МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«КУРГАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Кафедра «Программное обеспечение автоматизированных систем»

**Проект БД для магазина по продаже вычислительной техники**

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**РФ 41287462 260091 81**

**Листов 19**

Курган 2025

**АННОТАЦИЯ**

Документ содержит описание этапов проектирования, структуры базы данных и ключевых моделей, на которых строится база данных для магазина по продаже вычислительной техники.

Оглавление

[ВВЕДЕНИЕ 4](#_Toc198801106)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc198801107)

[2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 7](#_Toc198801108)

[2.1 Модель вариантов использования 7](#_Toc198801109)

[2.2 ER–модель 8](#_Toc198801110)

[2.3 Стадия технического проекта 10](#_Toc198801111)

[2.3.1 Получение исходной схемы реляционной базы данных 10](#_Toc198801112)

[2.3.2 Нормализация исходной схемы реляционной базы данных 12](#_Toc198801113)

[3 ОБОСНОВАНИЕ СРЕДСТВ РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ 15](#_Toc198801114)

[ЗАКЛЮЧЕНИЕ 18](#_Toc198801115)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 19](#_Toc198801116)

# ВВЕДЕНИЕ

Современный магазин по продаже вычислительной техники сталкивается с необходимостью эффективного управления ассортиментом, учёта продаж и обработки заказов клиентов. Системы управления базами данных позволяют автоматизировать ключевые процессы: приём и обработку заказов, отслеживание наличия товаров на складе, управление клиентскими данными, учёт поставок от поставщиков и формирование аналитической отчётности. В условиях роста числа клиентов и расширения ассортимента особенно важно иметь централизованное и масштабируемое решение для хранения и обработки данных.

Целью данного курсового проекта является разработка системы для уменьшения трудозатрат на основные бизнес-процессы розничного магазина, оптимизация учета складских запасов и формирования аналитических отчетов.

Для достижения этой задачи были поставлены следующие подзадачи:

1. Провести анализ предметной области;
2. Спроектировать базу данных:

* Диаграмма вариантов использования;
* ER-модель;
* Схема реляционной базы данных;

1. Реализовать базу данных:

* Реализация структуры базы;
* Реализация бизнес-логики.

# 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В розничных магазинах, специализирующихся на продаже вычислительной техники, ключевыми элементами являются управление ассортиментом товаров, обработка клиентских заказов и взаимодействие с поставщиками. В рамках таких систем необходимо учитывать наличие товаров на складе, оформление покупок, обработку возвратов, а также логистику поставок от поставщиков.

Каждый заказ клиента должен содержать информацию о покупателе, перечне выбранных товаров, статусе выполнения (например: оформлен, оплачен, в обработке, отправлен, завершён, возвращён) и, при необходимости, данные о доставке. Клиенты могут совершать покупки как через веб-интерфейс, так и непосредственно в физическом магазине.

Товары в каталоге должны быть организованы по категориям (ноутбуки, ПК, мониторы, комплектующие и т.д.), иметь характеристики, цену и информацию о наличии на складе. Для обеспечения бесперебойной работы необходим учёт связей между товарами и поставщиками, а также возможность формирования заявок на пополнение ассортимента.

Поставщики предоставляют товары по договорам, поэтому важно хранить контактные данные и историю поставок. Также может существовать система скидок или предложений от поставщиков на определённые группы товаров.

Сотрудники магазина работают с системой для обработки заказов, управления складскими остатками, регистрации возвратов и предоставления информации клиентам.

Типовой сценарий работы системы может выглядеть следующим образом:

1. Клиент заходит в интернет-магазин или приезжает в торговый зал.
2. Выбирает интересующие товары и добавляет их в корзину.
3. Оформляет заказ и адрес доставки при необходимости.
4. Заказ регистрируется в системе. Система проверяет наличие товара на складе и блокирует его.
5. После подтверждения наличия заказ переходит в статус «Обрабатывается».
6. Сотрудник готовит заказ к отгрузке, при необходимости формируется заявка на доукомплектование.
7. При получении товара клиентом, статус заказа изменяется на «Оплачен». Если возникает какая-то проблема с выполнением заказа, он может быть отменен.
8. Параллельно с этим сотрудники мониторят остатки товаров на складе. При снижении количества ниже порогового уровня формируется заявка на закупку у поставщика.
9. Все операции записываются в систему, что позволяет формировать аналитические отчёты по продажам, популярным товарам, загрузке сотрудников, эффективности поставщиков и другим метрикам.
10. Вся информация хранится в централизованной базе данных, которая обеспечивает целостность данных, безопасность и удобство администрирования. Такая система позволяет повысить прозрачность бизнес-процессов, минимизировать ошибки и улучшить качество обслуживания клиентов.

# 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

## 2.1 Модель вариантов использования

Диаграмма вариантов использования представляет собой инструмент для моделирования требований к разрабатываемой системе. Она позволяет определить участников системы, их роли, а также функциональные возможности, предоставляемые системой.

На диаграмме вариантов использования изображаются элементы:

* акторы – внешние сущности, взаимодействующие с системой;
* варианты использования – функции, которые система предоставляет акторам;
* отношения между акторами и вариантами использования, демонстрирующие взаимодействие участников с функциональными возможностями системы.

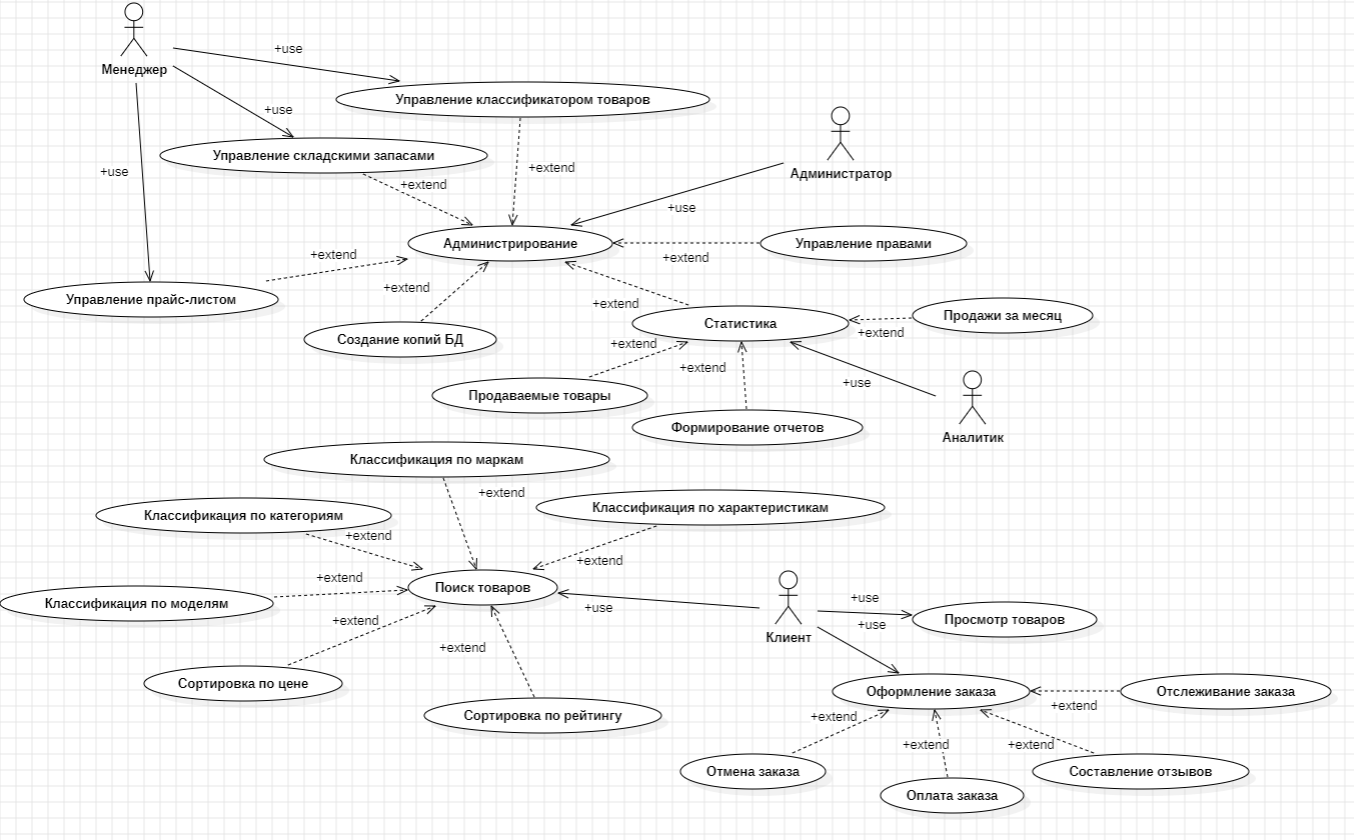
Диаграмма вариантов использования для проектируемой системы изображена на рисунке 1. 

Рисунок 1 – Диаграмма вариантов использования

На диаграмме вариантов использования (см. рисунок 1) представлены следующие акторы:

* Администратор
* Менеджер
* Аналитик
* Клиент

Диаграмма охватывает все ключевые бизнес-процессы розничного магазина. Акторы взаимодействуют с системой в рамках следующих сценариев:

* **Администратор** осуществляет управление правами, ведение учета товаров при необходимости, следит за корректностью работы базы, создание резервных копий;
* **Менеджер** занимается управлением складскими запасами, прайс-листом и классификатором товаров;
* **Клиенты** могут просматривать список товаров, искать товары с применением фильтров по разнообразным характеристикам, оформлять заказы, отслеживать их, оплачивать, отменять, а также пользоваться скидками и оставлять отзывы;
* **Отдел аналитики** формирует отчётность, анализирует объёмы продаж и продаваемые товары.

Таким образом, диаграмма чётко отображает распределение функций между участниками бизнес-процесса и логически структурирует взаимодействие всех ролей с системой.

## 2.2 ER–модель

Следующим этапом, идущим после выявления всех требований, является построение концептуальной модели базы данных ER-диаграммы. ER-диаграмма – схема, представляющая сущности и связи между ними.

Сущность представляет собой набор однотипных реальных объектов, каждый из которых является экземпляром данной сущности. Каждая сущность характеризуется атрибутами - именованными свойствами, описывающими ее характеристики. Атрибуты подразделяются на идентифицирующие, которые однозначно определяют экземпляр сущности, и описательные, содержащие дополнительную информацию.

Связь является абстрактным представлением отношений между объектами разных сущностей. Благодаря связям можно находить связанные экземпляры других сущностей в базе данных. Связи обладают следующими свойствами:

- арность - количество участвующих в связи сущностей (унарные, бинарные и т.д.);

- кратность - максимальное число экземпляров одной сущности, которые могут быть связаны с экземпляром другой сущности. Выделяют связи «один-к-одному» (1:1), «один-ко-многим» (1:М) и «многие-ко-многим» (М:М).

При разработке диаграммы «сущность-связь» (ER-диаграммы) необходимо на основе требований к системе определить:

- перечень сущностей предметной области;

- атрибуты для каждой сущности;

- связи между сущностями и их свойства.

ER-диаграмма для проектируемой системы «Управления клиентскими заказами» изображена на рисунке 2.

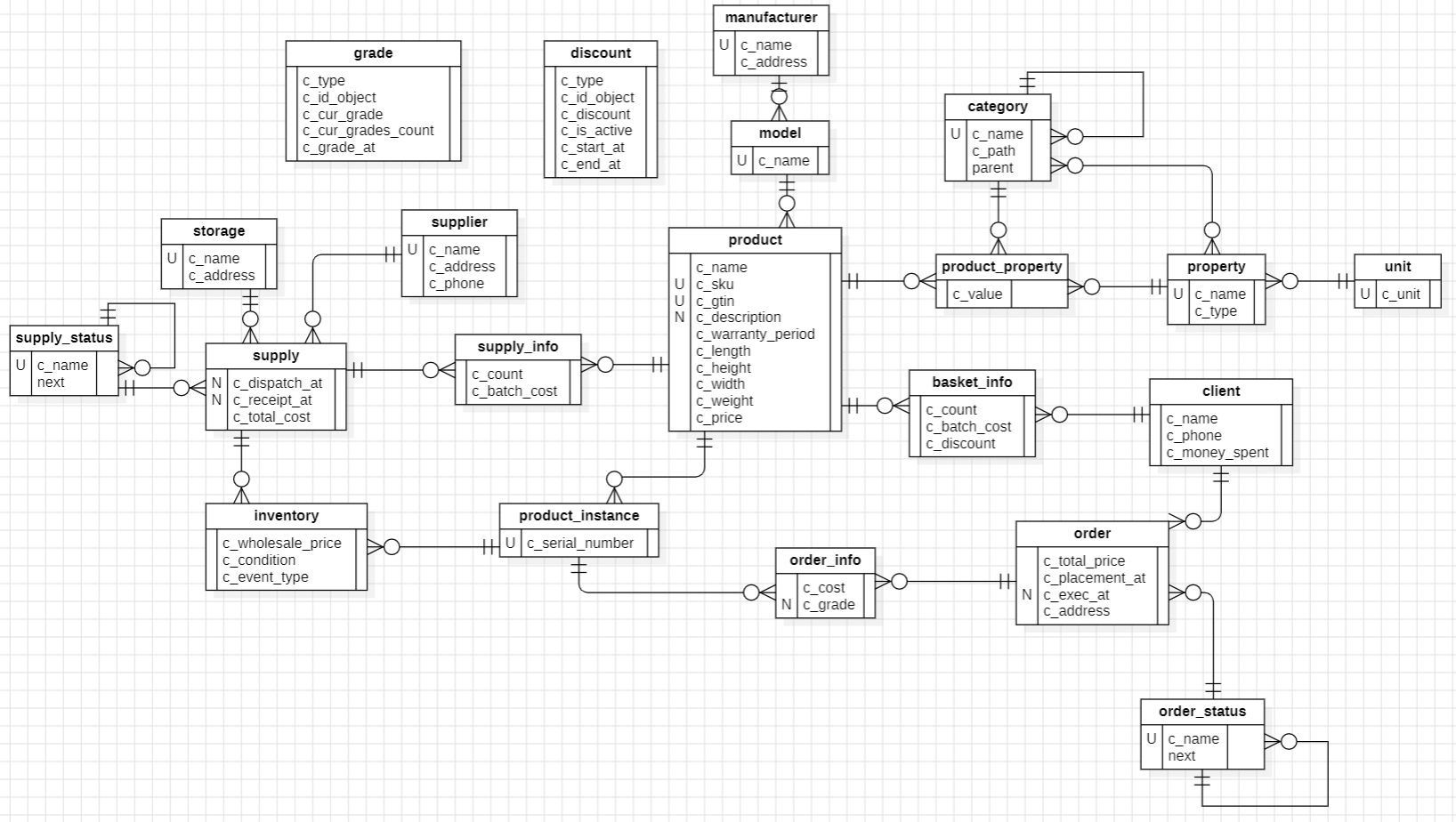


Рисунок 2 – ER-диаграмма

## 2.3 Стадия технического проекта

### 2.3.1 Получение исходной схемы реляционной базы данных

Следующим этапом проектирования базы данных является разработка схемы реляционной базы данных, которая происходит путем преобразования ER-модели при помощи специальных правил. Реляционная база данных – множество связанных между собой таблиц.

Правила преобразования ER-модели в схему реляционной базы данных:

1. Для каждой независимой сущности в ER-модели создается отдельная таблица. Простые атрибуты сущности становятся столбцами таблицы, а ключевой атрибут становится первичным ключом таблицы.

2. Если между двумя сущностями есть связь «многие-ко-многим», создается дополнительная таблица, содержащая только первичные ключи двух сущностей в качестве составного первичного ключа.

3. Для связи «один-к-одному» между двумя сущностями в таблицу одной из сущностей добавляется внешний ключ, ссылающийся на первичный ключ таблицы другой сущности.

4. Для связи «один-ко-многим» в таблицу сущности на стороне «многие» добавляется внешний ключ, ссылающийся на первичный ключ таблицы сущности на стороне «один».

5. Если связь включает более двух сущностей, создается отдельная таблица, содержащая первичные ключи всех участвующих сущностей в качестве составного первичного ключа.

6. Если связь имеет собственные атрибуты, они добавляются как дополнительные столбцы в ту таблицу, которая содержит внешний ключ, представляющий данную связь.

Схема реляционной базы данных для проектируемой системы изображена на рисунке 3.

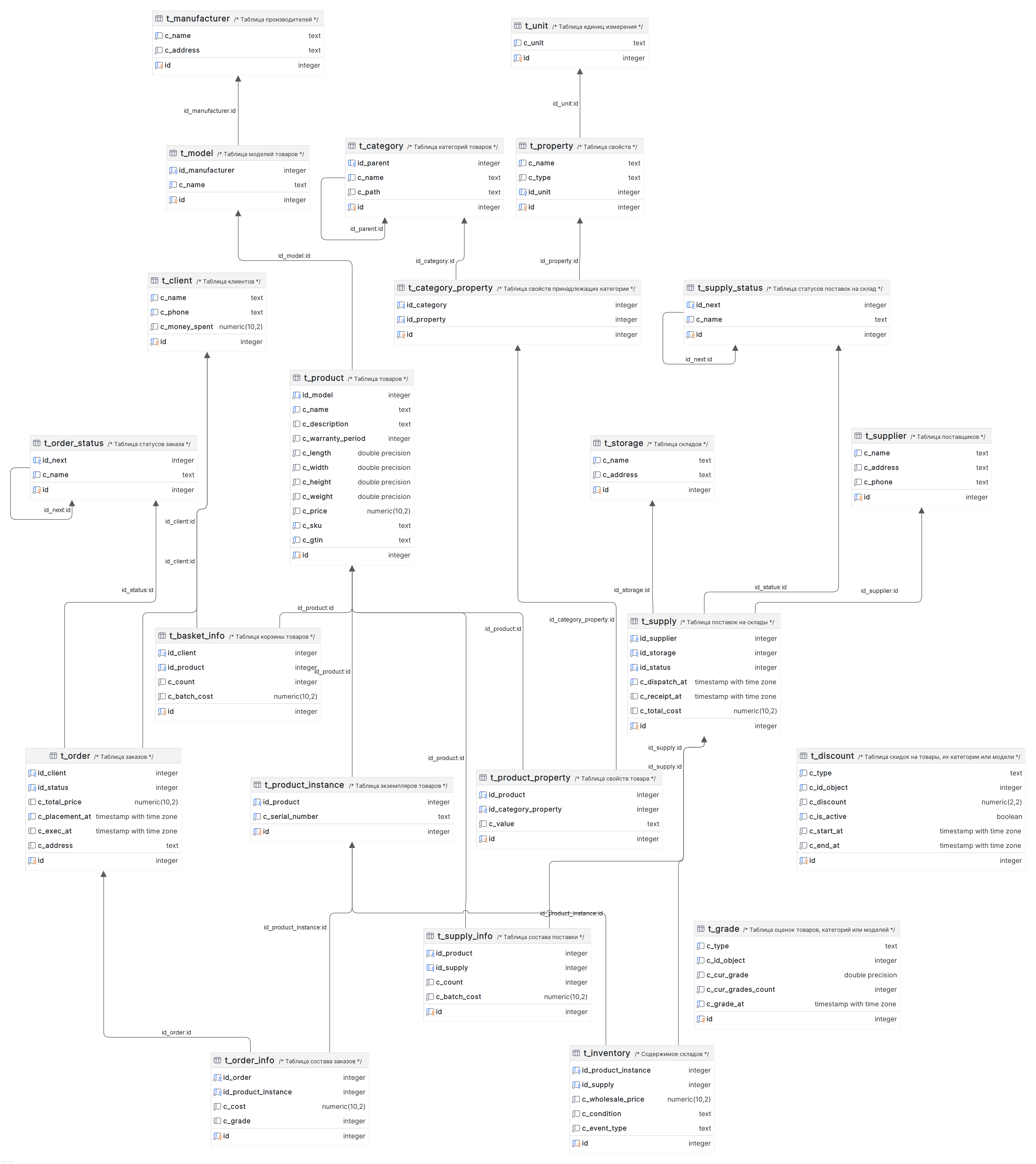


Рисунок 3 – Схема реляционной базы данных

### 2.3.2 Нормализация исходной схемы реляционной базы данных

Нормализация – процесс проектирования базы данных с использованием нормальных форм, подразумевающий в себя последовательное преобразование отношений из 1-й НФ в нормальную форму более высокого порядка по определённым правилам. Каждая следующая нормальная форма ограничивается определёнными типами функциональных зависимостей и устраняет соответствующие аномалии при выполнении операций над отношениями базы данных, сохраняя при этом свойства предыдущих нормальных форм.

Отношения могут находиться в одной из следующих нормальных форм (сокращенно НФ):

- первая нормальная форма (1-я НФ);

- вторая нормальная форма (2-я НФ);

- третья нормальная форма (3-я НФ);

- нормальная форма Бойса-Кодда (НФБК);

- четвертая нормальная форма (4-я НФ);

- пятая нормальная форма (5-я НФ);

- шестая нормальная форма (6-я НФ).

Соответствие отношения одной из сильных нормальных форм (3НФ и больше) дает определенные гарантии отсутствия аномалий.

Возьмем таблицу «t\_order» и проверим, находится ли она в третьей нормальной форме. Если не находится, то приведем её к 3НФ и докажем соответствие.

Исходя из описания первых трех нормальных форм можно сделать следующие выводы:

**Первая нормальна форма:** отношение находится в 1НФ, если все его атрибуты являются простыми, все используемые домены должны содержать только скалярные значения. Не должно быть повторений строк в таблице.

Таблица «t\_order» удовлетворяет требованиям 1НФ, так как все атрибуты являются простыми (атомарными) и не содержат повторяющихся групп данных (рис. 4):

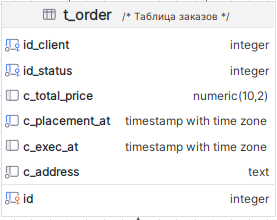


Рисунок 4 – Таблица «t\_order»

**Вторая нормальная форма:** отношение находится во 2НФ, если оно находится в 1НФ и каждый не ключевой атрибут неприводимо зависит от первичного ключа. Неприводимость означает, что в составе потенциального ключа отсутствует меньшее подмножество атрибутов, от которого можно также вывести данную функциональную зависимость.

Таблица находится в 2НФ, так как:

* первичный ключ состоит из одного поля — id (простая форма),
* все остальные атрибуты функционально зависят от id и не от его части.

**Третья нормальная форма:** таблица находится в третьей нормальной форме, если она находится в 2НФ и все не ключевые атрибуты являются независимыми друг от друга (нет транзитивных зависимостей).

В таблице нет транзитивных зависимостей между не ключевыми атрибутами:

* id\_client → c\_name, c\_phone и т.д. (зависимость вне таблицы),
* не ключевые атрибуты, такие как c\_total\_price и c\_address, напрямую зависят только от id.

Таким образом, таблица «t\_order» удовлетворяет требованиям 3НФ.

**Нормальная форма Бойса-Кодда:** таблица находится в данной нормальной форме, если она находится в 3НФ и каждая её нетривиальная и неприводимая слева функциональная зависимость имеет в качестве своего детерминанта некоторый потенциальный ключ.

В таблице присутствуют два детерминанта: суррогатный ключ id и уникальный ключ (id\_client, c\_placement\_at), от каждого из которых полностью функционально зависят остальные атрибуты отношения, соответственно они являются потенциальными ключами.

# 3 ОБОСНОВАНИЕ СРЕДСТВ РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ

В качестве СУБД выбран PostgreSQL. Для работы с компонентами PostreSQL была использована утилита DataGrip.

PostgreSQL является оптимальным выбором для этого проекта по нескольким факторам:

* Соответствие ACID и целостность данных: схема включает финансовые данные (цены, скидки, затраты), управление запасами и обработку заказов. Соответствие ACID (атомарность, согласованность, изоляция, долговечность) имеет решающее значение для обеспечения целостности данных и предотвращения потери или повреждения данных во время транзакций. PostgreSQL обеспечивает превосходное соответствие ACID, что делает его подходящим для управления конфиденциальными финансовыми данными.
* Расширенные типы данных: база данных использует различные типы данных (INTEGER, TEXT, NUMERIC, TIMESTAMP WITH TIME ZONE, BOOLEAN, DOUBLE PRECISION). PostgreSQL предлагает богатую поддержку типов данных, включая:
  + NUMERIC: для точного хранения денежных значений, избегая ошибок представления с плавающей точкой.
  + TIMESTAMPTZ: имеет решающее значение для обработки чувствительных ко времени данных (размещение заказа, выполнение, даты начала/окончания скидки) в разных часовых поясах.
  + BOOLEAN: для флагов типа c\_is\_active (статус скидки).
  + INTEGER, TEXT, DOUBLE PRECISION: для хранения данных общего назначения.
* Процедурный язык (PL/pgSQL): дизайн в значительной степени опирается на хранимые процедуры и функции (например, category\_set, discount\_set, get\_sales\_analytics). PL/pgSQL PostgreSQL — это мощный и надежный процедурный язык для создания сложной логики базы данных. Он предлагает:
  + Полную поддержку SQL: позволяет встраивать операторы SQL в код.
  + Управляющие структуры: предоставляет циклы, условные операторы и обработку ошибок.
  + Управление транзакциями: обеспечивает атомарное выполнение нескольких операций.
  + Функции безопасности: поддерживает функции прав определяющего, повышая безопасность.
* Расширенная индексация: PostgreSQL поддерживает различные методы индексации (B-дерево, GiST, GIN, BRIN), которые необходимы для оптимизации производительности запросов, особенно для:
  + Фильтрация продуктов по категориям: индексация id\_category в products.t\_model или связующей таблицы между продуктами и категориями.
  + Поиск продуктов: использование полнотекстовой индексации по c\_name и c\_description.
  + Анализ данных о продажах: индексация по столбцам даты и атрибутам группировки.
* Расширяемость: PostgreSQL обладает высокой расширяемостью, что позволяет добавлять пользовательские типы данных, функции и операторы для удовлетворения конкретных требований приложения.
* Управление параллелизмом: схема указывает на управление запасами и обработку заказов. Надежные механизмы управления параллелизмом PostgreSQL (MVCC — Multi-Version Concurrency Control) гарантируют, что несколько пользователей могут получать доступ к данным и изменять их одновременно без конфликтов. Это необходимо для поддержания согласованности данных в среде электронной коммерции с высоким трафиком.
* Безопасность: PostgreSQL предоставляет надежные функции безопасности, в том числе:
  + Аутентификация и авторизация: контроль того, кто может получить доступ к базе данных и что они могут делать.
  + Безопасность на уровне строк: контроль того, к каким строкам пользователь может получить доступ в таблице.
  + Аудит: отслеживание активности базы данных в целях безопасности и соответствия требованиям.
* Поддержка JSON и JSONB: схема не использует JSON напрямую, но если вам нужно хранить гибкие структуры данных (например, спецификации продуктов с различными атрибутами), типы данных JSON и JSONB PostgreSQL обеспечивают эффективные возможности хранения и запросов.
* Сообщество и поддержка: PostgreSQL имеет большое и активное сообщество разработчиков ПО с открытым исходным кодом, предоставляющее обширную документацию, учебные пособия и форумы поддержки. Также доступны варианты коммерческой поддержки.
* Экономическая эффективность: как СУБД с открытым исходным кодом, PostgreSQL устраняет затраты на лицензирование, что делает его привлекательным вариантом для проектов с бюджетными ограничениями.

# ЗАКЛЮЧЕНИЕ

В рамках выполнения курсового проекта были решены следующие задачи:

1. Проведён анализ предметной области магазина по продаже вычислительной техники.
2. Выполнено проектирование базы данных:

* разработана диаграмма вариантов использования;
* построена ER-модель и схема реляционной базы данных;
* произведена нормализация ключевых таблиц до третьей нормальной формы.

1. Выполнена программная реализация базы данных:

* создана структура таблиц в PostgreSQL;
* реализованы хранимые функции для поддержки бизнес-логики системы.

Разработанная база данных способна масштабироваться, поддерживает автоматизацию обработки заказов и может быть интегрирована в информационные системы розничных магазинов.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Волк, В. К. Базы данных. Часть 1. Проектирование и программирование [Текст]/ В. К. Волк. // Учебное пособие – Курган: Издательство Курганского государственного университета, 2018. – 181 с.
2. . Волк, В. К. Базы данных. Часть 2. Администрирование [Текст]/ В. К. Волк. // Учебное пособие – Курган: Издательство курганского государственного университета, 2018. – 128 с.
3. Руководство по PostgreSQL [Электронный ресурс]: [сайт] – URL: https://metanit.com/sql/postgresql/ (дата обращения 10.02.2025).
4. Описание основных приемов нормализации базы данных [Электронный ресурс]: [сайт] - URL: <https://learn.microsoft.com/ru-ru/office/troubleshoot/access/database-normalization-description>