РЕФЕРАТ

Расчетно-пояснительная записка 58 с., 26 рис., 14 табл., 17 ист., 1 прил.

Ключевые слова: база данных, модель организации данных, реляционная база данных, первичный ключ, внешний ключ, СУБД, индекс.

Цель работы – разработка базы данных для хранения информации о кофейнях и напитках из их меню, а также создание приложения, предоставляющего интерфейс для доступа к ней.

В процессе работы был проведен анализ предметной области и существующих решений, были спроектированы и разработаны база данных и приложение для доступа к ней, а также исследовано влияние индекса на время выполнения запросов к базе данных.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью работы является разработка базы данных для хранения информации о кофейнях и напитках из их меню, а также создание приложения, предоставляющего интерфейс для доступа к ней.

Для достижения поставленной цели необходимо выполнить следующие задачи:

— проанализировать предметную область;
— формализовать задачу;
— сформулировать описание пользователей проектируемого приложения;
— спроектировать сущности базы данных и ролевую модель на уровне базы данных;
— создать и заполнить базу данных;
— разработать приложение для доступа к базе данных;
— исследовать влияние индекса на время выполнения запросов к базе данных.

1 Аналитическая часть

1.1 Анализ предметной области

Предметной областью данной работы является сегмент предприятий общественного питания, специализирующихся на реализации напитков – кофейни.

Основные компоненты предметной области включают:

— сетевые структуры кофеен, состоящие из одного или более заведений;

— кофейни, относящиеся к определенной сети;

— напитки, которые включает в себя меню заведений;

— меню, в котором для покупателей указываются цены и размеры напитков;

— программы лояльности, помогающие кофейням удерживать и привлекать клиентов за счет, например, различных акций и скидок на напитки.

1.2 Анализ существующих решений

Существует достаточно большое количество приложений, предоставляющих пользователям возможность просмотра информации о кофейнях и поиска среди них подходящего заведения. Наиболее известными являются:

— Кофейная карта [1];

— Coffee Forest [2];

— Яндекс Карты [3].

Для сравнения существующих решений были выделены следующие критерии:

— возможность просмотра информации о кофейнях в приложении;

— возможность просмотра меню кофеен в приложении;

— возможность составления списка избранных кофеен;

— возможность составления списка избранных напитков;

— возможность поиска кофеен на основе наличия в их меню определенного напитка.

В таблице 1.1 представлены результаты сравнения упомянутых ранее приложений по выделенным критериям.

Таблица 1.1 — Сравнение существующих решений

Критерий сравнения	**Кофейная карта**	**Coffee Forest**	**Яндекс Карты**
Возможность просмотра информации о кофейнях в приложении	Да	Да	Да
Возможность составления списка избранных кофеен	Да	Да	Да
Возможность просмотра меню кофеен в приложении	Нет	Нет	Да
Возможность составления списка избранных напитков	Нет	Нет	Нет
Возможность поиска кофеен на основе наличия в их меню определенного напитка	Нет	Нет	Нет

Таким образом, вышеперечисленные приложения не предоставляют пользователям возможностей создания списка избранных напитков и поиска кофеен на основе наличия в их меню определенного напитка.

1.3 Анализ существующих моделей баз данных

База данных – это самодокументированное собрание интегрированных записей. База данных хранит сведения о предметной области, используемые в прикладных системах для удовлетворения информационных потребностей пользователя [4].

Разработка базы данных всегда сопровождается моделью данных, которая позволяет наглядно представить структуру хранимых данных и связи между ними. Выделяют 3 основных типа моделей организации данных:

— дореляционная модель;
— реляционная модель;
— постреляционная модель.

1.3.1 Дореляционная модель

Дореляционная модель организации данных разделяется на:

— иерархическую;

— сетевую.

Иерархическая модель данных представляется набором деревьев, связанных друг с другом по принципу построения иерархических структур [4]. Связи между записями выражаются в виде отношений предок/потомок, при чем у каждой записи есть ровно одна родительская запись. При этом если удаляется родительская запись, то автоматически должны быть удалены все дочерние записи. Это помогает поддерживать ссылочную целостность. В иерархической модели реализуется только связь типа 1:N («один–ко–многим»), при чем только в направлении от родительского элемента к дочернему. На рисунке 1.1 представлена общая схема иерархической модели.

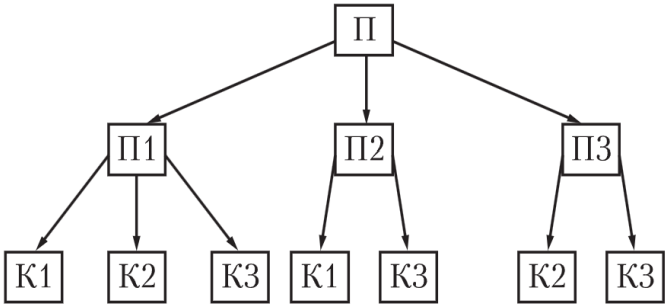

Рисунок 1.1 — Общая схема иерархической модели данных

Сетевая модель данных является расширением иерархической модели. Сетевая модель позволяет описывать связи M:N («многие–ко–многим»), чтобы одна запись могла участвовать в нескольких отношениях предок/потомок. Сетевые базы данных могут быть представлены в виде графа. Для данной модели характерна высокая сложность организации структуры памяти и жесткость модели, что приводит к необходимости полного пересмотра модели при появлении новых условий в предметной области, требующих внесения изменений в структуру данных [4]. На рисунке 1.2 представлена структура сетевой модели организации данных.

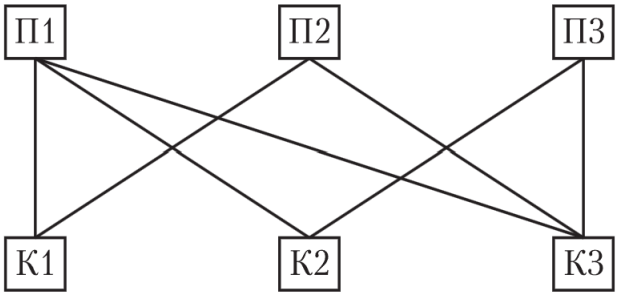

Рисунок 1.2 — Структура сетевой модели данных

Достоинством дореляционной модели данных является скорость доступа к данным [5].

К **недостаткам** дореляционной модели можно отнести необходимость значительных ресурсов дисковой и основной памяти компьютера, поскольку каждый элемент данных должен содержать ссылки на некоторые другие элементы [5]. Трудоемкость процесса изменения структуры данных и перегруженность логики деталями организации доступа к базе данных также являются недостатками этой модели.

1.3.2 Реляционная модель данных

В реляционной модели все данные и связи между ними хранятся в таблицах, а для определения структуры данных и манипулирования их значениями используют язык SQL (Structured Query Language – структурированный язык запросов). Реляционная модель опирается на систему понятий, важнейшими из которых являются тип данных, домен, атрибут, кортеж, первичный и внешний ключ.

Домен – допустимое потенциальное множество значений какого-то типа данных.

Атрибут отношения – это пара вида <имя_атрибута, имя_домена>. Атрибут является некоторой характеристикой сущности.

Схемой отношения называется именованное множество упорядоченных пар <имя_атрибута, имя_домена>.

Кортеж – это множество упорядоченных пар вида <имя_атрибута, значение_атрибута>, которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения.

Отношение, определенное на множестве из n доменов (не обязательно различных), содержит две части: заголовок (схему отношения) и тело (множество из m кортежей).

Атрибут, значение которого идентифицирует кортеж, называется ключом отношения. Среди всех ключей выбирается один первичный, который должен быть уникальным и неизбыточным. Для организации взаимосвязи отношений используется внешний ключ.

Реляционная база данных – это набор отношений, имена которых совпадают с именами схем отношений в схеме базы данных. На рисунке 1.3 представлен пример отношения (таблицы) реляционной базы данных.

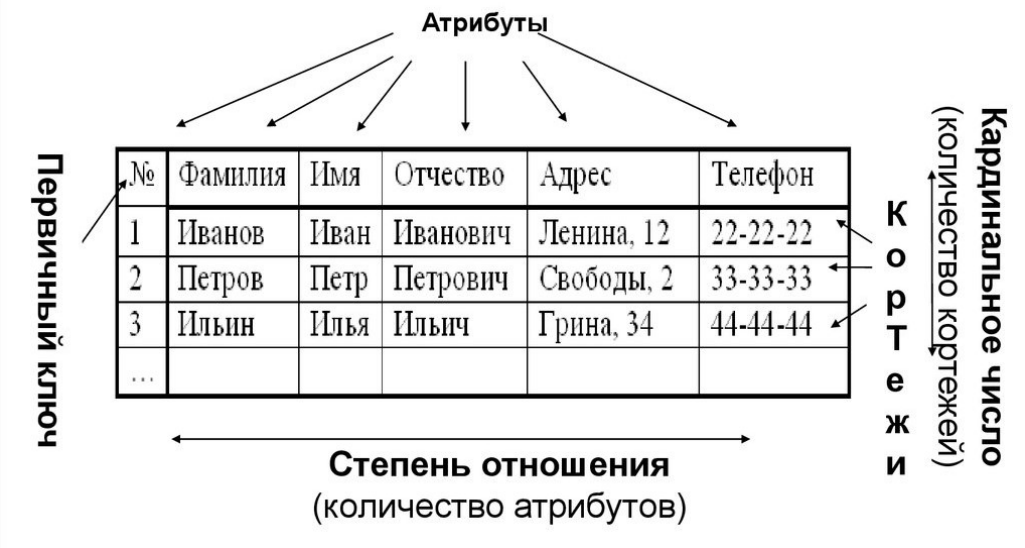

Рисунок 1.3 — Пример таблицы реляционной базы данных

Преимуществами реляционной модели являются согласованность данных, сравнительная простота инструментальных средств ее поддержки [5], независимость прикладных программ от правил хранения и размещения сведений в базе данных [4].

К **недостаткам** реляционной модели относят зависимость скорости выполнения операций от размера таблиц [5], ограничение в структурах представления данных, так как все данные хранятся в виде отношений, состоящих из простых атрибутов [6].

1.3.3 Постреляционные базы данных

Постреляционная модель организации данных представляет собой развитие реляционного подхода. Ее основной целью является расширение возможностей баз данных в тех областях, где реляционная модель и SQL недостаточно гибки [6].

Постреляционная модель снимает ограничение неделимости данных, допуская многозначные поля, и не накладывает требования на длину и количество полей в записях, что делает структуру таблиц более гибкой и наглядной.

К **достоинствам** постреляционной модели относят возможность работы со сложными и неструктурированными данными, а также большими объемами информации и большим количеством пользователей.

Недостатком постреляционной модели является сложность обеспечения целостности и непротиворечивости данных, хранимых в ней [6].

1.4 Выбор модели хранения данных

В качестве модели организации данных, которая будет использована в курсовой работе, была выбрана реляционная модель, поскольку она позволяет обеспечить структурированное хранение данных и их целостность.

1.5 Формализация задачи

Необходимо разработать базу данных, которая будет хранить информацию о кофейнях и их меню напитков, а также приложение для доступа к ней. Приложение должно предоставлять пользователям возможность регистрации и авторизации, просмотра информации о сетях кофеен, их меню и открытых филиалах, поиска кофеен, в которых продается определенный напиток. Также приложение должно позволять составлять список избранных напитков и список избранных кофеен. Реализовать возможность четырех уровней доступа: для гостей (неавторизованных пользователей), обычных пользователей (авторизованных), модераторов и администраторов.

1.5.1 Типы пользователей

Пользователи проектируемого приложения делятся на 4 типа:

— неавторизованный пользователь (гость);

— авторизованный пользователь;

— модератор;

— администратор.

В таблице 1.2 представлено описание типов пользователей проектируемого приложения.

Таблица 1.2 — Описание типов пользователей приложения

Тип пользователя	**Возможности**
Неавторизованный пользователь (Гость)	Регистрация, авторизация.
Авторизованный пользователь	Просмотр информации о сетях кофеен, напитках, кофейнях и программах лояльности сетей кофеен, составление списка избранных напитков и кофеен, а также поиск сети, в меню которой есть выбранный пользователем напиток, изменение данных профиля и удаление аккаунта .
Модератор	Все возможности авторизованного пользователя, а также добавление и удаление записей таблицы меню, таблицы напитков, таблицы кофеен.
Администратор	Все возможности модератора, а также просмотр списка пользователей, изменение прав доступа пользователей и удаление их аккаунтов.

На рисунке 1.4 представлена диаграмма прецедентов.

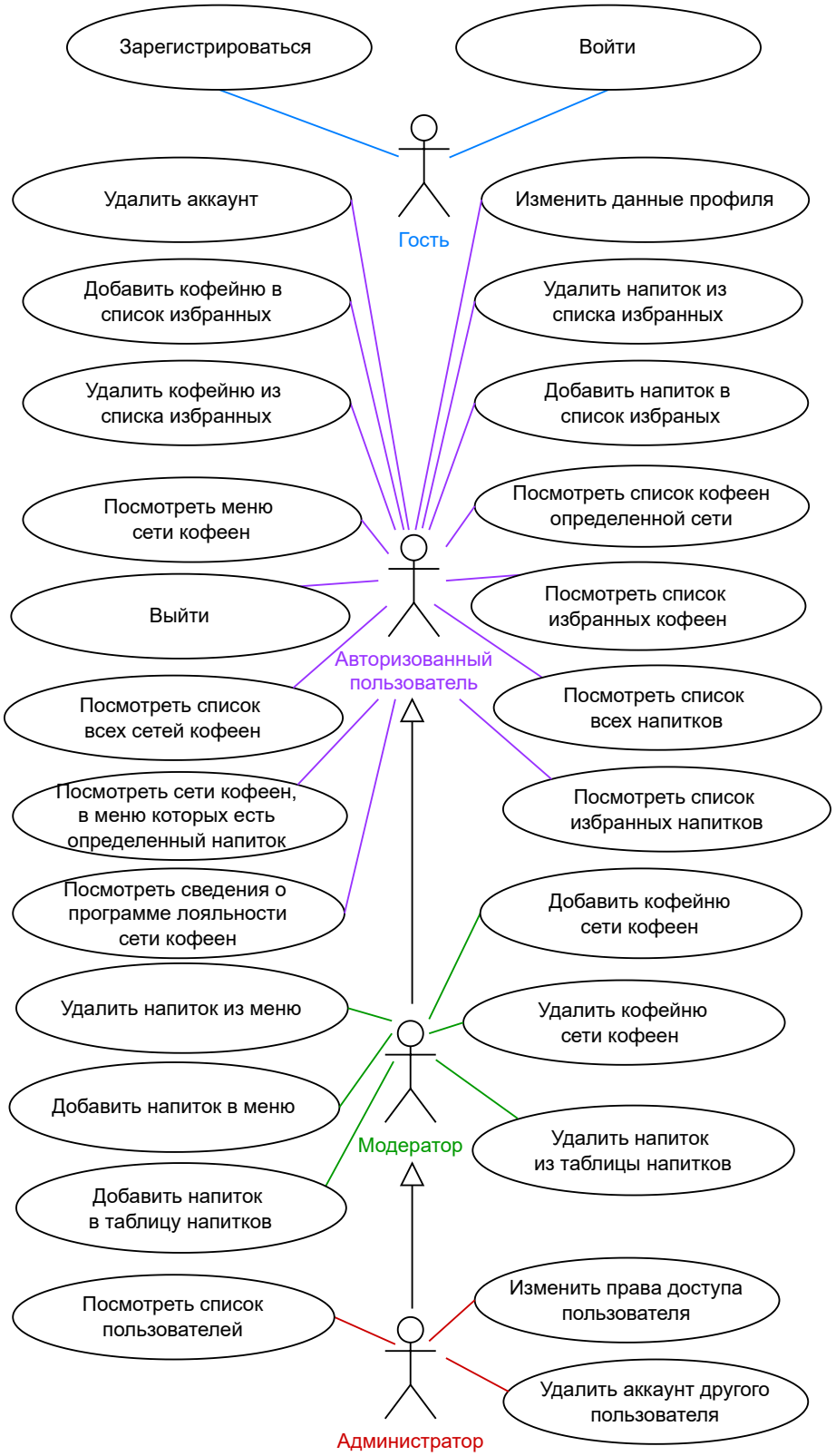

Рисунок 1.4 — Диаграмма прецедентов

1.5.2 Формализация данных

Согласно поставленной задаче, база данных должна содержать сущности, описанные в таблице 1.3.

Таблица 1.3 — Описание сущностей базы данных

Сущность	**Описание**
Пользователь	Id, логин, пароль, дата рождения, почта, Id роли
Роль	Id, название
Сеть кофеен	Id, название, сайт, количество кофеен
Кофейня	Id, Id сети кофеен, адрес, часы работы
Программа лояльности	Id, Id сети кофеен, тип, описание
Напиток	Id, название
Категория напитка	Id, название

На рисунке 1.5 представлена диаграмма сущность-связь базы данных.

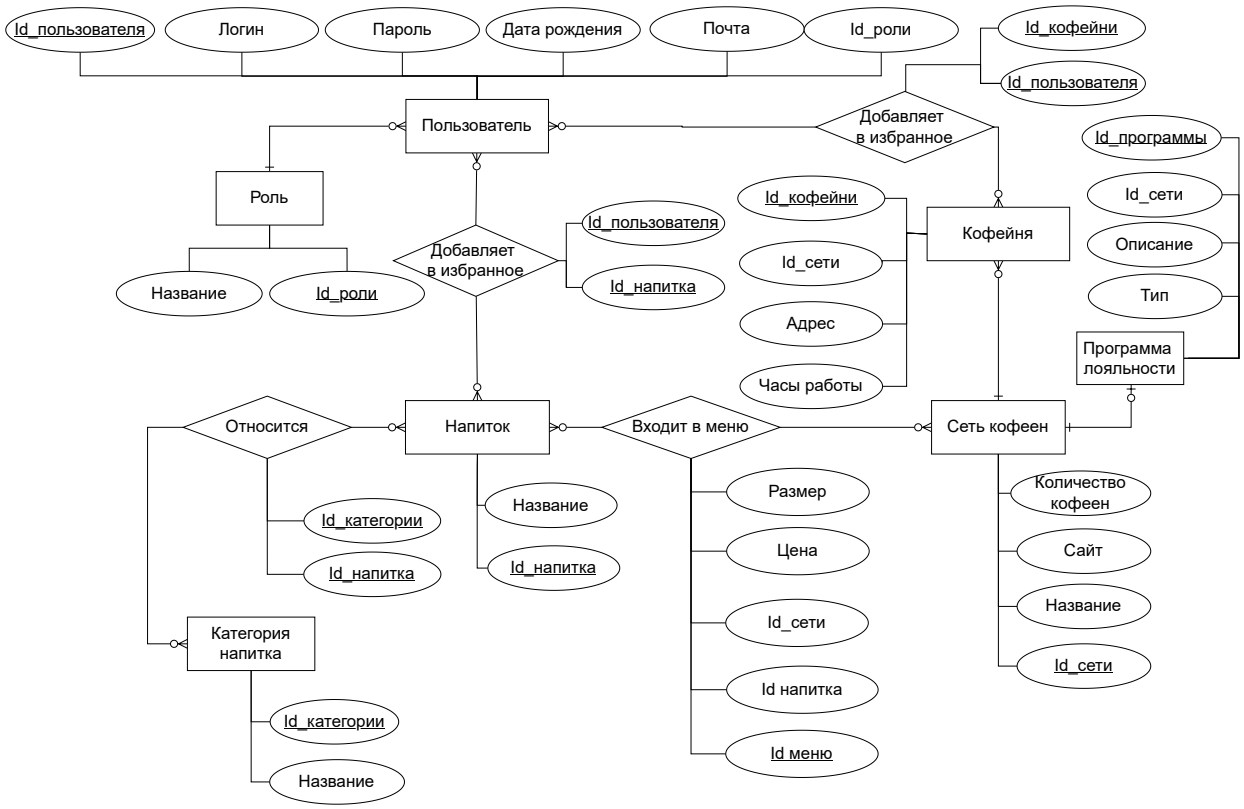

Рисунок 1.5 — Диаграмма сущность-связь базы данных

Вывод

В данной части был проведен анализ предметной области, существующих решений и моделей баз данных, описаны типы пользователей проектируемого приложения и данные, хранящиеся в базе данных. В качестве модели организации данных была выбрана реляционная модель.

2 Конструкторская часть

2.1 Проектирование базы данных

В соответствии с выделенными сущностями и связями между ними, показанными на диаграмме сущность-связь, представленной на рисунке 1.5, проектируемая база данных должна содержать следующие таблицы:

— таблица **users**, хранящая информацию о пользователях приложения;
— таблица **roles** типов пользователей приложения;
— таблица **drinks** всех напитков, имеющихся в системе;
— таблица **categories**, хранящая категории напитков;
— таблица **drinkscategory**, являющая связующей таблицей между напитком и категорией (формализует связь «многие–ко–многим»);
— таблица **favdrinks** избранных напитков (формализует связь «многие–ко–многим» между пользователем и напитком);
— таблица **companies**, хранящая информацию о сетях кофеен;
— таблица **menu**, хранящая информацию о меню напитков;
— таблица **loyaltyprograms**, хранящая информацию о программе лояльности сети кофеен;
— таблица **coffeeshops**, хранящая информацию о кофейнях сети;
— таблица **favcoffeeshops** избранных кофеен (формализует связь «многие–ко–многим» между пользователем и кофейней).

На рисунке 2.1 представлена диаграмма проектируемой базы данных.

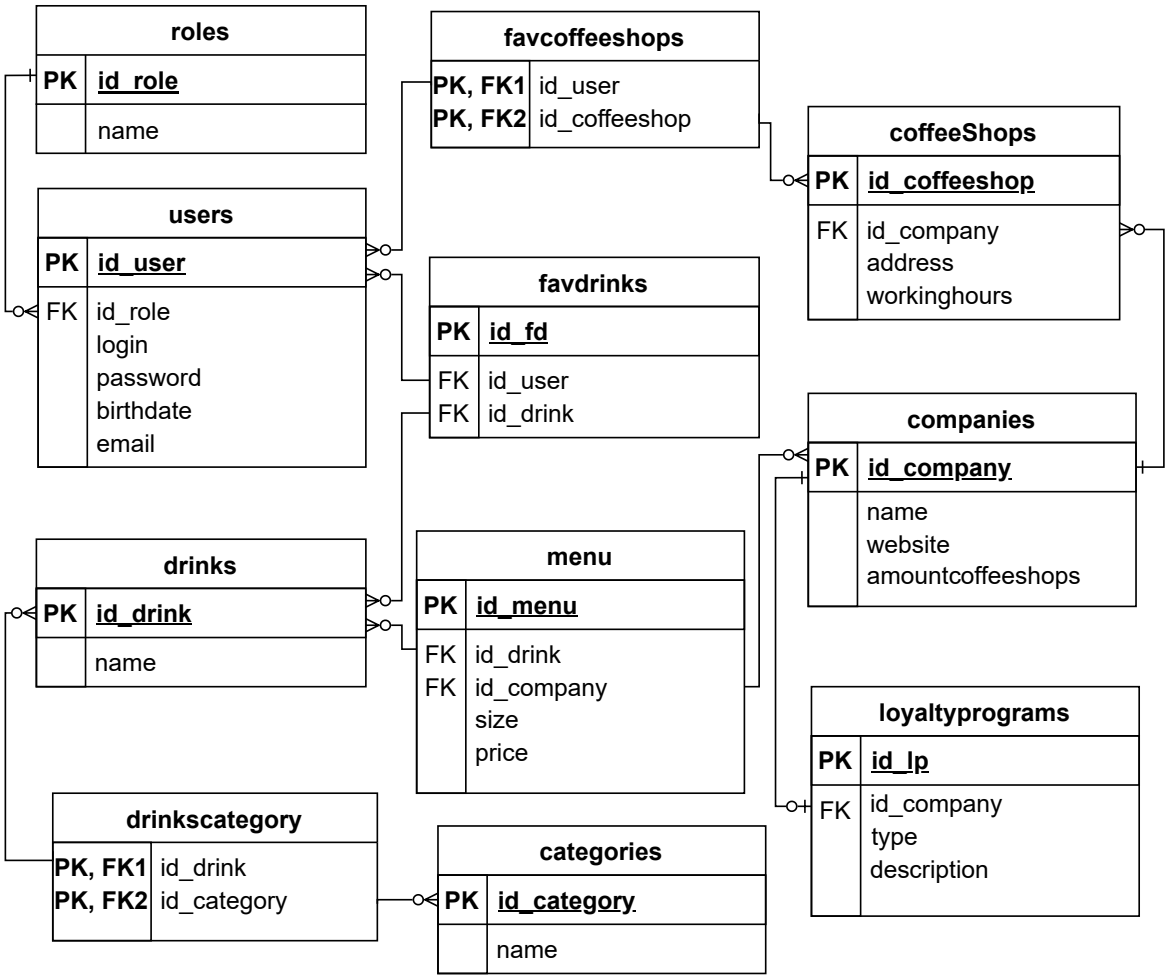

Рисунок 2.1 — Диаграмма базы данных

Таблица **users** содержит следующие поля:

- id_user – идентификатор пользователя, первичный ключ;
- login – логин пользователя;
- password – пароль пользователя;
- birthdate – дата рождения пользователя;
- email – электронная почта пользователя;
- id_role – идентификатор роли пользователя, внешний ключ.

Таблица **roles** содержит следующие поля:

- id_role – идентификатор роли пользователя, первичный ключ;
- name – название роли.

Таблица **drinks** содержит следующие поля:

- id_drink – идентификатор напитка, первичный ключ;
- name – название напитка.

Таблица **categories** содержит следующие поля:

- id_category – идентификатор категории напитка, первичный ключ;
- name – название категории напитка.

Таблица **drinkscategory** содержит следующие поля:

- id_category – идентификатор категории напитка, внешний ключ;
- id_drink – идентификатор напитка, внешний ключ.

Таблица **favdrinks** содержит следующие поля:

- id_user – идентификатор пользователя, внешний ключ;
- id_drink – идентификатор напитка, внешний ключ.

Таблица **companies** содержит следующие поля:

- id_company – идентификатор сети кофеен, первичный ключ;
- name – название сети;
- website – веб-сайт;
- amountcoffeeshops – количество открытых кофеен сети.

Таблица **menu** содержит следующие поля:

- id_menu – первичный ключ;
- id_drink – идентификатор напитка, внешний ключ;
- id_company – идентификатор сети кофеен, внешний ключ;
- size – объем напитка (в миллилитрах);
- price – цена (в рублях).

Таблица **loyaltyprograms** содержит следующие поля:

- id_lp – идентификатор программы лояльности, первичный ключ;
- id_company – идентификатор сети кофеен, внешний ключ;
- type – тип программы лояльности(например, пластиковая карта, мобильное приложение и тд.);
- description – краткое описание предоставляемых программой лояльности предложений для покупателей.

Таблица **coffeeshops** содержит следующие поля:

- id_coffeeshop – идентификатор кофейни, первичный ключ;
- id_company – идентификатор сети кофеен, внешний ключ;
- address – адрес кофейни;
- workinghours – часы работы.

Таблица **favcoffeeshops** содержит следующие поля:

— id_user – идентификатор пользователя, внешний ключ;
— id_coffeeshop – идентификатор кофейни, внешний ключ.

2.2 Ограничения целостности данных

2.2.1 Целостность таблиц

Для обеспечения целостности таблиц каждая строка должна иметь уникальный идентификатор (первичный ключ). Как было описано в разделе 2.1, каждая из таблиц имеет первичный ключ:

— поле id_role в таблице roles;
— поле id_user в таблице users;
— поле id_drink в таблице drinks;
— поле id_category в таблице categories;
— поле id_company в таблице companies;
— поле id_coffeeshop в таблице coffeeshops;
— поле id_lp в таблице loyaltyprograms;
— поле id_menu в таблице menu;
— поля id_drink и id_category в таблице drinkscategory (составной первичный ключ);
— поля id_user и id_drink в таблице favdrinks (составной первичный ключ);
— поля id_user и id_coffeeshop в таблице favcoffeeshops(составной первичный ключ).

2.2.2 Целостность полей

Чтобы обеспечить целостность полей таблицы, следует указать набор допустимых значений для них.

Для таблицы **roles** необходимо выполнить следующие условия:

— поле name может принимать значения «user», «moderator», «guest», «administrator»;
— любые поля не должны быть пустыми.

Для таблицы **users** необходимо выполнить следующие условия:

— значение поля login должно быть уникальным, поскольку оно используется для авторизации пользователей;

— значение поля birthdate, хранящее дату рождения пользователя, не должно превышать значение текущей даты;
— любые поля не должны быть пустыми.

Для таблицы **drinks** необходимо выполнить следующие условия:
— поле name должно быть уникальным;
— все поля не могут быть пустыми.

Для таблицы **categories** необходимо выполнить следующие условия:
— поле name должно быть уникальным;
— все поля не могут быть пустыми.

Для таблицы **companies** необходимо выполнить следующие условия:
— значение поля amountcoffeeshops не может быть отрицательным;
— все поля, кроме поля website, не могут быть пустыми.

Для таблицы **menu** необходимо выполнить следующие условия:
— значение поля size должно быть > 0;
— значение поля price должно быть неотрицательным;
— все поля не могут быть пустыми.

Для таблицы **loyaltyprograms** необходимо выполнить следующие условия:
— все поля не могут быть пустыми.

Для всех остальных таблиц необходимо выполнить следующие условия:
— любые поля не должны быть пустыми.

2.2.3 Целостность ссылок

Связь между таблицами организована с помощью внешних ключей. Для обеспечения ссылочной целостности необходимо, чтобы для каждого значения внешнего ключа, появляющегося в дочерней таблице, в родительской таблице существовал кортеж с таким же значением первичного ключа. В таблице 2.1 представлены все внешние ключи таблиц, имеющихся в базе данных.

Таблица 2.1 — Внешние ключи таблиц

Таблица	**Внешний ключ**	**Таблица, на поле которой ссылается внешний ключ**
users	id_role	roles(id_role)
favdrinks	id_user id_drink	users(id_user) drinks(id_drink)
favcoffeeshops	id_user id_coffeeshop	users(id_user) cofeeshops(id_coffeeshop)
menu	id_company id_drink	companies(id_company) drinks(id_drink)
drinkscategory	id_category id_drink	categories(id_category) drinks(id_drink)
loyaltyprograms	id_company	companies(id_company)

2.3 Триггеры

Для поддержания ссылочной целостности необходимо реализовать следующие триггеры.

Триггер, алгоритм работы которого представлен на рисунке 2.2, удаляет из таблиц menu, favdrinks и drinkscategory все записи, связанные с удаляемым из таблицы drinks напитком.

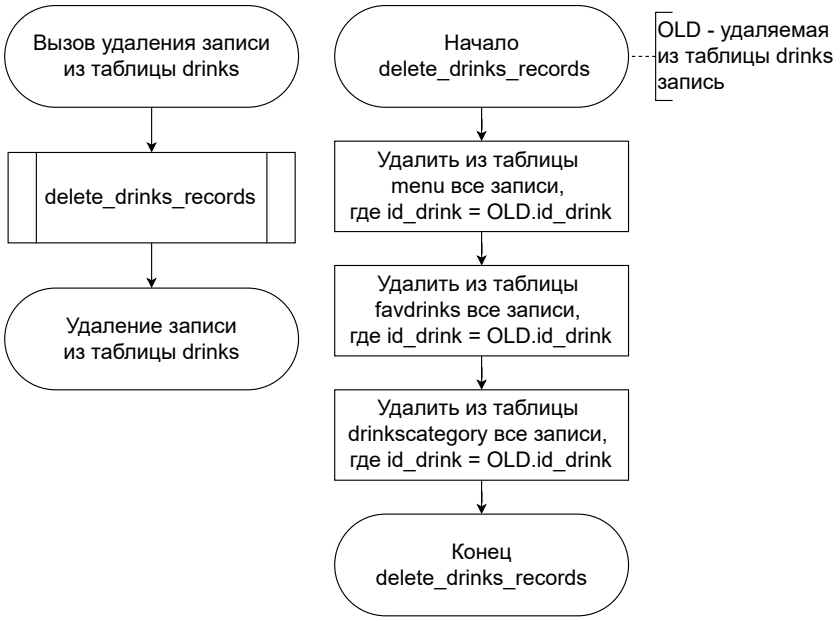

Рисунок 2.2 — Триггер на удаление всех записей из таблиц menu, favdrinks и drinkscategory, связанных с удаляемым из таблицы drinks напитком

Триггер, удаляющий из таблиц favcoffeeshops и favdrinks все записи, связанные с удаляемым из таблицы users пользователем. Алгоритм работы этого триггера представлен на рисунке 2.3.

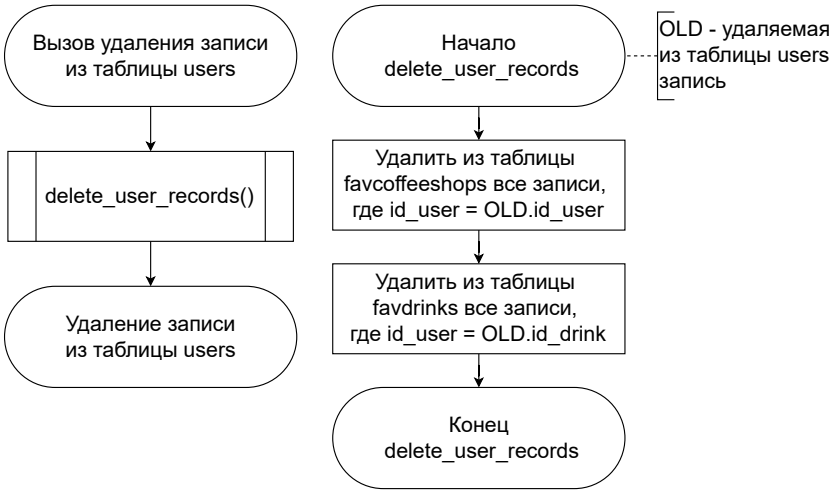

Рисунок 2.3 — Триггер на удаление всех записей из таблиц favcoffeeshops и favdrinks, связанных с удаляемым из таблицы users пользователем

Триггер, алгоритм работы которого представлен на рисунке 2.4, удаляет из таблицы favcoffeeshops все записи, связанные с удаляемой из таблицы coffeeshops кофейней.

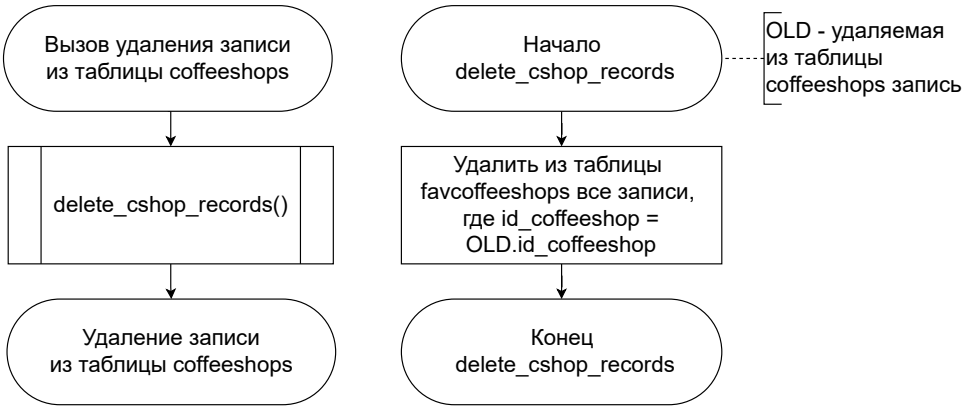

Рисунок 2.4 — Триггер на удаление всех записей из таблицы favcoffeeshops, связанных с удаляемой из таблицы coffeeshops кофейней

Помимо описанных выше триггеров необходимо также реализовать триггеры, которые будет увеличивать и уменьшать значение поля amountcoffeeshops записи из таблицы companies при добавлении в таблицу coffeeshops кофейни со значением внешнего ключа, соответствующего первичному ключу упомянутой

ранее записи. Алгоритм работы триггера, инкрементирующего значение поля amountcoffeeshops, представлен на рисунке 2.5, а триггера, декрементирующего это значение, – на рисунке 2.6.

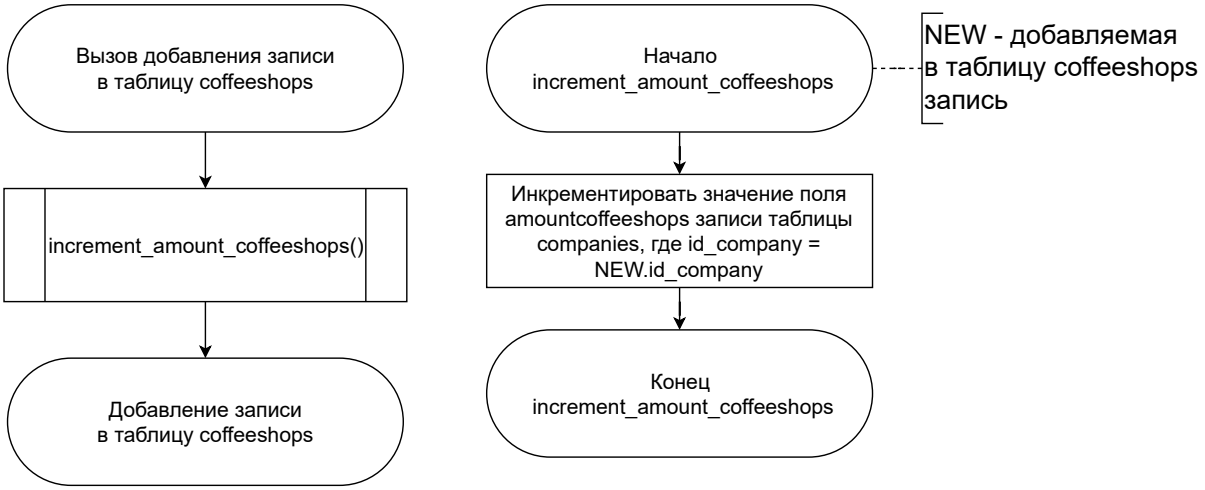

Рисунок 2.5 — Триггер, инкрементирующий значение поля amountcoffeeshops таблицы companies

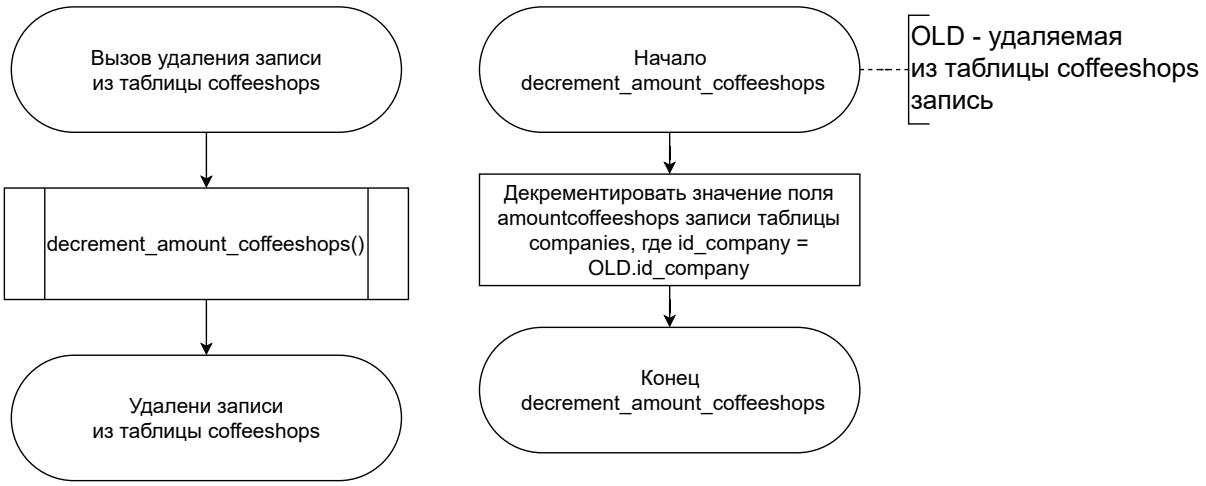

Рисунок 2.6 — Триггер, декрементирующий значение поля amountcoffeeshops таблицы companies

2.4 Функции

Для поиска по идентификатору напитка сетей кофеен, в меню которых представлен данный напиток, определена функция companies_by_drink. Алгоритм этой функции представлен на рисунке 2.7.

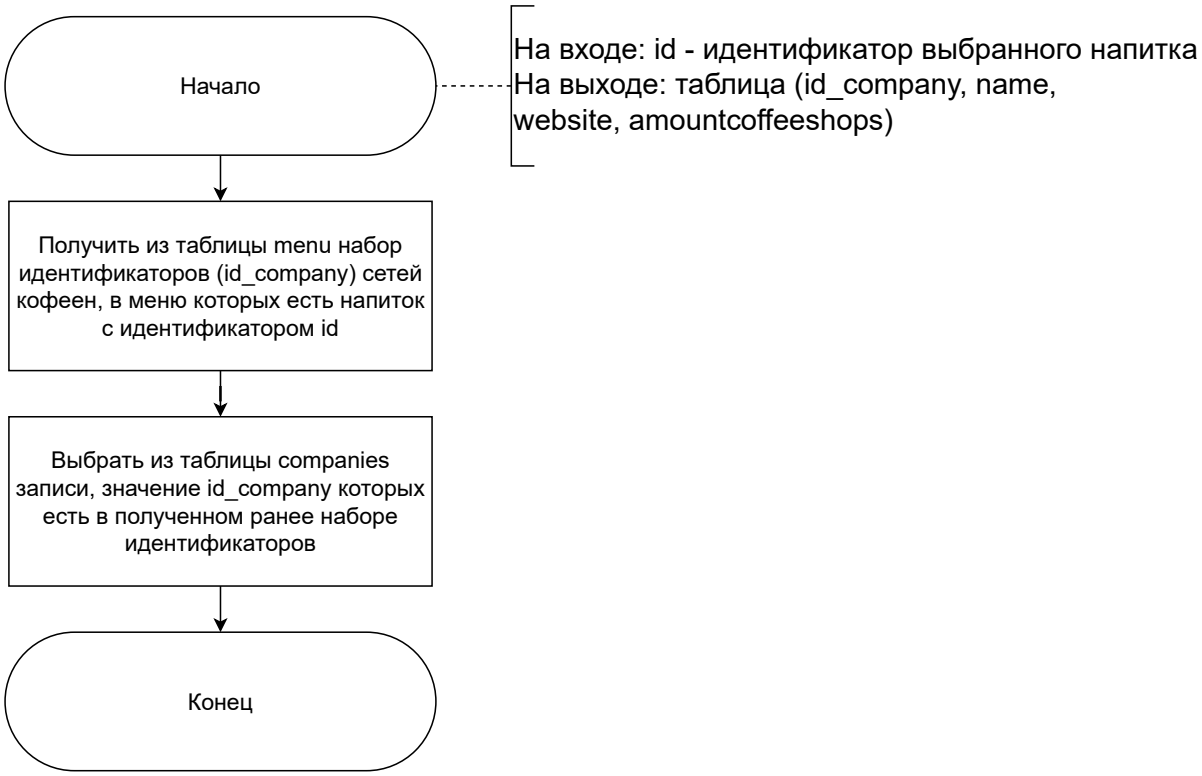

Рисунок 2.7 — Функция поиска сетей кофеен, в меню которых есть выбранный пользователем напиток

Для изменения прав доступа пользователя определена хранимая процедура update_user_rights. Алгоритм этой процедуры представлен на рисунке 2.8.

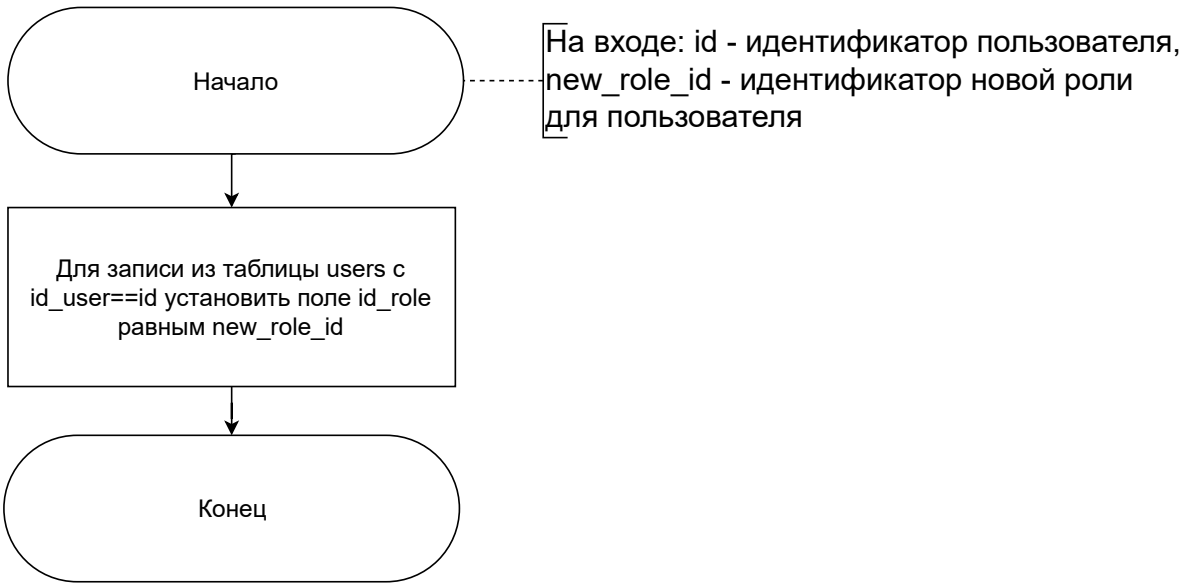

Рисунок 2.8 — Хранимая процедура, выполняющая изменение прав доступа пользователя

2.5 Ролевая модель на уровне базы данных

В проектируемой базе данных определены следующие роли: гость, обычный пользователь, модератор, администратор.

В соответствии с диаграммой прецедентов, представленной на рисунке 1.4, **гостю** выдаются следующие права на таблицы базы данных:

— выборка данных из таблиц users, roles;

— добавление данных в таблицу users.

В соответствии с диаграммой прецедентов, представленной на рисунке 1.4, **обычному пользователю** выдаются следующие права на таблицы базы данных:

— выборка данных из таблиц users, drinks, companies, coffeeshops, favdrinks, favcoffeeshops, categories, drinkscategory, menu, loyaltyprograms, roles;

— добавление данных в таблицы users, favdrinks и favcoffeeshops.

— удаление данных из таблиц favdrinks и favcoffeeshops.

— обновление данных таблицы users.

В соответствии с диаграммой прецедентов, представленной на рисунке 1.4, **модератору** выдаются следующие права на таблицы базы данных:

— выборка данных из таблиц users, drinks, companies, coffeeshops, favdrinks, favcoffeeshops, categories, drinkscategory, menu, loyaltyprograms, roles;

— добавление данных в таблицы users, favdrinks, favcoffeeshops, drinks, menu, coffeeshops, drinkscategory, categories.

— удаление данных из таблиц users, favdrinks, favcoffeeshops, drinks, menu, coffeeshops, drinkscategory;

— обновление данных таблицы users.

В соответствии с диаграммой прецедентов, представленной на рисунке 1.4, **администратору** выдаются следующие права на таблицы базы данных:

— выборка данных из таблиц users, drinks, companies, coffeeshops, favdrinks, favcoffeeshops, categories, drinkscategory, menu, loyaltyprograms, roles;

— добавление данных в таблицы users, favdrinks, favcoffeeshops, drinks, menu, coffeeshops, drinkscategory, categories.

— удаление данных из таблиц users, favdrinks, favcoffeeshops, drinks, menu, coffeeshops, drinkscategory;

— обновление данных таблицы users.

Вывод

В данной части были спроектированы таблицы базы данных, описаны ограничения целостности, проектируемые триггеры и функции, а также представлена ролевая модель на уровне базы данных.

3 Технологическая часть

3.1 Выбор СУБД

Одними из наиболее популярных СУБД являются:

— PostgreSQL [7];

— MySQL [8];

— Microsoft SQL Server [9];

— Oracle [10].

Для сравнения перечисленных выше СУБД были выделены следующие критерии:

— распространяется бесплатно;

— наличие документации;

— возможность реализации триггеров;

— возможность реализации пользовательских функций;

— возможность индексирования;

— возможность реализации ролевой модели;

— поддержка стандарта SQL.

В таблице 3.1 приведено сравнение перечисленных ранее СУБД по выделенным критериям.

Таблица 3.1 — Сравнение СУБД

Критерий сравнения	**PostgreSQL**	**MySQL**	**Microsoft SQL Server**	**Oracle**
Распространяется бесплатно	+	+	–	–
Наличие документации	+	+	+	+
Возможность реализации триггеров	+	+	+	+
Возможность реализации пользовательских функций	+	+	+	+
Возможность индексирования	+	+	+	+
Возможность реализации ролевой модели	+	+/– (с 8.0)	+	+
Поддержка стандарта SQL	+	+/– (частично)	+	+

Для решения поставленной задачи в качестве СУБД будет использоваться PostgreSQL, в силу бесплатного распространения, возможности индексирования, создания пользовательских функций и триггеров, а также реализации ролевой модели.

3.2 Выбор средств реализации пользовательского интерфейса

Для разработки интерфейса доступа к базе данных были выбраны язык программирования C# [11] и платформа .NET [12], поскольку они предоставляют возможность использования бесплатных инструментов для создания веб-приложений и библиотеки (драйвера) Npgsql [13] для работы с PostgreSQL.

В качестве среды разработки была выбрана Visual Studio 2022 [14], потому что она является бесплатной и автоматизирует этапы сборки и отладки приложения.

3.3 Реализация объектов базы данных

3.3.1 Создание таблиц

На листингах 3.1–3.11 представлено создание всех таблиц базы данных, а также реализация упомянутых ранее ограничений целостности.

Листинг 3.1 — Создание таблицы roles

```
CREATE TABLE IF NOT EXISTS roles (
    id_role INT PRIMARY KEY,
    name VARCHAR(128) NOT NULL,
    CHECK (name = 'ordinary_user' OR name = 'moderator' OR name =
       'administrator' OR name='guest')
);
```

Листинг 3.2 — Создание таблицы users

```
CREATE TABLE IF NOT EXISTS users (
    id_user UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    id_role INT NOT NULL,
    login VARCHAR(128) NOT NULL UNIQUE,
    password VARCHAR(128) NOT NULL,
    birthdate DATE NOT NULL,
    CHECK (birthdate < CURRENT_DATE),
    email VARCHAR(256) NOT NULL
);
ALTER TABLE users ADD FOREIGN KEY(id_role) REFERENCES roles(id_role);
```

Листинг 3.3 — Создание таблицы drinks

```
CREATE TABLE IF NOT EXISTS drinks(
    id_drink UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    name VARCHAR(128) NOT NULL UNIQUE
);
```

Листинг 3.4 — Создание таблицы categories

```
CREATE TABLE IF NOT EXISTS categories(
    id_category UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    name VARCHAR(128) NOT NULL UNIQUE
);
```

Листинг 3.5 — Создание таблицы favdrinks

```
CREATE TABLE IF NOT EXISTS favdrinks(
    id_user UUID NOT NULL,
    id_drink UUID NOT NULL,
    PRIMARY KEY (id_user, id_drink)
);
ALTER TABLE favdrinks ADD FOREIGN KEY(id_drink) REFERENCES drinks(id_drink);
ALTER TABLE favdrinks ADD FOREIGN KEY(id_user) REFERENCES users(id_user);
```

Листинг 3.6 — Создание таблицы drinkscategory

```
CREATE TABLE IF NOT EXISTS drinkscategory(
    id_drink UUID NOT NULL,
    id_category UUID NOT NULL,
    PRIMARY KEY (id_drink, id_category)
);
ALTER TABLE drinkscategory ADD FOREIGN KEY(id_drink) REFERENCES
   drinks(id_drink);
ALTER TABLE drinkscategory ADD FOREIGN KEY(id_category) REFERENCES
   categories(id_category);
```

Листинг 3.7 — Создание таблицы companies

```
CREATE TABLE IF NOT EXISTS companies(
    id_company UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    name VARCHAR(128) NOT NULL,
    website VARCHAR (256),
    amountcoffeeshops INT NOT NULL DEFAULT 0,
    CHECK (amountcoffeeshops >= 0)
);
```

Листинг 3.8 — Создание таблицы loyaltyprograms

```
CREATE TABLE IF NOT EXISTS loyaltyprograms(
    id_lp UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    id_company UUID NOT NULL,
    type TEXT,
    description TEXT
);
ALTER TABLE loyaltyprograms ADD FOREIGN KEY(id_company) REFERENCES
   companies(id_company);
```

Листинг 3.9 — Создание таблицы menu

```
CREATE TABLE IF NOT EXISTS menu(
    id_menu UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    id_drink UUID NOT NULL,
    id_company UUID NOT NULL,
    size INT NOT NULL,
    price NUMERIC(10,2) NOT NULL CHECK (price >= 0)
);
ALTER TABLE menu ADD FOREIGN KEY(id_drink) REFERENCES drinks(id_drink);
ALTER TABLE menu ADD FOREIGN KEY(id_company) REFERENCES
   companies(id_company);
```

Листинг 3.10 — Создание таблицы coffeeshops

```
CREATE TABLE IF NOT EXISTS coffeeshops(
    id_coffeeshop UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    id_company UUID NOT NULL,
    address VARCHAR(256) NOT NULL,
    workinghours VARCHAR(64) NOT NULL
);
ALTER TABLE coffeeshops ADD FOREIGN KEY(id_company) REFERENCES
   companies(id_company);
```

Листинг 3.11 — Создание таблицы favcoffeeshops

```
CREATE TABLE IF NOT EXISTS favcoffeeshops(
    id_user UUID NOT NULL,
    id_coffeeshop UUID NOT NULL,
    PRIMARY KEY (id_user, id_coffeeshop)
);
ALTER TABLE favcoffeeshops ADD FOREIGN KEY(id_coffeeshop) REFERENCES
   coffeeshops(id_coffeeshop);
ALTER TABLE favcoffeeshops ADD FOREIGN KEY(id_user) REFERENCES
   users(id_user);
```

3.3.2 Создание триггеров

На листингах 3.12–3.16 представлено создание описанных в конструкторской части триггеров. Для каждого из триггеров сначала создается функция, которая будет выполнена автоматически при наступлении соответствующего события.

Листинг 3.12 — Создание триггера, удаляещего из таблиц menu, favdrinks и drinkscategory все записи, связанные с удаляемым из таблицы drinks напитком

```
CREATE OR REPLACE FUNCTION delete_drinks_records()
RETURNS TRIGGER AS $$
BEGIN
    DELETE FROM menu WHERE id_drink = OLD.id_drink;
    DELETE FROM favdrinks WHERE id_drink = OLD.id_drink;
    DELETE FROM drinkscategory WHERE id_drink = OLD.id_drink;
    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER before_delete_drink BEFORE DELETE ON drinks FOR EACH ROW
   EXECUTE FUNCTION delete_drinks_records();
```

Листинг 3.13 — Создание триггера, удаляещего из таблицы favcoffeeshops все записи, связанные с удаляемой из таблицы coffeeshops кофейней

```
CREATE OR REPLACE FUNCTION delete_cshop_records()
RETURNS TRIGGER AS $$
BEGIN
    DELETE FROM favcoffeeshops WHERE id_coffeeshop = OLD.id_coffeeshop;
    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER before_delete_coffeeshop BEFORE DELETE ON coffeeshops FOR
   EACH ROW EXECUTE FUNCTION delete_cshop_records();
```

Листинг 3.14 — Создание триггера, удаляещего из таблиц favcoffeeshops и favdrinks все записи, связанные с удаляемым из таблицы users пользователем

```
CREATE OR REPLACE FUNCTION delete_user_records()
RETURNS TRIGGER AS $$
BEGIN
    DELETE FROM favcoffeeshops WHERE id_user = OLD.id_user;
    DELETE FROM favdrinks WHERE id_user = OLD.id_user;
    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER before_delete_user BEFORE DELETE ON user FOR EACH ROW
   EXECUTE FUNCTION delete_user_records();
```

Листинг 3.15 — Создание триггера, инкрементирующего значение поля amountcoffeeshops таблицы companies

```
CREATE OR REPLACE FUNCTION increment_amount_coffeeshops()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE companies
    SET amountcoffeeshops = amountcoffeeshops + 1
    WHERE id_company = NEW.id_company;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER after_insert_coffeeshop AFTER INSERT ON coffeeshops FOR EACH
   ROW EXECUTE FUNCTION increment_amount_coffeeshops();
```

Листинг 3.16 — Создание триггера, декрементирующего значение поля amountcoffeeshops таблицы companies

```
CREATE OR REPLACE FUNCTION decrement_amount_coffeeshops()
RETURNS TRIGGER AS $$
BEGIN
    UPDATE companies
    SET amountcoffeeshops = amountcoffeeshops - 1
    WHERE id_company = OLD.id_company;
    RETURN OLD;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER after_delete_coffeeshop AFTER DELETE ON coffeeshops
FOR EACH ROW EXECUTE FUNCTION decrement_amount_coffeeshops();
```

3.3.3 Создание функций

На листинге 3.17 представлено создание подставляемой табличной функции, которая по идентификатору напитка находит сети кофеен, в меню которых представлен данный напиток.

Листинг 3.17 — Создание функции, которая по идентификатору напитка находит сети кофеен, в меню которых представлен этот напиток

```
CREATE OR REPLACE FUNCTION companies_by_drink(drink_id uuid)
RETURNS TABLE ( id_company uuid, name VARCHAR(128), website VARCHAR(256),
    amountcoffeeshops INT
)
AS $$
BEGIN
RETURN QUERY
    SELECT c.id_company, c.name, c.website, c.amountcoffeeshops
    FROM companies c
    WHERE  c.id_company in (select m.id_company from menu m where
        m.id_drink = drink_id);
END;
$$ LANGUAGE plpgsql;
```

На листинге 3.18 представлено создание хранимой процедуры, которая изменяет права доступа пользователей.

Листинг 3.18 — Создание хранимой процедуры, которая изменяет права доступа пользователей.

```
CREATE OR REPLACE PROCEDURE update_user_rights(user_id UUID, new_role_id
    INT)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE users SET id_role = new_role_id WHERE id_user = user_id;
END;
$$;
```

3.3.4 Создание ролевой модели

На листинге 3.19 представлено создание роли гостя и выдача ему необходимых прав в соответствии с диаграммой прецедентов, представленной на рисунке 1.4.

Листинг 3.19 — Создание роли гостя

```
CREATE ROLE guest WITH LOGIN PASSWORD 'guest2025';
GRANT SELECT, INSERT ON users TO guest;
GRANT SELECT ON roles TO guest;
```

На листинге 3.20 представлено создание роли обычного пользователя и выдача ему необходимых прав в соответствии с диаграммой прецедентов, представленной на рисунке 1.4.

Листинг 3.20 — Создание роли обычного пользователя

```
CREATE ROLE ordinary_user WITH LOGIN PASSWORD 'user2025';
GRANT SELECT, INSERT, DELETE, UPDATE ON users TO ordinary_user;
GRANT SELECT ON drinks TO ordinary_user ;
GRANT SELECT ON companies TO ordinary_user ;
GRANT SELECT ON coffeeshops TO ordinary_user ;
GRANT SELECT ON categories TO ordinary_user ;
GRANT SELECT ON drinkscategory TO ordinary_user ;
GRANT SELECT ON menu TO ordinary_user ;
GRANT SELECT ON loyaltyprograms TO ordinary_user ;
GRANT SELECT, INSERT, DELETE ON favdrinks TO ordinary_user ;
GRANT SELECT,INSERT, DELETE ON favcoffeeshops TO ordinary_user ;
GRANT SELECT ON roles TO ordinary_user ;
```

На листинге 3.21 представлено создание роли модератора и выдача ему необходимых прав в соответствии с диаграммой прецедентов, представленной на рисунке 1.4.

Листинг 3.21 — Создание роли модератора

```
CREATE ROLE moderator WITH LOGIN PASSWORD 'moder2025';
GRANT SELECT, INSERT, DELETE, UPDATE ON users TO moderator;
GRANT SELECT, INSERT, DELETE ON drinks TO moderator ;
GRANT SELECT, UPDATE ON companies TO moderator;
GRANT SELECT, INSERT, DELETE ON coffeeshops TO moderator;
GRANT SELECT, INSERT ON categories TO moderator;
GRANT SELECT, INSERT, DELETE, update ON drinkscategory TO moderator;
GRANT SELECT,INSERT, DELETE ON menu TO moderator;
GRANT SELECT ON loyaltyprograms TO moderator;
GRANT SELECT, INSERT, DELETE ON favdrinks TO moderator;
GRANT SELECT, INSERT, DELETE ON favcoffeeshops TO moderator;
GRANT SELECT ON roles TO moderator;
```

На листинге 3.22 представлено создание роли администратора и выдача ему необходимых прав в соответствии с диаграммой прецедентов, представленной на рисунке 1.4.

Листинг 3.22 — Создание роли администратора

```
CREATE ROLE administrator WITH LOGIN PASSWORD 'admin2025';
GRANT SELECT,INSERT, DELETE ON drinks TO administrator;
GRANT SELECT, UPDATE ON companies TO administrator;
GRANT SELECT,INSERT, DELETE ON coffeeshops TO administrator;
GRANT SELECT, INSERT ON categories TO administrator;
GRANT SELECT, INSERT, DELETE, UPDATE ON drinkscategory TO administrator;
GRANT SELECT,INSERT, DELETE ON menu TO administrator;
GRANT SELECT ON loyaltyprograms TO administrator;
GRANT SELECT, INSERT, DELETE ON favdrinks TO administrator;
GRANT SELECT,INSERT, DELETE ON favcoffeeshops TO administrator;
GRANT SELECT ON roles TO administrator;
GRANT SELECT, INSERT, DELETE, UPDATE ON users TO administrator;
```

3.4 Тестирование

Корректность работы триггеров на удаление записей была проверена путем тестирования. В таблицах 3.2–3.4 описаны тестовые случаи. Все тесты были успешно пройдены.

Таблица 3.2 — Тесты для триггера, который удаляет из таблиц menu, favdrinks и drinkscategory все записи, связанные с удаляемым из таблицы drinks напитком.

Описание теста	**Ожидаемый результат**	**Полученный результат**
Напиток присутствует в связанных таблицах		
Напиток присутствует во всех таблицах menu, favdrinks, drinkscategory	Удалены запись из drinks и все связанные записи из menu, favdrinks и drinkscategory	Удалены запись из drinks и все связанные записи из menu, favdrinks и drinkscategory
Напиток присутствует только в таблице menu	Удалены запись из drinks и menu	Удалены запись из drinks и menu
Напиток присутствует только в таблице drinkscategory	Удалена запись из drinks и удалены и drinkcategory	Удалена запись из drinks и удалены и drinkcategory
Напиток присутствует только в таблице favdrinks	Удалена запись из drinks и удалены и favdrinks	Удалена запись из drinks и удалены и favdrinks
Напиток отсутствует в связанных таблицах		
Напиток отсутствует в таблицах menu, favdrinks, drinkscategory	Удалена только запись из drinks	Удалена только запись из drinks
Некорректный drink_id напитка		
Попытка удаления несуществующего в drinks напитка	Никаких изменений в БД	Никаких изменений в БД
Удаление напитка с id_drink=NULL	Никаких изменений в БД	Никаких изменений в БД

Таблица 3.3 — Тесты для триггера, который удаляет из таблиц favcoffeeshops и favdrinks все записи, связанные с удаляемым из таблицы users пользователем

Описание теста	**Ожидаемый результат**	**Полученный результат**
Пользователь присутствует в связанных таблицах		
Пользователь имеет записи в favcoffeeshops и favdrinks	Удалены запись из users и все связанные записи из favcoffeeshops и favdrinks	Удалены запись из users и все связанные записи из favcoffeeshops и favdrinks
Пользователь имеет записи только в favcoffeeshops	Удалены записи из users и favcoffeeshops	Удалены записи из users и favcoffeeshops
Пользователь имеет записи только в favdrinks	Удалены записи из users и favdrinks	Удалены записи из users и favdrinks
Пользователь отсутствует в связанных таблицах		
Пользователь не имеет записей в favcoffeeshops и favdrinks	Удалена только запись из users	Удалена только запись из users
Некорректный id_user		
Попытка удаления несуществующего в users пользователя	Никаких изменений в БД	Никаких изменений в БД
Удаление пользователя с id_user=NULL	Никаких изменений в БД	Никаких изменений в БД

Таблица 3.4 — Тесты для триггера, который удаляет из таблицы favcoffeeshops все записи, связанные с удаляемой из таблицы coffeeshops кофейней.

Описание теста	**Ожидаемый результат**	**Полученный результат**
Кофейня присутствует в связанной таблице		
Кофейня присутствует в таблице favcoffeeshops	Удалены запись из coffeeshops и все связанные записи из favcoffeeshops	Удалены запись из coffeeshops и все связанные записи из favcoffeeshops
Кофейня отсутствует в связанной таблице		
Кофейня отсутствует в таблице favcoffeeshops	Удалена только запись из coffeeshops	Удалена только запись из coffeeshops
Некорректный id_coffeeshop		
Попытка удаления несуществующей кофейни	Никаких изменений в БД	Никаких изменений в БД
Удаление кофейни с id_coffeeshop=NULL	Никаких изменений в БД	Никаких изменений в БД

Корректность работы триггеров на обновление в таблице coffeeshops значения поля amountcoffeeshops была проверена путем тестирования. В таблицах 3.5–3.6 описаны тестовые случаи. Все тесты были успешно пройдены.

Таблица 3.5 — Тесты для триггера, инкрементирующего значение поля amountcoffeeshops таблицы companies

Описание теста	**Ожидаемый результат**	**Полученный результат**
Добавление новой кофейни, когда для соответствующей записи в companies amountcoffeeshops = 0	amountcoffeeshops стал равным 1	amountcoffeeshops стал равным 1
Добавление новой кофейни, когда для соответствующей записи в companies amountcoffeeshops = 5	amountcoffeeshops стал равным 6	amountcoffeeshops стал равным 6

Таблица 3.6 — Тесты для триггера, декрементирующего значение поля amountcoffeeshops таблицы companies

Описание теста	**Ожидаемый результат**	**Полученный результат**
Успешное инкрементирование поля amountcoffeeshops		
Удаление кофейни, когда для соответствующей записи в companies amountcoffeeshops = 1	amountcoffeeshops стал равным 0	amountcoffeeshops стал равным 0
Удаление кофейни, когда для соответствующей записи в companies amountcoffeeshops = 4	amountcoffeeshops стал равным 3	amountcoffeeshops стал равным 3
Граничные случаи		
Удаление кофейни, когда для соответствующей записи в companies amountcoffeeshops = 0	Значение amountcoffeeshops остаётся равным 0	Значение amountcoffeeshops остаётся равным 0

Корректность работы функции companies_by_drink была проверена путем тестирования. В таблице 3.7 описаны тестовые случаи. Все тесты были успешно пройдены.

Таблица 3.7 — Тесты для функции companies_by_drink

Описание теста	**Ожидаемый результат**	**Полученный результат**
Напиток принадлежит меню хотя бы одной компании		
Напиток есть в меню у одной компании	Возвращается 1 запись с данными компании	Возвращается 1 запись с данными компании
Напиток есть в меню у 3 компаний	Возвращается 3 записи с данными компаний	Возвращается 3 записи с данными компаний
Напиток не принадлежит ни одной компании		
Напиток существует в drinks, но отсутствует в таблице menu	Возвращается пустой результат (0 записей)	Возвращается пустой результат (0 записей)
Функция вызывна для некорректного id_drink		
Передан несуществующий drink_id	Возвращается пустой результат (0 записей)	Возвращается пустой результат (0 записей)
Передан NULL в качестве drink_id	Возвращается пустой результат (0 записей)	Возвращается пустой результат (0 записей)

Корректность работы процедуры update_user_rights, изменяющей права доступа пользователя, была проверена путем тестирования. В таблице 3.8 описаны тестовые случаи. Все тесты были успешно пройдены.

Таблица 3.8 — Тесты для процедуры update_user_rights

Описание теста	**Ожидаемый результат**	**Полученный результат**
В процедуру переданы корректные параметры		
Обновление роли пользователя с id_user, сущ. в users, на new_role_id, сущ. в roles	id_role изменён на new_role_id	id_role изменён на new_role_id
Обновление роли пользователя с id_user, сущ. в users, на значение new_role_id равное текущему id_role	id_role остался прежним	id_role остался прежним
В процедуру переданы некорректные параметры		
Передан несуществующий user_id	0 строк обновлено	0 строк обновлено
Передан несуществующий new_role_id	Ошибка внешнего ключа	Ошибка внешнего ключа

3.5 Интерфейс приложения

Интерфейс веб-приложения разделен на страницы, каждая из которых предоставляет пользователю некоторые опции. На рисунках 3.1–3.11 продемонстрирован интерфейс разработанного приложения.

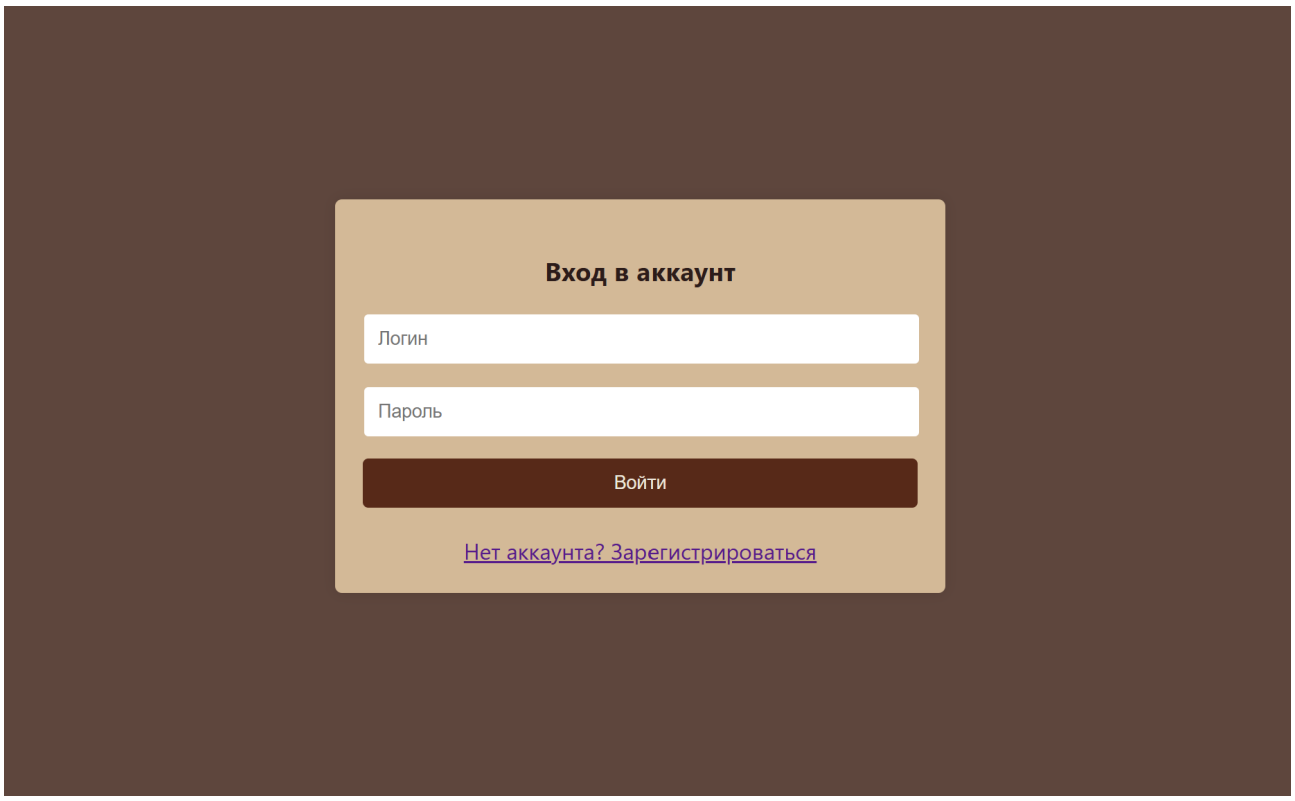

Рисунок 3.1 — Страница авторизации

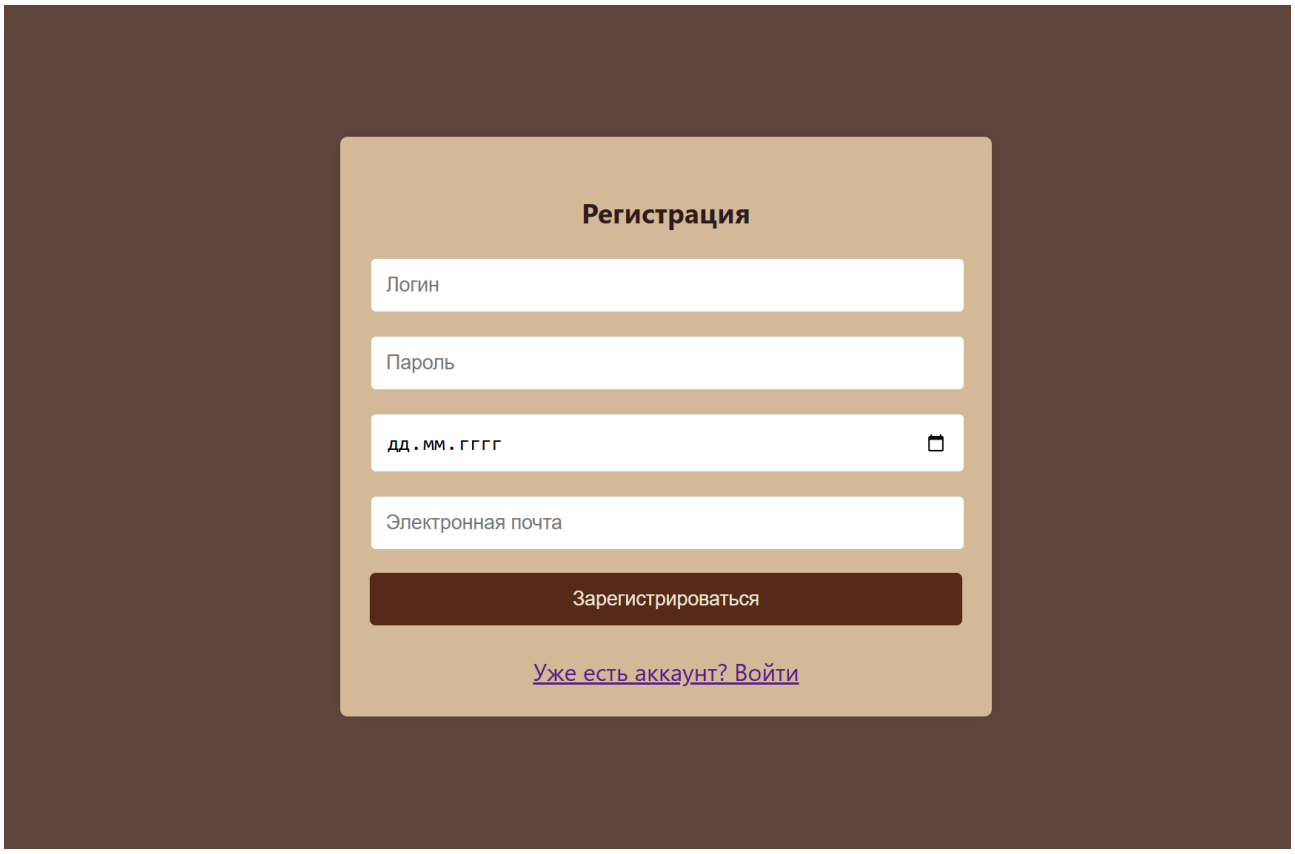

Рисунок 3.2 — Страница регистрации

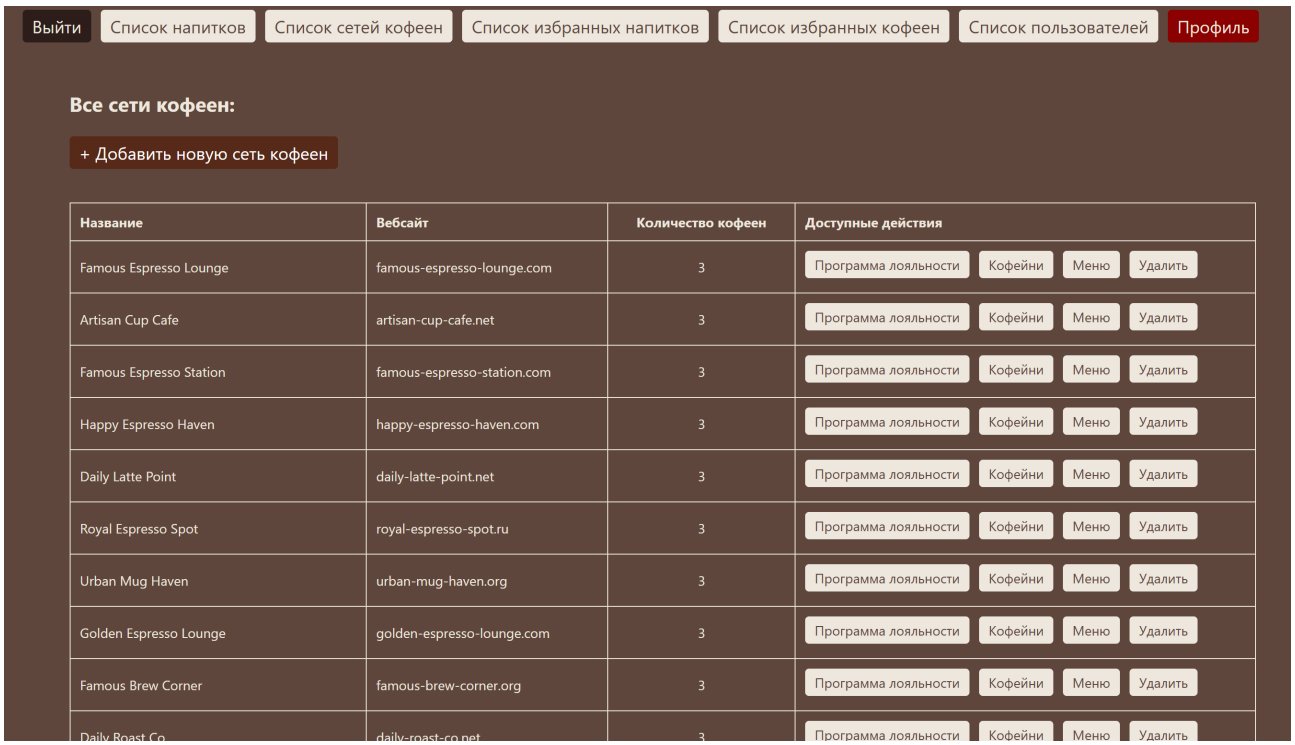

Рисунок 3.3 — Страница с информацией о сетях кофеен (для обычных пользователей кнопки добавления и удаления сетей кофеен недоступны)

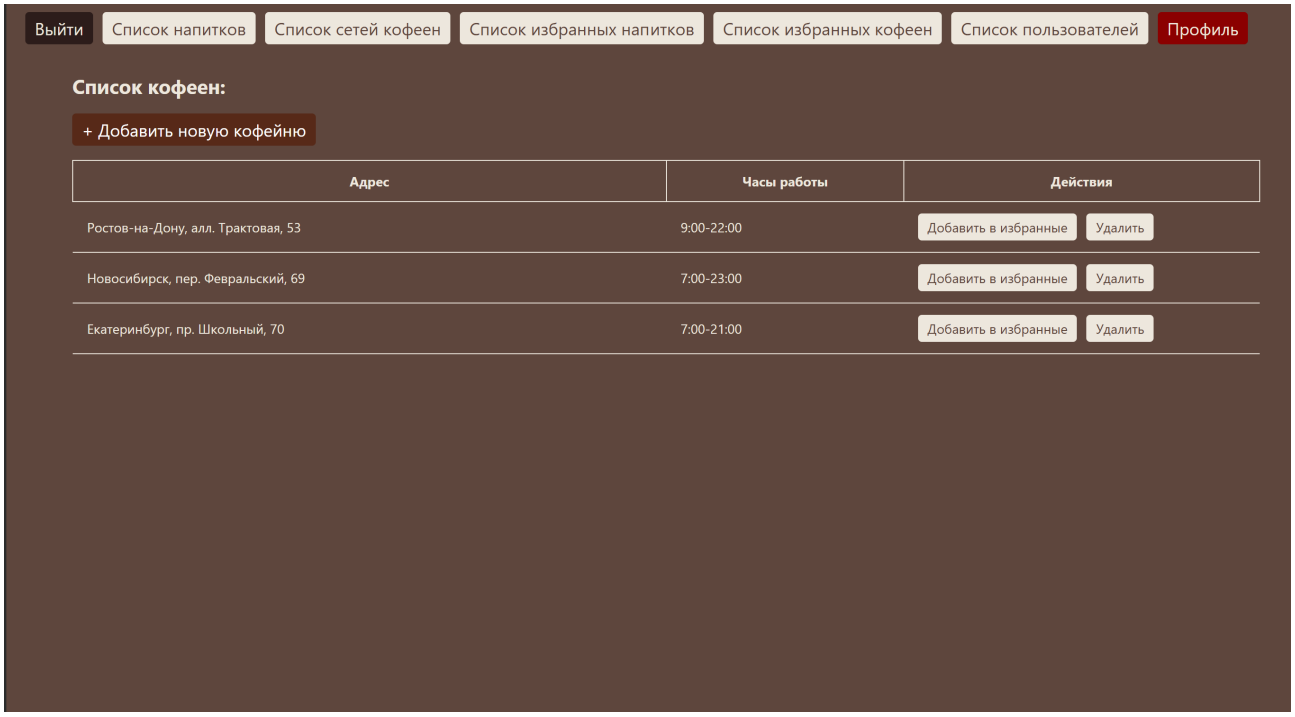

Рисунок 3.4 — Страница с информацией о кофейнях конкретной сети (для обычных пользователей кнопки добавления и удаления кофеен недоступны)

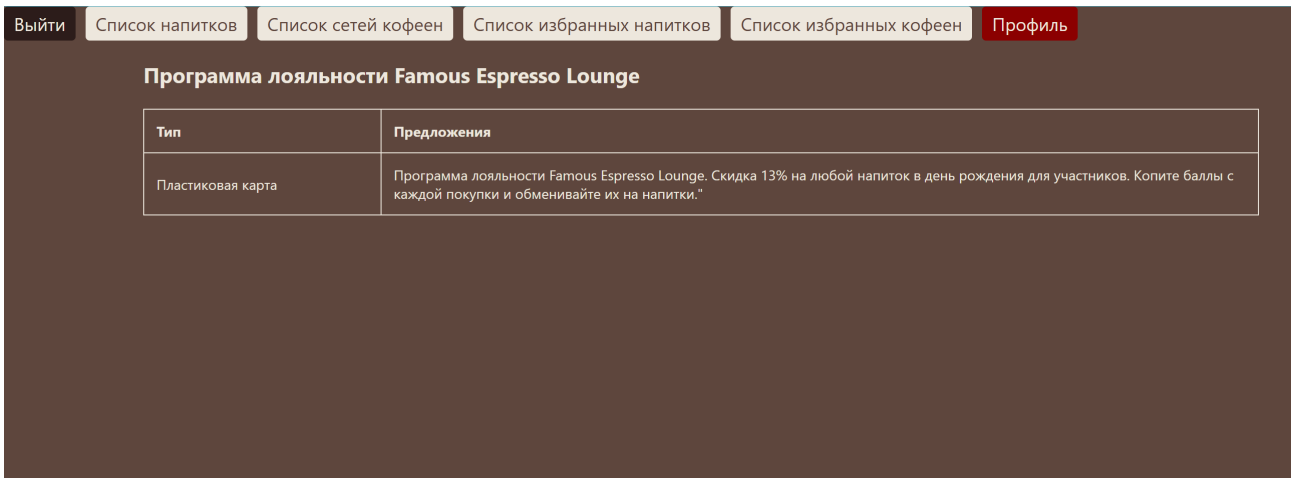

Рисунок 3.5 — Страница с информацией о программе лояльности конкретной сети кофеен

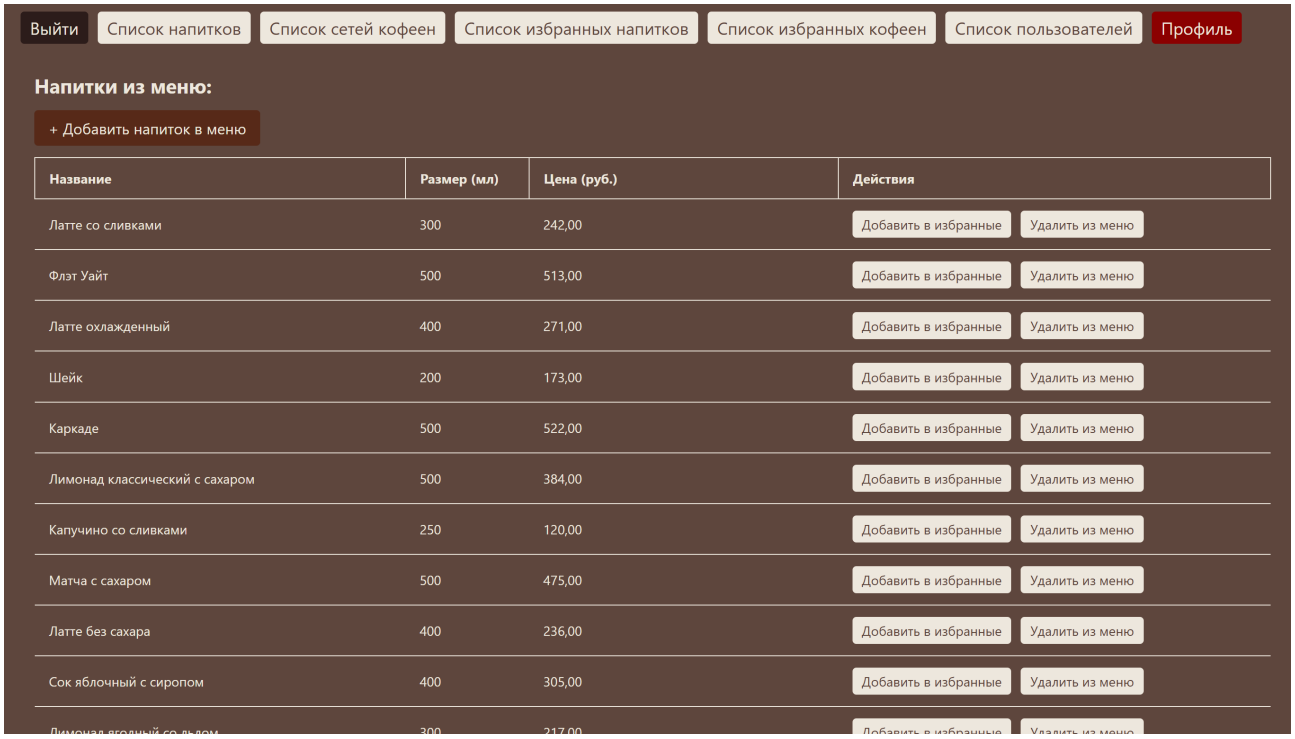

Рисунок 3.6 — Страница с меню конкретной сети кофеен (для обычных пользователей кнопки добавления и удаления позиций меню недоступны)

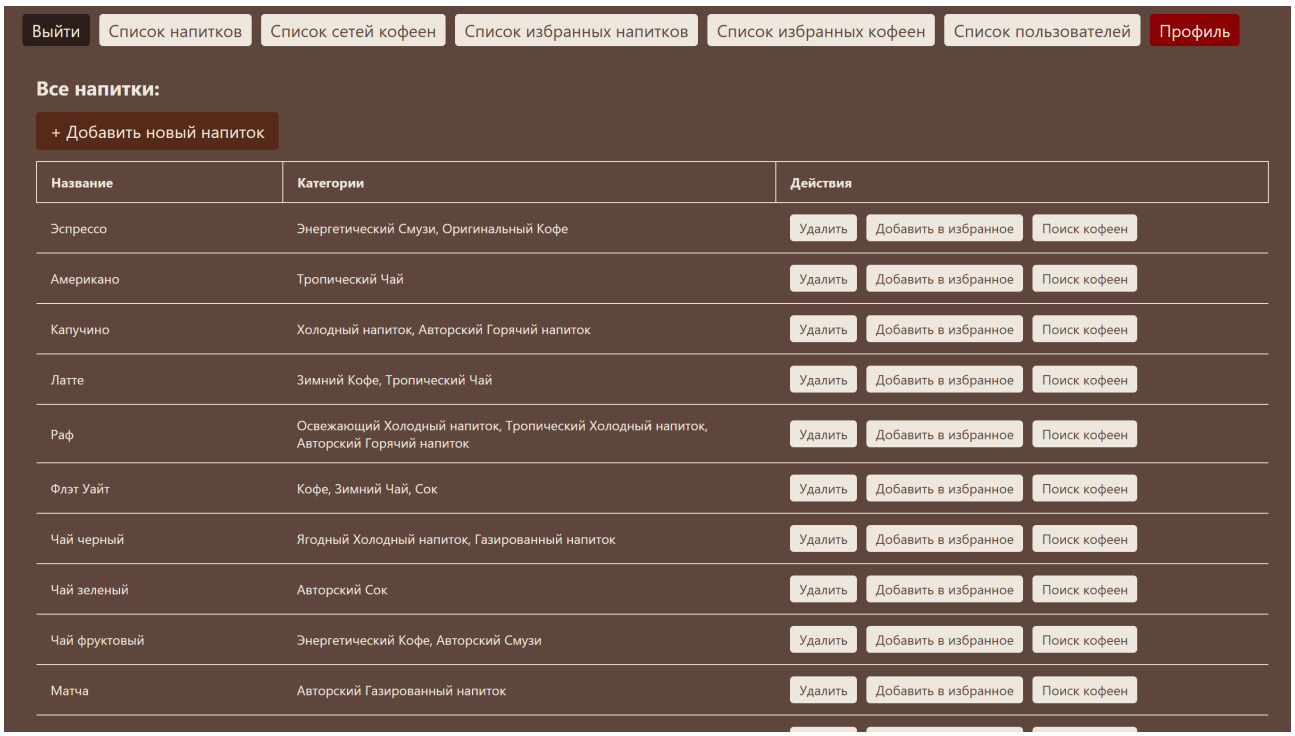

Рисунок 3.7 — Страница с всеми напитками, имеющимися в базе данных (для обычных пользователей кнопки добавления и удаления напитков недоступны)

Рисунок 3.8 — Страница с избранными напитками

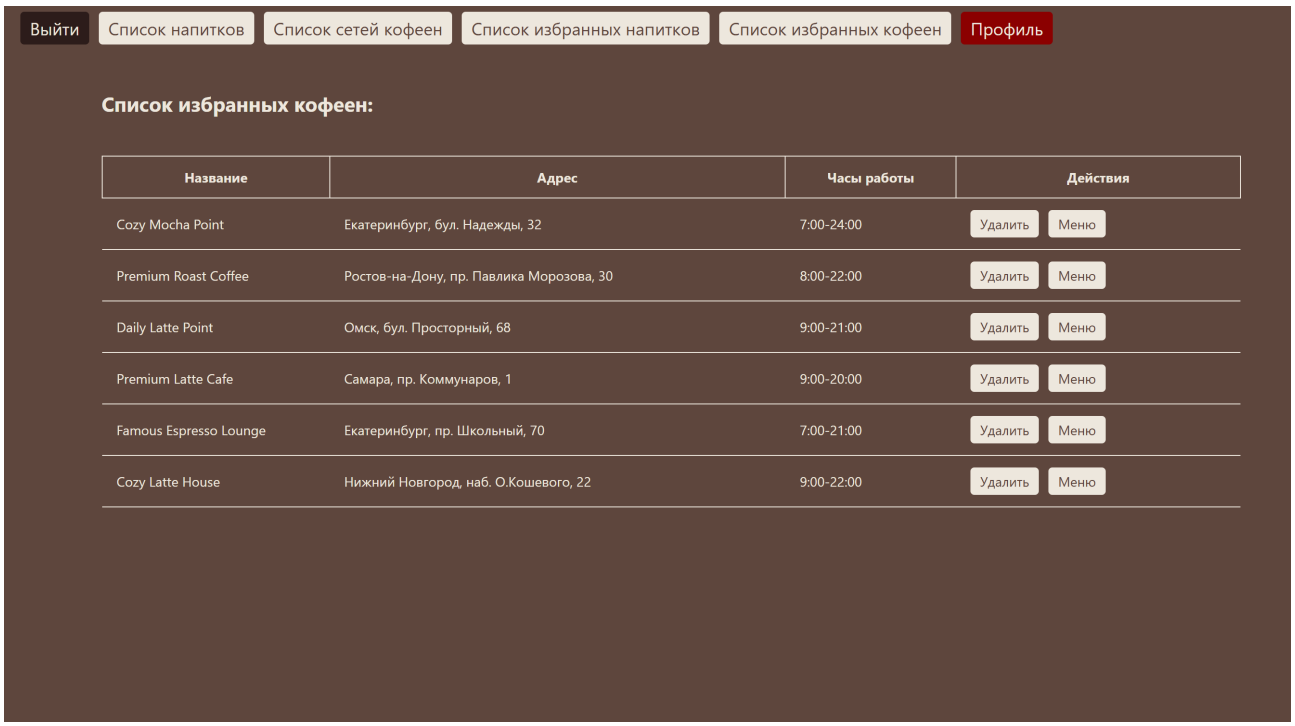

Рисунок 3.9 — Страница с избранными кофейнями

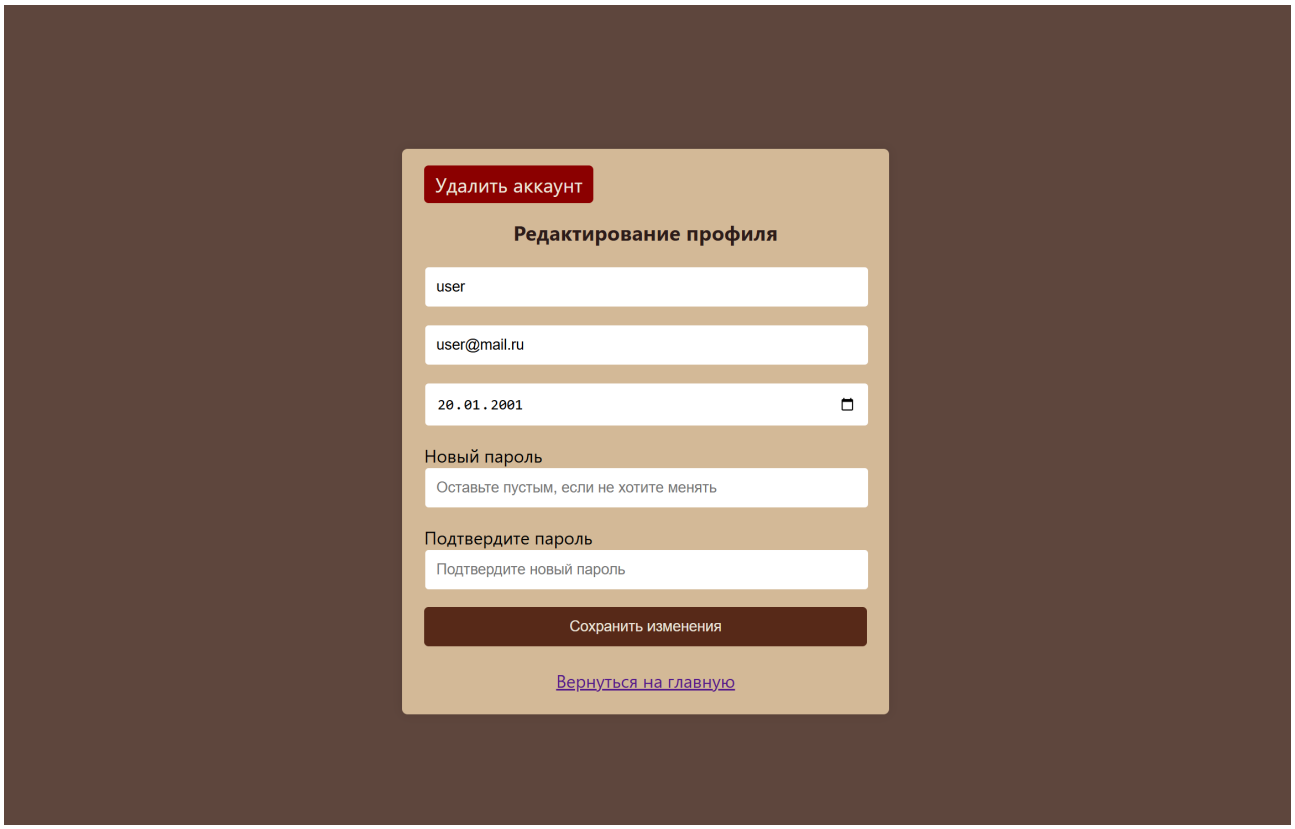

Рисунок 3.10 — Страница с изменением данных профиля

Рисунок 3.11 — Страница с информацией о пользователях (доступна только администраторам)

Вывод

В данной части было проведено сравнение СУБД, в результате чего выбрана PostgreSQL. Также были разработаны описанные ранее объекты базы данных и приложение для доступа к ней.

4 Исследовательская часть

В данной части будет проведено исследование влияния индекса на время выполнения запросов к базе данных.

4.1 Описание исследования

Индекс – это объект базы данных, который обеспечивает дополнительные способы поиска и извлечения данных. Индекс может создаваться на одном или нескольких столбцах.

Для исследовании будет использован тип индекса B–дерево. B–дерево является самой распространенной индексной структурой и состоит из иерархически организованных узлов, связанных с блоками, хранящимися на диске [15].

Замеры времени выполнения запросов будут проводиться с использованием функции $time.perf\_counter$ [16] языка $Python$ [17].

4.2 Технические характеристики устройства

Технические характеристики устройства, на котором проводилось исследование:

— операционная система Windows 11 64-разрядная операционная система;

— оперативная память 16 ГБ;

— процессор AMD Ryzen 5 5600U with Radeon Graphics 2.30 ГГц.

4.3 Результаты исследования

4.3.1 Поиск по полю с фильтрацией

Для создания индекса использовалась команда:

```
CREATE INDEX index_users_birthdate ON users using btree(birthdate)
```

В таблице 4.1 представлены результаты замеров времени выполнения запроса в миллисекундах.

Таблица 4.1 — Время выполнения запроса в миллисекундах

Количество записей в таблице	**Без индекса**	**С индексом**
1000	0.178	0.139
5000	0.398	0.168
10000	0.705	0.172
25000	1.601	0.169
50000	3.213	0.184
100000	7.403	0.173
250000	17.551	0.176

На рисунке 4.1 приведены результаты замеров времени выполнения запроса в виде графика.

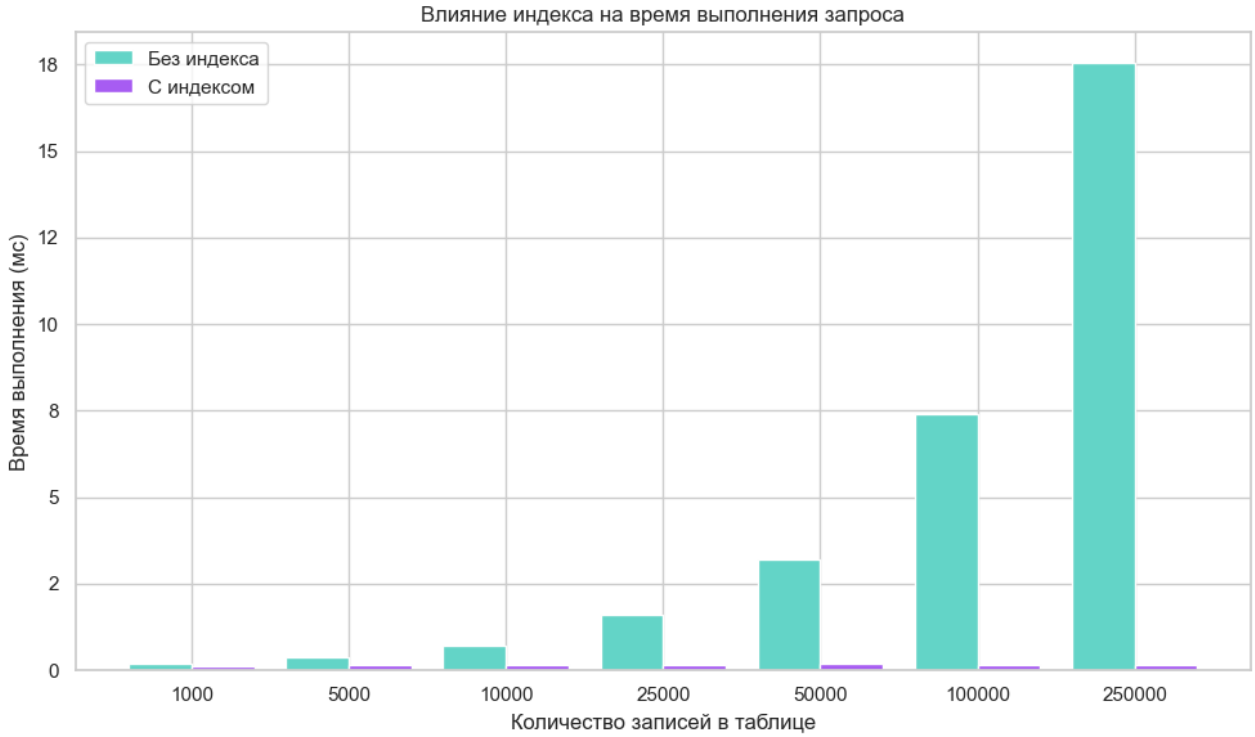

Рисунок 4.1 — Исследование влияния индекса на время выполнения запроса

По результатам исследования сделан вывод о том, что использование индекса позволяет уменьшить время выполнения запросов к базе данных. Согласно полученным замерам, время выполнения запроса максимально сократилось примерно в 100 раз. Такой результат связан с тем, что при отсутствии индекса поиск нужных данных в таблице выполняется простым сканированием всех ее строк.

4.3.2 Поиск по первичному ключу

Для создания индекса использовалась команда:

```
CREATE INDEX index_users_id ON users using btree(id_user)
```

В таблице 4.2 представлены результаты замеров времени выполнения в миллисекундах.

Таблица 4.2 — Время выполнения запроса в миллисекундах

Количество записей в таблице	**Без индекса**	С индексом
1000	0.105	0.149
5000	0.100	0.151
10000	0.119	0.170
25000	0.107	0.160
50000	0.120	0.171
100000	0.118	0.161
250000	0.135	0.154

На рисунке 4.2 приведены результаты замеров времени выполнения запроса в виде графика.

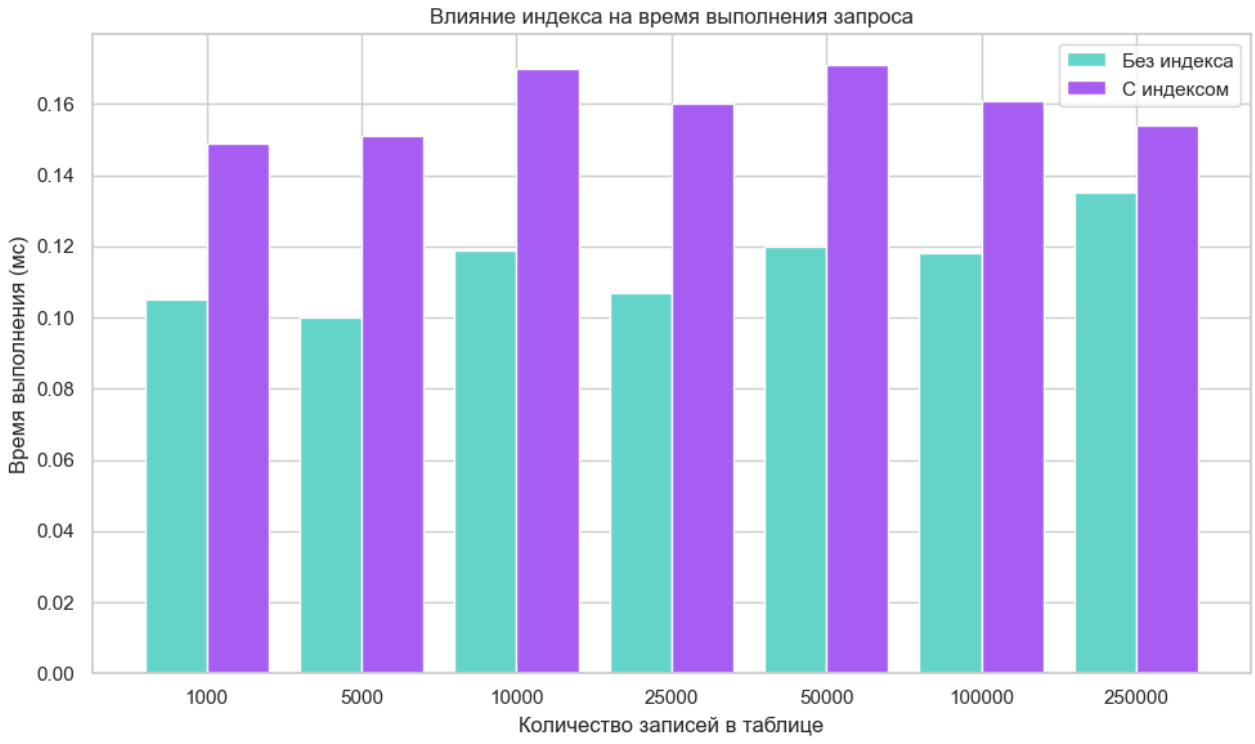

Рисунок 4.2 — Исследование влияния индекса на время выполнения запроса

По результатам исследования сделан вывод о том, что использование индекса не сократило время выполнения запроса. Полученный результат связан с тем, что при создании таблиц для столбца первичного ключа автоматически создается индекс и создание дополнительного индекса на данное поле избыточно.

Вывод

В данной части было проведено исследование влияния индекса на время выполнения запросов к базе данных. По результатам проведенных замеров сделан вывод о том, что наличие индекса позволяет уменьшить время поиска нужных строк в таблице. В то же время создание индексов для первичных ключей не приводит к повышению эффективности, поскольку индексы для этих полей создаются автоматически.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы была достигнута поставленная цель: разработана база данных для хранения информации о кофейнях и напитках из их меню и приложение, предоставляющее интерфейс для доступа к ней.

Также были решены все поставленные задачи:

— проанализирована предметная область;

— формализована задача;

— сформулировано описание пользователей проектируемого приложения;

— спроектированы сущности базы данных и ролевая модель на уровне базы данных;

— создана и заполнена база данных;

— разработано приложение для доступа к базе данных;

— исследовано влияние индекса на время выполнения запросов к базе данных.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Кофейная карта Москвы [Электронный ресурс]. – Режим доступа: https://coffeemap.ru/ (дата обращения: 14.03.2025).

2. Coffee Forest [Электронный ресурс]. – Режим доступа: https://coffeeforest.ru/main_map/ (дата обращения: 14.03.2025).

3. Яндекс Карты [Электронный ресурс]. – Режим доступа: https://yandex.ru/maps/ (дата обращения: 14.03.2025).

4. Стружкин Н.П. Базы данных: проектирование учебник для вузов / Н. П. Стружкин, В. В. Годин. – Москва: Издательство Юрайт, 2025 – 477 с. ISBN 978-5-534-00229-4

5. Илюшечкин В.М. Основы проектирования баз данных: учебник для вузов – Москва: Издательство Юрайт, 2025 – 213 с. ISBN 978-5-5 34-03617-6

6. Маркин А.В. Постреляционные базы данных. MongoDB : учебное пособие. – 3-е изд. – Москва : Ай Пи Ар Медиа, 2024. – 383с. ISBN 978-5-4497-3243-9

7. Документация PostgreSQL [Электронный ресурс]. — Режим доступа: https://www.postgresql.org/docs/ (дата обращения: 25.04.2025).

8. Документация MySQL [Электронный ресурс]. — Режим доступа: https://dev.mysql.com/doc/ (дата обращения: 25.04.2025).

9. Документация Microsoft SQL Server [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/en-us/sql/?view=sql-server-ver17 (дата обращения: 25.04.2025).

10. Документация Oracle [Электронный ресурс]. — Режим доступа: https://docs.oracle.com/en/database/oracle/oracle-database/index.html (дата обращения: 25.04.2025).

11. Документация языка программирования C# [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/en-us/dotnet/csharp/ (дата обращения: 25.04.2025).

12. Документация .NET [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/en-us/dotnet/ (дата обращения: 25.04.2025)

13. Документация Npgsql [Электронный ресурс]. — Режим доступа: https://www.npgsql.org/ (дата обращения: 25.04.2025)

14. Документация Visual Studio [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/ru-ru/visualstudio/windows/?view=vs-2022 (дата обращения: 25.04.2025).

15. Домбровская Г., Новиков Б., Бейликова А. Оптимизация запросов в PostgreSQL / пер. с англ. Д.А. Беликова. – Москва: ДМК Пресс, 2022 – 278 с. ISBN 978-5-97060-963-7

16. Документация модуля time – Python [Электронный ресурс]. — Режим доступа: https://docs.python.org/3.12/library/time.html (дата обращения: 25.04.2025)

17. Документация языка программирования Python [Электронный ресурс]. — Режим доступа: https://docs.python.org/3.12/index.html (дата обращения: 25.04.2025)

ПРИЛОЖЕНИЕ А

Презентация к курсовой работе состоит из 14 слайдов.