1. Для проектирования базы данных был составлен список типичных бизнес-задач для магазина:

* **Управление ассортиментом**: Выбор и закупка товаров, управление запасами и оптимизация складского хранения.
* **Ценообразование**: Установление конкурентоспособных и прибыльных цен, проведение акций и скидок.
* **Продвижение и маркетинг**: Реклама, PR, социальные медиа, email-маркетинг, SEO и контекстная реклама для привлечения новых клиентов.
* **Оптимизация продаж**: Тренировка персонала, разработка программы лояльности, анализ продаж и поведения покупателей.

2) список пользователей проекта:

1. Администратор
2. Менеджер
3. Пользователь
4. Оценить сроки проекта и требуемые ресурсы

* **Анализ и планирование:**
  1. Сбор требований: 1 неделя
  2. Проектирование архитектуры: 1 неделя
  3. Планирование: 2 неделя
* **Разработка:**
  1. Разработка базы данных: 2 недели
  2. Разработка бэкенда: 4 недели
  3. Разработка фронтенда (интерфейс пользователя): 2 недели
* **Тестирование и QA:**
  1. Функциональное тестирование: 1 неделя
  2. Тестирование производительности и безопасности: 1 неделя
  3. Исправление ошибок: 1 неделя
* **Запуск и поддержка:**
  1. Настройка и поддержка: 1 неделя
* **Мониторинг и поддержка: постоянно после запуска**

Общий прогнозируемый срок: примерно 4 месяцев.

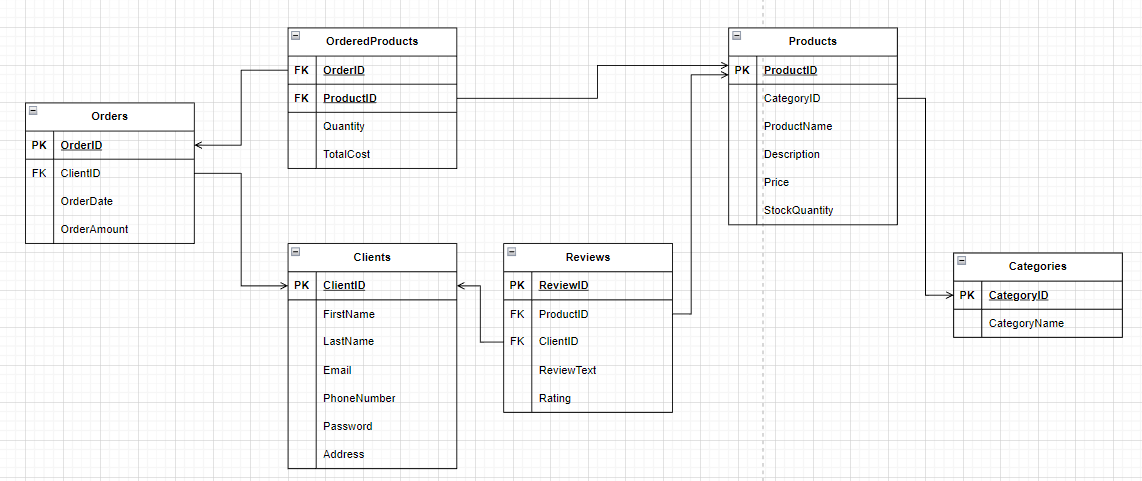
**Требуемые ресурсы:**

* Команда разработчиков: 2-3 разработчика (1 бэкенд, 1-2 фронтенд)
* Тестировщик/QA инженер: 1
* DBA/Разработчик баз данных: 1
* Менеджер проекта: 1
* Системный администратор/DevOps инженер: 1 для настройки серверов

**Инфраструктура:**

* Сервера (либо физические, либо облачные, например AWS, Azure)
* Системы управления версиями (например, Git)
* Системы отслеживания ошибок и задач (например, Jira)
* Среды для непрерывной интеграции и доставки (CI/CD)

**Логическая схема**



Нормальные формы:

* 1NF: Все атрибуты имеют простые (атомарные) значения, нет повторяющихся групп или атрибутов.
* 2NF: Каждый атрибут зависит от первичного ключа. Например, в таблице OrderedProducts, Quantity и TotalCost зависят от сочетания OrderID и ProductID.
* 3NF: Все атрибуты функционально зависят только от первичного ключа. Например, в таблице Products, Description, Price, и StockQuantity зависят только от ProductID, а не от других атрибутов.
* 4NF: Отсутствуют многозначные зависимости. Например, каждый заказанный продукт связан только с одним заказом и одним продуктом, и нет необходимости в разделении на дополнительные таблицы.

Денормализованные атрибуты:

Атрибут OrderAmount в таблице Orders может рассматриваться как денормализованный, так как этот атрибут может быть вычислен путем суммирования TotalCost из таблицы OrderedProducts для каждого заказа. Однако его включение ускоряет доступ к общей сумме заказа без необходимости выполнять дополнительные вычисления.

Обеспечение безопасности:

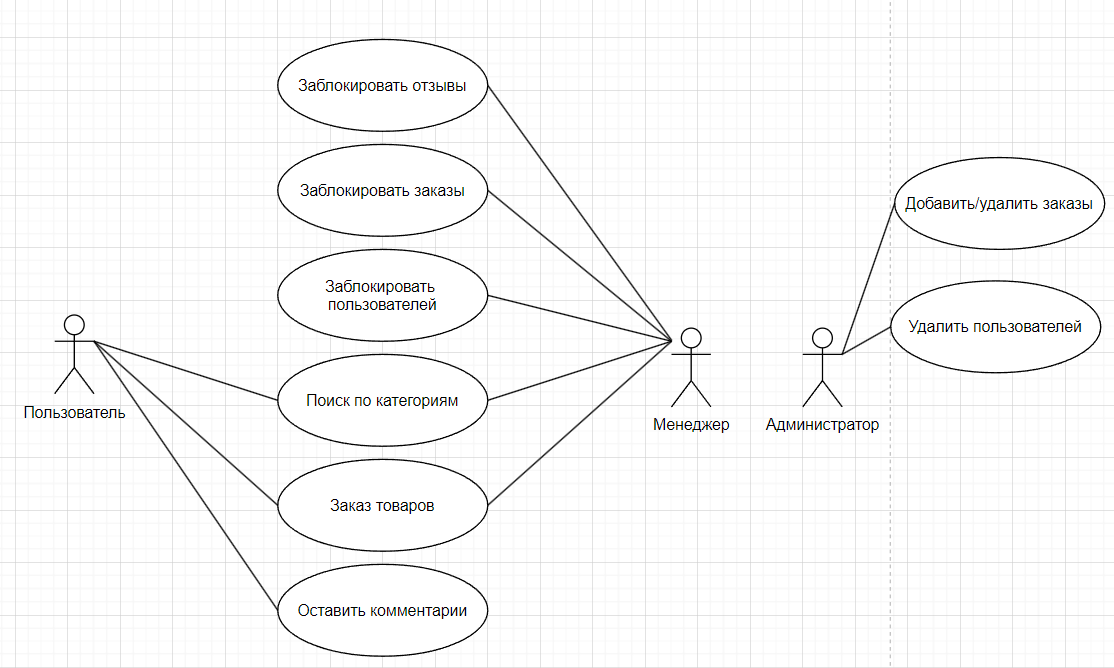
Атрибут Password в таблице Clients хранится в зашифрованном виде, что обеспечивает безопасность учетных данных клиентов.

Используются ограничения целостности данных, такие как FOREIGN KEY, чтобы гарантировать корректные отношения между таблицами.

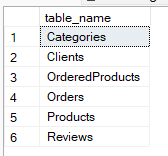
Ограничения на примере CHECK в таблице Reviews гарантирует, что рейтинг находится в допустимом диапазоне от 1 до 5.

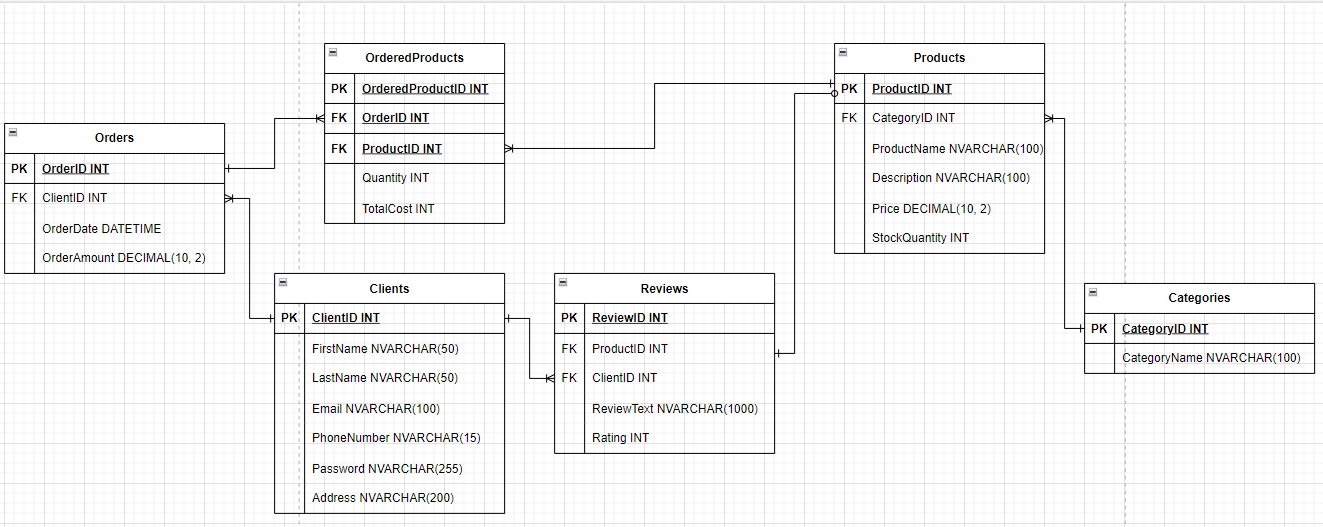
Атрибут Email в таблице Clients уникален, что предотвращает создание дубликатов и обеспечивает уникальное средство идентификации клиента.

**UML-Диаграмма**



**Физическая схема**





Скрипт для MsSQL:

|  |
| --- |
| use shop;  -- Table "Clients"  CREATE TABLE Clients (  ClientID INT PRIMARY KEY IDENTITY(1,1),  FirstName NVARCHAR(50) NOT NULL,  LastName NVARCHAR(50) NOT NULL,  Email NVARCHAR(100) UNIQUE NOT NULL,  PhoneNumber NVARCHAR(15),  Password NVARCHAR(255) NOT NULL, --encrypted  Address NVARCHAR(200)  );  -- Table "Categories"  CREATE TABLE Categories (  CategoryID INT PRIMARY KEY IDENTITY(1,1),  CategoryName NVARCHAR(100) NOT NULL  );  -- Table "Products"  CREATE TABLE Products (  ProductID INT PRIMARY KEY IDENTITY(1,1),  CategoryID INT FOREIGN KEY REFERENCES Categories(CategoryID),  ProductName NVARCHAR(100) NOT NULL,  Description NVARCHAR(1000),  Price DECIMAL(10, 2) NOT NULL,  StockQuantity INT NOT NULL  );  -- Table "Orders"  CREATE TABLE Orders (  OrderID INT PRIMARY KEY IDENTITY(1,1),  ClientID INT FOREIGN KEY REFERENCES Clients(ClientID),  OrderDate DATETIME NOT NULL,  OrderAmount DECIMAL(10, 2) --denormalized attribute  );  -- Table "OrderedProducts"  CREATE TABLE OrderedProducts (  OrderID INT FOREIGN KEY REFERENCES Orders(OrderID),  ProductID INT FOREIGN KEY REFERENCES Products(ProductID),  Quantity INT NOT NULL,  TotalCost DECIMAL(10, 2) NOT NULL,  PRIMARY KEY (OrderID, ProductID)  );  -- Table "Reviews"  CREATE TABLE Reviews (  ReviewID INT PRIMARY KEY IDENTITY(1,1),  ProductID INT FOREIGN KEY REFERENCES Products(ProductID),  ClientID INT FOREIGN KEY REFERENCES Clients(ClientID),  ReviewText NVARCHAR(1000) NOT NULL,  Rating INT CHECK (Rating BETWEEN 1 AND 5)  );  SELECT name AS table\_name  FROM sys.tables  ORDER BY name; |

Скрипт для Oracle:

|  |
| --- |
| -- Sequence for Clients  CREATE SEQUENCE seq\_clients START WITH 1;  -- Table "Clients"  CREATE TABLE Clients (  ClientID NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,  FirstName NVARCHAR2(50) NOT NULL,  LastName NVARCHAR2(50) NOT NULL,  Email NVARCHAR2(100) UNIQUE NOT NULL,  PhoneNumber NVARCHAR2(15),  Password NVARCHAR2(255) NOT NULL, --encrypted  Address NVARCHAR2(200)  );  -- Sequence for Categories  CREATE SEQUENCE seq\_categories START WITH 1;  -- Table "Categories"  CREATE TABLE Categories (  CategoryID NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,  CategoryName NVARCHAR2(100) NOT NULL  );  -- Sequence for Products  CREATE SEQUENCE seq\_products START WITH 1;  -- Table "Products"  CREATE TABLE Products (  ProductID NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,  CategoryID NUMBER REFERENCES Categories(CategoryID),  ProductName NVARCHAR2(100) NOT NULL,  Description NVARCHAR2(1000),  Price DECIMAL(10, 2) NOT NULL,  StockQuantity NUMBER NOT NULL  );  -- Sequence for Orders  CREATE SEQUENCE seq\_orders START WITH 1;  -- Table "Orders"  CREATE TABLE Orders (  OrderID NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,  ClientID NUMBER REFERENCES Clients(ClientID),  OrderDate DATE NOT NULL,  OrderAmount DECIMAL(10, 2)  );  -- Table "OrderedProducts"  CREATE TABLE OrderedProducts (  OrderID NUMBER REFERENCES Orders(OrderID),  ProductID NUMBER REFERENCES Products(ProductID),  Quantity NUMBER NOT NULL,  TotalCost DECIMAL(10, 2) NOT NULL,  PRIMARY KEY (OrderID, ProductID)  );  -- Sequence for Reviews  CREATE SEQUENCE seq\_reviews START WITH 1;  -- Table "Reviews"  CREATE TABLE Reviews (  ReviewID NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,  ProductID NUMBER REFERENCES Products(ProductID),  ClientID NUMBER REFERENCES Clients(ClientID),  ReviewText NVARCHAR2(1000) NOT NULL,  Rating NUMBER CHECK (Rating BETWEEN 1 AND 5)  );  SELECT table\_name  FROM user\_tables  ORDER BY table\_name; |