Содержание

Введение

На сегодняшний день современное общество всё больше ведёт своё развитие к тому, чтобы как можно сильнее упростить повседневную жизнь, собственную работу. Одним из способов является внедрение компьютерных технологий в любую деятельность человека.

Потоки информации, циркулирующие в мире, который нас окружает, огромны. Во времени они имеют тенденцию к увеличению. Поэтому в любой организации, как большой, так и маленькой, возникает проблема управления данными, которое обеспечило бы наиболее эффективную работу.

Рассуждая о современных требованиях, предъявленных к качеству работы торговых предприятий, отмечается, что эффективная работа полностью зависит от оснащения оборудованием компании информационных средств на системе автоматизированного учета склада.

Компьютерный учет услуг полностью отличается от обычного. Компьютерные программы упрощают учет, сокращая время, которое требуется на накопленные данные и оформление документов для анализа деятельности в сфере услуг.

Следовательно, при применении компьютерных программ увеличивается скорость расчетов, дает возможность качественному улучшенному построению схем торговли.

Результаты выполнения торговых операций записываются в надлежащих журналах. Автоматизация данных процессов позволит сохранить информацию в базе, в которую вводится данная информация с помощью удобного интерфейса программы. Основное преимущество автоматизации - это экономия объема памяти, снижение затрат на операции обновления.

Информационная подсистема автоматизирует и ведет учёт услуг и расходников со склада на предприятии. Осуществление последней задачи будет сопутствовать более качественному обслуживанию и повысит точность учёта.

Целью курсового проекта выявляется проектирование и разработка информационной системы для фотоцентра. Данная программа нужна для произведения учета услуг и подсчёт расходников для услуг.

Глава 1. ПОСТАНОВКА ЗАДАЧИ

1.1 Назначение программного продукта, изучение предметной области

Объектом предметной области является магазин фотоцентр. Основное направление деятельности – оказание фотоуслуг. Главными функциями данного магазина являются:

- создание фотоколлажей;
- печать фото на холсте;
- печать фото;
- фотомонтаж;
- обработка фото;
- создание фото.

В настоящее время большая часть информационных процессов, протекающих в фотоцентре, реализуется вручную. Для хранения информации применяются бумажные носители. Основными целями создаваемой автоматизированной информационной системы являются:

- повышение эффективности деятельности работы магазина;
- увеличение производительности труда персонала;
- повышение оперативности и точности циркулирующей в магазине информации;
- снижение затрат;
- сокращение времени выполнения операций.

1.2 Требования к программному продукту

Программа «Фотоцентр» имеет много пользовательский режим. Все пользователи программы могут быть отнесены к типам: отдел продаж, склад и администратор. В зависимости от выбранного пользователя могут быть доступны различные варианты интерфейса.

Приложение должно представлять собой список задач, с которыми работает пользователь.

Пользователю должны быть доступны следующие функции:

- Занесение информации в базу данных;
- Редактирование и удаление имеющейся в базе данных информации;
- Загрузка в программу отчетов со склада;
- Просмотр и редактирование журналов;

В программе необходимо предусмотреть возможность корректировки настроек системы; наличие встроенной справочной системы.

В программной системе необходимо предусмотреть защиту данных от случайного удаления и изменения. В целях надежности программного обеспечения она должна удовлетворять следующим требованиям:

- разработанная программа должна обладать средствами защиты от ошибочных действий пользователей;
- все ошибки должны отображаться с комментариями или подсказками по их устранению;
- исключить возможность доступа к файлам конфигурации пользователям.

Для повышения надежности необходимо принять следующие меры:

- сконфигурировать аппаратные и программные средства в соответствии с техническими требованиями;
- периодически осуществлять резервное копирование информации;
- регулярно проверять целостность базы данных;
- поддерживать исправность сетевого оборудования.

Необходимы компьютеры с характеристиками не хуже:

Для клиентского приложения:

- процессор Intel Core Duo 1,8 ГГц;
- оперативную память объемом 2Гб;
- HDD 40 Гигабайт;
- операционную систему Windows 7.

Сервер баз данных:

При использовании программного продукта до 50 пользователей базу данных можно установить на один компьютер. Технические характеристики компьютера и операционная система должны соответствовать требованиям Postgresql:

- процессор (Intel/AMD-совместимый x86/x64) 4 ядра с частотой 2 ГГц;
- оперативную память объемом 12Гб;
- HDD 1;
- операционную систему Windows 7 или Linux Ubuntu 16.04.

Для сети:

- связь 3G и выше;
- скорость передачи данных 515 Кбит/сек;
- уровень потерь сетевых пакетов не выше 5%.

1.3 Описание входной информации

Входной информацией для таблицы orders является: Номера сотрудников, клиентов и списков товара, а также дата оформления заказа.

Входной информацией для таблицы client является: ФИО клиентов, а также их номер телефон и почта (необязательно).

Входной информацией для таблицы passport является: Почти что все данные паспорта, серия, номер, кем выдан, дата выдачи, дата рождения

Входной информацией для таблицы services является: Название услуг и их цена

Входной информацией для таблицы rashod является: Название расходников и их количество на складе

Входной информацией для таблицы adress является: данные улицы, области, города, а также номер дома, номер квартиры (необязательно) и индекс

Входной информацией для таблицы city является: Название города

Входной информацией для таблицы oblast является: Название области

Входной информацией для таблицы street является: Название улицы

Входной информацией для таблицы employs является: Номер паспорта, ФИО сотрудника, их снилс и ИНН

Входной информацией для таблицы ord_serv является: Номер услуги, Номер расходника и количество

Источником входной информацией является следующий перечень документов: Паспортные данные, данные клиента, трудовой договор,

1.4 Описание выходной информации

Выходная информация представлена в виде журналов и отчетов, выводимых по необходимости, и рассчитана на получение администрацией и сотрудниками.

Проверка правильности выходной информации должна проверяться пользователями.

Сроки выдачи информации - по мере необходимости.

Глава 2. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Проектирование информационных систем всегда начинается с определения цели проекта как решение ряда взаимосвязанных задач, включающих в себя обеспечение на момент запуска системы и в течение всего времени ее эксплуатации:

- требуемой функциональности системы и уровня ее адаптивности к изменяющимся условиям функционирования;
- требуемой пропускной способности системы;
- требуемого времени реакции системы на запрос;
- безотказной работы системы;
- необходимого уровня безопасности;
- простоты эксплуатации и поддержки системы.

Процесс построения и последовательного преобразования ряда согласованных моделей на всех этапах жизненного цикла (ЖЦ) информационной системы. На каждом этапе ЖЦ создаются специфичные для него модели:

- организации,
- требований к ИС,
- проекта ИС,
- требований к приложениям и т.д.

Модели формируются рабочими группами команды проекта, сохраняются и накапливаются в репозитории проекта.

Создание моделей, их контроль, преобразование и предоставление в коллективное пользование осуществляется с использованием специальных программных инструментов - CASE-средств.

Процесс создания ИС делится на ряд этапов, ограниченных некоторыми временными рамками и заканчивающихся выпуском конкретного продукта (моделей, программных продуктов, документации и пр.).

Обычно выделяют следующие этапы создания ИС:

- формирование требований к системе,
- проектирование,
- реализация,
- тестирование,
- ввод в действие,
- эксплуатация,
- сопровождение

Одновременно идёт оформление документации.

2.1 Концептуальное проектирование реляционной базы данных

Концептуальное (инфологическое) проектирование - построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных. Термины "семантическая модель", "концептуальная модель" и "инфологическая модель" являются синонимами. Кроме того, в этом контексте равноправно могут использоваться слова "модель базы данных" и "модель предметной области" (например, "концептуальная модель базы данных" и "концептуальная модель предметной области"), поскольку такая модель является как образом реальности, так и образом проектируемой базы данных для этой реальности.

Конкретный вид и содержание концептуальной модели базы данных определяется выбранным для этого формальным аппаратом.

Чаще всего концептуальная модель базы данных включает в себя:

-описание информационных объектов или понятий предметной области и связей между ними;

-описание ограничений целостности, т.е. требований к допустимым значениям данных и к связям между ними.

Концептуальная модель базы данных представлена в приложении А.

Также в ИС можно выделить:

Сущности – представляют собой тип объектов, которые должны храниться в базе данных. Каждая таблица в базе данных должна представлять одну сущность. Как правило, сущности соответствуют объектам из реального мира. У каждой сущности определяют набор атрибутов.

В моей информационной системе можно выделить 12 сущностей:

1. client
2. orders
3. ord_serv
4. services
5. employs
6. passport
7. address
8. street
9. oblast
10. city
11. serv_rashod
12. reshod

Подписи связей – они нужны по большей части для удобства, для понимания как таблицы взаимосвязаны между собой

Кардинальность – это отношение строк одного столбца к другой, примером такого отношения является

- 1-1(одна строка в таблице А относится к одной строке в таблице В)
- 1-М (одна строка в таблице А относится ко многим строкам в таблице В)
- М-М (многие строки в таблице А относятся ко многим строкам в таблице В)

Матрица связей – показывает связи между таблицами (см. Рисунок б.1)

	клиент	заказ	услуги	улсуги_заказы	работники	паспорт	расход_услуг	расход	адрес	улица	город	область
клиент		имеет										
заказ	включает			включает	обрабатываются							
услуги					выполняются							
услуги_заказы		входят	включает									
работники		обрабатывают	выполняют			имеют			имеют			
паспорт					имеется							
расход_услуг			учитывает					сохраняет				
расход							сохраняется					
адрес					имеется					включает	включает	включает
улица									включена			
город									включён			
область									включена			

Рисунок б.1 – Матрица связей

2.2 Логическое проектирование реляционной базы данных

Логическая модель данных является визуальным графическим представлением структур данных, их атрибутов и связей. Логическая модель представляет данные таким образом, чтобы они легко воспринимались бизнес-пользователями. Проектирование логической модели должно быть свободно от требований платформы и языка реализации или способа дальнейшего использования данных.

При разработке используются требования к данным и результаты анализа для формирования логической модели данных. Логическую модель приводят к третьей нормальной форме, и проверяет ее на соответствие модели процессов.

Нормализация — это процесс организации данных в базе данных.

Первая нормальная форма(1NF)

- Данные являются элементарными (все атрибуты должны быть с одним значением).
- Записи в столбце должны быть одинакового типа.
- В таблице не должно быть повторяющихся строк, что значит, что таблица должна иметь группу столбцов, которая уникально определяет строку.

Вторая нормальная форма(2NF)

- Должна быть 1-ой NF.
- Первичный ключ с одним столбцом.

Третья нормальная форма(3NF)

- Должна быть 2-ой NF.
- Не имеет транзитивных функциональных зависимостей.

Документ сопоставления таблиц

Имя таблицы	Краткое имя таблицы	
orders	ord	
KeyType	Optionality	Column Name
fk	*	id_emp
fk	*	id_k
pk	*	id_ord
	◦	oplachen
	*	data_ord
fk	*	id_os

Рисунок а.1 – Таблица orders

Имя таблицы	Краткое имя таблицы	
services	serv	
KeyType	Optionality	Column Name
pk	*	id_serv
fk	*	id_emp
	*	name_serv
	*	price

Рисунок а.2 – Таблица services

Имя таблицы	Краткое имя таблицы	
rashod	ras	
KeyType	Optionalit	Column Name
pk	*	id_ras
	*	amount_ras
	*	name_rash

Рисунок а.3 – Таблица rashod

Имя таблицы	Краткое имя таблицы	
employs	emp	
KeyType	Optionalit	Column Name
pk	*	id_pass
fk,uk	*	number_pass
	*	seria
	*	kem_vidan
	*	date_of_issue
uk	*	birthday
uk	*	kod_pod
fk	*	Id_adr

Рисунок а.4 – Таблица employs

Имя таблицы	Краткое имя таблицы	
adress	adr	
KeyType	Optionalit	Column Name
pk	*	id_adr
	*	dom
	*	index
	◦	kv
	*	id_s
fl	*	id_obl
fl	*	id_city
fk	*	Id_adr

Рисунок а.5 – Таблица adress

Имя таблицы	Краткое имя таблицы	
street	st	
KeyType	Optionalit	Column Name
pk	*	id_s
	*	name_street

Рисунок а.6 - Таблица street

Имя таблицы	Краткое имя таблицы	
oblast	obl	
KeyType	Optionalit	Column Name
pk	*	id_obl
	*	name_obl

Рисунок а.7 – Таблица oblast

Имя таблицы	Краткое имя таблицы	
city	ct	
KeyType	Optionalit	Column Name
pk	*	id_city
	*	name_city

Рисунок а.8 – Таблица city

Имя таблицы	Краткое имя таблицы	
passport	pas	
KeyType	Optionalit	Column Name
pk	*	id_pass
uk	*	number_pass
	*	seria
	*	kem_vidan
	*	date_of_issue
	*	birthday
	*	kod_pod

Рисунок а.9 – Таблица passport

Имя таблицы	Краткое имя таблицы		
ord_serv	ord_s		
KeyType	Optionalit	Column Name	DataType
pk	*	id_os	integer
fk	*	id_serv	integer
fk	*	id_ras	integer
	*	amount_s	integer

Рисунок а.10 – Таблица ord_serv

Логическая модель базу данных представлена в приложении Б.

2.3 Выбор СУБД. Физическое проектирование реляционной БД

При выборе СУБД необходимо учитывать множество аспектов, каждый из которых накладывает определенные ограничения на выбор продукта. Это не только технические показатели, но и экономические соображения, учет рынка, а также определенные культурологические аспекты, связанные с базой данных.

Для реализации базы данных была выбрана СУБД PostgreSQL.

PostgreSQL - это свободно распространяемая объектно-реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых СУБД в мире и являющаяся реальной альтернативой коммерческим базам данных. PostgreSQL произносится как postgresSQL (можно скачать mp3 файл postgresql. Из преимуществ стоит отметить простоту использования, гибкость, низкую стоимость владения (относительно платных СУБД), а также масштабируемость и производительность.

Физическая модель – логическая модель базы данных, выраженная в терминах языка описания данных конкретной СУБД.

Физическая модель базы данных содержит все детали, необходимые конкретной СУБД для создания базы: наименования таблиц и столбцов, типы данных, определения первичных и внешних ключей и т.п.

Физическая модель строится на основе логической с учетом ограничений, накладываемых возможностями выбранной СУБД.

Таблицы физической модели

Имя таблицы	Краткое имя таблицы			
orders	ord			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_emp	integer	
fk	*	id_k	integer	
fk	*	id_ord	integer	
	◦	oplachen	boolean	
	*	data_ord	date	
fk	*	id_os	integer	

Рисунок с.1 – Таблица orders

Имя таблицы	Краткое имя таблицы			
ord_serv	ord_s			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_os	integer	
fk	*	id_serv	integer	
fk	*	id_ras	integer	
	*	amount_s	integer	

Рисунок с.2 – Таблица ord_serv

Имя таблицы	Краткое имя таблицы			
services	serv			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_serv	integer	
fk	*	id_emp	integer	
	*	name_serv	varchar	60
	*	price	decimal	5,2

Рисунок с.3 – Таблица services

Имя таблицы	Краткое имя таблицы			
rashod	ras			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_ras	integer	
	*	amount_ras	integer	
	*	name_rash	varchar	100

Рисунок с.4 – Таблица rashod

Имя таблицы	Краткое имя таблицы			
employs	emp			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_emp	integer	
fk_uk	*	id_pass	integer	
	*	name_emp	varchar	60
	*	sur_name_emp	varchar	60
	*	otchest_emp	varchar	60
uk	*	INN	char	12
uk	*	snils	char	11
fk	*	Id_adr	integer	

Рисунок с.5 – Таблица employs

Имя таблицы	Краткое имя таблицы			
adress	adr			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_adr	integer	
	*	dom	varchar	20
	◦	kv	integer	
	*	index	char	6
fk	*	id_s	integer	
fk	*	id_obl	integer	
fk	*	id_city	integer	

Рисунок с.6 – Таблица adress

Имя таблицы	Краткое имя таблицы			
street	st			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_s	integer	
	*	name_street	varchar	40

Рисунок с.7 – Таблица street

Имя таблицы	Краткое имя таблицы			
passport	pas			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_obl	integer	
	*	name_obl	varchar	40

Рисунок с.8 – Таблица passport

Имя таблицы	Краткое имя таблицы			
city	ct			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_city	integer	
	*	name_city	varchar	60

Рисунок с.9 – Таблица city

Имя таблицы	Краткое имя таблицы			
passport	pas			
KeyType	Optionalit	Column Name	DataType	Size
pk	*	id_pass	integer	
	*	seria	char	4
	*	number_pas	char	6
	*	kem_vidan	varchar	60
	*	date_of_issue	date	
	*	birthday	date	
	*	kod_pod	char	

Рисунок с.10 – Таблица passport

Физическая модель базы данных представлена в приложении В.

2.4Проектирование функционала программного продукта

Диаграмма вариантов использования (сценариев поведения, прецедентов) является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Данная диаграмма состоит из актеров, вариантов использования и отношений между ними. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества актеров, взаимодействующих с системой с помощью так называемых вариантов использования. При этом актером называется любой объект, субъект или система, взаимодействующая с моделируемой системой извне. В свою очередь вариант использования – это спецификация сервисов (функций), которые система предоставляет актеру. Другими словами, каждый вариант использования определяет некоторый набор действий, совершаемых системой при взаимодействии с актером. При этом в модели никак не отражается то, каким образом будет реализован этот набор действий.

Администратор – имеет все права бд. Он будет следить за всем состоянием информационной системы

Работник – Он будет обрабатывать заказы и также выполнять услуги, сможет вносить, удалять или изменять данные в таблицах orders и services

Диаграмма вариантов использования представлена в приложении Г.

Глава 3. РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА

3.1 Выбор средств разработки программного продукта. Обоснование выбора

При планировании программного проекта имеется огромный выбор языков программирования, в лабиринтах которых легко заблудиться. Не существует какого-то одного языка, который является наилучшим выбором. Можно отдать предпочтение определенным факторам, таким как производительность и безопасность корпоративных приложений, по сравнению с другими факторами, такими как количество строк кода.

В качестве языка программирования для реализации данного проекта был выбран C# - объектно- и компонентно-ориентированный язык программирования. C# подходит для создания и применения программных компонентов. С момента создания язык C# обогатился функциями для поддержки новых рабочих нагрузок и современными рекомендациями по разработке ПО. В C# особое внимание уделяется управлению версиями для обеспечения совместимости программ и библиотек при их изменении.

Инструментарий C# позволяет решать широкий круг задач, язык действительно очень мощный и универсальный. На нем разрабатывают:

–приложения для WEB;

–Различные игровые программы.

–Приложения платформ Андроид или iOS.

–Программы для Windows.

В языке принята общая система работы с типами, начиная от примитивов и заканчивая сложными, в том числе, пользовательскими наборами. Применяется единый набор операций для обработки и хранения значений типизации. Также можно использовать ссылочные типы пользователя, что позволит динамически выделить память под объект или хранить упрощенную структуру в сети.

Для языка программирования C# имеется множество систем программирования, позволяющих создавать программы, работающие в среде DOS, Windows и др.

Для реализации приложения была выбрана среда разработки MicrosoftVisualStudio. VisualStudio является одним из самых мощных и распространённых инструментов разработки с 20-летней историй. Среда разработки представляет так много инструментов для работы с кодом, что в этих инструментах легко потеряться, упустив из виду какую-нибудь важную фишку. Не всегда понятно, какие инструменты из огромного набора реально полезны в повседневной жизни разработчика, а какие используются редко или добавлены в маркетинговых целях.

3.2 Реализация базы данных информационной системы

Создание базы данных с именем photocentr выполняется командой в соответствии с рисунком 1.

```
postgres=# create database photocentr;
CREATE DATABASE
```

Рисунок 1 - Создание БД

Созданную базу необходимо проверить командой (см. Рисунок 2).

```
                                  Список баз данных
    Имя     | Владелец | Кодировка |     LC_COLLATE     |     LC_CTYPE      |  Права доступа
------------+----------+-----------+--------------------+--------------------+-----------------------
 ads        | student  | UTF8      | Russian_Russia.1251 | Russian_Russia.1251 |
 avtoserv   | postgres | UTF8      | Russian_Russia.1251 | Russian_Russia.1251 |
 avtoserves | postgres | UTF8      | Russian_Russia.1251 | Russian_Russia.1251 |
 db10       | usr      | UTF8      | Russian_Russia.1251 | Russian_Russia.1251 |
 db5        | postgres | UTF8      | Russian_Russia.1251 | Russian_Russia.1251 |
 db6        | postgres | UTF8      | Russian_Russia.1251 | Russian_Russia.1251 |
 helo       | postgres | UTF8      | Russian_Russia.1251 | Russian_Russia.1251 |
 photocentr | postgres | UTF8      | Russian_Russia.1251 | Russian_Russia.1251 |
```

Рисунок 2 - Проверка БД

Выполнить подключение к созданной БД photocenter необходимо командой \c (см. Рисунок 3).

```
postgres=# \c photocentr
Вы подключены к базе данных "photocentr" как пользователь "postgres".
```

Рисунок 3 – Подключение к БД

Создание таблицы Passport (Паспорт) (см. Рисунок 4), идентификатор «id_pass» был задан с типом integer, а поле «number» заданы уникальными, так как номер паспорта у всех уникальны.

```
photocentr=# create table passport (id_pass integer primary key, seria char(4) not null, number_pas char(6) not null uni
que,date_of_issue date not null, kem_vidan varchar(60) not null, birthday date not null, kod_pod char(6) not null);
```

Рисунок 4 – Создание таблицы passport

Создание таблицы employs (Сотрудники) (см. Рисунок 5).

```
photocentr=# create table employs (id_emp integer primary key, name_emp varchar(60) not null, sur_name_emp varchar(60) n
ot null, otchest_emp  varchar(60) not null, INN char(12) not null unique,snils char(11) not null unique, id_pass integer
 references passport(id_pass) not null unisque, id_adr integer references adress(id_adr) not null, zanyat bool);
```

Рисунок 5 - Создание таблицы employs

Создание таблицы client (Клиенты) (см. Рисунок 6).

```
photocentr=# create table client(id_k integer primary key, surname varchar(60) not null, firstname varchar(60) not null,
 mob_p char(11) not null, otchest varchar(60) not null, e_mail varchar(100));
```

Рисунок 6 - Создание таблицы Customers

Создание таблицы services (Услуги) (см. Рисунок 7).

```
photocentr=# create table services (id_serv integer primary key, id_emp integer references employs(id_emp) not null,name_serv va
rchar(60) not null,price decimal(5,4) not null);
CREATE TABLE
```

Рисунок 7 – Создание таблицы services

Создание таблицы rashod (Остаток) (см. Рисунок 8).

```
photocentr=# create table rashod(id_ras integer primary key, amount_ras integer not null, name_rash varchar(100) not nul
l);
CREATE TABLE
```

Рисунок 8 – Создание таблицы rashod

Создание таблицы orders (Заказы) (см. Рисунок 9).

```
photocentr=# create table orders(id_k integer references client(id_k) not null,id_emp integer references employs(id_emp)
not null, id_ord integer primary key, data_ord date not null, oplachen bool, id_os integer references ord_serv(id_os) no
t null);
```

Рисунок 9 - Создание таблицы Orders

Создание таблицы ord_serv (Услуги в заказе) (см. Рисунок 10).

```
photocentr=# create table ord_serv ( id_os integer primary key, id_serv integer references services(id_serv) not null, i
d_ras integer references  rashod(id_ras) not null, ammount_s integer not null);
```

Рисунок 10 - Создание таблицы ord_serv

Создание таблицы oblast (Область) (см. Рисунок 11).

```
photocentr=# create table oblast(id_obl integer primary key, name_obl varchar(40) unique not null);
CREATE TABLE
```

Рисунок 11 - Создание таблицы oblast

Создание таблицы street (Улица) (см. Рисунок 12).

```
photocentr=# create table street(id_s integer primary key, name_street varchar(40) not null unique);
CREATE TABLE
```

Рисунок 12 - Создание таблицы street

Создание таблицы city (Город) (см. Рисунок 13).

```
photocentr=# create table city(id_city integer primary key, name_city varchar(40) not null unique);
CREATE TABLE
```

Рисунок 13 - Создание таблицы city

Создание таблицы adress (Адрес) (см. Рисунок 14).

```
photocentr=# create table adress(id_adr integer primary key, dom varchar(20) not null, index char(6) not null, kv intege
r, id_s integer references street(id_s) not null,id_city integer references city(id_city) not null,id_obl integer refere
nces oblast(id_obl));
CREATE TABLE
```

Рисунок 14 - Создание таблицы address

Проверяем наличие созданных таблиц (см. Рисунок 15).

```
photocentr=# \d
             Список отношений
 Схема  |     Имя      |   Тип   | Владелец
--------+--------------+---------+----------
 public | adress       | таблица | postgres
 public | city         | таблица | postgres
 public | client       | таблица | postgres
 public | employs      | таблица | postgres
 public | oblast       | таблица | postgres
 public | ord_serv     | таблица | postgres
 public | orders       | таблица | postgres
 public | passport     | таблица | postgres
 public | rashod       | таблица | postgres
 public | serv_rashod  | таблица | postgres
 public | services     | таблица | postgres
 public | street       | таблица | postgres
(12 строк)
```

Рисунок 15- Просмотр всех таблиц

Далее заполним все таблицы.
Таблицу паспортов (см. Рисунок 16).

```
photocentr=# insert into passport values(1,4356,654387,'2000-11-21','Отделом УФМС по Ростовской области','1986-10-22',905621);
INSERT 0 1
photocentr=# insert into passport values(2,4374,651387,'2000-02-04','Отделом УФМС по Ростовской области','1986-01-28',905864);
INSERT 0 1
photocentr=# insert into passport values(3,7674,321387,'1999-06-06','Отделом УФМС по Ростовской области','1987-06-02',543864);
INSERT 0 1
```

Рисунок 16 – Заполнение таблицы

Проверим (см. Рисунок 17).

```
photocentr=# select * from passport;
 id_pass | seria | number_pas | date_of_isse |             kem_vidan              |  birthday  | kod_pod
---------+-------+------------+--------------+------------------------------------+------------+---------
       1 | 4356  | 654387     | 2000-11-21   | Отделом УФМС по Ростовской области | 1986-10-22 | 905621
       2 | 4374  | 651387     | 2000-02-04   | Отделом УФМС по Ростовской области | 1986-01-28 | 905864
       3 | 7674  | 321387     | 1999-06-06   | Отделом УФМС по Ростовской области | 1987-06-02 | 543864
(3 строки)
```

Рисунок 17 – Просмотр таблицы

Проверка уникальности (см. Рисунок 18).

```
photocentr=# insert into passport values(3,7674,321387,'1999-06-06','Отделом УФМС по Ростовской области','1987-06-02',543864);
ОШИБКА:  повторяющееся значение ключа нарушает ограничение уникальности "passport_pkey"
ПОДРОБНОСТИ:  Ключ "(id_pass)=(3)" уже существует.
```

Рисунок 18 – Проверка уникальности

Таблицу город (см. Рисунок 19).

```
photocentr=# insert into city values(1,'Ростов-на-Дону');
INSERT 0 1
photocentr=# insert into city values(2,'Москва');
INSERT 0 1
photocentr=# insert into city values(3,'Краснодар');
INSERT 0 1
```

Рисунок 19 - Заполнение таблицы

Проверим (см. Рисунок 20).

```
photocentr=# select * from city;
 id_city |   name_city
---------+---------------
       1 | Ростов-на-Дону
       2 | Москва
       3 | Краснодар
(3 строки)
```

Рисунок 20 – Просмотр таблицы

Таблицу область (см. Рисунок 21).

```
photocentr=# insert into oblast values(1,'Московакая область');
INSERT 0 1
photocentr=# insert into oblast values(2,'Ростовкая область');
INSERT 0 1
photocentr=# insert into oblast values(3,'Краснодарская область');
INSERT 0 1
```

Рисунок 21 – Заполнение таблицы

Проверим (см. Рисунок 22).

```
photocentr=# select * from oblast;
 id_obl |       name_obl
--------+----------------------
      1 | Московакая область
      2 | Ростовкая область
      3 | Краснодарская область
(3 строки)
```

Рисунок 22 – Просмотр таблицы

Таблицу улица (см. Рисунок 23).

```
photocentr=# insert into street values(1,'ул.Большая садовая');
INSERT 0 1
photocentr=# insert into street values(2,'ул.Стачки');
INSERT 0 1
photocentr=# insert into street values(3,'ул.Московкая');
INSERT 0 1
```

Рисунок 23 – Заполнение таблицы

Проверим (см. Рисунок 24).

```
photocentr=# select * from street;
 id_s |    name_street
------+-------------------
    1 | ул.Большая садовая
    2 | ул.Стачки
    3 | ул.Московкая
(3 строки)
```

Рисунок 24 Просмотр таблицы

Таблицу клиентов (см. Рисунок 25).

```
photocentr=# insert into client values(1,'Ященко','Артём','Николаевич','hellohek232@gmail.com',89924567255);
INSERT 0 1
photocentr=# insert into client (id_k,surname,firstname,otchest,mob_p) values(2,'Молчанов','Бронислав','Мэлсович',897645
67255);
INSERT 0 1
photocentr=# insert into client (id_k,surname,firstname,otchest,mob_p) values(3,'Кудряшов','Наум','Германович',897325672
67);
INSERT 0 1
```

Рисунок 25 - Заполнение таблицы

Проверим (см. Рисунок 26).

```
photocentr=# select * from client;
 id_k | surname  | firstname |  otchest   |        e_mail         |    mob_p
------+----------+-----------+------------+-----------------------+-------------
    1 | Ященко   | Артём     | Николаевич | hellohek232@gmail.com | 89924567255
    2 | Молчанов | Бронислав | Мэлсович   |                       | 89764567255
    3 | Кудряшов | Наум      | Германович |                       | 89732567267
(3 строки)
```

Рисунок 26 - Просмотр таблицы

Таблицу адрес (см. Рисунок 27).

```
photocentr=# insert into adress(id_adr,dom,index,id_s,id_city,id_obl) values(1,'3г',543442,1,1,2);
INSERT 0 1
photocentr=# insert into adress(id_adr,dom,index,id_s,id_city,id_obl) values(2,'24',543442,1,1,2);
INSERT 0 1
photocentr=# insert into adress(id_adr,dom,index,id_s,id_city,id_obl) values(3,'56',543442,1,1,2);
INSERT 0 1
```

Рисунок 27– Заполнение таблицы

Проверим (см. Рисунок 28).

```
photocentr=# select * from adress;
 id_adr | dom | index  | kv | id_s | id_city | id_obl
--------+-----+--------+----+------+---------+--------
      1 | 3г  | 543442 |    |    1 |       1 |      2
      2 | 24  | 543442 |    |    1 |       1 |      2
      3 | 56  | 543442 |    |    1 |       1 |      2
(3 строки)
```

Рисунок 28 - Просмотр таблицы

Таблицу работников (см. Рисунок 29)

```
photocentr=# insert into employs values(1,'Дмитрий','Демидов','Варданович',8524064052,65392407007,1,1);
INSERT 0 1
photocentr=# insert into employs values(2,'Константин','Гришин','Федотович',8524224052,65392787007,2,2);
INSERT 0 1
photocentr=# insert into employs values(3,'Дмитрий','Демидов','Варданович',8524064345,65392407884,3,3);
INSERT 0 1
```

Рисунок 29 – Заполнение таблицы employs

Проверим (см. Рисунок 30).

```
photocentr=# select * from employs;
 id_emp |  name_emp  | sur_name_emp | otchest_emp |     inn      |    snils    | id_pass | id_adr | zanyat
--------+------------+--------------+-------------+--------------+-------------+---------+--------+-------
      1 | Дмитрий    | Демидов      | Варданович  | 8524064052   | 65392407007 |       1 |      1 |
      3 | Дмитрий    | Демидов      | Варданович  | 8524064345   | 65392407884 |       3 |      3 |
      2 | Константин | Гришин       | Федотович   | 8524224052   | 65392787007 |       2 |      2 |
(3 строки)
```

Рисунок 30 – Проверка таблицы employs

Таблицу заказ (см. Рисунок 31)

```
photocentr=# insert into orders values (1,1,1,current_date);
INSERT 0 1
photocentr=# insert into orders values (2,3,2,current_date);
INSERT 0 1
```

Рисунок 31–Заполнение таблицы

Проверим (см. Рисунок 32).

```
photocentr=# select * from orders;
 id_k | id_emp | id_ord |  data_ord  | oplachen
------+--------+--------+------------+----------
    1 |      1 |      1 | 2021-04-28 |
    2 |      3 |      2 | 2021-04-28 |
(2 строки)
```

Рисунок 32 – Проверка таблицы orders

Таблицу расход (см. Рисунок 33)

```
photocentr=# insert into rashod values(1,20,'Глянцевая бумага 10x15');
INSERT 0 1
photocentr=# insert into rashod values(2,30,'Глянцевая бумага 15x20');
INSERT 0 1
photocentr=# insert into rashod values(3,10,'Матовая бумага 15x20');
INSERT 0 1
```

Рисунок 33 – Внесение данных в таблицу rashod

Проверим (см. Рисунок 34)

```
photocentr=# select * from rashod;
 id_ras | amount_ras |       name_rash
--------+------------+-----------------------
      1 |         20 | Глянцевая бумага 10x15
      2 |         30 | Глянцевая бумага 15x20
      3 |         10 | Матовая бумага 15x20
(3 строки)
```

Рисунок 34 – Вывод данных из таблицы расход

Таблицу услуги (см. Рисунок 35)

```
photocentr=# insert into services values(1,'Распечатка фотографии на глянцевой бумаги 10x15',2,9);
INSERT 0 1
photocentr=# insert into services values(2,'Распечатка фотографии на глянцевой бумаги 15x20',2,18);
INSERT 0 1
photocentr=# insert into services values(3,'Распечатка фотографии на матовой бумаги 10x15',2,9);
INSERT 0 1
```

Рисунок 35 – Внесение в данных в таблицу services

Проверим (см. Рисунок 36)

```
photocentr=# select * from services;
 id_serv |                    name_serv                    | id_emp | price
---------+-------------------------------------------------+--------+-------
       1 | Распечатка фотографии на глянцевой бумаги 10x15 |      2 |  9.00
       2 | Распечатка фотографии на глянцевой бумаги 15x20 |      2 | 18.00
       3 | Распечатка фотографии на матовой бумаги 10x15   |      2 |  9.00
(3 строки)
```

Рисунок 36 – Вывод данных из таблицы services

Таблицу ord_serv (см. Рисунок 37)

```
photocentr=# insert into ord_serv values(20,1,1);
INSERT 0 1
photocentr=# insert into ord_serv values(10,2,2);
INSERT 0 1
```

Рисунок 37 – Внесение в данных в таблицу ord_serv

Проверим (см. Рисунок 38)

```
photocentr=# select * from ord_serv;
 ammount_s | id_serv | id_ord
-----------+---------+--------
        20 |       1 |      1
        10 |       2 |      2
(2 строки)
```

Рисунок 38 – Вывод данных из таблицы ord_serv

Создадим функции для ИС

Создание функции для проверки занятости работника (см. Рисунок 39)

```
photocentr=# create or replace function zanyat_t() returns trigger as $$ begin update employs set zanyat = true from ser
vices where(employs.id_emp =new.id_emp); return new; end; $$ language plpgsql;
CREATE FUNCTION
```

Рисунок 39 – создание функции на проверку занятости

Создание функции для проверки занятости работника (для таблицы orders) (см. Рисунок 40)

```
photocentr=# create or replace function zanyat_t2() returns trigger as $$ begin update employs set zanyat = true from odres where (employs.id_emp = new.id_emp); return new; end; $$ language plpgsql;
CREATE FUNCTION
```

Рисунок 40 – Создание функции на проверку занятости (для таблицы orders)

Создание функции для уменьшения количества расходников (см. Рисунок 41)

```
photocentr=# create or replace function um_koll() returns trigger as $$ declare bob integer; begin select into bob rashod.amount_ras from rashod where(rashod.id_ras=new.id_ras); if new.ammount_s<=bob the
n update rashod set amount_ras=amount_ras-new.ammount_s from ord_serv where (rashod.id_ras=new.id_serv); return new; end if; if new.ammount_s>bob then raise exception 'Не хватает'; end if; end; $$ langu
age plpgsql;
CREATE FUNCTION
```

Рисунок 41 – Создание функции на уменьшение количества расходников

Создание функции для удаления (см. Рисунок 42)

```
photocentr=# create or replace function check_del() returns trigger as $$ begin delete from client where check_del= true
; return new; end; $$ language plpgsql;
CREATE FUNCTION
```

Рисунок 42 – Создание функции для удаления

Для работы функций нужно создать триггеры для проверки занятости (см. Рисунок 43,44), уменьшение количества расходников (см. Рисунок 45) и на удаление (см. Рисунок 46).

```
photocentr=# create trigger zanyat_t2 after insert on orders for each row execute procedure zanyat_t2();
CREATE TRIGGER
```

Рисунок 43– Создание триггера

```
photocentr=# create trigger zanyat_t after insert on services for each row execute procedure zanyat_t();
CREATE TRIGGER
```

Рисунок 44 – Создание триггера

```
photocentr=# create trigger um_koll after insert on ord_serv for each row execute procedure um_koll();
```

Рисунок 45 – Создание триггера

```
photocentr=# create trigger check_del after update on client for each row execute procedure check_del();
CREATE TRIGGER
```

Рисунок 46 – Создание триггера

Просмотрим содержимое таблицы расход (см. Рисунок 47).

```
photocentr=# select * from rashod;
 id_ras | amount_ras |       name_rash
--------+------------+-----------------------
      2 |         30 | Глянцевая бумага 15x20
      3 |         10 | Матовая бумага 15x20
      1 |         40 | Глянцевая бумага 10x15
(3 строки)
```

Рисунок 47 – Просмотр таблицы

Проверим работу триггера на вычитание расходников (см. Рисунок 49), на удаление (см. Рисунок 50) и на занятость (см. Рисунок 48).

```
photocentr=# select * from employs;
 id_emp |  name_emp  | sur_name_emp | otchest_emp |     inn      |    snils    | id_pass | id_adr | zanyat
--------+------------+--------------+-------------+--------------+-------------+---------+--------+--------
      1 | Дмитрий    | Демидов      | Варданович  | 8524064052   | 65392407007 |       1 |      1 |
      3 | Дмитрий    | Демидов      | Варданович  | 8524064345   | 65392407884 |       3 |      3 | t
      2 | Константин | Гришин       | Федотович   | 8524224052   | 65392787007 |       2 |      2 | t
(3 строки)


photocentr=# select * from orders;
 id_k | id_emp | id_ord | data_ord | oplachen | id_os
------+--------+--------+----------+----------+-------
(0 строк)
photocentr=# insert into orders values(1,1,1,current_date,false,2);
INSERT 0 1
photocentr=# select * from orders;
 id_k | id_emp | id_ord |  data_ord  | oplachen | id_os
------+--------+--------+------------+----------+-------
    1 |      1 |      1 | 2021-05-06 | f        |     2
(1 строка)


photocentr=# select * from employs;
 id_emp |  name_emp  | sur_name_emp | otchest_emp |     inn      |    snils    | id_pass | id_adr | zanyat
--------+------------+--------------+-------------+--------------+-------------+---------+--------+--------
      3 | Дмитрий    | Демидов      | Варданович  | 8524064345   | 65392407884 |       3 |      3 | t
      2 | Константин | Гришин       | Федотович   | 8524224052   | 65392787007 |       2 |      2 | t
      1 | Дмитрий    | Демидов      | Варданович  | 8524064052   | 65392407007 |       1 |      1 | t
(3 строки)
```

Рисунок 48 – Проверка работы триггера

```
photocentr=# select * from rashod;
 id_ras | amount_ras |       name_rash
--------+------------+------------------------
      2 |         30 | Глянцевая бумага 15x20
      3 |         10 | Матовая бумага 15x20
      1 |         40 | Глянцевая бумага 10x15
(3 строки)


photocentr=# select * from ord_serv;
 ammount_s | id_serv | id_ras | id_os
-----------+---------+--------+-------
(0 строк)


photocentr=# insert into ord_serv values(20,1,1,2);
INSERT 0 1
photocentr=# select * from ord_serv;
 ammount_s | id_serv | id_ras | id_os
-----------+---------+--------+-------
        20 |       1 |      1 |     2
(1 строка)


photocentr=# select * from rashod
photocentr-# ;
 id_ras | amount_ras |       name_rash
--------+------------+------------------------
      2 |         30 | Глянцевая бумага 15x20
      3 |         10 | Матовая бумага 15x20
      1 |         20 | Глянцевая бумага 10x15
(3 строки)
```

Рисунок 49 – Проверка работы триггера

```
photocentr=# insert into client values(4,'Demidov','Dima','Kulkovich','sfsdsfd@gmai.com',89281178374, false);
INSERT 0 1
photocentr=# select * from client;
 id_k | surname  | firstname |  otchest   |       e_mail         |    mob_p    | check_del
------+----------+-----------+------------+----------------------+-------------+-----------
    1 | Ященко   | Артём     | Николаевич | hellohek232@gmail.com | 89924567255 |
    2 | Молчанов | Бронислав | Мэлсович   |                      | 89764567255 |
    3 | Кудряшов | Наум      | Германович |                      | 89732567267 |
    4 | Demidov  | Dima      | Kulkovich  | sfsdsfd@gmai.com     | 89281178374 | f
(4 строки)


photocentr=# update client set check_del=true where check_del=false;
UPDATE 1
photocentr=# select * from client;
 id_k | surname  | firstname |  otchest   |       e_mail         |    mob_p    | check_del
------+----------+-----------+------------+----------------------+-------------+-----------
    1 | Ященко   | Артём     | Николаевич | hellohek232@gmail.com | 89924567255 |
    2 | Молчанов | Бронислав | Мэлсович   |                      | 89764567255 |
    3 | Кудряшов | Наум      | Германович |                      | 89732567267 |
(3 строки)
```

Рисунок 50 – Проверка работы триггера

Создание индексов (см. Рисунок 51).

```
photocentr=# create index ind_emp on employs(id_emp);
CREATE INDEX
photocentr=# create index ind_ord on orders(id_ord);
CREATE INDEX
photocentr=# create index ind_k on client(id_k);
CREATE INDEX
photocentr=# create index ind_serv on services(id_serv);
CREATE INDEX
photocentr=# create index ind_pass on  passport(id_pass);
CREATE INDEX
photocentr=# create index ind_s on  street(id_s);
CREATE INDEX
photocentr=# create index ind_ob on  oblast(id_obl);
CREATE INDEX
photocentr=# create index ind_city on  city(id_city);
CREATE INDEX
photocentr=# create index ind_adr on  adress(id_adr);
CREATE INDEX
photocentr=# create index ind_rash on  rashod(id_ras);
CREATE INDEX
```

Рисунок 51 – Создание индексов

Создадим представление с информацией паспортов (см. Рисунок 52).

```
photocentr=# create view pass_inf as select seria as Серия,number_pas as Номер, kem_vidan as Кем_выдан,birthday as День_
рождение, kod_pod as Номер_подразделения, date_of_isse Дата_выдачи from passport inner join employs on passport.id_pass=
employs.id_pass;
CREATE VIEW
photocentr=# select * from pass_inf;
 Серия | Номер  |              Кем_выдан               | День_рождение | Номер_подразделения | Дата_выдачи
-------+--------+--------------------------------------+---------------+---------------------+-------------
 4356  | 654387 | Отделом УФМС по Ростовской области | 1986-10-22    | 905621              | 2000-11-21
 4374  | 651387 | Отделом УФМС по Ростовской области | 1986-01-28    | 905864              | 2000-02-04
 7674  | 321387 | Отделом УФМС по Ростовской области | 1987-06-02    | 543864              | 1999-06-06
(3 строки)
```

Рисунок 52 – Создание представления с информацией паспартов

Создадим представления с информацией о услугах (см. Рисунок 53)

```
photocentr=# create view serv_inf as select name_serv as Название_услуги,price as цена from services inner join ord_serv
 on services.id_serv=ord_serv.id_serv;
CREATE VIEW
photocentr=# select * from serv_inf
photocentr-# ;
                Название_услуги                | цена
-----------------------------------------------+-------
 Распечатка фотографии на глянцевой бумаги 10x15 |  9.00
 Распечатка фотографии на глянцевой бумаги 15x20 | 18.00
```

Рисунок 53 – Создание представления с информацией о услугах

Также создадим представления с информацией о сотрудниках (см. Рисунок 54)

```
photocentr=# create view emp_inf as select name_emp as Имя_сотрудника, sur_name_emp as Фамилия_сотрудника, otchest_emp a
s Отчество_сотрудника, INN as ИНН, snils as снилс, zanyat as занят from employs inner join orders on employs.id_emp=orde
rs.id_emp  inner join services on employs.id_emp=services.id_emp;
CREATE VIEW
```

Рисунок 54 – Создание представления с информацией о сотрудниках

3.3 Администрирование и защита базы данных

Понятие защиты данных также включает в себя множество аспектов: от защиты персональных данных в БД, до периодического мониторинга выделенных пользователям СУБД прав и ролей. Правильно организованная политика защиты данных требует журналирования операций доступа к данным, а также средств предотвращения несанкционированного доступа, средств единого управления ключами шифрования, и доступа и т.д.

PostgreSQL основан на дискреционной защите СУБД, которая предполагает разграничение доступа между поименованными объектами и субъектами.

В системе предусмотрено использование трёх категорий субъектов: администратор – с полными правами, сотрудник, который имеет право на просмотр и изменение некоторых таблиц и клиент, который сможет оформлять заказ на услугу, а также просматривать виды услуг.

Для работы с базой данных ИС были созданы следующие пользователи (см. рисунок 55).

```
photocentr=# create role admin_p with login createrole createdb password '111';
CREATE ROLE
photocentr=# create role worker login password '222';
CREATE ROLE
photocentr=# grant select, insert, update ,delete on table services to worker;
GRANT
photocentr=# grant select, insert, update ,delete on table orders to worker;
GRANT
```

Рисунок 55 - Создание пользователей

Далее дадим администратору все права на базу данных (см. рисунок56)

```
photocentr=# grant all  privileges on table orders to admin_p;
GRANT
photocentr=# grant all  privileges on table client to admin_p;
GRANT
photocentr=# grant all  privileges on table adress to admin_p;
GRANT
photocentr=# grant all  privileges on table passport to admin_p;
GRANT
photocentr=# grant all  privileges on table city to admin_p;
GRANT
photocentr=# grant all  privileges on table oblast to admin_p;
GRANT
photocentr=# grant all  privileges on table street to admin_p;
GRANT
photocentr=# grant all  privileges on table rashod to admin_p;
GRANT
photocentr=# grant all  privileges on table serv_rashod to admin_p;
GRANT
photocentr=# grant all  privileges on table services to admin_p;
GRANT
photocentr=# grant all  privileges on table ord_serv to admin_p;
GRANT
photocentr=# grant all  privileges on table employs to admin_p;
GRANT
```

Рисунок 56 - Присваивание прав

Проверим права администратора (см. рисунок 57)

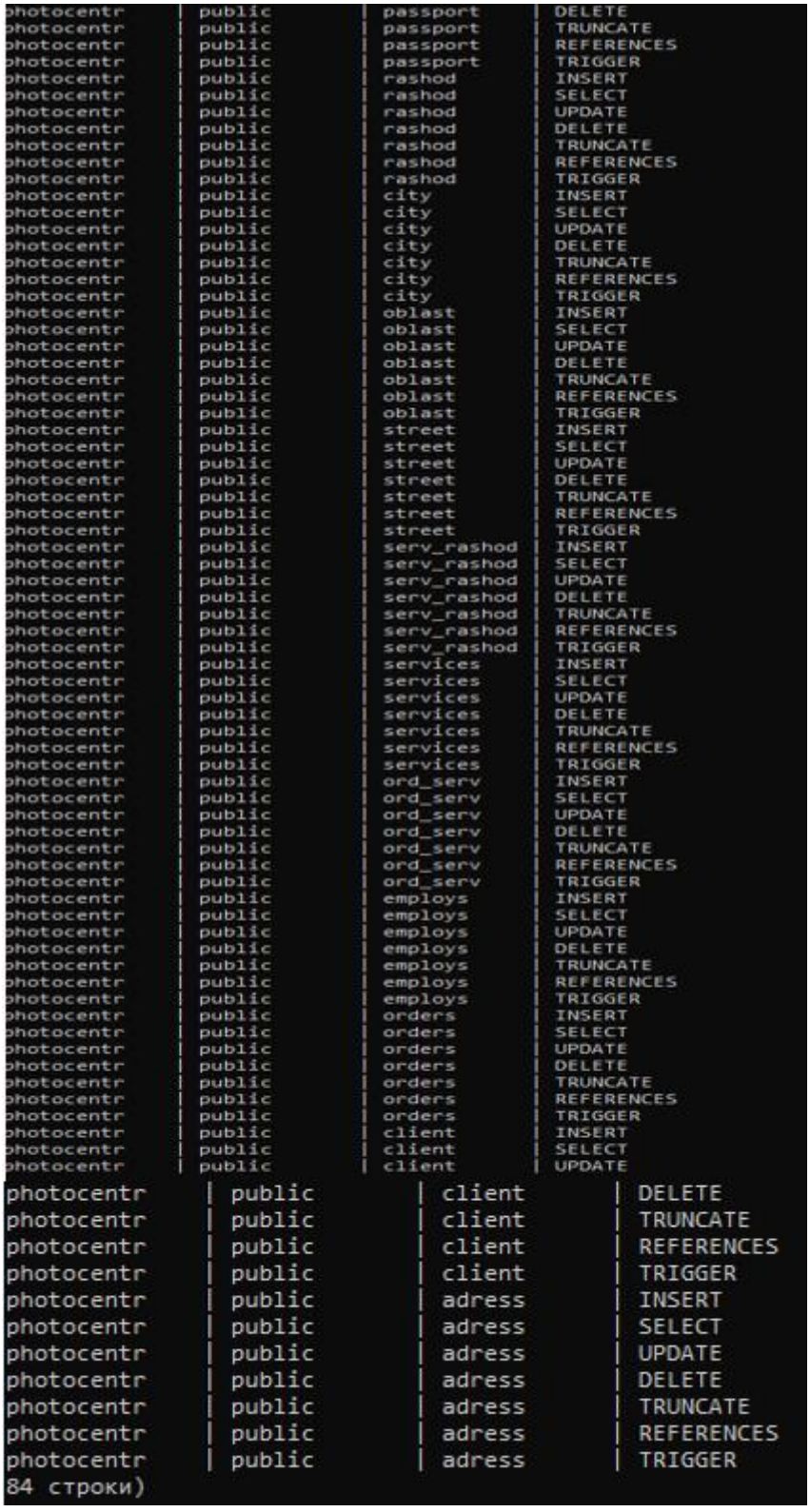

Рисунок 57 - Просмотр прав

Даем сотруднику право на просмотр, обновление и вставку в некоторых таблицах (см. рисунок 58).

```
photocentr=# grant select,update,insert,delete on table orders to worker;
GRANT
photocentr=# grant select,update,insert,delete on table services to worker;
GRANT
```

Рисунок 58 - Присваивание прав на просмотр, изменение и обновление некоторых таблиц

И проверим права сотрудника. (см. Рисунок 59).

```
photocentr=# select table_catalog,table_schema,table_name,privilege_type from information_schema.table_privileges where
grantee='worker';
 table_catalog | table_schema | table_name | privilege_type
---------------+--------------+------------+----------------
 photocentr    | public       | orders     | INSERT
 photocentr    | public       | orders     | SELECT
 photocentr    | public       | orders     | UPDATE
 photocentr    | public       | orders     | DELETE
 photocentr    | public       | services   | INSERT
 photocentr    | public       | services   | SELECT
 photocentr    | public       | services   | UPDATE
 photocentr    | public       | services   | DELETE
(8 строк)
```

Рисунок 59 – Просмотр прав

Для создания резервных копий, воспользуемся консольной утилитой pg_dump. Для этого сначала зайдем в папку с файлом pg_dump помощью командной строки. Далее введем команду (см. рисунок 60). После выполнения команды, система спросит у вас пароль пользователя PostgreSQL, указанного в команде, от имени которого выполняется создание резервной копии.

```
c:\Program Files\PostgreSQL\9.3\bin>pg_dump  -U postgres photocentr > ql.hello
Пароль:

c:\Program Files\PostgreSQL\9.3\bin>
```

Рисунок 60 - Ввод команды для резервного копирования

3.4 Реализация функций программного продукта.

После запуска программы открывается окно авторизации, форма которого приведена на рисунке 61.

Для авторизации Вам необходимо сделать следующее:

1) выберите нужный логин в списке пользователей;
2) введите пароль (при первом запуске для пользователя Администратор – пароль 111, для пользователя Сотрудник – пароль 222);
3) нажмите на кнопку «Войти». После успешной авторизации Вы сможете продолжить работу с программой. В случае ошибки на экран будет выведено соответствующее сообщение.

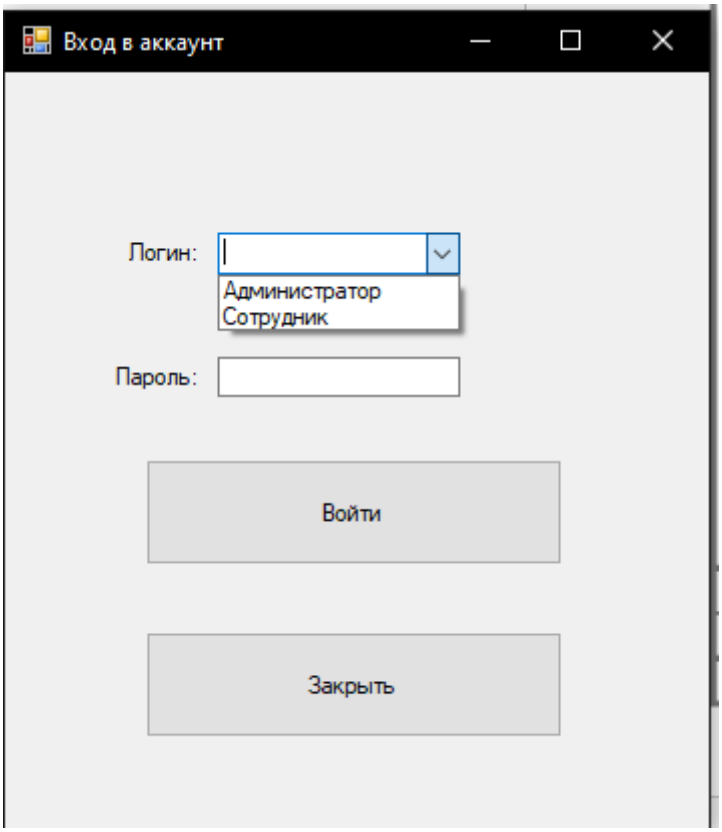

Рисунок 61. Окно авторизации

Программный код формы авторизации представлен в приложении Д.

После успешной авторизации открывается основное окно программы. В зависимости от того, под каким пользователем произведен вход в систему, вид меню окна будет несколько различным. (см. Рисунок 62).

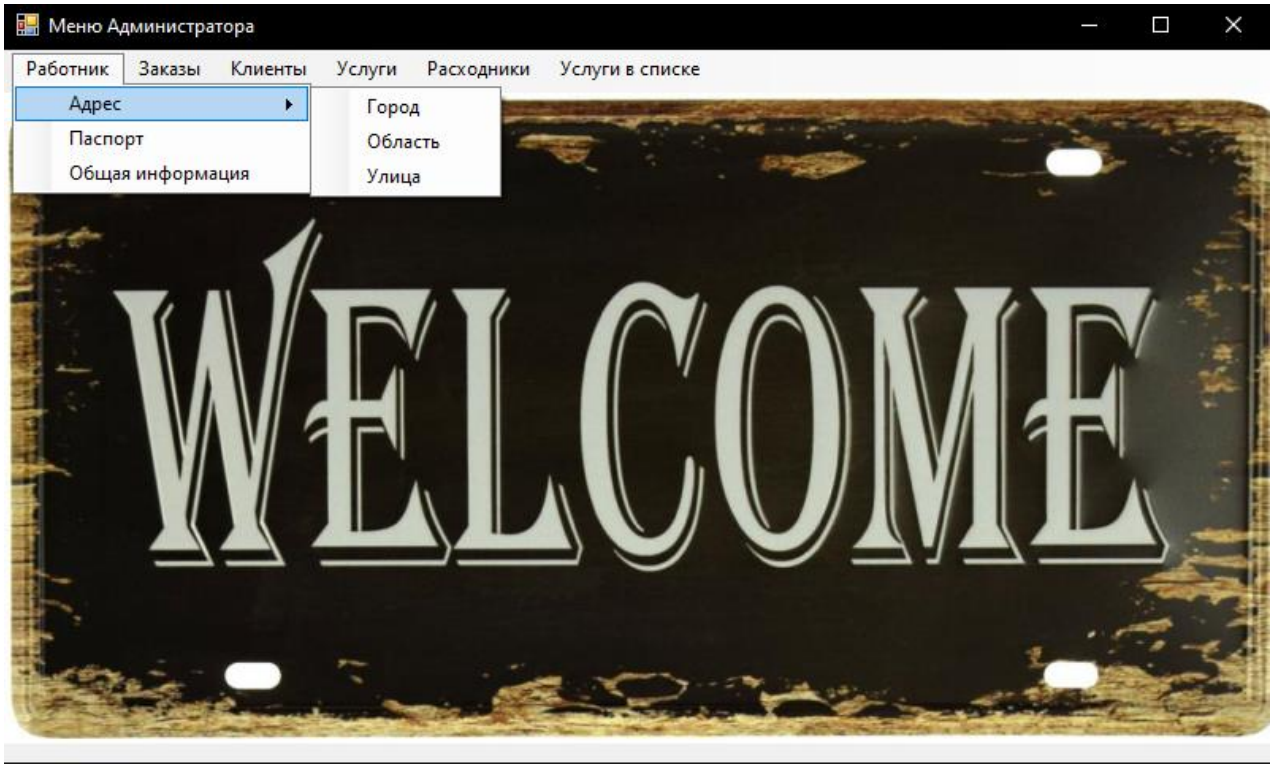

Рисунок 62. Основное окно программы

Программный код формы авторизации представлен в приложении Е.

Запуск функциональных модулей программы осуществляется с помощью выбора пунктов из меню «Работники», «Услуги», «Заказы», «Клиенты», «Расходники», «Услуги в списке». Результаты работы этих модулей отображаются в дополнительно открываемых окнах (формах). Для выбора параметров поиска на формах запросов предусмотрен набор переключателей (радиокнопок).

Создание и редактирование картотек данных объектов.

Выбрать нужный пункт из меню «Работник». При этом откроется соответствующее окно экземпляра картотеки, например, окно «Паспорт» картотеки «Passport», структура которого приведена на рисунке 74.

Для добавления нового экземпляра картотеки в базу данных нужно перейти в конец картотеки, ввести необходимую информацию в текстовые поля формы и сохранить внесенные данные, нажав кнопку «Добавить» (см. Рисунок 63).

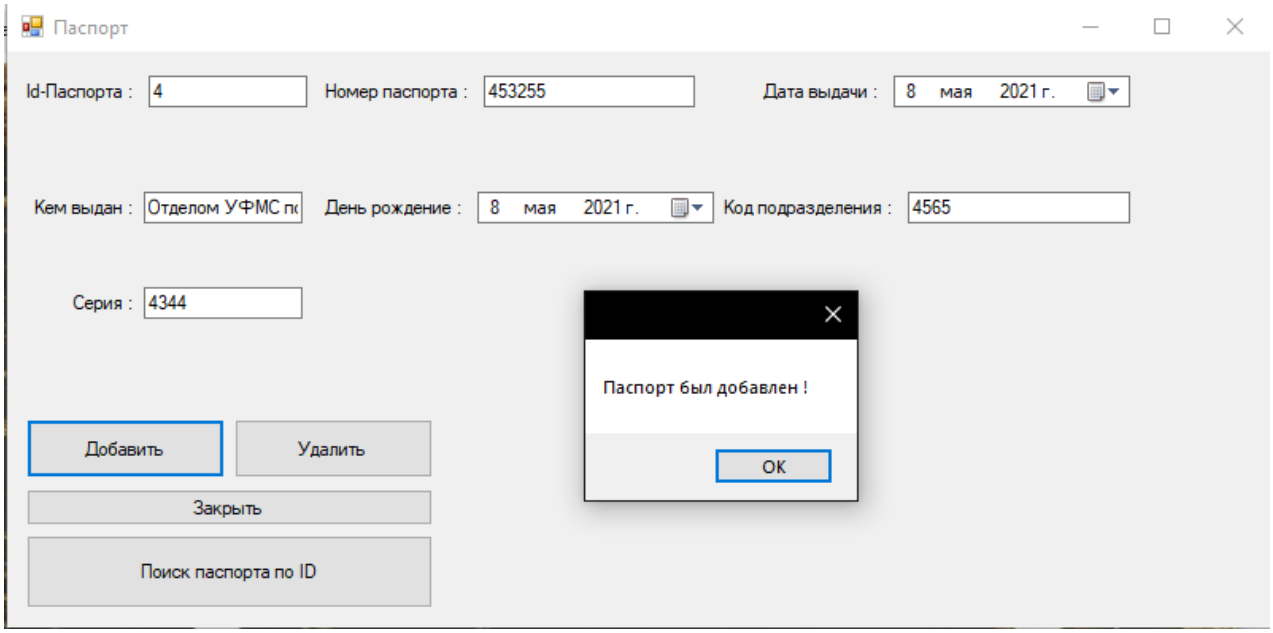

Рисунок 63. Окно «Паспорт»

Программный код формы авторизации представлен в приложении Ж.

Для изменения данных текущего экземпляра картотеки в базе данных необходимо, откорректировав необходимую информацию в текстовых полях формы, нажать кнопку «Добавить».

Для удаления текущего экземпляра картотеки из базы данных необходимо нажать кнопку «Удалить» и далее подтвердить удаление (см. Рисунок 64).

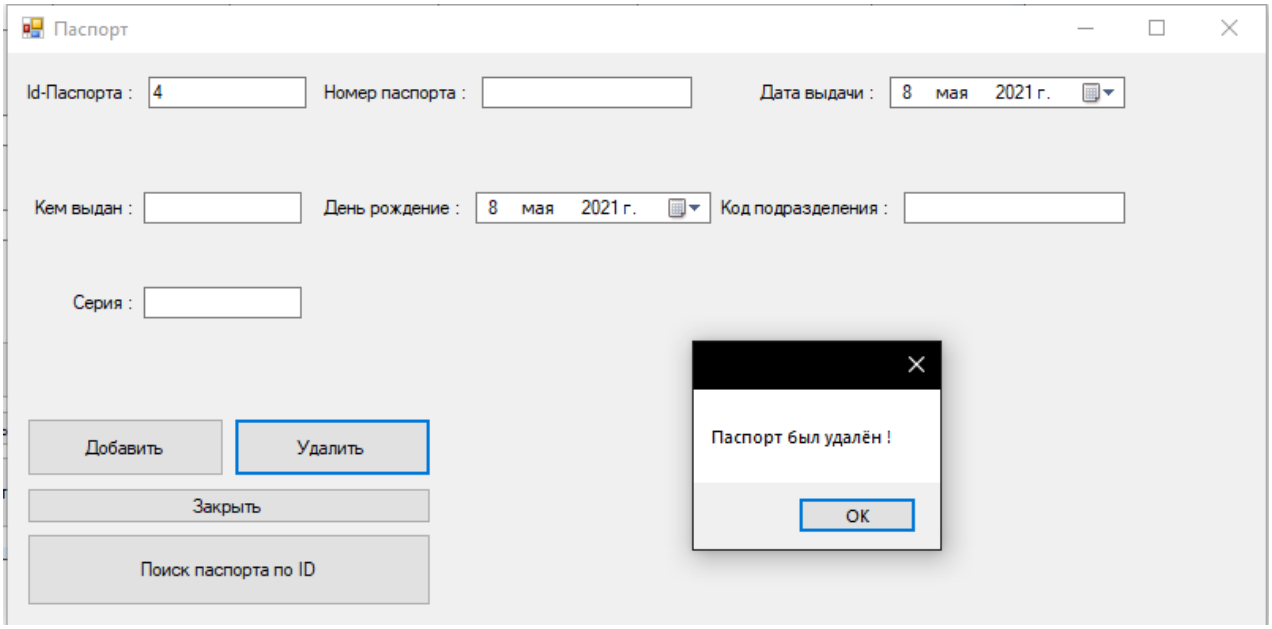

Рисунок 64. Удаление экземпляра

Программный код формы представлен в приложении И.

Вывод всех данных из таблицы passport через кнопку “Вывод всех данных” (см. Рисунок 65)

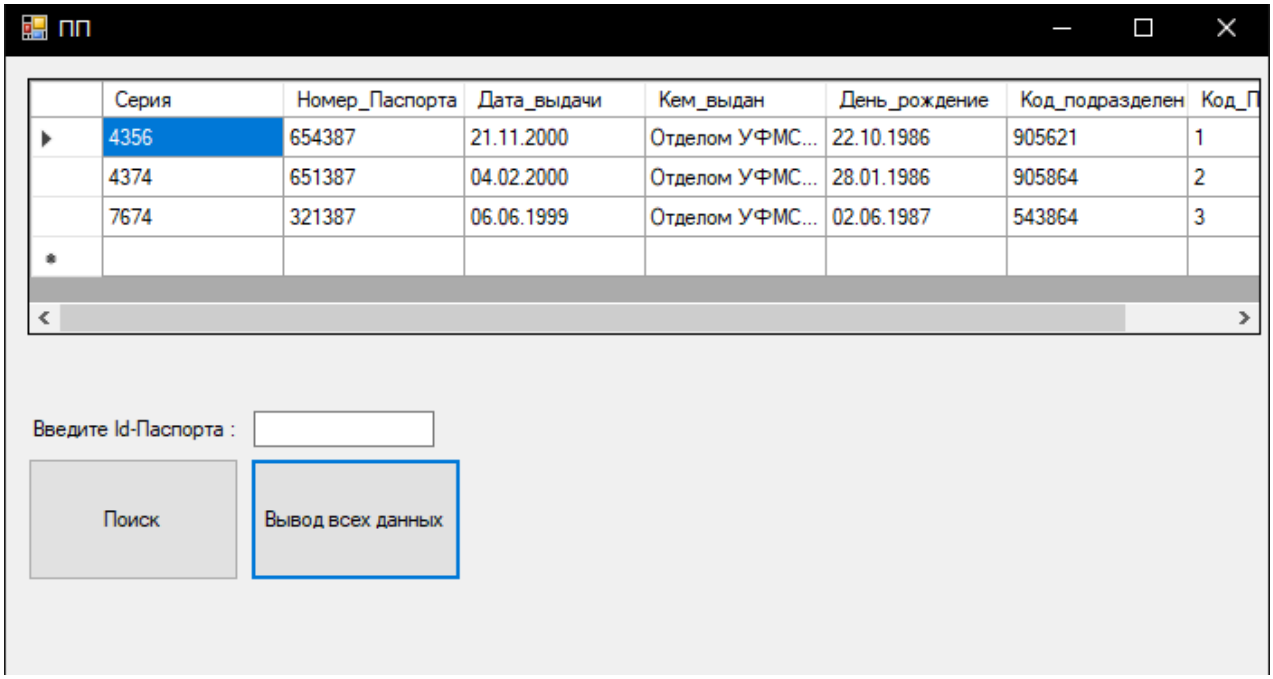

Рисунок 65 - Окно «Услуги»

Программный код формы представлен в приложении К.

Глава 4. ТЕСТИРОВАНИЕ И ОТЛАДКА ПРОГРАММНОГО ПРОДУКТА

Отладка программы так или иначе принимает рассмотрение и логическое решение доступной информации о ошибках. Большая часть ошибок может быть узнана к косвенным знакам посредством тщательного анализа текстов программ и результатов тестирования, не получая дополнительной информации.

Метод ручного тестирования - самый простой и естественный метод. В обнаружении ошибок необходимо выполнить протестированную программу вручную, работая с указанной ошибкой.

Из всех инструментов встроенного отладчика наиболее часто использовались точки остановки. После установки точки, приложение будет работать до тех пор, пока не достигнет ее, после чего работа программы будет остановлена и управление переходит к отладчику VisualStudio. Точки остановки удобнее всего снимать и ставить нажатием правой кнопки мыши на строке, где должны находиться точка останова, либо зайти в меню «Отладка» -> «Создать точку останова».

После остановки работы программы, анализируются значения локальных переменных процедуры, в точке прерывания работы программы. Кроме этого необходимо изучить стек вызовов, производимых до вызова данной процедуры.

Целью тестирования является выявление дефектов в программной системе.

Чтобы охватить все аспекты удобства использования проводилось тестирование удобства использования. Целью такого тестирования является оценка приемлемости пользовательского интерфейса приложения (время, затраченное на достижение цели, полученный результат, легкость доступа к нужной информации, интерпретация ответов системы и т.д.)

В ходе тестирования безопасности проводилась оценка уязвимости системы по отношению к атакам. Тестирование безопасности проверяет программу на попытки их взлома и обхода.

Тест кейсы представлены в таблицах п.1-п.3

Название:	Тест добавления информации о паспорте	
Функция:	Добавление информации	
Действие:	Ожидаемый результат	Результат теста: выполнен
Предусловие:		
Запуск приложения	Приложение запустилось	пройден
Появление главной формы	Главная форма присутствует	пройден
Шаги теста:		
Нажать на пункт меню “Работник”	Меню открывается	пройден
Нажать на пункт меню “Паспорт”	Открывается форма “Паспорт”	пройден
Постусловие:		
Сохранение информации о клиенте в базу данных	Информация о паспорте успешно сохранена в таблицу базы данных	пройден

Рисунок п.1 – Тест кейс на добавление информации о паспорте

Название:	Тест вывода информации о паспортах	
Функция:	Вывод информации	
Действие:	Ожидаемый результат	Результат теста: выполнен
Предусловие:		
Запуск приложения	Приложение запустилось	пройден
Появление главной формы	Главная форма присутствует	пройден
Шаги теста:		
Нажать на пункт меню “Работник”	Меню открывается	пройден
Нажать на пункт меню “Паспорт”	Открывается форма “Паспорт”	пройден
Нажать на кнопку “Поиск паспорта по ID”	Открывается форма с поиском паспорта	пройден
Нажать на кнопку “вывод всех данных”	Выводится информация о паспортах	пройден
Постусловие:		
Вывод данных в форме и их просмотр	Информация о паспортах Была успешно выведена	пройден

Рисунок п.2 – Тест кейс на вывод информации о паспортах

Название:	Тест входа в аккаунт	
Функция:	Вход	
Действие:	Ожидаемый результат	Результат теста: выполнен
Предусловие:		
Запуск приложения	Приложение запустилось	пройден
Появление главной формы	Главная форма присутствует	пройден
Шаги теста:		
Ввести логина	Данные вводятся	пройден
Ввести пароль	Пароль вводится	пройден
Нажать на кнопку “Вход”	Открывается форма в зависимости от выбранного аккаунта	пройден
Постусловие:		
Вход в аккаунт и получение его прав	Авторизация прошла успешно, и права были получены	пройден

Рисунок п.3 – Тест кейс на вход в аккаунт

Заключение

В ходе проектирования и реализации БД были получены знания об основах работы с СУБД PostgreSQL, написании подпрограмм на языке SQL, получен практический опыт самостоятельного проектирования.

В курсовой работе поставлены и решены следующие задачи:

- Выполнен анализ предметной области на примере работы фотоцентра, построена его информационно-логическая модель.
- Спроектирована схема данных и выполнена ее нормализация.
- Разработаны таблицы, формы, запросы и отчеты.
- Выполнена верификация БД.

Спроектированная база данных состоит из 11 связанных таблиц: паспорт, сотрудники, клиенты, расходники, услуги, улица, адрес, область, город, заказы, товары в заказе. На их основе созданы запросы.

Разработка и внедрение базы данных «Фотоцентр» предполагает уменьшение временных затрат на поиск и оперативное получение необходимой информации об услугах и заказах.

В процессе выполнения курсового проекта было разработано приложение, которое обеспечивает выполнение следующих функций:

- Введение журнала услуг.
- Введение журнала заказов.
- Поиск по базе данных информации по услугам и паспортам.
- Возможность внесения данных о клиентах, сотрудниках и услугах в базу данных.
- Редактирование и удаление данных.

При написании программы основное внимание было уделено удобству работы пользователя с программой и построению дружественного интерфейса.

Результаты тестирования показали, что программа работает верно во всех предполагаемых ситуациях и отвечает требованиям задания на курсовой проект студента.

Список использованных источников и литературы

1. С.Тарасов СУБД для программиста 2017г. 312с.
2. Документация mysql - https://dev.mysql.com/doc/
3. Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2017. - 633 с.
4. Кузнецов, С. Д. Основы баз данных / С.Д. Кузнецов. - М.: Бином. Лаборатория знаний, Интернет-университет информационных технологий, 2017. - 488 с.
5. Свиридова, М. Ю. Система управления базами данных/ М.Ю. Свиридова. - М.: Академия, 2018. - 192 с.
6. Латыпова, Р. Р. Базы данных. Курс лекций / Р.Р. Латыпова. - Москва: Высшая школа, 2016. - 177 с.
7. Остроух, А. В. Ввод и обработка цифровой информации / А.В. Остроух. - М.: Академия, 2016. - 288 с
8. Кириллов, В.В. Введение в реляционные базы данных (+ CD-ROM) / В.В. Кириллов. - М.: БХВ-Петербург, 2016. - 318 с.
9. Каратыгин, С. Базы данных / С. Каратыгин, А. Тихонов, В. Долголаптев. - М.: ABF, 2016. - 352 с.
10. Мюллер, Р.Дж. Базы данных и UML. Проектирование / Р.Дж. Мюллер. - М.: ЛОРИ, 2015. - 420 с.
11. Хансен Базы данных: разработка и управление / Хансен, Хансен Генри; Джеймс. - М.: Бином, 2015. - 704 с.
12. Фленов, Михаил Библия C# (+ CD-ROM) / Михаил Фленов. - М.: БХВ-Петербург, 2017. - 560 с.
13. Фримен, Адам LINQ. Язык интегрированных запросов в C# 2010 для профессионалов / Адам Фримен, Джозеф Раттц-мл. - М.: Вильямс, 2017. - 656 с.
14. Бишоп, Дж. C# в кратком изложении / Дж. Бишоп, Н. Хорспул. - М.: Бином. Лаборатория знаний, 2018. - 472 с.
15. Павловская Т. А. C#. Программирование на языке высокого уровня: Учебник для вузов. - СПб: БХВ-Петербург. 2017.
16. https://en.wikipedia.org/wiki/Cardinality_%28data_modeling%29
17. https://coderlessons.com/tutorials/bazy-dannykh/osnovy-sql/4-normalizatsiia-2#1

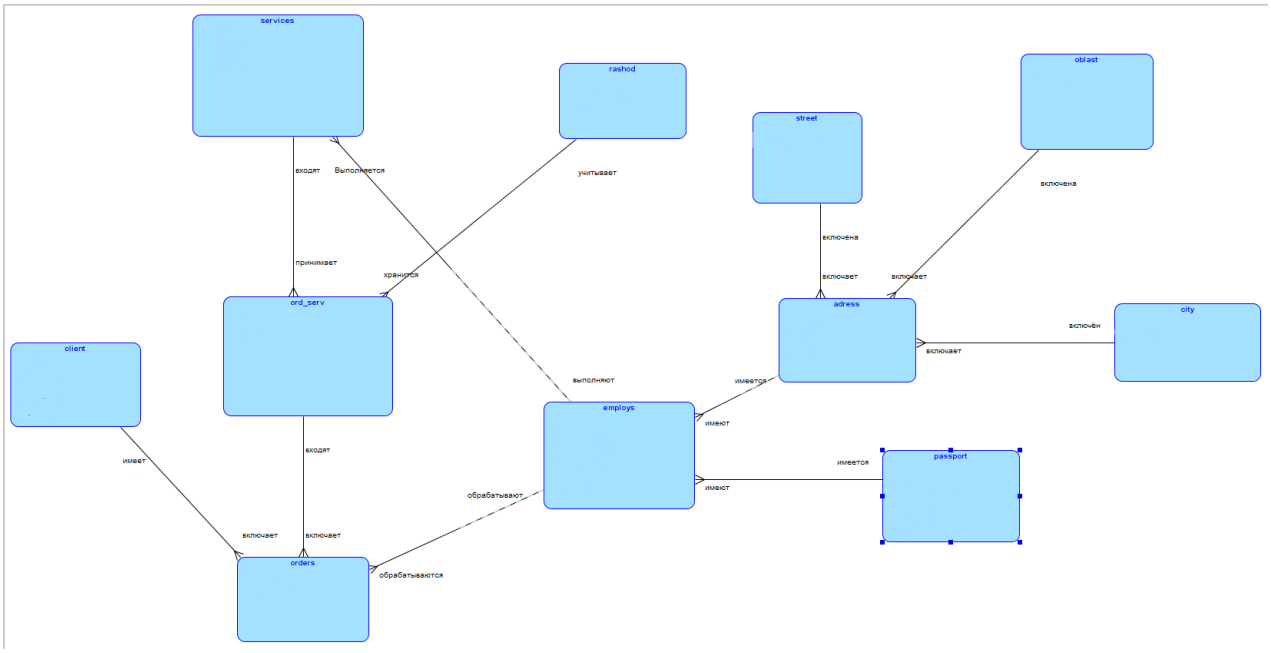

Рисунок А.1. – Концептуальная модель

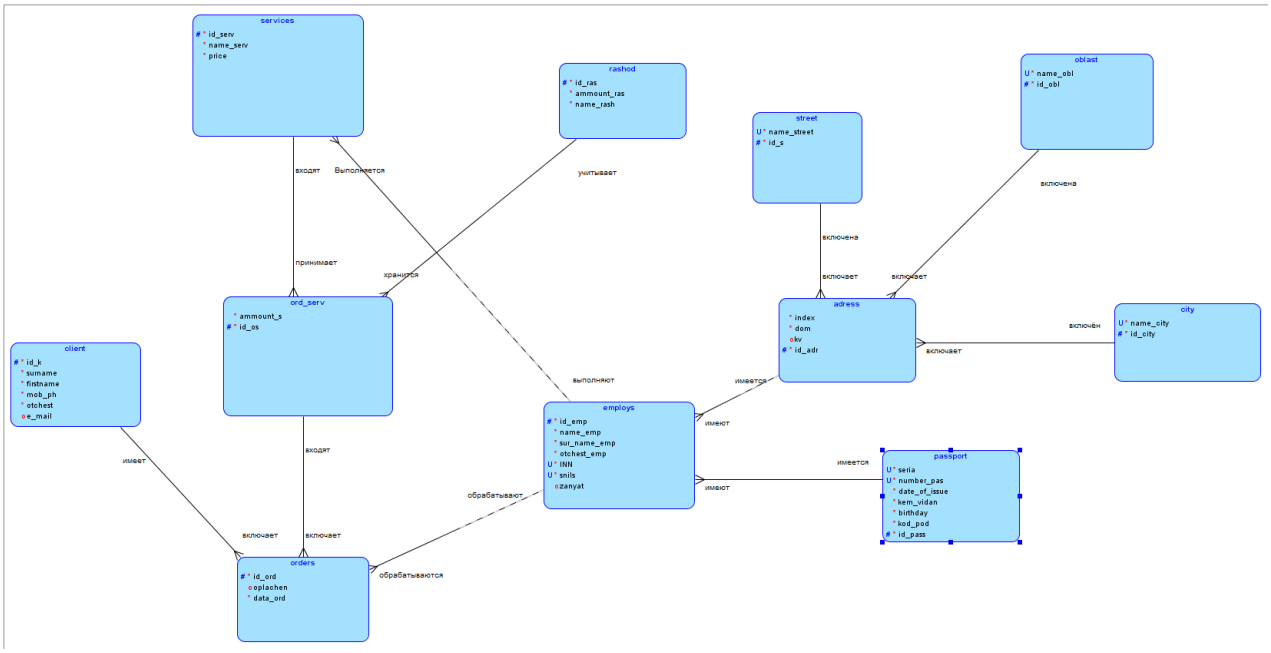

Рисунок Б.1. – Логическая модель

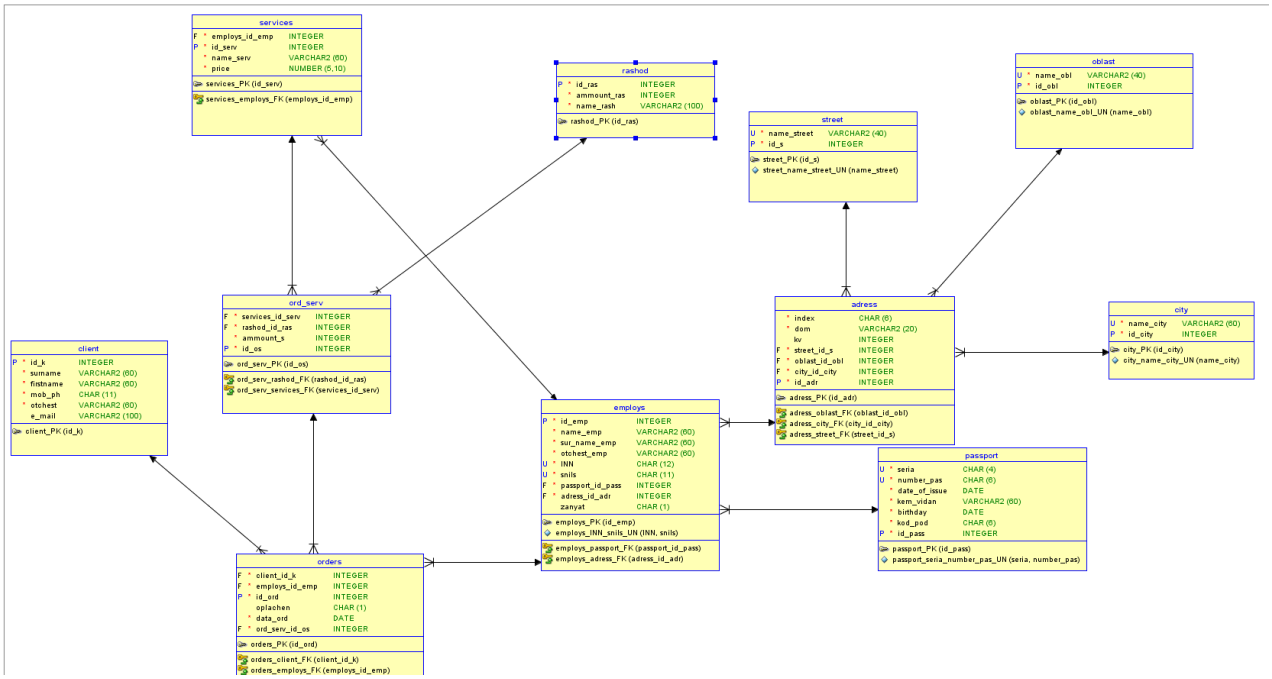

Рисунок В.1. – Физическая модель

Приложение Г

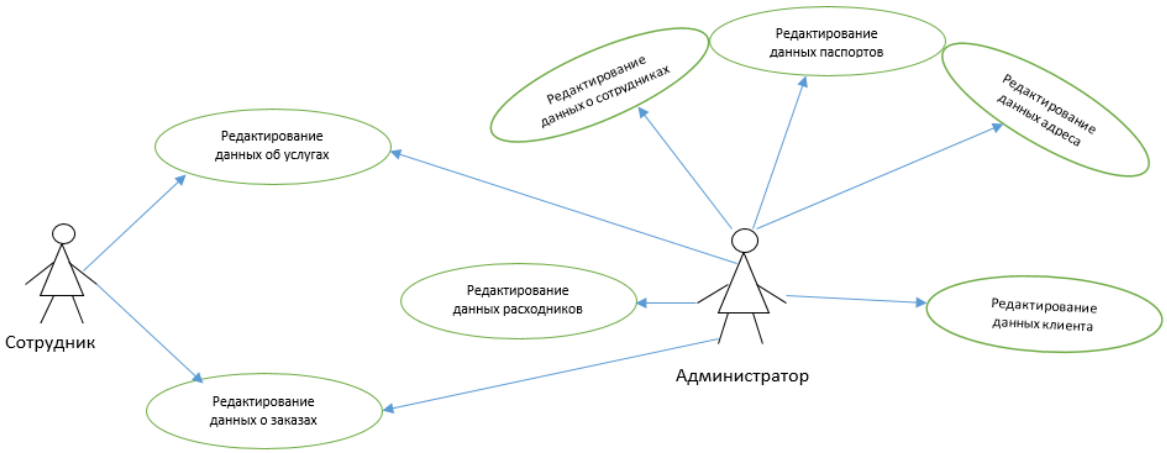

Рисунок Г.1. – Диаграмма использования

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Npgsql;

namespace PhotoCentR
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            if ((comboBox1.Text == "Администратор") && (textBox1.Text =="111"))
            {
                String connectionString =
"Host=localhost;Username=admin_p;Password=111;Database=photocentr";
                NpgsqlConnection npgsqlConnection = new
NpgsqlConnection(connectionString);
                Form5 form5 = new Form5();
                form5.Show();
                this.Hide();
            }
            else if ((comboBox1.Text =="Сотрудник") && (textBox1.Text == "222"))
            {
                String connectionString =
"Host=localhost;Username=worker;Password=222;Database=photocentr";
                NpgsqlConnection npgsqlConnection = new
NpgsqlConnection(connectionString);
                Form4 form4 = new Form4();
                form4.Show();
                this.Hide();
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Close();
        }
    }
        }
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Npgsql;
namespace PhotoCentR
{
    public partial class Form5 : Form
    {
        public Form5()
        {
            InitializeComponent();
        }

        private void городToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Город город = new Город();
            город.Show();
        }

        private void областьToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Область область = new Область();
            область.Show();
        }

        private void общаяИнформацияToolStripMenuItem_Click(object sender, EventArgs e)
        {

        }

        private void паспортToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Паспорт паспорт = new Паспорт();
            паспорт.Show();
        }

        private void услугиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Услуги услуги = new Услуги();
            услуги.Show();

        }

        private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)
        {

        }
    }
        }
```

```
String connectionString =
"Host=localhost;Username=admin_p;Password=111;Database=photocentr";
            NpgsqlConnection npgsqlConnection = new NpgsqlConnection(connectionString);
            npgsqlConnection.Open();
            NpgsqlCommand npgsqlCommand = new NpgsqlCommand("insert into passport
values(@id_pas,@seria,@number_pas,@date_of_issue,@kem_vidan,@birthday,@kod_pod)",
npgsqlConnection);
            int id_pas = Convert.ToInt32(textBox1.Text);
            DateTime date_of_issue =Convert.ToDateTime (dateTimePicker1.Text);
            string kem_vidan = textBox4.Text;
            string seria = textBox7.Text;
            string number_pas = textBox2.Text;
            DateTime birthday =Convert.ToDateTime (dateTimePicker2.Text);
            string kod_pod = textBox6.Text;
            NpgsqlParameter npgsqlParameterIdPas = new NpgsqlParameter("@id_pas",
NpgsqlTypes.NpgsqlDbType.Integer);
            NpgsqlParameter npgsqlParameterSeria = new NpgsqlParameter("@seria",
NpgsqlTypes.NpgsqlDbType.Char);
            NpgsqlParameter npgsqlParameterKodPod = new NpgsqlParameter("@kod_pod",
NpgsqlTypes.NpgsqlDbType.Char);
            NpgsqlParameter npgsqlParameterNumberPas = new NpgsqlParameter("@number_pas",
NpgsqlTypes.NpgsqlDbType.Char);
            NpgsqlParameter npgsqlParameterBirthday = new NpgsqlParameter("@birthday",
NpgsqlTypes.NpgsqlDbType.Date);
            NpgsqlParameter npgsqlParameterKemVidan = new NpgsqlParameter("@kem_vidan",
NpgsqlTypes.NpgsqlDbType.Varchar);
            NpgsqlParameter npgsqlParameterDateOfIssue = new
NpgsqlParameter("@date_of_issue", NpgsqlTypes.NpgsqlDbType.Date);
            npgsqlParameterIdPas.Value = id_pas;
            npgsqlParameterSeria.Value = seria;
            npgsqlParameterKodPod.Value = kod_pod;
            npgsqlParameterNumberPas.Value = number_pas;
            npgsqlParameterBirthday.Value = birthday;
            npgsqlParameterKemVidan.Value = kem_vidan;
            npgsqlParameterDateOfIssue.Value = date_of_issue;
            npgsqlCommand.Parameters.AddRange(new NpgsqlParameter[] {
npgsqlParameterIdPas, npgsqlParameterSeria, npgsqlParameterKodPod,
npgsqlParameterNumberPas, npgsqlParameterBirthday, npgsqlParameterKemVidan,
npgsqlParameterDateOfIssue });
            int count6 = npgsqlCommand.ExecuteNonQuery();
            npgsqlConnection.Close();
            if (count6 == 1)
            {
                MessageBox.Show("Паспорт был добавлен !");
            }
            else
            {
                MessageBox.Show("Ошибка что то введено не так");
                    }
```

```
String connectionString =
"Host=localhost;Username=admin_p;Password=111;Database=photocentr";
            NpgsqlConnection npgsqlConnection = new NpgsqlConnection(connectionString);
            npgsqlConnection.Open();
            NpgsqlCommand npgsqlCommand = new NpgsqlCommand("delete from passport where
id_pass = @id_pass", npgsqlConnection);
            int id_pass = Convert.ToInt32(textBox1.Text);
            NpgsqlParameter npgsqlParameterIdPas = new NpgsqlParameter("@id_pass",
NpgsqlTypes.NpgsqlDbType.Integer);
            npgsqlParameterIdPas.Value = id_pass;
            npgsqlCommand.Parameters.AddRange(new NpgsqlParameter[] { npgsqlParameterIdPas
});
            int count7 = npgsqlCommand.ExecuteNonQuery();
            npgsqlConnection.Close();
            if (count7 == 1)
            {
                MessageBox.Show("Паспорт был удалён !");
            }
            else
            {
                MessageBox.Show("Ошибка что то введено не так");
                    }
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Npgsql;
namespace PhotoCentR
{
    public partial class ПП : Form
    {
        public ПП()
        {
            InitializeComponent();
        }
        private DataSet ds = new DataSet();
        private DataTable dt = new DataTable();
        private void button1_Click(object sender, EventArgs e)
        {



        }

        private void button2_Click(object sender, EventArgs e)
        {
            NpgsqlConnection npgSqlConnection = new
NpgsqlConnection("Host=localhost;Username=admin_p;Password=111;Database=photocentr");
            npgSqlConnection.Open();
            string x = ("select  seria as Серия, number_pas as Номер_Паспорта,
date_of_isse as Дата_выдачи, kem_vidan as Кем_выдан, birthday as День_рождение, kod_pod as
Код_подразделение, id_pass as Код_Паспорта from passport");
            NpgsqlDataAdapter npgsqlDataAdapter = new NpgsqlDataAdapter(x,
npgSqlConnection);

            ds.Reset();
            npgsqlDataAdapter.Fill(ds);
            dataGridView1.ReadOnly = true;
            dt = ds.Tables[0];
            dataGridView1.DataSource = dt;
            npgSqlConnection.Close();
        }
    }
}
```