РЕФЕРАТ

Расчетно-пояснительная записка 68 с., 17 рис., 18 табл., 37 источн., 1 прил.

Ключевые слова: базы данных, продуктовая корзина, акции на товары, сертификаты на товары, сравнение цены на товар.

Объектом разработки является база данных.

Цель работы – разработка базы данных для сравнительного анализа цен на элементы продуктовой корзины первой необходимости.

В аналитической части (1) был проведен анализ предметной области, были рассмотрены существующие решения, формализованы роли пользователей и описаны возможные действия для каждой роли. Также была создана ER-диаграмма проектируемой базы данных и выбрана реляционная СУБД исходя из их классификации по модели данных.

В конструкторской части (2) была представленая ER-модель разрабатываемой базы данных, описаны сущности и ограничения целостности. Были разработаны after триггер для пересчета среднего рейтинга продаваемого товара и функция для получения среднего рейтинга продаваемого товара.

В технологической части (3) были выбраны СУБД PostgreSQL для реализации базы данных, язык Golang для написания приложения для взаимодействия с базой данных, среда разработки. Были приведены листинги команд для создания базы данных, таблиц, ограничений целостности. Также был описано консольный интерфейс приложения для взаимодействия с базой данных.

В исследовательской части (4) было проведено исследование зависимости среднего времени ответа от числа запросов в секунду с использованием кеширования приложения и без использования кеширования приложения. Результаты исследования показали, что использование кеширования приложения увеличивает число обрабатываемых запросов в секунду в ≈ 2.46 и уменьшает среднее время ответа ≈ 1.71 раз, а использование индекса с ростом числа записей в таблице сокращает время выполнения запроса $\approx$ в 2.77 раза.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Клиенты всегда в поиске выгодных предложений [1]. Современный рынок предлагает широкий спектр продовольственных товаров, и среди этого многообразия бывает сложно определить оптимальное предложение по цене. Для решения этой проблемы создаются сервисы сравнения цен, которые позволяют удобным образом сравнивать цены на товары в различных магазинах [2].

На сегодняшний день пользователи все чаще используют сервисы сравнения цен на товары [3]. Эти платформы не только помогают сэкономить деньги, но и позволяют находить альтернативные продукты, которые могут быть более качественными или полезными. В результате, такие сервисы становятся важным инструментом для осознанного потребления и рационального выбора.

Целью данного курсового проекта является разработка базы данных для сравнительного анализа цен на элементы продуктовой корзины первой необходимости.

Для достижения поставленной цели необходимо решить следующие задачи:

1) провести анализ предметной области;
2) проанализировать существующие решения;
3) спроектировать базу данных, описать ее сущности и связи между сущностями;
4) выбрать подходящие средства реализации;
5) реализовать базу данных;
6) провести исследования созданной базы данных.

1 Аналитический раздел

В данном разделе будет проведен анализ предметной области, будет проведено сравнение существующих решений. Также будет проведена формализация и описание информации, подлежащей хранению в проектируемой базе данных, будут выделены типы пользователей проектируемого приложения к базе данных, а также будет выбрана модель базы данных.

1.1 Анализ предметной области

В данном подразделе будет формально описан список продуктов первой необходимости, будут определены основные типы акций на продукты и основные участники в цепи поставок товаров. Также будет описан «Меркурий» — автоматизированная система для отслеживания продуктов питания на всей цепи их производства и будут формально описаны виды сертификатов на товары.

1.1.1 Элементы продуктовой корзины первой необходимости

Согласно постановлению Правительства № 530 [4], элементами продуктовой корзины первой необходимости являются следующие продукты:

- говядина, баранина, свинина (кроме бескостного мяса);
- куры (кроме куриных окорочков);
- рыба мороженая неразделанная;
- масло сливочное;
- масло подсолнечное;
- молоко питьевое;
- яйца куриные;
- сахар-песок;
- соль поваренная пищевая;

– чай черный байховый;

– мука пшеничная;

– хлеб ржаной, ржано-пшеничный;

– хлеб и булочные изделия из пшеничной муки;

– рис шлифованный;

– пшено;

– крупа гречневая – ядрица;

– вермишель;

– картофель;

– капуста белокочанная свежая;

– лук репчатый;

– морковь;

– яблоки.

1.1.2 Акции на товары

На данный момент рынок розничной торговли очень насыщен. С каждым днем среди производителей конкуренция растет высокими темпами. Одной рекламы порой бывает не достаточно, чтобы привлечь внимание к производителю и его продукции. Чтобы хоть как то выделиться среди многочисленных фирм и предприятий, компаниям необходимо прибегать к методам стимулирования сбыта продукции [5].

Одна из форм стимулирования — **ценовая** [5]. Ее суть заключается в установлении скидки на определенный товар или группу товаров. Можно выделить 5 основных типов акций со снижением цены [6; 7]:

– скидка «товар дня»;

– скидка за объем, в том числе по принципу «1+1=3»;

– скидка отдельным категориям покупателей;

– скидка, приуроченная к определенному событию;

– сезонная распродажа.

1.1.3 Цепь поставок

Предприятия, принимающие участие в создании, распространении и реализации товаров, взаимодействуют в рамках **цепи поставок**. Они рассматриваются как изолированные элементы, которые самостоятельно планируют свои потребности и покупки с взаимодействуют друг с другом [8].

Основными участниками цепи поставок являются:

1) **ритейлер** – организация, которая продает товар конечным потребителям, реализуя его в торговых точках. Ритейлер закупает товар у дистрибьютора;

2) **дистрибьютор** – посредник, который покупает товар у производителя оптовыми партиями для последующей продажи розничным продавцам;

3) **производитель** – организация, создающая товары или услуги, используя сырьевые материалы и компоненты, а также отвечающая за первоначальное распределение продукции.

1.1.4 ФГИС «Меркурий»

ФГИС «Меркурий» — автоматизированная система, предназначенная для отслеживания продуктов питания на всей цепи производства и перемещения до точки реализации [9]. Работа с «Меркурием» заключается в создании и «гашении» **ветеринарно-сопроводительных документов (ВСД)** на всех этапах движения товара: от производства и переработки до продажи или утилизации.

Создание ФГИС «Меркурий» позволило достичь следующих целей [10]:

– защита потребителя от некачественной и небезопасной продукции, а все население страны от экономических и социальных угроз;

– обеспечение прозрачности и эффективности действий надзорных органов в борьбе с мошенничеством;

– минимизация бюрократии и предоставление удобного прозрачного механизма для комфортной работы частного бизнеса.

Помимо ВСД существует еще 3 типа разрешительных документов на пищевую продукцию [11]:

– декларация о соответствии на пищевую продукцию (ДС);

– добровольный сертификат на пищевую продукцию;

– свидетельство о государственной регистрации пищевой продукции (СГР на пищевую продукцию).

1.2 Существующие решения

На рынке существует большое количество сервисов для мониторинга цен на продукты в различных магазинах. Наиболее популярными являются:

– «Едадил» [12];

– SkidkaOnline [13];

– Price.ru [14].

Сравнение существующих решений было произведено в таблице 1.1:

Таблица 1.1 – Сравнение существующих решений

Критерии	«Едадил»	SkidkaOnline	Price.ru
возможность сравнения цены на конкретный товар в разных магазинах	+	-	+
возможность оставить отзыв о товаре	+	+	+
наличие информации об акциях на товары	+	+	+
возможность просмотра сертификатов соответствия конкретного товара	-	-	-

1.3 Формализация и описание информации, подлежащей хранению в проектируемой базе данных

На основе анализа предметной области, в разрабатываемой базе данных были выделены сущности, приведенные в таблице 1.2:

Таблица 1.2 – Выделенные сущности предметной области и их описание

Сущность	**Сведения**
Пользователь	ФИО, номер телефона, пароль, дата регистрации
Магазин	Название, телефон, адрес, ФИО директора
Товар	Название, категория, бренд, состав, масса брутто, масса нетто, тип упаковки
Сертификат соответствия	Тип сертификата, номер сертификата, нормативный документ, дата регистрации сертификата, дата окончания действия сертификата, статус соответствия
Акция	Тип акции, дата начала, дата конца, размер скидки, описание
Оценка товара	Отзыв, рейтинг
Производитель	Название, адрес, номер телефона, ФИО представителя
Дистрибьютор	Название, адрес, номер телефона, ФИО представителя
Ритейлер	Название, адрес, номер телефона, ФИО представителя

На рисунке 1.1 представлена ER-диаграмма сущностей проектируемой базы данных в нотации Чена:

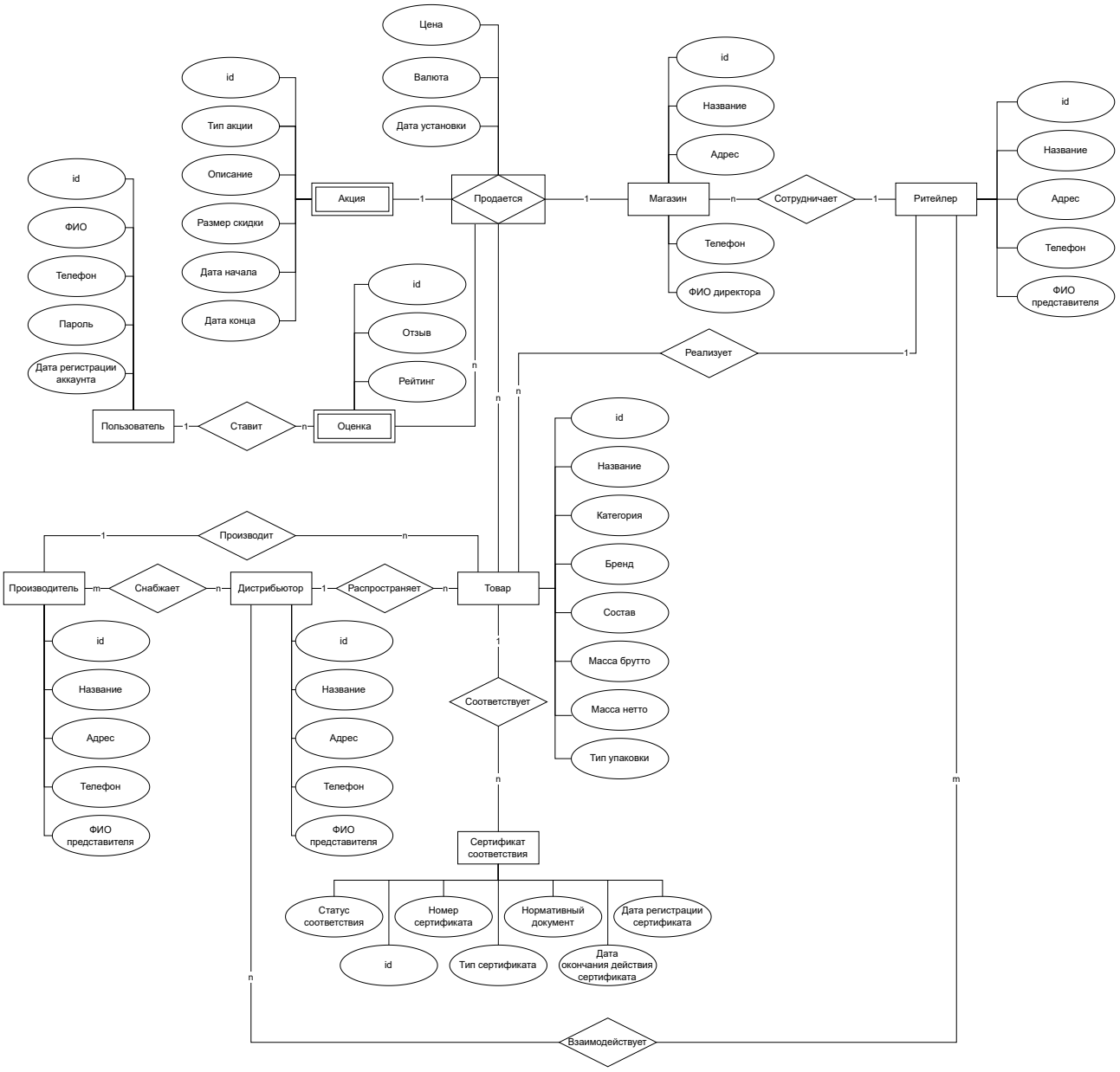

Рисунок 1.1 – ER-диаграмма сущностей проектируемой базы данных в нотации Чена

1.4 Формализация и описание пользователей проектируемого приложения к базе данных

В соответствии выделенными в разрабатываемой базе данных сущностями выделяется 3 типа пользователей:

Таблица 1.3 – Типы пользователей и их описание

Тип пользователя	**Описание**
Гость	Может просматривать всю информацию о товарах, сравнивать цены.
Зарегистрированный пользователь	Может все то же, что и гость, а также ставить оценки на товары, добавлять новые товары, вносить изменения в цену товара, а также добавлять новые магазины.
Администратор	Управляет всей системой. Может все то же, что и гость, а также удалять магазины, добавлять/удалять сертификаты на товары.

На рисунке 1.2 представлены диаграммы вариантов использования для различных пользователей системы:

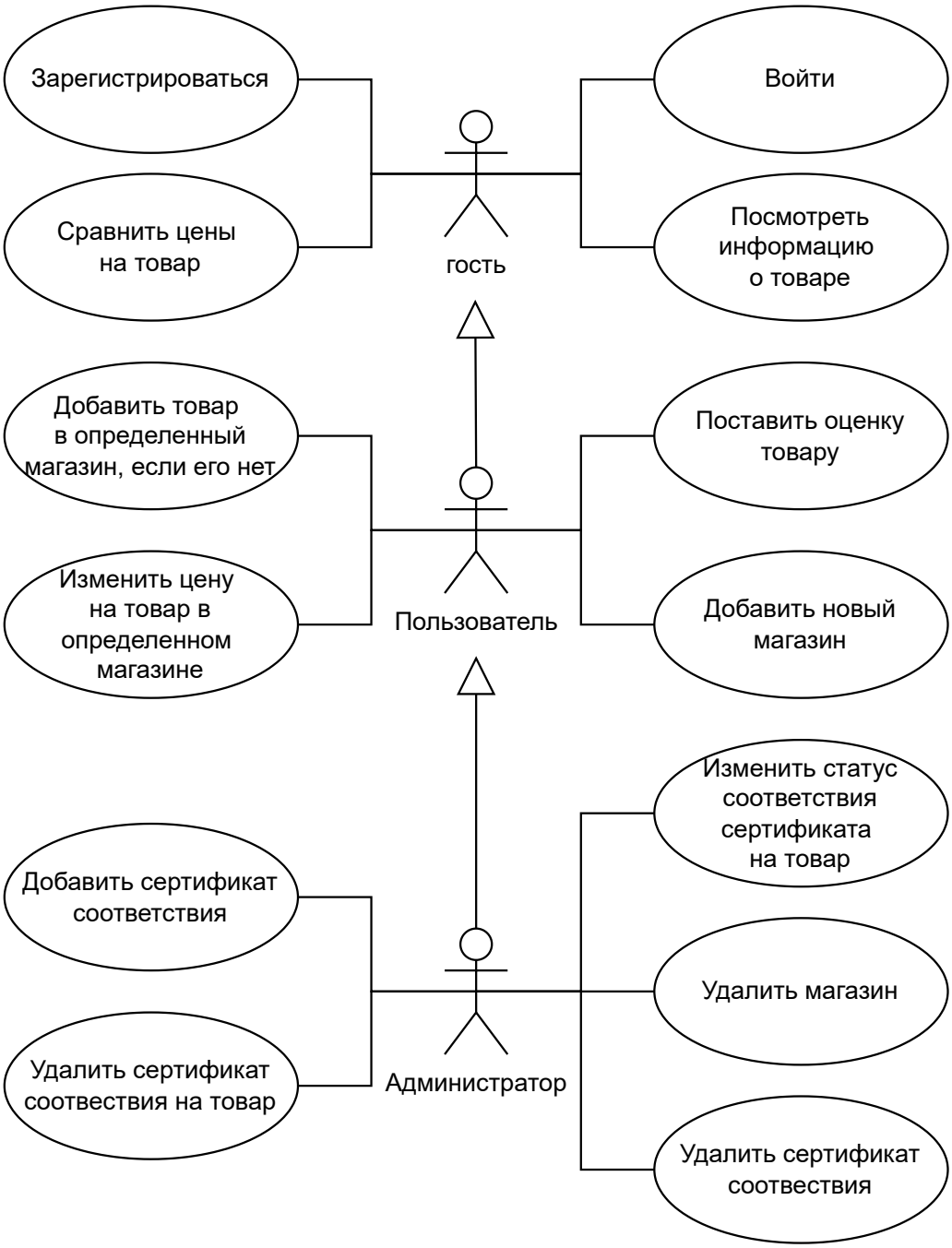

Рисунок 1.2 – Диаграмма вариантов использования для различных пользователей системы

1.5 Классификация и выбор СУБД по модели данных

СУБД — приложение, обеспечивающее создание, хранение, обновление и поиск информации в базах данных.

Модель данных — это абстрактное и логическое определение объектов и его поведение, в совокупности составляющих доступ к данным, с которой взаимодействует пользователь [15]. С помощью модели данных могут быть представлены объекты предметной области и взаимосвязи между ними.

По модели данных СУБД разделяются на:

1) дореляционные модели, которые, в свою очередь, делятся на:
   - инвертированные списки;
   - иерархические;
   - сетевые.
2) реляционные модели данных;
3) постреляционные модели данных.

1.5.1 Дореляционные базы данных

Дореляционные БД подразделяются на инвертированные списки, иерархические БД и сетевые БД.

БД на основе инвертированных списков состоит из двух областей: основная и индексная области. Основная область представляет собой наборы файлов с записями. Записи в файле имеют фиксированную длину и расположены либо в произвольном порядке, либо упорядочены в соответствии с физической организацией хранения данных. Индексная область представляет собой индексные файлы – файлы, содержащие значения ключей поиска и соответствующие этим значениям номера записей из файлов основной области [16]. В такой системе отсутствуют механизмы поддержания целостности хранимых данных, эта работа возлагается на программу, которая работает с такой БД.

Иерархическая БД состоих из трех основных элементов: физическая база данных, сегмент и поле. Поле представляет собой минимальную, неделимую единицу данных. Сегмент – запись в базе данных, состоящая из полей. Для

идентификации отдельного сегмента используется некоторый набор ключевых полей. В такой модели сегменты образуют ациклический древовидный граф, называемый физической базой данных. Физическая база данных должна удовлетворять следующим ограничениям [16]:

- в каждой физической БД должен существовать только один корневой сегмент;
- сегмент-предок может быть связан с произвольным числом сегментов-потомков;
- сегмент-потомок может быть связан только с одним сегментом-предком.

Благодаря данным ограничениям осуществляется поддержка ссылочной целостности между сегментами-предками и сегментами-потомками, однако нет поддержки целостности данных между сегментами, не входящими в одну иерархию.

Сетевая модель БД есть расширение иерархической: в иерархических структурах запись-потомок должна иметь в точности одного предка; в сетевой структуре данных потомок может иметь любое число предков[17]. Основные термины сетевой модели баз данных включают элемент (узел) и связь. Узел представляет собой набор атрибутов, описывающих определённый объект. Сетевые базы данных можно визуализировать в виде графа. Логика извлечения данных в сетевой БД зависит от их физической организации, что делает эту модель не полностью независимой от приложения. Иными словами, при необходимости изменения структуры данных потребуется также внести коррективы в само приложение [18].

Преимущество дореляционных БД состоит в том, что они позволяют управлять данными на низком уровне. Недостатком является необходимость знания физической организации данных и зависимость прикладных систем от этой организации [17].

1.5.2 Реляционные базы данных

Реляционная модель состоит из следующих трех частей:

1) структурная — описывает, из каких объектов состоит реляционная модель. Определяется, что единственной структурой данных, используемой в реляционных БД, являются нормализованное n-арное отношение [17];

2) целостная — отношения должны удовлетворять определенным условиям целостности [18]:

   – целостность сущности – любой кортеж любого отношения должен отличаться от любого другого кортежа этого же отношения;

   – ссылочная целостность – для каждого значения внешнего ключа, присутствующего в дочернем отношении, в родительском отношении должен существовать кортеж с соответствующим значением первичного ключа.

3) манипуляционная [18] — манипулирования отношениями осуществляется средствами реляционной алгебры и/или реляционного исчисления.

Основными достоинствами реляционных БД является наличие небольшого набора абстракций, простого и в то же время мощного математического аппарата, возможность ненавигационного манипулирования данными без необходимости знания конкретной физической организации баз данных во внешней памяти [17].

1.5.3 Постреляционные базы данных

Постреляционная модель в общем случае есть расширение классической реляционной модели. В такой модели используются структуры, позволяющие хранить в полях таблицы другие таблицы, а в качестве языка запросов используется расширенный SQL [19].

Преимуществом такой модели данных является возможность представления связанных реляционных таблиц одной постреляционной таблицей, что расширяет возможности описания сложных предметов реального мира, а недостатком – сложность в обеспечении целостности данных [20].

Вывод

Таблица 1.4 – Сравнение баз данных по модели данных

Модель данных	Обеспечение целостности сущностей	Обеспечение ссылочной целостности	Независимость от физической организации данных
Дореляционная	+	+	-
Реляционная	+	+	+
Постреляционная	-	+	+

В данном разделе была проанализирована предметная область, рассмотрены существующие решения. На основе анализа предметной области была формализована задача и данные, описаны типы пользователей.

Для разрабатываемой базы данных необходимо обеспечение целостности сущностей, целостных связей между сущностями, а также независимость от физической организации данных. Дореляционные модели данных не подходят, поскольку они зависимы от физической организации хранения данных, а в моделях на основе инвертированных списков отсутствуют ограничения целостности. Не подходит также и постреляционная модель, поскольку в этой модели возникают сложности с обеспечением целостного хранения данных из-за отмены ограничения на атомарность значений атрибутов. Таким образом, согласно с таблицей 1.4, для хранения данных была выбрана реляционная модель, поскольку она удовлетворяет всем необходимым требованиям.

2 Конструкторский раздел

В данном разделе будет представлена диаграмма проектируемой базы данных, будут описаны ее сущности, а также будут описаны требуемые ограничения целостности. Будут представлены схемы алгоритмов триггеров и функций.

2.1 Таблицы базы данных

На рисунке 2.1 представлена ER-модель разрабатывамой базы данных:

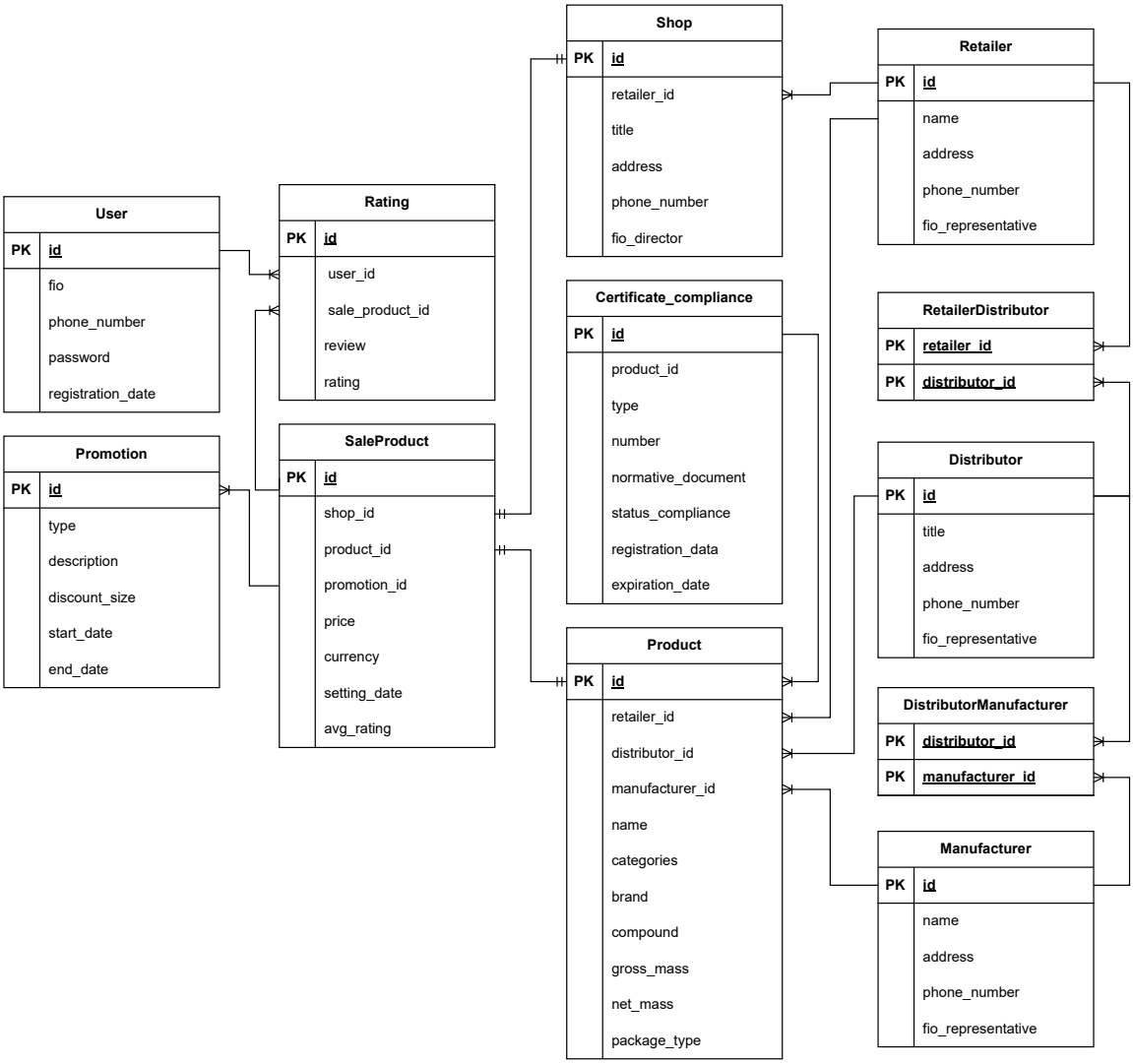

Рисунок 2.1 – ER-модель базы данных

2.2 Описание сущностей

В соответствии с диаграммой проектируемой БД были составлены следующие сущности:

1) **User** – представляет пользователя системы. Содержит следующие поля:
   - id – уникальный идентификатор пользователя;
   - fio – ФИО пользователя;
   - phone_number – номер телефона пользователя;
   - password – поле, хранящее хешированный пароль аккаунта пользователя.
   - registration_date – дата регистрации пользователя.
2) **Shop** – представляет объект магазина. Содержит следующие поля:
   - id – уникальный идентификатор магазина;
   - retailer_id – идентификатор ритейлера, с которым работает сотрудничает магазин;
   - title – название магазина;
   - address – адрес магазина;
   - phone_number – номер телефона магазина;
   - fio_director – ФИО директора магазина;
3) **Product** – представляет объект товара. Содержит следующие поля:
   - id – уникальный идентификатор магазина;
   - retailer_id – идентификатор ритейлера, который реализует товар;
   - distributor_id – идентификатор дистрибьютора, который распространяет товар;
   - manufacturer_id – идентификатор производителя, который производит товар;
   - name – название товара;
   - categories – категория товара;

- brand – бренд товара;
- compound – состав товара;
- gross_mass – масса брутто;
- net_mass – масса нетто;
- package_type – тип упаковки.

4) **Certificate_compliance** – представляет объект сертификата соответствия. Содержит следующие поля:
   - id – уникальный идентификатор сертификата;
   - product_id – идентификатор товара, который соответствует данному сертификату;
   - type – тип сертификата;
   - number – номер сертификата;
   - normative_document – нормативный документ, которому удовлетворяет товар;
   - status_compliance – статус соответствия;
   - registration_data – дата регистрации сертификата;
   - expiration_date – дата окончания действия сертификата.

5) **Promotion** – представляет объект акции на товар. Содержит следующие поля:
   - id – уникальный идентификатор акции;
   - type – тип акции;
   - description – описание акции;
   - discount_size – размер скидки в процентах;
   - start_date – дата начала акции;
   - end_date – дата конца акции.

6) **Rating** – представляет объект оценки товара. Содержит следующие поля:

   - id – уникальный идентификатор оценки;
   - user_id – идентификатор пользователя, оставившего отзыв;
   - sale_product_id – идентификатор продажи конкретного товара в конкретном магазине;
   - review – отзыв;
   - rating – рейтинг.

7) **Manufacturer** – представляет объект производителя товара. Содержит следующие поля:

   - id – уникальный идентификатор производителя;
   - title – название производителя;
   - address – адрес производителя;
   - phone_number – номер телефона;
   - fio_representative – ФИО представителя.

8) **Distributor** – представляет объект дистрибьютора товара. Содержит следующие поля:

   - id – уникальный идентификатор дистрибьютора;
   - title – название дистрибьютора;
   - address – адрес дистрибьютора;
   - phone_number – номер телефона;
   - fio_representative – ФИО представителя.

9) **Retailer** – представляет объект ритейлера товара. Содержит следующие поля:

   - id – уникальный идентификатор ритейлера;
   - title – название ритейлера;
   - address – адрес ритейлера;

- phone_number – номер телефона;
- fio_representative – ФИО представителя.

10) **SaleProduct** – представляет объект продажи товара. Содержит следующие поля:
    - id – уникальный идентификатор продажи;
    - shop_id – идентификатор магазина, продающего товар;
    - product_id – идентификатор продаваемого товара;
    - promotion_id – идентификатор применяемой к товару акции (если она есть);
    - price – цена товара;
    - currency – валюта;
    - setting_date – дата установки цены.
    - avg_rating – средний рейтинг товара в магазине.
11) **ReatailerDistributor** – реализует связь «многие-ко-многим» для отношений Reatailer и Distributor. Содержит следующие поля:
    - retailer_id – идентификатор ритейлера;
    - distributor_id – идентификатор дистрибьютора.
12) **DistributorManufacturer** – реализует связь «многие-ко-многим» для отношений Distributor и Manufacturer. Содержит следующие поля:
    - distributor_id – идентификатор дистрибьютора.
    - manufacturer_id – идентификатор производителя.

2.3 Ограничения целостности

В таблицах 2.1 – 2.10 приведены ограничения целостности для каждой таблицы разрабатываемой БД.

Таблица 2.1 – Ограничения целостности таблицы User

Поле	**Тип**	**Ограничение**
id	UUID	Не должно быть пустым. Является первичным ключом записи
fio	строка	Не должно быть пустым
phone_number	строка	Не должно быть пустым, должно быть уникальным
password	строка	Не должно быть пустым
registration_date	дата	Не должно быть пустым

Таблица 2.2 – Ограничения целостности таблицы Shop

Поле	**Тип**	**Ограничение**
id	UUID	Не должно быть пустым. Является первичным ключом записи
retailer_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Retailer
title	строка	Не должно быть пустым
address	строка	Не должно быть пустым, должно быть уникальным
phone_number	строка	Не должно быть пустым
fio_director	строка	Не должно быть пустым

Таблица 2.3 – Ограничения целостности таблицы Rating

Поле	**Тип**	**Ограничение**
id	UUID	Не должно быть пустым. Является первичным ключом записи
user_id	UUID	Не должно быть пустым. Внешний ключ на запись в User
sale_product_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблицу SaleProduct
review	строка	Не должно быть пустым
rating	число	Не должно быть пустым. Не может быть меньше 0 и больше 5

Таблица 2.4 – Ограничения целостности таблицы Promotion

Поле	**Тип**	**Ограничение**
id	UUID	Не должно быть пустым. Является первичным ключом записи
type	строка	Не должно быть пустым
description	строка	Не должно быть пустым
discount_size	чисто	Не должно быть больше не должно быть меньше 0 и больше 100
start_date	дата	Не должно быть пустым
end_date	дата	Не должно быть пустым

В таблице Promotion для полей start_date и end_date также определено следующее ограничение: $start\_date < end\_date$.

Таблица 2.5 – Ограничения целостности таблицы Certificate_compliance

Поле	**Тип**	**Ограничение**
id	UUID	Не должно быть пустым. Является первичным ключом записи
product_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Product
type	строка	Не должно быть пустым
number	строка	Не должно быть пустым
normative_document	строка	Не должно быть пустым
status_compliance	boolean	Не должно быть пустым
registration_data	дата	Не должно быть пустым
expiration_date	дата	Не должно быть пустым

В таблице Certificate_compliance для полей registration_data и expiration_date также определено следующее ограничение: $registration\_data < expiration\_date$.

Для таблиц Retailer, Distributor, Manufacturer определены следующие ограничения полей:

Таблица 2.6 – Ограничения целостности таблиц Retailer, Distributor и Manufacturer

Поле	**Тип**	**Ограничение**
id	UUID	Не должно быть пустым. Является первичным ключом записи
title	строка	Не должно быть пустым
address	строка	Не должно быть пустым. Должно быть уникальным.
phone_number	строка	Не должно быть пустым. Должно быть уникальным.
fio_representative	строка	Не должно быть пустым. Должно быть уникальным

Таблица 2.7 – Ограничения целостности таблицы Product

Поле	**Тип**	**Ограничение**
id	UUID	Не должно быть пустым. Является первичным ключом записи
retailer_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Retailer
distributor_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Distributor
manufacturer_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Manufacturer
name	строка	Не должно быть пустым
categories	строка	Не должно быть пустым
brand	строка	Не должно быть пустым
compound	строка	Не должно быть пустым
gross_mass	число	Не должно быть пустым. Не должно быть меньше 0
net_mass	число	Не должно быть пустым. Не должно быть меньше 0
package_type	строка	Не должно быть пустым

В таблице Product для полей gross_mass и net_mass также определено следующее ограничение: $gross\_mass \geq net\_mass$.

Таблица 2.8 – Ограничения целостности таблицы SaleProduct

Поле	**Тип**	**Ограничение**
id	UUID	Не должно быть пустым. Является первичным ключом записи
shop_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Shop
product_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Product
promotion_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Promotion
price	число	Не должно быть пустым. Не должно быть меньше 0
currency	строка	Не должно быть пустым
setting_date	дата	Не должно быть пустым
avg_rating	число	Не должно быть пустым. Не может быть меньше 0 и больше 5

Таблица 2.9 – Ограничения целостности таблицы RetailerDistributor

Поле	**Тип**	**Ограничение**
retailer_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Retailer
distributor_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Distributor

В таблице RetailerDistributor также определено следующее ограничение: в совокупности поля retailer_id и distributor_id определяют первичный ключ записи.

Таблица 2.10 – Ограничения целостности таблицы DistributorManufacturer

Поле	**Тип**	**Ограничение**
distributor_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Distributor
manufacturer_id	UUID	Не должно быть пустым. Внешний ключ на запись в таблице Manufacturer

В таблице DistributorManufacturer также определено следующее ограничение: в совокупности поля distributor_id и manufacturer_id определяют первичный ключ записи.

2.4 Триггеры базы данных

В разрабатываемой системе у каждого продающегося товара есть средний рейтинг, который определяется как среднее арифметическое рейтингов в отзывах пользователей. Для корректного отображения среднего рейтинга необходимо вручную делать запросы к базе данных Rating, считать средний рейтинг как сумму рейтингов, деленную на их количество, а затем обновлять поле rating в записи в таблице SaleProduct. Чтобы автоматизировать процесс пересчета рейтинга, можно написать after триггер, который будет выполнять описанные выше действия после добавления или удаления отзыва на товар.

Схема работы данного триггера представлена на рисунке 2.2:

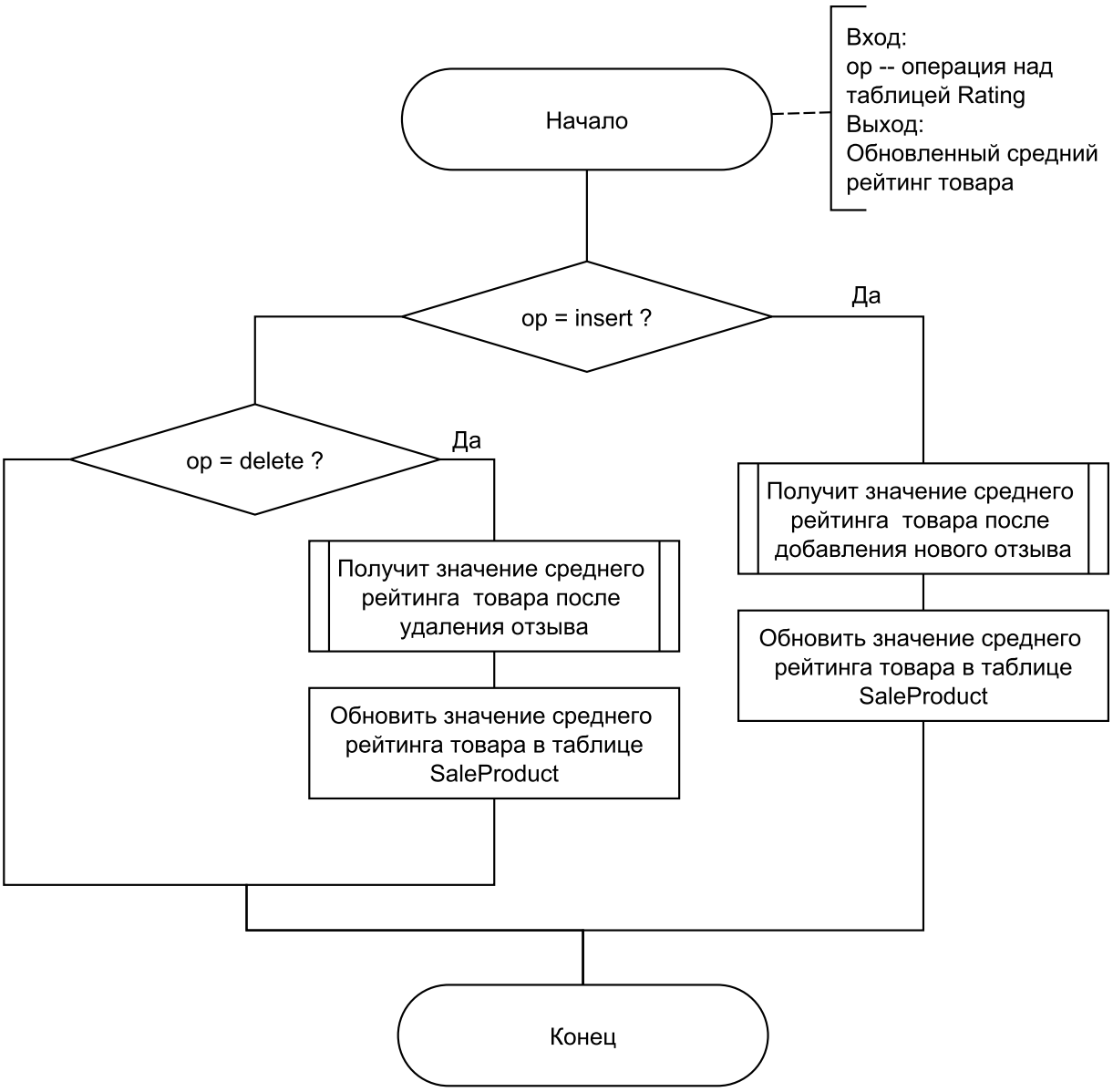

Рисунок 2.2 – Схема работы триггера для обновления среднего рейтинга продаваемого товара

2.5 Функции базы данных

На рисунке 2.3 изображена схема алгоритма функции для получения среднего рейтинга продаваемого товара.

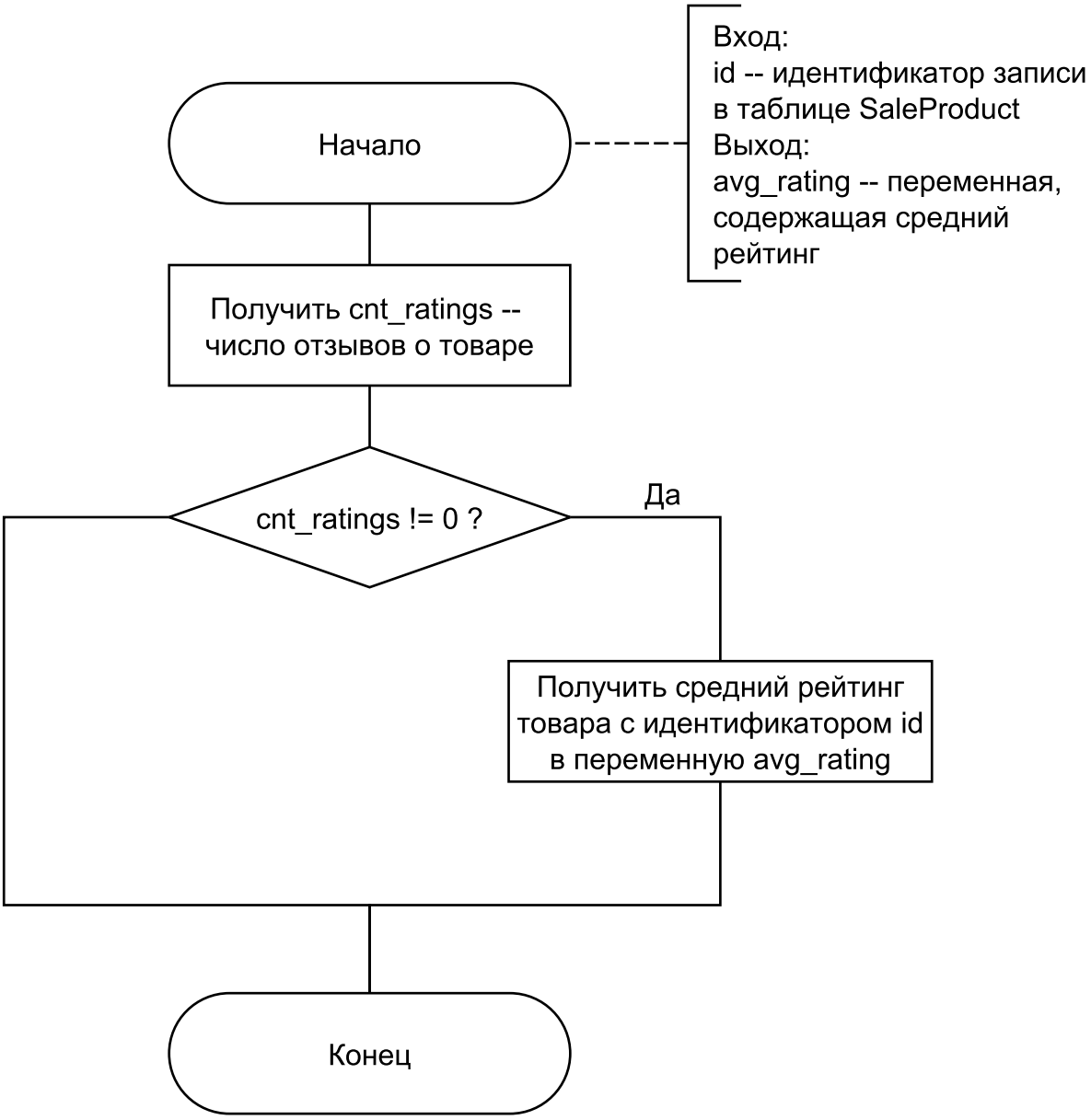

Рисунок 2.3 – Схема работы функции для получения среднего рейтинга продаваемого товара

2.6 Описание проектируемой ролевой модели на уровне базы данных

Разрабатываемая база данных должна содержать следующие роли со следующими правами доступа к таблицам:

1) **гость**. Данной роли предоставляются права на чтение всех таблиц. Также пользователи данной роли имеют права на создание записей в таблице User;

2) **зарегистрированный пользователь**. Данная роль имеет все права роли гость, а также права на создание записей в таблицах Promotion, SaleProduct, Rating, Shop, Product, Retailer, Distributor, Manufacturer, RetailerDistributor, DistributorManufacturer и права на обновление записей в таблице Price;

3) **администратор**. Данная роль имеет все права в отношении всех таблиц базы данных.

Вывод

В данном разделе была представлена диаграмма проектируемой базы данных, были описаны ее сущности, а также были описаны требуемые ограничения целостности. Была представлена схема алгоритма работы триггера на обновление среднего рейтинга продаваемого товара при добавлении/удалении отзыва, а также схема алгоритма функции для получения среднего рейтинга продаваемого товара.

3 Технологический раздел

В данном разделе будет сделан обоснованный выбор средств реализации, будут представлены листинги команд для создания базы данных, таблиц базы данных, ограничений целостности. Также будут описаны способы тестирования триггеров и функций базы данных и будет описан интерфейс доступа к базе данных.

3.1 Средства реализации

В данном подразделе будут выбраны средства реализации базы данных и приложения.

3.1.1 Выбор СУБД для реализации базы данных

В качестве СУБД для реализации проектируемой базы данных рассматривались следующие варианты:

- **MySQL**. Данная СУБД имеет бесплатную версию с ограниченным функционалом и обладает простым синтаксисом, но не полностью соответствует стандартам SQL [21];
- **Oracle** – мультимодельная СУБД с подробной технической документацией. Бесплатные версии имеют очень ограниченный функционал, а платные имеют высокую стоимость [21];
- **MS SQL** – коммерческая СУБД с высокой стоимостью распространения. Имеет полную техническую документацию [21];
- **PostgreSQL** – СУБД с открытым исходным кодом и полной технической документацией [21].

В таблице 3.1 приведено сравнение рассматриваемых СУБД.

Таблица 3.1 – Сравнение СУБД

Критерии	MySQL	Oracle	MS SQL	PostgreSQL
бесплатное распространение СУБД	+	-	-	+
наличие подробной документации	+	+	+	+
наличие опыта работы с СУБД	-	-	-	+

По результатам сравнения выбор был сделан в пользу СУБД PostgreSQL.

3.1.2 Выбор средств реализации приложения

В качестве типа приложения был сделан выбор в пользу Web приложения.

Для реализации backend и frontend частей приложения был выбран язык программирования Go [22] по следующим причинам:

– Go обладает самым быстрым временем компиляции среди других языков, а также одним из лучших показателей производительности [23; 24];

– в стандартной библиотеке языка Go есть библиотека net\http [25], предоставляющая весь необходимый функционал для создания http-сервера и маршрутизации и библиотека database\sql [26], предоставляющая весь необходимый функционал для работы с SQL базами данных;

– наличие опыта работы с данным языком в рамкам дисциплины «Проектирование программного обеспечения».

В качестве среды разработки приложения была выбрана GoLand [27] от компании JetBrains [28] по следующим причинам:

– данная среда разработки создана специально для написания приложений на Go [29];

– в данной среде разработки «из коробки» предоставляются поддержки работы с различными типами СУБД, Git-ом [30] и Docker-ом [31];

3.2 Реализация

В данном подразделе будут описаны сущности реализованной базы данных, ограничения целостности, реализации триггера пересчета рейтинга продаваемого товара и функции получения среднего рейтинга, а также описание реализованной ролевой модели на уровне базы данных.

3.2.1 Создание базы данных и сущностей базы данных

На рисунках 3.1 и 3.2 представлены скрипты для создания базы данных и схемы в базе данных.

Листинг 3.1 – Команда для создания базы данных

```
create database smartshopper;
```

Листинг 3.2 – Команда для создания схемы в базе данных

```
create schema if not exists ss;
```

На рисунках 3.3-3.14 представлены скрипты для создания сущностей базы данных.

Листинг 3.3 – Команда создания таблицы Retailer

```
create table if not exists ss.retailer
(
    id                  uuid primary key,
    title               text,
    address             text,
    phone_number        varchar(30),
    fio_representative text
);
```

Листинг 3.4 – Команда создания таблицы Distributor

```
create table if not exists ss.distributor
(
    id                  uuid primary key,
    title               text,
    address             text,
    phone_number        varchar(30),
    fio_representative text
);
```

Листинг 3.5 – Команда создания таблицы Manufacturer

```
create table if not exists ss.manufacturer
(
    id                  uuid primary key,
    title               text,
    address             text,
    phone_number        varchar(30),
    fio_representative text
);
```

Листинг 3.6 – Команда создания таблицы Shop

```
create table if not exists ss.shop
(
    id           uuid primary key,
    retailer_id  uuid,
    title        text,
    address      text,
    phone_number varchar(30),
    fio_director text
);
```

Листинг 3.7 – Команда создания таблицы Product

```
create table if not exists ss.certificate_compliance
(
    id                  uuid primary key,
    product_id          uuid,
    type                text,
    number              text,
    normative_document text,
    status_compliance   boolean,
    registration_date   date,
    expiration_date     date
);
```

Листинг 3.8 – Команда создания таблицы Certificate_compliance

```
create table if not exists ss.product
(
    id              uuid primary key,
    retailer_id     uuid,
    distributor_id  uuid,
    manufacturer_id uuid,
    name            text,
    categories      text,
    brand           text,
    compound        text,
    gross_mass      numeric,
    net_mass        numeric,
    package_type    text
);
```

Листинг 3.9 – Команда создания таблицы User

```
create table if not exists ss.user
(
    id                uuid primary key,
    fio               text,
    phone_number      varchar(30),
    password          text,
    registration_date date
);
```

Листинг 3.10 – Команда создания таблицы Promotion

```
create table if not exists ss.promotion
(
    id            uuid primary key,
    type          text,
    description   text,
    discount_size numeric,
    start_date    date,
    end_date      date
);
```

Листинг 3.11 – Команда создания таблицы SaleProduct

```
create table if not exists ss.sale_product
(
    id           uuid primary key,
    shop_id      uuid,
    product_id   uuid,
    promotion_id uuid,
    price        numeric,
    currency     text,
    setting_date date,
    avg_rating   float
);
```

Листинг 3.12 – Команда создания таблицы Rating

```
create table if not exists ss.rating
(
    id              uuid primary key,
    user_id         uuid,
    sale_product_id uuid,
    review          text,
    rating          numeric
);
```

Листинг 3.13 – Команда создания таблицы RetailerDistributor

```
create table if not exists ss.retailer_distributor
(
    retailer_id    uuid,
    distributor_id uuid,
    primary key (retailer_id, distributor_id)
);
```

Листинг 3.14 – Команда создания таблицы DistributorManufacturer

```
create table if not exists ss.distributor_manufacturer
(
    distributor_id  uuid,
    manufacturer_id uuid,
    primary key (distributor_id, manufacturer_id)
);
```

3.2.2 Создание ограничений целостности

На листингах 3.15-3.28 представлены команды создания ограничений целостности всех таблиц базы данных.

Листинг 3.15 – Команды создания ограничений целостности для таблицы Retailer

```
alter table if exists ss.retailer
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column title set not null,
    alter column address set not null,
    add unique (address),
    alter column phone_number set not null,
    add unique (phone_number),
    alter column fio_representative set not null,
    add unique (fio_representative);
```

Листинг 3.16 – Команды создания ограничений целостности для таблицы Distributor

```
alter table if exists ss.distributor
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column title set not null,
    alter column address set not null,
    add unique (address),
    alter column phone_number set not null,
    add unique (phone_number),
    alter column fio_representative set not null,
    add unique (fio_representative);
```

Листинг 3.17 – Команды создания ограничений целостности для таблицы Manufacturer (начало)

```
alter table if exists ss.manufacturer
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column title set not null,
    alter column address set not null,
    add unique (address),
    alter column phone_number set not null,
    add unique (phone_number),
    alter column fio_representative set not null,
```

Листинг 3.18 – Команды создания ограничений целостности для таблицы Manufacturer (конец)

```
    add unique (fio_representative);
```

Листинг 3.19 – Команды создания ограничений целостности для таблицы Shop

```
alter table if exists ss.shop
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column retailer_id set not null,
    add foreign key (retailer_id) references retailer (id) on
       delete cascade on update cascade,
    alter column title set not null,
    alter column address set not null,
    add unique (address),
    alter column phone_number set not null,
    add unique (phone_number),
    alter column fio_director set not null;
```

Листинг 3.20 – Команды создания ограничений целостности для таблицы Product (начало)

```
alter table if exists ss.product
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column retailer_id set not null,
    add foreign key (retailer_id) references retailer (id) on
       delete cascade on update cascade,
    alter column distributor_id set not null,
    add foreign key (distributor_id) references distributor (id)
       on delete cascade on update cascade,
    alter column manufacturer_id set not null,
    add foreign key (manufacturer_id) references manufacturer
       (id) on delete cascade on update cascade,
    alter column name set not null,
    alter column categories set not null,
    alter column brand set not null,
    alter column compound set not null,
    alter column gross_mass set not null,
    alter column net_mass set not null,
    alter column package_type set not null,
    add check (gross_mass > 0 and net_mass > 0),
```

Листинг 3.21 – Команды создания ограничений целостности для таблицы Product (конец)

```
    add check (net_mass <= gross_mass);
```

Листинг 3.22 – Команды создания ограничений целостности для таблицы Certificate_compliance

```
alter table if exists ss.certificate_compliance
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column product_id set not null,
    add foreign key (product_id) references product (id) on
       delete cascade on update cascade,
    alter column type set not null,
    alter column number set not null,
    alter column normative_document set not null,
    alter column status_compliance set not null,
    alter column registration_date set not null,
    alter column expiration_date set not null,
    add check (registration_date < expiration_date);
```

Листинг 3.23 – Команды создания ограничений целостности для таблицы User

```
alter table ss.user
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column fio set not null,
    alter column phone_number set not null,
    add unique (phone_number),
    alter column password set not null,
    alter column registration_date set not null;
```

Листинг 3.24 – Команды создания ограничений целостности для таблицы Promotion

```
alter table ss.promotion
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column type set not null,
    alter column description set not null,
    alter column start_date set not null,
    alter column end_date set not null,
    add check (discount_size > 0 and discount_size < 100),
    add check (start_date < end_date);
```

Листинг 3.25 – Команды создания ограничений целостности для таблицы SaleProduct

```
alter table ss.sale_product
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column shop_id set not null,
    add foreign key (shop_id) references shop (id) on delete
       cascade on update cascade,
    alter column product_id set not null,
    add foreign key (product_id) references product (id) on
       delete cascade on update cascade,
    add foreign key (promotion_id) references promotion (id) on
       delete cascade on update cascade,
    alter column price set not null,
    alter column currency set not null,
    alter column setting_date set not null,
    add check (avg_rating >= 0 and avg_rating <= 5),
    add check (price > 0),
    add unique (shop_id, product_id);
```

Листинг 3.26 – Команды создания ограничений целостности для таблицы Rating

```
alter table ss.rating
    alter column id set not null,
    alter column id set default uuid_generate_v4(),
    alter column user_id set not null,
    add foreign key (user_id) references "user" (id) on delete
       cascade on update cascade,
    alter column sale_product_id set not null,
    add foreign key (sale_product_id) references sale_product
       (id) on delete cascade on update cascade,
    alter column review set not null,
    alter column rating set not null,
    add check (rating >= 0 and rating <= 5);
```

Листинг 3.27 – Команды создания ограничений целостности для таблицы RetailerDistributor

```
alter table ss.retailer_distributor
    alter column retailer_id set not null,
    add foreign key (retailer_id) references retailer (id) on
       delete cascade on update cascade,
    alter column distributor_id set not null,
    add foreign key (distributor_id) references distributor (id)
       on delete cascade on update cascade;
```

Листинг 3.28 – Команды создания ограничений целостности для таблицы DistributorManufacturer

```
alter table ss.distributor_manufacturer
    alter column distributor_id set not null,
    add foreign key (distributor_id) references distributor (id)
       on delete cascade on update cascade,
    alter column manufacturer_id set not null,
    add foreign key (manufacturer_id) references manufacturer
       (id) on delete cascade on update cascade;
```

3.2.3 Создание триггеров и функций базы данных

На рисунке 3.1 изображена реализация триггера для обновления рейтинга продаваемого товара в формате схемы алоритма.

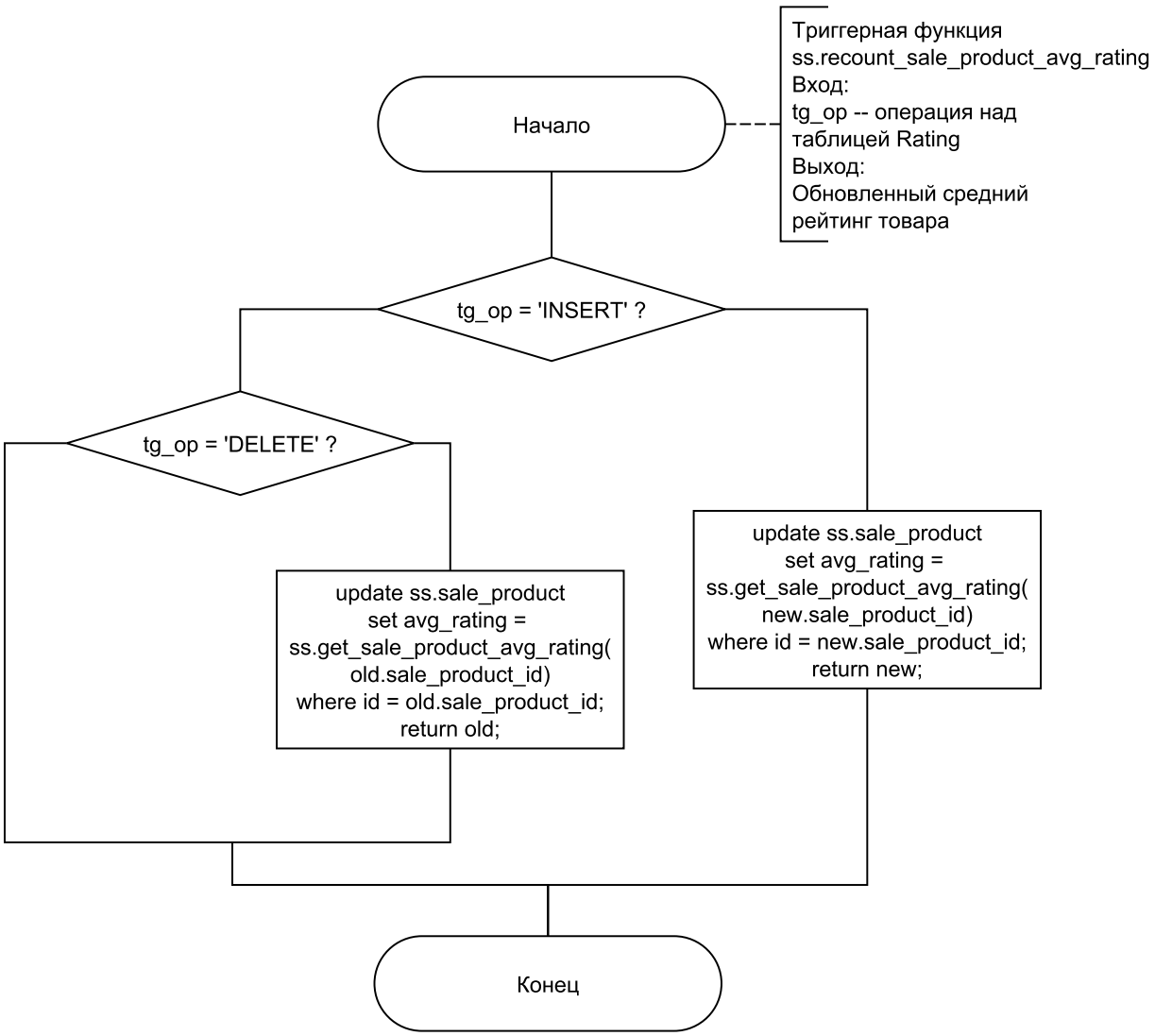

Рисунок 3.1 – Реализация триггера для обновления среднего рейтинга

На рисунке 3.2 изображена реализация функции для получения среднего рейтинга продаваемого товара в формате схемы алоритма.

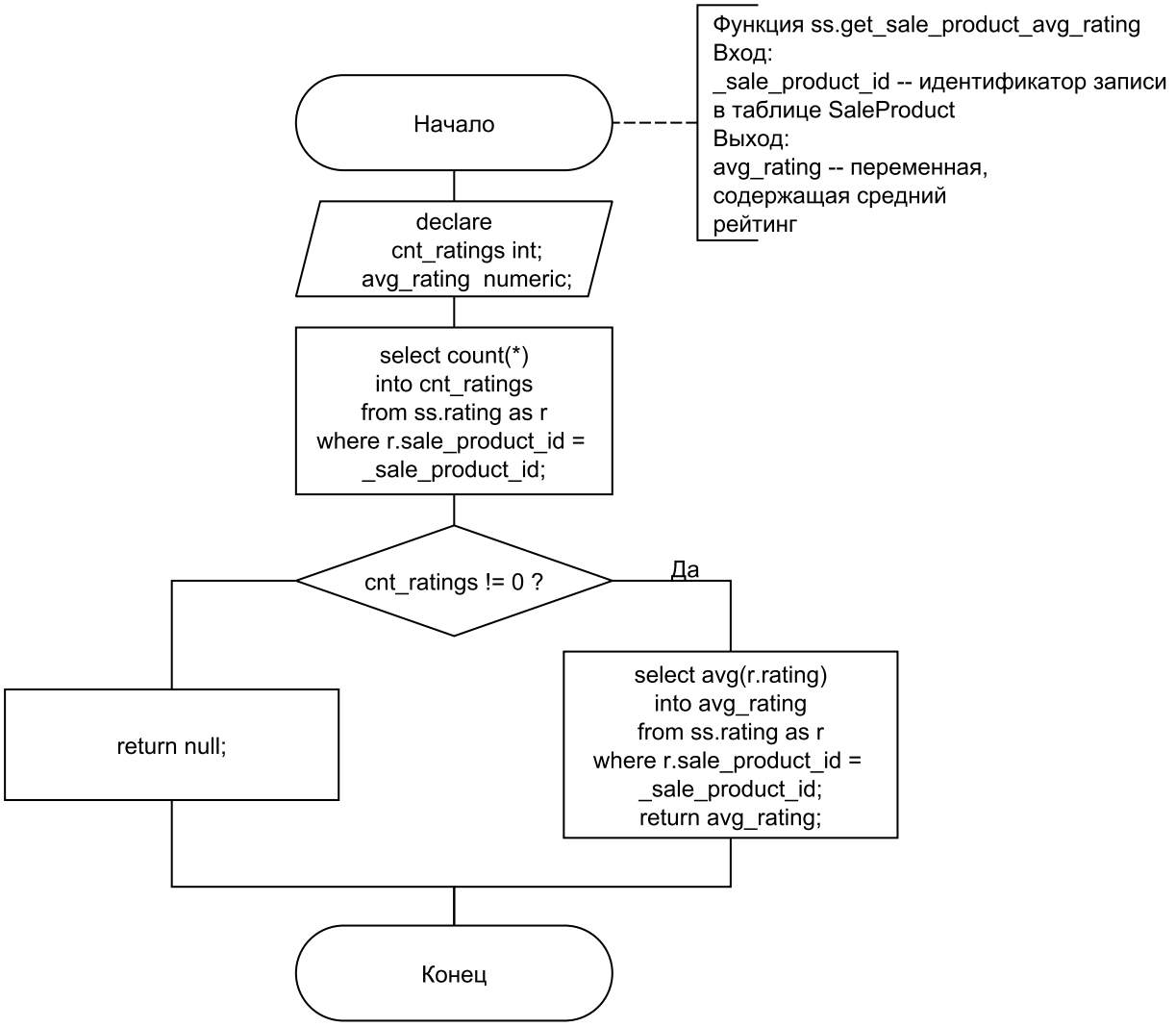

Рисунок 3.2 – Реализация функции для получения среднего рейтинга

На листинге 3.29 показаны команды для создания триггера на обновление рейтинга продаваемого товара.

Листинг 3.29 – Команды создания триггера на обновления среднего рейтинга продаваемого товара

```
create or replace trigger recount_sale_product_avg_rating
    after insert or delete
    on ss.rating
    for each row
execute function ss.recount_sale_product_avg_rating();
```

3.2.4 Создание ролевой модели

На листингах 3.30-3.32 представлено создание ролевой модели на уровне базы данных.

Листинг 3.30 – Команды создания роли гостя

```
create role guest;
grant usage on schema ss to guest;
grant select on all tables in schema ss to guest;
grant insert on table ss.user to guest;
```

Листинг 3.31 – Команды создания роли зарегистрированного пользователя

```
create role registered;
grant guest to registered;
grant insert on table ss.promotion to registered;
grant insert on table ss.sale_product to registered;
grant insert on table ss.rating to registered;
grant insert on table ss.shop to registered;
grant insert on table ss.product to registered;
grant insert on table ss.retailer to registered;
grant insert on table ss.distributor to registered;
grant insert on table ss.manufacturer to registered;
grant insert on table ss.retailer_distributor to registered;
grant insert on table ss.distributor_manufacturer to registered;
grant update on table ss.sale_product to registered;
```

Листинг 3.32 – Команды создания роли администратора

```
create role administrator;
grant registered to administrator;
grant all privileges on all tables in schema ss to administrator;
```

На листинге 3.33 представлено создание пользователей базы данных.

Листинг 3.33 – Команды создания пользователей

```
create user guest_user with password 'guest';
create user registered_user with password 'registered';
create user admin_user with password 'admin';
grant guest to guest_user;
grant registered to registered_user;
grant administrator to admin_user;
```

3.3 Тестирование созданных триггеров и функций базы данных

В данном подразделе будут описаны тесты для функции получения среднего рейтинга продаваемого товара и тесты для триггера обновления среднего рейтинга при добавлении или удалении

Для триггеров и функций были написаны unit тесты [32] с использованием расширения pgTAP [33] для PostgreSQL.

3.3.1 Тестирование функции

Для тестирования функции получения срелнего рейтинга продаваемого товара было написано 5 unit тестов – 2 позитивных и 3 негативных:

1) у продаваемого товара нет отзывов. Ожидаемое значение среднего рейтинга – null;

2) у продаваемого товара есть 2 отзыва. Ожидаемое значение среднего рейтинга – 3.5;

3) передача пустой строки в качестве id записи в таблице sale_product. Ожидаемое значение – исключение;

4) не переданы аргументы в тестируемую функцию. Ожидаемое значение – исключение;

5) передан некорректный аргумент в тестируемую функцию. Ожидаемое значение – исключение.

Листинг 3.34 – Тестовые случаи для функции получения среднего рейтинга продаваемого товара (начало)

```
begin;

select plan(5);

select is(
                ss.get_sale_product_avg_rating(
                        '53b7ff0a-b5cd-4107-85af-1617b137aa23'
                ),
                null
```

Листинг 3.35 – Тестовые случаи для функции получения среднего рейтинга продаваемого товара (конец)

```
        );

insert into ss.rating (user_id, sale_product_id, review, rating)
values ('362b79f6-d671-404a-b1a0-5a655aebc1b6',
   '53b7ff0a-b5cd-4107-85af-1617b137aa23', 'great', 5);

insert into ss.rating (user_id, sale_product_id, review, rating)
values ('8d9b001f-5760-4c40-bc60-988e0ca54d18',
   '53b7ff0a-b5cd-4107-85af-1617b137aa23', 'bad', 2);


select is(
                ss.get_sale_product_avg_rating(
                        '53b7ff0a-b5cd-4107-85af-1617b137aa23'
                ),
                3.5
        );

select throws_ok(
                'ss.get_sale_product_avg_rating('')'
        );

select throws_ok(
                'ss.get_sale_product_avg_rating()'
        );

select throws_ok(
                'ss.get_sale_product_avg_rating(''tsgsddgd'')'
        );

select *
from finish();

rollback;
```

Все тесты были пройдены успешно.

3.3.2 Тестирование триггера

Для тестирования триггера обновления среднего рейтинга продаваемого товара было написано 2 unit теста:

1) у продаваемого товара нет отзывов. Добавляется один отзыв с рейтингом, равным 5. Ожидаемое значение среднего рейтинга – 5;

2) у продаваемого товара есть один отзыв. Отзыв удаляется. Ожидаемое значение среднего рейтинга – null;

Листинг 3.36 – Тестовые случаи для триггера обновления рейтинга продаваемого товара

```
begin;

select plan(2);

insert into ss.rating (user_id, sale_product_id, review, rating)
values ('362b79f6-d671-404a-b1a0-5a655aebc1b6',
   '53b7ff0a-b5cd-4107-85af-1617b137aa23', 'great', 5);

select is(
                (select avg_rating from ss.sale_product where id
                   = '53b7ff0a-b5cd-4107-85af-1617b137aa23'),
                5.0);

delete from ss.rating
where user_id = '362b79f6-d671-404a-b1a0-5a655aebc1b6' and
   sale_product_id = '53b7ff0a-b5cd-4107-85af-1617b137aa23';

select is(
                (select avg_rating from ss.sale_product where id
                   = '53b7ff0a-b5cd-4107-85af-1617b137aa23'),
                null);

select *
from finish();

rollback;
```

Все тесты были пройдены успешно

3.4 Интерфейс доступа к базе данных

Для взаимодействия пользователей с базой данных было написано монолитное Web-приложение с использованием архитектурного подхода REST API [34] и консольным интерфейсом.

На рисунках 3.3-3.10 представлены интерфейсы приложения для взаимодействия с базой данных.

```
Главное меню:
        1 -- Зарегистрироваться
        2 -- Войти
        3 -- Войти как администратор
        4 -- Перейти к каталогу товаров
        0 -- Остановить выполнение программы
Введите пункт меню:
```

Рисунок 3.3 – Интерфейс главного меню приложения

Интерфейс главного меню позволяет зарегистрировать нового пользователя в системе, войти в систему как пользователь и как администратор, а также перейти к каталогу с товарами и остановить работу интерфейса.

```
Меню каталога:
        1 -- Вывести товары
        2 -- Перейти к товару
        0 -- Вернуться в главное меню
Введите пункт меню:
```

Рисунок 3.4 – Интерфейс меню каталога

Интерфейс меню каталога позволяет вывести страницу товаров (10 элементов на странице) и перейти к выбранному товару из списка выведенных страниц.

```
Меню обработки товара:
        1 -- Сравнить цену на товар
        4 -- Посмотреть сертификаты соответствия на товар
        0 -- Вернуться в главное меню
Введите пункт меню:
```

Рисунок 3.5 – Интерфейс меню для обработки действий с товаром

Интерфейс меню для обработки действий с товаром позволяет сравнить цены на товар в различных магазинах, посмотреть, каким сертификатам соответствует товар.

```
Меню обработки товара:
        1 -- Сравнить цену на товар
        2 -- Посмотреть сертификаты соответствия на товар
        3 -- Добавить сертификат соответствия на товар
        4 -- Удалить сертификат соответствия на товар
        5 -- Обновить статус соответствия на товар
        0 -- Вернуться в главное меню
Введите пункт меню:
```

Рисунок 3.6 – Интерфейс меню для обработки действий с товаром для администратора

Администратор получает расширенный набор опций для взаимодействия с товаром: к уже существующим добавляются возможности добавления нового сертификата на товар, удаления уже имеющегося сертификата и обновления статуса соответствия сертификата.

```
Главное меню:
        1 -- Перейти в каталог товаров
        2 -- Добавить новый магазин
        3 -- Перейти к обработке магазинов
        0 -- выйти
Введите пункт меню:
```

Рисунок 3.7 – Интерфейс главного меню аутентифицированного пользователя

Интерфейс главного меню аутентифицированного пользователя позволяет перейти в каталог товаров, добавить новый магазин, а также перейти к обработке магазинов.

```
Меню обработки магазинов:
        1 -- Вывести магазины
        2 -- Следующая страница
        3 -- Перейти к магазину
        0 -- Вернуться в главное меню
Введите пункт меню:
```

Рисунок 3.8 – Интерфейс меню для обработки магазинов

Аутентифицированный пользователь в интерфейсе меню для обработки магазинов может вывести страницу магазинов и перейти к конкретному магазину.

```
Меню обработки магазинов:
        1 -- Вывести магазины
        2 -- Следующая страница
        3 -- Удалить магазин
        4 -- Перейти к магазину
        0 -- Вернуться в главное меню
Введите пункт меню:
```

Рисунок 3.9 – Интерфейс меню для обработки магазинов для администратора

Администратор, помимо выше изложенных возможностей, также может удалить конкретный магазин.

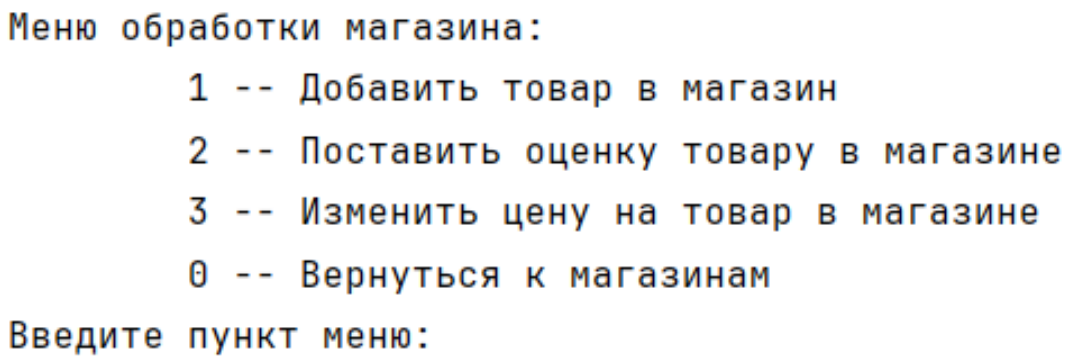

Рисунок 3.10 – Интерфейс меню для обработки магазина

Интерфейс меню для обработки магазина позволяет аутентифицированному пользователю добавить новый товар, оценить товар и изменить его цену.

Вывод

В данном разделе был сделан обоснованный выбор средств реализации, были представлены листинги команд для создания базы данных, таблиц базы данных, ограничений целостности. Также были описаны способы тестирования триггеров и функций базы данных и был описан интерфейс доступа к базе данных.

4 Исследовательский раздел

В данном разделе будут описаны технические характеристики устройства, на котором проводились исследования и будет проведено исследования среднего времени ответа на запросы от числа запросов в секунду с использованием кеша приложения и без и зависимость времени выполнения запроса от наличия индексов.

4.1 Технические характеристики устройства

Технические характеристики устройства, на котором проводилось исследование:

- операционная система: Майкрософт Windows 10 Домашняя для одного языка [35];
- оперативная память: 16 Гб;
- процессор: 11th Gen Intel® Core™ i7-1185G7 @ 3.00 ГГц × 8.

Исследование проводилось на ноутбуке, не подключенном к сети электропитания. В процессе проведения исследования были запущены Docker-контейнеры с базой данных и приложением. Также был запущена среда разработки GoLand, через которую запускались скрипты для проведения исследования.

4.2 Проведение первого исследования

В данном подразделе будет проведено нагрузочное тестирование созданного приложения и будет исследована зависимость среднего времени ответа на запросы от числа запросов в секунду с использованием кеша приложения и без.

Для проведения нагрузочного тестирования был использован инструмент для нагрузочного тестирования Locust [36].

4.2.1 Цель первого исследование

Целью исследования является проведение сравнительного анализа зависимости среднего времени ответа на запросы от числа запросов в секунду с использованием кеша приложения и без.

4.2.2 Наборы варьируемых и фиксированных параметров

В качестве фиксированных параметров были выбраны следующие:

– максимальное число пользователей (равно 500);

– прирост пользователей в секунду (равно 10);

– длительность нагрузочного тестирования (равно 1 минуте);

Для проведения исследования были отобраны запросы на выполнение четырех самых основных действий в рамках созданной системы:

– запрос на получение информации о товаре;

– запрос на получение страницы с товарами из каталога товаров;

– запрос на выполнение сравнения цен на выбранный товар;

– запрос на получение сертификатов соответствия товара.

В качестве кеша приложения использовалось in-memory хранилище данных Redis [37].

4.2.3 Результаты первого исследования

Результаты замеров среднего времени ответа на запросы от числа запросов в секунду с использованием кеширования и без использования кеширования представлены в таблицах 4.1 и 4.2 на рисунке 4.1.

Таблица 4.1 – Результаты замеров среднего времени ответа на запрос от числа запросов в секунду без использования кеширования

Число запросов в секунду	**Среднее время ответа, мс**
0	25.846
78	50.045
275	73.115
327	97.479
347	121.171
362	144.350
399	167.822
406	187.674
420	209.065
425	228.758
443	250.598
450	269.343
469	289.805
473	309.270
470	334.602
453	355.607
455	377.230
442	398.857
448	417.814
452	436.418
453	461.885
437	477.444
442	501.625
431	517.916
422	534.373
435	558.219
439	584.407
442	607.249

Таблица 4.2 – Результаты замеров среднего времени ответа на запрос от числа запросов в секунду с использованием кеширования

Число запросов в секунду	**Среднее время ответа, мс**
0	28.747
108	58.885
947	93.528
1007	130.303
1035	166.300
1057	200.663
1141	235.700
1127	260.495
1138	275.387
1145	286.911
1146	296.106
1154	304.076
1164	311.796
1165	317.920
1165	322.696
1161	327.387
1149	330.579
1152	334.151
1158	337.071
1152	340.013
1164	342.504
1163	344.504
1153	346.230
1158	347.954
1165	349.993
1153	351.469
1153	352.436
1165	354.096

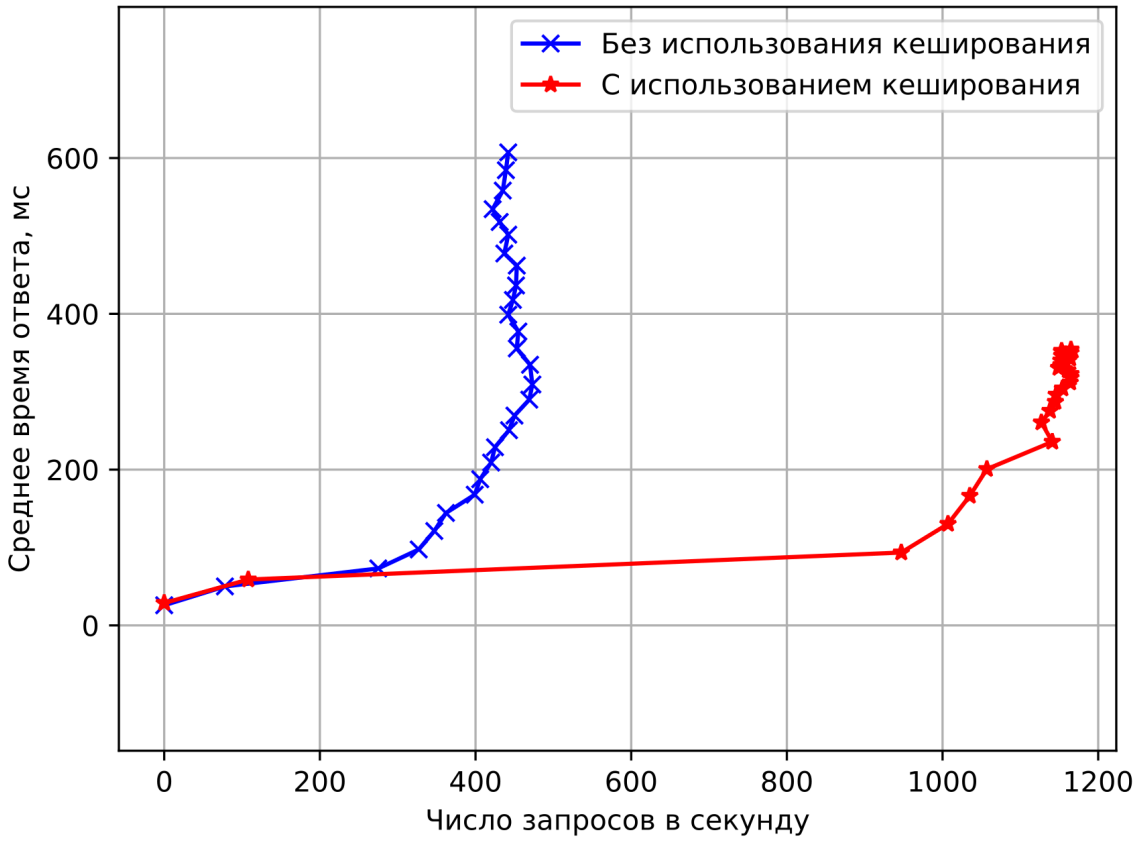

Рисунок 4.1 – Графики зависимости среднего времени ответа от числа запросов в секунду с использованием кеширования и без

Анализируя выборки, представленные в таблицах 4.1 и 4.2, можно увидеть, что при числе запросов, меньших 275, среднее время ответа при использовании кеширования практически не отличается от среднего времени ответа без использования кеширования (из таблицы 4.1 при 78 запросах в секунду, время ответа составляет $\approx$ 50 мс, а из таблицы 4.2 при 108 запросах $\approx$ 58 мс). Это обусловлено тем, что в начале кеш был пуст и первые запросы выполнялись именно к базе данных.

При числе запросов, больших 275, можно увидеть резкий рост среднего времени ответа без использования кеширования (при 327 запроса в секунду среднее время ответа 97.479 мс, а при 442 запросах – 607.249, что примерно в 6.23 раза больше). Число обрабатываемых запросов в секунду также ограничивается сверху числом 473.

Использование кеша позволило обрабатывать значительно больше запросов в секунду (максимальное число обрабатываемых запросов в секунду

равно 1165, что в ≈ 2.46 раза больше, чем без использования кеша) и значительно сократило среднее время ответа (максимальное среднее время ответа без кеша, равное 607.249 мс, достигалось мне обработке 442 запросов в секунду, а максимальное время ответа с кешем, равное 354.096 мс, достигалось при обработке 1165 запросов в секунду, что меньше ≈ 1.71 раза). Само среднее время ответа при использовании кеша также выросло незначительно (при обработке 947 запросов в секунду среднее время ответа составляет 93.528, а при 1165 запросах в секунду среднее время ответа составляет 354.096, что примерно в 3.78 раз больше).

4.3 Проведение второго исследования

В данном подразделе будет проведено исследование зависимости времени выполнения запроса от наличия или отсутствия индексов при различном числе записей.

4.3.1 Цель второго исследование

Целью исследования является проведение сравнительного анализа зависимости времени выполнения запроса от наличия или отсутствия индекса при различном числе записей.

4.3.2 Наборы варьируемых и фиксированных параметров

Замеры времени проводились для числа записей от 10 до 1000. Для каждого числа записей для случая с индексом и без индекса выполнялось 10 запросов, после чего вычислялось среднее арифметическое и отмечалось в качестве результата.

На листинге 4.1 представлена команда создания индекса для таблицы certificate_compliance.

Листинг 4.1 – Команда создания индекса

```
create index idx_product_id on
   ss.certificate_compliance(product_id);
```

Для проведения исследования был выбран запрос на получение всех сертификатов товара по его идентификатору в таблице certificate_compliance.

Листинг 4.2 – Запрос на получение сертификатов соответтствия товара

```
select
    id,
    product_id,
    type,
    number,
    normative_document,
    status_compliance,
    registration_date,
    expiration_date
from ss.certificate_compliance
where product_id = '$1';
```

Вместо $1 подставляется идентификатор товара, сертификаты соответствия которого необходимо получить.

4.3.3 Результаты второго исследования

Результаты замеров времени выполнения запроса от наличия или отсутствия индекса при различном числе записей представлены в таблице 4.3 и на рисунке 4.2.

Таблица 4.3 – Результаты замеров времени выполнения запроса от наличия или отсутствия индекса при различном числе записей

Число записей	**Без индекса, мкс**	**С индексом, мкс**
10	25.9	25.6
100	37.7	26.1
200	52.0	26.6
300	60.1	36.8
400	87.7	37.6
500	101.7	38.3
600	129.4	45.5
700	134.3	46.8
800	159.5	51.3
900	160.7	59.0
1000	165.8	59.9

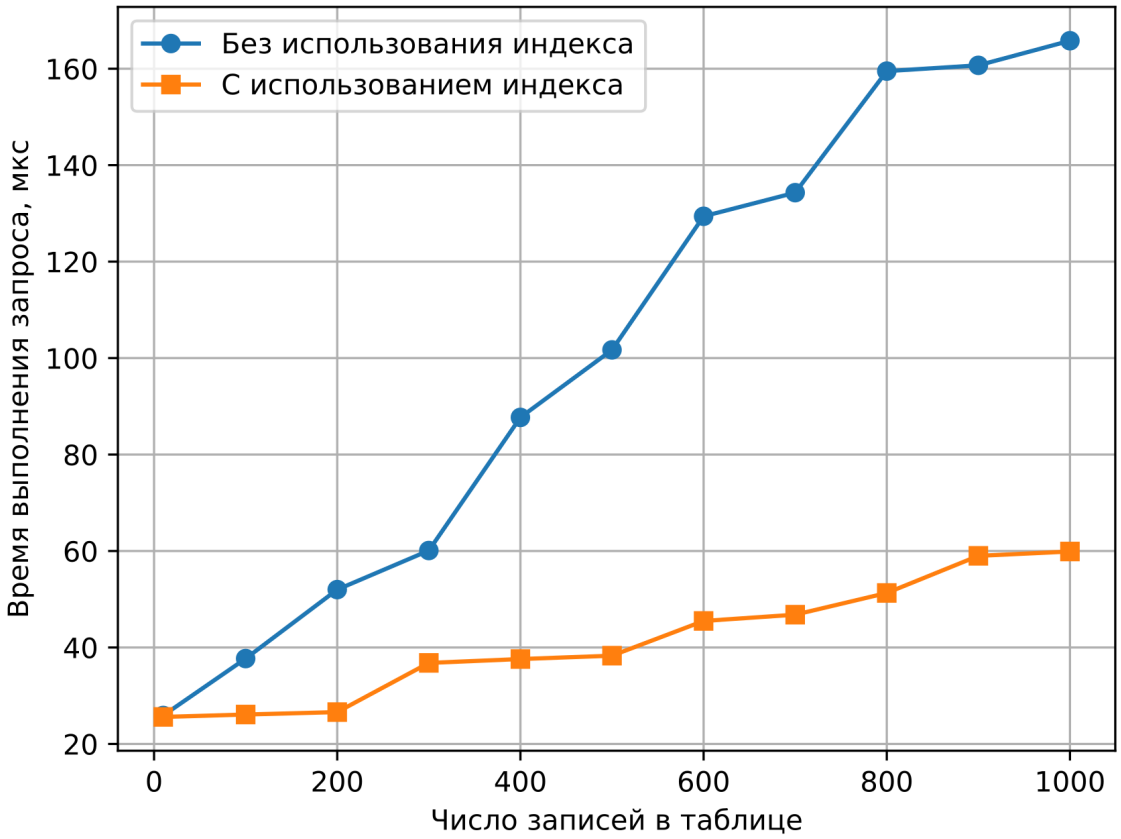

Рисунок 4.2 – Графики зависимости времени выполнения запроса при наличии и отсутствии индекса

Анализируя полученную в результате исследования выборку, представленную в таблице 4.3, можно заметить, что использование индекса с ростом числа записей позволило сократить выполнение запроса практически в 3 раза (при 10 записях в таблице разница во времени выполнения была всего 0.3 мкс, в то время как при 1000 записей в таблице использование индекса сокращает время выполнения ≈ 2.77 раза).

Вывод

В данном разделе были описаны технические характеристики устройства, на котором проводились исследование и было проведено исследование среднего времени ответа на запросы от числа запросов в секунду с использованием кеша приложения и без.

В результате первого исследования было выяснено, что использование кеша приложения позволяет значительно увеличить число обрабатываемых запросов в секунду (с использованием кеша максимальное число обрабатываемых запросов в секунду увеличилось с 473 до 1165, что в ≈ 2.46 раза больше, чем без использования кеша) а также сократить время ответа на запросы практически в 2 раза (максимальное среднее время ответа без кеша равно 607.249, а максимальное время ответа с кешем равно 354.096 мс, что меньше ≈ 1.71 раза).

В результате второго исследования выяснилось, что использование индекса значительно сокращает время выполнения запроса (при 10 записях в таблице разница во времени выполнения была всего 0.3 мкс, в то время как при 1000 записей в таблице запрос с использованием индекса выполняется быстрее $\approx$ в 2.77 раза).

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта были решены следующие задачи:

1) проведен анализ предметной области;
2) проанализированы существующие решения;
3) спроектирована база данных, описаны ее сущности и связи между сущностями;
4) выбраны подходящие средства реализации;
5) реализована база данных;
6) проведено исследования созданной базы данных.

Цель работы, а именно разработка базы данных для сравнительного анализа цен на элементы продуктовой корзины первой необходимости, также была достигнута.

Добавление кеширования на уровне приложения позволило значительно увеличить число обрабатываемых запросов в секунду (с использованием кеша максимальное число обрабатываемых запросов в секунду увеличилось с 473 до 1165, что в ≈ 2.46 раза больше, чем без использования кеша) а также сократить время ответа на запросы практически в 2 раза (максимальное среднее время ответа без кеша равно 607.249, а максимальное время ответа с кешем равно 354.096 мс, что меньше ≈ 1.71 раза).

Использование индекса позволило значительно сократить время выполнения запроса (при 10 записях в таблице разница во времени выполнения была всего 0.3 мкс, в то время как при 1000 записей в таблице запрос с использованием индекса выполняется быстрее $\approx$ в 2.77 раза).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. 25 приложений и веб-сайтов для сравнения цен с лучшими предложениями [Электронный ресурс]. — Режим доступа: https://www.shopify.com/blog/7068398-10-best-comparison-shopping-engines-to-increase-ecommerce-sales (дата обращения: 02.04.2024).
2. Сервисы сравнения цен: как они работают и почему они полезны для потребителей [Электронный ресурс]. — Режим доступа: http://istoki.tv/news/company/servisy-sravneniya-tsen-kak-oni-rabotayut-i-pochemu-oni-polezny-dlya-potrebiteley/ (дата обращения: 02.04.2024).
3. Price.ru: популярность сервиса для сравнения цен и товаров выросла на 50 процентов [Электронный ресурс]. — Режим доступа: https://www.novostiitkanala.ru/news/detail.php?ID=160679 (дата обращения: 02.04.2024).
4. Постановление Правительства РФ от 15 июля 2010 г. N 530 [Электронный ресурс]. — Режим доступа: https://base.garant.ru/12177401/(дата обращения: 01.04.2024).
5. Стимулирование продаж [Электронный ресурс]. — Режим доступа: https://dgpu-journals.ru/wp-content/uploads/2019/12/semahin.pdf (дата обращения: 02.04.2024).
6. Типы акций в рознице и как на них заработать [Электронный ресурс]. — Режим доступа: https://kub-24.ru/2021/07/20/tipy-aktsij-v-roznitse-i-kak-na-nih-zarabotat/ (дата обращения: 09.04.2024).
7. Лучшие идеи для акций в розничном магазине [Электронный ресурс]. — Режим доступа: https://www.ekam.ru/blogs/pos/aktsii-v-roznichnom-magazine (дата обращения: 09.04.2024).
8. Моделирование цепи поставок в условиях колебаний спроса [Электронный ресурс]. — Режим доступа: http://www.salogistics.ru/magazine/19/9_andronov_52-63.pdf (дата обращения: 02.04.2024).
9. ФГИС «Меркурий» [Электронный ресурс]. — Режим доступа: https://www.vetrf.ru/vetrf/materials/ (дата обращения: 23.03.2024).

10. Федеральная государственная информационная система (ФГИС) «Меркурий» как решение проблемы прослеживаемой продукции [Электронный ресурс]. — Режим доступа: https://cyberleninka.ru/article/n/federalnaya-gosudarstvennaya-informatsionnaya-sistema-fgis-merkuriy-kak-reshenie-problemy-proslezhivaemosti-produktsii/viewer (дата обращения: 23.03.2024).

11. Подтверждение соответствия пищевой продукции [Электронный ресурс]. — Режим доступа: https://www.gostest.com/pishchevaya-produktsiya/ (дата обращения: 09.04.2024).

12. Едадил [Электронный ресурс]. — Режим доступа: https://edadeal.ru (дата обращения: 01.04.2024).

13. SkidkaOnline [Электронный ресурс]. — Режим доступа: https://skidkaonline.ru (дата обращения: 01.04.2024).

14. Price.ru [Электронный ресурс]. — Режим доступа: https://price.ru (дата обращения: 01.04.2024).

15. *Дж. Д. К.* Введение в системы баз данных: 8-е издание //. — «Вильямс», 2006. — С. 1328.

16. *Т.С. К.* Базы данных: модели, разработка, реализация: Учебник для вузов. //. — СПб.: Питер, 2002. — С. 304.

17. *Д. К. С.* Основы современных баз данных //. — Центр Информационных Технологий, 1998. — С. 251.

18. *М. Г. Ю.* Курс лекций по дисциплине «Базы данных» ИУ7 //. — МГТУ им. Н. Э. Баумана, 2023.

19. Постреляционные СУБД [Электронный ресурс]. — Режим доступа: https://dit.isuct.ru/IVT/BOOKS/DBMS/DBMS14/ch_6_2.html (дата обращения: 01.04.2024).

20. *Д.А. Попова-Коварцев Е. С.* Основы проектирования баз данных //. — Изд-во Самарского университета, 2019. — С. 112.

21. Анализ популярных реляционных систем управления базами данных (2022 г.) [Электронный ресурс]. — Режим доступа: https://drach.pro/blog/hi-tech/item/196-popular-relational-dbms-2022 (дата обращения: 31.08.2024).

22. Build simple, secure, scalable systems with Go [Электронный ресурс]. — Режим доступа: https://go.dev/ (дата обращения: 31.08.2024).

23. В Google провели сравнение производительности C++, Java, Go и Scala [Электронный ресурс]. — Режим доступа: https://www.opennet.ru/opennews/art.shtml?num=30784 (дата обращения: 31.08.2024).

24. Сравнение производительности Golang | Почему GO Fast? [Электронный ресурс]. — Режим доступа: https://www.golinuxcloud.com/golang-performance/ (дата обращения: 31.08.2024).

25. http [Электронный ресурс]. — Режим доступа: https://pkg.go.dev/net/http (дата обращения: 31.08.2024).

26. sql [Электронный ресурс]. — Режим доступа: https://pkg.go.dev/database/sql (дата обращения: 31.08.2024).

27. oLand [Электронный ресурс]. — Режим доступа: https://www.jetbrains.com/go/ (дата обращения: 31.08.2024).

28. JetBrains [Электронный ресурс]. — Режим доступа: https://ru.wikipedia.org/wiki/JetBrains (дата обращения: 31.08.2024).

29. Getting started | GoLand Documentation [Электронный ресурс]. — Режим доступа: https://www.jetbrains.com/help/go/getting-started.html (дата обращения: 31.08.2024).

30. Git [Электронный ресурс]. — Режим доступа: https://git-scm.com/ (дата обращения: 31.08.2024).

31. Docker Docs [Электронный ресурс]. — Режим доступа: https://docs.docker.com/ (дата обращения: 31.08.2024).

32. Unit тестирование [Электронный ресурс]. — Режим доступа: https://www.elibrary.ru/download/elibrary_46461181_32137124.pdf (дата обращения: 31.08.2024).

33. pgTAP [Электронный ресурс]. — Режим доступа: https://pgtap.org/ (дата обращения: 31.08.2024).

34. *Jim Webber Savas Parastatidis I. R.* REST in Practice Hypermedia and Systems Architecture //. — O'Reilly Media, Incorporated, 2010. — С. 446.

35. Microsoft Windows 10 Домашняя [Электронный ресурс]. — Режим доступа: https://www.microsoft.com/ru-ru/software-download/windows10 (дата обращения: 31.08.2024).

36. What is Locust? [Электронный ресурс]. — Режим доступа: https://docs.locust.io/en/stable/what-is-locust.html (дата обращения: 31.08.2024).

37. Introduction to Redis [Электронный ресурс]. — Режим доступа: https://master--redis-doc.netlify.app/docs/about/ (дата обращения: 31.08.2024).

ПРИЛОЖЕНИЕ А

Презентация к курсовой работе

Презентация содержит 15 слайдов.