Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Сибирский государственный автомобильно-дорожный университет (СибАДИ)»

Институт Информационные системы, экономика и управление

Кафедра Информационная безопасность

Направление Информационная безопасность

Профиль Информационная безопасность автоматизированных систем

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовому проекту**

по дисциплине: «Безопасность систем баз данных»

Тема работы «Безопасность данных в автоматизированной системе потребительского кредитования (на примере использования PostgreSQL и языка программирования Python)»

Выполнил:

студент группы БИб-21Э1 Чигарев Александр Юрьевич

Курсовой проект сдан на проверку:

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г.

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Чигарев Александр Юрьевич

(подпись студента)

Курсовой проект допущен к защите:

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г.

Руководитель проекта

доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.М. Опарина

Консультант проекта

д.п.н., проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ З.В. Семенова

Курсовой проект защищен с оценкой:

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г.

Руководитель проекта

доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.М. Опарина

Консультант проекта

д.п.н., проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ З.В. Семенова

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Сибирский государственный автомобильно-дорожный университет (СибАДИ)»

Кафедра «Информационная безопасность»

УТВЕРЖДАЮ

И.О., Зав. кафедрой ИБ

\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.А. Панков

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

**ЗАДАНИЕ**

**к курсовому проекту**

студенту группы БИб-21Э1 Чигареву Александру Юрьевичу

1. Тема курсовому проекту: «Безопасность данных в автоматизированной системе потребительского кредитования (на примере использования PostgreSQL 16 и языка программирования Python)»
2. Исходные данные к курсовому проекту
   * Предметная область: Подразделение банковской организации по работе с потребительскими кредитами, предоставляющих населению денежные средства с условием возврата данных средств организации;
   * Минимальное количество таблиц – 8, минимальное количество информационных запросов – 8, минимальное количество ролей – 3;
   * Механизмы защиты:
     + Триггеры;
     + Политика RLC;
     + Шифрование данных;
     + Аутентификация встроенными средствами PostgreSQL 16.
   * Используемые инструменты:
     + СУБД, PostgreSQL 16 – серверная часть;
     + IDE PyCharm Community Edition 2023 – инструмент разработки клиентской части;
     + библиотека PyQt6 – клиентская часть;
     + psycopg2 – реализация связи клиент – сервер.
   * Официальная документация: Python, PostgreSQL 16;
3. Содержание пояснительной записки

* титульный лист;
* задание к курсовому;
* содержание;
* введение;
* технические аспекты предметной области ИС «Потребительское кредитование»;
* исследование базовых возможностей обеспечения безопасности СУБД PostgreSQL;
* проектировка и разработка ИС «Потребительское кредитование»;
* реализация основных механизмов защиты данных ИС «Потребительское кредитование»;
* заключение;
* список используемых источников.

1. Перечень демонстрационного материала для сопровождения доклада при защите проекта:
   * Презентация Microsoft PowerPoint.
2. Срок сдачи работы: «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г.

Задание выдано «\_12\_» \_\_февраля\_\_ 2024 г.

Руководитель проекта: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.М. Опарина

(подпись преподавателя)

Консультант проекта: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ З.В. Семенова

(подпись преподавателя)

Задание к исполнению принял «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г.

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. Ю. Чигарев

(подпись студента)

**Содержание**

[**Введение** 5](#_Toc169701295)

[**1. Анализ предметной области ИС «Потребительское кредитование»** 6](#_Toc169701296)

[**1.1 Описание предметной области** 6](#_Toc169701297)

[**1.2 Ключевые бизнес-процессы ИС «Потребительское кредитования»** 7](#_Toc169701298)

[**1.3 Формирование запросов к базе данных для удовлетворения потребностей бизнес-процессов** 11](#_Toc169701299)

[**2. Основные возможности обеспечения безопасности СУБД PostgreSQL** 13](#_Toc169701300)

[**2.1 Краткая характеристика основных возможностей PostgreSQL 16 для обеспечения безопасности** 13](#_Toc169701301)

[**2.2 Встроенные механизмы аутентификации и авторизации в PostgreSQL16** 14](#_Toc169701302)

[**2.3 Изучение методов шифрования данных в PostgreSQL 16** 17](#_Toc169701303)

[**2.4 Изучение политики RLC в PostgreSQL 16** 20](#_Toc169701304)

[**3. Реализация ИС «Потребительское кредитование» и обеспечение безопасности.** 22](#_Toc169701305)

[**3.1 Проектирование базы данных ИС «Потребительское кредитование»** 22](#_Toc169701306)

[**3.1.1 Логическое проектирование в ИС «Потребительское кредитование»** 22](#_Toc169701307)

[**3.1.2 Физическое проектирование в ИС «Потребительское кредитование» средствами СУБД** 24](#_Toc169701308)

[**3.2 Реализация отдельных механизмов защиты ИС «Потребительского кредитования»** 30](#_Toc169701309)

[**3.2.1 Реализация ролей в ИС «Потребительское кредитование» средствами СУБД** 30](#_Toc169701310)

[**3.2.2 Реализация RLC и разграничение доступа в ИС «Потребительское кредитование» средствами СУБД** 31](#_Toc169701311)

[**3.2.3 Реализация триггеров и шифрования в ИС «Потребительское кредитование» средствами СУБД** 34](#_Toc169701312)

[**3.3 Реализация пользовательского интерфейса для ИС «Потребительское кредитование» средствами Python и PyQt6** 39](#_Toc169701313)

[**4. Документация на ИС «Потребительское кредитование»** 43](#_Toc169701314)

[**4.1 Руководство по использованию ИС «Потребительское кредитование» для пользователя** 43](#_Toc169701315)

[**4.2 Руководство по использованию ИС «Потребительское кредитование» для менеджера** 44](#_Toc169701316)

[**4.3 Руководство по использованию ИС «Потребительское кредитование» для администратора** 47](#_Toc169701317)

[**Заключение** 50](#_Toc169701318)

[**Список используемых источников** 51](#_Toc169701319)

# **Введение**

Цифровизация финансового сектора становится все более значимой, требуя высоких стандартов безопасности для защиты конфиденциальной информации. В автоматизированных системах потребительского кредитования обрабатываются большие объемы данных о клиентах и их финансовом положении. Обеспечение безопасности данных в таких системах является критически важной задачей, поскольку любые утечки могут привести к серьезным последствиям.

Постоянный рост объема обрабатываемой информации и увеличивающиеся угрозы со стороны злоумышленников требуют применения современных технологий для защиты данных. Системы управления базами данных, такие как PostgreSQL 16, предоставляют широкий спектр возможностей для обеспечения безопасности. Использование языка программирования Python позволяет реализовать дополнительные механизмы защиты, улучшая общую устойчивость системы к внешним и внутренним угрозам [20].

В условиях стремительного развития цифровых технологий финансовые организации должны оперативно реагировать на возникающие угрозы и внедрять надежные методы защиты данных. Автоматизированные системы потребительского кредитования не являются исключением, и для них критически важно не только обеспечить надежное хранение и обработку данных, но и обеспечить их защиту от несанкционированного доступа.

**Цель:** обеспечение безопасности в информационной системе потребительского кредитования с использованием языка программирования Python и базы данных PostgreSQL 16.

**Задачи:**

1. Провести анализ предметной области "Потребительское кредитование" выделив бизнес-процессы и сформировав запросы к базе данных удовлетворяющие потребности бизнес-процессов;
2. Исследовать возможности обеспечения безопасности СУБД PostgreSQL;
3. Спроектировать и разработать ИС "Потребительское кредитование";
4. Реализовать основные механизмы защиты данных ИС «Потребительское кредитование».

# **1. Анализ предметной области ИС «Потребительское кредитование»**

## **1.1 Описание предметной области**

Потребительский кредит – это вид кредита, предоставляемый банками или другими финансовыми организациями физическим лицам для удовлетворения их личных потребностей, таких как покупка товаров, оплата услуг, ремонт жилья и т.д.

Информационная система потребительского кредитования (далее ИСПК) является ключевым инструментом для управления процессами кредитования, анализа рисков, мониторинга погашений и просрочек, а также для обеспечения эффективной работы кредитных организаций.

Предметная область ИСПК включает в себя совокупность объектов, процессов и данных, необходимых для управления потребительским кредитованием. Далее приводится описание основных компонентов предметной области:

1. Клиенты – являются одним из ключевых объектов предметной области ИСПК. Информация о клиентах включает в себя личные данные, контактную информацию, историю кредитования, кредитный рейтинг и финансовое положение. ИСПК должна предоставлять возможность для сбора, хранения, обработки и анализа информации о клиентах, а также для обеспечения конфиденциальности и безопасности данных.

2. Кредиты – являются основным объектом предметной области ИСПК. Информация о кредитах включает в себя сумму кредита, процентную ставку, срок погашения, расписание платежей, текущий статус и историю изменений. ИСПК должна предоставлять возможность для сбора, хранения, обработки и анализа информации о кредитах, а также для обеспечения эффективного управления процессами кредитования.

3. Заявки – на кредиты являются важным компонентом предметной области ИСПК. Информация о заявках на кредиты включает в себя дату подачи заявки, сумму запрашиваемого кредита, цель кредита, информацию о заемщике и созаемщиках, статус заявки и историю изменений. ИСПК должна предоставлять возможность для сбора, хранения, обработки и анализа информации о заявках на кредиты, а также для обеспечения эффективного управления процессами рассмотрения заявок.

4. Погашения – являются важными аспектами предметной области ИСПК. Информация о погашениях включает в себя дату погашения, сумму погашения, текущий статус и историю изменений. ИСПК должна предоставлять возможность для сбора, хранения, обработки и анализа информации о погашениях, а также для обеспечения эффективного управления процессами мониторинга и восстановления долгов.

5. Условия кредитования – являются важным компонентом предметной области ИСПК. Информация об условиях кредитования включает в себя процентные ставки, сроки погашения, комиссии, страхование и другие условия, предоставляемые заемщикам. ИСПК должна предоставлять возможность для сбора, хранения, обработки и анализа информации об условиях кредитования, а также для обеспечения эффективного управления процессами ценообразования и конкуренции.

6. Отчеты и аналитика – являются важным компонентом предметной области ИСПК. Информация об отчетах и аналитике включает в себя финансовые показатели, эффективность кредитования, риски и другие аспекты деятельности потребительского кредитования. ИСПК должна предоставлять возможность для сбора, хранения, обработки и анализа информации об отчетах и аналитике, а также для обеспечения эффективного управления процессами принятия решений и стратегического планирования.

## **1.2 Ключевые бизнес-процессы ИС «Потребительское кредитования»**

Ключевыми группами сотрудников в кредитной организации являются:

1. Кредитные менеджеры – ответственны за принятие решений о выдаче кредитов, мониторинг кредитных портфелей, взаимодействие клиентов при возникновении спорных вопросов по кредитованию.
2. Менеджеры по обслуживанию клиентов – ответственны за предоставление информации о продуктах и услугах компании, решение вопросов клиентов, управление жалобами и конфликтами связанных с обслуживанием клиентов, оформление заявок клиентов, пришедших в организацию лично.
3. Администраторы баз данных – ответственны за установку, настройку и обновление систем управления базами данных, создание и поддержание структуры баз данных, обеспечение безопасности данных, резервного копирования и восстановления.
4. Системные администраторы – ответственны за управление IT-инфраструктурой компании, установку и настройку ПО и ОС, а также драйверов и антивирусных систем, обеспечение сети и данных, обеспечение непрерывной работы инфраструктуры.
5. Кредитные аналитики – ответственны за проведение анализа финансового положения и других факторов, влияющих на риск кредитования. Аналитики также могут участвовать в разработке политик кредитования, моделей оценки риска и других инструментов, необходимых для эффективного управления рисками в потребительском кредитовании.

Рассмотрим пример функционирования информационной системы потребительских кредитов на примере выдачи потребительского кредита физическому лицу.

Работа кредитного отдела предполагает, что техническими специалистами проведена работа по установке и настройке автоматизированного рабочего места менеджера по обслуживанию клиентов, которое включает в себя рабочее пространство с необходимым оборудованием (ПК, монитор, клавиатура, мышь, принтер и т.д.), и установленным на это оборудование ПО (ОС, Программа по работе с БД, антивирусное программное обеспечение и т.д.).

По каждому рабочему месту присутствует информация о содержащемся на рабочем месте оборудовании, которое имеет уникальные номера устройств, присвоенные системными администраторами, и занесённые в реестр имеющихся у организации устройств.

После чего менеджер по обслуживанию клиентов может начинать работать с клиентами. Для выдачи клиенту кредита необходимо сначала зарегистрировать клиента в системе или проверить наличие клиента в системе по его уникальному идентификационному номеру клиента. Далее клиент должен предоставить все необходимые данные менеджеру, а тот в свою очередь внести эти данные в систему, после чего уточнить у клиента какую сумму кредита он желает получить, а также по какой тарифной ставке (из предложенных кредитной организацией). После чего менеджеру система сообщает может ли клиент, пришедший в организацию брать кредиты или нет. В случае если клиент имеет право получения кредита, в системе составляется заявка на кредит, выдаётся клиенту его уникальный идентификационный номер клиента (если такой у него отсутствует) и сообщает клиенту что в личном кабинете можно наблюдать за статусом одобрения заявки.

После всей проделанной работы менеджером по обслуживанию клиентов заявка через систему отправляется кредитному менеджеру, который просматривает всю имеющуюся у него информацию по клиенту и одобряет или отказывает в потребительском кредите клиенту, с объяснением причин в случае отказа.

Клиент, проверяя состояние заявки в личном кабинете может ознакомиться с решением по кредиту, и в случае, если кредит одобрен, клиент может прийти и получить его в организации. В случае отказа клиент может скачать файл в личном кабинете и ознакомится с причинами отказа, либо прийти лично в организацию и уточнить причину отказа.

После получения кредита, клиент в соответствии с составленным в процессе договором и указанными в договоре сроками и условиями обязан погасить кредит.

Таким образом на основе описаний предметной области и процесса выдачи кредита в информационной системе можно выделить следующие основные бизнес-процессы:

1. Процесс авторизации (регистрации) клиента;

2. Процесс подачи заявки на кредит;

3. Процесс удаления пользователя из системы.

Схема бизнес-процесса авторизации (регистрации) клиента указана на рисунке 1.

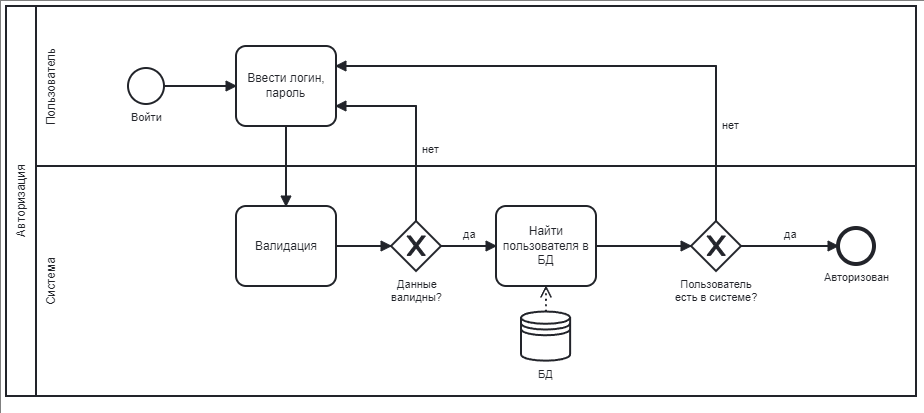


Рисунок 1 – Процесс авторизации (регистрации) клиента

На рисунке 2 предоставлена схема подачи заявки на кредит.

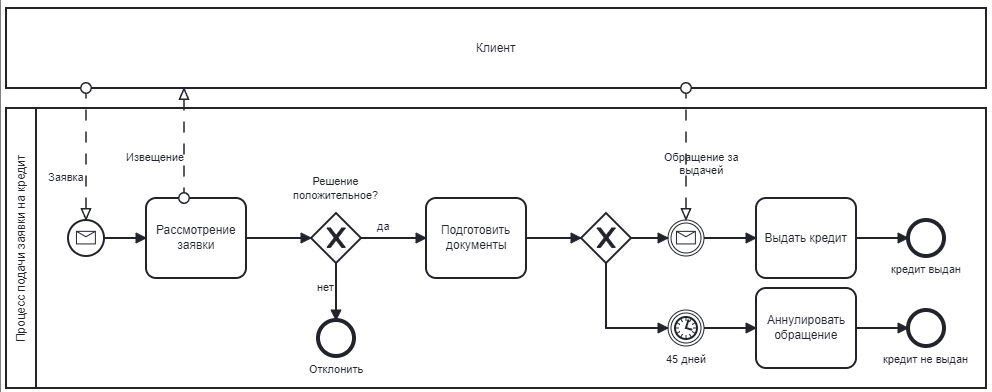


Рисунок 2 – Процесс подачи заявки на кредит

На рисунке 3 предоставлена схема удаления пользователя из системы.

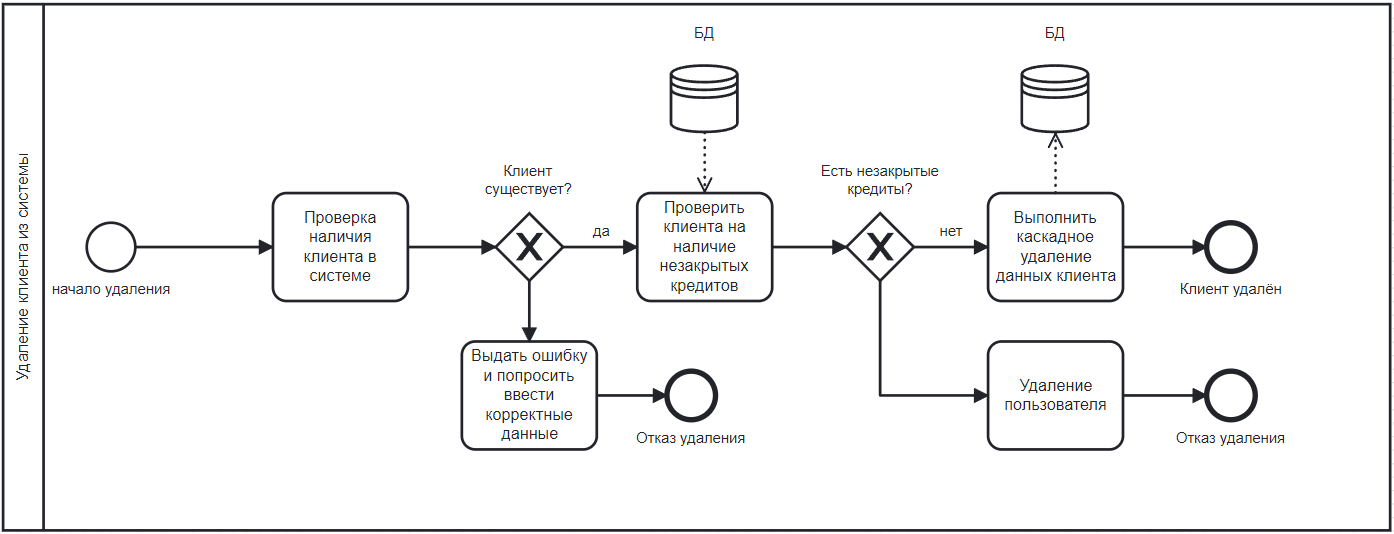


Рисунок 3 – Удаления пользователя из системы

## **1.3 Формирование запросов к базе данных для удовлетворения потребностей бизнес-процессов**

Для удовлетворения информационных потребностей пользователей информационной системы необходимо сформулировать информационные запросы, а также указать какая группа пользователей будет иметь право их выполнять.

Таблица 1 – Запросы

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Группа пользователей** | **Запрос** | **Описание** |
| 1 | Менеджеры по обслуживанию клиентов | Получить список всех зарегистрированных в системе клиентов, имеющих задолженность по кредиту | Нужен для сбора данных о долгах клиента, а так же проверки присутствия клиента в базе. |
| 2 | Менеджеры по обслуживанию клиентов,  Клиент | Проверить статус заявки на кредит для определённого клиента | Нужен для мониторинга состояния заявки. |
| 3 | Кредитные менеджеры | Посмотреть все заявки на кредит ожидающие одобрения | Нужен для сбора данных, для статистики востребованности кредитов. |
| 4 | Кредитные менеджеры,  Менеджеры по обслуживанию клиентов | Проверить историю выдачи кредитов для конкретного клиента | Нужен для решения о выдаче кредита. |
| 5 | Кредитные менеджеры,  Менеджеры по обслуживанию клиентов,  Кредитные аналитики | Получить список всех открытых кредитов | Нужен для получения статистики о деятельности кредитной организации. |
| 6 | Кредитные менеджеры,  Менеджеры по обслуживанию клиентов,  Кредитные аналитики | Посмотреть список отказанных заявок на кредит за определённый период | Для сбора статистики по истории выдачи кредитов. |
| 7 | Кредитные аналитики | Получить общую сумму всех открытых кредитов | Для аналитики доходов и расходов кредитной организации. |
| 8 | Менеджеры по обслуживанию клиентов,  Клиент | Проверить текущий баланс по кредитным счетам клиента | Для получения данных о состоянии счёта и сумме требующийся для закрытия кредита. |

Описание запросов в информационной системе предоставляет фундаментальную информацию о том, как система будет взаимодействовать с данными, какие операции будут выполняться и какие результаты следует ожидать. Ключевое значение имеет ясность и полнота сформулированных запросов для успешной реализации информационной системы.

Правильное изложение запросов способствует эффективному функционированию системы, повышает производительность и удовлетворяет потребности пользователей

# **2. Основные возможности обеспечения безопасности СУБД PostgreSQL**

## **2.1 Краткая характеристика основных возможностей PostgreSQL 16 для обеспечения безопасности**

PostgreSQL 16 является одной из самых популярных и широко используемых систем управления базами данных (СУБД) в мире. Она известна своей надежностью, масштабируемостью и безопасностью. Далее приведен обзор некоторых возможностей PostgreSQL 16 для обеспечения безопасности баз данных в том числе и возможностей, которые доступны в Postgres Pro Enterprise.

1. Аутентификация и авторизация

PostgreSQL 16 предоставляет несколько методов аутентификации пользователей, включая пароль, сертификаты, LDAP и радиус. Кроме того, PostgreSQL поддерживает разграничение прав доступа на уровне таблиц, строк и столбцов, что позволяет администраторам баз данных предоставлять пользователям только те данные, которые они должны видеть.

2. Шифрование данных

PostgreSQL 16 поддерживает шифрование данных как на уровне приложения, так и на уровне базы данных. На уровне приложения можно использовать SSL/TLS для шифрования данных, передаваемых между клиентом и сервером. На уровне базы данных PostgreSQL поддерживает несколько методов шифрования, включая PGP, GPG и OpenSSL.

3. Аудит и мониторинг

PostgreSQL 16 предоставляет несколько инструментов для аудита и мониторинга баз данных, включая pgAudit, pgBadger и pgStatStatements. Эти инструменты позволяют администраторам баз данных отслеживать активность пользователей, выявлять подозрительные действия и оптимизировать производительность баз данных.

4. Безопасность сети

PostgreSQL 16 предоставляет несколько методов для обеспечения безопасности сети, включая межсетевые экраны (файрволы), виртуальные частные сети (VPN) и сетевые прокси-серверы. Кроме того, PostgreSQL поддерживает IP-фильтрацию, которая позволяет администраторам баз данных разрешать или запрещать доступ к базам данных на основе IP-адресов.

5. Защита от угроз

PostgreSQL 16 предоставляет несколько инструментов для защиты от угроз, включая pg\_dump и pg\_restore для резервного копирования и восстановления данных, а также pg\_rewind для восстановления после сбоя. Кроме того, PostgreSQL поддерживает Point-in-Time Recovery (PITR), который позволяет администраторам баз данных восстанавливать базы данных до определенного момента времени.

Но так как Postgres Pro Enterprise доступна только в платном формате, а в бесплатном свободном доступе имеется только PostgreSQL 16 и ранее выпущенные версии, которые имеют ограничение по функциям, было принято к реализации только аутентификация и авторизация пользователей в системе и шифрование данных.

## **2.2 Встроенные механизмы аутентификации и авторизации в PostgreSQL16**

Процесс подключение можно разделить на три этапа:

Идентификация — определение имени роли базы данных.

Аутентификация — проверка того, что пользователь тот, за кого себя выдаёт. Есть много разных методов аутентификации, например проверка пароля.

Авторизация — проверка прав этого пользователя. Например, может ли этот пользователь подключаться к этой базе данных или нет [2].

PostgreSQL предлагает к использованию широкий набор методов аутентификации пользователей:

1. Аутентификация trust, при которой сервер доверяет пользователям, никак не проверяя их. Аутентификация уместна и очень удобна для локальных подключений на однопользовательской рабочей станции. Как правило, она сама по себе не подходит для многопользовательской машины. Тем не менее, можно использовать ее даже на многопользовательской машине, если ограничить доступ к файлу Unix-сокета сервера с помощью разрешений файловой системы [4].
2. Аутентификация password, требующая ввода пароля пользователем. Метод SCRAM-SHA-256 выполняет аутентификацию SCRAM-SHA-256. Это схема «запрос-ответ», которая предотвращает перехват паролей в ненадежных соединениях и поддерживает хранение паролей на сервере в криптографически хешированной форме, которая считается безопасной. Это самый безопасный из предоставляемых в настоящее время методов, но он не поддерживается более старыми клиентскими библиотеками [5].
3. Аутентификация GSSAPI, использующая библиотеку безопасности, совместимую с GSSAPI. Обычно этот метод применяется при использовании специальной службы аутентификации, Kerberos или Microsoft Active Directory. GSSAPI — это отраслевой стандарт для безопасной аутентификации. PostgreSQL поддерживает GSSAPI для аутентификации, шифрования связи или и того, и другого. GSSAPI обеспечивает автоматическую аутентификацию (единый вход) для систем, которые ее поддерживают. Сама аутентификация безопасна. Если используется шифрование GSSAPI или SSL, данные, отправляемые по соединению с базой данных, будут зашифрованы. В противном случае не будет [6].
4. Аутентификация SSPI, использующая протокол, подобный GSSAPI, но предназначенный для Windows. Для безопасной проверки подлинности с единым входом. PostgreSQL будет использовать SSPI в режиме, который будет использовать Kerberos, когда это возможно, и автоматически возвращаться к NTLM в других случаях. SSPI и GSSAPI взаимодействуют как клиенты и серверы, например, клиент SSPI может аутентифицироваться на сервере GSSAPI. Рекомендуется использовать SSPI на клиентах и серверах Windows, а GSSAPI — на платформах, отличных от Windows [7].
5. Аутентификация ident, для которой используется служба, реализующая «Identification Protocol» (RFC 1413) на клиентском компьютере (Для подключений через локальный сокет Unix этот метод работает как peer.). Метод идентификации работает путем получения имени пользователя операционной системы клиента с сервера идентификации и использования его в качестве разрешенного имени пользователя базы данных (с необязательным сопоставлением имен пользователей). Поддерживается только в соединениях TCP/IP [8].
6. Аутентификация peer, которая полагается на средства операционной системы, позволяющие узнать пользователя процесса на другой стороне локального подключения. Метод одноранговой аутентификации работает путем получения имени пользователя операционной системы клиента из ядра и использования его в качестве разрешенного имени пользователя базы данных (с необязательным сопоставлением имен пользователей). Этот метод поддерживается только для локальных подключений. Для удалённых подключений она не поддерживается [9].
7. Аутентификация LDAP, работающая с сервером аутентификации LDAP. Этот метод аутентификации использует LDAP в качестве метода проверки пароля. LDAP используется только для проверки пар имя пользователя/пароль. Таким образом, пользователь уже должен существовать в базе данных, прежде чем LDAP можно будет использовать для аутентификации [10].
8. Аутентификация RADIUS, работающая с сервером аутентификации RADIUS. Этот метод проверки подлинности работает аналогично LDAP, за исключением того, что в качестве метода проверки пароля используется RADIUS. RADIUS используется только для проверки пар имя пользователя/пароль [11].
9. Аутентификация по сертификату, требующая использования клиентами SSL-подключения и построенная на проверке передаваемых ими сертификатов SSL [12].
10. Аутентификация PAM, реализуемая с использованием библиотеки PAM. Этот метод аутентификации работает аналогично LDAP, за исключением того, что он использует PAM (подключаемые модули аутентификации) в качестве механизма аутентификации [13].
11. Аутентификация BSD, основанная на использовании механизма аутентификации BSD (в настоящее время поддерживается только в системе OpenBSD). Этот метод аутентификации работает аналогично LDAP, за исключением того, что он использует BSD Authentication для проверки пароля [14].

Для локальных подключений обычно рекомендуется использовать метод peer, хотя в некоторых обстоятельствах может быть достаточно и режима trust. Для удалённых подключений самой простой будет аутентификация по паролю. Все остальные варианты требуют использования некоторой внешней инфраструктуры безопасности (обычно это служба аутентификации или центр сертификации, выдающий сертификаты SSL) либо поддерживаются не на всех платформах [3].

В данном случае аутентификация по паролю — самый простой выбор для удаленных подключений. Все остальные варианты требуют какой-либо внешней инфраструктуры безопасности (обычно сервера аутентификации или центра сертификации для выпуска SSL-сертификатов) или зависят от платформы [15].

## **2.3 Изучение методов шифрования данных в PostgreSQL 16**

PostgreSQL предлагает шифрование на нескольких уровнях и обеспечивает гибкость в защите данных от разглашения из-за кражи сервера баз данных, недобросовестных администраторов и небезопасных сетей. Шифрование также может потребоваться для защиты конфиденциальных данных, таких как медицинские записи или финансовые транзакции.

1. Шифрование паролем – пароли пользователей базы данных хранятся в виде хэшей (определяется параметром password\_encryption), поэтому администратор не может определить фактический пароль, назначенный пользователю. Если для аутентификации клиента используется шифрование SCRAM или MD5, незашифрованный пароль никогда даже временно не присутствует на сервере, поскольку клиент шифрует его перед отправкой по сети. Предпочтительнее использовать SCRAM, поскольку это стандарт интернета, и он более безопасен, чем специфичный для PostgreSQL протокол аутентификации MD5.
2. Шифрование для определенных столбцов – модуль pgcrypto позволяет хранить определенные поля в зашифрованном виде. Это полезно, если конфиденциальными являются только некоторые данные. Клиент предоставляет ключ расшифровки, и данные расшифровываются на сервере, а затем отправляются клиенту. Расшифрованные данные и ключ дешифрования присутствуют на сервере в течение короткого времени, пока они расшифровываются и передаются между клиентом и сервером. Здесь представлен краткий момент, когда данные и ключи могут быть перехвачены кем-либо, имеющим полный доступ к серверу базы данных, например системным администратором.
3. Шифрование разделов данных – шифрование хранилища может выполняться на уровне файловой системы или на уровне блоков. Параметры шифрования файловой системы Linux включают eCryptfs и EncFS, в то время как FreeBSD использует PEFS. Варианты шифрования на уровне блоков или всего диска включают dm-crypt + LUKS в Linux и модули GEOM geli и gbde во FreeBSD. Многие другие операционные системы поддерживают эту функциональность, включая Windows. Этот механизм предотвращает чтение незашифрованных данных с дисков в случае кражи дисков или всего компьютера. Это не защищает от атак во время монтирования файловой системы, поскольку при монтировании операционная система предоставляет незашифрованный вид данных. Однако для монтирования файловой системы вам нужен какой-то способ передачи ключа шифрования операционной системе, и иногда ключ хранится где-то на хосте, который монтирует диск.
4. Шифрование данных по сети – SSL-соединения шифруют все данные, отправляемые по сети: пароль, запросы и возвращаемые данные. pg\_hba.conf Файл позволяет администраторам указывать, какие хосты могут использовать незашифрованные соединения (host), а для каких требуются соединения с SSL-шифрованием (hostssl). Кроме того, клиенты могут указать, что они подключаются к серверам только через SSL. Соединения с GSSAPI-шифрованием шифруют все данные, отправляемые по сети, включая запросы и возвращаемые данные. (Пароль по сети не передается.) pg\_hba.conf Файл позволяет администраторам указывать, какие хосты могут использовать незашифрованные соединения (host), а для каких требуются соединения, зашифрованные по GSSAPI (hostgssenc). Кроме того, клиенты могут указать, что они подключаются к серверам только по соединениям, зашифрованным с помощью GSSAPI (gssencmode=require). Для шифрования передач также можно использовать Stunnel или SSH.
5. Аутентификация хоста по протоколу SSL – как клиент, так и сервер могут предоставлять SSL-сертификаты друг другу. Это требует некоторой дополнительной настройки с каждой стороны, но это обеспечивает более надежную проверку личности, чем простое использование паролей. Это не позволяет компьютеру притворяться сервером достаточно долго, чтобы прочитать пароль, отправленный клиентом. Это также помогает предотвратить атаки по принципу “человек посередине”, когда компьютер между клиентом и сервером выдает себя за сервер, считывает и передает все данные между клиентом и сервером.
6. Шифрование на стороне клиента – если системному администратору компьютера сервера нельзя доверять, клиенту необходимо зашифровать данные; таким образом, незашифрованные данные никогда не появляются на сервере базы данных. Данные шифруются на клиенте перед отправкой на сервер, и результаты базы данных должны быть расшифрованы на клиенте перед использованием [17].

В случае с данной ИС требуется шифровать пароли и шифровать некоторую информацию, связанную с клиентами, для этого подходят такие методы шифрования как:

1. Шифрование паролей.
2. Шифрование для определенных столбцов.

## **2.4 Изучение политики RLC в PostgreSQL 16**

Безопасность на уровне строк, также известная как "RLS", позволяет администратору базы данных определять, должен ли пользователь иметь возможность просматривать определенные строки данных в таблице или манипулировать ими в соответствии с политикой [16].

По умолчанию таблицы не имеют никаких политик, так что, если пользователь имеет права доступа к таблице в соответствии с системой привилегий SQL, все строки в ней одинаково доступны для запроса или обновления.

Когда в таблице включена защита строк (c помощью ALTER TABLE ... ENABLE ROW LEVEL SECURITY) весь обычный доступ к таблице для выбора строк или изменения строк должен быть разрешен политикой безопасности строк. (Однако на владельца таблицы обычно не распространяются политики безопасности строк.) Если для таблицы не существует политики запрета по умолчанию, используется политика, означающая, что строки не видны и не могут быть изменены. Операции, которые применяются ко всей таблице, такие как TRUNCATE и REFERENCES, не подлежат защите строк.

Политики безопасности строк могут быть специфичными для команд, или для ролей, или для того и другого. Политика может быть указана для применения ко всем командам, или к SELECT, INSERT, UPDATE, или DELETE в частности. Для данной политики может быть назначено несколько ролей, при этом применяются обычные правила членства в ролях и наследования.

Чтобы указать, какие строки являются видимыми или могут быть изменены в соответствии с политикой, требуется выражение, возвращающее логический результат. Это выражение будет вычисляться для каждой строки до выполнения любых условий или функций, поступающих из запроса пользователя. (Единственными исключениями из этого правила являются leakproof функции, которые гарантированно не допускают утечки информации; оптимизатор может применить такие функции перед проверкой безопасности строк.) Строки, для которых выражение не возвращает true обработка не будет выполняться. Могут быть указаны отдельные выражения для обеспечения независимого контроля над строками, которые видны, и строками, которые разрешено изменять. Выражения политики выполняются как часть запроса и с правами пользователя, выполняющего запрос, хотя функции определения безопасности могут использоваться для доступа к данным, недоступным вызывающему пользователю.

Суперпользователи и роли с атрибутом BYPASSRLS всегда обходят систему безопасности строк при доступе к таблице. Владельцы таблиц обычно также обходят систему безопасности строк, хотя владелец таблицы может выбрать режим защиты строк с помощью ALTER TABLE ... FORCE ROW LEVEL SECURITY.

Включение и отключение безопасности строк, а также добавление политик в таблицу всегда является привилегией только владельца таблицы.

Политики создаются с помощью команды CREATE POLICY , изменяются с помощью команды ALTER POLICY и удаляются с помощью команды DROP POLICY . Чтобы включить или отключить защиту строк для данной таблицы, используется команда ALTER TABLE [19].

Каждая политика имеет имя, и для таблицы может быть определено несколько политик. Поскольку политики зависят от таблицы, каждая политика для таблицы должна иметь уникальное имя. Разные таблицы могут иметь политики с одинаковым именем.

Когда к запросу применяется несколько политик, они объединяются с использованием либо OR (для разрешительных политик, которые используются по умолчанию), либо с использованием AND (для ограничительных политик). Это похоже на правило, согласно которому данная роль обладает привилегиями всех ролей, членом которых она является [18].

# **3. Реализация ИС «Потребительское кредитование» и обеспечение безопасности.**

## **3.1 Проектирование базы данных ИС «Потребительское кредитование»**

### **3.1.1 Логическое проектирование в ИС «Потребительское кредитование»**

Используя данные, полученные ранее в ходе анализа предметной области, была спроектирована следующая структура базы данных, представленная на рисунке 5.

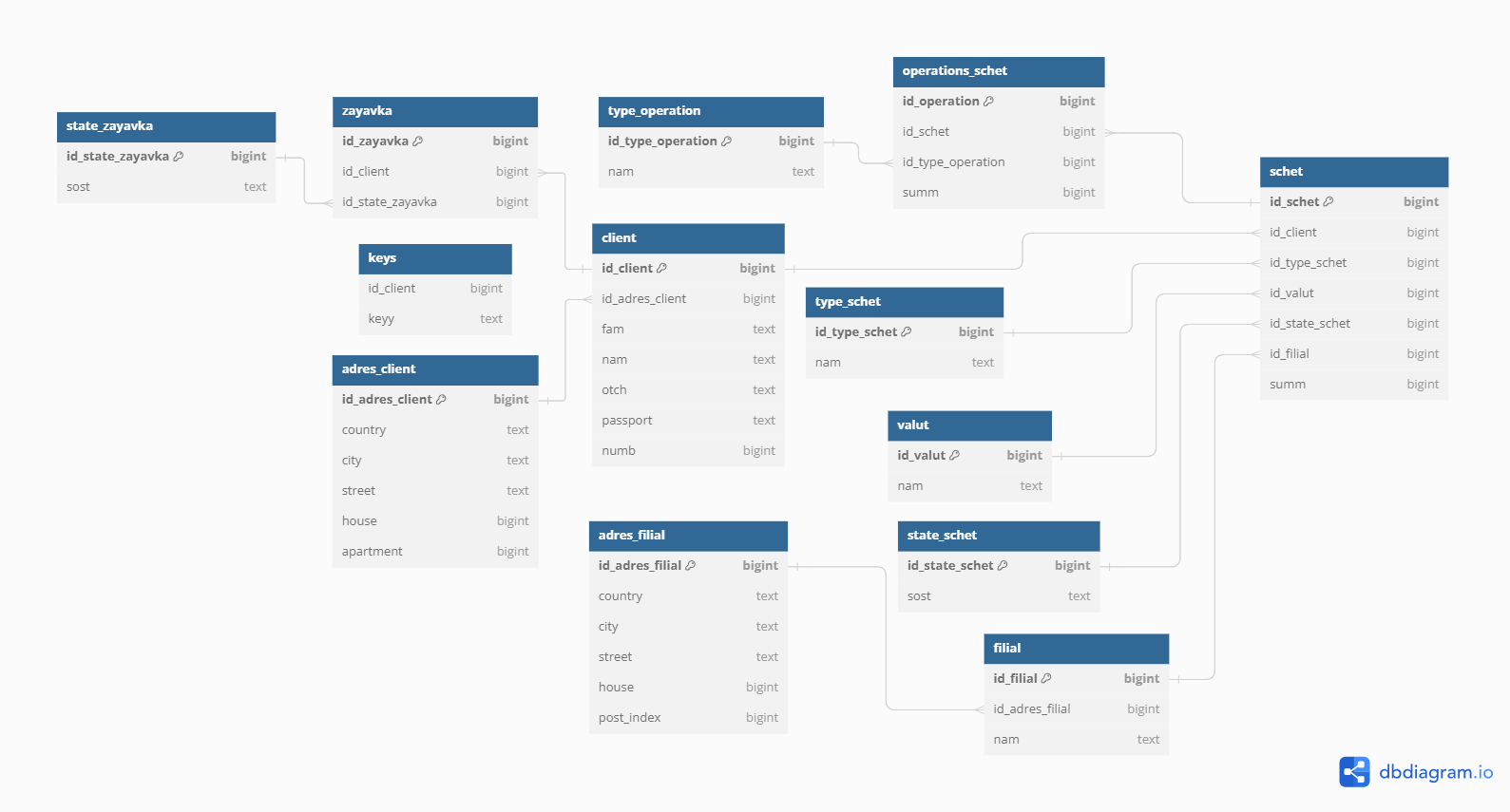


Рисунок 5 – Схема БД ИС «Потребительское кредитование»

Схема включает в себя такие наименования таблиц, которые представлены в таблице 2, а также в данной таблице продемонстрированы атрибуты таблиц системы.

Таблица 2 – Наименование и атрибуты таблиц

|  |  |  |
| --- | --- | --- |
| № | Наименование | Атрибуты и типы данных |
| 1 | keys | id\_client bigint  keyy text |
| 2 | adres\_filial | id\_adres\_filial bigint [primary key]  country text  city text  street text  house bigint  post\_index bigint |
| 3 | type\_operation | id\_type\_operation bigint [primary key]  nam text |
| 4 | type\_schet | id\_type\_schet bigint [primary key]  nam text |
| 5 | state\_schet | id\_state\_schet bigint [primary key]  sost text |
| 6 | valut | id\_valut bigint [primary key]  nam text |
| 7 | state\_zayavka | id\_state\_zayavka bigint [primary key]  sost text |
| 8 | adres\_client | id\_adres\_client bigint [primary key]  country text  city text  street text  house bigint  apartment bigint |
| 9 | client | id\_client bigint [primary key]  id\_adres\_client bigint [ref: > adres\_client.id\_adres\_client]  fam text  nam text  otch text  passport text  numb bigint |
| 10 | zayavka | id\_zayavka bigint [primary key]  id\_client bigint [ref: > client.id\_client]  id\_state\_zayavka bigint [ref: > state\_zayavka.id\_state\_zayavka] |
| 11 | filial | id\_filial bigint [primary key]  id\_adres\_filial bigint [ref: > adres\_filial.id\_adres\_filial]  nam text |
| 12 | schet | id\_schet bigint [primary key]  id\_client bigint [ref: > client.id\_client]  id\_type\_schet bigint [ref: > type\_schet.id\_type\_schet]  id\_valut bigint [ref: > valut.id\_valut]  id\_state\_schet bigint [ref: > state\_schet.id\_state\_schet]  id\_filial bigint [ref: > filial.id\_filial]  summ bigint |
| 13 | operations\_schet | id\_operation bigint [primary key]  id\_schet bigint [ref: > schet.id\_schet]  id\_type\_operation bigint [ref: > type\_operation.id\_type\_operation]  summ bigint |

Далее описано кратко содержание таблиц, какие данные хранятся в таблицах и для чего эти таблицы нужны:

* + - 1. Таблица keys – в данной таблицы хранятся уникальные ключи клиентов, для того чтобы никто кроме самих клиентов не могли просматривать конфиденциальную информацию о себе.
      2. Таблица adres\_filial – здесь хранится информация о адресах в которых расположены филиалы.
      3. Таблица type\_operation – здесь хранится информация о типе операций, совершаемых со счётом.
      4. Таблица type\_schet – здесь хранится информация о типах счетов, которые существуют в системе.
      5. Таблица state\_schet – здесь хранится состояние счёта (открыт либо закрыт).
      6. Таблица valut – здесь хранится информация о типе валют, в которых может быть открыт счёт.
      7. Таблица state\_zayavka – здесь хранится информация о состоянии рассмотрения заявки.
      8. Таблица adres\_client – здесь хранится информация о адресе проживания клиента.
      9. Таблица client – здесь хранится общая информация о клиенте, в том числе фамилия, имя, отчество, паспорт, и номер телефона клиента
      10. Таблица autorization – в данной таблице хранятся логины и пароли клиентов, которые зарегистрированы в системе.
      11. Таблица zayavka – здесь хранятся все поданные клиентами заявки.
      12. Таблица filial – здесь хранится информация о филиалах.
      13. Таблица schet – здесь хранится информация о счетах, принадлежащих клиенту, а точнее уникальны идентификаторы счёта, клиента которому этот счёт принадлежит, типа счёта, типа валюты, состояния счёта, филиала в котором был открыт счёт, и начальную сумму счёта.
      14. Таблица operations\_schet – в данную таблицу заносятся последние действия со счетами.

### **3.1.2 Физическое проектирование в ИС «Потребительское кредитование» средствами СУБД**

При реализации БД в PostgreSQL необходимо сначала составить таблицы, для чего были использованы команды для создания таблиц на языке SQL.

Для создания таблицы «keys» была использована команда, представленная на рисунке 6.

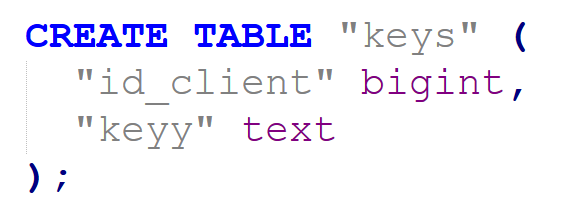


Рисунок 6 – Команда создания таблицы «keys»

Для создания таблицы «adres\_filial» была использована команда, представленная на рисунке 7.

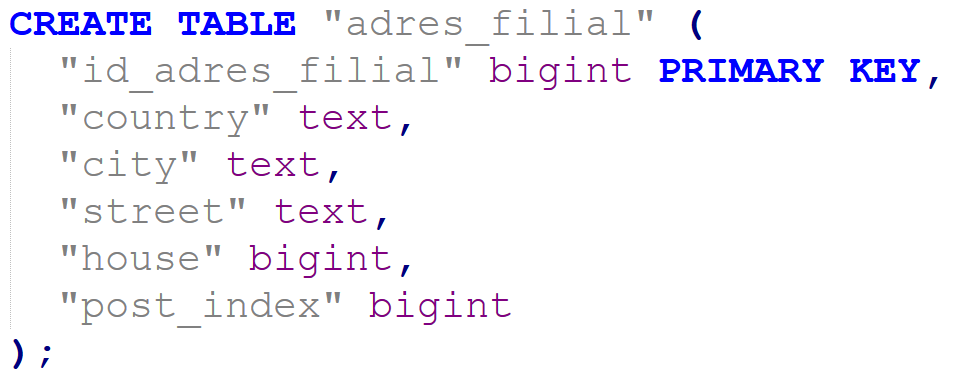


Рисунок 7 – Команда создания таблицы «adres\_filial»

Для создания таблицы «type\_operation» была использована команда, представленная на рисунке 8.

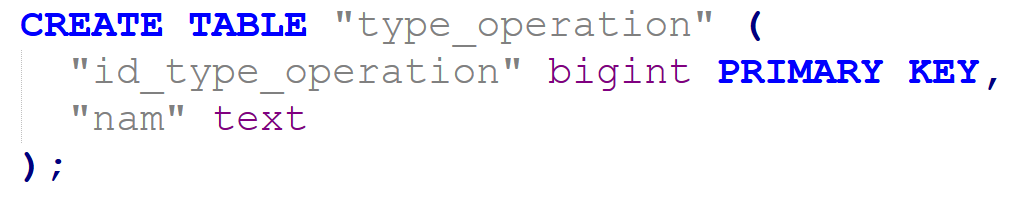


Рисунок 8 – Команда создания таблицы «type\_operation»

Для создания таблицы «type\_schet» была использована команда, представленная на рисунке 9.

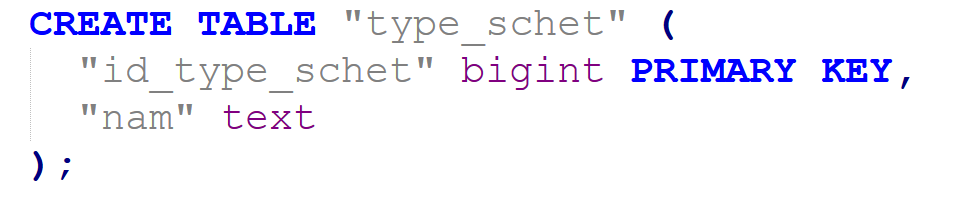


Рисунок 9 – Команда создания таблицы «type\_schet»

Для создания таблицы «state\_schet» была использована команда, представленная на рисунке 11.

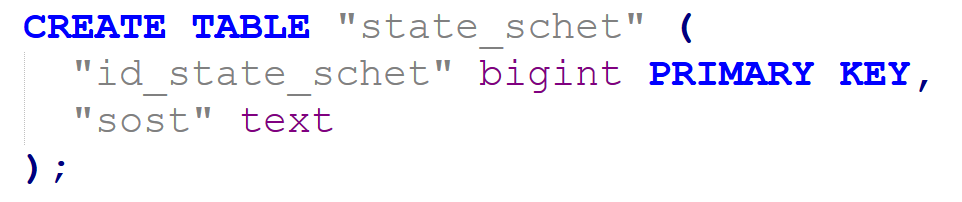


Рисунок 10 – Команда создания таблицы «state\_schet»

Для создания таблицы «valut» была использована команда, представленная на рисунке 11.

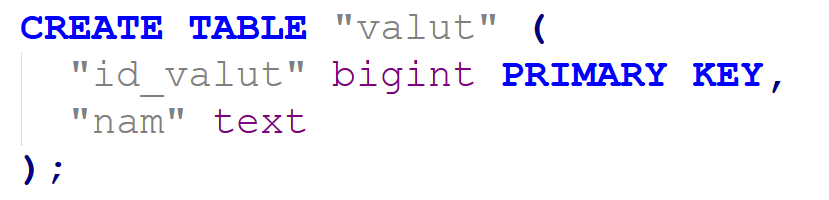


Рисунок 11 – Команда создания таблицы «valut»

Для создания таблицы «state\_zayavka» была использована команда, представленная на рисунке 12.

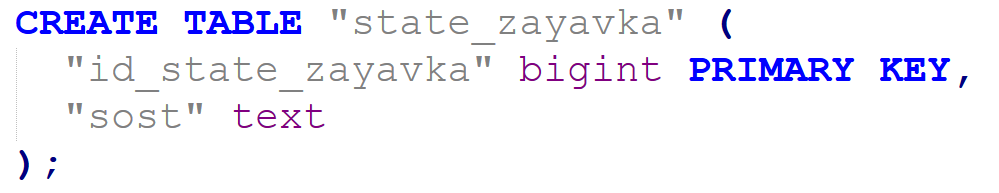


Рисунок 12 – Команда создания таблицы «state\_zayavka»

Для создания таблицы «adres\_client» была использована команда, представленная на рисунке 13.

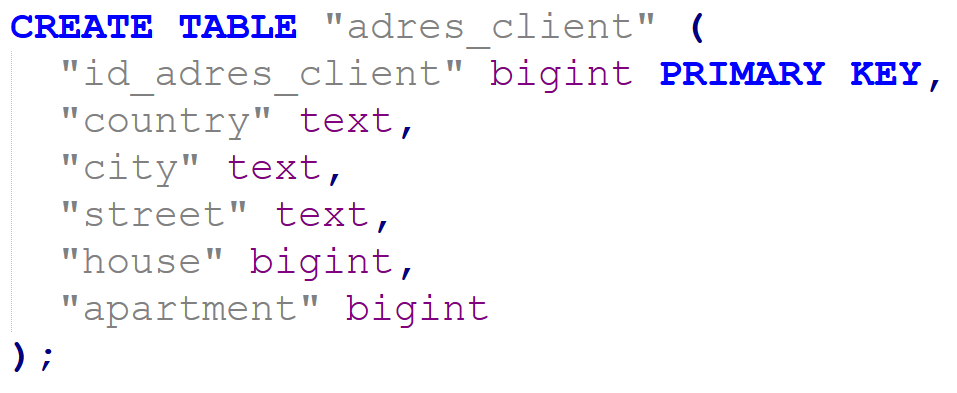


Рисунок 13 – Команда создания таблицы «adres\_client»

Для создания таблицы «client» была использована команда, представленная на рисунке 14.

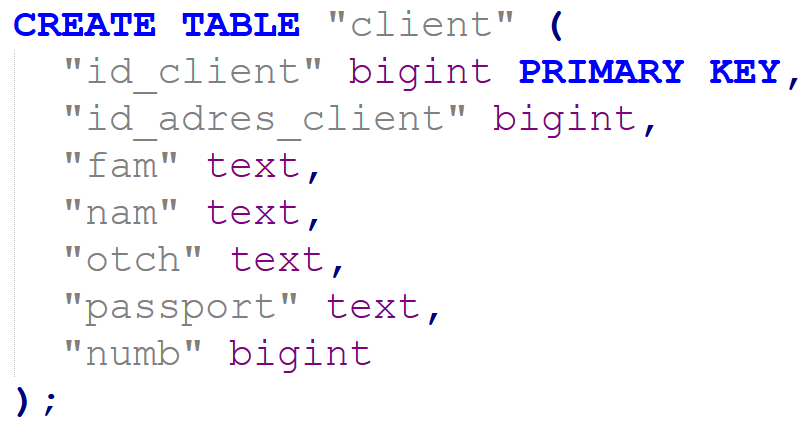


Рисунок 14 – Команда создания таблицы «client»

Для создания таблицы «zayavka» была использована команда, представленная на рисунке 15.

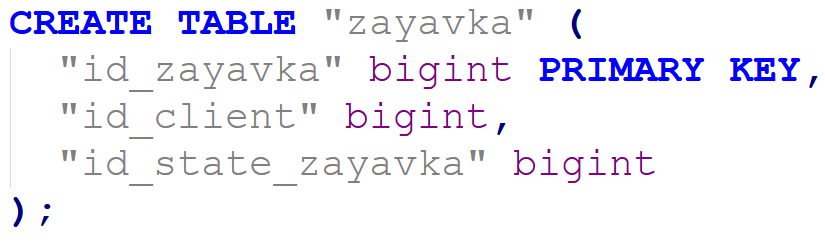


Рисунок 15 – Команда создания таблицы «zayavka»

Для создания таблицы «filial» была использована команда, представленная на рисунке 16.

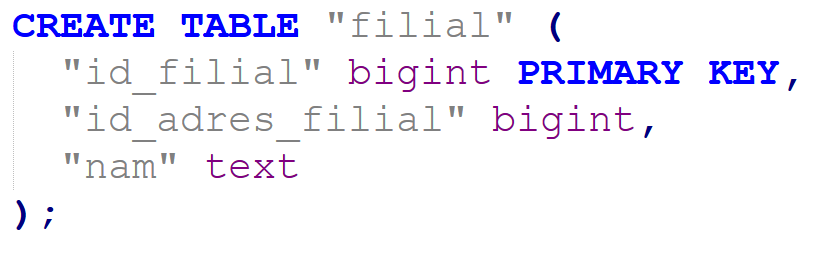


Рисунок 16 – Команда создания таблицы «filial»

Для создания таблицы «schet» была использована команда, представленная на рисунке 17.

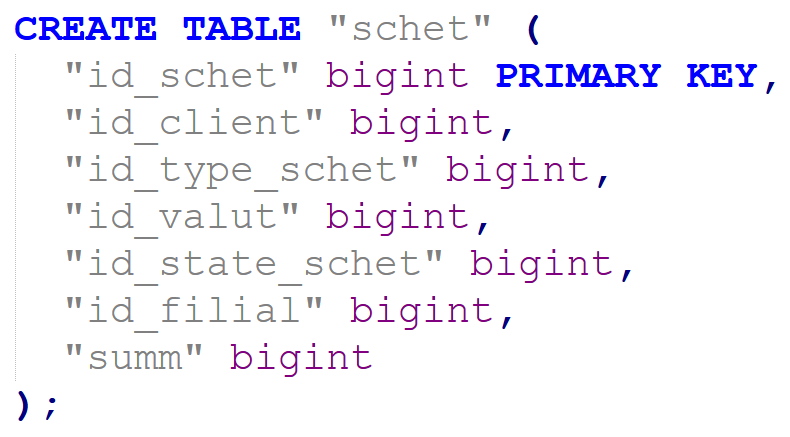


Рисунок 17 – Команда создания таблицы «schet»

Для создания таблицы «operations\_schet» была использована команда, представленная на рисунке 18.

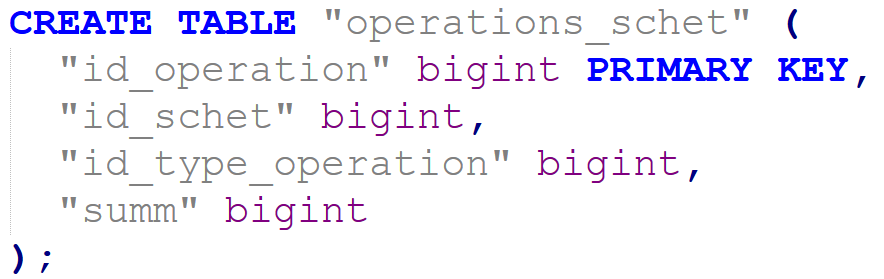


Рисунок 18 – Команда создания таблицы «operations\_schet»

Так же после создания таблиц отдельно были реализованы связи таблиц как показано на рисунке 19.

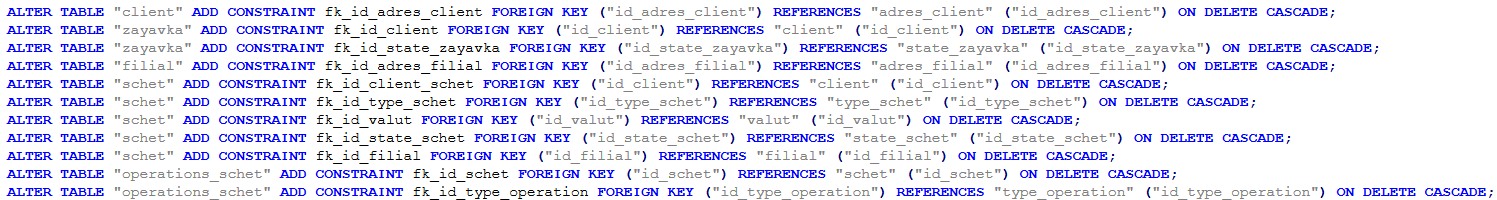


Рисунок 19 – Связи таблиц

После реализации таблиц необходимо реализовать разграничение ролевого доступа к ним, для этого есть необходимость создать роли, советующие выделенным группам пользователей.

## **3.2 Реализация отдельных механизмов защиты ИС «Потребительского кредитования»**

### **3.2.1 Реализация ролей в ИС «Потребительское кредитование» средствами СУБД**

Для разграничения ролей, было создано 3 роли, советующие 3 группам пользователей:

1. Пользователи

2. Менеджеры

3. Администраторы

Создание роли «cliet» и добавления в нее пользователей представлено на рисунке

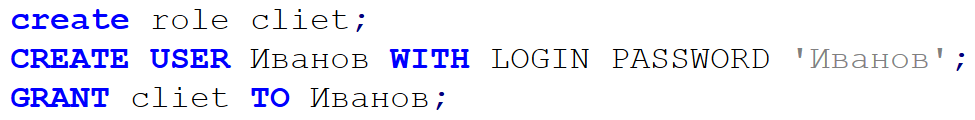


Рисунок – 20 Создание роли «cliet»

Создание роли «administrator» и добавления в нее пользователей представлено на рисунке

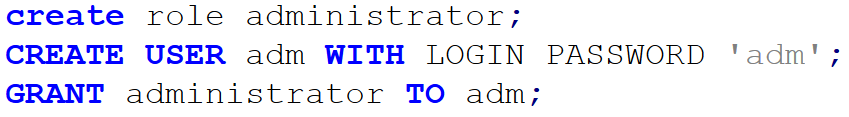


Рисунок – 21 Создание роли «administrator»

Создание роли «menedjer» и добавления в нее пользователей представлено на рисунке

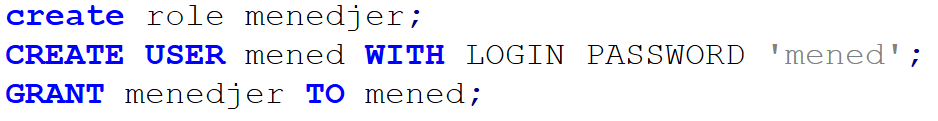


Рисунок – 22 Создание роли «menedjer»

Таким образом были созданы роли для разных групп пользователей, для дальнейшего разграничения доступа

### **3.2.2 Реализация RLC и разграничение доступа в ИС «Потребительское кредитование» средствами СУБД**

Для реализации RLC и разграничения доступа были сделаны 3 политики и выданы права на таблицы для ролей.

На рисунке 23 представлена политика для таблицы «client»

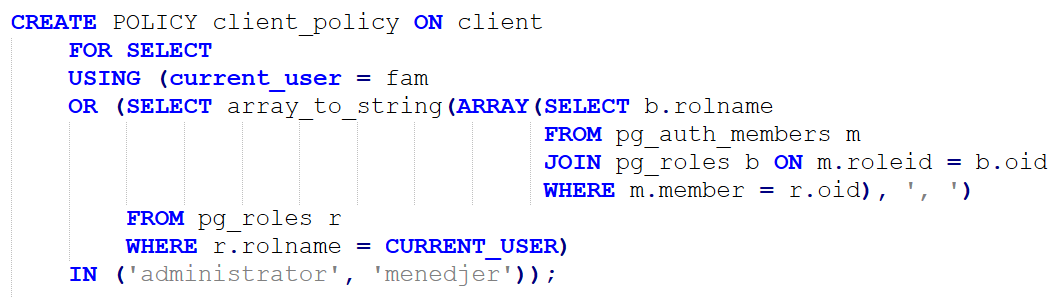


Рисунок 23 – Политика для таблицы «client»

На рисунке 24 представлена политика для таблицы «schet»

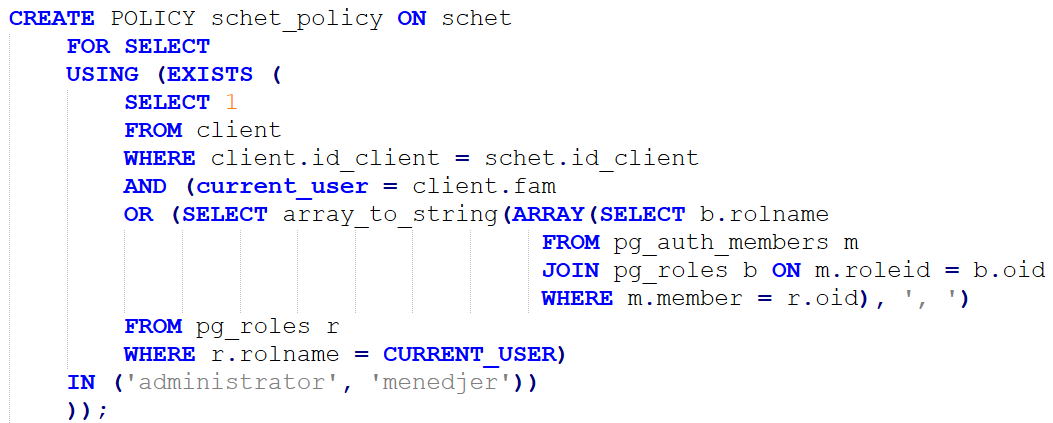


Рисунок 24 – Политика для таблицы «schet»

На рисунке 25 представлена политика для таблицы «operations\_schet»

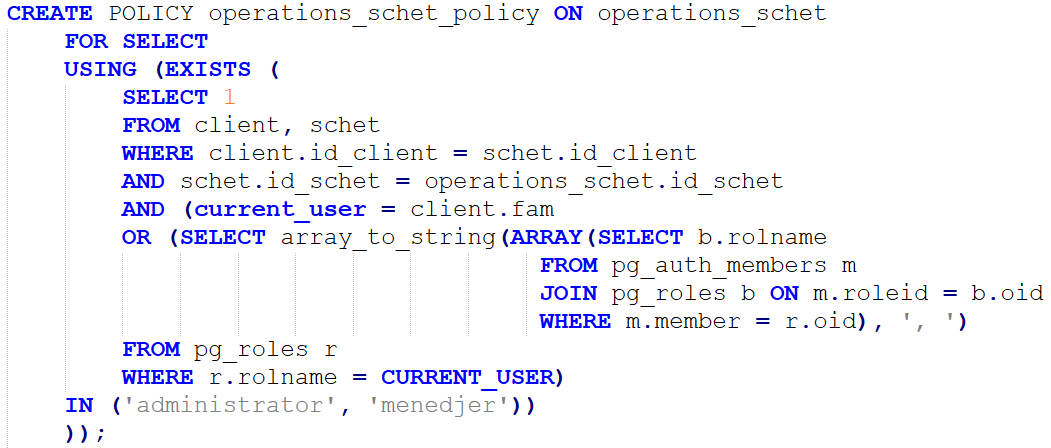


Рисунок 25 – Политика для таблицы «operations\_schet»

На рисунке 26 представлено включение и применений RLS политик.

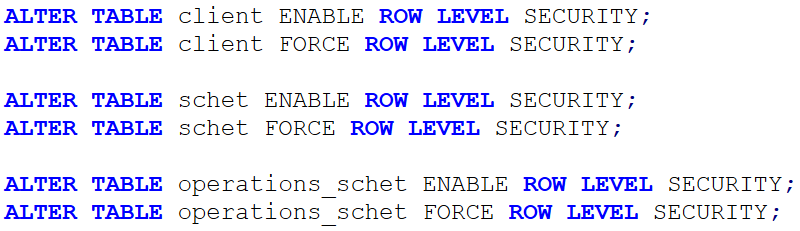


Рисунок 26 – Применение политик RLC

На рисунке 27 представлена выдача прав для ролей.

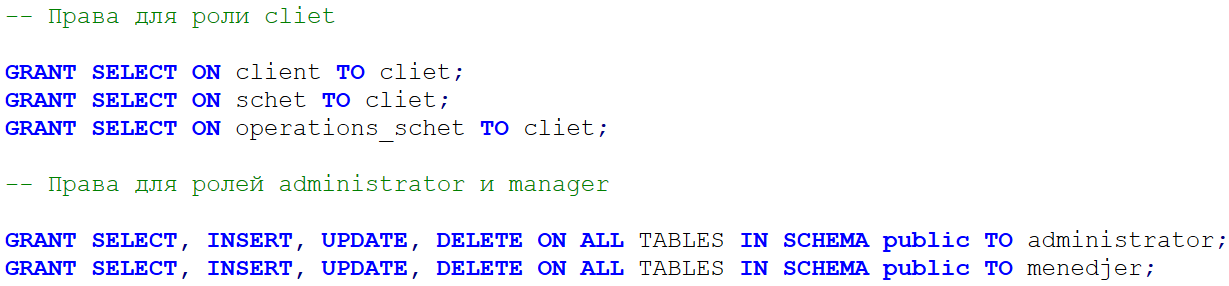


Рисунок 27 – Выдача прав для ролей

Таким образом было произведено ограничение доступа к таблицам, для пользователей.

### **3.2.3 Реализация триггеров и шифрования в ИС «Потребительское кредитование» средствами СУБД**

Теперь необходимо реализовать несколько триггеров для безопасности и в некоторых моментах автономной работы СУБД.

Первым был реализован «Триггер для автоматического обновления суммы счёта при внесении операций» он представлен на рисунке 28.

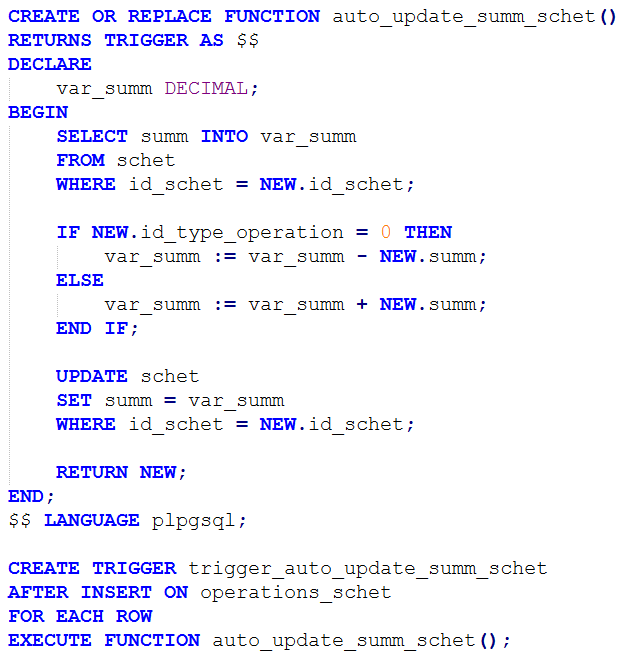


Рисунок 28 – Триггер для автоматического обновления суммы счёта при внесении операций

Следующим был реализован «Триггер для предотвращения удаления клиента с активными счетами» он представлен на рисунке 29.

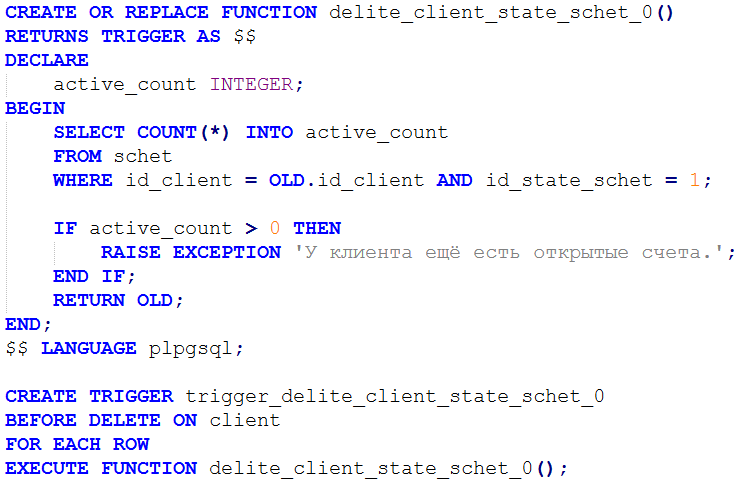


Рисунок 29 – Триггер для предотвращения удаления клиента с активными счетами

Далее идет «Триггер для шифрования паспорта с генерацией ключа, и записью этого ключа в таблицу keys» он представлен на рисунке 30.

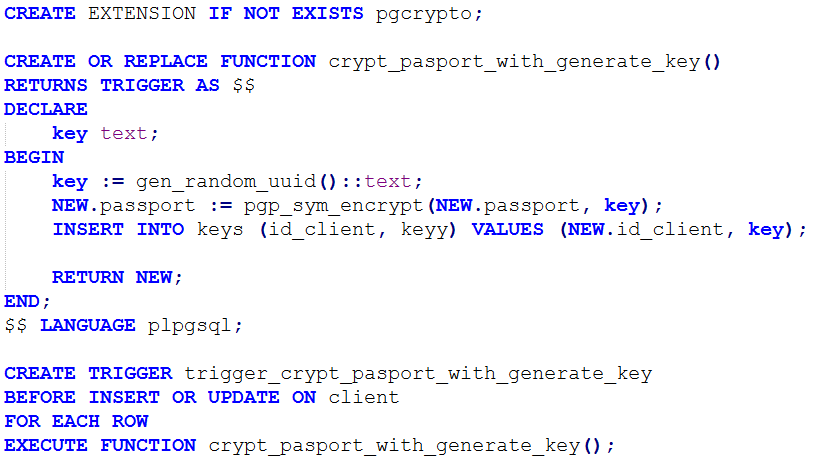


Рисунок 30 – Триггер для шифрования паспорта с генерацией ключа, и записью этого ключа в таблицу keys

Так же для него была реализована «Функция для расшифрования шифрования паспорта» представлена на рисунке 31.

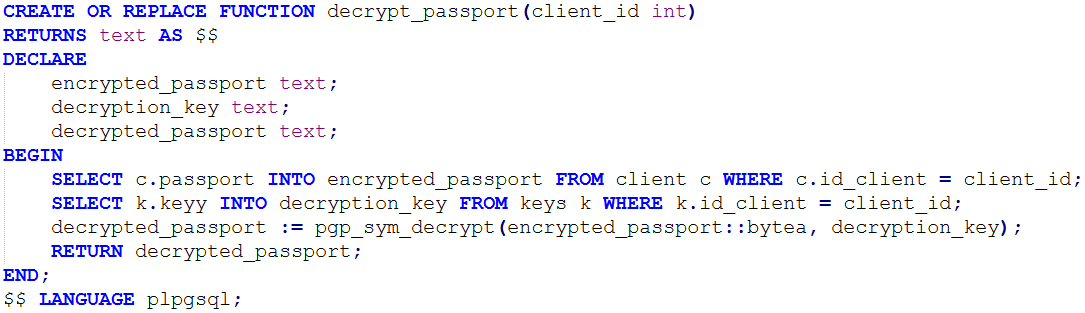


Рисунок 31 – Функция для расшифрования шифрования паспорта

Следующим был реализован «Триггер для предотвращения отрицательного баланса на счете» но представлен на рисунке 32.

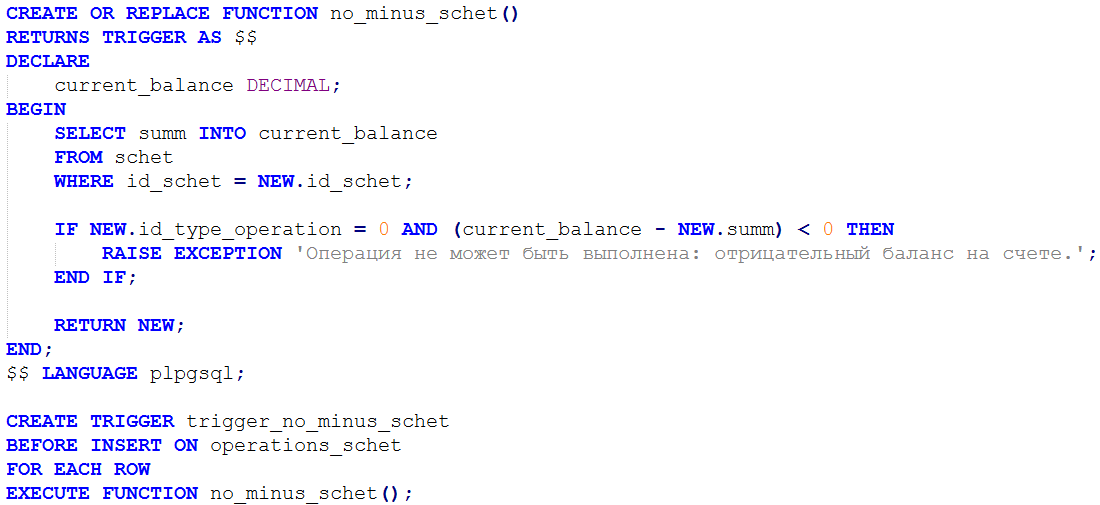


Рисунок 32 – Триггер для предотвращения отрицательного баланса на счете

Потом был реализован «Триггер для проверки существования клиента при добавлении нового счета» но представлен на рисунке 33.

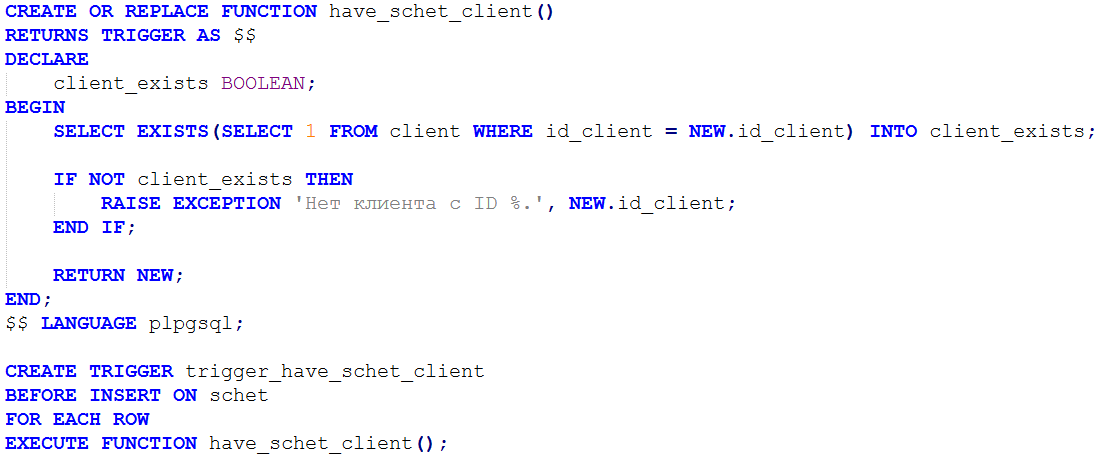


Рисунок 33 – Триггер для проверки существования клиента при добавлении нового счета

Пред последним был реализован «Триггер для предотвращения вставки операции на закрытом счете» он представлен на рисунке 34.

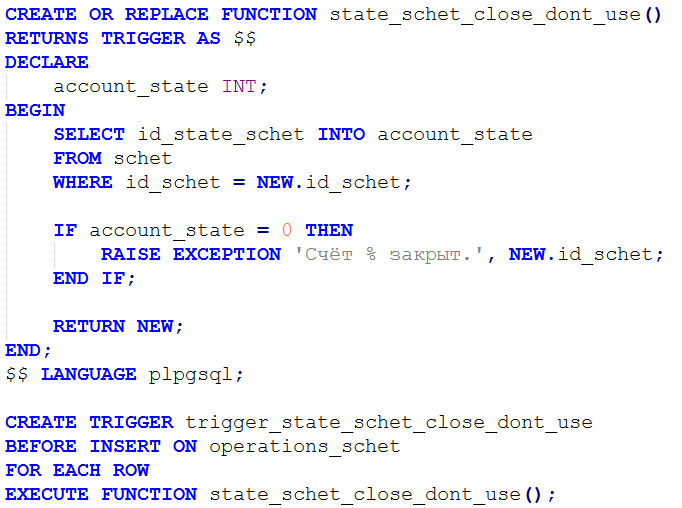


Рисунок 34 – Триггер для предотвращения вставки операции на закрытом счете

И последним был реализован «Триггер для ограничения количества счетов у одного клиента» с ограничением в количестве счетов на клиента равным трем, он представлен на рисунке 35.

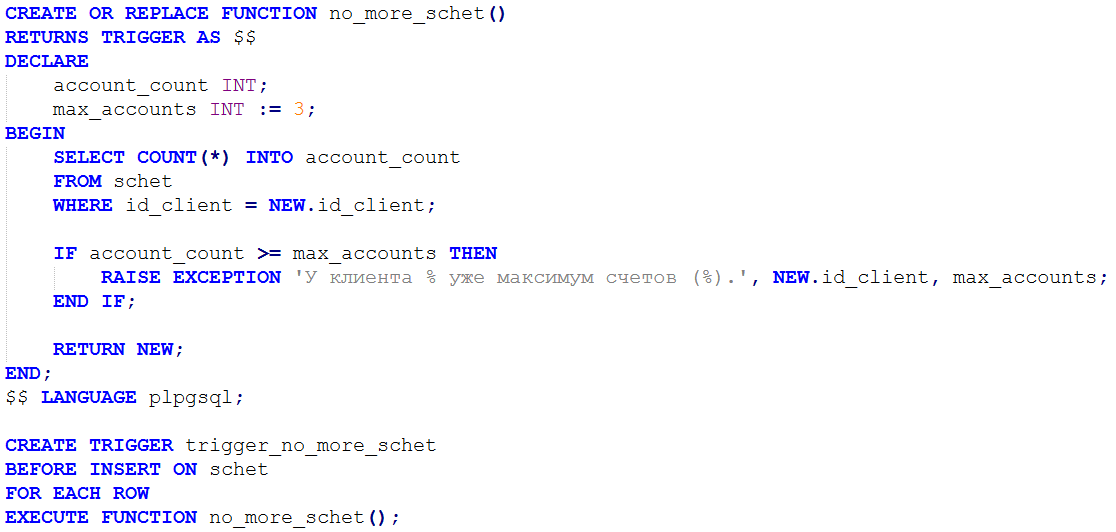


Рисунок 35 – Триггер для ограничения количества счетов у одного клиента

Таким образом была реализована целостность данных и шифрование в БД.

## **3.3 Реализация пользовательского интерфейса для ИС «Потребительское кредитование» средствами Python и PyQt6**

Реализация функций в Python производилась с использованием PyQt6 для графического представления, psycopg2 для подключения к базе данных PostgreSQL.

При запуске программы первым появляется окно авторизации как показано на рисунке 36.

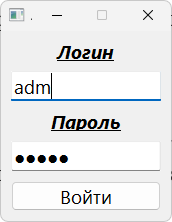


Рисунок 36 – Окно авторизации

При реализации данного окна использовались следующие виджеты:

* QLineEdit;
* QLabel;
* QPushButton;

В данном случае QLineEdit получает значения от пользователя при этом в поле пароля происходит скрытие пароля в целях безопасности, чтобы нельзя было скопировать пароль. QLabel в данном случае нужен просто для отображения надписей логин и пароль. А QPushButton используется для активации функции login которая производит авторизацию в системе Postrges.

После авторизации в зависимости от того, кто авторизовался может появится один из трёх вариантов окна, показанных на рисунке 37.

|  |  |
| --- | --- |
|  |  |
| а) | б) |
|  | |
| в) | |

Рисунок 37 – Окна пользователей: а) Клиент, б) Администратор, в) Менеджер.

При реализации данных окон использовались следующие виджеты:

* QLabel;
* QPushButton;
* QComboBox.

В данном случае QLabel нужен для нужен просто для отображения надписей. QComboBox используется в окнах администратора и менеджера, в случае менеджера это выбор таблицы с которой будет производится работа и выбор стандартного запроса, доступного менеджеру, в случае администратора используется для быстрого и удобного отображения таблиц, т.к. администратор в основном работает через терминал, а как известно терминал это не графический интерфейс, то отображение таблиц в терминале происходит не очень удобно.

QPushButton выполняет роль активатора функций для выполнения определённых действий, например, у пользователя кнопки вызывают QDialog в котором отображается таблицы с его личной информацией, у администратора, открывает окно терминала и так же использует QDialog, для отображения таблиц. В случае с добавлением удалением и обновлением данных QPushButton вызывает другие два вида окна.

При добавлении вызывается окно, которое показано на рисунке 38.

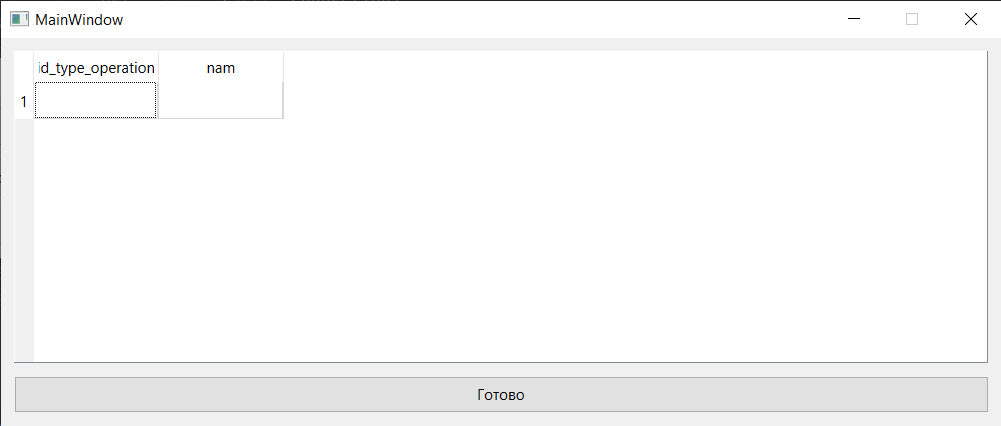


Рисунок 38 – Окно добавления данных.

В окне обновления данных отображается таблица, в которую менеджер собирается добавлять данные и кнопка готова по нажатию на которую происходит добавление данных

При обновлении данных показывается окно как на рисунке 39.



Рисунок 39 – Окно редактирование данных

В окне обновления данных отображается таблица, в которой менеджер может редактировать данные в таблицах.

При удалении данных открывается окно как показано на рисунке 40.

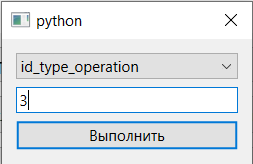


Рисунок 40 – Окно удаления данных

В окне удаления выбирается элемент, по которому требуется удалить данные, и вводится значение, которое хотят удалить.

# **4. Документация на ИС «Потребительское кредитование»**

## **4.1 Руководство по использованию ИС «Потребительское кредитование» для пользователя**

Для начала работы в программе пользователю требуется авторизоваться. Для этого вводим логин и пароль в окне авторизации рисунок 41. Где логин — это фамилия клиента, а пароль — это пароль, придуманный клиентом при регистрации.

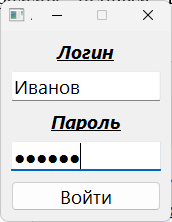


Рисунок 41 – Окно авторизации

После авторизации открывается окно пользователя рисунок 42

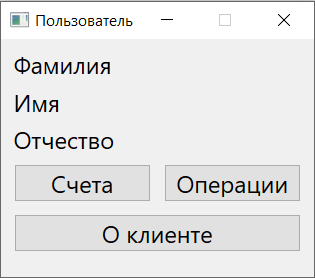


Рисунок 42 – Окно пользователя

При нажатии на кнопки будут появляться окна как на рисунке 43 где будет отображаться информация о клиенте, его счетах, и операциях по счетам.

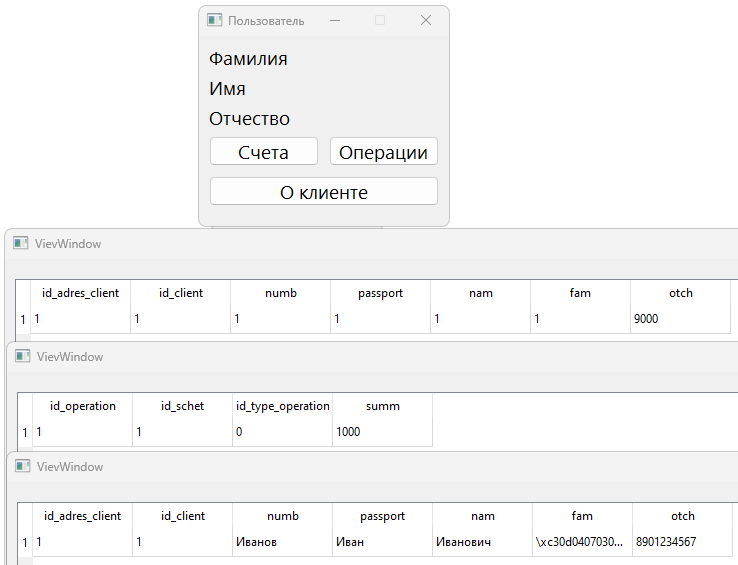


Рисунок 43 – Просмотр данных

На этом интерфейс пользователя заканчивается. Функционал, который необходим для пользователя выполняется.

## **4.2 Руководство по использованию ИС «Потребительское кредитование» для менеджера**

Для начала работы необходима авторизоваться как на рисунке 44.

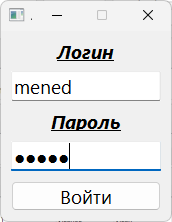


Рисунок 44 – Окно авторизации

После авторизации появится окно как на рисунке 45. В данном окне предоставляется базовый набор возможностей для менеджера. В частности, добавление удаление редактирование данных в таблицах, возможность выбирать базовые запросы к базе данных, а таже одновременный просмотр двух таблиц, и работа с этими таблицами.

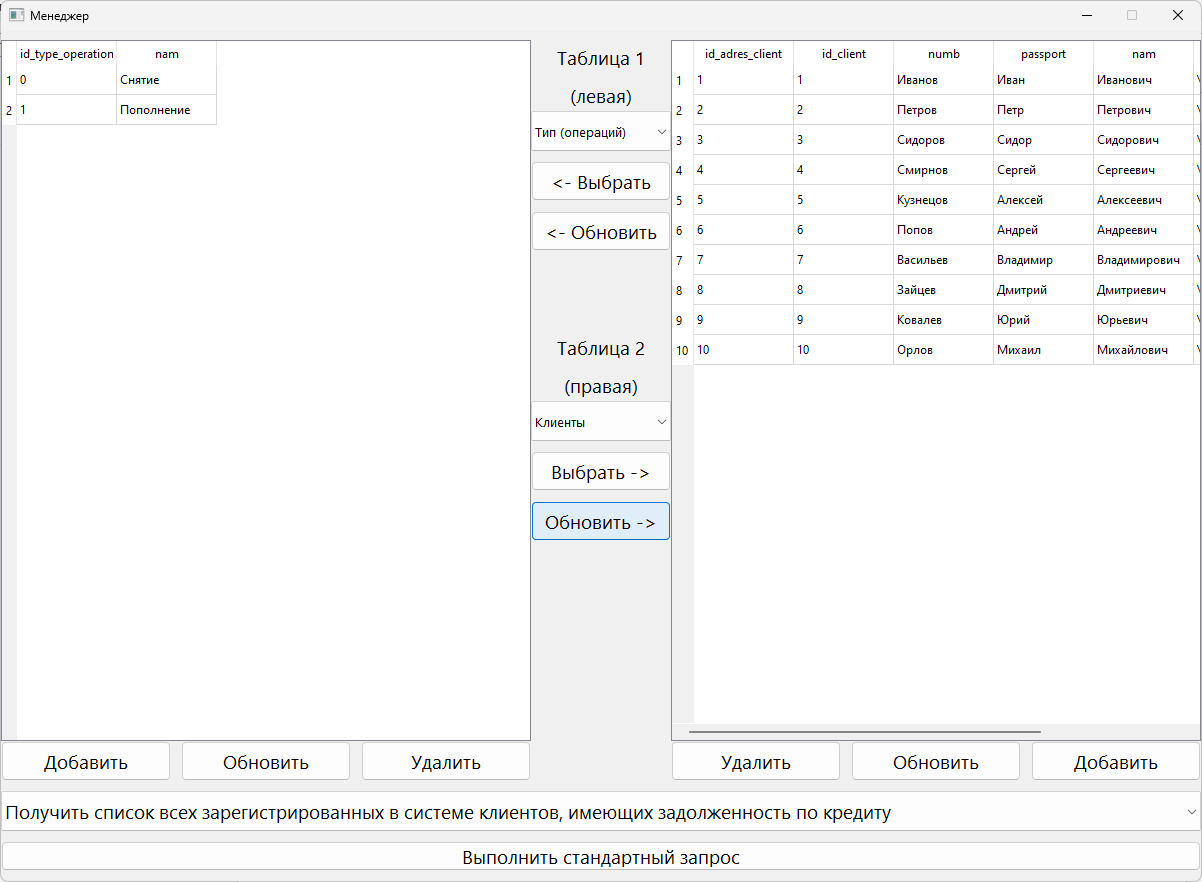


Рисунок 45 – Окно менеджера

На рисунке представлено окно добавления данных, для того чтобы данные добавились их необходимо ввести в соответствующие столбцы таблицы, и нажать кнопку выполнить после чего данные будут внесены в БД.

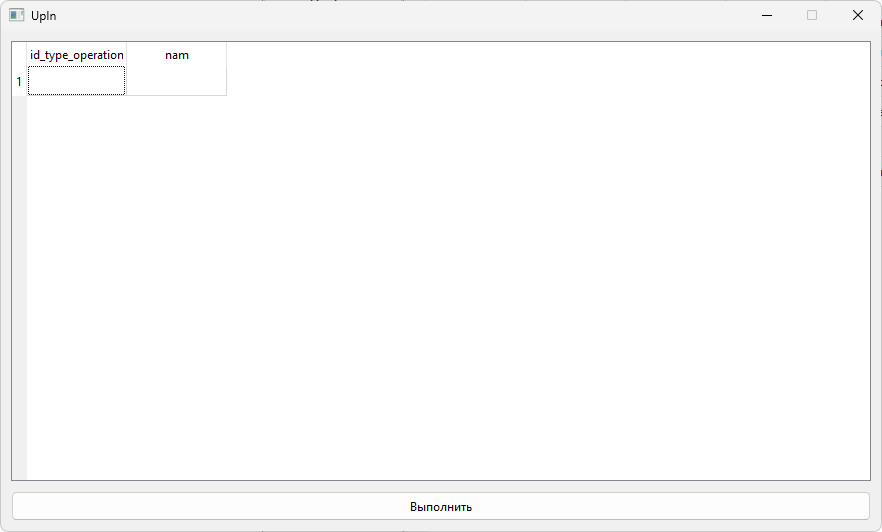


Рисунок 46 – Окно добавления данных

На рисунке 47 представлено окно редактирования данных в котором необходимо дважды нажать по данным которым требуется отредактировать, изменить их и нажать кнопку выполнить, данные после этого будут изменены в БД.

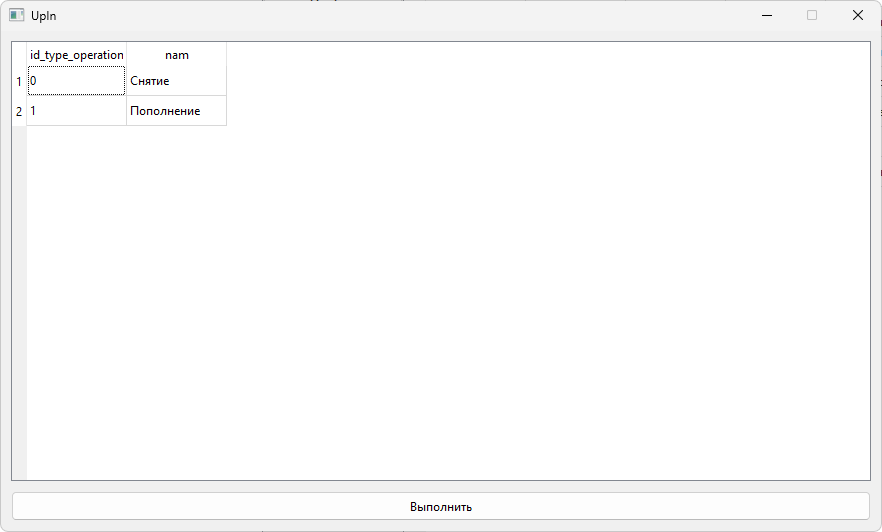


Рисунок 47 – Окно редактирования

На рисунке 48 представлено окно удаления данных. Для выполнения данной операции необходима выбрать идентификатор по которому будет происходить удаление и ввести значение, связанные данные с котором требуется удалить.

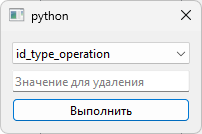


Рисунок 48 – Окно удаления данных

На рисунке 49 продемонстрировано окно просмотра результатов базового запроса. На нем можно ознакомится с данными, полученными по запросу.

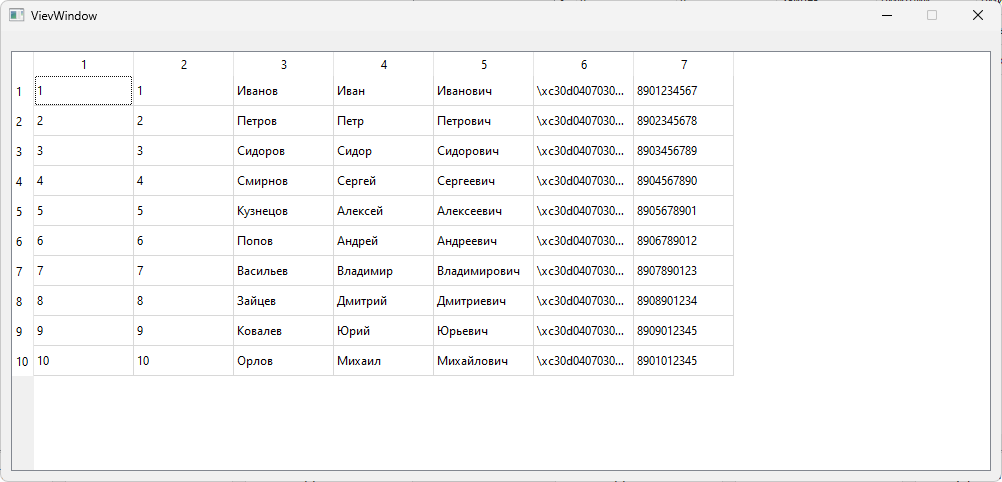


Рисунок 49 – Окно просмотра результатов базовых запросов

На данном моменте функционал интерфейса менеджера заканчивается.

## **4.3 Руководство по использованию ИС «Потребительское кредитование» для администратора**

Для начала работы необходимо авторизоваться как это показано на рисунке 50.

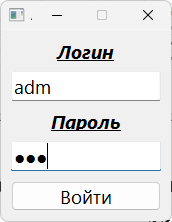


Рисунок 50 – Окно авторизации

После авторизации появляется окно, которое представлено на рисунке 51. В данном окне можно увидеть напоминание о том, что у человека, имеющего доступ к этому окну очень много прав по работе с БД, так как вся работа происходит непосредственно в терминале, который вызывается кнопкой с надписью «Открыть терминал». Также присутствует возможность в удобном формате просматривать содержание таблиц БД. Это сделано потому, что в терминале отображение таблиц происходит не всегда удобно и корректно, а также для того, чтобы администратор мог увидеть, как данные, с которыми он работает через терминал будут видеть пользователи и менеджеры.

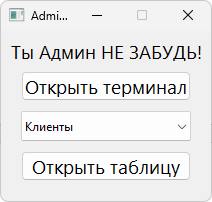


Рисунок 51 – Окно администратора

На рисунке 52 продемонстрирован пример отображения таблиц.

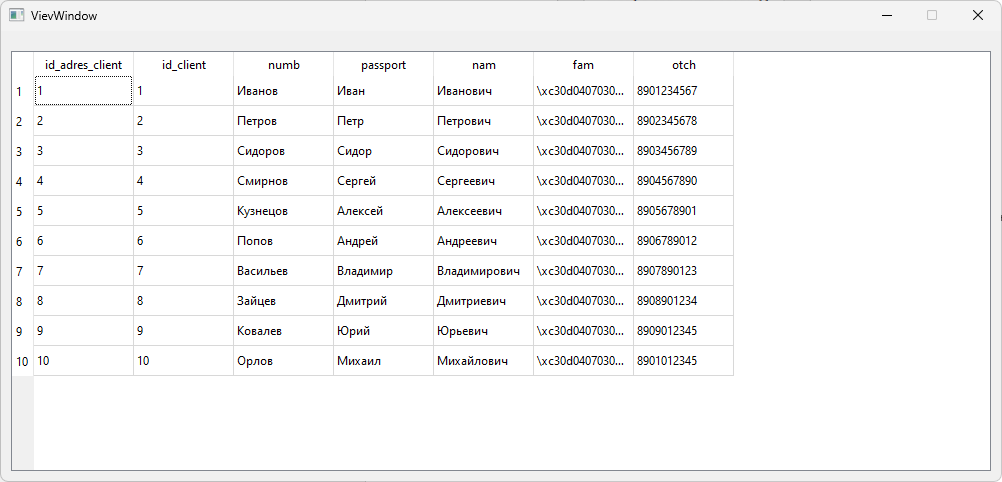


Рисунок 52 – Отображение таблиц

На рисунке 53 продемонстрированно окно терминала, довольно мощного инструмента в руках администратора по управлению базой данных. При использовании терминала администратор имеет наиболее полный доступ к системе PostgreSQL.

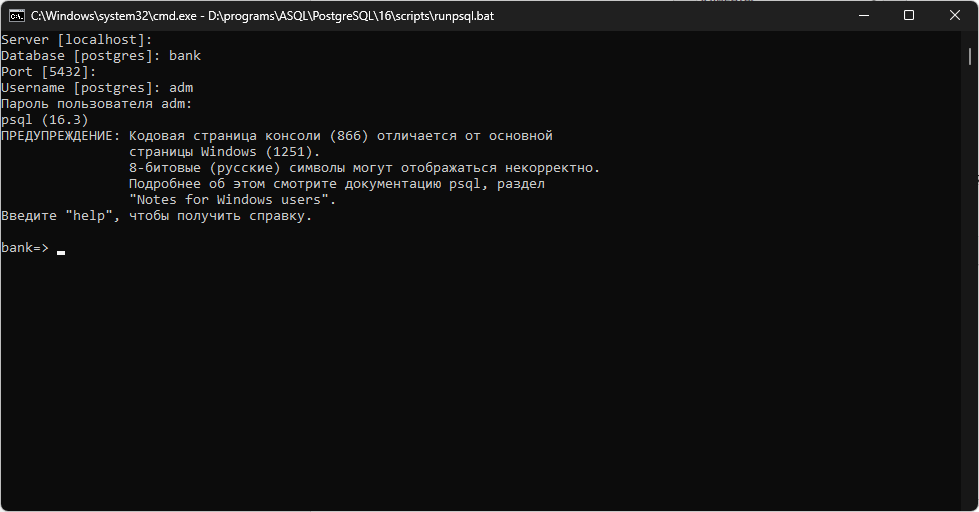


Рисунок 53 – Окно терминала

psql — это терминальный клиент для работы с PostgreSQL.

Он позволяет:

* интерактивно вводить запросы;
* передавать их в PostgreSQL;
* видеть результаты.

Также запросы могут быть получены из файла или из аргументов командной строки.

Кроме того, psql предоставляет ряд метакоманд и различные возможности, подобные тем, что имеются у командных оболочек, для облегчения написания скриптов и автоматизации широкого спектра задач.

[https://postgrespro.ru/docs/postgresql/15/app-psql]

Таким образом одно окно терминала в полной мере обеспечивает администратора функциональной мобильностью в отношении базы данных.

# **Заключение**

Благодаря проведённому анализу предметной области «Потребительское кредитования», были выявлены основные бизнес-процессы, определены группы пользователей, а также информационные запросы, выполняемые в информационной системе.

Были исследованы возможности обеспечения безопасности СУБД, также реализована схема БД и описана её структура. Была реализована система, включающая в себя триггеры, функции, политики доступа, шифрование некоторых данных и разграничение доступа. Все это в совокупность с использованием все возможных методов обеспечения безопасности и создания пользовательского интерфейса на Python, была реализована информационная система «Потребительское кредитование».

Составленная документация позволяет удостоверится что выбранные группы пользователей смогут пользоваться ПО. Так же интерфейс позволяет минимизировать риски нарушения целостности или доступности, однако не исключить их, т.к. в случае с администратором происходит полный контроль над БД.

Таким образом все поставленные задачи выполнены, цель достигнута.

# **Список используемых источников**

* + - 1. Разработка автоматизированной информационной системы // Библиофонд : Сайт. – URL: https://www.bibliofond.ru/view.aspx?id=699023 (дата обращения: 18.06.2024).
      2. Методы аутентификации в PostgreSQL // SysAdminium : Сайт. – URL: https://sysadminium.ru/metody\_autentifikacii\_v\_postgresql/ (дата обращения: 18.06.2024).
      3. Authentication Methods // PostgreSQL Documentation : Сайт. – URL: https://postgrespro.ru/docs/postgresql/16/auth-methods (дата обращения: 18.06.2024).
      4. Trust Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-trust.html (дата обращения: 18.06.2024).
      5. Password Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-password.html (дата обращения: 18.06.2024).
      6. GSSAPI Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/gssapi-auth.html (дата обращения: 18.06.2024).
      7. SSPI Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/sspi-auth.html (дата обращения: 18.06.2024).
      8. Ident Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-ident.html (дата обращения: 18.06.2024).
      9. Peer Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-peer.html (дата обращения: 18.06.2024).
      10. LDAP Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-ldap.html (дата обращения: 18.06.2024).
      11. RADIUS Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-radius.html (дата обращения: 18.06.2024).
      12. Certificate Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-cert.html (дата обращения: 18.06.2024).
      13. PAM Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-pam.html (дата обращения: 18.06.2024).
      14. BSD Authentication // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-bsd.html (дата обращения: 18.06.2024).
      15. Authentication Methods // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/auth-methods.html (дата обращения: 18.06.2024).
      16. A PostgreSQL Row Level Security Primer: Creating Large Policies // Crunchy Data : Сайт. – URL: https://www.crunchydata.com/blog/a-postgresql-row-level-security-primer-creating-large-policies (дата обращения: 18.06.2024).
      17. Encryption Options // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/encryption-options.html (дата обращения: 18.06.2024).
      18. Row Security Policies // PostgreSQL Documentation : Сайт. – URL: https://postgrespro.com/docs/postgresql/16/ddl-rowsecurity (дата обращения: 18.06.2024).
      19. Row Security Policies // PostgreSQL Documentation : Сайт. – URL: https://www.postgresql.org/docs/current/ddl-rowsecurity.html (дата обращения: 18.06.2024).
      20. Row Level Security: Your Data’s Bodyguard // Otus : Сайт. – URL: https://habr.com/ru/companies/otus/articles/722304/ (дата обращения: 18.06.2024).