10, в прямом направлении: 05:48, 07:03, 08:37, 09:27, 10:42, 11:57, 14:02, 15:42, 18:13, 19:10, в обратном направлении: 05:53, 07:20, 08:12, 10:41, 12:46, 14:26, 16:06, 18:04 |
| Наличными, банковской картой, транспортной картой, по QR-коду. Оплата через валидатор или у водителя | Круглогодично (Янв, Фев, Мар, Апр, Май, Июн, Июл, Авг, Сен, Окт, Нояб, Дек), ежедневно (Пн, Вт, Ср, Чт, Пт, Сб, Вс) | Точное расписание. По будням с 06:12 до 08:07, в прямом направлении: 06:12, 06:32, 07:14, в обратном направлении: 07:03, 07:23, 08:07. По выходным с 06:10 до 08:07, в прямом направлении: 06:10, 06:30, 07:18, в обратном направлении: 07:03, 07:23, 08:07 |

### Первая нормальная форма

Определение: отношение находится в первой нормальной форме, когда значение ячеек является атомарным (неделимым) и атрибуты имеют уникальные названия.

Критика: в ненормализованной таблице (см. табл. 1) неатомарными являются следующие атрибуты:

* Начальная и конечная остановка;
* Характеристики транспортных средств;
* Тарифы;
* Способы оплаты;
* График движения;
* Расписание.

Предложение: для приведения ненормализованной таблицы в 1 нормальную форму необходимо выполнить следующие преобразования:

* атрибут «Начальная и конечная остановка» разделить на атрибуты «Начальная остановка», «Улица начальной остановки», «Населенный пункт начальной остановки», «Конечная остановка», «Улица конечной остановки», «Населенный пункт конечной остановки»;
* атрибут «Характеристики транспортных средств» разделить на атрибуты «Класс транспорта», «Тип пола», «Тип топлива», «Кондиционер», «Пандус», «Wi-Fi», «USB-зарядки», «Автоинформатор», «Информационные экраны»;
* атрибут «Тарифы» необходимо разделить на атрибуты «Наличный тариф», «Безналичный тариф», «Льготный тариф», «Проездные», «Бесплатная пересадка»;
* атрибут «Способы оплаты» разделить на атрибуты «Оплата наличными», «Оплата банковской картой», «Оплата транспортной картой», «Оплата по QR-коду», «Оплата у кондуктора», «Оплата через валидатор», «Оплата у водителя», «Оплата через автомат», «Оплата на автовокзале», «Оплата онлайн»;
* атрибут «График движения» необходимо разделить на атрибуты «Январь», «Февраль», «Март», «Апрель», «Май», «Июнь», «Июль», «Август», «Сентябрь», «Октябрь», «Ноябрь», «Декабрь», «Понедельник», «Вторник», «Среда», «Четверг», «Пятница», «Суббота», «Воскресенье»;
* атрибут «Расписание» необходимо разделить на атрибуты «Тип расписания», «Время начала работы», «Время окончания работы», «Начало интервала», «Конец интервала», «Интервал», «Направление», «Время отправления».

Отношение в первой нормальной форме представлено в табл. 2.

Таблица 2

Маршруты

16

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № марш. | Тип транспорта | Тип маршрута | Нач. остановка | Улица нач. остановки | Н. п. нач. остановки | Кон. остановка | Улица кон. остановки | Н. п. кон. остановки | Перевозчик | Работает | Класс транспорта |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 1 | Автобус | Городской | Боршодская улица | Боршодская улица | Череповец | ЧЛМЗ | Улица Стройиндустрии | Череповец | Автоколонна №1456 | + | Большой |
| 1 | Автобус | Городской | Боршодская улица | Боршодская улица | Череповец | ЧЛМЗ | Улица Стройиндустрии | Череповец | Автоколонна №1456 | + | Большой |
| 1 | Автобус | Городской | Боршодская улица | Боршодская улица | Череповец | ЧЛМЗ | Улица Стройиндустрии | Череповец | Автоколонна №1456 | + | Большой |
| 1к | Автобус | Городской | Кладбище №5 | Боршодская улица | Череповец | ЧЛМЗ | Улица Стройиндустрии | Череповец | Автоколонна №1456 | + | Большой |
| 1к | Автобус | Городской | Кладбище №5 | Боршодская улица | Череповец | ЧЛМЗ | Улица Стройиндустрии | Череповец | Автоколонна №1456 | + | Большой |
| 2 | Автобус | Городской | Олимпийская улица | Проспект Победы | Череповец | ЧЛМЗ | Улица Стройиндустрии | Череповец | Автоколонна №1456 | + | Большой |
| 2 | Автобус | Городской | Олимпийская улица | Проспект Победы | Череповец | ЧЛМЗ | Улица Стройиндустрии | Череповец | Автоколонна №1456 | + | Большой |

Продолжение табл. 2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Тип пола | Тип топлива | Конди-ционер | Пандус | Wi-Fi | USB-зарядки | Автоин- форматор | Инф. экраны | Наличный тариф | Безналичный тариф | Льготный тариф | Проездные | Бесплатная пересадка | Оплата наличными |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| Низко-  польный | Газ | + | + |  |  | + |  | 41 | 38 | 30 | + | + | + |
| Низко-  польный | Газ | + | + |  |  | + |  | 41 | 38 | 30 | + | + | + |
| Низко-  польный | Газ | + | + |  |  | + |  | 41 | 38 | 30 | + | + | + |
| Низко-  польный | Газ | + | + |  |  | + |  | 41 | 38 | 30 | + | + | + |
| Низко-  польный | Газ | + | + |  |  | + |  | 41 | 38 | 30 | + | + | + |
| Низко-  польный | Газ | + | + |  |  | + |  | 41 | 38 | 30 | + | + | + |
| Низко-  польный | Газ | + | + |  |  | + |  | 41 | 38 | 30 | + | + | + |

Продолжение табл. 2

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Оплата банк. картой | Оплата транс. картой | Оплата по QR-коду | Оплата у кондук-тора | Оплата ч-з вали-датор | Оплата у води-теля | Оплата ч-з автомат | Оплата на авто-вокзале | Оплата он-лайн | Янв | Фев | Мар | Апр | Май | Июн | Июл | Авг | Сен | Окт |
| 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 |
| + | + | + |  | + | + |  |  |  | + | + | + | + | + | + | + | + | + | + |
| + | + | + |  | + | + |  |  |  | + | + | + | + | + | + | + | + | + | + |
| + | + | + |  | + | + |  |  |  | + | + | + | + | + | + | + | + | + | + |
| + | + | + |  | + | + |  |  |  | + | + | + | + | + | + | + | + | + | + |
| + | + | + |  | + | + |  |  |  | + | + | + | + | + | + | + | + | + | + |
| + | + | + |  | + | + |  |  |  | + | + | + | + | + | + | + | + | + | + |
| + | + | + |  | + | + |  |  |  | + | + | + | + | + | + | + | + | + | + |

Продолжение табл. 2

17

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Нояб | Дек | Пн | Вт | Ср | Чт | Пт | Сб | Вс | Тип расписания | Время начала работы | Время оконча-ния работы | Начало интервала | Конец интервала | Интервал | Направление | Время отправления |
| 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 |
| + | + | + | + | + | + | + | + | + | Интервальное | 06:00 | 19:37 | 05:00 | 10:00 | 25 | - | - |
| + | + | + | + | + | + | + | + | + | Интервальное | 06:00 | 19:37 | 15:00 | 19:00 | 25 | - | - |
| + | + | + | + | + | + | + | + | + | Интервальное | 06:00 | 19:37 | 19:00 | 00:00 | 25 | - | - |
| + | + | + | + | + | + | + | + | + | Интервальное | 07:03 | 15:38 | 05:00 | 10:00 | 25 | - | - |
| + | + | + | + | + | + | + | + | + | Интервальное | 07:03 | 15:38 | 10:00 | 15:00 | 25 | - | - |
| + | + | + | + | + | + | + |  |  | Интервальное | 05:35 | 23:28 | 05:00 | 10:00 | 10 | - | - |
| + | + | + | + | + | + | + |  |  | Интервальное | 05:35 | 23:28 | 10:00 | 15:00 | 12 | - | - |

### Вторая нормальная форма

Определение: отношение находится во второй нормальной форме, если оно находится в первой нормальной форме, и каждый неключевой атрибут полностью функционально зависит от первичного ключа.

Критика: в отношении имеются частичные зависимости, т. е. зависимости неключевых атрибутов от части составного ключа:

* атрибуты «Начальная остановка», «Улица начальной остановки», «Населенный пункт начальной остановки», «Конечная остановка», «Улица Конечной остановки», «Населенный пункт конечной остановки» зависят от атрибута «Номер маршрута»;
* атрибуты «Класс транспорта», «Тип пола», «Тип топлива», «Кондиционер», «Пандус», «Wi-Fi», «USB-зарядки», «Автоинформатор», «Информационные экраны» зависят от атрибута «Номер маршрута»;
* атрибуты «Наличный тариф», «Безналичный тариф», «Льготный тариф», «Проездные», «Бесплатная пересадка» зависят от атрибута «Номер маршрута»;
* атрибуты «Оплата наличными», «Оплата банковской картой», «Оплата транспортной картой», «Оплата по QR-коду» зависят от атрибута «Номер маршрута»;
* атрибуты «Оплата у кондуктора», «Оплата через валидатор», «Оплата у водителя», «Оплата через автомат», «Оплата на автовокзале», «Оплата онлайн» зависят от атрибута «Номер маршрута»;
* атрибуты «Тип расписания», «Время начала работы», «Время окончания работы» зависят от атрибута «Номер маршрута»;
* атрибуты «Январь», «Февраль», «Март», «Апрель», «Май», «Июнь», «Июль», «Август», «Сентябрь», «Октябрь», «Ноябрь», «Декабрь» зависят от атрибута «Номер маршрута»;
* атрибуты «Понедельник», «Вторник», «Среда», «Четверг», «Пятница», «Суббота», «Воскресенье» зависят от атрибута «Номер маршрута»;

Предложение: выделить вышеперечисленные группы атрибутов в отдельные сущности:

* сущность «Остановки» (атрибуты: «Остановка», «Улица», «Населенный пункт»);
* сущность «Транспортные средства» (атрибуты: «Класс транспорта», «Тип пола», «Тип топлива», «Кондиционер», «Пандус», «Wi-Fi», «USB-зарядки», «Автоинформатор», «Информационные экраны»);
* сущность «Тарифы» (атрибуты: «Наличный тариф», «Безналичный тариф», «Льготный тариф», «Проездные», «Бесплатная пересадка»);
* сущность «Способы оплаты» (атрибуты: «Оплата наличными», «Оплата банковской картой», «Оплата транспортной картой», «Оплата по QR-коду»);
* сущность «Места оплаты» (атрибуты: «Оплата у кондуктора», «Оплата через валидатор», «Оплата у водителя», «Оплата через автомат», «Оплата на автовокзале», «Оплата онлайн»);
* сущность «Время работы» (атрибуты: «График по месяцам», «График по дням недели», «Тип расписания», «Время начала работы», «Время окончания работы»);
* сущность «График по месяцам» (атрибуты: «Январь», «Февраль», «Март», «Апрель», «Май», «Июнь», «Июль», «Август», «Сентябрь», «Октябрь», «Ноябрь», «Декабрь»);
* сущность «График по дням недели» (атрибуты: «Понедельник», «Вторник», «Среда», «Четверг», «Пятница», «Суббота», «Воскресенье»);
* сущность «График по времени суток» (атрибуты: «Начало интервала», «Конец интервала»);
* сущность «Интервальное расписание» (атрибуты: «Время работы», «Время суток», «Интервал»);
* сущность «Точное расписание» (атрибуты: «Время работы», «Направление», «Время отправления»).

Связь между сущностями реализуется через механизм ключей, который обеспечивает целостность и согласованность данных. В созданных сущностях помимо выделенных атрибутов добавляется новый, с названием вида «Код …», который является первичным ключом – уникальным идентификатором кортежа. Он гарантирует, что каждый кортеж однозначно определяется своим значением первичного ключа и не повторяется. В исходной сущности вместо выделенных атрибутов появляются атрибуты, являющиеся внешними ключами, которые ссылаются на первичные ключи других сущностей.

Например, атрибуты «Начальная остановка» и «Конечная остановка» сущности «Маршруты» являются внешними ключами, ссылающимися на первичный ключ «Код остановки» сущности «Остановки».

Отношения во второй нормальной форме представлены в табл. 3-14.

Таблица 3

Маршруты

20

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код марш- рута | Номер марш- рута | Тип транс- порта | Тип маршрута | Начальная остановка | Конечная остановка | Перевозчик | Работает | Транс- портные средства | Тариф | Способы оплаты | Места оплаты |
| 1 | 1 | Автобус | Городской | 17 | 13 | Автоколонна №1456 | + | 8 | 1 | 4 | 2 |
| 2 | 1к | Автобус | Городской | 18 | 13 | Автоколонна №1456 | + | 8 | 1 | 4 | 2 |
| 3 | 2 | Автобус | Городской | 1 | 13 | Автоколонна №1456 | + | 8 | 1 | 4 | 2 |
| 4 | 3 | Автобус | Городской | 7 | 13 | Автоколонна №1456 | + | 1 | 1 | 4 | 1 |
| 5 | 4 | Автобус | Городской | 4 | 19 | Автоколонна №1456 | + | 8 | 1 | 4 | 2 |
| 6 | 4п | Автобус | Городской | 20 | 19 | Автоколонна №1456 | + | 8 | 1 | 4 | 2 |
| 7 | 5 | Автобус | Городской | 5 | 21 | Автоколонна №1456 | + | 8 | 1 | 4 | 2 |

Таблица 4

Остановки

|  |  |  |  |
| --- | --- | --- | --- |
| Код остановки | Остановка | Улица | Населенный пункт |
| 1 | Олимпийская улица | Проспект Победы | Череповец |
| 2 | Олимпийская улица | Олимпийская улица | Череповец |
| 3 | Проспект Победы | Олимпийская улица | Череповец |
| 4 | Санаторий Адонис | Олимпийская улица | Череповец |
| 5 | Солнечная улица | Улица Краснодонцев | Череповец |
| 6 | Макаринская роща | Улица Командарма Белова | Череповец |
| 7 | Улица Гоголя | Улица Гоголя | Череповец |

Таблица 5

Транспортные средства

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код ТС | Класс транспорта | Тип пола | Тип топлива | Кондиционер | Пандус | Wi-Fi | USB-зарядки | Автоинформатор | Информационные экраны |
| 1 | Малой вместимости | Высокопольный | Бензин |  |  |  |  |  |  |
| 2 | Большой вместимости | Высокопольный | Бензин |  |  |  |  | + |  |
| 3 | Большой вместимости | Высокопольный | Дизель |  |  |  |  | + |  |
| 4 | Большой вместимости | Полунизкопольный | Бензин |  | + |  |  | + |  |
| 5 | Большой вместимости | Полунизкопольный | Дизель |  | + |  |  | + |  |
| 6 | Большой вместимости | Низкопольный | Дизель |  | + |  |  | + | 21 |
| 7 | Большой вместимости | Низкопольный | Газ |  | + |  |  | + |  |

Таблица 6

Тарифы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Код тарифа | Наличный | Безналичный | Льготный | Проездные | Бесплатная пересадка |
| 1 | 41 | 38 | 30 | + | + |
| 2 | 41 | 38 | 30 | + |  |
| 3 | 21 | 18 | 18 | + |  |
| 4 | 69 | 69 | 30 | + | + |

Таблица 7

Способы оплаты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код способа оплаты | Наличные | Банковская карта | Транспортная карта | QR-код |
| 1 | + |  |  |  |
| 2 | + | + |  |  |
| 3 | + | + | + |  |
| 4 | + | + | + | + |

Таблица 8

Места оплаты

22

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Код места оплаты | Кондуктор | Валидатор | Водитель | Автомат | Автовокзал | Онлайн |
| 1 | + |  |  |  |  |  |
| 2 |  | + | + |  |  |  |
| 3 |  | + |  | + |  |  |
| 4 | + | + |  |  |  |  |
| 5 |  |  | + |  |  |  |
| 6 |  |  |  |  | + |  |
| 7 |  |  |  |  | + | + |

Таблица 9

Время работы

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Код времени работы | Маршрут | График по месяцам | График по дням недели | Тип расписания | Время начала работы | Время окончания работы |
| 1 | 1 | 1 | 1 | Интервальное | 06:00 | 19:37 |
| 2 | 2 | 1 | 1 | Интервальное | 07:03 | 15:38 |
| 3 | 3 | 1 | 3 | Интервальное | 05:35 | 23:28 |
| 4 | 3 | 1 | 4 | Интервальное | 05:35 | 23:23 |
| 5 | 4 | 1 | 3 | Интервальное | 05:45 | 20:21 |
| 6 | 4 | 1 | 4 | Интервальное | 06:00 | 20:20 |
| 7 | 5 | 1 | 3 | Интервальное | 05:25 | 22:54 |

Таблица 10

График по месяцам

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код графика | Янв | Фев | Мар | Апр | Май | Июн | Июл | Авг | Сен | Окт | Нояб | Дек |
| 1 | + | + | + | + | + | + | + | + | + | + | + | + |
| 2 |  |  |  |  | + | + | + | + | + | + |  |  |
| 3 | + | + | + | + | + |  |  |  | + | + | + | + |
| 4 | + | + | + | + |  |  |  |  |  |  | + | + |

Таблица 11

График по дням недели

23

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Код графика | Пн | Вт | Ср | Чт | Пт | Сб | Вс |
| 1 | + | + | + | + | + | + | + |
| 2 | + | + | + | + | + | + |  |
| 3 | + | + | + | + | + |  |  |
| 4 |  |  |  |  |  | + | + |
| 5 |  |  |  |  |  | + |  |
| 6 |  |  |  |  |  |  | + |

Таблица 12

График по времени суток

|  |  |  |
| --- | --- | --- |
| Код графика | Начало интервала | Конец интервала |
| 1 | 05:00 | 10:00 |
| 2 | 10:00 | 15:00 |
| 3 | 15:00 | 19:00 |
| 4 | 19:00 | 00:00 |
| 5 | 05:00 | 00:00 |

Таблица 13

Интервальное расписание

|  |  |  |  |
| --- | --- | --- | --- |
| Код расписания | Время работы | Время суток | Интервал |
| 1 | 1 | 1 | 25 |
| 2 | 1 | 3 | 25 |
| 3 | 1 | 4 | 25 |
| 4 | 2 | 1 | 25 |
| 5 | 2 | 2 | 25 |
| 6 | 3 | 1 | 10 |
| 7 | 3 | 2 | 12 |

Таблица 14

Точное расписание

24

|  |  |  |  |
| --- | --- | --- | --- |
| Код расписания | Время работы | Направление | Время отправления |
| 1 | 9 | Прямое | 05:37 |
| 2 | 9 | Прямое | 06:09 |
| 3 | 9 | Прямое | 06:57 |
| 4 | 9 | Прямое | 08:16 |
| 5 | 9 | Прямое | 10:14 |
| 6 | 9 | Прямое | 11:18 |
| 7 | 9 | Прямое | 13:48 |

### Третья нормальная форма

Определение: отношение находится в третьей нормальной форме в том случае, когда оно находится во второй нормальной форме, и каждый не ключевой атрибут не транзитивно зависит от первичного ключа.

Критика: некоторые атрибуты имеют транзитивную зависимость:

* «Тип транспорта», «Тип маршрута», «Перевозчик» в сущности «Маршруты»;
* «Улица» и «Населенный пункт» в сущности «Остановки»;
* «Класс транспорта», «Тип пола» и «Тип топлива» в сущности «Транспортные средства»;
* «Тип расписания» в сущности «Время работы»;
* «Направление движения» в сущности «Точное расписание».

Предложение: выделить вышеперечисленные атрибуты в отдельные сущности:

* атрибут «Тип транспорта» в сущность «Типы транспорта»;
* атрибут «Тип маршрута» в сущность «Типы маршрута»;
* атрибут «Перевозчик» в сущность «Перевозчики»;
* атрибут «Улица» в сущность «Улицы»;
* атрибут «Населенный пункт» в сущность «Населенные пункты»;
* атрибут «Класс транспорта» в сущность «Классы транспорта»;
* атрибут «Тип пола» в сущность «Типы пола»;
* атрибут «Тип топлива» в сущность «Типы топлива»;
* атрибут «Тип расписания» в сущность «Типы расписания»;
* атрибут «Направление движения» в сущность «Направления движения»;

Связь между сущностями обеспечивается через механизм ключей, так же, как и в 2НФ. Например, атрибут «Тип транспорта» в сущности «Маршруты» является внешним ключом, ссылающимся на первичный ключ «Код типа транспорта» сущности «Типы транспорта». А атрибут «Улица» сущности «Остановки» является внешним ключом, ссылающимся на первичный ключ «Код улицы» сущности «Улицы».

Отношения в третьей нормальной форме представлены в табл. 15-37.

На данном этапе процесс нормализации можно завершить, так как большинство проблем ненормализованной таблицы уже решены – избыточность данных устранена, логическая целостность данных обеспечена. Дальнейшая нормализация может только усложнить схему отношений и замедлить работу с данными.

После того, как таблица приведена к необходимой нормальной форме, можно переходить к построению логической модели.

Таблица 15

Маршруты

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код марш-рута | Номер маршрута | Тип транспорта | Тип маршрута | Начальная остановка | Конечная остановка | Перевозчик | Рабо-тает | Транс- портные средства | Тариф | Способы оплаты | Места оплаты |
| 1 | 1 | 1 | 1 | 17 | 13 | 1 | + | 8 | 1 | 4 | 2 |
| 2 | 1к | 1 | 1 | 18 | 13 | 1 | + | 8 | 1 | 4 | 2 |
| 3 | 2 | 1 | 1 | 1 | 13 | 1 | + | 8 | 1 | 4 | 2 |
| 4 | 3 | 1 | 1 | 7 | 13 | 1 | + | 1 | 1 | 4 | 1 |
| 5 | 4 | 1 | 1 | 4 | 19 | 1 | + | 8 | 1 | 4 | 2 |
| 6 | 4п | 1 | 1 | 20 | 19 | 1 | + | 8 | 1 | 4 | 2 |
| 7 | 5 | 1 | 1 | 5 | 21 | 1 | + | 8 | 1 | 4 | 2 |

Таблица 16

26

Тип транспорта

|  |  |
| --- | --- |
| Код типа транспорта | Тип транспорта |
| 1 | Автобус |
| 2 | Трамвай |
| 3 | Троллейбус |
| 4 | Маршрутное такси |
| 5 | Метрополитен |
| 6 | Речной трамвай |
| 7 | Электричка |

Таблица 17

Тип маршрута

|  |  |
| --- | --- |
| Код типа маршрута | Тип транспорта |
| 1 | Городской |
| 2 | Пригородный |
| 3 | Междугородний |
| 4 | Межрегиональный |
| 5 | Международный |
| 6 | Служебный |
| 7 | Заказной |

Таблица 18

Остановки

27

|  |  |  |
| --- | --- | --- |
| Код остановки | Остановка | Улица |
| 1 | Олимпийская улица | 1 |
| 2 | Олимпийская улица | 2 |
| 3 | Проспект Победы | 2 |
| 4 | Санаторий Адонис | 2 |
| 5 | Солнечная улица | 3 |
| 6 | Макаринская роща | 4 |
| 7 | Улица Гоголя | 5 |

Таблица 19

Улицы

|  |  |  |
| --- | --- | --- |
| Код улицы | Улица | Населенный пункт |
| 1 | Проспект Победы | 1 |
| 2 | Олимпийская улица | 1 |
| 3 | Улица Краснодонцев | 1 |
| 4 | Улица Командарма Белова | 1 |
| 5 | Улица Гоголя | 1 |
| 6 | Улица Ветеранов | 1 |
| 7 | Улица Мира | 1 |

Таблица 20

Населенные пункты

28

|  |  |
| --- | --- |
| Код населенного пункта | Населенный пункт |
| 1 | Череповец |
| 2 | Нова |
| 3 | Шайма |
| 4 | Романда |
| 5 | Тоншалово |
| 6 | СНТ Металлист |

Таблица 21

Перевозчики

|  |  |
| --- | --- |
| Код перевозчика | Перевозчик |
| 1 | Автоколонна №1456 |
| 2 | Новотранс |
| 3 | ЧереповецТрансАгенство |
| 4 | Электротранс |

Таблица 22

Транспортные средства

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код ТС | Класс транспорта | Тип пола | Тип топлива | Кондиционер | Пандус | Wi-Fi | USB-зарядки | Автоинформатор | Информационные экраны |
| 1 | 2 | 3 | 1 |  |  |  |  |  |  |
| 2 | 4 | 3 | 1 |  |  |  |  | + |  |
| 3 | 4 | 3 | 2 |  |  |  |  | + |  |
| 4 | 4 | 2 | 1 |  | + |  |  | + |  |
| 5 | 4 | 2 | 2 |  | + |  |  | + |  |
| 6 | 4 | 1 | 2 |  | + |  |  | + |  |
| 7 | 4 | 1 | 3 |  | + |  |  | + |  |

Таблица 23

Класс транспорта

29

|  |  |
| --- | --- |
| Код класса | Класс транспорта |
| 1 | Особо малой вместимости |
| 2 | Малой вместимости |
| 3 | Средней вместимости |
| 4 | Большой вместимости |
| 5 | Особо большой вместимости |

Таблица 24

Тип пола

|  |  |
| --- | --- |
| Код типа пола | Тип пола |
| 1 | Низкопольный |
| 2 | Полунизкопольный |
| 3 | Высокопольный |

Таблица 25

Тип топлива

|  |  |
| --- | --- |
| Код типа топлива | Тип топлива |
| 1 | Бензин |
| 2 | Дизель |
| 3 | Газ |
| 4 | Электричество |

Таблица 26

Тарифы

30

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Код тарифа | Наличный | Безналичный | Льготный | Проездные | Бесплатная пересадка |
| 1 | 41 | 38 | 30 | + | + |
| 2 | 41 | 38 | 30 | + |  |
| 3 | 21 | 18 | 18 | + |  |
| 4 | 69 | 69 | 30 | + | + |

Таблица 27

Способы оплаты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код способа оплаты | Наличные | Банковская карта | Транспортная карта | QR-код |
| 1 | + |  |  |  |
| 2 | + | + |  |  |
| 3 | + | + | + |  |
| 4 | + | + | + | + |

Таблица 28

Места оплаты

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Код места оплаты | Кондуктор | Валидатор | Водитель | Автомат | Автовокзал | Онлайн |
| 1 | + |  |  |  |  |  |
| 2 |  | + | + |  |  |  |
| 3 |  | + |  | + |  |  |
| 4 | + | + |  |  |  |  |
| 5 |  |  | + |  |  |  |
| 6 |  |  |  |  | + |  |
| 7 |  |  |  |  | + | + |

Таблица 29

Время работы

31

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Код времени работы | Маршрут | График по месяцам | График по дням недели | Тип расписания | Время начала работы | Время окончания работы |
| 1 | 1 | 1 | 1 | 1 | 06:00 | 19:37 |
| 2 | 2 | 1 | 1 | 1 | 07:03 | 15:38 |
| 3 | 3 | 1 | 3 | 1 | 05:35 | 23:28 |
| 4 | 3 | 1 | 4 | 1 | 05:35 | 23:23 |
| 5 | 4 | 1 | 3 | 1 | 05:45 | 20:21 |
| 6 | 4 | 1 | 4 | 1 | 06:00 | 20:20 |
| 7 | 5 | 1 | 3 | 1 | 05:25 | 22:54 |

Таблица 30

График по месяцам

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код графика | Янв | Фев | Мар | Апр | Май | Июн | Июл | Авг | Сен | Окт | Нояб | Дек |
| 1 | + | + | + | + | + | + | + | + | + | + | + | + |
| 2 |  |  |  |  | + | + | + | + | + | + |  |  |
| 3 | + | + | + | + | + |  |  |  | + | + | + | + |
| 4 | + | + | + | + |  |  |  |  |  |  | + | + |

Таблица 31

График по дням недели

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Код графика | Пн | Вт | Ср | Чт | Пт | Сб | Вс |
| 1 | + | + | + | + | + | + | + |
| 2 | + | + | + | + | + | + |  |
| 3 | + | + | + | + | + |  |  |
| 4 |  |  |  |  |  | + | + |
| 5 |  |  |  |  |  | + |  |
| 6 |  |  |  |  |  |  | + |

Таблица 32

Типы расписаний

32

|  |  |
| --- | --- |
| Код типа расписания | Тип расписания |
| 1 | Интервальное |
| 2 | Точное |

Таблица 33

Интервальное расписание

|  |  |  |  |
| --- | --- | --- | --- |
| Код расписания | Время работы | Время суток | Интервал |
| 1 | 1 | 1 | 25 |
| 2 | 1 | 3 | 25 |
| 3 | 1 | 4 | 25 |
| 4 | 2 | 1 | 25 |
| 5 | 2 | 2 | 25 |
| 6 | 3 | 1 | 10 |
| 7 | 3 | 2 | 12 |

Таблица 34

График по времени суток

|  |  |  |
| --- | --- | --- |
| Код графика | Начало интервала | Конец интервала |
| 1 | 05:00 | 10:00 |
| 2 | 10:00 | 15:00 |
| 3 | 15:00 | 19:00 |
| 4 | 19:00 | 00:00 |
| 5 | 05:00 | 00:00 |

Таблица 35

Точное расписание

33

|  |  |  |  |
| --- | --- | --- | --- |
| Код расписания | Время работы | Направление | Время отправления |
| 1 | 9 | 1 | 05:37 |
| 2 | 9 | 1 | 06:09 |
| 3 | 9 | 1 | 06:57 |
| 4 | 9 | 1 | 08:16 |
| 5 | 9 | 1 | 10:14 |
| 6 | 9 | 1 | 11:18 |
| 7 | 9 | 1 | 13:48 |

Таблица 36

Направление движения

|  |  |
| --- | --- |
| Код направления | Направление |
| 1 | Прямое |
| 2 | Обратное |

## Логическая модель

Логическая модель предметной области – это абстрактное описание понятий предметной области, которое определяет сущности, атрибуты, связи между сущностями и ограничения. Она не зависит от конкретной СУБД и служит основой для проектирования физической модели данных [7].

Логическую модель предметной области можно представить в виде ER-диаграммы. ER-диаграмма (Entity-Relationship Diagram, ERD) или диаграмма «сущность-связь» – это графическое представление логической модели, которое показывает сущности, их атрибуты и связи между ними [28]. Основными компонентами ER-диаграммы являются:

* сущность (entity) – объект, о котором хранится информация, обозначается прямоугольником;
* атрибут (attribute) – характеристика сущности, обозначается овалом;
* связь (relationship) – отношение между сущностями, обозначается линией с указанием типа связи (1:1, 1:M, M:M).

В ходе логического моделирования на основе третьей нормальной формы были выделены сущности и их атрибуты (табл. 37-58).

Таблица 37

Сущность «Маршруты»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код маршрута | Первичный ключ |
| Номер маршрута | Номер маршрута |
| Код типа транспорта | Тип транспорта (FK) |
| Код типа маршрута | Тип маршрута (FK) |
| Код начальной остановки | Начальная остановка (FK) |
| Код конечной остановки | Конечная остановка (FK) |
| Код перевозчика | Наименование перевозчика (FK) |
| Работает | Работает ли маршрут |
| Код ТС | Характеристики транспортного средства (FK) |
| Код тарифа | Тариф (FK) |
| Код способа оплаты | Способы оплаты (FK) |
| Код места оплаты | Места оплаты (FK) |

Таблица 38

Сущность «Тип транспорта»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код типа транспорта | Первичный ключ |
| Тип транспорта | Тип транспорта |

Таблица 39

Сущность «Тип маршрута»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код типа маршрута | Первичный ключ |
| Тип маршрута | Тип транспорта |

Таблица 40

Сущность «Остановки»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код остановки | Первичный ключ |
| Остановка | Наименование остановки |
| Код улицы | Улица, на которой расположена остановка (FK) |

Таблица 41

Сущность «Улицы»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код улицы | Первичный ключ |
| Улица | Улица, на которой расположена остановка |
| Код населенного пункта | Населенный пункт, в котором расположена улица (FK) |

Таблица 42

Сущность «Населенные пункты»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код населенного пункта | Первичный ключ |
| Населенный пункт | Населенный пункт, в котором расположена улица |

Таблица 43

Сущность «Перевозчики»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код перевозчика | Первичный ключ |
| Перевозчик | Наименование перевозчика |

Таблица 44

Сущность «Транспортные средства»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код ТС | Первичный ключ |
| Код класса | Класс вместимости транспорта (FK) |
| Код типа пола | Тип пола (FK) |
| Код типа топлива | Тип топлива (FK) |
| Кондиционер | Наличие кондиционера |
| Пандус | Наличие пандуса |
| Wi-Fi | Наличие Wi-Fi |
| USB-зарядки | Наличие USB-зарядок |
| Автоинформатор | Наличие автоинформатора |
| Информационные экраны | Наличие информационных экранов |

Таблица 45

Сущность «Классы транспорта»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код класса | Первичный ключ |
| Класс транспорта | Класс вместимости транспорта |

Таблица 46

Сущность «Типы пола»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код типа пола | Первичный ключ |
| Тип пола | Тип пола |

Таблица 47

Сущность «Типы топлива»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код типа топлива | Первичный ключ |
| Тип топлива | Тип топлива |

Таблица 48

Сущность «Тарифы»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код тарифа | Первичный ключ |
| Наличный | Тариф при наличной оплате |
| Безналичный | Тариф при безналичной оплате |
| Льготный | Тариф для льготников |
| Проездные | Принимаются ли проездные |
| Бесплатная пересадка | Есть ли возможность бесплатной пересадки |

Таблица 49

Сущность «Способы оплаты»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код способа оплаты | Первичный ключ |
| Наличные | Доступна ли оплата наличными |
| Банковская карта | Доступна ли оплата банковской картой |
| Транспортная карта | Доступна ли оплата транспортной картой |
| QR-код | Доступна ли оплата по QR-коду |

Таблица 50

Сущность «Места оплаты»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код способа оплаты | Первичный ключ |
| Кондуктор | Есть ли кондуктор |
| Валидатор | Есть ли валидаторы |
| Водитель | Возможно ли купить билет у водителя |
| Автомат | Есть ли автомат для оплаты |
| Автовокзал | Возможно ли купить билет на автовокзале |
| Онлайн | Возможно ли купить билет онлайн |

Таблица 51

Сущность «Время работы»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код времени работы | Первичный ключ |
| Код маршрута | Номер маршрута (FK) |
| Код графика по месяцам | В какие месяца работает маршрут (FK) |
| Код графика по дням недели | В какие дни недели работает маршрут (FK) |
| Код типа расписания | Тип расписания (FK) |
| Время начала работы | Время первого рейса |
| Время окончания работы | Время последнего рейса |

Таблица 52

Сущность «График по месяцам»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код графика по месяцам | Первичный ключ |
| Январь | Работает ли маршрут в январе |
| Февраль | Работает ли маршрут в феврале |
| Март | Работает ли маршрут в марте |
| Апрель | Работает ли маршрут в апреле |
| Май | Работает ли маршрут в мае |
| Июнь | Работает ли маршрут в июне |
| Июль | Работает ли маршрут в июле |
| Август | Работает ли маршрут в августе |
| Сентябрь | Работает ли маршрут в сентябре |
| Октябрь | Работает ли маршрут в октябре |
| Ноябрь | Работает ли маршрут в ноябре |
| Декабрь | Работает ли маршрут в декабре |

Таблица 53

Сущность «График по дням недели»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код графика по дням недели | Первичный ключ |
| Понедельник | Работает ли маршрут по понедельникам |
| Вторник | Работает ли маршрут по вторникам |
| Среда | Работает ли маршрут по средам |
| Четверг | Работает ли маршрут по четвергам |
| Пятница | Работает ли маршрут по пятницам |
| Суббота | Работает ли маршрут по субботам |
| Воскресенье | Работает ли маршрут по воскресеньям |

Таблица 54

Сущность «Типы расписания»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код типа расписания | Первичный ключ |
| Тип расписания | Тип расписания |

Таблица 55

Сущность «Интервальное расписание»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код расписания | Первичный ключ |
| Код времени работы | Время работы маршрута (FK) |
| Код графика по времени суток | В какое время суток работает маршрут (FK) |
| Интервал | Время между рейсами |

Таблица 56

Сущность «График по времени суток»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код графика по времени суток | Первичный ключ |
| Начало интервала | Начало времени суток |
| Конец интервала | Конец времени суток |

Таблица 57

Сущность «Точное расписание»

|  |  |
| --- | --- |
| Атрибут | Описание |
| Код расписания | Первичный ключ |
| Код времени работы | Время работы маршрута (FK) |
| Код направления | В каком направлении следует (FK) |
| Время отправления | Время отправления |

Таблица 58

Сущность «Направление движения»

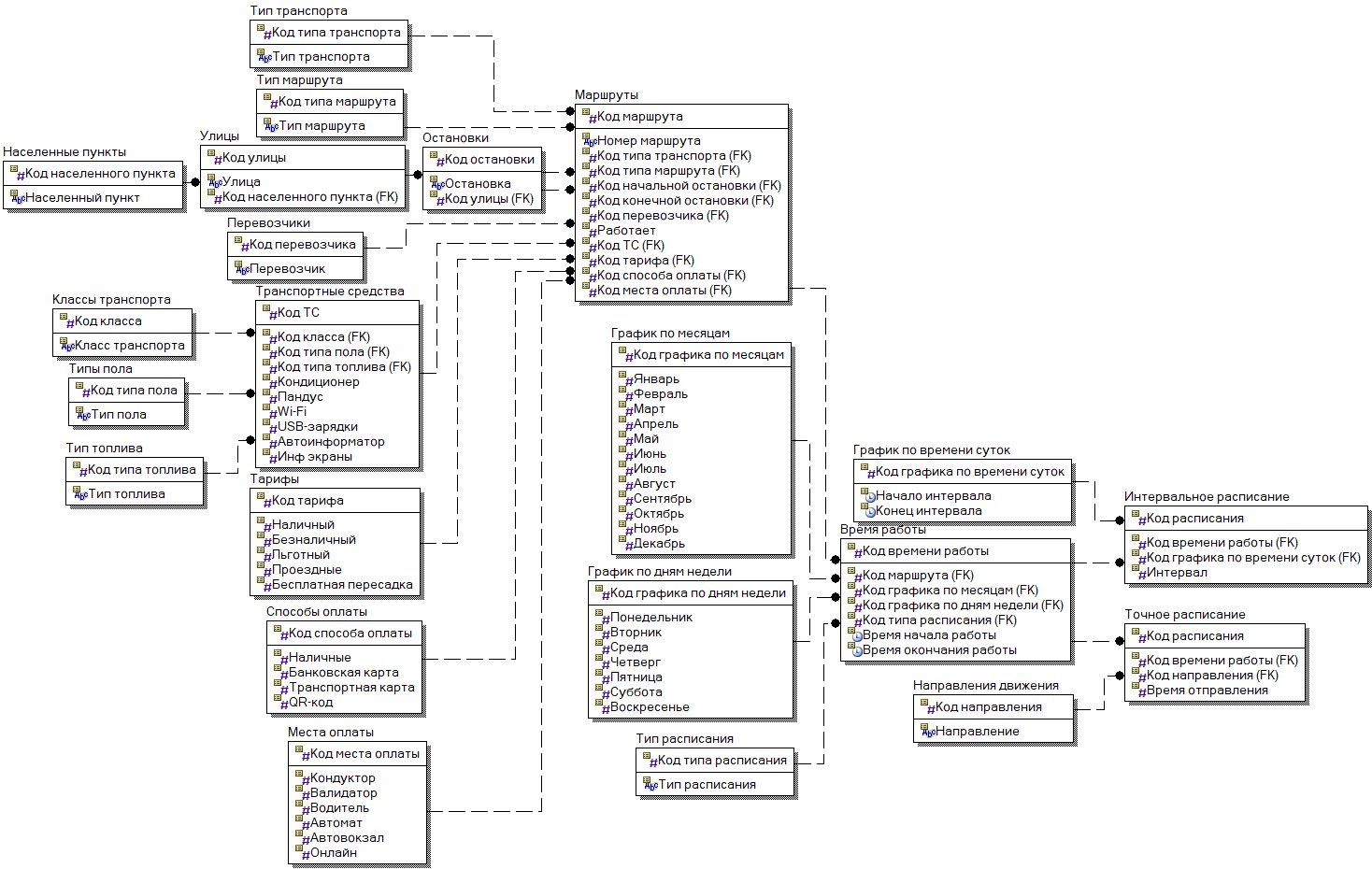
|  |  |
| --- | --- |
| Атрибут | Описание |
| Код направления | Первичный ключ |
| Направление | В каком направлении следует |

Для построения ER-диаграммы логической модели предметной области была использована программа ERwin.

ERwin – CASE-средство для концептуального, логического и физического моделирования данных, предназначенное для проектирования, анализа, документирования и управления структурами баз данных [29].

ER-диаграмма логической модели предметной области представлена на рис. 5.

После построения логической модели предметной области можно переходить к построению физической модели предметной области.



39

Рис. 5. ER-диаграмма логической модели предметной области

## Физическая модель

Физическая модель предметной области – это реализация логической модели предметной области с учётом особенностей конкретной СУБД. Она определяет физическое представление данных на диске, включая структуру таблиц, типы данных полей, связи между таблицами, индексы, ограничения целостности и другие параметры хранения [22].

Перед построением физической модели предметной области необходимо выбрать СУБД, под которую эта модель будет строится. СУБД – это программное обеспечение для создания, управления и взаимодействия с базами данных. Рассмотрим наиболее популярные СУБД.

Microsoft Access – реляционная СУБД с интуитивным интерфейсом, разработанная компанией Microsoft. Она входит в пакет Microsoft Office и предназначена для создания небольших локальных баз данных. Access отличается простым графическим интерфейсом, который позволяет пользователям без глубоких знаний программирования создавать формы, отчеты и запросы. Основные преимущества Access включают удобную интеграцию с другими продуктами Microsoft, такими как Excel и Word. Однако у Access есть серьезные ограничения: он обладает низкой производительностью при больших объемах данных [2].

MySQL – это одна из самых популярных реляционных СУБД с открытым исходным кодом, разработанная Oracle Corporation. Она широко используется в web-разработке, особенно в связке с PHP. MySQL отличается высокой скоростью работы, надежностью и простотой настройки, что делает ее отличным выбором для небольших и средних проектов. Однако у MySQL есть и недостатки, такие как ограниченная поддержка сложных SQL-функций и менее гибкая система хранения процедур и триггеров [31].

SQLite – это встраиваемая реляционная СУБД, которая хранит всю базу данных в одном файле. Она не требует сервера и работает напрямую в приложении, что делает ее идеальным выбором для мобильных приложений, десктопных программ и встроенных систем. SQLite отличается минимализмом, высокой скоростью и отказоустойчивостью. Однако SQLite не подходит для высоконагруженных многопользовательских приложений, так как не поддерживает сетевой доступ и имеет ограничения на одновременные операции записи [35].

PostgreSQL – это мощная реляционная СУБД с открытым исходным кодом, известная своей расширяемостью и поддержкой сложных запросов. PostgreSQL поддерживает множество типов данных, включая JSON, XML и геопространственные данные, а также предоставляет развитые механизмы индексирования и оптимизации. Эта СУБД подходит для крупных проектов, где важны надежность и масштабируемость. Среди недостатков PostgreSQL можно отметить более высокие требования к ресурсам и сложность начальной настройки для новичков [33].

Oracle – это коммерческая реляционная СУБД, разработанная корпорацией Oracle. Она считается одной из самых мощных и надежных систем для корпоративного использования, поддерживая огромные объемы данных, сложные транзакции и высокую доступность. Однако эта СУБД требует значительных финансовых затрат на лицензирование и обслуживание, а также обладает высокой сложностью администрирования, что делает ее менее доступной для небольших компаний и стартапов [32].

Для разработки физической модели была выбрана СУБД Microsoft Access, так как она соответствует следующим критериям: простота разработки, удобство интерфейса, низкие требования к установке и эксплуатации, а также достаточная функциональность для реализации проекта малого масштаба.

В ходе создания физической модели для каждого поля был определен свой тип данных (табл. 59-80).

Таблица 59

Таблица «Маршруты»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код маршрута | Counter |
| Номер маршрута | Text(20) |
| Код типа транспорта | Long Integer |
| Код типа маршрута | Long Integer |
| Код начальной остановки | Long Integer |
| Код конечной остановки | Long Integer |
| Код перевозчика | Long Integer |
| Работает | Yes/No |
| Код ТС | Long Integer |
| Код тарифа | Long Integer |
| Код способа оплаты | Long Integer |
| Код места оплаты | Long Integer |

Таблица 60

Сущность «Тип транспорта»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код типа транспорта | Counter |
| Тип транспорта | Text(20) |

Таблица 61

Сущность «Тип маршрута»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код типа маршрута | Counter |
| Тип маршрута | Text(20) |

Таблица 62

Сущность «Остановки»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код остановки | Counter |
| Остановка | Text(20) |
| Код улицы | Long Integer |

Таблица 63

Сущность «Улицы»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код улицы | Counter |
| Улица | Text(20) |
| Код населенного пункта | Long Integer |

Таблица 64

Сущность «Населенные пункты»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код населенного пункта | Counter |
| Населенный пункт | Text(20) |

Таблица 65

Сущность «Перевозчики»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код перевозчика | Counter |
| Перевозчик | Text(20) |

Таблица 66

Сущность «Транспортные средства»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код ТС | Counter |
| Код класса | Long Integer |
| Код типа пола | Long Integer |
| Код типа топлива | Long Integer |
| Кондиционер | Yes/No |
| Пандус | Yes/No |
| Wi-Fi | Yes/No |
| USB-зарядки | Yes/No |
| Автоинформатор | Yes/No |
| Информационные экраны | Yes/No |

Таблица 67

Сущность «Классы транспорта»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код класса | Counter |
| Класс транспорта | Text(20) |

Таблица 68

Сущность «Типы пола»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код типа пола | Counter |
| Тип пола | Text(20) |

Таблица 69

Сущность «Типы топлива»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код типа топлива | Counter |
| Тип топлива | Text(20) |

Таблица 70

Сущность «Тарифы»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код тарифа | Counter |
| Наличный | Long Integer |
| Безналичный | Long Integer |
| Льготный | Long Integer |
| Проездные | Yes/No |
| Бесплатная пересадка | Yes/No |

Таблица 71

Сущность «Способы оплаты»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код способа оплаты | Counter |
| Наличные | Yes/No |
| Банковская карта | Yes/No |
| Транспортная карта | Yes/No |
| QR-код | Yes/No |

Таблица 72

Сущность «Места оплаты»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код способа оплаты | Counter |
| Кондуктор | Yes/No |
| Валидатор | Yes/No |
| Водитель | Yes/No |
| Автомат | Yes/No |
| Автовокзал | Yes/No |
| Онлайн | Yes/No |

Таблица 73

Сущность «Время работы»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код времени работы | Counter |
| Код маршрута | Long Integer |
| Код графика по месяцам | Long Integer |
| Код графика по дням недели | Long Integer |
| Код типа расписания | Long Integer |
| Время начала работы | Date/Time |
| Время окончания работы | Date/Time |

Таблица 74

Сущность «График по месяцам»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код графика по месяцам | Counter |
| Январь | Yes/No |
| Февраль | Yes/No |
| Март | Yes/No |
| Апрель | Yes/No |
| Май | Yes/No |
| Июнь | Yes/No |
| Июль | Yes/No |
| Август | Yes/No |
| Сентябрь | Yes/No |
| Октябрь | Yes/No |
| Ноябрь | Yes/No |
| Декабрь | Yes/No |

Таблица 75

Сущность «График по дням недели»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код графика по дням недели | Counter |
| Понедельник | Yes/No |
| Вторник | Yes/No |
| Среда | Yes/No |
| Четверг | Yes/No |
| Пятница | Yes/No |
| Суббота | Yes/No |
| Воскресенье | Yes/No |

Таблица 76

Сущность «Типы расписания»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код типа расписания | Counter |
| Тип расписания | Text(20) |

Таблица 77

Сущность «Интервальное расписание»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код расписания | Counter |
| Код времени работы | Long Integer |
| Код графика по времени суток | Long Integer |
| Интервал | Long Integer |

Таблица 78

Сущность «График по времени суток»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код графика по времени суток | Counter |
| Начало интервала | Date/Time |
| Конец интервала | Date/Time |

Таблица 79

Сущность «Точное расписание»

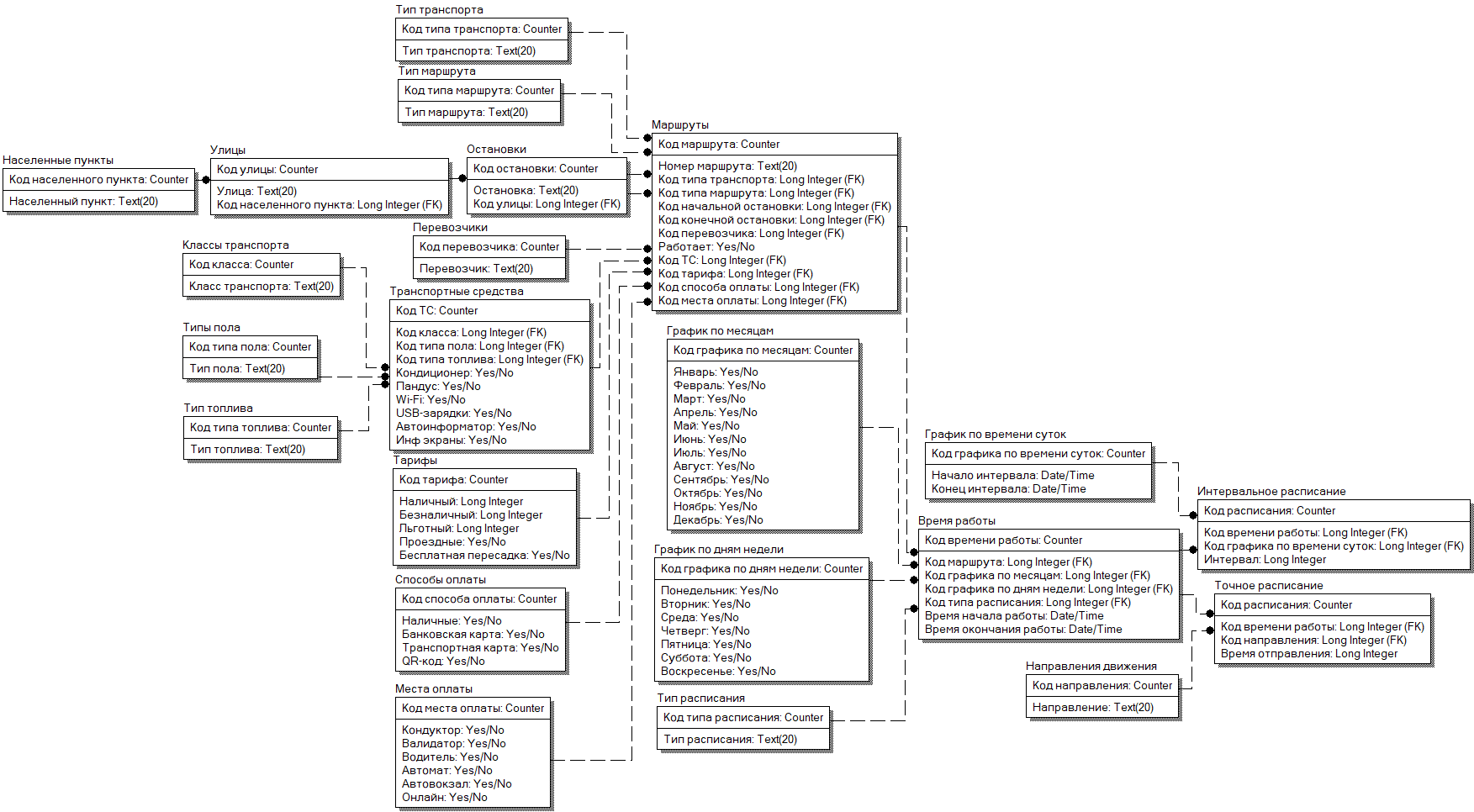
|  |  |
| --- | --- |
| Поле | Тип данных |
| Код расписания | Couner |
| Код времени работы | Long Integer |
| Код направления | Long Integer |
| Время отправления | Date/Time |

Таблица 80

Сущность «Направление движения»

|  |  |
| --- | --- |
| Поле | Тип данных |
| Код направления | Counter |
| Направление | Text(20) |

С помощью программы ERwin на основе логической модели предметной области была создана физическая модель предметной области (рис. 6).



46

Рис. 6. Физическая модель предметной области

Затем с помощью функции «Forward Engineer» («Прямое проектирование») в ERwin была создана структура базы данных в формате СУБД Access. Фрагмент схемы данных приведен на рис. 7. Структуры некоторых таблиц в Конструкторе приведены на рис. 8-9.

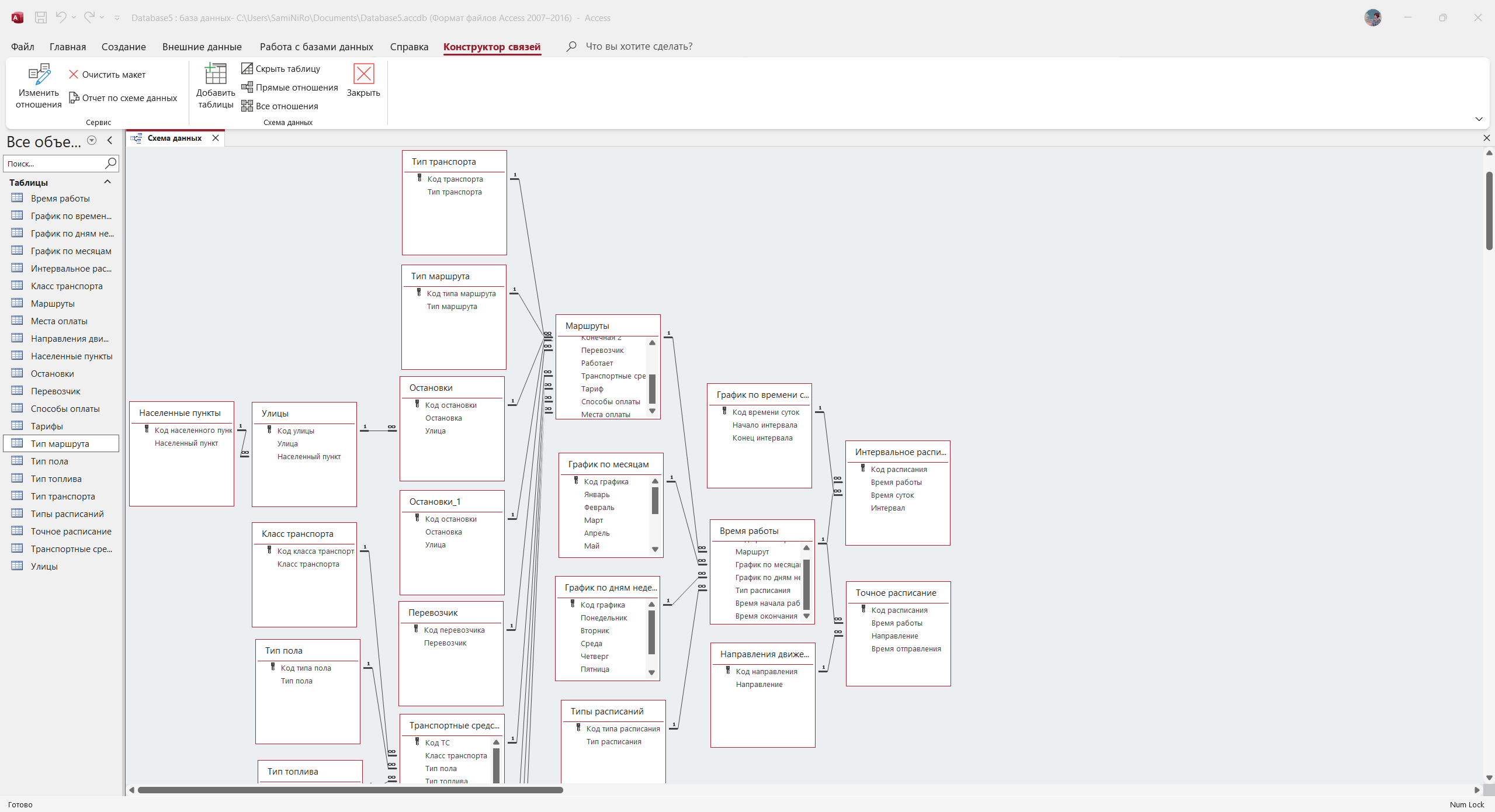


Рис. 7. Фрагмент схемы данных в Access

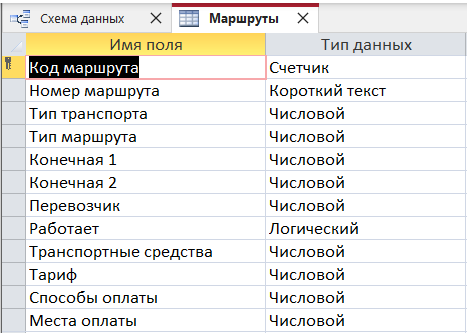


Рис. 8. Структура таблицы «Маршруты»

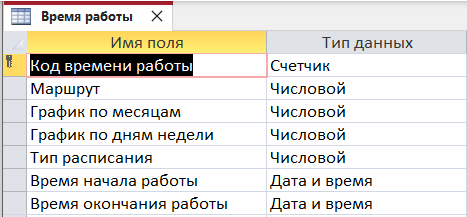


Рис. 9. Структура таблицы «Время работы»

Теперь база данных готова к использованию и ее можно наполнять информацией. Пример некоторых таблиц со внесенными записями представлен на рис. 10-11.

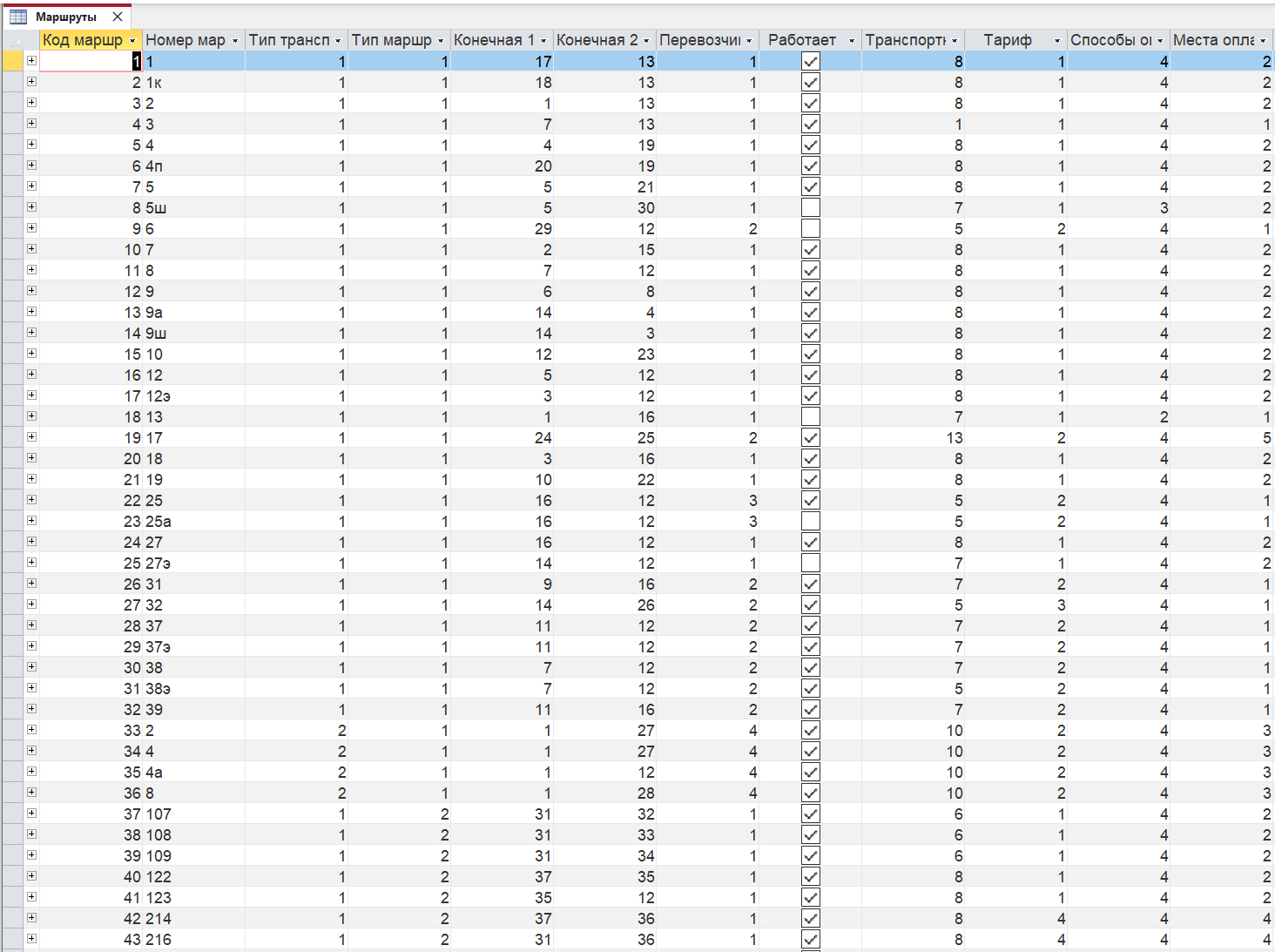


Рис. 10. Таблица «Маршруты»

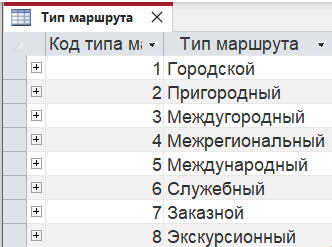


Рис. 11. Таблица «Тип маршрута»

# Описание программы

Управлять созданной базой данных можно с помощью СУБД Microsoft Access. Однако она может быть неудобна для неподготовленных пользователей, так как имеет излишний функционал, который может вызвать непонимание, трудности при работе и увеличение риска ошибок, которые могут повлечь за собой потерю данных и повреждение БД. Кроме того, представление данных в третьей нормальной форме хоть и обеспечивает оптимальную структуру, но не всегда удобно для визуального восприятия и анализа информации конечными пользователями.

В связи с этим возникла необходимость написания отдельной программы, которая будет иметь более удобный интерфейс с минимальным функционалом, необходимым для работы с данной БД без необходимости прямого доступа к СУБД.

Для написания программы необходимо выбрать один из высокоуровневых языков программирования. Рассмотрим наиболее популярные из них.

C – это процедурный язык программирования общего назначения, разработанный в 1972 году для создания операционной системы UNIX. Сочетает высокоуровневый синтаксис с низкоуровневыми возможностями (указатели, работа с памятью). Основное применение: системное программное обеспечение (ПО), драйверы, встраиваемые системы. Для работы с базами данных используются библиотеки типа SQLite API или MySQL Connector/C. При работе с Microsoft Access применяется низкоуровневый доступ через интерфейс ODBC API или набор компонентов Microsoft Data Access Components (MDAC), что требует ручного управления соединениями. Главные преимущества языка – эффективность, переносимость и полный контроль над аппаратурой. К недостаткам относятся сложность ручного управления памятью, отсутствие встроенной системы безопасности и современных парадигм программирования, таких как объектно-ориентированное программирование (ООП) [26].

C++ – это расширение C с добавлением ООП и других возможностей. Он сохранил производительность C, но приобрёл классы, шаблоны, встроенную библиотеку шаблонов STL и современные функциональные возможности (например, умные указатели). Используется в играх, высоконагруженных сервисах и графических приложениях. Для работы с БД используются драйверы ODBC, библиотеки типа SOCI или ORM. Для интеграции с Microsoft Access часто применяются технологии OLE DB и ActiveX Data Objects (ADO), позволяющие напрямую работать с COM-объектами Access. Однако он сложен для изучения из-за обилия возможностей и тонкостей [19].

C# – это язык высокого уровня с удобным синтаксисом и большой стандартной библиотекой. Разработан компанией Microsoft и работает на платформе .NET. На нём пишут десктопные приложения, игры и backend. Широко используется для работы с базами данных через библиотеки ADO.NET и Dapper, а также через фреймворк Entity Framework. Наиболее удобен для работы с Microsoft Access благодаря встроенной поддержке через библиотеку System.Data.OleDb и возможности использования Entity Framework. C# проще C++, но требует .NET, что ограничивает его кроссплатформенность [27].

Python – интерпретируемый язык с динамической типизацией и простым синтаксисом. Он идеально подходит для начинающих программистов. Используется для написания скриптов, web-разработки, анализа данных и работы с искусственным интеллектом (ИИ). Поддерживает все популярные СУБД через драйверы (psycopg2 для PostgreSQL, PyMySQL для MySQL) и технологию ORM (SQLAlchemy, Django ORM). Для работы с Microsoft Access использует драйверы pyodbc или pypyodbc, требует настройки источника данных DSN. Также возможен доступ через COM-объекты с помощью библиотеки win32com. Минусы – низкая скорость выполнения и высокое потребление памяти [34].

Java – объектно-ориентированный язык, основной особенностью которого является компиляция в промежуточный байт-код, который затем выполняется виртуальной машиной JVM. Благодаря кроссплатформенности и автоматическому управлению памятью он широко используется в корпоративной разработке, серверных и мобильных приложениях. Основные технологии работы с базами: JDBC, Hibernate, JPA. Для подключения к Microsoft Access использует драйвер UCanAccess. Основные преимущества – надежность, большие библиотеки и высокая производительность. Главный недостаток – необходимость писать больше кода по сравнению с современными языками [30].

Для создания приложения был выбран язык C#. Выбор данного языка для работы с базой данных Microsoft Access обусловлен его полной совместимостью с технологиями Microsoft, что обеспечивает простую и надежную интеграцию. Встроенные средства .NET предоставляют готовые решения для подключения и работы с Access-базами без необходимости установки дополнительного ПО. C# сочетает высокую производительность с удобством разработки, предлагая как прямое выполнение SQL-запросов через библиотеку ADO.NET, так и возможность использования более высокоуровневых подходов.

Разработка программы велась в среде Visual Studio (рис. 12). Visual Studio – это интегрированная среда разработки от Microsoft, в которой можно писать программы на разных языках (C#, C++, Python, JavaScript и др.). Она подходит для создания приложений под Windows, web-сайтов, мобильных приложений и игр. В ней имеется встроенный отладчик, инструменты для работы с базами данных, поддержка Git и облачных сервисов Azure [36].

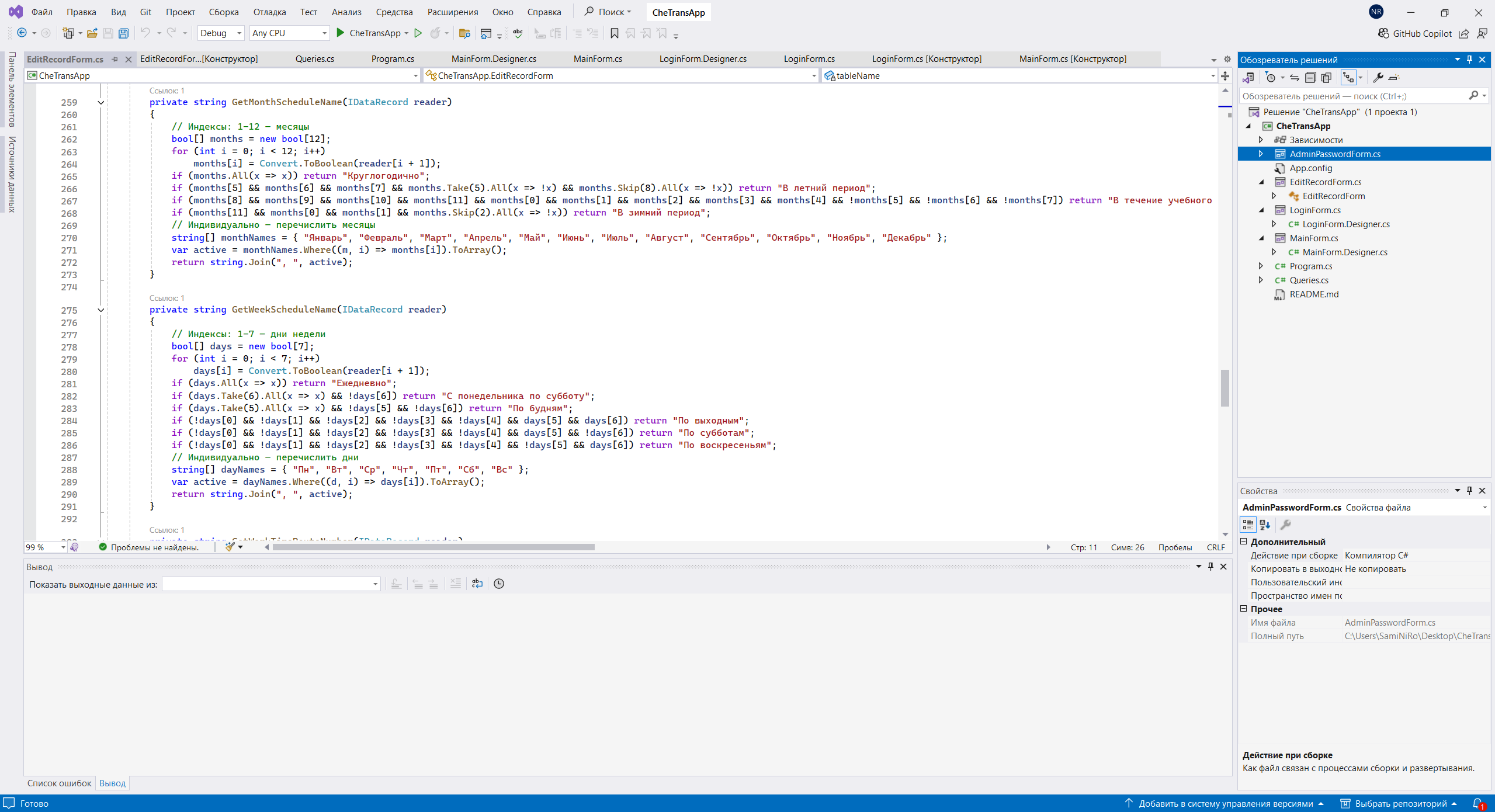


Рис. 12. Среда разработки Visual Studio

Файл программы носит название «CheTransApp.exe».

Для работы с базой данных используется технология OLE DB (Object Linking and Embedding Database) – это интерфейс доступа к данным от компании Microsoft [16], позволяющий работать с разными типами источников данных: базами данных (Access, SQL Server, Oracle), Excel, текстовыми файлами и т. д.

Для подключения к базе данных используется следующий код:

private string connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=CheTrans.accdb;";

OleDbConnection conn = new OleDbConnection(connectionString);

Этот код создаёт подключение к базе данных Access (.accdb) с помощью технологии OLE DB.

Для разработки интерфейса программы использовалась технология Windows Forms (рис. 13). Windows Forms – это технология создания графических пользовательских интерфейсов (GUI) для приложений под Windows на языке C# [13]. Это часть .NET Framework (а также она поддерживается в .NET Core / .NET 5+), предоставляющая разработчику инструменты для построения оконных приложений с элементами управления – кнопками, текстовыми полями, таблицами и т. д.

В программе были использованы следующие элементы визуального интерфейса:

* DataGridView – таблица для вывода и редактирования данных;
* Button – кнопка;
* TextBox – текстовое поле ввода;
* ComboBox – выпадающий список;
* Label – текст;
* CheckBox – флаг выбора.

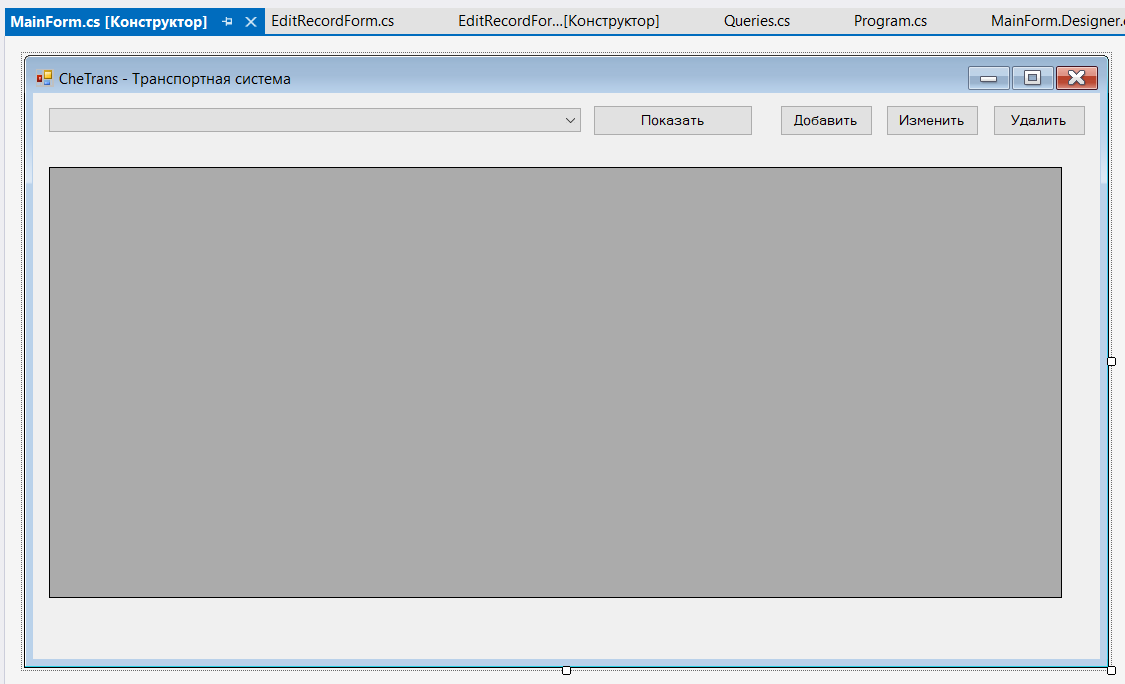


Рис. 13. Макет окна программы, разработанный через Windows Forms

В программе было реализовано 2 пользователя – «Пассажир» (обычный пользователь) и «Администратор». Если выбран «Администратор» – программа запросит ввести пароль. Выбор пользователя и ввод пароля не зависят от базы данных. Они реализованы программно с помощью следующего кода:

private void btnPassenger\_Click(object sender, EventArgs e)

{

UserRole = "Пассажир";

DialogResult = DialogResult.OK;

}

private void btnAdmin\_Click(object sender, EventArgs e)

{

using (var passwordForm = new AdminPasswordForm())

{

if (passwordForm.ShowDialog() == DialogResult.OK)

{

UserRole = "Администратор";

DialogResult = DialogResult.OK;

}

}

}

if (txtPassword.Text == "1234")

{

IsPasswordCorrect = true;

DialogResult = DialogResult.OK;

}

else

{

MessageBox.Show("Неверный пароль!");

txtPassword.Clear();

}

Функционал программы для «Пассажира» представляет собой 30 запросов к базе данных, которые выбираются через выпадающий список.

Запросы к базе данных производятся с помощью языка SQL. SQL (Structured Query Language) – это язык структурированных запросов к базе данных. Он используется для хранения, извлечения, изменения и удаления данных в реляционных БД (таких как Access, MySQL, PostgreSQL, SQL Server и др.) [6].

В программе были реализованы следующие запросы:

* список всех маршрутов с начальными и конечными остановками:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип маршрута].[Тип маршрута],

Начальная.Остановка AS Начальная,

Конечная.Остановка AS Конечная

FROM

Маршруты

INNER JOIN Остановки AS Начальная

ON Маршруты.Начальная = Начальная.[Код остановки]

INNER JOIN Остановки AS Конечная

ON Маршруты.Конечная = Конечная.[Код остановки]

INNER JOIN [Тип маршрута]

ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = TRUE;

* список всех автобусов с начальными и конечными остановками:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип маршрута].[Тип маршрута],

Начальная.Остановка AS Начальная,

Конечная.Остановка AS Конечная

FROM

Маршруты

INNER JOIN Остановки AS Начальная

ON Маршруты.Начальная = Начальная.[Код остановки]

INNER JOIN Остановки AS Конечная

ON Маршруты.Конечная = Конечная.[Код остановки]

INNER JOIN [Тип маршрута]

ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = TRUE

AND [Тип транспорта].[Тип транспорта] = 'Автобус';

* список всех трамваев с начальными и конечными остановками:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип маршрута].[Тип маршрута],

Начальная.Остановка AS Начальная,

Конечная.Остановка AS Конечная

FROM

Маршруты

INNER JOIN Остановки AS Начальная

ON Маршруты.Начальная = Начальная.[Код остановки]

INNER JOIN Остановки AS Конечная

ON Маршруты.Конечная = Конечная.[Код остановки]

INNER JOIN [Тип маршрута]

ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = TRUE

AND [Тип транспорта].[Тип транспорта] = 'Трамвай';

* список всех городских маршрутов:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип маршрута].[Тип маршрута],

Начальная.Остановка AS Начальная,

Конечная.Остановка AS Конечная

FROM

Маршруты

INNER JOIN Остановки AS Начальная

ON Маршруты.Начальная = Начальная.[Код остановки]

INNER JOIN Остановки AS Конечная

ON Маршруты.Конечная = Конечная.[Код остановки]

INNER JOIN [Тип маршрута]

ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = TRUE

AND [Тип маршрута].[Тип маршрута] = 'Городской';

* список всех пригородных маршрутов:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип маршрута].[Тип маршрута],

Начальная.Остановка AS Начальная,

Конечная.Остановка AS Конечная

FROM

Маршруты

INNER JOIN Остановки AS Начальная

ON Маршруты.Начальная = Начальная.[Код остановки]

INNER JOIN Остановки AS Конечная

ON Маршруты.Конечная = Конечная.[Код остановки]

INNER JOIN [Тип маршрута]

ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = TRUE

AND [Тип маршрута].[Тип маршрута] = 'Пригородный';

* список всех маршрутов с тарифами:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Тарифы.Наличный,

Тарифы.Безналичный,

Тарифы.Льготный,

Тарифы.Проездные,

Тарифы.[Бесплатная пересадка]

FROM

Маршруты

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

INNER JOIN Тарифы

ON Маршруты.[Тариф] = Тарифы.[Код тарифа]

WHERE

Маршруты.Работает = TRUE;

* список всех маршрутов со способами оплаты:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Способы оплаты].Наличные,

[Способы оплаты].[Банковская карта],

[Способы оплаты].[Транспортная карта],

[Способы оплаты].[QR-код]

FROM

Маршруты

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

INNER JOIN [Способы оплаты]

ON Маршруты.[Способы оплаты] = [Способы оплаты].[Код способа оплаты]

WHERE

Маршруты.Работает = TRUE;

* список всех маршрутов с местами оплаты:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Места оплаты].Кондуктор,

[Места оплаты].Валидатор,

[Места оплаты].Водитель,

[Места оплаты].Автомат,

[Места оплаты].Автовокзал,

[Места оплаты].Онлайн

FROM

Маршруты

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

INNER JOIN [Места оплаты]

ON Маршруты.[Места оплаты] = [Места оплаты].[Код места оплаты]

WHERE

Маршруты.Работает = TRUE;

* список всех перевозчиков:

SELECT

Перевозчик

FROM

Перевозчики;

* список всех остановок с улицами и населенными пунктами:

SELECT

Остановки.Остановка,

Улицы.Улица,

[Населенные пункты].[Населенный пункт]

FROM

(Остановки

INNER JOIN Улицы

ON Остановки.Улица = Улицы.[Код улицы])

INNER JOIN [Населенные пункты]

ON Улицы.[Населенный пункт] = [Населенные пункты].[Код населенного пункта];

* список всех маршрутов с характеристиками транспортных средств:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Класс транспорта].[Класс транспорта],

[Тип пола].[Тип пола],

[Тип топлива].[Тип топлива],

[Транспортные средства].Кондиционер,

[Транспортные средства].Пандус,

[Транспортные средства].[Wi-Fi],

[Транспортные средства].[USB-зарядки],

[Транспортные средства].Автоинформатор,

[Транспортные средства].[Информационные экраны]

FROM

Маршруты

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

INNER JOIN [Транспортные средства]

ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]

INNER JOIN [Класс транспорта]

ON [Транспортные средства].[Класс транспорта] = [Класс транспорта].[Код класса транспорта]

INNER JOIN [Тип пола]

ON [Транспортные средства].[Тип пола] = [Тип пола].[Код типа пола]

INNER JOIN [Тип топлива]

ON [Транспортные средства].[Тип топлива] = [Тип топлива].[Код типа топлива]

WHERE

Маршруты.Работает = TRUE

ORDER BY

Маршруты.[Код маршрута];

* список всех маршрутов по перевозчикам:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Перевозчики.Перевозчик

FROM

Маршруты

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

INNER JOIN Перевозчики

ON Маршруты.Перевозчик = Перевозчики.[Код перевозчика]

WHERE

Маршруты.Работает = TRUE;

* список всех маршрутов по времени работы:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Switch(

[Время работы].[График по месяцам] = 1, "Ежегодно",

[Время работы].[График по месяцам] = 2, "В летний период",

[Время работы].[График по месяцам] = 3, "В течение учебного года",

[Время работы].[График по месяцам] = 4, "В зимний период"

) AS [График по месяцам],

Switch(

[Время работы].[График по дням недели] = 1, "Ежедневно",

[Время работы].[График по дням недели] = 2, "С понедельника по субботу",

[Время работы].[График по дням недели] = 3, "По будням",

[Время работы].[График по дням недели] = 4, "По выходным",

[Время работы].[График по дням недели] = 5, "По субботам",

[Время работы].[График по дням недели] = 6, "По воскресеньям"

) AS [График по дням недели],

Format([Время работы].[Время начала работы], "hh:nn") AS [Время начала],

Format([Время работы].[Время окончания работы], "hh:nn") AS [Время окончания]

FROM

([Время работы]

INNER JOIN Маршруты

ON [Время работы].Маршрут = Маршруты.[Код маршрута])

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = TRUE;

* интервальное расписание маршрутов:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Switch(

[Время работы].[График по дням недели] = 1, "Ежедневно",

[Время работы].[График по дням недели] = 2, "С понедельника по субботу",

[Время работы].[График по дням недели] = 3, "По будням",

[Время работы].[График по дням недели] = 4, "По выходным",

[Время работы].[График по дням недели] = 5, "По субботам",

[Время работы].[График по дням недели] = 6, "По воскресеньям"

) AS [График по дням недели],

IIf(

SUM(IIf([Интервальное расписание].[Время суток] = 1 OR [Интервальное расписание].[Время суток] = 5, [Интервальное расписание].[Интервал], 0)) = 0,

Null,

SUM(IIf([Интервальное расписание].[Время суток] = 1 OR [Интервальное расписание].[Время суток] = 5, [Интервальное расписание].[Интервал], 0))

) AS [05:00-10:00],

IIf(

SUM(IIf([Интервальное расписание].[Время суток] = 2 OR [Интервальное расписание].[Время суток] = 5, [Интервальное расписание].[Интервал], 0)) = 0,

Null,

SUM(IIf([Интервальное расписание].[Время суток] = 2 OR [Интервальное расписание].[Время суток] = 5, [Интервальное расписание].[Интервал], 0))

) AS [10:00-15:00],

IIf(

SUM(IIf([Интервальное расписание].[Время суток] = 3 OR [Интервальное расписание].[Время суток] = 5, [Интервальное расписание].[Интервал], 0)) = 0,

Null,

SUM(IIf([Интервальное расписание].[Время суток] = 3 OR [Интервальное расписание].[Время суток] = 5, [Интервальное расписание].[Интервал], 0))

) AS [15:00-19:00],

IIf(

SUM(IIf([Интервальное расписание].[Время суток] = 4 OR [Интервальное расписание].[Время суток] = 5, [Интервальное расписание].[Интервал], 0)) = 0,

Null,

SUM(IIf([Интервальное расписание].[Время суток] = 4 OR [Интервальное расписание].[Время суток] = 5, [Интервальное расписание].[Интервал], 0))

) AS [19:00-00:00]

FROM

((Маршруты

INNER JOIN [Тип транспорта]

ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Время работы]

ON Маршруты.[Код маршрута] = [Время работы].[Маршрут])

INNER JOIN [Интервальное расписание]

ON [Время работы].[Код времени работы] = [Интервальное расписание].[Время работы]

WHERE

[Время работы].[Тип расписания] = (

SELECT [Код типа расписания]

FROM [Типы расписаний]

WHERE [Тип расписания] = 'Интервальное'

)

AND Маршруты.Работает = TRUE

GROUP BY

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Время работы].[График по дням недели];

* список всех маршрутов, работающих летом:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Switch(

[Время работы].[График по месяцам] = 1, "Ежегодно",

[Время работы].[График по месяцам] = 2, "В летний период",

[Время работы].[График по месяцам] = 3, "В течение учебного года",

[Время работы].[График по месяцам] = 4, "В зимний период"

) AS [График по месяцам],

Switch(

[Время работы].[График по дням недели] = 1, "Ежедневно",

[Время работы].[График по дням недели] = 2, "С понедельника по субботу",

[Время работы].[График по дням недели] = 3, "По будням",

[Время работы].[График по дням недели] = 4, "По выходным",

[Время работы].[График по дням недели] = 5, "По субботам",

[Время работы].[График по дням недели] = 6, "По воскресеньям"

) AS [График по дням недели],

Format([Время работы].[Время начала работы], "hh:nn") AS [Время начала],

Format([Время работы].[Время окончания работы], "hh:nn") AS [Время окончания]

FROM

(([Время работы]

INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута])

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [График по месяцам] ON [Время работы].[График по месяцам] = [График по месяцам].[Код графика]

WHERE

Маршруты.Работает = TRUE

AND [График по месяцам].Июнь = TRUE

AND [График по месяцам].Июль = TRUE

AND [График по месяцам].Август = TRUE;

* список всех маршрутов с низкопольным транспортом:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип пола].[Тип пола]

FROM

((Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС])

INNER JOIN [Тип пола] ON [Транспортные средства].[Тип пола] = [Тип пола].[Код типа пола])

WHERE

Маршруты.Работает = TRUE

AND [Тип пола].[Тип пола] = 'Низкопольный'

ORDER BY

Маршруты.[Код маршрута];

* список всех маршрутов с транспортом на газу:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип топлива].[Тип топлива]

FROM

((Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС])

INNER JOIN [Тип топлива] ON [Транспортные средства].[Тип топлива] = [Тип топлива].[Код типа топлива])

WHERE

Маршруты.Работает = TRUE

AND [Тип топлива].[Тип топлива] = 'Газ'

ORDER BY

Маршруты.[Код маршрута];

* список всех маршрутов с транспортом на электричестве:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип топлива].[Тип топлива]

FROM

((Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС])

INNER JOIN [Тип топлива] ON [Транспортные средства].[Тип топлива] = [Тип топлива].[Код типа топлива])

WHERE

Маршруты.Работает = TRUE

AND [Тип топлива].[Тип топлива] = 'Электричество'

ORDER BY

Маршруты.[Код маршрута];

* список всех маршрутов с кондиционерами:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Транспортные средства].Кондиционер

FROM

(Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]

WHERE

Маршруты.Работает = TRUE

AND [Транспортные средства].Кондиционер = TRUE;

* список всех маршрутов с USB-зарядками:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Транспортные средства].[USB-зарядки]

FROM

(Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]

WHERE

Маршруты.Работает = TRUE

AND [Транспортные средства].[USB-зарядки] = TRUE;

* список всех маршрутов с пандусами:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Транспортные средства].Пандус

FROM

(Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]

WHERE

Маршруты.Работает = TRUE

AND [Транспортные средства].Пандус = TRUE

ORDER BY

Маршруты.[Код маршрута];

* список всех маршрутов с Wi-Fi:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Транспортные средства].[Wi-Fi]

FROM

(Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]

WHERE

Маршруты.Работает = TRUE

AND [Транспортные средства].[Wi-Fi] = TRUE;

* список всех маршрутов с валидаторами:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Места оплаты].Валидатор

FROM

(Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Места оплаты] ON Маршруты.[Места оплаты] = [Места оплаты].[Код места оплаты]

WHERE

Маршруты.Работает = TRUE

AND [Места оплаты].Валидатор = TRUE;

* список всех маршрутов с кондукторами:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Места оплаты].Кондуктор

FROM

(Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [Места оплаты] ON Маршруты.[Места оплаты] = [Места оплаты].[Код места оплаты]

WHERE

Маршруты.Работает = TRUE

AND [Места оплаты].Кондуктор = TRUE;

* список всех маршрутов, работающих по воскресеньям:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Switch(

[Время работы].[График по месяцам]=1, "Ежегодно",

[Время работы].[График по месяцам]=2, "В летний период",

[Время работы].[График по месяцам]=3, "В течение учебного года",

[Время работы].[График по месяцам]=4, "В зимний период"

) AS [График по месяцам],

Switch(

[Время работы].[График по дням недели]=1, "Ежедневно",

[Время работы].[График по дням недели]=2, "С понедельника по субботу",

[Время работы].[График по дням недели]=3, "По будням",

[Время работы].[График по дням недели]=4, "По выходным",

[Время работы].[График по дням недели]=5, "По субботам",

[Время работы].[График по дням недели]=6, "По воскресеньям"

) AS [График по дням недели],

Format([Время работы].[Время начала работы], "hh:nn") AS [Время начала],

Format([Время работы].[Время окончания работы], "hh:nn") AS [Время окончания]

FROM

(([Время работы]

INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута])

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN [График по дням недели] ON [Время работы].[График по дням недели] = [График по дням недели].[Код графика]

WHERE

Маршруты.Работает = TRUE

AND [График по дням недели].Воскресенье = TRUE;

* список всех маршрутов, работающих после 22:00:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Switch(

[Время работы].[График по месяцам]=1, "Ежегодно",

[Время работы].[График по месяцам]=2, "В летний период",

[Время работы].[График по месяцам]=3, "В течение учебного года",

[Время работы].[График по месяцам]=4, "В зимний период"

) AS [График по месяцам],

Switch(

[Время работы].[График по дням недели]=1, "Ежедневно",

[Время работы].[График по дням недели]=2, "С понедельника по субботу",

[Время работы].[График по дням недели]=3, "По будням",

[Время работы].[График по дням недели]=4, "По выходным",

[Время работы].[График по дням недели]=5, "По субботам",

[Время работы].[График по дням недели]=6, "По воскресеньям"

) AS [График по дням недели],

Format([Время работы].[Время начала работы], "hh:nn") AS [Время начала],

Format([Время работы].[Время окончания работы], "hh:nn") AS [Время окончания]

FROM

([Время работы]

INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута])

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = TRUE

AND (

[Время работы].[Время окончания работы] > #22:00:00#

OR [Время работы].[Время окончания работы] < #01:00:00#

);

* список всех маршрутов, работающих до 06:00:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Switch(

[Время работы].[График по месяцам]=1, "Ежегодно",

[Время работы].[График по месяцам]=2, "В летний период",

[Время работы].[График по месяцам]=3, "В течение учебного года",

[Время работы].[График по месяцам]=4, "В зимний период"

) AS [График по месяцам],

Switch(

[Время работы].[График по дням недели]=1, "Ежедневно",

[Время работы].[График по дням недели]=2, "С понедельника по субботу",

[Время работы].[График по дням недели]=3, "По будням",

[Время работы].[График по дням недели]=4, "По выходным",

[Время работы].[График по дням недели]=5, "По субботам",

[Время работы].[График по дням недели]=6, "По воскресеньям"

) AS [График по дням недели],

Format([Время работы].[Время начала работы], "hh:nn") AS [Время начала],

Format([Время работы].[Время окончания работы], "hh:nn") AS [Время окончания]

FROM

([Время работы]

INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута])

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = TRUE

AND [Время работы].[Время начала работы] < #06:00:00#;

* список неработающих маршрутов:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

[Тип маршрута].[Тип маршрута],

Начальная.Остановка AS Начальная,

Конечная.Остановка AS Конечная

FROM

(((Маршруты

INNER JOIN Остановки AS Начальная ON Маршруты.Начальная = Начальная.[Код остановки])

INNER JOIN Остановки AS Конечная ON Маршруты.Конечная = Конечная.[Код остановки])

INNER JOIN [Тип маршрута] ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута])

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]

WHERE

Маршруты.Работает = FALSE;

* список всех маршрутов, где доступна бесплатная пересадка:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Тарифы.[Бесплатная пересадка]

FROM

((Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN Тарифы ON Маршруты.[Тариф] = Тарифы.[Код тарифа])

WHERE

Маршруты.Работает = TRUE

AND Тарифы.[Бесплатная пересадка] = TRUE;

* список всех маршрутов, где доступны проездные:

SELECT

Маршруты.[Номер маршрута],

[Тип транспорта].[Тип транспорта],

Тарифы.Проездные

FROM

((Маршруты

INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта])

INNER JOIN Тарифы ON Маршруты.[Тариф] = Тарифы.[Код тарифа])

WHERE

Маршруты.Работает = TRUE

AND Тарифы.Проездные = TRUE;

Функционал программы для Администратора представляет возможность редактирования базы данных: добавление, изменение и удаление записей. Данные возможности также реализованы через SQL-запросы:

* добавление записей:

INSERT INTO [ИмяТаблицы] ([Колонка1], [Колонка2], ...)

VALUES (Значение1, Значение2, ...);

* изменение записей:

UPDATE [ИмяТаблицы]

SET [Колонка1] = НовоеЗначение1, [Колонка2] = НовоеЗначение2, ...

WHERE [Код Колонки] = ЗначениеКода;

* удаление записей:

DELETE FROM [ИмяТаблицы]

WHERE [Код Колонки] = ЗначениеКода;

Полный текст программы представлен в прил. 2. Руководство пользователя представлено в прил. 3

# Заключение

В ходе выполнения курсовой работы была спроектирована база данных «Общественный транспорт г. Череповца», а также разработано программное обеспечение для работы с этой базой данных.

Созданная база данных позволяет хранить в удобном виде информацию о маршрутах общественного транспорта г. Череповца, расписаниях, тарифах, характеристиках подвижного состава. Приложение позволит пассажирам общественного транспорта быстро получать всю необходимую информацию из этой базы данных, что повысит комфорт при планировании поездок.

Программное обеспечение предназначено для установки на терминалах, которые могут быть размещены на остановках общественного транспорта, а также в самих транспортных средствах. Программа не требует подключения к интернету, так как база данных будет хранится локально на жестком диске терминала.

# Список литературы

1. Методика и организация самостоятельной работы студентов: учебно-методическое пособие / Е.В. Ершов, Л.Н. Виноградова, В.В. Селивановских [и др.]. – Череповец: ФГБОУ ВПО ЧГУ, 2015. – 243 с.
2. Знакомство с Microsoft Access – IT-Black [электр.ресурс] https://www.it-black.ru/tpost/nfipdi6fn1-znakomstvo-s-microsoft-access. Дата обращения: 15.04.2025.
3. История МУП «Череповецкая автоколонна №1456» – Автоколонна №1456 [электр.ресурс] https://avto1456.ru/company/istoriya/. Дата обращения: 03.03.2025.
4. История трамвая в Череповце – Электротранс [электр.ресурс] https://www.cher-tram.ru/history.html. Дата обращения: 03.03.2025.
5. Как разработать даталогическую модель без ошибок – Mksegment [электр.ресурс] https://mksegment.ru/d/kak-razrabotat-datalogicheskuyu-model-bazy-dannyh-sql-bez-oshibok. Дата обращения: 30.03.2025.
6. Как устроен язык SQL и почему он так востребован – Яндекс.Практикум [электр.ресурс] https://practicum.yandex.ru/blog/chto-takoe-sql/. Дата обращения: 20.04.2025.
7. Логическая модель данных: что это и как её создать – Skypro [электр.ресурс] https://sky.pro/wiki/sql/logicheskaya-model-bazy-dannyh-chto-eto-i-kak-eyo-sozdat/. Дата обращения: 08.04.2025.
8. Местоположение транспорта в режиме реального времени – 35bus.ru [электр.ресурс] https://35bus.ru/. Дата обращения: 03.03.2025.
9. Моделирование данных: что это и зачем нужно – Яндекс.Практикум [электр.ресурс] https://practicum.yandex.ru/blog/chto-takoe-modelirovanie-dannyh-i-kak-ego-ispolzovat/. Дата обращения: 15.03.2025.
10. Модель данных – что такое – SkyEng [электр.ресурс] https://skyeng.ru/magazine/wiki/it-industriya/chto-takoe-model-dannykh/. Дата обращения: 15.03.2025.
11. Нормализация данных: что это и зачем она нужна – Skypro [электр.ресурс] https://sky.pro/wiki/sql/normalizaciya-dannyh-chto-eto-i-zachem-ona-nuzhna/. Дата обращения: 02.04.2025.
12. Нормализация отношений. Шесть нормальных форм – Хабр [электр.ресурс] https://habr.com/ru/articles/254773/. Дата обращения: 02.04.2025.
13. Обзор Windows Forms – Microsoft Learn [электр.ресурс] https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/?view=netdesktop-9.0. Дата обращения: 20.04.2025.
14. Общественный транспорт – Википедия [электр.ресурс] https://ru.wikipedia.org/wiki/Общественный\_транспорт. Дата обращения: 03.03.2025.
15. Общественный транспорт Череповца – Яндекс.Карты [электр.ресурс] https://yandex.ru/maps/968/cherepovets/transport/. Дата обращения: 03.03.2025.
16. Общие сведения о программировании OLE DB – Microsoft Learn [электр.ресурс] https://learn.microsoft.com/ru-ru/cpp/data/oledb/ole-db-programming-overview. Дата обращения: 20.04.2025.
17. Основные особенности систем, основанных на инвертированных списках – CitForum [электр.ресурс] https://citforum.ru/database/osbd/glava\_10.shtml. Дата обращения: 15.03.2025.
18. Основные положения инфологического подхода к проектированию баз данных. Модель сущность-связь – LibQ [электр.ресурс] https://libq.ru/articles/informatsionnyie-sistemyi-i-tehnologii/osnovnye-polozheniya-infologicheskogo-podxoda-k-proektirovaniyu-baz-dannyx/. Дата обращения: 15.03.2025.
19. Основы C++ – Яндекс.Образование [электр.ресурс] https://education.yandex.ru/handbook/cpp/article/introduction. Дата обращения: 20.04.2025.
20. Система управления базами данных (СУБД): что это такое и зачем нужна – Cloud [электр.ресурс] https://cloud.ru/blog/chto-takoye-subd. Дата обращения: 15.04.2025.
21. Сущности и связи: как и для чего системные аналитики создают ER‑диаграммы – Яндекс.Практикум [электр.ресурс] https://practicum.yandex.ru/blog/chto-takoe-er-diagramma/. Дата обращения: 08.04.2025.
22. Физическая модель данных – AppMaster [электр.ресурс] https://appmaster.io/ru/glossary/fizicheskaia-model-dannykh. Дата обращения: 15.04.2025.
23. Череповец – Городской электротранспорт [электр.ресурс] https://transphoto.org/city/173/. Дата обращения: 03.03.2025.
24. Череповецкий автобус – Википедия [электр.ресурс] https://ru.wikipedia.org/wiki/Череповецкий\_автобус. Дата обращения: 03.03.2025.
25. Череповецкий трамвай – Википедия [электр.ресурс] https://ru.wikipedia.org/wiki/Череповецкий\_трамвай. Дата обращения: 03.03.2025.
26. Язык программирования C (Си): что это такое, где и для чего используется – Gitverse [электр.ресурс] https://gitverse.ru/blog/articles/development/519-yazyk-programmirovaniya-c-si-chto-eto-takoe-gde-i-dlya-chego-ispolzuetsya. Дата обращения: 20.04.2025.
27. Язык С#: где используют, что пишут, как появился и чем хорош – Skillbox [электр.ресурс] https://skillbox.ru/media/code/yazyk-s-gde-ispolzuyut-chto-pishut-kak-poyavilsya-i-chem-khorosh/. Дата обращения: 20.04.2025.
28. ER-диаграмма – Бизнес-секреты [электр.ресурс] https://secrets.tbank.ru/glossarij/er-diagramma/. Дата обращения: 08.04.2025.
29. ERwin Data Modeler – Википедия [электр.ресурс] https://ru.wikipedia.org/wiki/ERwin\_Data\_Modeler. Дата обращения: 08.04.2025.
30. Java. Рассказываем о языке, который завоевал мир программирования – Skillbox [электр.ресурс] https://blog.skillbox.by/kod/java-rasskazyvaem-o-jazyke-kotoryj-zavoeval-mir-programmirovanija/. Дата обращения: 20.04.2025.
31. MySQL – SkillFactory [электр.ресурс] https://blog.skillfactory.ru/glossary/mysql/. Дата обращения: 15.04.2025.
32. Oracle Database – SkillFactory [электр.ресурс] https://blog.skillfactory.ru/glossary/oracle-database/. Дата обращения: 15.04.2025.
33. PostgreSQL – SkillFactory [электр.ресурс] https://blog.skillfactory.ru/glossary/postgresql/. Дата обращения: 15.04.2025.
34. Python – SkillFactory [электр.ресурс] https://blog.skillfactory.ru/glossary/python/. Дата обращения: 20.04.2025.
35. SQLite – SkillFactory [электр.ресурс] https://blog.skillfactory.ru/glossary/sqlite/. Дата обращения: 15.04.2025.
36. Visual Studio: что это и для чего нужно – Skypro [электр.ресурс] https://sky.pro/wiki/javascript/visual-studio-chto-eto-i-dlya-chego-nuzhno/. Дата обращения: 20.04.2025.

Приложение 1

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

Образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

                                     Институт информационных технологий

*наименование института (факультета)*

                           Математическое и программное обеспечение ЭВМ

*наименование кафедры*

                               Проектирование баз данных

*наименование дисциплины в соответствии с учебным планом*

УТВЕРЖДАЮ

Зав. кафедрой                  ,

д. т. н., профессор                    Ершов Е. В.

«      »                       2025 г.

Проектирование базы данных общественного транспорта г. Череповца

Техническое задание на курсовую работу

Листов 5

|  |  |
| --- | --- |
| Руководитель | Селяничев О. Л. |
|  | Ф.И.О. преподавателя |
| Исполнитель |  |
| студент | 1ПИб-02-3оп-22 |
|  | группа |
|  | Маркелов С. А. |
|  | Фамилия, имя, отчество |

2025 год

Введение

Данная курсовая работа посвящена проектированию и нормализации базы данных общественного транспорта г. Череповца, а также разработке приложения для взаимодействия с этой базой данных. База данных поможет структурировать данные о маршрутах общественного транспорта, расписаниях, тарифах, способах оплаты и упростить получение данной информации для пассажиров.

1. Основания для разработки

Основанием для разработки является задание на курсовую работу по дисциплине «Проектирование баз данных», выданное на кафедре МПО ЭВМ ИИТ ЧГУ.

Дата утверждения: 21 февраля 2025 года.

Наименование темы разработки: Проектирование базы данных общественного транспорта г. Череповца.

2. Назначение разработки

Разрабатываемая база данных позволит улучшить доступность данных о маршрутах общественного транспорта города Череповца, тем самым ускорив и упростив поиск необходимой информации, и повысив уровень комфорта пассажиров при использовании общественного транспорта.

3. Требования к программе

* 1. Требования к функциональным характеристикам

Разрабатываемая база данных должна:

* содержать актуальную и достоверную информацию об общественном транспорте г. Череповца: списки маршрутов, остановок, расписания, характеристики транспортных средств, тарифы, способы оплаты и другие данные;
* состоять как минимум из 15 таблиц;
* быть в формате Microsoft Access.

Разрабатываемая программа должна:

* иметь доступ к базе данных;
* иметь выбор из 2 пользователей: «Пассажир» и «Администратор»;
* для пользователя «Пассажир» иметь возможность просмотра информации из базы данных через выполнение как минимум 30 запросов к ней;
* для пользователя «Администратор» иметь возможность добавления, редактирования и удаления записей в базе данных (для администратора);
* для пользователя «Администратор» запрашивать пароль при входе в программу, чтобы защитить информацию в базе данных;
* иметь удобный интерфейс в стиле Microsoft Office.
  1. Требования к надежности

Разрабатываемая база данных должна:

* быстро загружаться;
* корректно вносить записи;
* корректно отображать записи;
* корректно выполнять запросы.
  1. Условия эксплуатации

Для корректной работы базы данных и программы необходимо:

* наличие устройства, на котором возможно запускать файлы формата .exe (компьютер, ноутбук);
* наличие на устройстве операционной системы Windows 7, 8.1, 10 или 11.
  1. Требования к составу и параметрам технических средств

Для корректной работы базы данных необходимо:

* процессор не менее чем с 2 ядрами и тактовой частотой не ниже 1,6 ГГц;
* оперативная память: не менее 1 Гб для 32-разрядной ОС, 2 Гб для 64-разрядной ОС;
* 8 Гб свободного места на диске;
* разрешение экрана не менее 800 х 600;
* видеокарта с видеопамятью не менее 1 Гб;
* наличие мыши и клавиатуры.
  1. Требования к информационной и программной совместимости

Для корректной работы базы данных и программы необходимо:

* установленный на устройстве пакет библиотек «.NET Framework» версии 7.0 или выше;
* установленный на устройстве компонент доступа к базам данных «Microsoft Access Database Engine».
  1. Требования к маркировке и упаковке

База данных и программа распространяются на носителях информации объемом не менее 8 Гб (USB-флешка, CD-диск).

* 1. Требования к транспортированию и хранению

База данных и программа должны храниться на носителе информации объемом не менее 8 Гб (HDD или SSD).

* 1. Специальные требования

К разрабатываемой базе данных и программе предъявляются следующие требования:

* база данных регулярно обновляться;
* программа должна быть удобна для использования, в том числе для людей, плохо умеющих пользоваться компьютером.

4. Требование к программной документации

* 1. Содержание расчетно-пояснительной записки

Программная документация должна содержать расчётно-пояснительную записку с содержанием:

Титульный лист;

Оглавление;

Введение;

1. Описание предметной области;

2. Инфологическое проектирование;

3. Даталогическое проектирование;

4. Описание программы;

Заключение;

Список литературы;

Приложения.

* 1. Требования к оформлению

В данном пункте представлены требования к оформлению документации в соответствии с ГОСТ (табл. П1.1).

Таблица П1.1

Требования к оформлению

|  |  |
| --- | --- |
| Документ | Печать на отдельных листах формата А4 (210х297 мм); оборотная сторона не заполняется; листы нумеруются. Печать возможна ч/б. |
| Страницы | Ориентация – книжная; отдельные страницы, при необходимости, альбомная. Поля: верхнее, нижнее – по 2 см, левое – 3 см, правое – 1 см. |
| Абзацы | Межстрочный интервал – 1, перед и после абзаца – 0. |
| Шрифты | Кегль – 14. В таблицах шрифт 12. Шрифт листинга – 10 (возможно в 2 колонки). |
| Рисунки | Подписывается под ним по центру: Рис.Х. Название. В приложениях: Рис.П1.3. Название |
| Таблицы | Подписывается: над таблицей, выравнивание по правому: «Таблица Х». В следующей строке по центру Название Надписи в «шапке» (имена столбцов, полей) – по центру. В теле таблицы (записи) текстовые значения – выровнены по левому краю, числа, даты – по правому. |

5. Стадии и этапы разработки

В данном пункте описаны стадии и этапы разработки сайта (табл. П1.2).

Таблица П1.2

Стадии и этапы разработки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапа разработки | Сроки разработки | Результат выполнения | Отметка о выполнении |
| Определение темы для курсовой работы | 21.02.2025 | Определена тема курсовой работы |  |
| Оформление технического задания | 28.02.2025 | Оформленное техническое задание |  |
| Изучение предметной области | 07.03.2025 | Проведен анализ предметной области |  |
| Инфологическое проектирование | 21.03.2025 | Выбрана модель данных |  |
| Нормализация | 04.04.2025 | Готовые таблицы в третьей нормальной форме |  |
| Разработка логической модели | 11.04.2025 | Готовая логическая модель |  |
| Разработка физической модели | 18.04.2025 | Готовая физическая модель |  |
| Разработка программы | 25.04.2025 | Готовая программа |  |
| Тестирование программы | 16.05.2025 | Конечный вариант программы |  |
| Оформление сопроводительной документации | 20.05.2025 | Оформленная сопроводительная документация |  |

6. Порядок контроля и приемки

В данном пункте описан порядок контроля и приемки курсовой работы (табл. П1.3).

Таблица П1.3

Порядок контроля и приемки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование контрольного этапа выполнения курсовой работы | Сроки контроля | Результат выполнения | Отметка о приемке результата контрольного этапа |
| Сдача технического задания | 04.04.2025 | Согласованное техническое задание |  |
| Сдача расчетно-пояснительной записки | 20.05.2025 | Согласованная расчетно-пояснительная записка |  |
| Сдача курсовой работы | 22.05.2025 | Получение оценки за курсовую работу |  |

# Приложение 2

Листинг

Файл AdminPasswordForm.cs:

using System;

using System.Windows.Forms;

namespace CheTransApp

{

public partial class AdminPasswordForm : Form

{

public bool IsPasswordCorrect { get; private set; } = false;

public AdminPasswordForm()

{

InitializeComponent();

}

private void btnOK\_Click(object sender, EventArgs e)

{

if (txtPassword.Text == "1234")

{

IsPasswordCorrect = true;

DialogResult = DialogResult.OK;

}

else

{

MessageBox.Show("Неверный пароль!");

txtPassword.Clear();

}

}

private void btnCancel\_Click(object sender, EventArgs e)

{

DialogResult = DialogResult.Cancel;

}

private void txtPassword\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar == (char)Keys.Enter)

{

btnOK\_Click(sender, e);

e.Handled = true;

}

}

}

}

Файл AdminPasswordForm.Designer.cs:

namespace CheTransApp

{

partial class AdminPasswordForm

{

private System.ComponentModel.IContainer components = null;

private System.Windows.Forms.TextBox txtPassword;

private System.Windows.Forms.Label lblPassword;

private System.Windows.Forms.Button btnOK;

private System.Windows.Forms.Button btnCancel;

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

private void InitializeComponent()

{

this.txtPassword = new System.Windows.Forms.TextBox();

this.lblPassword = new System.Windows.Forms.Label();

this.btnOK = new System.Windows.Forms.Button();

this.btnCancel = new System.Windows.Forms.Button();

this.SuspendLayout();

//

// txtPassword

//

this.txtPassword.Location = new System.Drawing.Point(12, 25);

this.txtPassword.Name = "txtPassword";

this.txtPassword.PasswordChar = '\*';

this.txtPassword.Size = new System.Drawing.Size(200, 20);

this.txtPassword.TabIndex = 0;

this.txtPassword.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.txtPassword\_KeyPress);

//

// lblPassword

//

this.lblPassword.AutoSize = true;

this.lblPassword.Location = new System.Drawing.Point(12, 9);

this.lblPassword.Name = "lblPassword";

this.lblPassword.Size = new System.Drawing.Size(45, 13);

this.lblPassword.TabIndex = 1;

this.lblPassword.Text = "Пароль";

//

// btnOK

//

this.btnOK.Location = new System.Drawing.Point(12, 51);

this.btnOK.Name = "btnOK";

this.btnOK.Size = new System.Drawing.Size(95, 23);

this.btnOK.TabIndex = 2;

this.btnOK.Text = "OK";

this.btnOK.UseVisualStyleBackColor = true;

this.btnOK.Click += new System.EventHandler(this.btnOK\_Click);

//

// btnCancel

//

this.btnCancel.Location = new System.Drawing.Point(117, 51);

this.btnCancel.Name = "btnCancel";

this.btnCancel.Size = new System.Drawing.Size(95, 23);

this.btnCancel.TabIndex = 3;

this.btnCancel.Text = "Отмена";

this.btnCancel.UseVisualStyleBackColor = true;

this.btnCancel.Click += new System.EventHandler(this.btnCancel\_Click);

//

// AdminPasswordForm

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(224, 86);

this.Controls.Add(this.btnCancel);

this.Controls.Add(this.btnOK);

this.Controls.Add(this.lblPassword);

this.Controls.Add(this.txtPassword);

this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;

this.MaximizeBox = false;

this.MinimizeBox = false;

this.Name = "AdminPasswordForm";

this.StartPosition = System.Windows.Forms.FormStartPosition.CenterParent;

this.Text = "Ввод пароля";

this.ResumeLayout(false);

this.PerformLayout();

}

}

}

Файл EditRecordForm.cs:

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.OleDb;

using System.Drawing;

using System.Linq;

using System.Windows.Forms;

namespace CheTransApp

{

public partial class EditRecordForm : Form

{

private readonly string tableName;

private readonly Dictionary<string, object> recordData;

private readonly bool isEdit;

private readonly OleDbConnection connection;

private readonly List<Control> controls = new List<Control>();

private readonly List<string> autoIncrementFields = new List<string> { "Код времени работы", "Код времени суток", "Код графика", "Код расписания", "Код класса транспорта", "Код маршрута", "Код места оплаты", "Код направления", "Код населенного пункта", "Код остановки", "Код перевозчика", "Код способа оплаты", "Код тарифа", "Код типа маршрута", "Код типа пола", "Код типа топлива", "Код транспорта", "Код типа расписания", "Код улицы", "Код ТС" };

private readonly List<string> tableNames = new List<string> {

"Время работы",

"График по времени суток",

"График по дням недели",

"График по месяцам",

"Интервальное расписание",

"Класс транспорта",

"Маршруты",

"Места оплаты",

"Направления движения",

"Населенные пункты",

"Остановки",

"Перевозчики",

"Способы оплаты",

"Тарифы",

"Тип маршрута",

"Тип пола",

"Тип топлива",

"Тип транспорта",

"Типы расписаний",

"Точное расписание",

"Транспортные средства",

"Улицы"

};

// Явное описание внешних ключей для каждой таблицы: поле -> связанная таблица

private readonly Dictionary<string, Dictionary<string, string>> explicitForeignKeys = new Dictionary<string, Dictionary<string, string>>

{

{ "Время работы", new Dictionary<string, string> {

{ "Маршрут", "Маршруты" },

{ "График по месяцам", "График по месяцам" },

{ "График по дням недели", "График по дням недели" },

{ "Тип расписания", "Типы расписаний" }

}},

{ "Интервальное расписание", new Dictionary<string, string> {

{ "Время работы", "Время работы" },

{ "Время суток", "График по времени суток" }

}},

{ "Точное расписание", new Dictionary<string, string> {

{ "Время работы", "Время работы" },

{ "Направление", "Направления движения" }

}},

{ "Маршруты", new Dictionary<string, string> {

{ "Тип транспорта", "Тип транспорта" },

{ "Тип маршрута", "Тип маршрута" },

{ "Начальная", "Остановки" },

{ "Конечная", "Остановки" },

{ "Перевозчик", "Перевозчики" },

{ "Транспортные средства", "Транспортные средства" },

{ "Тариф", "Тарифы" },

{ "Способы оплаты", "Способы оплаты" },

{ "Места оплаты", "Места оплаты" }

}},

{ "Остановки", new Dictionary<string, string> {

{ "Улица", "Улицы" }

}},

{ "Улицы", new Dictionary<string, string> {

{ "Населенный пункт", "Населенные пункты" }

}},

{ "Транспортные средства", new Dictionary<string, string> {

{ "Класс транспорта", "Класс транспорта" },

{ "Тип пола", "Тип пола" },

{ "Тип топлива", "Тип топлива" }

}},

};

public Dictionary<string, object> ResultData { get; private set; }

public EditRecordForm(string tableName, Dictionary<string, object> recordData = null, OleDbConnection connection = null)

{

this.tableName = tableName;

this.recordData = recordData;

this.isEdit = recordData != null;

this.connection = connection;

this.Text = (isEdit ? "Изменить" : "Добавить") + $" запись: {tableName}";

BuildForm();

}

private void BuildForm()

{

// Получаем схему таблицы

DataTable schema = GetTableSchema();

int y = 15;

foreach (DataRow row in schema.Rows)

{

string columnName = row["COLUMN\_NAME"].ToString();

string dataType = row["DATA\_TYPE"].ToString();

bool isAutoInc = autoIncrementFields.Contains(columnName);

if (isAutoInc && !isEdit) continue; // Не показываем автоинкрементные поля при добавлении

Label lbl = new Label { Location = new Point(15, y + 3), AutoSize = true };

// Определяем текст для подписи

string labelText = columnName;

if (tableName == "Интервальное расписание" && columnName == "Время работы")

{

labelText = "Маршрут";

}

else if (tableName == "Точное расписание" && columnName == "Время работы")

{

labelText = "Маршрут";

}

lbl.Text = labelText;

Control ctrl;

if (dataType.Contains("Boolean"))

{

ctrl = new CheckBox { Location = new Point(180, y), Width = 200 };

if (isEdit && recordData.ContainsKey(columnName))

((CheckBox)ctrl).Checked = Convert.ToBoolean(recordData[columnName]);

}

else if (dataType.Contains("Date"))

{

ctrl = new DateTimePicker { Location = new Point(180, y), Width = 200, Format = DateTimePickerFormat.Custom, CustomFormat = "dd.MM.yyyy HH:mm" };

if (isEdit && recordData.ContainsKey(columnName))

((DateTimePicker)ctrl).Value = Convert.ToDateTime(recordData[columnName]);

}

else if (IsExplicitForeignKey(tableName, columnName))

{

ComboBox cb = new ComboBox { Location = new Point(180, y), Width = 200, DropDownStyle = ComboBoxStyle.DropDownList };

var items = GetForeignKeyItemsByExplicit(tableName, columnName);

cb.DataSource = new BindingSource(items, null);

cb.DisplayMember = "Value";

cb.ValueMember = "Key";

if (isEdit && recordData.ContainsKey(columnName))

cb.SelectedValue = recordData[columnName];

ctrl = cb;

}

else

{

ctrl = new TextBox { Location = new Point(180, y), Width = 200 };

if (isEdit && recordData.ContainsKey(columnName))

((TextBox)ctrl).Text = recordData[columnName]?.ToString();

}

ctrl.Name = "ctrl\_" + columnName;

// Делаем автоинкрементные поля недоступными для редактирования при изменении записи

if (isAutoInc && isEdit)

{

ctrl.Enabled = false; // Отключаем элемент управления

}

controls.Add(ctrl);

this.Controls.Add(lbl);

this.Controls.Add(ctrl);

y += 35;

}

Button btnOk = new Button { Text = "Сохранить", Location = new Point(60, y), DialogResult = DialogResult.OK };

Button btnCancel = new Button { Text = "Отмена", Location = new Point(180, y), DialogResult = DialogResult.Cancel };

btnOk.Click += BtnOk\_Click;

this.Controls.Add(btnOk);

this.Controls.Add(btnCancel);

this.ClientSize = new Size(400, y + 50);

}

private DataTable GetTableSchema()

{

using (var cmd = new OleDbCommand($"SELECT \* FROM [{tableName}] WHERE 1=0", connection))

using (var adapter = new OleDbDataAdapter(cmd))

{

DataTable dt = new DataTable();

adapter.Fill(dt);

DataTable schema = new DataTable();

schema.Columns.Add("COLUMN\_NAME", typeof(string));

schema.Columns.Add("DATA\_TYPE", typeof(string));

foreach (DataColumn col in dt.Columns)

{

var row = schema.NewRow();

row["COLUMN\_NAME"] = col.ColumnName;

row["DATA\_TYPE"] = col.DataType.Name;

schema.Rows.Add(row);

}

return schema;

}

}

private bool IsExplicitForeignKey(string table, string column)

{

return explicitForeignKeys.ContainsKey(table) && explicitForeignKeys[table].ContainsKey(column);

}

private Dictionary<int, string> GetForeignKeyItemsByExplicit(string table, string column)

{

string refTable = explicitForeignKeys[table][column];

var items = new Dictionary<int, string>();

// Определяем кастомные подписи для некоторых таблиц

List<string> customLabels = null;

if (refTable == "Способы оплаты")

{

customLabels = new List<string> {

"Только наличные", "Наличные и банковская карта", "Наличные и любые карты", "Все способы"

};

}

else if (refTable == "Места оплаты")

{

customLabels = new List<string> {

"Кондуктор", "Валидатор+Водитель", "Валидатор+Автомат", "Кондутор+Валидатор", "Водитель", "Автовокзал", "Автовокзал+Онлайн", "Кодуктор+Автовокзал", "Кондуктор+Автовокзал+Онлайн", "Водитель+Автовокзал", "Водитель+Автовокзал+Онлайн"

};

}

else if (refTable == "Тарифы")

{

customLabels = new List<string> {

"Городской с бесплатной пересадкой", "Городской без бесплатной пересадки", "Для коротких маршрутов", "Пригородный"

};

}

int index = 0;

using (var cmd = new OleDbCommand($"SELECT \* FROM [{refTable}]", connection))

using (var reader = cmd.ExecuteReader())

{

while (reader.Read())

{

int key = Convert.ToInt32(reader[0]);

string value;

if (customLabels != null && index < customLabels.Count)

{

value = customLabels[index];

}

else if (refTable == "График по месяцам")

value = GetMonthScheduleName(reader);

else if (refTable == "График по дням недели")

value = GetWeekScheduleName(reader);

else if (refTable == "Время работы")

value = GetWorkTimeRouteNumber(reader);

else if (refTable == "График по времени суток")

value = GetTimeOfDayName(reader);

else if (refTable == "Транспортные средства")

value = "Вариант " + reader[0].ToString(); // Используем ID с префиксом "Вариант " для транспортных средств

else

value = reader[GetDisplayField(refTable)]?.ToString();

items[key] = value;

index++;

}

}

return items;

}

private string GetMonthScheduleName(IDataRecord reader)

{

// Индексы: 1-12 — месяцы

bool[] months = new bool[12];

for (int i = 0; i < 12; i++)

months[i] = Convert.ToBoolean(reader[i + 1]);

if (months.All(x => x)) return "Круглогодично";

if (months[5] && months[6] && months[7] && months.Take(5).All(x => !x) && months.Skip(8).All(x => !x)) return "В летний период";

if (months[8] && months[9] && months[10] && months[11] && months[0] && months[1] && months[2] && months[3] && months[4] && !months[5] && !months[6] && !months[7]) return "В течение учебного года";

if (months[11] && months[0] && months[1] && months.Skip(2).All(x => !x)) return "В зимний период";

// Индивидуально — перечислить месяцы

string[] monthNames = { "Январь", "Февраль", "Март", "Апрель", "Май", "Июнь", "Июль", "Август", "Сентябрь", "Октябрь", "Ноябрь", "Декабрь" };

var active = monthNames.Where((m, i) => months[i]).ToArray();

return string.Join(", ", active);

}

private string GetWeekScheduleName(IDataRecord reader)

{

// Индексы: 1-7 — дни недели

bool[] days = new bool[7];

for (int i = 0; i < 7; i++)

days[i] = Convert.ToBoolean(reader[i + 1]);

if (days.All(x => x)) return "Ежедневно";

if (days.Take(6).All(x => x) && !days[6]) return "С понедельника по субботу";

if (days.Take(5).All(x => x) && !days[5] && !days[6]) return "По будням";

if (!days[0] && !days[1] && !days[2] && !days[3] && !days[4] && days[5] && days[6]) return "По выходным";

if (!days[0] && !days[1] && !days[2] && !days[3] && !days[4] && days[5] && !days[6]) return "По субботам";

if (!days[0] && !days[1] && !days[2] && !days[3] && !days[4] && !days[5] && days[6]) return "По воскресеньям";

// Индивидуально — перечислить дни

string[] dayNames = { "Пн", "Вт", "Ср", "Чт", "Пт", "Сб", "Вс" };

var active = dayNames.Where((d, i) => days[i]).ToArray();

return string.Join(", ", active);

}

private string GetWorkTimeRouteNumber(IDataRecord reader)

{

// Читаем поле Маршрут (второе поле после ключа)

object routeId = reader[1];

if (routeId == DBNull.Value) return "";

int id = Convert.ToInt32(routeId);

// Ищем номер маршрута в таблице Маршруты по ID

using (var cmd = new OleDbCommand($"SELECT [Номер маршрута] FROM [Маршруты] WHERE [Код маршрута] = ?", connection))

{

cmd.Parameters.AddWithValue("@id", id);

var result = cmd.ExecuteScalar();

return result?.ToString() ?? "";

}

}

private string GetTimeOfDayName(IDataRecord reader)

{

// Читаем Время начала и Время окончания интервала

DateTime startTime = Convert.ToDateTime(reader["Начало интервала"]);

DateTime endTime = Convert.ToDateTime(reader["Конец интервала"]);

// Определяем название по времени

if (startTime.TimeOfDay == new TimeSpan(5, 0, 0) && endTime.TimeOfDay == new TimeSpan(10, 0, 0))

return "05:00-10:00";

if (startTime.TimeOfDay == new TimeSpan(10, 0, 0) && endTime.TimeOfDay == new TimeSpan(15, 0, 0))

return "10:00-15:00";

if (startTime.TimeOfDay == new TimeSpan(15, 0, 0) && endTime.TimeOfDay == new TimeSpan(19, 0, 0))

return "15:00-19:00";

if (startTime.TimeOfDay == new TimeSpan(19, 0, 0) && endTime.TimeOfDay == new TimeSpan(0, 0, 0))

return "19:00-00:00";

if (startTime.TimeOfDay == new TimeSpan(0, 0, 0) && endTime.TimeOfDay == new TimeSpan(23, 59, 59))

return "Весь день";

return $"{startTime:HH:mm} - {endTime:HH:mm}"; // По умолчанию показываем интервал

}

private string GetDisplayField(string table)

{

// Простая эвристика: ищем первое текстовое поле, иначе первое поле после ключа

using (var cmd = new OleDbCommand($"SELECT \* FROM [{table}] WHERE 1=0", connection))

using (var adapter = new OleDbDataAdapter(cmd))

{

DataTable dt = new DataTable();

adapter.Fill(dt);

foreach (DataColumn col in dt.Columns)

{

if (col.DataType == typeof(string) && !col.ColumnName.StartsWith("Код "))

return col.ColumnName;

}

// fallback: второе поле

if (dt.Columns.Count > 1)

return dt.Columns[1].ColumnName;

return dt.Columns[0].ColumnName;

}

}

private void BtnOk\_Click(object sender, EventArgs e)

{

ResultData = new Dictionary<string, object>();

foreach (var ctrl in controls)

{

string col = ctrl.Name.Substring(5);

object val = null;

if (ctrl is TextBox tb)

val = tb.Text;

else if (ctrl is CheckBox cb)

val = cb.Checked;

else if (ctrl is DateTimePicker dtp)

val = dtp.Value;

else if (ctrl is ComboBox cmb)

val = ((KeyValuePair<int, string>)cmb.SelectedItem).Key;

ResultData[col] = val;

}

this.DialogResult = DialogResult.OK;

}

}

}

Файл LoginForm.cs:

using System;

using System.Windows.Forms;

namespace CheTransApp

{

public partial class LoginForm : Form

{

public string UserRole { get; private set; } = "Пассажир";

public LoginForm()

{

InitializeComponent();

}

private void btnPassenger\_Click(object sender, EventArgs e)

{

UserRole = "Пассажир";

DialogResult = DialogResult.OK;

}

private void btnAdmin\_Click(object sender, EventArgs e)

{

using (var passwordForm = new AdminPasswordForm())

{

if (passwordForm.ShowDialog() == DialogResult.OK)

{

UserRole = "Администратор";

DialogResult = DialogResult.OK;

}

}

}

}

}

Файл LoginForm.Designer.cs:

namespace CheTransApp

{

partial class LoginForm

{

private System.ComponentModel.IContainer components = null;

private System.Windows.Forms.Button btnPassenger;

private System.Windows.Forms.Button btnAdmin;

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

private void InitializeComponent()

{

this.btnPassenger = new System.Windows.Forms.Button();

this.btnAdmin = new System.Windows.Forms.Button();

this.SuspendLayout();

//

// btnPassenger

//

this.btnPassenger.Location = new System.Drawing.Point(30, 30);

this.btnPassenger.Name = "btnPassenger";

this.btnPassenger.Size = new System.Drawing.Size(200, 30);

this.btnPassenger.TabIndex = 0;

this.btnPassenger.Text = "Войти как пассажир";

this.btnPassenger.UseVisualStyleBackColor = true;

this.btnPassenger.Click += new System.EventHandler(this.btnPassenger\_Click);

//

// btnAdmin

//

this.btnAdmin.Location = new System.Drawing.Point(30, 80);

this.btnAdmin.Name = "btnAdmin";

this.btnAdmin.Size = new System.Drawing.Size(200, 30);

this.btnAdmin.TabIndex = 1;

this.btnAdmin.Text = "Войти как администратор";

this.btnAdmin.UseVisualStyleBackColor = true;

this.btnAdmin.Click += new System.EventHandler(this.btnAdmin\_Click);

//

// LoginForm

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(264, 141);

this.Controls.Add(this.btnAdmin);

this.Controls.Add(this.btnPassenger);

this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;

this.MaximizeBox = false;

this.MinimizeBox = false;

this.Name = "LoginForm";

this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;

this.Text = "Авторизация";

this.ResumeLayout(false);

}

}

}

Файл MainForm.cs:

using System;

using System.Data;

using System.Data.OleDb;

using System.Windows.Forms;

using System.Linq;

using System.Collections.Generic;

namespace CheTransApp

{

public partial class MainForm : Form

{

private string connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=CheTrans.accdb;";

private string userRole;

private readonly string[] tableNames = new string[]

{

"Время работы",

"График по времени суток",

"График по дням недели",

"График по месяцам",

"Интервальное расписание",

"Класс транспорта",

"Маршруты",

"Места оплаты",

"Направления движения",

"Населенные пункты",

"Остановки",

"Перевозчики",

"Способы оплаты",

"Тарифы",

"Тип маршрута",

"Тип пола",

"Тип топлива",

"Тип транспорта",

"Типы расписаний",

"Точное расписание",

"Транспортные средства",

"Улицы"

};

public MainForm(string role)

{

InitializeComponent();

userRole = role;

InitUI();

btnAdd.Click += btnAdd\_Click;

btnEdit.Click += btnEdit\_Click;

btnShowTable.Click += btnShowTable\_Click;

btnDelete.Click += btnDelete\_Click;

}

private void InitUI()

{

if (userRole == "Пассажир")

{

comboBoxQueries.Visible = true;

btnShowQuery.Visible = true;

comboBoxTables.Visible = false;

btnShowTable.Visible = false;

btnAdd.Visible = btnEdit.Visible = btnDelete.Visible = false;

dataGridView1.ReadOnly = true;

dataGridView1.AllowUserToAddRows = false;

dataGridView1.AllowUserToDeleteRows = false;

dataGridView1.AllowUserToOrderColumns = false;

dataGridView1.AllowUserToResizeColumns = false;

dataGridView1.AllowUserToResizeRows = false;

dataGridView1.MultiSelect = false;

dataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

comboBoxQueries.Items.Clear();

comboBoxQueries.Items.AddRange(new string[] {

"Список всех маршрутов с конечными",

"Список всех автобусов с конечными",

"Список всех трамваев с конечными",

"Список всех городских маршрутов",

"Список всех пригородных маршрутов",

"Список всех маршрутов с тарифами",

"Список всех маршрутов со способами оплаты",

"Список всех маршрутов с местами оплаты",

"Список всех перевозчиков",

"Список всех остановок с улицами и населенными пунктами",

"Список всех маршрутов по транспортным средствам",

"Список всех маршрутов по перевозчикам",

"Список всех маршрутов по времени работы",

"Список всех маршрутов с интервальным расписанием",

"Список всех маршрутов, работающих летом",

"Список всех маршрутов с низкопольным транспортом",

"Список всех маршрутов с транспортом на газу",

"Список всех маршрутов с транспортом на электричестве",

"Список всех маршрутов с кондиционерами",

"Список всех маршрутов с USB-зарядками",

"Список всех маршрутов с пандусами",

"Список всех маршрутов с Wi-Fi",

"Список всех маршрутов с валидаторами",

"Список всех маршрутов с кондукторами",

"Список всех маршрутов, работающих по воскресеньям",

"Список всех маршрутов, работающих после 22:00",

"Список всех маршрутов, работающих до 06:00",

"Список неработающих маршрутов",

"Список всех маршрутов, где доступна бесплатная пересадка",

"Список всех маршрутов, где доступны проездные"

});

if (comboBoxQueries.Items.Count > 0)

comboBoxQueries.SelectedIndex = 0;

}

else

{

comboBoxQueries.Visible = false;

btnShowQuery.Visible = false;

comboBoxTables.Visible = true;

btnShowTable.Visible = true;

btnAdd.Visible = btnEdit.Visible = btnDelete.Visible = true;

comboBoxTables.Items.Clear();

comboBoxTables.Items.AddRange(tableNames);

if (comboBoxTables.Items.Count > 0)

comboBoxTables.SelectedIndex = 0;

comboBoxTables.SelectedIndexChanged -= comboBoxTables\_SelectedIndexChanged;

}

}

private void comboBoxTables\_SelectedIndexChanged(object sender, EventArgs e)

{

if (userRole == "Пассажир")

{

btnShowQuery\_Click(sender, e);

}

}

private void btnShowTable\_Click(object sender, EventArgs e)

{

string table = comboBoxTables.SelectedItem as string;

if (!string.IsNullOrEmpty(table))

{

string query = $"SELECT \* FROM [{table}]";

dataGridView1.DataSource = ExecuteQuery(query);

}

}

private void btnShowQuery\_Click(object sender, EventArgs e)

{

int idx = comboBoxQueries.SelectedIndex;

if (idx >= 0 && idx < Queries.QueryList.Count)

dataGridView1.DataSource = ExecuteQuery(Queries.QueryList[idx]);

}

public DataTable ExecuteQuery(string query)

{

using (OleDbConnection conn = new OleDbConnection(connectionString))

{

conn.Open();

OleDbDataAdapter adapter = new OleDbDataAdapter(query, conn);

DataTable dt = new DataTable();

adapter.Fill(dt);

return dt;

}

}

private string GetQueryByIndex(int index)

{

switch (index)

{

case 0: return "SELECT \* FROM Перевозчики";

case 1: return "SELECT \* FROM Маршруты";

default: return "";

}

}

private void btnAdd\_Click(object sender, EventArgs e)

{

string table = comboBoxTables.SelectedItem as string;

if (string.IsNullOrEmpty(table)) return;

using (var conn = new OleDbConnection(connectionString))

{

conn.Open();

using (var form = new EditRecordForm(table, null, conn))

{

if (form.ShowDialog() == DialogResult.OK)

{

// Сформировать INSERT

var data = form.ResultData;

var columns = string.Join(", ", data.Keys.Select(k => $"[{k}]"));

var values = string.Join(", ", data.Keys.Select(k => "?"));

string sql = $"INSERT INTO [{table}] ({columns}) VALUES ({values})";

using (var cmd = new OleDbCommand(sql, conn))

{

foreach (var kv in data)

cmd.Parameters.AddWithValue("", kv.Value ?? DBNull.Value);

cmd.ExecuteNonQuery();

}

// Обновить таблицу

dataGridView1.DataSource = null;

dataGridView1.DataSource = ExecuteQuery($"SELECT \* FROM [{table}]");

dataGridView1.Refresh();

}

}

}

}

private void btnEdit\_Click(object sender, EventArgs e)

{

string table = comboBoxTables.SelectedItem as string;

if (string.IsNullOrEmpty(table) || dataGridView1.SelectedRows.Count == 0) return;

var row = dataGridView1.SelectedRows[0];

var data = new Dictionary<string, object>();

foreach (DataGridViewCell cell in row.Cells)

data[cell.OwningColumn.HeaderText] = cell.Value;

using (var conn = new OleDbConnection(connectionString))

{

conn.Open();

using (var form = new EditRecordForm(table, data, conn))

{

if (form.ShowDialog() == DialogResult.OK)

{

// Сформировать UPDATE

var newData = form.ResultData;

var setClause = string.Join(", ", newData.Keys.Select(k => $"[{k}]=?"));

// Найти имя ключевого поля

string keyField = data.Keys.FirstOrDefault(k => k.StartsWith("Код "));

if (string.IsNullOrEmpty(keyField)) return;

string sql = $"UPDATE [{table}] SET {setClause} WHERE [{keyField}]=?";

using (var cmd = new OleDbCommand(sql, conn))

{

foreach (var kv in newData)

{

// Находим порядок параметра, соответствующий полю в SET-выражении

// Это важно, так как порядок параметров должен совпадать с порядком полей в SET

var key = kv.Key;

// Добавляем параметр для значения поля

cmd.Parameters.AddWithValue("", kv.Value ?? DBNull.Value);

}

// Добавляем параметр для условия WHERE (первичный ключ)

cmd.Parameters.AddWithValue("", data[keyField]);

cmd.ExecuteNonQuery();

}

// Обновить таблицу

dataGridView1.DataSource = null;

dataGridView1.DataSource = ExecuteQuery($"SELECT \* FROM [{table}]");

dataGridView1.Refresh();

}

}

}

}

// Обработчик кнопки Удалить

private void btnDelete\_Click(object sender, EventArgs e)

{

string table = comboBoxTables.SelectedItem as string;

if (string.IsNullOrEmpty(table) || dataGridView1.SelectedRows.Count == 0) return;

// Запрос подтверждения

DialogResult result = MessageBox.Show(

"Вы уверены, что хотите удалить выбранную запись?",

"Подтверждение удаления",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

var row = dataGridView1.SelectedRows[0];

var data = new Dictionary<string, object>();

foreach (DataGridViewCell cell in row.Cells)

data[cell.OwningColumn.HeaderText] = cell.Value;

// Найти имя ключевого поля

string keyField = data.Keys.FirstOrDefault(k => k.StartsWith("Код "));

if (string.IsNullOrEmpty(keyField)) return;

// Получить значение первичного ключа

object keyValue = data[keyField];

if (keyValue == null || keyValue == DBNull.Value) return; // Невозможно удалить без ключа

// Сформировать DELETE запрос

string sql = $"DELETE FROM [{table}] WHERE [{keyField}]=?";

using (var conn = new OleDbConnection(connectionString))

{

try

{

conn.Open();

using (var cmd = new OleDbCommand(sql, conn))

{

cmd.Parameters.AddWithValue("", keyValue);

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

// Обновить таблицу после успешного удаления

dataGridView1.DataSource = null;

dataGridView1.DataSource = ExecuteQuery($"SELECT \* FROM [{table}]");

dataGridView1.Refresh();

MessageBox.Show("Запись успешно удалена.", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Не удалось удалить запись.", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при удалении записи: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

}

}

Файл MainForm.Designer.cs:

namespace CheTransApp

{

partial class MainForm

{

private System.ComponentModel.IContainer components = null;

private System.Windows.Forms.DataGridView dataGridView1;

private System.Windows.Forms.ComboBox comboBoxQueries;

private System.Windows.Forms.Button btnShowQuery;

private System.Windows.Forms.Button btnAdd;

private System.Windows.Forms.Button btnEdit;

private System.Windows.Forms.Button btnDelete;

private System.Windows.Forms.ComboBox comboBoxTables;

private System.Windows.Forms.Button btnShowTable;

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

private void InitializeComponent()

{

this.dataGridView1 = new System.Windows.Forms.DataGridView();

this.comboBoxQueries = new System.Windows.Forms.ComboBox();

this.btnShowQuery = new System.Windows.Forms.Button();

this.btnAdd = new System.Windows.Forms.Button();

this.btnEdit = new System.Windows.Forms.Button();

this.btnDelete = new System.Windows.Forms.Button();

this.comboBoxTables = new System.Windows.Forms.ComboBox();

this.btnShowTable = new System.Windows.Forms.Button();

((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).BeginInit();

this.SuspendLayout();

//

// dataGridView1

//

this.dataGridView1.ColumnHeadersHeightSizeMode = System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;

this.dataGridView1.Location = new System.Drawing.Point(12, 60);

this.dataGridView1.Name = "dataGridView1";

this.dataGridView1.Size = new System.Drawing.Size(760, 350);

this.dataGridView1.TabIndex = 0;

//

// comboBoxQueries

//

this.comboBoxQueries.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

this.comboBoxQueries.FormattingEnabled = true;

this.comboBoxQueries.Location = new System.Drawing.Point(12, 12);

this.comboBoxQueries.Name = "comboBoxQueries";

this.comboBoxQueries.Size = new System.Drawing.Size(400, 21);

this.comboBoxQueries.TabIndex = 1;

//

// btnShowQuery

//

this.btnShowQuery.Location = new System.Drawing.Point(420, 10);

this.btnShowQuery.Name = "btnShowQuery";

this.btnShowQuery.Size = new System.Drawing.Size(120, 25);

this.btnShowQuery.TabIndex = 2;

this.btnShowQuery.Text = "Показать";

this.btnShowQuery.UseVisualStyleBackColor = true;

this.btnShowQuery.Click += new System.EventHandler(this.btnShowQuery\_Click);

//

// btnAdd

//

this.btnAdd.Location = new System.Drawing.Point(560, 10);

this.btnAdd.Name = "btnAdd";

this.btnAdd.Size = new System.Drawing.Size(70, 25);

this.btnAdd.TabIndex = 3;

this.btnAdd.Text = "Добавить";

this.btnAdd.UseVisualStyleBackColor = true;

//

// btnEdit

//

this.btnEdit.Location = new System.Drawing.Point(640, 10);

this.btnEdit.Name = "btnEdit";

this.btnEdit.Size = new System.Drawing.Size(70, 25);

this.btnEdit.TabIndex = 4;

this.btnEdit.Text = "Изменить";

this.btnEdit.UseVisualStyleBackColor = true;

//

// btnDelete

//

this.btnDelete.Location = new System.Drawing.Point(720, 10);

this.btnDelete.Name = "btnDelete";

this.btnDelete.Size = new System.Drawing.Size(70, 25);

this.btnDelete.TabIndex = 5;

this.btnDelete.Text = "Удалить";

this.btnDelete.UseVisualStyleBackColor = true;

//

// comboBoxTables

//

this.comboBoxTables.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

this.comboBoxTables.FormattingEnabled = true;

this.comboBoxTables.Location = new System.Drawing.Point(12, 12);

this.comboBoxTables.Name = "comboBoxTables";

this.comboBoxTables.Size = new System.Drawing.Size(400, 21);

this.comboBoxTables.TabIndex = 6;

//

// btnShowTable

//

this.btnShowTable.Location = new System.Drawing.Point(420, 10);

this.btnShowTable.Name = "btnShowTable";

this.btnShowTable.Size = new System.Drawing.Size(120, 25);

this.btnShowTable.TabIndex = 7;

this.btnShowTable.Text = "Показать";

this.btnShowTable.UseVisualStyleBackColor = true;

//

// MainForm

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(800, 460);

this.Controls.Add(this.btnShowTable);

this.Controls.Add(this.comboBoxTables);

this.Controls.Add(this.btnDelete);

this.Controls.Add(this.btnEdit);

this.Controls.Add(this.btnAdd);

this.Controls.Add(this.btnShowQuery);

this.Controls.Add(this.comboBoxQueries);

this.Controls.Add(this.dataGridView1);

this.Name = "MainForm";

this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;

this.Text = "CheTrans - Транспортная система";

((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).EndInit();

this.ResumeLayout(false);

}

}

}

Файл Program.cs:

using System;

using System.Windows.Forms;

namespace CheTransApp

{

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

using (var login = new LoginForm())

{

if (login.ShowDialog() == DialogResult.OK)

{

Application.Run(new MainForm(login.UserRole));

}

}

}

}

}

Файл Queries.cs:

using System.Collections.Generic;

namespace CheTransApp

{

public static class Queries

{

public static List<string> QueryList = new List<string>

{

// 1. Список всех маршрутов с конечными

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип маршрута].[Тип маршрута], Начальная.Остановка AS Начальная, Конечная.Остановка AS Конечная FROM (((Маршруты INNER JOIN Остановки AS Начальная ON Маршруты.Начальная = Начальная.[Код остановки]) INNER JOIN Остановки AS Конечная ON Маршруты.Конечная = Конечная.[Код остановки]) INNER JOIN [Тип маршрута] ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = TRUE",

// 2. Список всех автобусов с конечными

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип маршрута].[Тип маршрута], Начальная.Остановка AS Начальная, Конечная.Остановка AS Конечная FROM (((Маршруты INNER JOIN Остановки AS Начальная ON Маршруты.Начальная = Начальная.[Код остановки]) INNER JOIN Остановки AS Конечная ON Маршруты.Конечная = Конечная.[Код остановки]) INNER JOIN [Тип маршрута] ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = TRUE AND [Тип транспорта].[Тип транспорта] = 'Автобус'",

// 3. Список всех трамваев с конечными

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип маршрута].[Тип маршрута], Начальная.Остановка AS Начальная, Конечная.Остановка AS Конечная FROM (((Маршруты INNER JOIN Остановки AS Начальная ON Маршруты.Начальная = Начальная.[Код остановки]) INNER JOIN Остановки AS Конечная ON Маршруты.Конечная = Конечная.[Код остановки]) INNER JOIN [Тип маршрута] ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = TRUE AND [Тип транспорта].[Тип транспорта] = 'Трамвай'",

// 4. Список всех городских маршрутов

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип маршрута].[Тип маршрута], Начальная.Остановка AS Начальная, Конечная.Остановка AS Конечная FROM (((Маршруты INNER JOIN Остановки AS Начальная ON Маршруты.Начальная = Начальная.[Код остановки]) INNER JOIN Остановки AS Конечная ON Маршруты.Конечная = Конечная.[Код остановки]) INNER JOIN [Тип маршрута] ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = TRUE AND [Тип маршрута].[Тип маршрута] = 'Городской'",

// 5. Список всех пригородных маршрутов

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип маршрута].[Тип маршрута], Начальная.Остановка AS Начальная, Конечная.Остановка AS Конечная FROM (((Маршруты INNER JOIN Остановки AS Начальная ON Маршруты.Начальная = Начальная.[Код остановки]) INNER JOIN Остановки AS Конечная ON Маршруты.Конечная = Конечная.[Код остановки]) INNER JOIN [Тип маршрута] ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = TRUE AND [Тип маршрута].[Тип маршрута] = 'Пригородный'",

// 6. Список всех маршрутов с тарифами

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Тарифы.Наличный, Тарифы.Безналичный, Тарифы.Льготный, Тарифы.Проездные, Тарифы.[Бесплатная пересадка] FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN Тарифы ON Маршруты.[Тариф] = Тарифы.[Код тарифа]) WHERE Маршруты.Работает = TRUE",

// 7. Список всех маршрутов со способами оплаты

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Способы оплаты].Наличные, [Способы оплаты].[Банковская карта], [Способы оплаты].[Транспортная карта], [Способы оплаты].[QR-код] FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Способы оплаты] ON Маршруты.[Способы оплаты] = [Способы оплаты].[Код способа оплаты]) WHERE Маршруты.Работает = TRUE",

// 8. Список всех маршрутов с местами оплаты

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Места оплаты].Кондуктор, [Места оплаты].Валидатор, [Места оплаты].Водитель, [Места оплаты].Автомат, [Места оплаты].Автовокзал, [Места оплаты].Онлайн FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Места оплаты] ON Маршруты.[Места оплаты] = [Места оплаты].[Код места оплаты]) WHERE Маршруты.Работает = TRUE",

// 9. Список всех перевозчиков

"SELECT Перевозчик FROM Перевозчики",

// 10. Список всех остановок с улицами и населенными пунктами

"SELECT Остановки.Остановка, Улицы.Улица, [Населенные пункты].[Населенный пункт] FROM (Остановки INNER JOIN Улицы ON Остановки.Улица = Улицы.[Код улицы]) INNER JOIN [Населенные пункты] ON Улицы.[Населенный пункт] = [Населенные пункты].[Код населенного пункта]",

// 11. Список всех маршрутов по транспортным средствам

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Класс транспорта].[Класс транспорта], [Тип пола].[Тип пола], [Тип топлива].[Тип топлива], [Транспортные средства].Кондиционер, [Транспортные средства].Пандус, [Транспортные средства].[Wi-Fi], [Транспортные средства].[USB-зарядки], [Транспортные средства].Автоинформатор, [Транспортные средства].[Информационные экраны] FROM ((((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]) INNER JOIN [Класс транспорта] ON [Транспортные средства].[Класс транспорта] = [Класс транспорта].[Код класса транспорта]) INNER JOIN [Тип пола] ON [Транспортные средства].[Тип пола] = [Тип пола].[Код типа пола]) INNER JOIN [Тип топлива] ON [Транспортные средства].[Тип топлива] = [Тип топлива].[Код типа топлива] WHERE Маршруты.Работает = TRUE ORDER BY Маршруты.[Код маршрута]",

// 12. Список всех маршрутов по перевозчикам

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Перевозчики.Перевозчик FROM (Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN Перевозчики ON Маршруты.Перевозчик = Перевозчики.[Код перевозчика] WHERE Маршруты.Работает = TRUE",

// 13. Список всех маршрутов по времени работы

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Switch([Время работы].[График по месяцам]=1,\"Ежегодно\", [Время работы].[График по месяцам]=2,\"В летний период\", [Время работы].[График по месяцам]=3,\"В течение учебного года\", [Время работы].[График по месяцам]=4,\"В зимний период\") AS [График по месяцам], Switch([Время работы].[График по дням недели]=1,\"Ежедневно\", [Время работы].[График по дням недели]=2,\"С понедельника по субботу\", [Время работы].[График по дням недели]=3,\"По будням\", [Время работы].[График по дням недели]=4,\"По выходным\", [Время работы].[График по дням недели]=5,\"По субботам\", [Время работы].[График по дням недели]=6,\"По воскресеньям\") AS [График по дням недели], Format([Время работы].[Время начала работы],\"hh:nn\") AS [Время начала], Format([Время работы].[Время окончания работы],\"hh:nn\") AS [Время окончания] FROM ([Время работы] INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = TRUE",

// 14. Список всех маршрутов с интервальным расписанием

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Switch([Время работы].[График по дням недели]=1,\"Ежедневно\",[Время работы].[График по дням недели]=2,\"С понедельника по субботу\",[Время работы].[График по дням недели]=3,\"По будням\",[Время работы].[График по дням недели]=4,\"По выходным\",[Время работы].[График по дням недели]=5,\"По субботам\",[Время работы].[График по дням недели]=6,\"По воскресеньям\") AS [График по дням недели], IIf(SUM(IIf([Интервальное расписание].[Время суток]=1 OR [Интервальное расписание].[Время суток]=5,[Интервальное расписание].[Интервал],0))=0,Null,SUM(IIf([Интервальное расписание].[Время суток]=1 OR [Интервальное расписание].[Время суток]=5,[Интервальное расписание].[Интервал],0))) AS [05:00-10:00], IIf(SUM(IIf([Интервальное расписание].[Время суток]=2 OR [Интервальное расписание].[Время суток]=5,[Интервальное расписание].[Интервал],0))=0,Null,SUM(IIf([Интервальное расписание].[Время суток]=2 OR [Интервальное расписание].[Время суток]=5,[Интервальное расписание].[Интервал],0))) AS [10:00-15:00], IIf(SUM(IIf([Интервальное расписание].[Время суток]=3 OR [Интервальное расписание].[Время суток]=5,[Интервальное расписание].[Интервал],0))=0,Null,SUM(IIf([Интервальное расписание].[Время суток]=3 OR [Интервальное расписание].[Время суток]=5,[Интервальное расписание].[Интервал],0))) AS [15:00-19:00], IIf(SUM(IIf([Интервальное расписание].[Время суток]=4 OR [Интервальное расписание].[Время суток]=5,[Интервальное расписание].[Интервал],0))=0,Null,SUM(IIf([Интервальное расписание].[Время суток]=4 OR [Интервальное расписание].[Время суток]=5,[Интервальное расписание].[Интервал],0))) AS [19:00-00:00] FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Время работы] ON Маршруты.[Код маршрута] = [Время работы].[Маршрут]) INNER JOIN [Интервальное расписание] ON [Время работы].[Код времени работы] = [Интервальное расписание].[Время работы] WHERE [Время работы].[Тип расписания] = (SELECT [Код типа расписания] FROM [Типы расписаний] WHERE [Тип расписания] = 'Интервальное') AND Маршруты.Работает = TRUE GROUP BY Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Время работы].[График по дням недели]",

// 15. Список всех маршрутов, работающих летом

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Switch([Время работы].[График по месяцам]=1,\"Ежегодно\",[Время работы].[График по месяцам]=2,\"В летний период\",[Время работы].[График по месяцам]=3,\"В течение учебного года\",[Время работы].[График по месяцам]=4,\"В зимний период\") AS [График по месяцам], Switch([Время работы].[График по дням недели]=1,\"Ежедневно\",[Время работы].[График по дням недели]=2,\"С понедельника по субботу\",[Время работы].[График по дням недели]=3,\"По будням\",[Время работы].[График по дням недели]=4,\"По выходным\",[Время работы].[График по дням недели]=5,\"По субботам\",[Время работы].[График по дням недели]=6,\"По воскресеньям\") AS [График по дням недели], Format([Время работы].[Время начала работы],\"hh:nn\") AS [Время начала], Format([Время работы].[Время окончания работы],\"hh:nn\") AS [Время окончания] FROM (([Время работы] INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [График по месяцам] ON [Время работы].[График по месяцам] = [График по месяцам].[Код графика] WHERE Маршруты.Работает = TRUE AND [График по месяцам].Июнь = TRUE AND [График по месяцам].Июль = TRUE AND [График по месяцам].Август = TRUE",

// 16. Список всех маршрутов с низкопольным транспортом

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип пола].[Тип пола] FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]) INNER JOIN [Тип пола] ON [Транспортные средства].[Тип пола] = [Тип пола].[Код типа пола] WHERE Маршруты.Работает = TRUE AND [Тип пола].[Тип пола] = 'Низкопольный' ORDER BY Маршруты.[Код маршрута]",

// 17. Список всех маршрутов с транспортом на газу

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип топлива].[Тип топлива] FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]) INNER JOIN [Тип топлива] ON [Транспортные средства].[Тип топлива] = [Тип топлива].[Код типа топлива] WHERE Маршруты.Работает = TRUE AND [Тип топлива].[Тип топлива] = 'Газ' ORDER BY Маршруты.[Код маршрута]",

// 18. Список всех маршрутов с транспортом на электричестве

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип топлива].[Тип топлива] FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС]) INNER JOIN [Тип топлива] ON [Транспортные средства].[Тип топлива] = [Тип топлива].[Код типа топлива] WHERE Маршруты.Работает = TRUE AND [Тип топлива].[Тип топлива] = 'Электричество' ORDER BY Маршруты.[Код маршрута]",

// 19. Список всех маршрутов с кондиционерами

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Транспортные средства].Кондиционер FROM (Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС] WHERE Маршруты.Работает = TRUE AND [Транспортные средства].Кондиционер = TRUE",

// 20. Список всех маршрутов с USB-зарядками

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Транспортные средства].[USB-зарядки] FROM (Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС] WHERE Маршруты.Работает = TRUE AND [Транспортные средства].[USB-зарядки] = TRUE",

// 21. Список всех маршрутов с пандусами

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Транспортные средства].Пандус FROM (Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС] WHERE Маршруты.Работает = TRUE AND [Транспортные средства].Пандус = TRUE ORDER BY Маршруты.[Код маршрута]",

// 22. Список всех маршрутов с Wi-Fi

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Транспортные средства].[Wi-Fi] FROM (Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Транспортные средства] ON Маршруты.[Транспортные средства] = [Транспортные средства].[Код ТС] WHERE Маршруты.Работает = TRUE AND [Транспортные средства].[Wi-Fi] = TRUE",

// 23. Список всех маршрутов с валидаторами

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Места оплаты].Валидатор FROM (Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Места оплаты] ON Маршруты.[Места оплаты] = [Места оплаты].[Код места оплаты] WHERE Маршруты.Работает = TRUE AND [Места оплаты].Валидатор = TRUE",

// 24. Список всех маршрутов с кондукторами

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Места оплаты].Кондуктор FROM (Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [Места оплаты] ON Маршруты.[Места оплаты] = [Места оплаты].[Код места оплаты] WHERE Маршруты.Работает = TRUE AND [Места оплаты].Кондуктор = TRUE",

// 25. Список всех маршрутов, работающих по воскресеньям

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Switch([Время работы].[График по месяцам]=1,\"Ежегодно\", [Время работы].[График по месяцам]=2,\"В летний период\", [Время работы].[График по месяцам]=3,\"В течение учебного года\", [Время работы].[График по месяцам]=4,\"В зимний период\") AS [График по месяцам], Switch([Время работы].[График по дням недели]=1,\"Ежедневно\", [Время работы].[График по дням недели]=2,\"С понедельника по субботу\", [Время работы].[График по дням недели]=3,\"По будням\", [Время работы].[График по дням недели]=4,\"По выходным\", [Время работы].[График по дням недели]=5,\"По субботам\", [Время работы].[График по дням недели]=6,\"По воскресеньям\") AS [График по дням недели], Format([Время работы].[Время начала работы],\"hh:nn\") AS [Время начала], Format([Время работы].[Время окончания работы],\"hh:nn\") AS [Время окончания] FROM (([Время работы] INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN [График по дням недели] ON [Время работы].[График по дням недели] = [График по дням недели].[Код графика] WHERE Маршруты.Работает = TRUE AND [График по дням недели].Воскресенье = TRUE",

// 26. Список всех маршрутов, работающих после 22:00

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Switch([Время работы].[График по месяцам]=1, \"Ежегодно\", [Время работы].[График по месяцам]=2, \"В летний период\", [Время работы].[График по месяцам]=3, \"В течение учебного года\", [Время работы].[График по месяцам]=4, \"В зимний период\") AS [График по месяцам], Switch([Время работы].[График по дням недели]=1, \"Ежедневно\", [Время работы].[График по дням недели]=2, \"С понедельника по субботу\", [Время работы].[График по дням недели]=3, \"По будням\", [Время работы].[График по дням недели]=4, \"По выходным\", [Время работы].[График по дням недели]=5, \"По субботам\", [Время работы].[График по дням недели]=6, \"По воскресеньям\") AS [График по дням недели], Format([Время работы].[Время начала работы], \"hh:nn\") AS [Время начала], Format([Время работы].[Время окончания работы], \"hh:nn\") AS [Время окончания] FROM ([Время работы] INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = TRUE AND [Время работы].[Время окончания работы] > #22:00:00# OR [Время работы].[Время окончания работы] < #01:00:00#",

// 27. Список всех маршрутов, работающих до 06:00

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Switch([Время работы].[График по месяцам]=1, \"Ежегодно\", [Время работы].[График по месяцам]=2, \"В летний период\", [Время работы].[График по месяцам]=3, \"В течение учебного года\", [Время работы].[График по месяцам]=4, \"В зимний период\") AS [График по месяцам], Switch([Время работы].[График по дням недели]=1, \"Ежедневно\", [Время работы].[График по дням недели]=2, \"С понедельника по субботу\", [Время работы].[График по дням недели]=3, \"По будням\", [Время работы].[График по дням недели]=4, \"По выходным\", [Время работы].[График по дням недели]=5, \"По субботам\", [Время работы].[График по дням недели]=6, \"По воскресеньям\") AS [График по дням недели], Format([Время работы].[Время начала работы], \"hh:nn\") AS [Время начала], Format([Время работы].[Время окончания работы], \"hh:nn\") AS [Время окончания] FROM ([Время работы] INNER JOIN Маршруты ON [Время работы].Маршрут = Маршруты.[Код маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = TRUE AND [Время работы].[Время начала работы] < #06:00:00#",

// 28. Список неработающих маршрутов

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], [Тип маршрута].[Тип маршрута], Начальная.Остановка AS Начальная, Конечная.Остановка AS Конечная FROM (((Маршруты INNER JOIN Остановки AS Начальная ON Маршруты.Начальная = Начальная.[Код остановки]) INNER JOIN Остановки AS Конечная ON Маршруты.Конечная = Конечная.[Код остановки]) INNER JOIN [Тип маршрута] ON Маршруты.[Тип маршрута] = [Тип маршрута].[Код типа маршрута]) INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта] WHERE Маршруты.Работает = FALSE",

// 29. Список всех маршрутов, где доступна бесплатная пересадка

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Тарифы.[Бесплатная пересадка] FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN Тарифы ON Маршруты.[Тариф] = Тарифы.[Код тарифа]) WHERE Маршруты.Работает = TRUE AND Тарифы.[Бесплатная пересадка] = TRUE",

// 30. Список всех маршрутов, где доступны проездные

"SELECT Маршруты.[Номер маршрута], [Тип транспорта].[Тип транспорта], Тарифы.Проездные FROM ((Маршруты INNER JOIN [Тип транспорта] ON Маршруты.[Тип транспорта] = [Тип транспорта].[Код транспорта]) INNER JOIN Тарифы ON Маршруты.[Тариф] = Тарифы.[Код тарифа]) WHERE Маршруты.Работает = TRUE AND Тарифы.Проездные = TRUE"

};

}

}

# Приложение 3

Руководство пользователя

1. Общие сведения о программе

Файл программы называется «CheTransApp.exe» (рис. П3.1). Программа позволяет работать с базой данных по общественному транспорту г. Череповца: просматривать, добавлять, редактировать и удалять информацию.

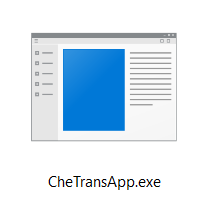


Рис. П3.1. Файл программы

1. Описание установки

Для работы программы файл с базой данных «CheTrans.accdb» и файл программы «CheTransApp.exe» необходимо скопировать с носителя информации (USB-флешки или CD-диска) на жесткий диск компьютера в одну папку (рис. П3.2).

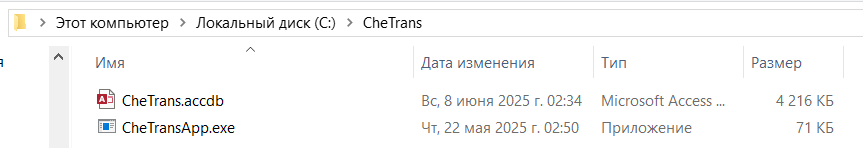


Рис. П3.2. База данных и программа после переноса на жесткий диск

Установка программы не требуется. Сторонних программ для работы также не требуется.

1. Описание запуска

Для запуска программы необходимо дважды нажать на файл «CheTransApp.exe», либо на его ярлык.

1. Инструкции по работе

В программе реализован интерфейс для двух видов пользователей: пассажиров и администраторов. При запуске программы появляется диалоговое окно, предлагающее выбрать пользователя (рис. П3.3).

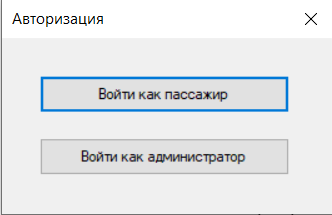


Рис. П3.3. Окно выбора пользователя

В случае выбора пользователя «Пассажир» сразу открывается основное окно программы (рис. П3.4). Элементы интерфейса подписаны на рисунке.

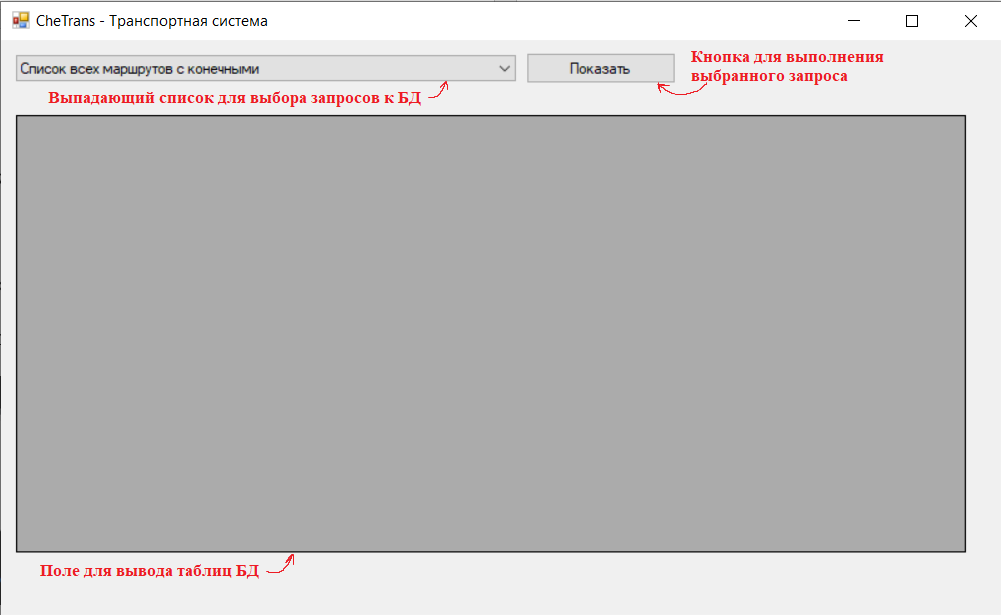


Рис П3.4. Элементы интерфейса основного окна программы пользователя «Пассажир»

Из выпадающего списка необходимо выбрать любой из 30 доступных запросов и затем нажать на кнопку «Показать» (рис. П3.5). Результат запроса будет выведен в поле для вывода таблиц (рис. П3.6).

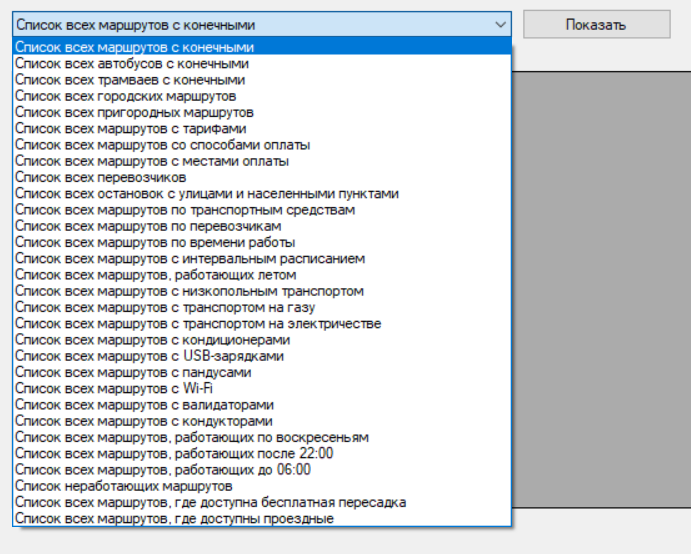


Рис. П3.5. Выпадающий список с запросами

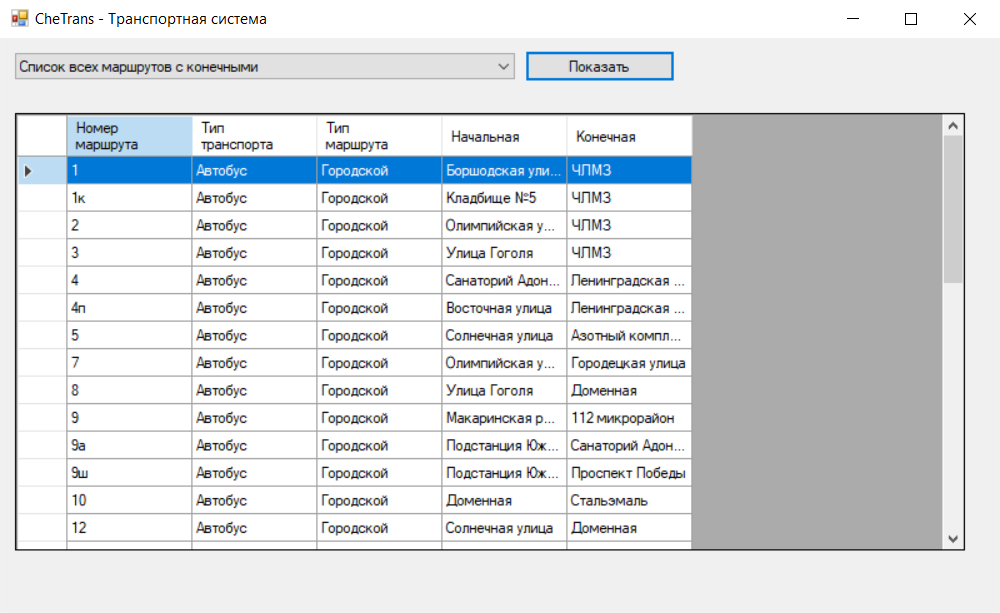


Рис. П3.6. Вывод результата запроса

В режиме пользователя «Пассажир» прямой доступ к базе данных и ее редактирование недоступно.

При выборе пользователя «Администратор» (см. рис. П3.3) программа выведет еще одно диалоговое окно, в котором запросит ввести пароль (рис. П3.7). Это необходимо для защиты базы данных от несанкционированного доступа. По умолчанию установлен пароль «1234». Для защиты от утечки пароля при его вводе текст заменяется на звездочки (рис. П3.8). При вводе неверного пароля программа предложит ввести его заново.

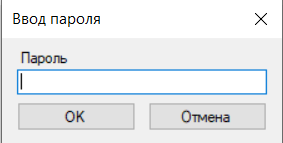


Рис. П3.7. Окно ввода пароля

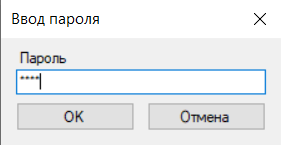


Рис. П3.8. Скрытие пароля

В случае ввода верного пароля открывается основное окно программы (рис. П3.9). Элементы интерфейса подписаны на рисунке.

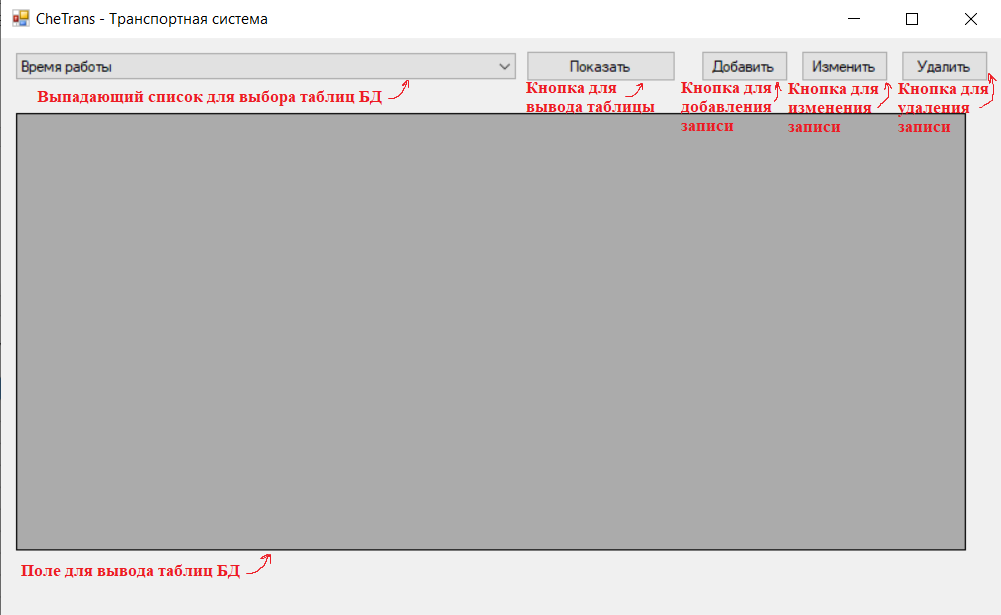


Рис П3.9. Элементы интерфейса основного окна программы пользователя «Администратор»

Из выпадающего списка необходимо выбрать любую из 22 таблиц БД и затем нажать на кнопку «Показать» (рис. П3.10). Таблица будет выведена в поле для вывода таблиц (рис. П3.11).

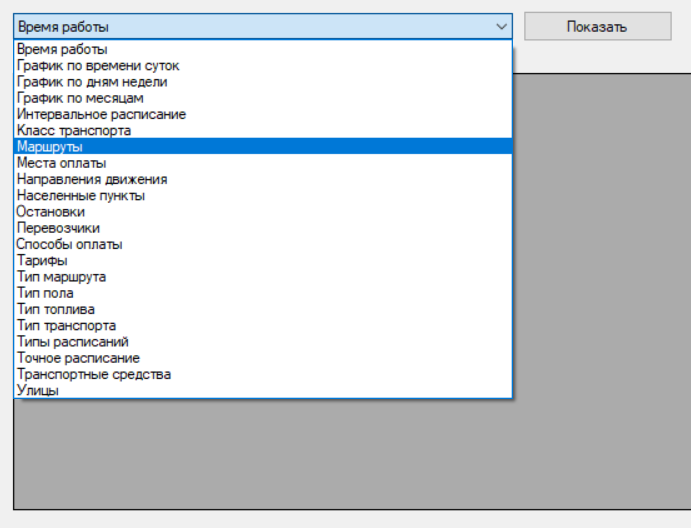


Рис. П3.10. Выпадающий список с таблицами

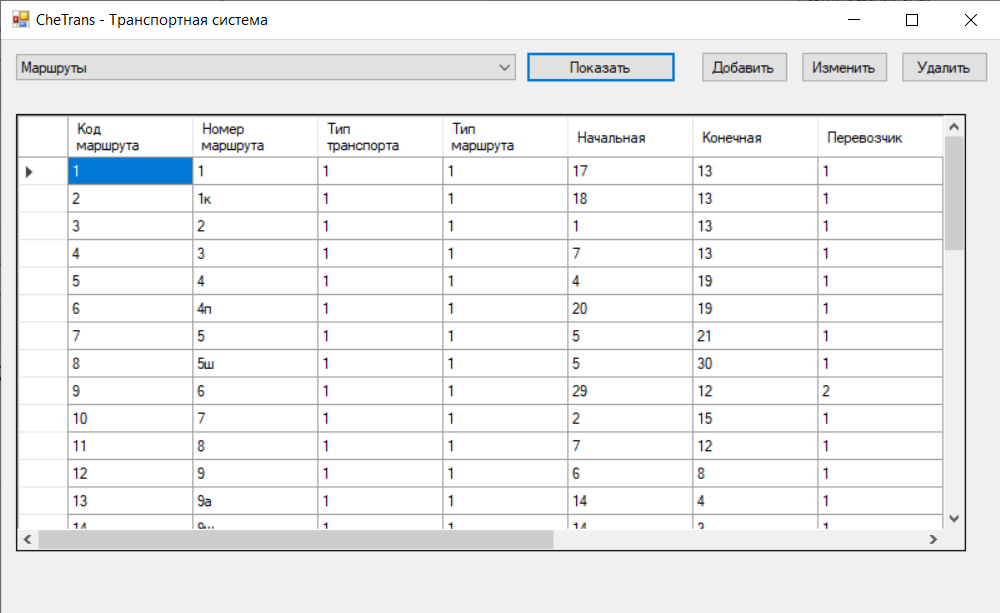


Рис. П3.11. Вывод таблицы

У администратора есть возможность при необходимости редактировать записи таблицы вручную, самостоятельно прописывая все значения ключей (рис. П3.12). Но для упрощения работы также был реализован удобный интерфейс для добавления, изменения и удаления записей таблиц.

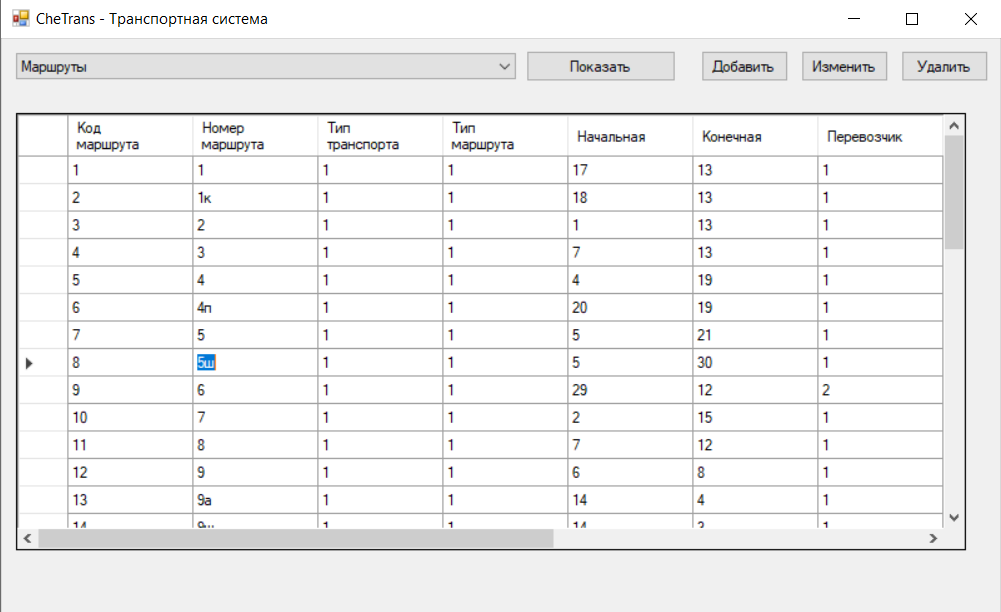


Рис П3.12. Редактирование таблицы вручную

Для добавления новой записи необходимо выбрать таблицу, в которую ее нужно добавить (см. рис. П3.10) и затем нажать на кнопку «Добавить» (см. рис П3.9). Далее программа выведет окно, в котором предложит заполнить все значения полей (рис. П3.13). После ввода значений необходимо нажать на кнопку «Сохранить». Чтобы не вносить изменения, нужно нажать на кнопку «Отмена».

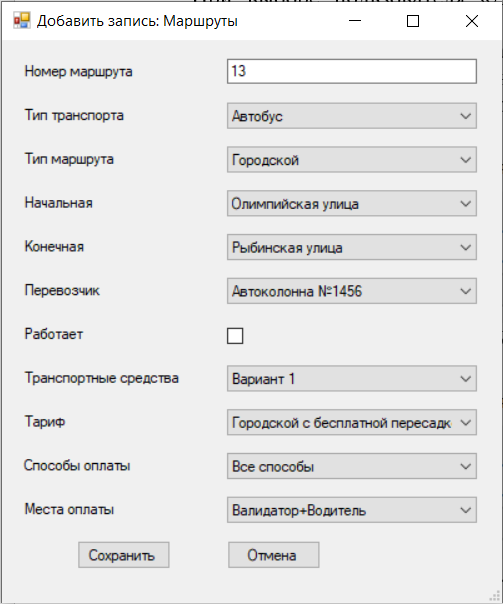


Рис. П3.13. Окно добавления новой записи

Для заполнения полей, являющихся внешними ключами, вместо индексов используются выпадающие списки (рис. П3.14). Для остальных полей в зависимости от типа данных используются текстовые поля ввода, поля выбора дат и времени, а также флаги выбора.

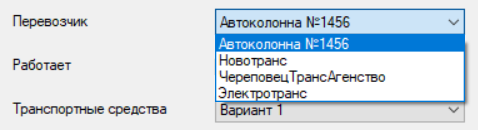


Рис. П3.14. Выпадающий список с допустимыми вариантами значений

Для внесения изменений в существующую запись необходимо в открытой таблице выделить одну из записей (рис. П3.15) и затем нажать на кнопку «Изменить» (см. рис П3.9). После этого откроется окно, аналогичное окну при добавлении записей (рис. П3.16), в котором необходимо внести изменения в уже заполненные поля.

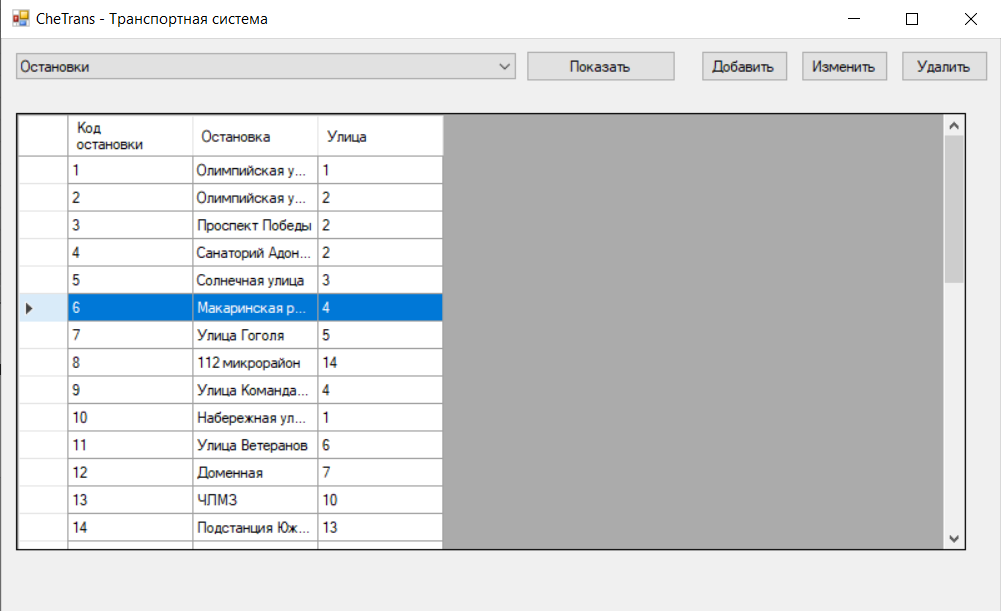


Рис П3.15. Выделение записи, в которую нужно внести изменения

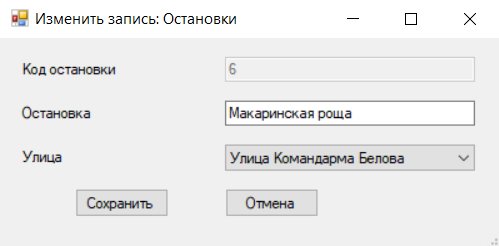


Рис П3.16. Окно изменения записи

Для удаления записи необходимо в открытой таблице выделить одну из записей (см. рис. П3.15) и затем нажать на кнопку «Удалить» (см. рис. П3.9). Во избежание случайных удалений записей программа выведет диалоговое окно, в котором попросит подтвердить удаление (рис. П3.17). При нажатии на кнопку «Да» запись будет удалена. При нажатии на кнопку «Нет» запись останется в таблице.

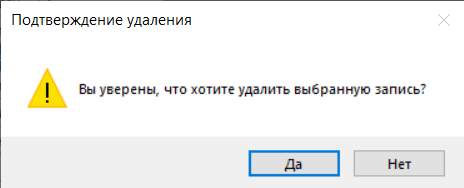


Рис. П3.17. Подтверждение удаления записи

Для выхода из программы необходимо нажать на иконку в виде крестика в правом верхнем углу (рис. П3.18).

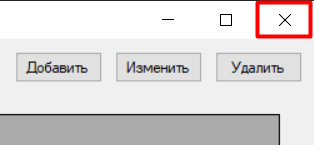


Рис. П3.18. Кнопка для закрытия программы

1. Сообщения пользователю

При попытке входа за пользователя «Администратор» в случае ввода неверного пароля программа выведет сообщение «Неверный пароль!» (рис. П3.19).

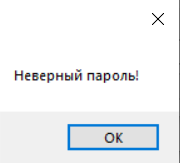


Рис. П3.19. Сообщение «Неверный пароль!»

При попытке удаления записи из таблицы программа выведет сообщение «Вы уверены, что хотите удалить выбранную запись?» (рис. П3.20). Для удаления записи необходимо нажать на кнопку «Да». Чтобы оставить запись, необходимо нажать на кнопку «Нет».

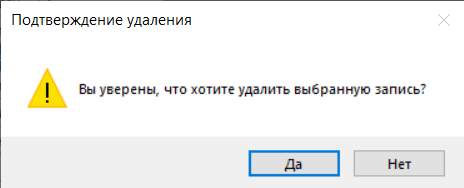


Рис. П3.20. Сообщение «Вы уверены, что хотите удалить выбранную запись?»