Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ (ИУ)

КАФЕДРА ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (ИУ7)

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

Разработка базы данных для хранения и обработки данных винного магазина

Студент группы **ИУ7-62Б** _____ **Обревская В. В.**

Руководитель курсовой работы _____ **Филиппов М. В.**

2023 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ

Заведующий кафедрой ИУ7

_____ Рудаков И. В.

«3» марта 2023 г.

З А Д А Н И Е
на выполнение курсовой работы

по дисциплине

Базы данных

Студент группы **ИУ7-62Б**

Обревская Вероника Владимировна

Тема курсовой работы

Разработка базы данных для хранения и обработки данных винного магазина

График выполнения работы: 25% к 4 нед., 50% к 8 нед., 75% к 13 нед., 100% к 15 нед.

Задание

Сформулировать требования и ограничения к разрабатываемой базе данных и приложению винного магазина. Спроектировать архитектуру базы данных и ограничения целостности. Спроектировать ролевую модель на уровне базы данных. Выбрать средства реализации. Реализовать спроектированную БД и необходимый интерфейс для взаимодействия с ней. Исследовать характеристики разработанного программного обеспечения.

Оформление курсовой работы:

Расчетно-пояснительная записка на 25-40 листах формата А4. Презентация на 12-18 слайдах.

Дата выдачи задания «3» марта 2023 г.

Руководитель курсовой работы _____ **Филиппов М. В.**

Студент _____ **Обревская В. В.**

РЕФЕРАТ

Расчетно-пояснительная записка 38 с., 7 рис., 2 табл., 24 источн., 1 прил.

БАЗЫ ДАННЫХ, ВИННЫЙ МАГАЗИН, PostgreSQL, SQL, АРХИТЕКТУРА ПРИЛОЖЕНИЯ, API.

Цель работы — разработка базы данных для винного магазина.

В процессе работы проанализированы существующие системы управления базами данных и известные решения в области винных магазинов. Были спроектированы база данных, реализованы ролевые модели и триггер базы данных. Разработано программное обеспечение. Проведено исследование зависимости времени обработки информации от распределения вычислений между базой данных и приложением.

В результате исследования сделан вывод, что вычисления для проверки данных на уровне базы данных быстрее, чем на уровне приложения.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В России алкогольная продукция пользуется спросом в настоящее время. Так, по данным ВЦИОМ[1], в 2021 году 62% россиян пьют алкогольные напитки один или более раз в месяц, из них 46% периодически пили вино. Вино, по мнению опрошенных, считается самым безвредным (39%). Кроме того, винные магазины обычно предлагают богатый выбор вин, в них можно попробовать вина из разных стран и регионов, а также купить редкие и эксклюзивные вина. А возможность просматривать каталог товаров и делать заказ онлайн облегчает покупки для пользователей.

Цель данной работы — разработка базы данных для хранения и обработки данных винного магазина.

Чтобы достигнуть поставленной цели, требуется решить следующие задачи:

1) формализовать задачу, определить необходимый функционал;
2) провести анализ существующих СУБД;
3) спроектировать базу данных, необходимую для хранения и структурирования данных;
4) программно реализовать спроектированную базу данных с использованием выбранной СУБД;
5) разработать программное обеспечение;
6) реализовать пользовательский интерфейс;
7) провести сравнительный анализ обработки данных в зависимости от расположения вычислений на приложении или на базе данных.

1 Аналитический раздел

1.1 Анализ предметной области

Продажи алкоголя в России выросли на 2% в 2022 году. Это следует из итогового доклада министерства здравоохранения России[2] за год. Российские виноделы увеличили производство за январь-апрель 2023 года по сравнению с аналогичным периодом прошлого года на 10%, по данным Минсельхоз РФ[3].

В Российской Федерации запрещена доставка алкогольной продукции, к которой относится и вино, пунктом 5 правил продажи товаров дистанционным способом[4]. Поэтому в России не нужно реализовывать функционал по доставке вин. После оформления заказов с алкоголем покупатели должны лично приехать в магазин или на склад и забрать товар.

Интернет магазин должен предоставлять доступ к перечню вин, а также поддерживать возможность добавления и удаления вин в заказе. Покупатели должны иметь возможность для легкого оформления заказа. Бонусные баллы сподвигнут пользователей на последующие покупки и помогут привлечь постоянных клиентов. Также раздел «любимое» поможет покупателям отложить вино, чтобы вернуться к нему попозже и купить.

1.2 Постановка задачи

Необходимо разработать приложение винного магазина. Авторизованным пользователям предоставлен доступ к ассортименту вин, а также возможность формировать заказы. Пользователь может добавлять и удалять вина из заказа. Заказы могут быть оплачены рублями или полностью баллами. Если заказ оплачен рублями, то клиенту начисляется баллами 10% от суммы заказа после подтверждения оплаты. Также пользователь может добавлять вина в раздел «любимое». Администратор может добавлять новую позицию вина в каталог, удалять и обновлять уже существующие позиции в каталоге, также администратор подтверждает оплату счета.

1.3 Формализация данных

База данных состоит из таблиц:

1) Таблица вин wine.

Хранит в себе сведения о винах: название, количество оставшихся бутылок, год, крепость, цену, страну и сорт.

2) Таблица пользователей user.

   Содержит данные пользователей: логин, пароль, ФИО, электронную почту, баллы, права доступа.

3) Таблица заказов order.

   Включает в себя владельца заказа, общую стоимость заказа, статус заказа, тип оплаты.

4) Таблица элементов заказа order_element.

   Содержит сведения об элементе заказа: вине; количестве заказанных бутылок; заказе, к которому принадлежит.

5) Таблица счетов bill.

   Хранит в себе сведения о счете: заказ, статус и сумма к оплате.

6) Таблица для «любимых» вин user_wine.

   Содержит идентификационный номер пользователя и идентификационный номер вина как внешние ключи.

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

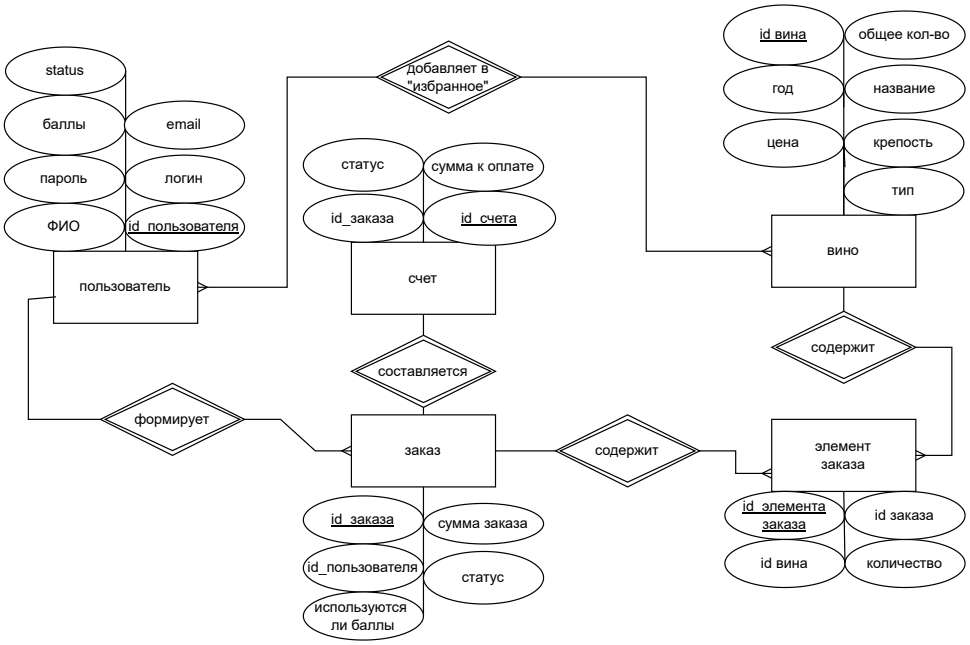

Рисунок 1.1 – ER-диаграмма сущностей проектируемой базы данных в нотации Чена

1.4 Типы пользователей

Выделено три вида пользователей: незарегистрированный пользователь, зарегистрированный пользователь и администратор. Для каждого типа пользователей предусмотрен свой набор возможностей.

На рисунке 1.2 представлена Use-Case диаграмма использования приложения.

Рисунок 1.2 – Use-Case диаграмма использования приложения

1.5 Общие сведения о БД и СУБД

База данных (БД) — это совокупность данных, хранимых в упорядоченной форме, с целью обеспечения доступа к этим данным и их использования каким-либо организационными или прикладными процессам [5].

Система управления базами данных (СУБД) – это совокупность программ и языковых средств, предназначенных для управления данными в базе данных, ведения базы данных и обеспечения взаимодействия её с прикладными программами [6].

1.6 Описание существующих моделей данных

Базы данных разделяют на два основных типа: реляционные и нереляционные. Последние делятся ещё на два: сетевые и иерархические. Хоть и существует очень большое число видов баз данных, но их всех можно классифицировать по модели данных, которая определяет логическую структуру БД:

1. сетевая модель — используют структуру в виде графа.;
2. иерархическая модель — использует структуру данных в виде деревьев;
3. реляционная модель — использует табличное представление данных.

1.6.1 Сетевая модель

Сетевая БД состоит из набора экземпляров определенного типа записи и набора экземпляров определенного типа связей между этими записями [7]. Данную модель можно представить в виде сети, состоящей из узлов, которые могут иметь несколько связей. Она облегчает доступ к данным, так как вместо указания адреса нужной ячейки, данные можно получить, сделав запрос на поиск дочернего объекта.

Сетевая модель имеет свои преимущества и недостатки. Основным преимуществом является гибкость и эффективность поиска информации. Однако главным недостаток остается невозможность изменить структуру после ввода данных. Также, в связи с трудностью моделирования сложных структур данных, могут возникать сложности в управлении и обслуживании базы данных.

1.6.2 Иерархическая модель

В иерархической модели данных используется представление базы данных в виде древовидной структуры, где между объектами существуют связи предок-потомок, при этом объект-предок может иметь неограниченное количество потомков, тогда как у объекта-потомка обязательно будет только один предок [8].

Основным преимуществом иерархической модели является высокая скорость доступа к данным. Это происходит за счет прямого доступа к узлам

в древовидной структуре иерархии данных. Кроме того, данная модель данных является простой в использовании и понимании, что позволяет легко создавать и изменять базы данных.

Главным недостатком иерархической модели является ограничение в возможностях представления сложных связей между данными. Это ограничивает варианты использования.

1.6.3 Реляционная модель

Реляционная модель данных является совокупностью данных и основана на использовании таблиц, которые связаны между собой посредством отношений [9]. Каждая таблица состоит из строк и столбцов, каждый столбец имеет имя и определенный тип данных. На пересечении строк и столбцов находятся конкретные значения.

Основным преимуществом реляционной модели является легкость ее использования и гибкость в изменении данных. Она позволяет быстро создавать и изменять таблицы и связи.

Недостатком реляционной модели является ограничение в представлении сложных структур данных. Кроме того, она может быть менее эффективной в случаях, когда требуется быстрый доступ к большим объемам данных.

В настоящее время реляционная модель является наиболее широко используемой формой представления данных в различных областях.

Для разработки была выбрана реляционная модель базы данных, будет использоваться PostgreSQL, как один из наиболее популярных реляционных СУБД и так как имеется опыт разработки на нем.

1.7 Анализ существующих решений

Из числа ранее существующих «конкурентов», было выделено 3 решения. Сравнение существующих решений проводилось по следующим критериям:

1) возможность покупки за баллы;

2) ориентированность на продажу вина;

3) поддержка русского языка.

В таблице 1.1 приведено сравнение других решений по данным критериям.

Таблица 1.1 – Сравнение существующих решений

Название	возможность покупки за баллы	ориентированность на продажу вина	поддержка русского языка
Красное&Белое[10]	нет	нет	есть
ВИНЛАБ[11]	есть	нет	есть
wine.com[12]	нет	есть	нет

Представленные решения не удовлетворяют всем требованиям: некоторые цифровые магазины не позволяют совершать покупки за баллы, многие продают не только вина, но и другие товары, а некоторые попросту не поддерживают русский язык, что приносит неудобства покупателям из России.

2 Конструкторский раздел

2.1 Проектирование БД

2.1.1 Описание сущностей проектируемой базы данных

База данных состоит из 6 табилиц:

1) Таблица вин wine.

   Хранит в себе сведения о винах:

   - id_wine – номер вина, первичный ключ;
   - name – название вина, символьный тип;
   - count – количество оставшегося вина, целочисленный тип;
   - year – год создания вина, целочисленный тип;
   - strength – крепость вина в %, целочисленный тип;
   - price – цена вина, целочисленный тип;
   - country – страна вина, символьный тип;
   - type – сорт вина, символьный тип.

2) Таблица пользователей user.

   Хранит в себе сведения о пользователях:

   - id_user – номер пользователя, первичный ключ;
   - login – логин пользователя, символьный тип;
   - password – пароль пользователя, символьный тип;
   - FIO – имя пользователя, символьный тип;
   - email – электронная почта пользователя, символьный тип;
   - points – баллы пользователя, целочисленный тип;
   - status – права доступа пользователя, целочисленный тип.

3) Таблица заказов order.

   Хранит в себе сведения о заказах:

- id_order – номер заказа, первичный ключ;
- id_user – номер пользователя, внешний ключ;
- total_price – общая сумма заказа, целочисленный тип;
- status – статус заказа, символьный тип;
- is_points – будет ли оплата баллами, булевый тип.

4) Таблица элементов заказа order_element.

   Хранит в себе сведения об элементах заказах:

   - id_order_element – номер элемента заказа, первичный ключ;
   - id_order – номер заказа, внешний ключ;
   - id_wine – номер вина, внешний ключ;
   - count – количество вина, целочисленный тип.

5) Таблица счетов bill.

   Хранит в себе сведения о счетах:

   - id_bill – номер счета, первичный ключ;
   - id_order – номер заказа, внешний ключ;
   - status – статус счета, символьный ключ;
   - price – сумма к оплате, целочисленный тип.

6) Таблица с «любимыми» винами user_wine.

   Хранит в себе сведения о «любимых» винах:

   - id_user – номер пользователя, внешний ключ;
   - id_wine – номер вина, внешний ключ;

   Вина и пользователь связаны отношением «многие-ко-многим», потому что у клиента может быть несколько «любимых» вин, и несколько клиентов могут любить одно и то же вино. Данное отношение реализуется через третью таблицу user_wine, которая хранит в себе идентификаторы пользователя и вина, а также через связь «один-ко-многим» таблицы user_wine с таблицами user и wine.

На рисунке 2.1 представлена ER диаграмма базы данных.

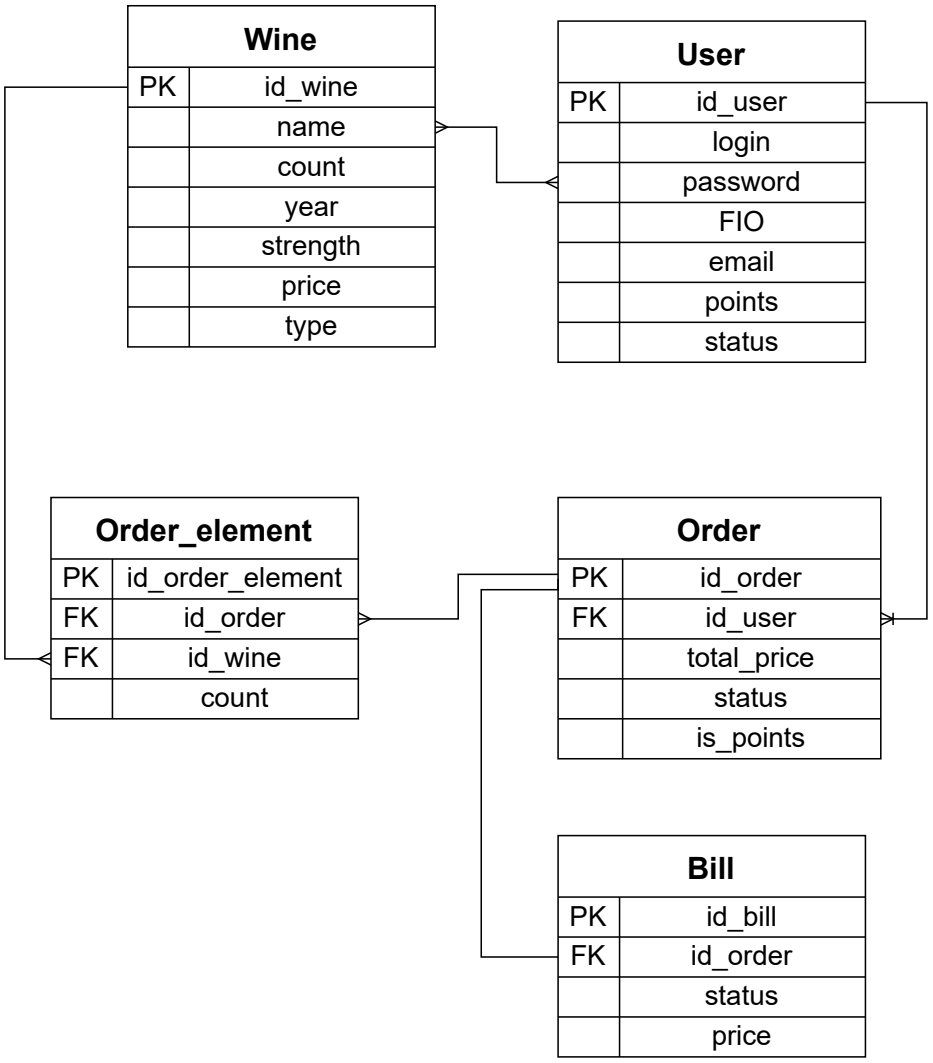

Рисунок 2.1 – ER диаграмма базы данных

2.1.2 Описание проектируемых ограничений целостности базы данных

1) Таблица вин wine.

   Данные поля не могут быть пустыми:

   – id_wine – номер вина, первичный ключ;

   – name – название вина, символьный тип;

   – year – год создания вина, целочисленный тип;

   – price – цена вина, целочисленный тип;

   – country – страна вина, символьный тип;

– type – сорт вина, символьный тип.

Поле count по умолчанию равно 10.

2) Таблица пользователей user.

Поля login, password и email должны быть не пустыми. Поле email должно быть заполнено по шаблону электронной почты вида "*@*.*". Где значок * означает любое количество символов. Количество баллов в таблице по умолчанию 0, а статус у пользователя по умолчанию стоит customer.

3) Таблица заказов order.

Статус заказа не может быть пустым.

4) Таблица счетов bill.

Статус счета не может быть пустым.

2.2 Требование к программе

Для успешной реализации поставленной задачи программа должна предоставлять следующие возможности для незарегистрированного пользователя:

– авторизация;

– регистрация;

– вывод списка всех вин.

У незарегистрированного пользователя есть доступ к просмотру таблицы Wine. А также на добавление в таблицу User.

Зарегистрированный пользователь должен иметь следующие возможности:

– авторизация;

– вывод списка всех вин;

– добавление вина в заказ;

– удаление вина из заказа;

– просмотр заказа;

– оформление заказа.

У зарегистрированного пользователя есть доступ к просмотру таблицы Wine, на добавление в таблицу User, есть доступ на создание, удаление и просмотр любимых вин, есть доступ на просмотр и добавление/удаление полей к таблицам Order и Order_element, а также на вставку и просмотр счета.

Возможности администратора:

– авторизация;

– добавление вина в каталог;

– удаление вина из каталога;

– обновление вина в каталоге;

– подтверждение оплаты счета;

– обновление бонусов у пользователя.

У администратора есть все права доступа над всеми таблицами базы данных.

2.3 Проектирование приложения

Программу можно разделить на 2 основные части:

– front-end – приложение с которым взаимодействует пользователь, отображение данных;

– back-end – взаимодействие с базой данных: получение, удаление и изменение данных.

На рисунке 2.2 представлена UML диаграмма классов моделей и представлений.

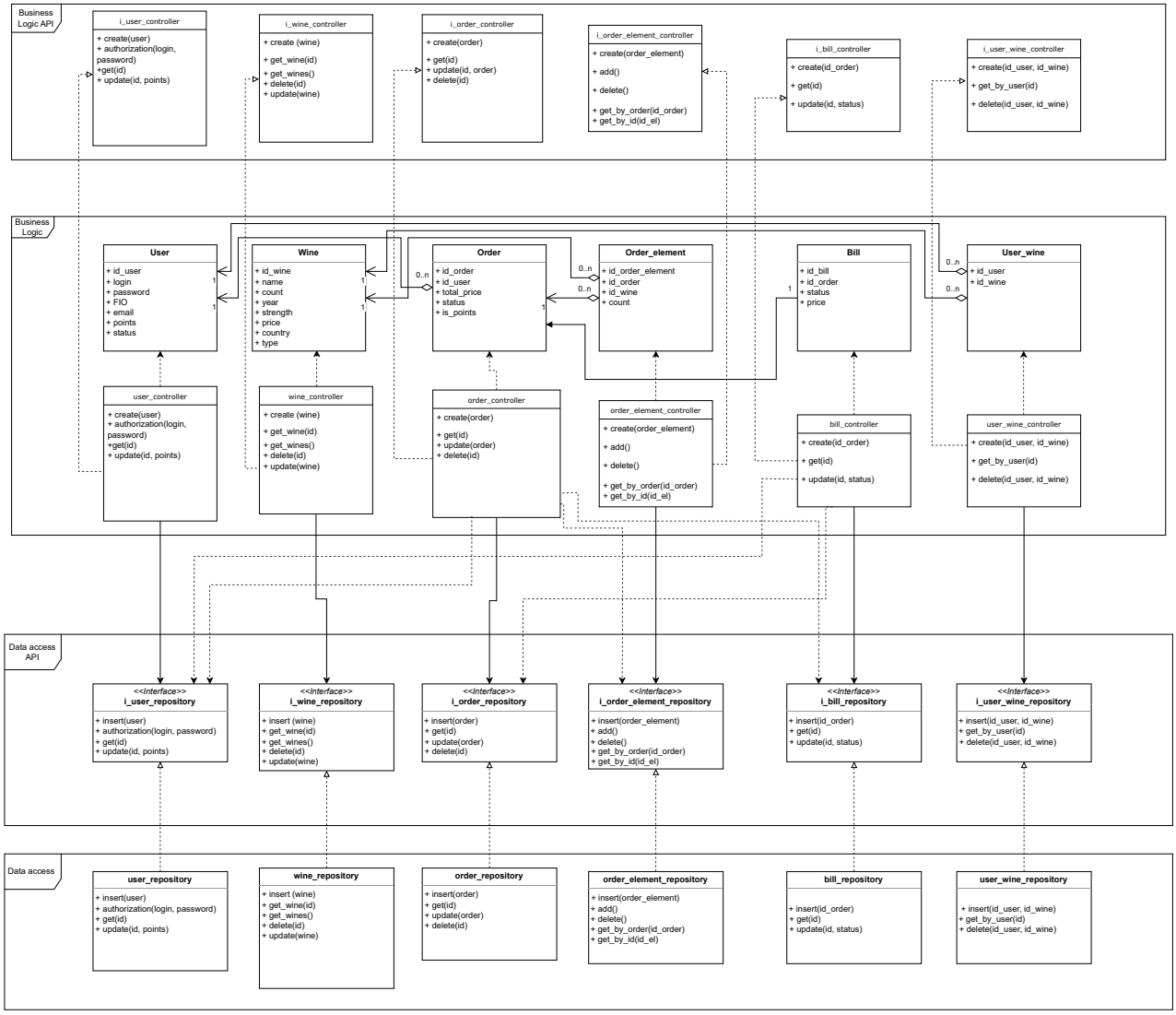

Рисунок 2.2 – UML диаграмма

2.4 Разработка триггера базы данных

Был спроектирован триггер для создания нового элемента в заказе. При создании элемента заказа, надо следить, чтобы данное вино не находилось уже в данном заказе, чтобы не было ситуаций, когда в заказе несколько элементов с одинаковыми винами, но разным количеством товара. Если покупатель хочет изменить количество товара в элементе заказа, то ему предоставляются функции по увеличению и уменьшению количества элементов заказа. Также необходимо следить, чтобы покупатель не положил в заказ вина больше, чем есть на складе. Чтобы обеспечить данные проверки был спроектирован триггер, который срабатывает при создании нового элемента заказа. Схема данного триггера представлена на рисунке 2.3.

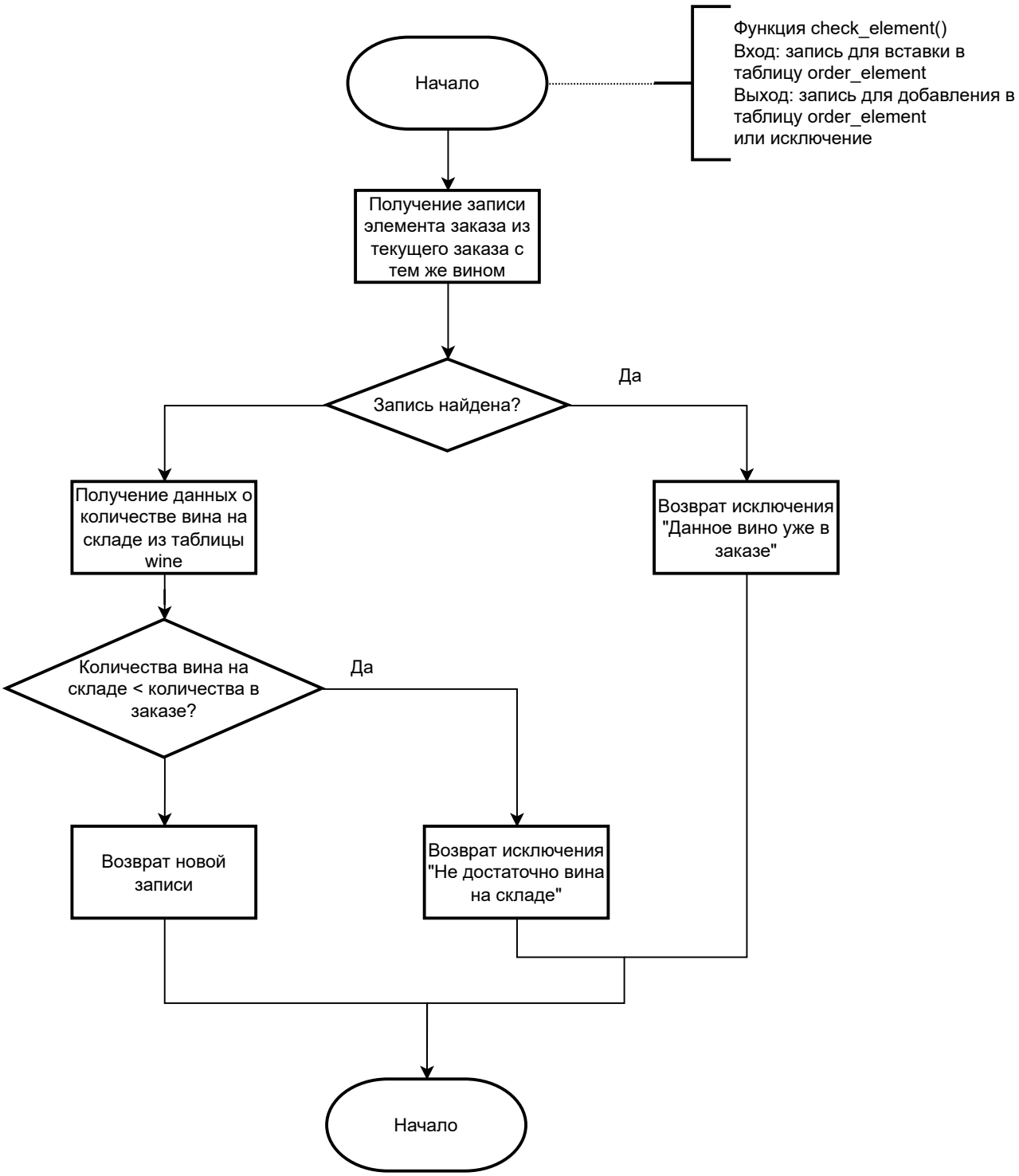

Рисунок 2.3 – Схема алгоритма функции спроектированного триггера, срабатывающего при добавлении элемента заказа

3 Технологический раздел

3.1 Обоснование выбора системы управления базами данных

Систем управления базами данных очень много. Чтобы выбрать наиболее подходящую для данного приложения был проведен сравнительный анализ одних из наиболее популярных СУБД: MySQL, Oracle, PostgreSQL.

При сравнении учитывалось:

— есть ли бесплатное и открытое программное обеспечение;

— поддерживаются ли процедурные расширения языка SQL;

— присутствует ли опыт работы с данной СУБД.

Таблица 3.1 – Сравнение СУБД

Система управления базами данных	Бесплатное и открытое программное обеспечение	Поддержка процедурных расширений языка SQL	Наличие личного опыта работы
MySQL [13]	есть	нет	нет
Oracle [14]	нет	есть	нет
PostgreSQL [15]	есть	есть	есть

Исходя из сравнения в таблице 3.1 в качестве системы управления базами данных в данной работе будет использоваться PostgreSQL.

Как следует из таблицы 3.1 лучшим выбором СУБД для данного проекта будет PostgreSQL, так как он удовлетворяет всем критериям для выбора.

3.2 Архитектура приложения

Для проектирования винного интернет-магазина подходит клиент-серверная архитектура, так как там присутствует возможность более чёткого разграничения полномочий доступа к разным уровням информационной системы. Каждому клиенту — свой уровень доступа.

Приложение винного магазина будет состоять из следующих компонентов:

— интерфейс;

— бизнес-логика;

— доступ к данным.

Верхнеуровневое разбиение на компоненты представлено на рисунке 3.1.

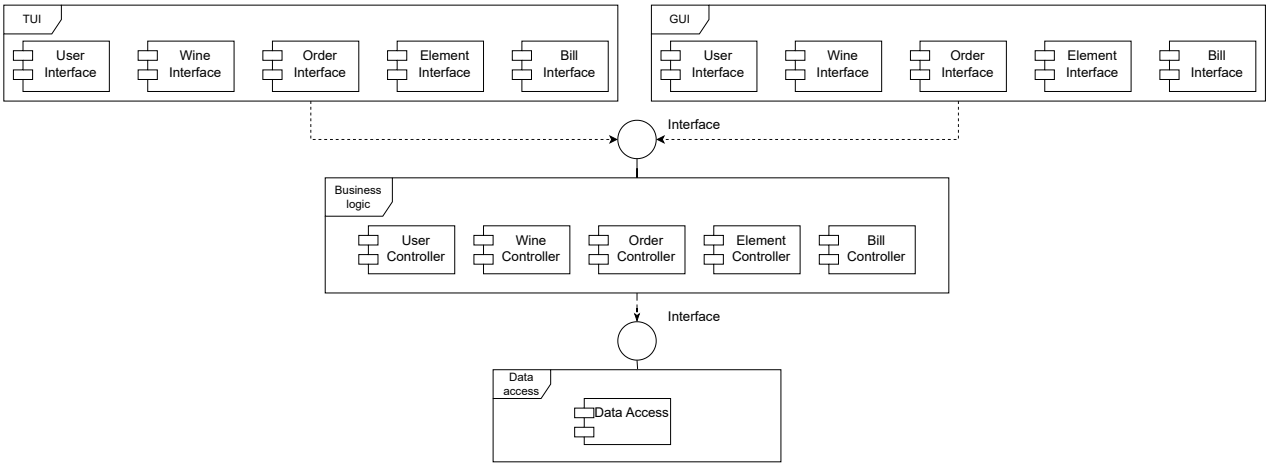

Рисунок 3.1 – Верхнеуровневое разбиение на компоненты

3.3 Средства реализации

Через API [16] будет предоставлен доступ на сервер. Средством реализации API был выбран Swagger [17], так как он автоматически описывает API на основе его кода. С помощью специального файла и OpenAPI Generator генерируются интерфейсы клиента и сервера. Обработчики запросов используют сгенерированные интерфейсы для осуществления http запросов.

Были реализованы следующие запросы:

1) /authorize

   Метод POST. Осуществляет авторизацию пользователя. В теле запроса находится структура состоящая из логина и пароля, которые являются строками. Ответом служит структура ошибочного ответа или структура пользователя, которая содержит поля:

   — id — строка;

   — login — строка;

   — password — строка;

— name — строка;

— email — строка;

— points — строка;

— status — строка;

2) /register

Метод POST. Осуществляет регистрацию пользователя. В теле запроса находится структура, состоящая из:

— login — строка;

— password — строка;

— name — строка;

— email — строка;

— status — строка.

Ответом служит структура ошибочного ответа или структура, которая содержит булевое поле registered.

3) /wines

Метод GET. Возвращает перечень вин. В теле запроса находится структура состоящая из строковых полей limit и skip. Ответом служит структура ошибочного ответа или структура, которая содержит массив вин.

4) /wines

Метод POST. Создает новое вино в каталоге. В теле запроса находится структура, состоящая из:

— name — строка;

— count — строка;

— year — целое число;

— strength — целое число;

— price — строка;

— type — строка.

Ответом служит структура ошибочного ответа или структура с булевым полем added.

5) /wines

Метод PUT. Изменяет вино в каталоге. В теле запроса находится структура вина. Ответом служит структура ошибочного ответа или структура с булевым полем updated.

6) /wines/id

Метод DELETE. Удаляет вино из каталога. Параметром является id. Ответом служит структура ошибочного ответа или структура с булевым полем deleted.

7) /wines/id

Метод GET. Получает конкретное вино из каталога. Параметром является id. Ответом служит структура ошибочного ответа или структура вина.

8) /elems

Метод POST. Создает элемент заказа с вином. В теле запроса находится структура элемента заказа. Ответом служит структура ошибочного ответа или структура с булевым полем created.

9) /elems

Метод GET. Получает все элементы данного заказа. В теле запроса находится структура со строчкой id заказа. Ответом служит структура ошибочного ответа или структура с массивом элементов заказа.

10) /elems/id/decrease

Метод PUT. Уменьшает количество вина в заказе на 1. Параметром является id элемента заказа. Ответом служит структура ошибочного ответа или структура с булевым полем decreased.

11) /elems/id/add

Метод PUT. Увеличивает количество вина в заказе на 1. Параметром является id элемента заказа. Ответом служит структура ошибочного ответа или структура с булевым полем added.

12) /elems/id

Метод DELETE. Удаляет элемент заказа. Параметром является id элемента заказа. Ответом служит структура ошибочного ответа или структура с булевым полем deleted.

13) /orders/id

Метод GET. Получает заказ по id. Параметром является id заказа. Ответом служит структура ошибочного ответа или структура заказа.

14) /orders

Метод PUT. Оформляет заказ. В теле запроса находится структура заказа. Ответом служит структура ошибочного ответа или структура с булевым полем placed.

15) /bills/id

Метод PUT. Подтверждает оплату счета. Параметром является id счета. Ответом служит структура ошибочного ответа или структура с булевым полем payed.

16) /users

Метод PUT. Обновляет баллы у пользователя. В теле запроса находится структура, которая содержит id пользователя и количество баллов, на которое нужно изменить баллы клиента. Ответом служит структура ошибочного ответа или структура с булевым полем updated.

17) /favourite/id

Метод GET. Получает любимые вина пользователя. Параметром является id пользователя. Ответом служит структура ошибочного ответа или структура с массивом любимых вин, которые состоят из идентификаторов пользователя и вина.

18) /favourite Метод POST. Добавляет вино в «любимое». В теле запроса находится структура, состоящая из идентификаторов пользователя и вина. Ответом служит структура ошибочного ответа или структура с булевым полем added.

19) /favourite Метод DELETE. Удаляет вино из «любимого». В теле запроса находится структура, состоящая из идентификаторов пользователя и вина. Ответом служит структура ошибочного ответа или структура с булевым полем deleted.

Приложение реализовано на языке Go[18], так как у него присутствуют библиотеки для написания тестов, для http-запросов[19], для подключения к базе данных[20]. Для доступа к базе данных был использован паттерн «репозиторий». Интерфейс приложения — консольный, который реализует MVC паттерн.

3.4 Детали реализации

3.4.1 Создание таблиц

В листингах 3.1 — 3.6представлен код создания 6 таблиц для приложения винного магазина. При создании сразу устанавливаются ограничения целостности. Таблицы создаются при миграции с помощью утилиты goose[21].

Листинг 3.1 – Создание таблицы пользователей

```
create table IF NOT EXISTS users (
    id UUID default uuid_generate_v4() primary key,
    login text not null,
    password text not null,
    fio text,
    email text not null check ( email like '%@%.%' ),
    points int default 0,
    status int default 0
);
```

Листинг 3.2 – Создание таблицы вин

```
create table IF NOT EXISTS wines (
        id UUID default uuid_generate_v4() primary key,
        name text not null,
        year int not null,
```

```
        strength int,
        price int not null,
        type text,
        count int default 10
);
```

Листинг 3.3 – Создание таблицы заказов

```
create table IF NOT EXISTS orders (
    id UUID default uuid_generate_v4() primary key,
    id_user UUID references users(id) on delete cascade,
    total_price int,
    status text not null ,
    is_points bool
);
```

Листинг 3.4 – Создание таблицы элементов заказа

```
create table IF NOT EXISTS order_elements (
    id UUID default uuid_generate_v4() primary key,
    id_order uuid references orders(id) on delete cascade,
    id_wine uuid references wines(id) on delete cascade,
    count int
);
```

Листинг 3.5 – Создание таблицы счетов

```
create table IF NOT EXISTS bills (
    id UUID default uuid_generate_v4() primary key,
    id_order uuid references orders(id) on delete cascade,
    price int,
    status text not null
);
```

Листинг 3.6 – Создание таблицы любимых вин

```
create table IF NOT EXISTS user_wines(
    id_user uuid references users(id) on delete cascade,
    id_wine uuid references wines(id) on delete cascade,
    primary key (id_user, id_wine)
);
```

3.4.2 Создание ролей на уровне базы данных

В данной работе представлено 3 роли на уровне базы данных: гость, пользователь и администратор. Создание ролей и выделение им прав, в соот-

ветствии с ролевой моделью, представлены в листинге 3.7.

Листинг 3.7 – Создание ролей на уровне базы данных

```
create role guest with login;
grant select on wines to guest;
grant select, insert on users to guest;

create role user with login password 'user_password';
grant guest to user;
grant select, insert, update, delete on orders to user;
grant select, insert, update, delete on order_elements to user;
grant insert, select on bills to user;
grant insert, delete, select on user_wines to user;

create role administrator login superuser;
```

Гостю предоставляется меню из следующих пунктов:

— Выход;

— Регистрация;

— Авторизация;

— Получить каталог вин.

Зарегистрированный пользователь получает меню, представленное ниже:

— Выход;

— Добавить вино в заказ;

— Увеличить количество вина в заказе;

— Уменьшить количество вина в заказе;

— Удалить вино из заказа;

— Получить каталог вин;

— Получить текущий заказ;

— Оформить заказ;

— Добавить вино в любимое;

— Получить список любимых вин;

— Удалить вино из любимого.

Для администратора выводится следующее меню:

— Выход;

— Регистрация;

— Получить каталог вин;

— Подтвердить оплату счета;

— Добавить вино;

— Удалить вино;

— Изменить вино.

3.4.3 Создание триггера

Разработан триггер для проверки элемента заказа перед его вставкой в таблицу. Необходимо удостовериться, что данного вина уже нет в заказе и что на складе хватает товара для продажи. Код триггера и соответствующей функции представлен в листинге 3.8.

Листинг 3.8 – Триггер, срабатывающий при добавлении нового элемента заказа

```
-- +goose Up
-- +goose StatementBegin
create or replace function check_element()
returns trigger as $$
declare
elem_count int;
wine_count int;
begin
select count(*)
into elem_count
from order_elements
where id_wine = new.id_wine and id_order = new.id_order;
```

```
if elem_count > 0 then
        raise exception
       'wine already in order.';
end if;


select w.count
into wine_count
from wines w
where w.id = new.id_wine;

if wine_count < new.count then
        raise exception
        'not enough count of wine.';
end if;

return new;
end;
$$ language plpgsql;

create trigger element_trigger
    before insert on order_elements
    for each row
    execute function check_element();

-- +goose StatementEnd

-- +goose Down
-- +goose StatementBegin
drop trigger if exists element_trigger on order_elements;
-- +goose StatementEnd
```

3.5 Тестирование

Тестирование было проведено с помощью модульных и интеграционных тестов. Для модульных тестов были сгенерированы заглушки на репозитории для досткупа к данным с помощью пакета minimock[22]. Модульные тесты написаны для компонента бизнес логики, а именно для контроллеров вина, пользователя, заказа, элемента заказа и счета. Каждый интеграционный

тест выполняется в отдельном Docker[23] контейнере, до теста запускается миграция для создания таблиц и скрипт по заполнению тестовыми данными. Интеграционными тестами проверено получение заказа и обновление баллов у пользователя. Все тесты пройдены успешно.

Вывод

В данном разделе была описана архитектура приложения, средства и детали его реализации.

4 Исследовательский раздел

4.1 Описание исследования

Целью данного исследования является сравнение времени выполнения запроса на создания нового элемента заказа в зависимости от расположения вычислений: на уровне базы данных или на уровне бизнес логики.

Для добавления нового элемента заказа необходимо проверить, хватает ли вина на складе и не добавлено ли уже такое вино в данный заказ. Для данных проверок был написан триггер, который срабатывает при каждом добавлении элемента заказа. А также реализована еще одна функция вставки на уровне бизнес-логики. С помощью библиотеки Faker[24] были сгенерированы правдоподобные данные для пользователей. На время исследования количество пользователей постоянно и равно 10. Также были сгенерированы заказы. Их количество тоже постоянно на протяжении всего исследования. В таблице с винами вставлены 3960 вин. Количество элементов заказов в таблице меняется от 10 до 100 с шагом в 10, от 100 до 1000 с шагом 100, от 1000 до 10000 с шагом 1000. Для каждого запуска исследования поднимается свой тестовый контейнер базы данных.

4.2 Результат исследования

Результаты замеров (в мс.) приведены на рисунке **??**. Для получения более достоверных данных, элемент заказа вставлялся 500 раз, а затем бралось среднее значение времени.

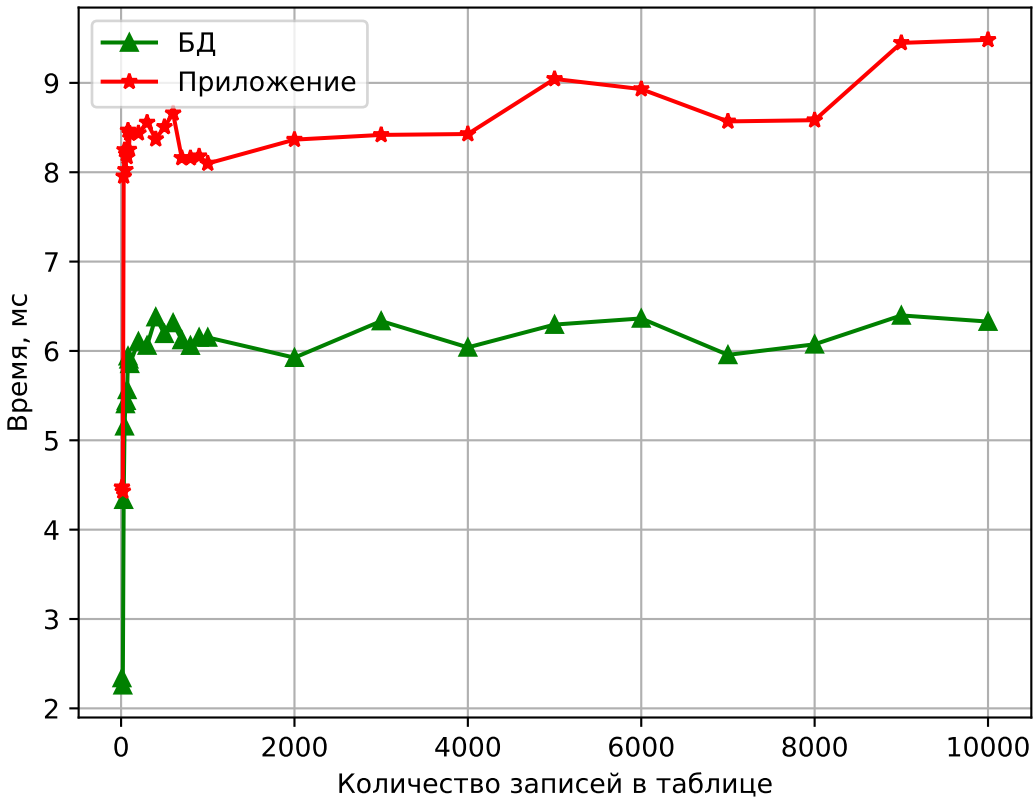

Рисунок 4.1 – Сравнение времени проверки нового элемента заказа от расположений вычислений

По полученным данным можно сказать, что вычисления на уровне базы данных в среднем в полтора раза быстрее, чем на уровне приложения, так как из приложения приходится несколько раз обращаться к базе данных, чтобы получить информацию из разных таблиц.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы достигнута поставленная цель: разработана база данных для винного магазина.

В ходе выполнения работы были решены все задачи:

1) формализована задачу, определен необходимый функционал;
2) проведен анализ существующих СУБД;
3) спроектирована база данных, необходимая для хранения и структурирования данных;
4) программно реализована спроектированная база данных с использованием выбранной СУБД;
5) разработано программное обеспечение;
6) реализован пользовательский интерфейс;
7) проведен сравнительный анализ обработки данных в зависимости от расположения вычислений на приложении или на базе данных.

Разработанную базу данных в будущем можно улучшить добавлением новых пользовательских сценариев, а также добавлением сортировки вина по цене, стране и крепости.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Алкоголь на фоне пандемии [Электронный ресурс]. — Режим доступа: https://wciom.ru/analytical-reviews/analiticheskii-obzor/alkogol-na-fone-pandemii (дата обращения: 10.04.2023).
2. Министерство здравоохранения Российской Федерации [Электронный ресурс]. — Режим доступа: https://minzdrav.gov.ru/ (дата обращения: 10.04.2023).
3. Министерство сельского хозяйства Российской Федерации [Электронный ресурс]. — Режим доступа: https://mcx.gov.ru/?ysclid=lmjyx2j5at578625547 (дата обращения: 10.09.2023).
4. Правила продажи товаров при дистанционном способе продажи товара по договору розничной купли-продажи [Электронный ресурс]. — Режим доступа: https://sudact.ru/law/postanovlenie-pravitelstva-rf-ot-31122020-n-2463/pravila-prodazhi-tovarov-po-dogovoru/i/pravila-prodazhi-tovarov-pri-distantsionnom/ (дата обращения: 10.04.2023).
5. Роб, П., Коронелл, К. Базы данных: концепции, технологии, применение. — БХВ-Петербург: Вильямс — 2004. — 15 с.
6. Карпова, И. П. Базы данных. Учебное пособие. — Московский государственный институт электроники и математики (Технический университет) — 2009.
7. Сетевая модель данных службы каталогов. — Труды Московского физико-технического института — 2014.
8. Борисова Д. А., Лядова Л. Н. Иерархическая модель данных как основа реализации информационной системы, управляемой метаданными. — Математика программных систем. – 2006. – 4-13 с.
9. Жалолов О. И., Хаятов Х. У. Понятие SQL и реляционной базы данных. — Universum: технические науки. – 2020. – №. 6-1 (75). – 26-29 с.
10. Красное&Белое. — Режим доступа: https://krasnoeibeloe.ru (дата обращения: 10.04.2023).

11. ВИНЛАБ. — Режим доступа: https://www.winelab.ru (дата обращения: 10.04.2023).

12. wine.com. — Режим доступа: https://www.wine.com (дата обращения: 10.04.2023).

13. MySQL – the world's most popular open source database [Электронный ресурс]. – Режим доступа: https://www.mysql.com/, свободный – (06.06.2023).

14. Oracle database services and products offer customers cost-optimized and high-performance versions of Oracle Database, the world's leading converged, multi-model database management system. [Электронный ресурс]. – Режим доступа: https://www.oracle.com/database/, свободный – (06.06.2023).

15. PostgreSQL: The World's Most Advanced Open Source Relational Database [Электронный ресурс]. – Режим доступа: https://www.postgresql.org/, свободный – (12.03.2023).

16. What is API? [Электронный ресурс]. – Режим доступа: https://www.ibm.com/topics/api свободный – (12.03.2023).

17. Документация по OpenAPI Generator [Электронный ресурс]. — Режим доступа: `https://openapi-generator.tech/` (дата обращения: 15.08.2023).

18. Golang – компилируемый многопоточный язык программированияе [Электронный ресурс]. – Режим доступа: https://pkg.go.dev/std/, свободный – (12.03.2023).

19. Документация пакета snet/http языка Go [Электронный ресурс]. — Режим доступа: `https://pkg.go.dev/net/http` (дата обращения: 13.05.2023).

20. Документация библиотеки GORM для Golang [Электронный ресурс]. — Режим доступа: `https://gorm.io/docs/` (дата обращения: 13.05.2023).

21. Документация пакета goose языка Go [Электронный ресурс]. — Режим доступа: `https://pkg.go.dev/github.com/pressly/goose` (дата обращения: 13.05.2023).

22. Документация пакета minimock языка Go [Электронный ресурс]. — Режим доступа: `https://pkg.go.dev/github.com/gojuno/minimock/v3` (дата обращения: 13.05.2023).

23. Docker is a platform designed to help developers build, share, and run modern application. [Электронный ресурс]. – Режим доступа: https://docs.docker.com/, свободный – (12.03.2023).

24. Faker is a Python package that generates fake data for you.: [Электронный ресурс]. – Режим доступа: https://faker.readthedocs.io/en/master/, свободный – (30.05.2023).

ПРИЛОЖЕНИЕ А