# **1) Бизнес-процесс создания заказа (использовать любую удобную нотацию моделирования бизнес-процессов).**

Описание бизнес-процесса создания заказа.

Я использую нотацию BPMN (Business Process Model and Notation) для описания бизнес-процесса.

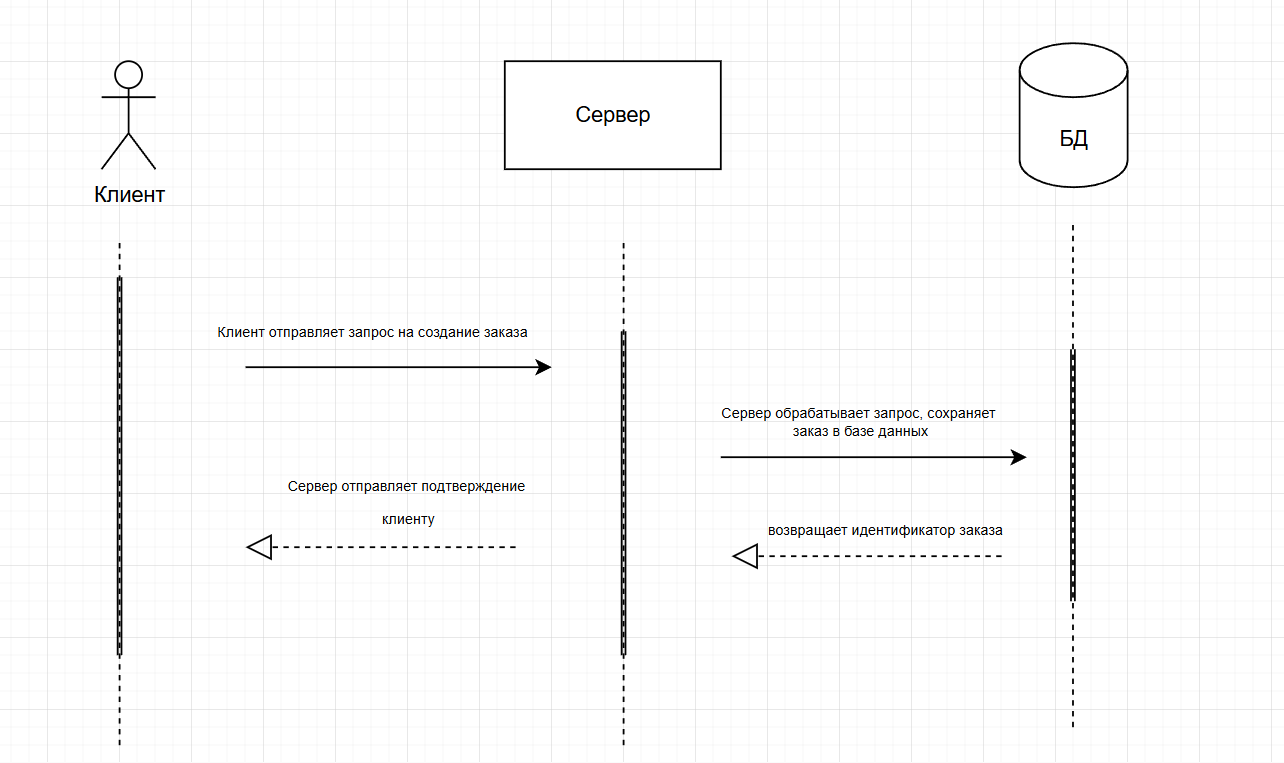
# 

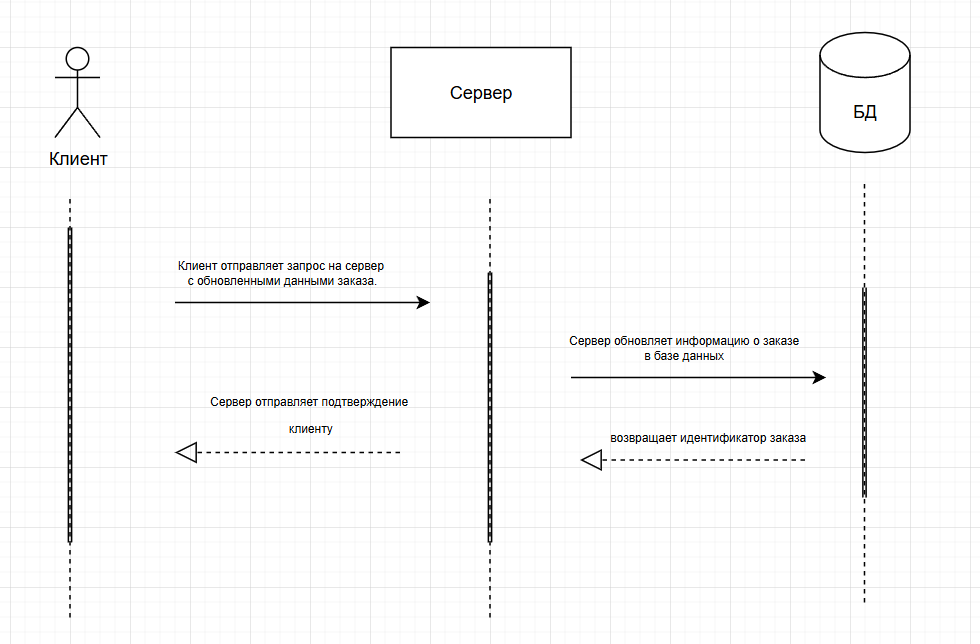
# **2) Опишите процесс синхронизации данных между клиентом и сервером**

# **(создание, редактирование и отмена заказа, изменение персональных данных, оплата заказа и т.д.). Представить все в диаграммах UML, API методах и других представлениях, также составить ER-диаграмму сущностей.**

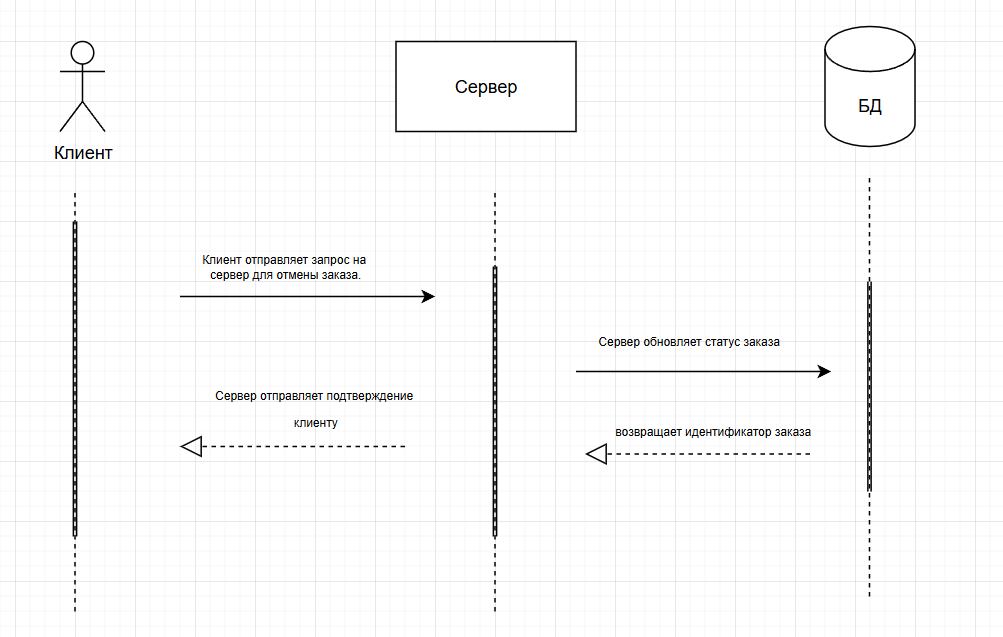
Процесс синхронизации данных между клиентом и сервером начинается с того, что клиент отправляет запрос на сервер для проверки актуальности данных. Сервер анализирует, какие данные изменились с последней синхронизации, и отправляет клиенту только обновленные записи. Клиент обновляет локальную базу данных и, при необходимости, решает конфликты, если изменения были сделаны как на сервере, так и на клиенте. После успешной синхронизации клиент подтверждает серверу успешную обработку данных, и процесс завершается.

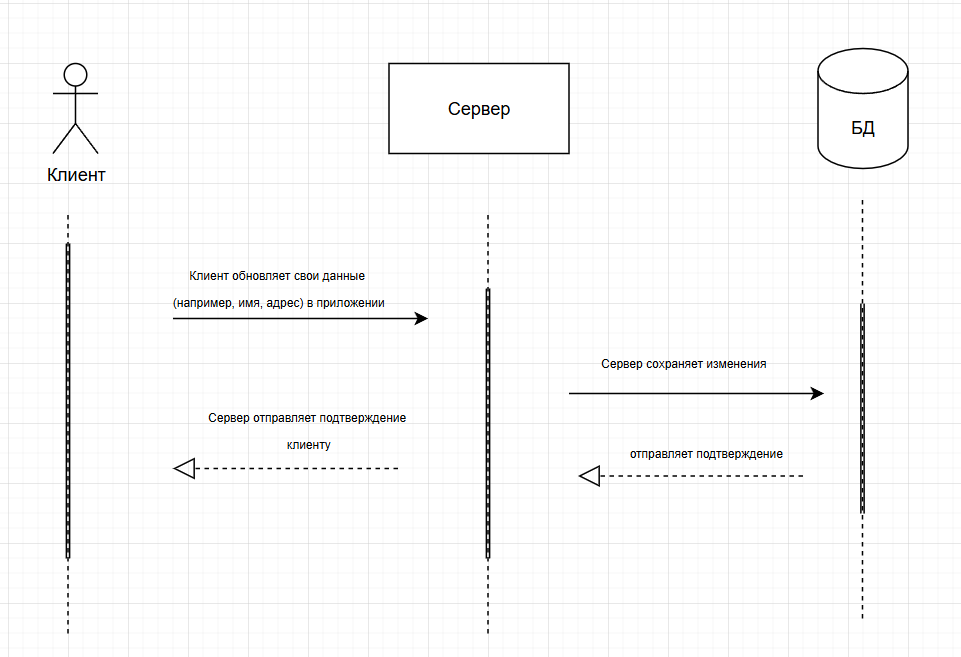
**UML диаграмма последовательности**

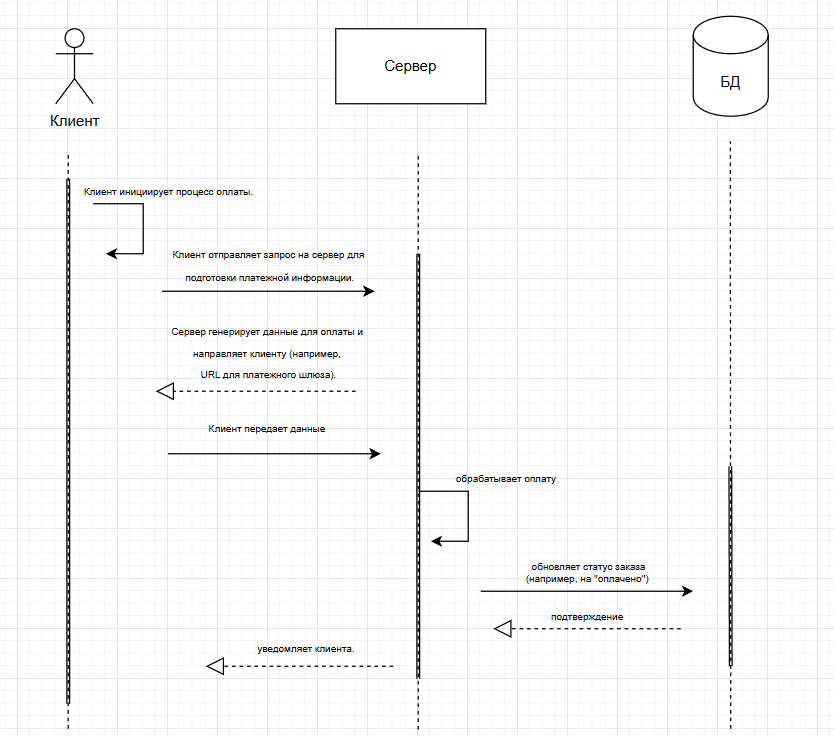
1. Создание заказа 
2. Редактирование заказа



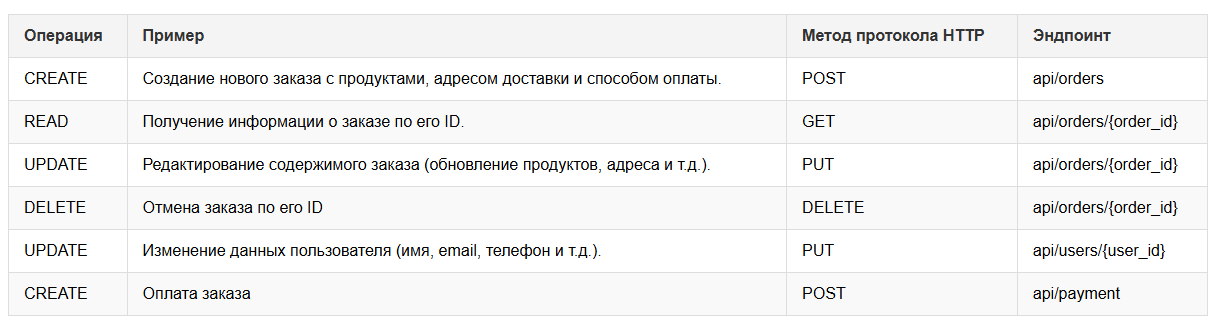
1. Отмена заказа



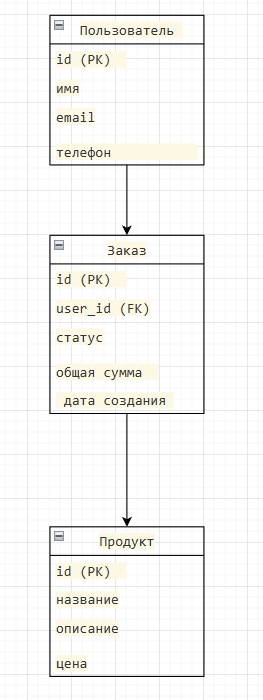
1. Изменение персональных данных
2. Оплата заказа



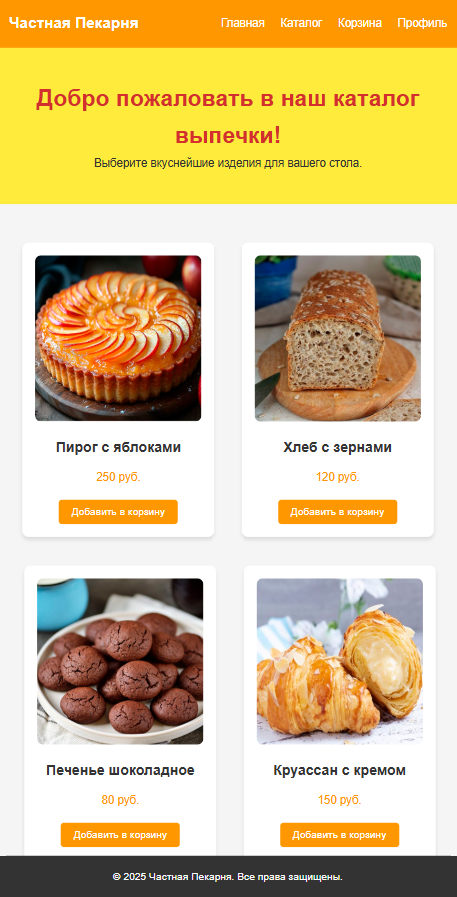
**API**



**ER-диаграмма**



# 3) Подготовить прототип одного из экранов данного мобильного приложения и описать пользовательский интерфейс для данного экрана (например, создание заказа)



Этот экран представляет собой **каталог товаров пекарни**, где пользователи могут ознакомиться с ассортиментом и выбрать выпечку для заказа. Интерфейс максимально прост и интуитивно понятен, что делает его удобным для пользователей. Все элементы дизайна нацелены на создание уютной и приятной атмосферы, соответствующей теме пекарни.

**1. Навигация (Header)**

В верхней части экрана находится **навигационная панель**. В ней расположены:

* **Логотип пекарни** — в левой части панели отображается логотип, который представляет собой название пекарни "Частная Пекарня".
* **Меню навигации** — справа от логотипа находятся ссылки:
  + **Главная** — переход на главную страницу сайта.
  + **Каталог** — текущая страница, отображающая список товаров.
  + **Корзина** — переход в корзину с возможностью просмотра и оформления выбранных товаров.
  + **Профиль** — ссылка на страницу пользователя с его данными и историей заказов.
* Все ссылки в навигационном меню выделяются оранжевым цветом при наведении, что помогает пользователю легко ориентироваться на странице.

**2. Приветственная секция (Hero Section)**

Прямо под навигационной панелью находится “**герой-секция”** с текстом:

* **Заголовок**: "Добро пожаловать в наш каталог выпечки!" — это приветственное сообщение, создающее теплую атмосферу.
* **Подзаголовок**: "Выберите вкуснейшие изделия для вашего стола." — короткий мотивирующий текст, который побуждает пользователя выбрать товары из каталога.

Этот блок выполнен в ярких, теплых цветах с желтым фоном, что ассоциируется с выпечкой.

**3. Каталог товаров**

Основная часть экрана — это **каталог с товарами**, каждый из которых представлен в виде карточки. Вот что включает в себя каждая карточка товара:

* **Изображение товара** — высококачественные изображения каждого продукта, чтобы пользователь мог оценить внешний вид выпечки. Это изображение имеет закругленные углы для более мягкого и привлекательного вида.
* **Название товара** — краткое название, например "Пирог с яблоками" или "Печенье шоколадное".
* **Цена товара** — отображается цена рядом с названием. Цена выделена **оранжевым цветом** для того, чтобы она привлекала внимание пользователя.
* **Кнопка "Добавить в корзину"** — по нажатию на эту кнопку товар добавляется в корзину для дальнейшего оформления. Кнопка имеет яркий оранжевый цвет, чтобы быть заметной и побуждать к действию.

Карточки товаров имеют **тени** и **округленные углы**, что делает их визуально легкими и привлекательными. Кроме того, при наведении на карточку кнопка "Добавить в корзину" изменяет цвет, что дает пользователю визуальную обратную связь о возможности взаимодействия с элементом.

**4. Футер (Footer)**

В нижней части экрана расположен **футер**, который содержит:

* **Информацию о праве собственности**: “© 2025 Частная Пекарня. Все права защищены.”.

Футер имеет темный фон и белый текст, что создает контраст с остальной частью страницы и четко выделяет эту часть интерфейса.

**5. Цветовая палитра и шрифты**

* **Основные цвета**: яркий **оранжевый** и **желтый**.
* **Дополнительные цвета**: **белый** (для фона и карточек товаров) и **темно-серый/черный** (для текста и футера) — эти нейтральные цвета помогают создать баланс и делают контент легким для восприятия.
* **Шрифты**: используется стандартный шрифт **Arial**, который легко читаем и приятен на глаз.

# 4) Подготовить подробное описание функции редактирования заказа , которую можно было бы использовать в качестве постановки задачи для разработки (помимо текстового описания, использовать UML диаграммы, указать используемые API методы, передаваемые и получаемые параметры, описать процесс хранения информации о покупках пользователя).

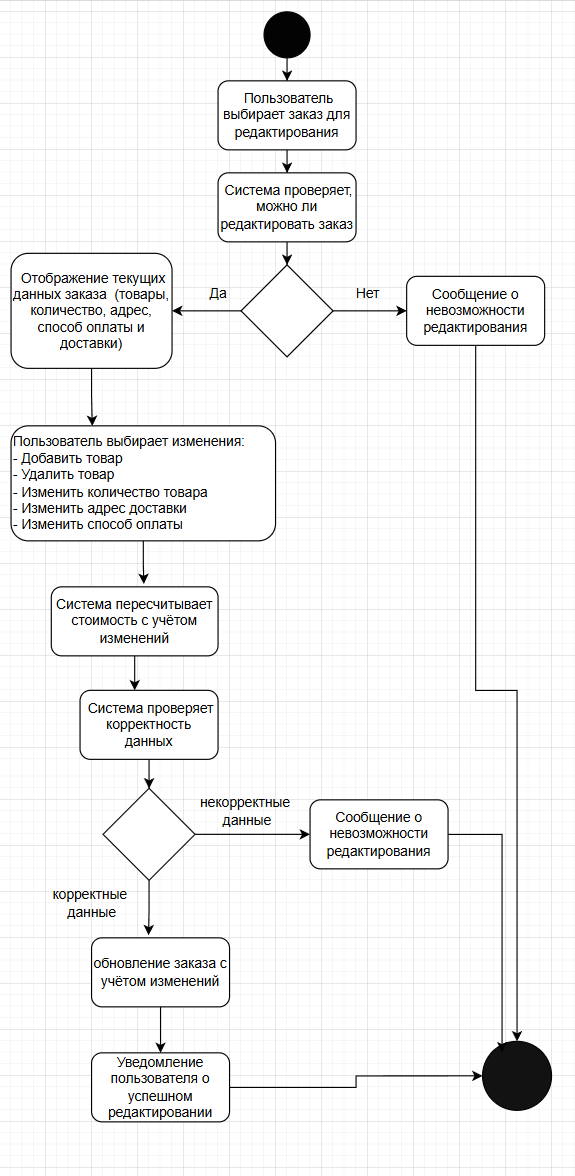
**1. Описание задачи**

Функция **редактирования заказа** позволяет пользователю изменять уже оформленный заказ, например, добавлять или удалять товары, менять количество товаров, а также изменять адрес доставки или способ оплаты. Это важная функция для улучшения пользовательского опыта, особенно в случае ошибок или изменений в планах пользователя.

**2. Функциональные требования**

* Пользователь должен иметь возможность **добавить или удалить товары** из уже оформленного заказа.
* Пользователь должен иметь возможность **изменить количество** каждого товара в заказе.
* Пользователь должен иметь возможность **изменить способ доставки** (например, изменить адрес или выбрать самовывоз вместо курьерской доставки).
* Пользователь должен иметь возможность **изменить способ оплаты** (например, изменить метод оплаты с карты на оплату при получении).
* После изменения заказа, система должна **пересчитать общую стоимость** с учетом всех изменений.
* Система должна **сохранить изменения** и обновить данные на сервере.

**UML**



**API Методы**

Метод 1: Получение информации о текущем заказе

**Метод**: GET

Метод 2: Обновление информации о заказе

**Метод**: PUT

**Передаваемые параметры:**

1. **Идентификатор заказа**

* Уникальный номер или код, который позволяет однозначно определить, какой заказ нужно изменить.

1. **Данные о заказе**

* Это информация о самом заказе, которую пользователь хочет изменить. Например:

1. Список выбранных товаров и их количество.
2. Новый адрес доставки, если пользователь решил изменить его.
3. Примечания или комментарии к заказу (например, «добавить меньше сахара»).
4. **Идентификатор пользователя**

* Номер или код, связанный с пользователем, чтобы убедиться, что заказ принадлежит именно этому человеку.

1. **Метод оплаты**

* Выбранный способ оплаты, если пользователь хочет изменить, например, с оплаты картой на оплату наличными.

**Получаемые параметры:**

1. **Статус выполнения операции**

* Сообщение о том, успешно ли выполнено редактирование (например, "успешно", "ошибка").

1. **Сообщение об ошибке**

* Если редактирование не удалось, то система возвращает причину (например, "заказ уже передан на обработку и не может быть изменен").

1. **Обновленный заказ**

* Информация о заказе после редактирования, чтобы пользователь мог проверить, что изменения были учтены.

1. **Общая стоимость**

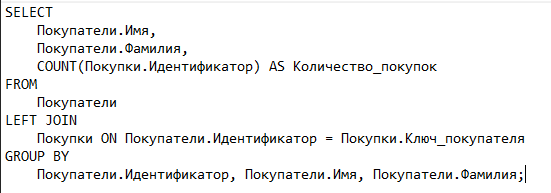
* Пересчитанная сумма заказа, если изменения затронули цену (например, добавлены новые товары или изменилось количество).

**Хранение данных о покупках пользователя**

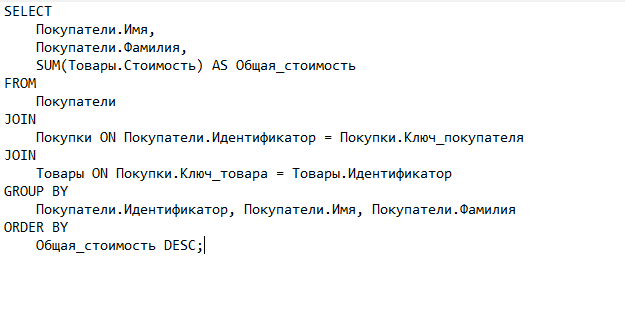
Информация о покупках пользователя должна храниться в базе данных, которая будет содержать таблицы для заказов, товаров, пользователей и их истории покупок.

# 5) Перед вами реляционная модель данных. Необходимо написать SQL-запросы.

1. Вывести покупателей с количеством осуществленных покупок:



1. **Вычислить общую стоимость товаров для каждого покупателя и отсортировать результат в порядке убывания:**



1. Получить покупателей, купивших только один товар:

