Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Модели данных и системы управления базами данных»

|  |  |
| --- | --- |
|  | «К ЗАЩИТЕ ДОПУСТИТЬ» |
|  | Руководитель курсового проекта  ассистент кафедры информатики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.В.Давыдчик |
|  | \_\_\_.\_\_\_\_.2024 |

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту

на тему

**ПРОГРАММНОЕ СРЕДСТВО ДЛЯ ИНТЕРНЕТ-МАГАЗИНА**

**ЭЛЕКТРОНИКИ**

БГУИР КП 1-40 04 01

|  |  |
| --- | --- |
|  | Выполнил студент группы 153501  Власенко Тимофей Павлович  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |
|  | Курсовой проект представлен на проверку \_\_\_.\_\_\_\_.2024  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |

Минск 2024

**СОДЕРЖАНИЕ**

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc182755270)

[1.1 Обзор веб-сайта *Onliner* 5](#_Toc182755271)

[2 ФОРМИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ 8](#_Toc182755272)

[2.1 Функциональные требования к разрабатываемой базе данных 8](#_Toc182755273)

[2.2 Анализ существующих подходов к разработке баз данных 9](#_Toc182755274)

[2.3 Выбор базы данных для разработки предметной области 12](#_Toc182755275)

[3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 14](#_Toc182755276)

[3.1 Инфологическая модель 14](#_Toc182755277)

[3.2 Разработка даталогической модели базы данных 14](#_Toc182755278)

[4 РАЗРАБОТКА БАЗЫ ДАННЫХ 15](#_Toc182755279)

[4.1 Создание исходных таблиц, индексов и ограничений 15](#_Toc182755280)

[4.2 Создание хранимых процедур 15](#_Toc182755281)

[4.3 Реализация триггеров 15](#_Toc182755282)

[ЗАКЛЮЧЕНИЕ 16](#_Toc182755283)

[СПИСОК ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ 17](#_Toc182755284)

[ПРИЛОЖЕНИЕ А (обязательное) Листинг кода программы 18](#_Toc182755285)

[ПРИЛОЖЕНИЕ Б (обязательное) Схема базы данных 19](#_Toc182755286)

**ВВЕДЕНИЕ**

Интернет-магазины стали неотъемлемой частью современной торговли и повседневной жизни. Они предоставляют пользователям удобный способ приобретать товары, экономя время и ресурсы. Среди разнообразия товаров, доступных онлайн, электроника занимает одно из ведущих мест благодаря постоянному спросу и технологическому прогрессу. С ростом популярности онлайн-покупок увеличивается и необходимость в эффективных системах, которые позволяют организовать, управлять и оптимизировать процессы продажи электроники.

Цифровые платформы для интернет-магазинов электроники помогают пользователям быстро находить нужные товары, сравнивать характеристики и цены, оформлять заказы и отслеживать их доставку. Они также предоставляют владельцам магазинов удобные инструменты для управления ассортиментом, аналитики и взаимодействия с клиентами. Однако многие существующие решения не полностью удовлетворяют запросы пользователей, такие как сложная навигация, недостаточная персонализация или ограниченные возможности аналитики. Это подчеркивает актуальность разработки более функциональных и удобных систем.

Целью данной курсовой работы является создание концепции и прототипа программного обеспечения для интернет-магазина электроники. Разрабатываемое ПО позволит пользователям легко находить и приобретать электронику, обеспечит гибкие функции фильтрации и сортировки, а также предоставит владельцам магазина инструменты для оптимизации управления и повышения качества обслуживания клиентов. Для достижения поставленной цели необходимо решить следующие задачи:

Подведем основные цели на данный курсовой проект:

– провести анализ предметной области и изучить существующие аналоги;

– описать основные сущности проектируемой базы данных и связи между ними;

– разработать структуру базы данных, включая индексы для оптимизации поиска;

– создать триггеры и хранимые процедуры для обеспечения корректной работы системы;

– разработать программное обеспечение, использующее данные, которые хранятся в созданной базе данных.

# 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Современные платформы для интернет-магазинов являются популярным инструментом для поиска и покупки товаров. Среди множества приложений и сервисов можно выделить один из наиболее популярных веб-сайтов ­– *Onliner*.

## Обзор веб-сайта *Onliner*

*Onliner*.*by* (стилизуется как *Onlíner*) – белорусский сайт, включающий в себя СМИ, маркетплейс товаров и услуг, а также форум. На рисунке 1.1 представлен графический интерфейс главной страницы веб-сайта:

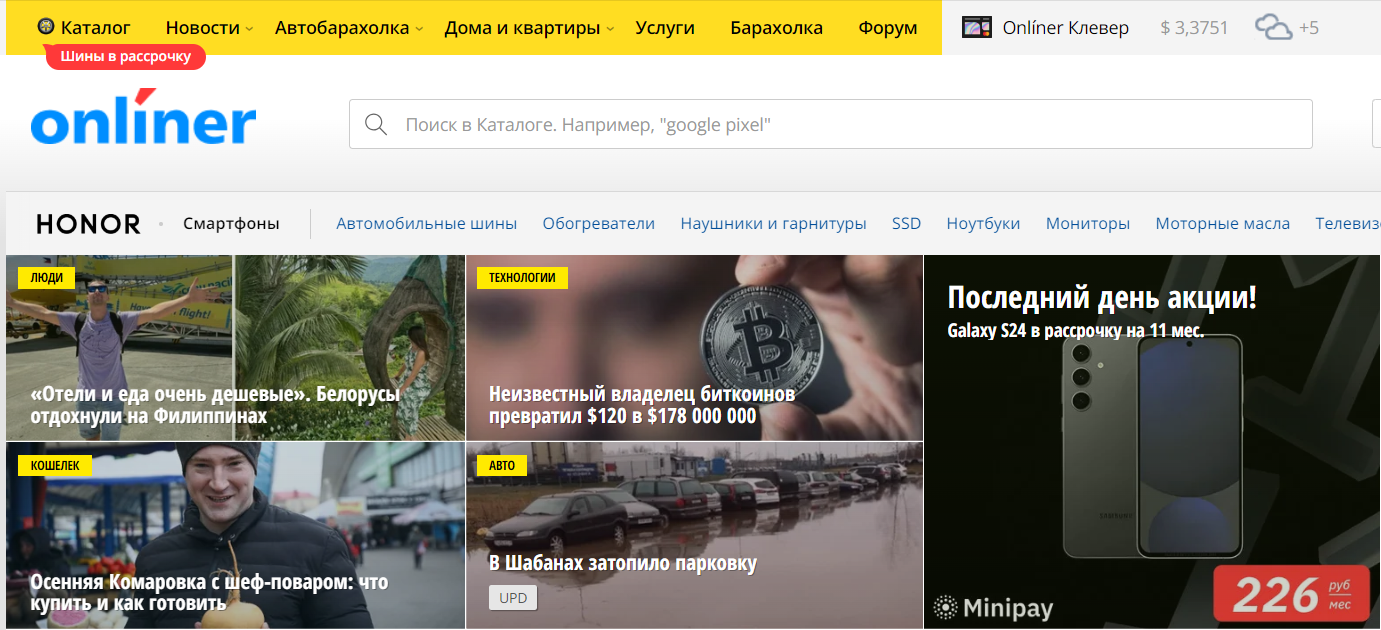


Рисунок 1.1 – Графический интерфейс главной страницы *Onliner*

*Onliner* предлагает пользователям мощный инструмент поиска и фильтрации, который позволяет находить товары по множеству параметров, таких как производитель, год выпуска, технические характеристики, рейтинг и другие.

Пример фильтров и параметров для поиска представлен на рисунке 1.2:

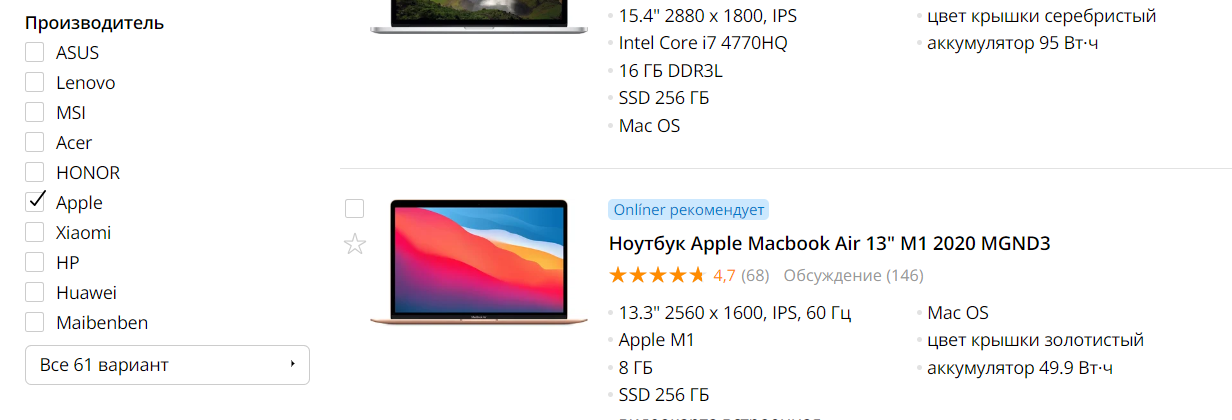


Рисунок 1.2 – Фрагмент списка ноутбуков на *Onliner*

Каждая единица контента на *Onliner* представлена в виде подробной страницы, где пользователи могут найти следующую информацию: краткое описание характеристик и составляющих товара, изображения, отзывы, стоимость, список продавцов.

Эта структура страницы позволяет быстро получить представление о любом товаре. Карточка также часто включает трейлеры, галерею изображений и ссылки на похожие фильмы.

На рисунке 1.3 продемонстрирован пример такой страницы для ноутбука:

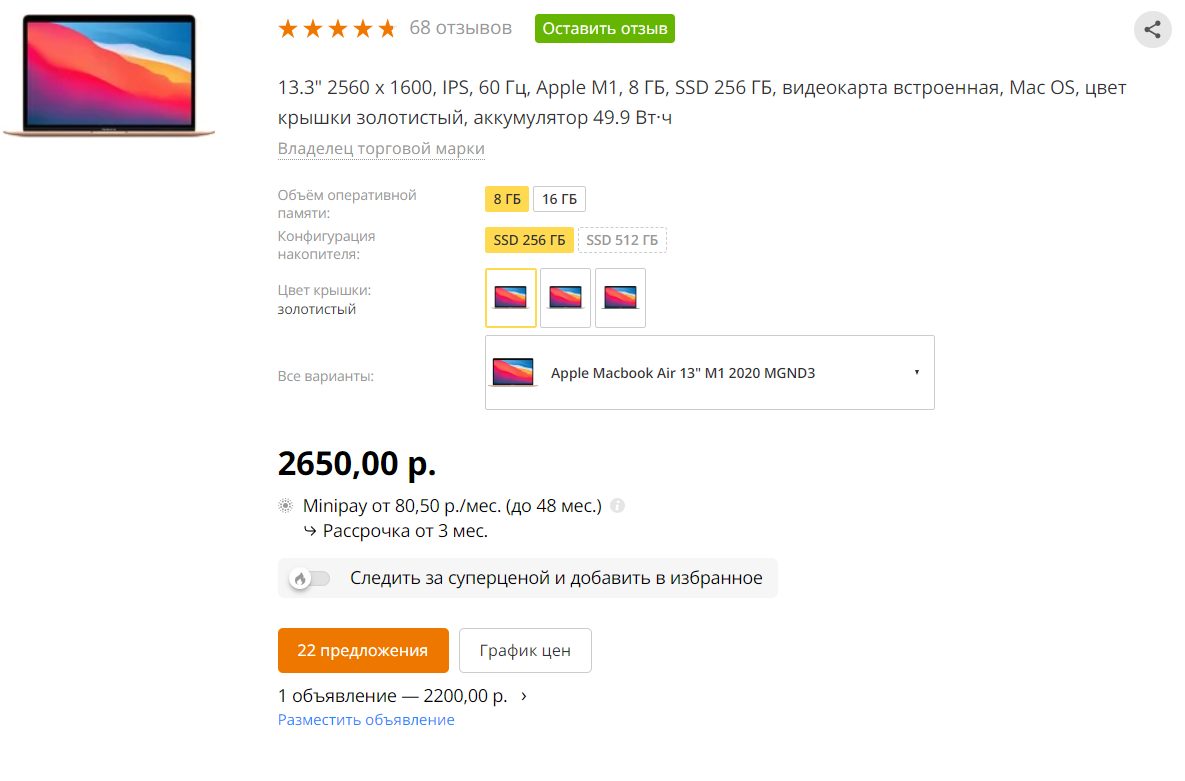


Рисунок 1.3 – Карточка с описанием выбранного ноутбука

*Onliner* предоставляет пользователям возможность оценивать товары и оставлять отзывы.

На рисунке 1.4 представлена страница с отзывами пользователей на выбранный ноутбук:

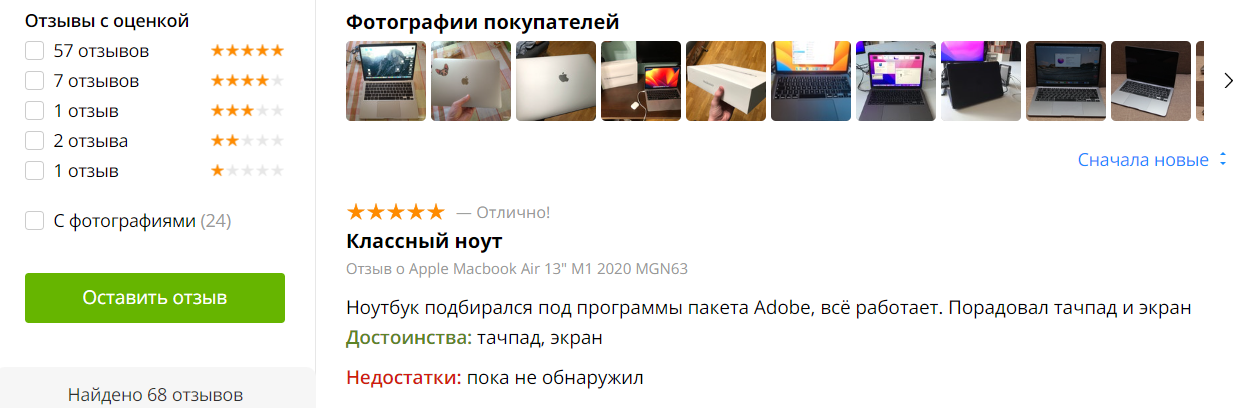


Рисунок 1.4 – Страница с отзывами на выбранный ноутбук

*Onliner* обладает рядом преимуществ, которые сделали его одной из ведущих платформ в своей области:

1 Широкий охват информации. Представлены товары различных видов и категорий.

2 Пользовательские рейтинги и отзывы.

3 Обширная база данных.

Однако *Onliner* также имеет некоторые недостатки:

1 Помимо маркетплейса представлено множество других возможностей, ненужных для интернет-магазина.

2 Отсутствие специализации на конкретных товарах.

*Onliner* является одним из самых известных маркетплейсов на белорусском рынке. Однако для создания полнофункциональной интерент магазина может потребоваться более персонализированное решение, которое позволит пользователям эффективнее совершать поиск и покупки необходимой техники.

# 2 ФОРМИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ

## 2.1 Функциональные требования к разрабатываемой базе данных

На основе анализа существующих аналогов выдвинем функциональные требования к разрабатываемому программному средству:

1 Авторизация пользователя. Пользователи должны иметь возможность авторизоваться в системе, используя свои учетные данные, и получать доступ к расширенным функциям в зависимости от своей роли.

2 Управление пользователями (CRUD). Система должна позволять создание, чтение, обновление и удаление учетных записей пользователей, обеспечивая гибкость и контроль для администраторов.

3 Ролевая система. Необходимо реализовать систему ролей, где предусмотрены роли администратора и обычного пользователя. Разграничение прав доступа позволит обеспечить безопасность и функциональность системы.

4 История действий пользователей. Система должна вести журнал всех действий пользователей для отслеживания истории взаимодействий. Пользователи смогут просматривать свои действия, а администраторы — действия всех пользователей.

После авторизации обычные пользователи должны иметь следующие возможности:

1 Возможность находить товары по категориям.

2 Получение информации о товаре.

3 Возможность оставить отзыв.

4 Создания заказа.

Пользователи с ролью администратора будут иметь доступ к расширенным функциям:

1 Доступ к функциям пользователей. Возможность выполнять все те же действия, что и обычные пользователи.

2 Просмотр истории действий пользователя и заказов.

3 Добавления новых сотрудников.

Эти требования обеспечат надежное управление данными, удобство использования для пользователей и функциональные возможности для администраторов при поддержке контроля доступа и безопасности системы.

## 2.2 Анализ существующих подходов к разработке баз данных

В современной разработке баз данных выделяют два основных типа: SQL и NoSQL. Их отличие связано не только с использованием SQL-запросов, но и с принципами организации данных. SQL базы данных, основанные на реляционной модели, представляют данные в виде таблиц с четко определенными связями, что обеспечивает целостность данных и удобную обработку с помощью SQL. NoSQL базы данных, напротив, не придерживаются реляционной теории и могут хранить данные в различных форматах (документы, графы, колонки и др.), что позволяет гибко адаптироваться к задачам масштабируемых систем.

Для лучшего понимания структуры и подвидов этих баз данных обратимся к рисунку 2.1, который иллюстрирует основные категории SQL и NoSQL баз.

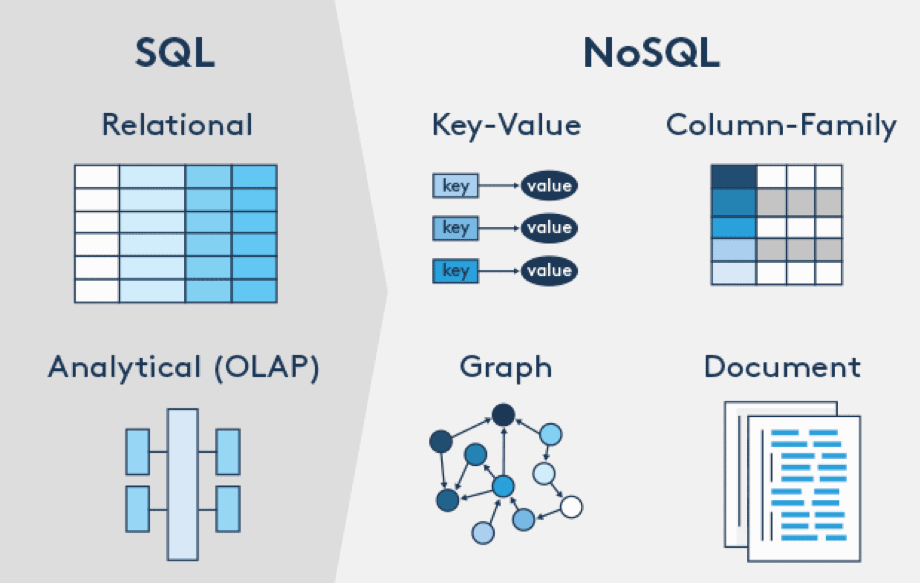


Рисунок 2.1 – Основные виды SQL и NoSQL баз данных

Далее рассмотрим более подробно основные отличия реляционных и нереляционных баз данных.

**2.2.1 Реляционные базы данных**

Реляционные базы данных – это базы данных, которые используют табличную структуру для хранения и обработки данных. В них информация организована в виде таблиц, состоящих из строк и столбцов, где каждая строка представляет собой запись с уникальным ключом. Столбцы содержат атрибуты, описывающие характеристики записей. Связи между таблицами устанавливаются с помощью ключей, что поддерживает целостность и согласованность данных.

Для управления реляционными базами данных применяется стандартный язык SQL, позволяющий выполнять различные операции с данными: выборку, добавление, обновление, удаление, объединение, агрегацию и другие. SQL также поддерживает транзакции – неделимые блоки операций, которые подчиняются принципам ACID:

1 Атомарность (Atomicity). Транзакция выполняется как единое целое, «всё или ничего». Это значит, что либо все операции в транзакции выполняются успешно, либо ни одна из них не выполняется. Если какая-то операция в транзакции не удается, происходит откат, и база данных возвращается к состоянию до начала транзакции. Например, при переводе денег между счетами атомарность гарантирует, что деньги спишутся с одного счета только в том случае, если они зачислятся на другой.

2 Согласованность (Consistency). После завершения транзакции база данных должна перейти из одного согласованного состояния в другое. Согласованность подразумевает соблюдение всех установленных правил и ограничений. Например, если для определённого столбца указано ограничение на уникальные значения, транзакция, нарушающая это правило, не будет завершена.

3 Изолированность (Isolation). Параллельные транзакции не должны влиять друг на друга, обеспечивая независимость их выполнения. Изолированность гарантирует, что одна транзакция не сможет «увидеть» изменения, вносимые другой, пока та не завершится. Существуют разные уровни изолированности, такие как «чтение с повторной фиксацией» и «серийная изолированность», которые регулируют степень изолированности транзакций.

4 Долговечность (Durability). После успешного завершения транзакции ее результаты сохраняются в базе данных даже в случае сбоя системы. Это означает, что изменения в данных, внесенные транзакцией, сохраняются в долгосрочной перспективе и остаются доступными после завершения транзакции.

**2.2.3 Нереляционные базы данных**

NoSQL – это обширная категория систем управления базами данных, которые не соответствуют традиционной реляционной модели. Основные типы NoSQL баз данных включают:

– документо-ориентированные базы данных;

– базы данных ключ-значение;

– графовые базы данных;

– колоночные базы данных.

Далее данные типы будут рассмотрены более подробно.

Документо-ориентированные базы данных хранят информацию в виде документов, которые имеют свою структуру и могут включать различные типы данных. Документы группируются в коллекции, которые могут иметь разные схемы. Для работы с такими базами данных используются специализированные языки запросов, позволяющие обращаться к данным по их атрибутам.

MongoDB – одна из самых известных и мощных документо-ориентированных баз данных, поддерживающая различные форматы документов, такие как JSON, BSON и XML. MongoDB характеризуется высокой производительностью, масштабируемостью и гибкостью, а также предоставляет множество функций для работы с данными, таких как агрегация, индексация, шардирование и репликация.

Далее перейдем к базам данных ключ-значение. Эти базы данных хранят данные в виде пар ключ-значение, где ключ служит уникальным идентификатором, а значение может быть любым типом данных. Базы данных ключ-значение обеспечивают быстрый доступ к данным по ключу, однако не поддерживают сложные запросы и связи между данными.

Redis – популярная база данных, которая хранит данные в оперативной памяти и поддерживает различные типы значений, такие как строки, списки и множества. Redis также предлагает функции, такие как транзакции, репликация и кэширование.

Графовые базы данных хранят данные в виде узлов и ребер, представляющих сущности и связи между ними. Они подходят для моделирования сложных сетей и отношений, таких как социальные сети и рекомендательные системы. Для работы с графовыми базами данных используются специализированные языки запросов, которые позволяют искать пути и паттерны в графах.

Графовые базы данных могут быть разделены на два основных типа: базы данных свойственных графов, где узлы и ребра имеют атрибуты, и базы данных знаний, где узлы и ребра имеют семантические метки. Примеры баз данных свойственных графов: Neo4j и ArangoDB.

Колоночные базы данных представляют собой один из типов NoSQL систем управления базами данных, которые хранят данные в виде колонок, а не строк. Это подход позволяет эффективно организовывать и обрабатывать большие объемы данных, особенно в сценариях, требующих высокой производительности при выполнении аналитических запросов. Колоночные базы данных оптимизированы для работы с большими наборами данных и часто используются в аналитических приложениях, бизнес-аналитике и хранилищах данных.

Основные характеристики колоночных баз данных включают в себя:

1 Хранение по колонкам. Данные хранятся в виде отдельных колонок, что позволяет эффективно считывать только необходимые данные без необходимости загружать всю строку. Это значительно повышает производительность при выполнении агрегирующих запросов и выборок по определённым колонкам.

2 Оптимизация для чтения. Колоночные базы данных идеально подходят для сценариев, где требуется высокая скорость обработки запросов и анализ данных. Это достигается благодаря тому, что данные одной колонки хранятся вместе, что позволяет лучше использовать кэш и сократить время доступа к данным.

3 Сжатие данных. Поскольку данные в колонках часто имеют одинаковый тип, это позволяет применять более эффективные алгоритмы сжатия, что приводит к уменьшению объема хранимых данных и экономии дискового пространства.

4 Гибкость схемы. Колоночные базы данных могут поддерживать динамические схемы, что позволяет добавлять новые колонки без необходимости изменения всей структуры базы данных.

Apache Cassandra – это распределенная колоночная база данных, предназначенная для обработки больших объемов данных. Cassandra обеспечивает высокую доступность и отказоустойчивость благодаря дублированию данных на нескольких узлах. Она идеально подходит для работы с веб-приложениями и IoT-решениями.

Таким образом, NoSQL базы данных предлагают гибкость, масштабируемость и производительность, что делает их подходящими для работы с большими объемами данных и динамически меняющимися схемами.

## 2.3 Выбор базы данных для разработки предметной области

Исходя из анализа подходов, представленных в предыдущем разделе, можно заключить, что предметная область для магазина электроники будет реализована с использованием реляционных баз данных. Это объясняется тем, что в данной области четко определены атрибуты, сущности и связи, которые удобно отображать в виде таблиц и ключей в рамках реляционной модели.

Для обеспечения корректной работы системы необходимо гарантировать целостность и согласованность данных, что особенно важно для избежания ошибок при создании заказов.

В качестве базы данных для разработки была выбрана PostgreSQL.

Преимущества PostgreSQL заключаются в следующем:

1 Эффективный механизм обработки запросов. PostgreSQL использует продвинутые алгоритмы, что позволяет оптимально работать с данными.

2 Поддержка параллелизма. Это означает, что несколько запросов могут выполняться одновременно, что существенно увеличивает общую производительность.

3 Масштабируемость. PostgreSQL может масштабироваться как вертикально, увеличивая ресурсы на одном сервере, так и горизонтально, распределяя нагрузку между несколькими серверами.

Система также подходит для различных типов приложений, включая:

1 Корпоративные решения. PostgreSQL идеальна для крупных организаций, которым необходимы высокая производительность и надежность.

2 Веб-приложения. Эта СУБД часто используется для разработки масштабируемых и надежных веб-приложений.

3 Интернет-магазины. PostgreSQL может эффективно обрабатывать транзакции и управлять товарными запасами и клиентами.

PostgreSQL обеспечивает высокий уровень безопасности данных, предлагая разнообразные методы аутентификации, такие как пароли, сертификаты и двухфакторная аутентификация. Она также поддерживает ролевые и объектно-ориентированные схемы авторизации, что позволяет администраторам контролировать доступ пользователей к данным и функциям. База данных включает методы шифрования, как для хранения, так и для передачи данных, защищая информацию от несанкционированного доступа.

Таким образом, PostgreSQL была выбрана благодаря своей высокой производительности, масштабируемости и богатым возможностям безопасности.

# 3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

## 3.1 Инфологическая модель

## 3.2 Разработка даталогической модели базы данных

# 4 РАЗРАБОТКА БАЗЫ ДАННЫХ

## 4.1 Создание исходных таблиц, индексов и ограничений

## 4.2 Создание хранимых процедур

## 4.3 Реализация триггеров

# ЗАКЛЮЧЕНИЕ

Результатом работы данного курсового проекта является база данных для интернет-магазина электроники.

Для разработки была выбрана реляционная СУБД PostgreSQL, обладающая своими преимуществами перед известными конкурентами. Как итог можно сказать, что цели данной курсовой работы достигнуты.

# СПИСОК ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЕ А  
(обязательное)  
Листинг кода программы

ПРИЛОЖЕНИЕ Б  
(обязательное)  
Схема базы данных