

1. Базы данных - тест

1. Содержит ли какую-то информацию таблица, в которой нет полей?

3 - Таблица без полей существовать не может

2. В записи файла реляционной БД может содержаться:

4 - Неоднородная информация (данные разных типов)

3. Чем первичный ключ отличается от внешнего ключа?

2 - Значения первичного ключа всегда должны быть уникальными и не могут быть null, значения внешнего ключа могут повторяться

4 - Первичный ключ является идентификатором для строки, а внешний ключ используется для связывания таблиц

4. В какой нормальной форме говорится о том, что все атрибуты зависят от первичного ключа, а не от его части?

2 - 2НФ

5. В каком порядке в СУБД выполняются операторы *SELECT*, *FROM*, *GROUP BY*?

4 - Сначала FROM, потом GROUP BY и только потом SELECT

6. Чем отличается оператор *WHERE* от *HAVING*

2 - Оператор HAVING применяется для фильтрации групп, а WHERE - для фильтрации отдельных строк

7. Какой результат покажет выполнение операторов *SELECT COUNT (*)*?

3 - Число строк таблицы, указанной во FROM, включая значение NULL

8. В таблице «Animals» базы данных зоопарка содержится информация обо всех обитающих там животных, в том числе о лисах: red fox, grey fox, little fox. Напишите запрос, возвращающий информацию о возрасте лис

1 - SELECT age

FROM Animals

WHERE Animal LIKE “%fox”

9. Чем отличается DELETE от TRUNCATE?

2 - DELETE используется для удаления одной или нескольких строк из таблицы, а TRUNCATE используется для удаления всех строк из таблицы.

3 - DELETE может использовать условие WHERE, а TRUNCATE всегда удаляет все записи из таблицы

10. Дана таблица:

COLOR
BLUE
RED
null
RED

Каким будет результат запроса?

SELECT COUNT (DISTINCT color) FROM Table

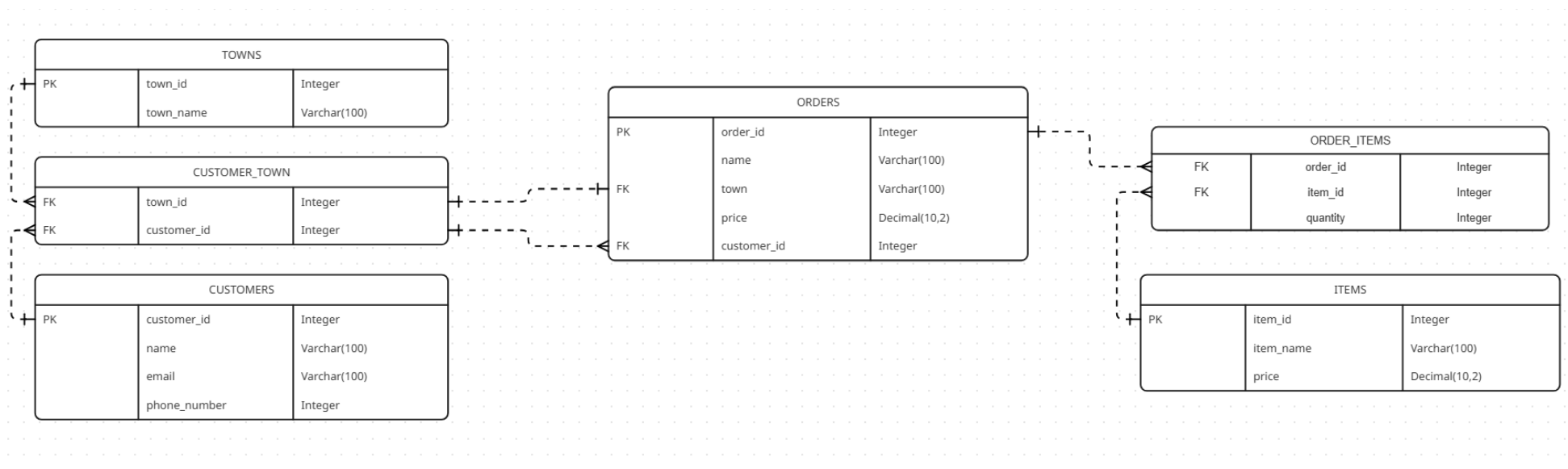
4 – 2

2. Базы данных - ER

В базе данных есть таблица заказов - orders. В ней есть поля: id (идентификатор заказа), name (название заказа), town (адрес доставки заказа), price (цена заказа), customer_id (идентификатор покупателя). Также есть таблицы: towns (справочник адресов), items (товары), customers (покупатели). Известно, что между orders и items предполагается связь многие-ко-многим.

Что нужно сделать:

Спроектируйте ER-диаграмму с учетом этих вводных. Состав полей таблиц укажите на свое усмотрение с учетом условия, но для таблицы orders учтите те поля, что указаны.



3. Интеграции

Представим, что Вы работаете аналитиком и проектируете работу приложения интернет-магазина. Вам нужно заложить и спроектировать следующий сценарий: отображение витрины товаров (список товаров с кратким описанием), переход с витрины на экран с детальным описанием конкретного товара, добавление товара в корзину.

Что нужно сделать:

1. Спроектируйте REST API, которые нужны для реализации описанного сценария. Решение должно включать описание запросов и описание или пример ответа для каждого из запросов в формате JSON. Способ описания - на ваше усмотрение. По составу полей товара можете ориентироваться на любой известный интернет-магазин.
- GET /api/products

```
{
  "products": [
    {
      "product_id": 1,
      "name": "Ноутбук ASUS",
      "shortDescription": "Мощный ноутбук для работы и игр",
      "price": 75000,
      "image": "https://shop.com/images/laptop1.jpg"
    },
    {
      "product_id": 2,
      "name": "Смартфон Samsung Galaxy",
      "shortDescription": "Флагманский смартфон с отличной камерой",
      "price": 50000,
      "image": "https://.com/images/phone1.jpg"
    },
    {
      "product_id": 3,
      "name": "Наушники Sony WH-1000XM4",
      "shortDescription": "Премиальные беспроводные наушники с шумоподавлением",
      "price": 25000,
      "image": "https://shop.com/images/headphones1.jpg"
    }
  ]
}
```

- GET /api/products/1

```
{
  "product_id ": 1,
  "name": "Ноутбук ASUS",
  "fullDescription": "Ноутбук ASUS ROG Strix G15 — это мощная игровая машина
с процессором Intel Core i7, 16 ГБ оперативной памяти и видеокартой NVIDIA
RTX 3060. Идеален для игр, работы с графикой и монтажа видео.",
  "price": 75000,
  "oldPrice": 85000,
  "brand": "ASUS",
  "inStock": 15,
  "category": "Ноутбуки",
  "specifications": {
    "processor": "Intel Core i7-11800H",
    "ram": "16 GB DDR4",
    "storage": "512 GB SSD",
    "display": "15.6\ Full HD 144Hz",
    "graphics": "NVIDIA RTX 3060"
  },
  "images": [
    "https://shop.com/images/laptop1_main.jpg",
    "https://shop.com/images/laptop1_side.jpg",
    "https://shop.com/images/laptop1_back.jpg"
  ]
}
```

- POST /api/cart/add

Тело запроса:

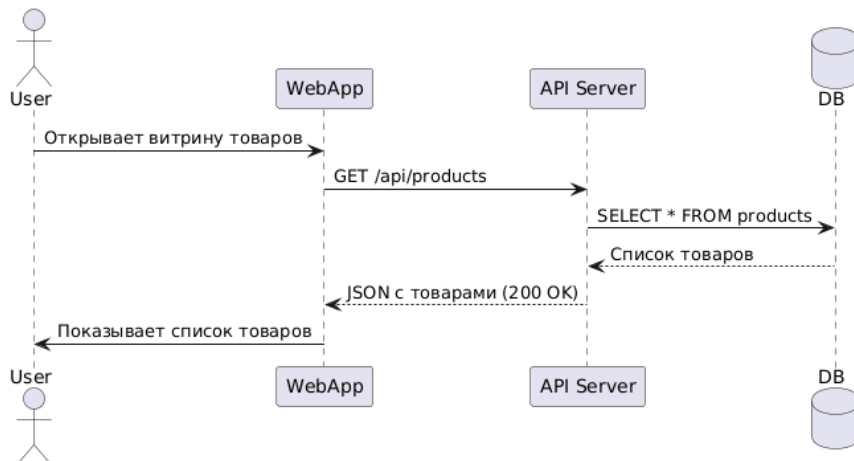
```
{
  "user_id": 12345,
  "product_id": 1,
  "quantity": 1
}
```

Тело ответа:

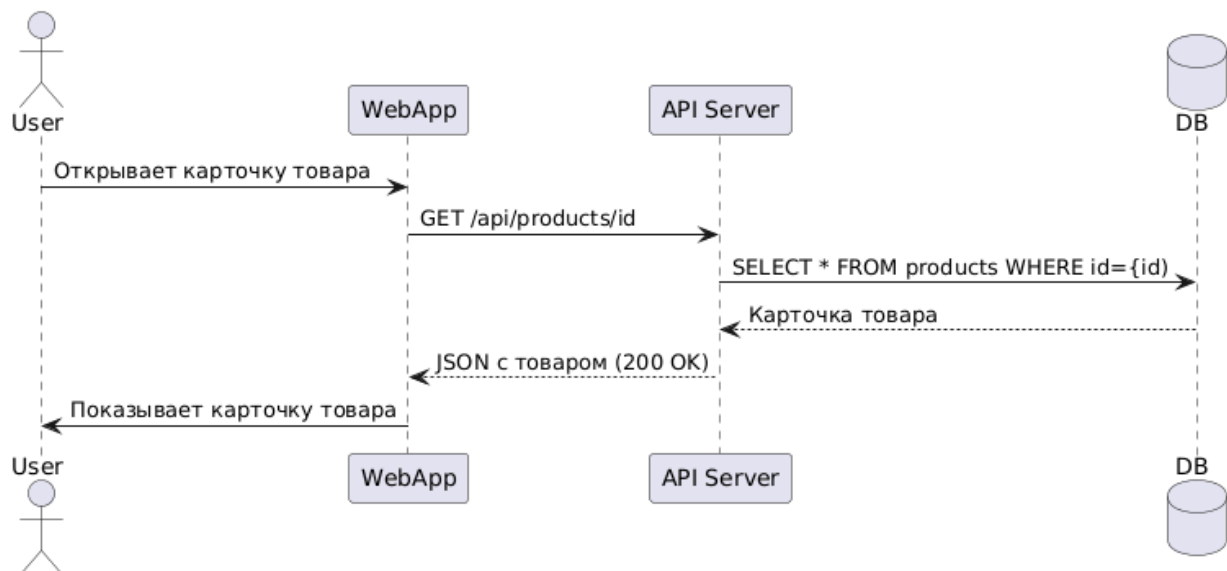
```
{
  "status": "success",
  "message": "Товар добавлен в корзину"
}
```

2. Постройте Sequence UML диаграмму для этого сценария.

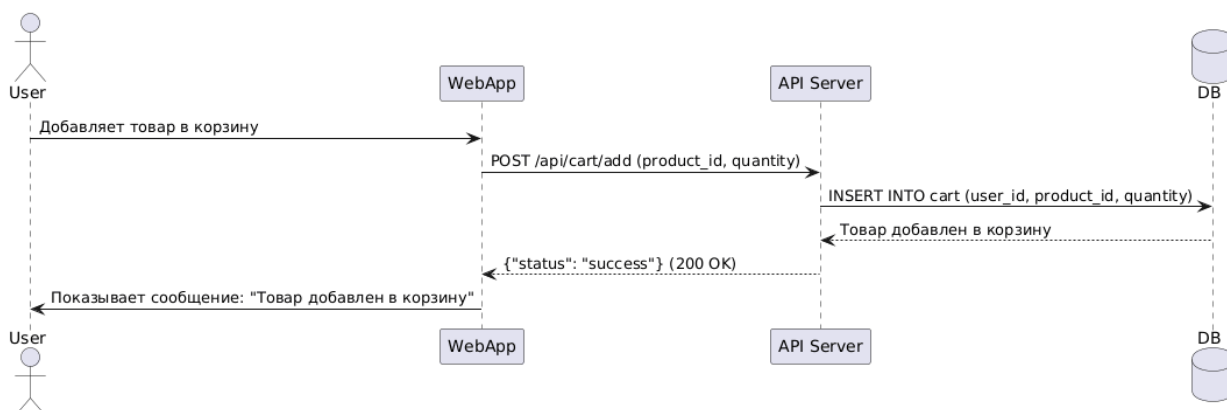
- GET /api/products



- GET /api/products/1



- POST /api/cart/add



4. Алгоритмическое мышление

Возьмем в качестве примера банковское мобильное приложение.

Исходные условия: у вас в руке смартфон, на котором установлено банковское приложение (телефон выключен).

Что нужно сделать:

Используя любую нотацию, опишите в виде диаграммы процесс, в рамках которого Вы, используя банковское приложение, пополните баланс своего телефона на 100Р. В качестве примера можете использовать любое мобильное банковское приложение, что у Вас есть.

В идеальном случае, если есть деньги, интернет и знаем пароль, иначе пришлось бы добавлять много условий и таблица была бы громоздкой:

