# E-commerce — Выявление профилей потребления

**Описание проекта:** Интернет-магазин товаров для дома «Пока все ещё тут» хочет проанализировать профили клиентов и товарный ассортимент.

**Цель проекта:** Узнать больше информации о пользователях, чтобы улучшить показатели интернет-магазина «Пока все ещё тут».

**Задача проекта:** Сегментировать покупателей по профилю потребления:

- Провести исследовательский анализ данных;
- Сегментировать покупателей на основе истории их покупок;
- Сформулировать и проверьте статистические гипотезы.

**Декомпозиция задачи:**

1. Загрузка данных и изучение общей информации:

- Изучить общую информацию о датасете;
- Сколько покупателей и заказов?;
- За какой период предоставлены данные?;
- Что можно сказать о каждом столбце?;
- Значения какого типа они хранят?

1. Предобработка данных:

- Проверить дубликаты в данных;
- Поискать пропуски: встречаются ли они, в каких столбцах? Можно ли их обработать или оставить как есть?;
- Проверить данные на соответствие бизнес-правилам "1 заказ- 1 клиент" и "1 заказ- 1 дата".

1. Исследовательсский анализ:

- Посмотреть распределение количества заказов по пользователям;
- Посмотреть распределение заказов и транзакций по датам;
- Разбить товары по категориям;
- Топ категорий по продажам
- Проверить сезонность продаж по пяти самым продаваемым категориям
- Посмотреть среднюю цену продукта по категориям.

1. Сегментация пользователей:

- Категоризация пользователей по тратам;
- Категоризация пользователей по количеству заказов;
- Категоризация пользователей по давности последнего заказа;
- Итоговая сегментация.

1. Проверка гипотез:

- Средние траты лояльных клиентов больше, чем у спящих;
- Средние траты новых и потерянных клиентов не отличаются.

***P.S. В процессе исследования могут появиться новые гипотезы и/или метрики***

**Описание данных:**

Датасет описывает транзакции интернет-магазина товаров для дома и быта «Пока все ещё тут». Колонки в `ecom_dataset_upd.csv` :

- `date` — дата заказа;
- `customer_id` — идентификатор покупателя;
- `order_id` — идентификатор заказа;
- `product` — наименование товара;
- `quantity` — количество товара в заказе;
- `price` — цена товара.

**Импортируем библиотеки**

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import plotly.express as px
from scipy import stats as st
```

## Шаг. Загрузка данных и изучение общей информации

In [2]:
```
# Откроем файл
path = 'https://code.s3.yandex.net/datasets/'
data = pd.read_csv(path + 'ecom_dataset_upd.csv')
# Выведем первые 20 строк
data.head(20)
```

Out[2]:

| | date | customer_id | order_id | product | quantity | price |
|---|---|---|---|---|---|---|
| **0** | 2018100100 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Комнатное растение в горшке Алоэ Вера, d12, h30 | 1 | 142.0 |
| **1** | 2018100100 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Комнатное растение в горшке Кофе Арабика, d12,... | 1 | 194.0 |
| **2** | 2018100100 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Радермахера d-12 см h-20 см | 1 | 112.0 |
| **3** | 2018100100 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Хризолидокарпус Лутесценс d-9 см | 1 | 179.0 |
| **4** | 2018100100 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Циперус Зумула d-12 см h-25 см | 1 | 112.0 |
| **5** | 2018100100 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Шеффлера Лузеана d-9 см | 1 | 164.0 |
| **6** | 2018100100 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Юкка нитчатая d-12 см h-25-35 см | 1 | 134.0 |
| **7** | 2018100108 | 375e0724-f033-4c76-b579-84969cf38ee2 | 68479 | Настенная сушилка для белья Gimi Brio Super 100 | 1 | 824.0 |
| **8** | 2018100108 | 6644e5b4-9934-4863-9778-aaa125207701 | 68478 | Таз пластмассовый 21,0 л круглый "Водолей" С61... | 1 | 269.0 |
| **9** | 2018100109 | c971fb21-d54c-4134-938f-16b62ee86d3b | 68480 | Чехол для гладильной доски Colombo Persia Beig... | 1 | 674.0 |
| **10** | 2018100111 | 161e1b98-45ba-4b4e-8236-e6e3e70f6f7c | 68483 | Вешалка для брюк металлическая с резиновым пок... | 10 | 82.0 |
| **11** | 2018100112 | 86432d8d-b706-463b-bd5d-6a9e170daee3 | 68484 | Сушилка для белья потолочная Zalger Lift Basic... | 1 | 614.0 |
| **12** | 2018100113 | 4d93d3f6-8b24-403b-a74b-f5173e40d7db | 68485 | Чехол Eurogold Clean Basic хлопок для досок 12... | 1 | 187.0 |
| **13** | 2018100115 | 0948b0c2-990b-4a11-b835-69ac4714b21d | 68486 | Крючок одежный 2-х рожковый серый металлик с п... | 96 | 38.0 |
| **14** | 2018100116 | a576fa59-7b28-4a4c-a496-92f128754a94 | 68487 | Корзина мягкая пластиковая 17 л, М-пластика, М... | 1 | 188.0 |
| **15** | 2018100118 | 17213b88-1514-47a4-b8aa-ce51378ab34e | 68476 | Мини-сковорода Marmiton "Сердце" с антипригарн... | 1 | 239.0 |
| **16** | 2018100118 | 17213b88-1514-47a4-b8aa-ce51378ab34e | 68476 | Сковорода алюминиевая с антипригарным покрытие... | 1 | 824.0 |
| **17** | 2018100118 | 17213b88-1514-47a4-b8aa-ce51378ab34e | 68476 | Стеклянная крышка для сковороды ALPENKOK 26 см... | 1 | 262.0 |
| **18** | 2018100118 | 17213b88-1514-47a4-b8aa-ce51378ab34e | 68476 | Сушилка для белья напольная Colombo Star 18, 3679 | 1 | 1049.0 |
| **19** | 2018100121 | b731df05-98fa-4610-8496-716ec530a02c | 68474 | Доска гладильная Eurogold Professional 130x48 ... | 1 | 3299.0 |

In [3]:
```
# Выведем информацию о данных
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7474 entries, 0 to 7473
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   date         7474 non-null   int64  
 1   customer_id  7474 non-null   object 
 2   order_id     7474 non-null   int64  
 3   product      7474 non-null   object 
 4   quantity     7474 non-null   int64  
 5   price        7474 non-null   float64
dtypes: float64(1), int64(3), object(2)
memory usage: 350.5+ KB
```

In [4]:
```
# Посчитаем количество пропусков
data.isna().sum()
```

Out[4]:
```
date           0
customer_id    0
order_id       0
product        0
quantity       0
price          0
dtype: int64
```

In [5]:
```
# Выведем список характерных значений для столбцов
data.describe()
```

Out[5]:

| | date | order_id | quantity | price |
|---|---|---|---|---|
| **count** | 7.474000e+03 | 7474.000000 | 7474.000000 | 7474.000000 |
| **mean** | 2.018913e+09 | 49449.884265 | 2.362858 | 478.737501 |
| **std** | 4.278878e+05 | 32679.642404 | 14.500497 | 901.615895 |
| **min** | 2.018100e+09 | 12624.000000 | 1.000000 | 9.000000 |
| **25%** | 2.019022e+09 | 14833.000000 | 1.000000 | 97.000000 |
| **50%** | 2.019051e+09 | 68768.000000 | 1.000000 | 135.000000 |
| **75%** | 2.019063e+09 | 71257.750000 | 1.000000 | 439.000000 |
| **max** | 2.020013e+09 | 112789.000000 | 1000.000000 | 14917.000000 |

*Пропусков нет. Дату необходимо привести к формату datetime, чтобы определить период, за который предоставлены данные.*

In [6]:
```
# Приведём столбец date к формату datetime
data['datetime'] = pd.to_datetime(data['date'], format='%Y%m%d%H')
# Создадим отдельные столбцы для дат, дат и времени, часов и дней недели
data['hour'] = pd.to_datetime(data['datetime']).dt.hour
data['weekday'] = pd.to_datetime(data['datetime']).dt.weekday
data['date'] = pd.to_datetime(data['datetime']).dt.date
data.tail()
```

Out[6]:

| | date | customer_id | order_id | product | quantity | price | datetime | hour | weekday |
|---|---|---|---|---|---|---|---|---|---|
| **7469** | 2020-01-30 | 63208953-a8e4-4f77-9b47-3a46e7b72eee | 104002 | томата (помидор) Черниченский черри № 116 сорт... | 2 | 38.0 | 2020-01-30 21:00:00 | 21 | 3 |
| **7470** | 2020-01-30 | d99d25f1-4017-4fcd-8d29-c580cc695a1a | 107336 | Дендробиум Санок Анна Грин 1 ствол d-12 см | 1 | 869.0 | 2020-01-30 22:00:00 | 22 | 3 |
| **7471** | 2020-01-31 | 2c9bd08d-8c55-4e7a-9bfb-8c56ba42c6d6 | 106336 | Подставка для обуви резиновая Attribute 80x40 ... | 1 | 354.0 | 2020-01-31 02:00:00 | 2 | 4 |
| **7472** | 2020-01-31 | cdd17932-623e-415f-a577-3b31312fd0e2 | 102002 | Тагетис крупноцветковый рассада однолетних цве... | 1 | 128.0 | 2020-01-31 12:00:00 | 12 | 4 |
| **7473** | 2020-01-31 | 2e460a26-35af-453d-a369-a036e95a40e0 | 103225 | Вешалка для блузок 41 см красный Attribute AHM781 | 1 | 104.0 | 2020-01-31 15:00:00 | 15 | 4 |

In [7]:
```
# Определим период, за который предоставлены данные
min_date = data['date'].min()
max_date = data['date'].max()
print(f'Данные предоставлены за период с {min_date} по {max_date}')
```

```
Данные предоставлены за период с 2018-10-01 по 2020-01-31
```

In [8]:
```
# Добавим столбец с выручкой перемножив цену за единицу товара на количество
data['revenue'] = data['quantity']*data['price']
data.head()
```

Out[8]:

| | date | customer_id | order_id | product | quantity | price | datetime | hour | weekday | revenue |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Комнатное растение в горшке Алое Вера, d12, h30 | 1 | 142.0 | 2018-10-01 | 0 | 0 | 142.0 |
| **1** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Комнатное растение в горшке Кофе Арабика, d12,... | 1 | 194.0 | 2018-10-01 | 0 | 0 | 194.0 |
| **2** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Радермахера d-12 см h-20 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 |
| **3** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Хризолидокарпус Лутесценс d-9 см | 1 | 179.0 | 2018-10-01 | 0 | 0 | 179.0 |
| **4** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Циперус Зумула d-12 см h-25 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 |

**Вывод:**

- Перед нами датасет, который содержит информацию о транзакциях интернет-магазина товаров для дома и быта «Пока все ещё тут» за период с 1 октября 2018 г. по 31 января 2020 г.;
- Всего в датасете 7474 строк, каждая из которых содержит информацию о транзакциях интернет-магазина, после проверки на дубликаты можно будет точнее посчитать их количество;
- В столбце `date` данные были приведены к типу datetime;
- Данные типа object содержаться в столбцах `customer_id` и `product`;
- Данные типа float содержаться в столбце `price`;
- Данные типа float содержаться в столбцах `order_id` и `quantity`;
- Пропусков обнаружено не было.

## Шаг. Предобработка данных

In [9]:
```
print('Количество явных дубликатов:', data.duplicated().sum())
```

```
Количество явных дубликатов: 0
```

In [10]:
```
# Проверим бизнес-правило "1 заказ - 1 клиент"
incorrect_orders_customer = data.groupby('order_id')['customer_id'].nunique()
incorrect_orders_customer = incorrect_orders_customer[incorrect_orders_customer > 1].index.tolist()

# Проверим бизнес-правило "1 заказ - 1 дата"
incorrect_orders_date = data.groupby('order_id')['date'].nunique()
incorrect_orders_date = incorrect_orders_date[incorrect_orders_date > 1].index.tolist()

print("Неверные данные для бизнес-правила '1 заказ - 1 клиент':", len(incorrect_orders_customer))
print("Неверные данные для бизнес-правила '1 заказ - 1 дата':", len(incorrect_orders_date))
```

```
Неверные данные для бизнес-правила '1 заказ - 1 клиент': 29
Неверные данные для бизнес-правила '1 заказ - 1 дата': 190
```

Данные не соответсвующие бизнес-правилам '1 заказ - 1 клиент' и '1 заказ - 1 дата' могут исказить результаты исследования и привести к принятию неверных решений, поэтому дальнейшее исследование лучше продолжить без них.

In [11]:
```
df = data.query('order_id != @incorrect_orders_date and order_id != @incorrect_orders_customer')

# Проверим бизнес-правила ещё раз
incorrect_orders_customer = df.groupby('order_id')['customer_id'].nunique()
incorrect_orders_customer = incorrect_orders_customer[incorrect_orders_customer > 1].index.tolist()

incorrect_orders_date = df.groupby('order_id')['date'].nunique()
incorrect_orders_date = incorrect_orders_date[incorrect_orders_date > 1].index.tolist()
```

```
print("Неверные данные для бизнес-правила '1 заказ - 1 клиент':", len(incorrect_orders_customer))
print("Неверные данные для бизнес-правила '1 заказ - 1 дата':", len(incorrect_orders_date))

df.head()
```

```
Неверные данные для бизнес-правила '1 заказ - 1 клиент': 0
Неверные данные для бизнес-правила '1 заказ - 1 дата': 0
```

Out[11]:

| | date | customer_id | order_id | product | quantity | price | datetime | hour | weekday | revenue |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Комнатное растение в горшке Алоэ Вера, d12, h30 | 1 | 142.0 | 2018-10-01 | 0 | 0 | 142.0 |
| **1** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Комнатное растение в горшке Кофе Арабика, d12,... | 1 | 194.0 | 2018-10-01 | 0 | 0 | 194.0 |
| **2** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Радермахера d-12 см h-20 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 |
| **3** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Хризолидокарпус Лутесценс d-9 см | 1 | 179.0 | 2018-10-01 | 0 | 0 | 179.0 |
| **4** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | Циперус Зумула d-12 см h-25 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 |

In [12]:

```
# Приведём наименования товаров к единой форме
df['product'] = df['product'].str.lower()
df['product'] = df['product'].str.replace('ё','е', regex=True)
df['product'] = df['product'].str.replace('(', '', regex=True)
df['product'] = df['product'].str.replace(')', '', regex=True)
df['product'] = df['product'].str.replace(',', '', regex=True)
df.head()
```

```
C:\Users\dimch\AppData\Local\Temp\ipykernel_13212\3923534725.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['product'] = df['product'].str.lower()
C:\Users\dimch\AppData\Local\Temp\ipykernel_13212\3923534725.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['product'] = df['product'].str.replace('ё','е', regex=True)
C:\Users\dimch\AppData\Local\Temp\ipykernel_13212\3923534725.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['product'] = df['product'].str.replace('(', '', regex=True)
C:\Users\dimch\AppData\Local\Temp\ipykernel_13212\3923534725.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['product'] = df['product'].str.replace(')', '', regex=True)
C:\Users\dimch\AppData\Local\Temp\ipykernel_13212\3923534725.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['product'] = df['product'].str.replace(',', '', regex=True)
```

Out[12]:

| | date | customer_id | order_id | product | quantity | price | datetime | hour | weekday | revenue |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | комнатное растение в горшке алоэ вера d12 h30 | 1 | 142.0 | 2018-10-01 | 0 | 0 | 142.0 |
| **1** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | комнатное растение в горшке кофе арабика d12 h25 | 1 | 194.0 | 2018-10-01 | 0 | 0 | 194.0 |
| **2** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | радермахера d-12 см h-20 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 |
| **3** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | хризолидокарпус лутесценс d-9 см | 1 | 179.0 | 2018-10-01 | 0 | 0 | 179.0 |
| **4** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | циперус зумула d-12 см h-25 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 |

In [13]:

```
# Посчитаем дубликаты по наименованию продукта и идентификатору заказа
df[df.duplicated(subset = ['product', 'order_id'], keep = False)]
```

Out[13]:

| | date | customer_id | order_id | product | quantity | price | datetime | hour | weekday | revenue |
|---|---|---|---|---|---|---|---|---|---|---|
| **80** | 2018-10-04 | 32de7df8-8d4f-4c84-a7b9-c41d00dd83ba | 68522 | эвкалипт гунни d-17 см h-60 см | 1 | 1409.0 | 2018-10-04 09:00:00 | 9 | 3 | 1409.0 |
| **94** | 2018-10-04 | 32de7df8-8d4f-4c84-a7b9-c41d00dd83ba | 68522 | эвкалипт гунни d-17 см h-60 см | 1 | 1409.0 | 2018-10-04 13:00:00 | 13 | 3 | 1409.0 |
| **124** | 2018-10-07 | ce0e0c29-8c8b-4714-84bd-80957224d4cf | 68557 | сумка-тележка хозяйственная gimi market синяя | 1 | 1874.0 | 2018-10-07 16:00:00 | 16 | 6 | 1874.0 |
| **126** | 2018-10-07 | ce0e0c29-8c8b-4714-84bd-80957224d4cf | 68557 | сумка-тележка хозяйственная gimi market синяя | 1 | 1874.0 | 2018-10-07 17:00:00 | 17 | 6 | 1874.0 |
| **278** | 2018-10-17 | 4d93d3f6-8b24-403b-a74b-f5173e40d7db | 68668 | щетка для посуды *мила* sv3182 1807009 | 50 | 27.0 | 2018-10-17 11:00:00 | 11 | 2 | 1350.0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **6697** | 2019-10-26 | 56710968-02ea-46b7-9638-0ad9fa8544d0 | 73095 | тимьян d-9 см | 8 | 119.0 | 2019-10-26 16:00:00 | 16 | 5 | 952.0 |
| **6698** | 2019-10-27 | f0fe3add-04d4-4488-8852-7e9e25b95fe9 | 73101 | хлорофитум d-7 см h-10 см укорененный черенок | 2 | 74.0 | 2019-10-27 13:00:00 | 13 | 6 | 148.0 |
| **6702** | 2019-10-27 | f0fe3add-04d4-4488-8852-7e9e25b95fe9 | 73101 | хлорофитум d-7 см h-10 см укорененный черенок | 2 | 74.0 | 2019-10-27 19:00:00 | 19 | 6 | 148.0 |
| **6704** | 2019-10-28 | cb65d08a-dae7-4890-aef0-bb9f79055e02 | 73108 | мирт d-9 см h-15 см | 1 | 134.0 | 2019-10-28 08:00:00 | 8 | 0 | 134.0 |
| **6711** | 2019-10-28 | cb65d08a-dae7-4890-aef0-bb9f79055e02 | 73108 | мирт d-9 см h-15 см | 1 | 134.0 | 2019-10-28 21:00:00 | 21 | 0 | 134.0 |

248 rows × 10 columns

От дубликатов и некорректных данных избавились. Теперь можно посчитать количество заказов и пользователей, а также посмотреть как изменился датасет.

In [14]:

```
print('Количество покупателей:', df['customer_id'].nunique())
print('Количество заказов:', df['order_id'].nunique())
min_dt = df['date'].min()
max_dt = df['date'].max()
print(f'Данные предоставлены за период с {min_dt} по {max_dt}')
print(100*'-')
df.info()
```

```
Количество покупателей: 2265
Количество заказов: 3327
Данные предоставлены за период с 2018-10-01 по 2020-01-31
----------------------------------------------------------------------------------------------------
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5253 entries, 0 to 7473
Data columns (total 10 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   date         5253 non-null   object
 1   customer_id  5253 non-null   object
 2   order_id     5253 non-null   int64
 3   product      5253 non-null   object
 4   quantity     5253 non-null   int64
 5   price        5253 non-null   float64
 6   datetime     5253 non-null   datetime64[ns]
 7   hour         5253 non-null   int64
 8   weekday      5253 non-null   int64
 9   revenue      5253 non-null   float64
dtypes: datetime64[ns](1), float64(2), int64(4), object(3)
memory usage: 451.4+ KB
```

**Вывод:** В ходе предобработки данных явных дубликатов обнаружено не было, были обработаны значения не подходящие под бизнес-правила '1 заказ - 1 клиент' и '1 заказ - 1 дата'. По итогу у нас датасет сократился до 5001 строки, уникальных покупателей 2185, уникальных заказов 3236, интервал дат не изменился

## Шаг. Исследовательсский анализ

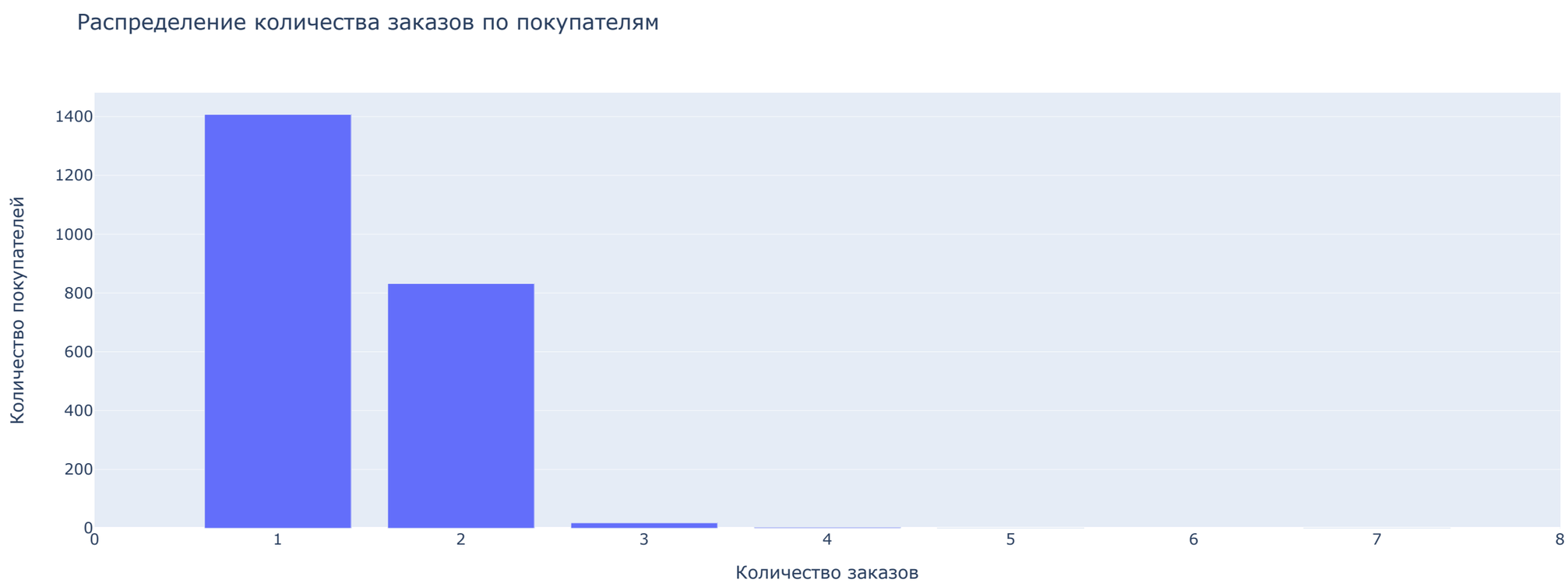
## Распределение заказов по покупателям

In [15]:
```
# Посчитаем количество заказов на количество пользователей
orders_by_customers = df.pivot_table(index='customer_id',
                                     values='order_id',
                                     aggfunc='nunique').reset_index().sort_values(by='order_id', ascending=False)
orders_by_customers = orders_by_customers.pivot_table(index='order_id',
                                                      values='customer_id',
                                                      aggfunc='count').sort_values(by='order_id',
                                                                                   ascending=False).reset_index().rename(columns={'order_id':'order_count', 'customer_id':'customer_count'})
orders_by_customers
```

Out[15]:

| | order_count | customer_count |
|---|---|---|
| **0** | 126 | 1 |
| **1** | 35 | 1 |
| **2** | 17 | 1 |
| **3** | 7 | 1 |
| **4** | 5 | 1 |
| **5** | 4 | 3 |
| **6** | 3 | 18 |
| **7** | 2 | 832 |
| **8** | 1 | 1407 |

In [16]:
```
fig = px.bar(orders_by_customers,
             x='order_count',
             y='customer_count',
             title='Распределение количества заказов по покупателям',
             labels={'order_count':'Количество заказов', 'customer_count':'Количество покупателей'})
fig.update_xaxes(range=[0,8])
fig.show()
```




Большинство пользователей совершило лишь один заказ(1407), два заказа совершило 832 клиента, больше двух заказов всего у нескольких десятков.

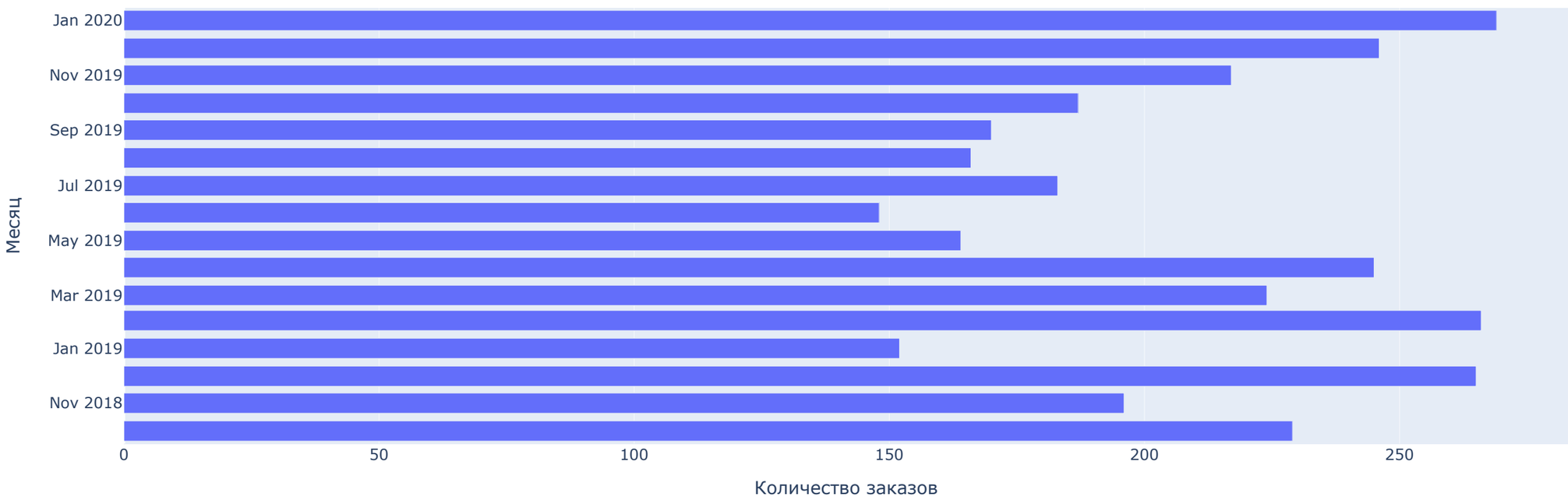
## Распределение заказов и транзакций по датам

In [17]:
```
# Посчитаем количество заказов по месяцам
orders_by_date = df.pivot_table(index=pd.to_datetime(df['date']).dt.strftime("%Y-%m"),
                                values='order_id',
                                aggfunc='nunique').sort_values(by='date',
                                                               ascending=False).reset_index().rename(columns={'order_id':'order_count'})
orders_by_date
```

Out[17]:

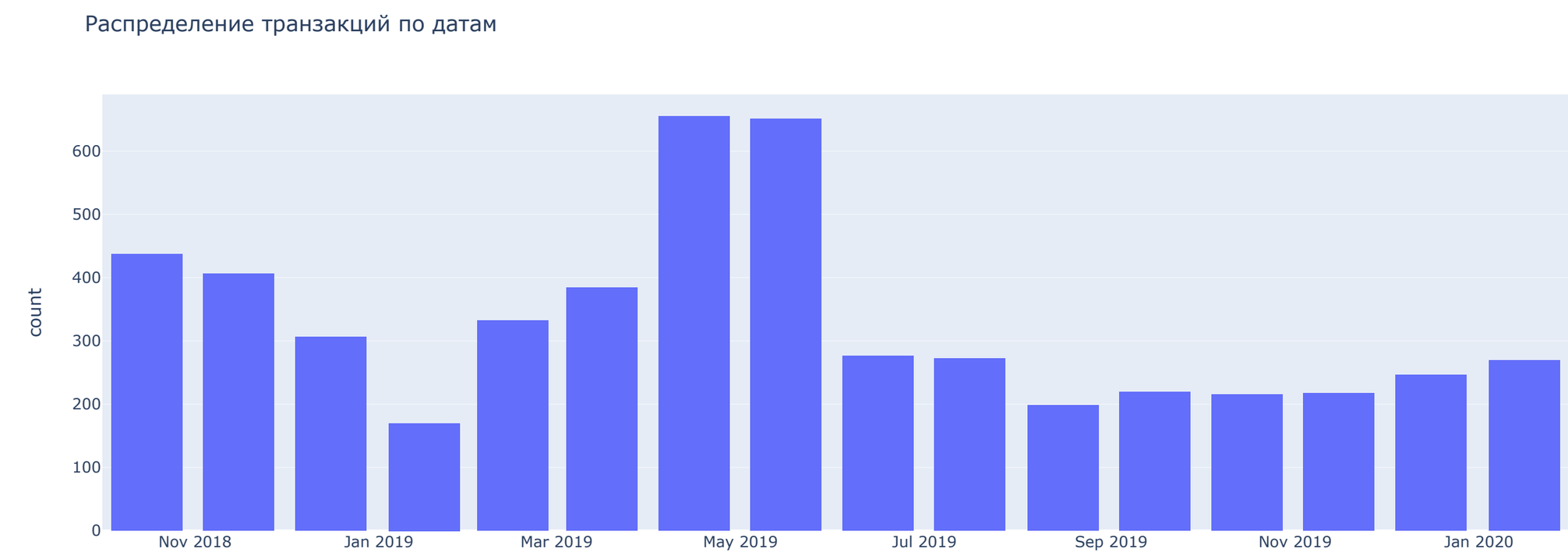
| | date | order_count |
|---|---|---|
| **0** | 2020-01 | 269 |
| **1** | 2019-12 | 246 |
| **2** | 2019-11 | 217 |
| **3** | 2019-10 | 187 |
| **4** | 2019-09 | 170 |
| **5** | 2019-08 | 166 |
| **6** | 2019-07 | 183 |
| **7** | 2019-06 | 148 |
| **8** | 2019-05 | 164 |
| **9** | 2019-04 | 245 |
| **10** | 2019-03 | 224 |
| **11** | 2019-02 | 266 |
| **12** | 2019-01 | 152 |
| **13** | 2018-12 | 265 |
| **14** | 2018-11 | 196 |
| **15** | 2018-10 | 229 |

In [18]:
```
# Визуализируем распределение заказов по месяцам
fig = px.bar(orders_by_date,
             x='order_count',
             y='date',
             title='Распределение количества заказов по месяцам',
             labels={'order_count':'Количество заказов', 'date':'Месяц'})
fig.show()
```




Лидирующими по количеству заказов месяцами стали январь 2020(269), февраль 2019(266) и декабрь 2018(265), самы бедный на заказы месяц- июнь 2019(148).

In [19]:
```
# Визуализируем распределение транзакций по датам
fig = px.histogram(df,
             x='date',
             title='Распределение транзакций по датам',
             labels={'date':'Дата'},
             nbins=30)
fig.update_layout(bargap=0.2)
fig.show()
```




Больше всего транзакций пришлось на апрель(655) и май(651) 2019, в январе 2019(170) их было меньше всего.

### Разбивка товаров по категориям

In [20]:
```
# функция для создания категорий по первому слову в наименовании товара
def get_first_word_category(product):
    first_word = product.split(' ')[0]
    return first_word


# применяем функцию к столбцу
df['category'] = df['product'].apply(get_first_word_category)

df.head()
```

```
C:\Users\dimch\AppData\Local\Temp\ipykernel_13212\237629905.py:8: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[20]:

| | date | customer_id | order_id | product | quantity | price | datetime | hour | weekday | revenue | category |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | комнатное растение в горшке алоэ вера d12 h30 | 1 | 142.0 | 2018-10-01 | 0 | 0 | 142.0 | комнатное |
| **1** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | комнатное растение в горшке кофе арабика d12 h25 | 1 | 194.0 | 2018-10-01 | 0 | 0 | 194.0 | комнатное |
| **2** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | радермахера d-12 см h-20 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 | радермахера |
| **3** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | хризолидокарпус лутесценс d-9 см | 1 | 179.0 | 2018-10-01 | 0 | 0 | 179.0 | хризолидокарпус |
| **4** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | циперус зумула d-12 см h-25 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 | циперус |

In [21]:
```
df['category'].sort_values(ascending=True).unique()
```

Out[21]:
```
array(['термокружка', 'автоматическая', 'адиантум', 'азалия', 'аквилегия',
       'алиссум', 'алоэ', 'альбука', 'амариллис', 'анемона', 'антижир',
       'антинакипин', 'антуриум', 'аптения', 'арбуз', 'аргирантерум',
       'ароматизированное', 'артемизия', 'аспарагус', 'астра',
       'афеляндра', 'базилик', 'бак', 'бакопа', 'бальзам', 'бальзамин',
       'банка', 'бархатцы', 'бегония', 'бельевые', 'бензин', 'бидон',
       'блюдо', 'блюдце', 'буддлея', 'бузульник', 'бульонница', 'ваза',
       'вакуумный', 'валериана', 'ванна', 'вантуз', 'ведро', 'веник',
       'венчик', 'вербейник', 'вербена', 'веревка', 'вероника', 'весы',
       'вешалка', 'вешалка-перекладина', 'вешалка-плечики',
       'вешалка-стойка', 'вешалка-сушилка', 'вешалки', 'вилка', 'виола',
       'вкладыши', 'газания', 'гайлардия', 'гардения', 'гвоздика',
       'георгина', 'герань', 'гербера', 'гиацинт', 'гимнокалициум',
       'гиностемма', 'гипоаллергенный', 'гипсофила', 'гладильная',
       'глоксиния', 'годеция', 'горох', 'гортензия', 'готовая', 'губка',
       'двуспальное', 'девичий', 'декабрист', 'декоративная',
       'дендробиум', 'держатель', 'джункус', 'диффенбахия', 'дозатор',
       'доска', 'драцена', 'душица', 'дыня', 'ель', 'емкость', 'ерш',
       'жестяная', 'жидкое', 'замиокулькас', 'запасная', 'защитный',
       'зверобой', 'земляника', 'змееголовник', 'зубная', 'измельчитель',
       'измерительный', 'импатиенс', 'искусственная', 'искусственный',
       'иссоп', 'каланхое', 'калатея', 'календула', 'калибрахоа', 'калла',
       'каллуна', 'калоцефалус', 'камнеломка', 'кампанула', 'капсикум',
       'капуста', 'карниз', 'картофелемялка', 'кастрюля', 'кипарисовик',
       'кипятильник', 'кисточка', 'клен', 'клубника', 'ключница', 'ковер',
       'коврик', 'ковш', 'кодонанта', 'колеус', 'колокольчик', 'кольца',
       'комнатное', 'комод', 'комплект', 'кондиционер', 'контейнер',
       'концентрат', 'кореопсис', 'корзина', 'корзинка', 'кориандр',
       'коробка', 'корыто', 'космея', 'котел', 'котовник', 'кофе', 'кофр',
       'крассула', 'крепеж', 'крокусы', 'кружка', 'крышка', 'крючок',
       'кувшин', 'кухонное', 'кухонные', 'лаванда', 'лаватера', 'лавр',
       'лантана', 'лапчатка', 'левкой', 'лен', 'лестница',
       'лестница-стремянка', 'лилейник', 'линейка', 'литопс', 'лобелия',
       'ложка', 'лопатка', 'лоток', 'лук', 'львиный', 'любисток',
       'мантоварка', 'мантоварка-пароварка', 'масленка', 'маттиола',
       'махровое', 'махровый', 'мединилла', 'мелисса', 'мерный', 'мешок',
       'миксер', 'мимоза', 'мирт', 'миска', 'многолетнее',
       'многофункциональный', 'модульная', 'молодило', 'монарда',
       'морковь', 'муляж', 'муррайя', 'мусорный', 'мыло', 'мыло-скраб',
       'мыльница', 'мята', 'набор', 'наволочка', 'наматрасник',
       'наматрацник', 'насадка', 'насадка-моп', 'насадка-отжим',
       'настенная', 'настольная', 'настурция', 'незабудка', 'нетканые',
       'нефролепис', 'нивянник', 'новогоднее', 'нож', 'ножеточка',
       'нолина', 'обувница-3', 'овощеварка', 'овощечистка', 'овсянница',
       'огурец', 'одеяло', 'однолетнее', 'окномойка', 'ополаскиватель',
       'орехоколка', 'осина', 'основание', 'отбеливатель', 'отделитель',
       'отжим', 'папоротник', 'паста', 'патиссон', 'пахира', 'пеларгония',
       'пена', 'пеперомия', 'перчатки', 'петля', 'петрушка', 'петуния',
       'пиретрум', 'платикодон', 'плед', 'плечики', 'подарочный',
       'подвесное', 'подголовник', 'подкладка', 'пододеяльник',
       'подрукавник', 'подсолнечник', 'подставка', 'подушка', 'покрывало',
       'покрытие', 'полка', 'полки', 'половник', 'полотенце', 'портулак',
       'пресс', 'примула', 'прищепки', 'пробка', 'просеиватель',
       'простынь', 'простыня', 'противень', 'пряные', 'пуансетия',
       'пуансеттия', 'пылесос', 'пьезозажигалка', 'радермахера',
       'разделочная', 'ранункулус', 'рассада', 'рассекатель', 'решетка',
       'роза', 'розмарин', 'ролик', 'рудбекия', 'рукав', 'ручка',
       'ручка-скоба', 'рыбочистка', 'салат', 'салатник', 'салфетка',
       'салфетница', 'сальвия', 'сантолина', 'сахарница', 'сверло',
       'сверло-фреза', 'светильник', 'седум', 'сельдерей',
       'сервировочная', 'сетка', 'сиденье', 'синнингия', 'сито',
       'скатерть', 'скиммия', 'складная', 'складной', 'сковорода',
       'скребок', 'сменная', 'сменный', 'сметка', 'смолевка', 'совок',
       'соковарка', 'соковыжималка', 'соланум', 'солидаго', 'сотейник',
       'спатифиллум', 'средство', 'стакан', 'стеклянная', 'стеллаж',
       'стиральный', 'столовая', 'столовый', 'стремянка',
       'стремянка-табурет', 'стремянки', 'стяжка', 'стяжки', 'суккулент',
       'сумка', 'сумка-тележка', 'сушилка', 'сциндапсус', 'табак',
       'тагетис', 'таз', 'тарелка', 'тележка', 'терка', 'термокружка',
       'термометр', 'термос', 'термостакан', 'тимьян', 'ткань',
       'толкушка', 'томат', 'томата', 'тряпка', 'тряпкодержатель',
       'тюльпан', 'увлажняющая', 'уголок', 'укроп', 'универсальное',
       'универсальный', 'урна', 'урна-пепельница', 'утюг', 'фал',
       'фаленопсис', 'фарфоровая', 'фатсия', 'фен', 'фиалка',
       'физостегия', 'фиксатор-шар', 'фикус', 'фиттония', 'флокс',
       'форма', 'фоторамка', 'фуксия', 'халат', 'хамедорея', 'хлебница',
       'хлорофитум', 'холодная', 'хоста', 'хризантема', 'хризолидокарпус',
       'цветок', 'цветущее', 'целозия', 'цикламен', 'цинерания',
       'цинерария', 'циния', 'цинния', 'циперус', 'цитрофортунелла',
       'чабер', 'чайная', 'чайник', 'чайный', 'чехол', 'чистящий',
       'шалфей', 'швабра', 'шеффлера', 'шило', 'шнур', 'шпагат',
       'шпингалет', 'штанга', 'штангенциркуль', 'штора', 'щетка',
       'щетка-сметка', 'щетка-утюжок', 'эвкалипт', 'энотера',
       'эпипремнум', 'этажерка', 'эхеверия', 'эхинокактус', 'эшшольция',
       'юкка', 'ясколка', 'ящик'], dtype=object)
```

Получилось слишком много категорий, поэтому разобьём на более общие категории по ключевым словам в списках.

In [22]:
```
plants = ['адиантум','азалия', 'аквилегия', 'алиссум', 'алоэ', 'альбука', 'амариллис', 'анемона', 'антуриум', 'аптения', 'арбуз',
          'артемизия', 'аспарагус', 'астра', 'афеляндра', 'бадан', 'базилик', 'баклажан', 'бакопа', 'бальзамин', 'барвинок',
          'бегония', 'буддлея', 'бузульник', 'вантуз', 'вербена', 'веревка', 'вероника', 'весы', 'виола', 'вкладыши', 'гайлардия',
          'гардения', 'гвоздика', 'герань', 'гербера', 'гиацинт', 'гимнокалициум', 'глоксиния', 'гортензия', 'декабрист',
          'дендробиум', 'джункус', 'диффенбахия', 'драцена', 'душица', 'дыня', 'замиокулькас', 'зверобой', 'земляника',
          'импатиенс', 'иссоп', 'кабачок', 'каланхое', 'калатея', 'калибрахоа', 'каллуна', 'калоцефалус', 'камнеломка', 'кампанула',
          'капсикум', 'кипарисовик', 'клен', 'колокольчик', 'комнатное', 'кореопсис', 'котовник', 'кофе', 'крассула', 'крокусы',
          'лаванда', 'лавр', 'лантана', 'лапчатка', 'лен', 'лилейник', 'литопс', 'лобелия', 'мединилла', 'мелисса', 'мимоза', 'мирт',
          'многолетнее', 'молодило', 'монарда', 'муррайя', 'мускари', 'мята', 'нефролепис', 'нивянник', 'новогоднее', 'нолина', 'овсянница',
          'папоротник', 'патиссон', 'пахира', 'пеларгония', 'пеперомия', 'петуния', 'пиретрум', 'платикодон', 'подарочный', 'подвесное',
          'подсолнечник', 'примула', 'просеиватель', 'пряные', 'пуансетия', 'пуансеттия', 'радермахера', 'роза', 'розмарин', 'рудбекия',
          'сантолина', 'седум', 'синнингия', 'скиммия', 'соланум', 'солидаго', 'спатифиллум', 'суккулент', 'сциндапсус', 'тимьян', 'тыква',
          'фаленопсис', 'фатсия', 'фиалка', 'физостегия', 'фикус', 'фиттония', 'флокс', 'фуксия', 'хамедорея', 'хлорофитум', 'хоста',
          'хризантема', 'хризолидокарпус', 'цветущее', 'цикламен', 'циперус', 'цитрофортунелла', 'чабер', 'шеффлера', 'эвкалипт', 'энотера',
          'эпипремнум', 'эхеверия', 'эхинацея', 'эхинокактус', 'юкка', 'ясколка', 'гипсофил', 'тюльпан', 'гиностемма', 'кодонанта ',
          'калла', 'георгина', 'осина', 'вербейник', 'ранункулус']
bags = ['сумка', 'тележка', 'тележки']
seeds = ['рассада', 'кассет', 'томат', 'настурция', 'космея', '1 г', '2 г', '3 г', '4 г', '5 г', '6 г', '7 г', '8 г', '9 г', '0 г',
         'цинния ацтек', 'девичий виноград', 'клубника']
artificial = ['муляж', 'искусственн']
kitchen = ['термокружка avex', 'банка', 'бидон', 'блюдо', 'блюдце', 'бульонница', 'ваза', 'вакуумный пакет', 'венчик', 'вилка',
           'жестяная банка', 'картофелемялка', 'кастрюля', 'ковш', 'контейнер', 'котел', 'кружка', 'кувшин', 'ложка',
           'лопатка', 'мантоварка', 'мерный стакан', 'миксер', 'набор бокалов', 'набор кружек', 'набор ножей', 'набор посуды',
           'набор стаканов', 'стакан', 'рыбочистка', 'набор столовых приборов', 'набор форм', 'набор фужеров', 'нож',
           'ножеточка', 'овощеварка', 'овощечистка', 'орехоколка', 'подарочный набор', 'подставка', 'половник',
           'разделочная доска', 'салфетка', 'салфетница', 'сахарница', 'свч', 'сито', 'скалка', 'миска', 'выпеч', 'скатерть',
           'сковорода', 'соковарка', 'соковыжималка', 'сотейник', 'тарелка', 'терка', 'термос', 'термостакан', 'толкушка',
           'тортница', 'форма', 'хлебница', 'ложка', 'чайник', 'чайный набор', 'пресс для чеснока', 'противень', 'электроштопор',
           'кухонное полотенце', 'набор кухонных полотенец', 'набор махровых салфеток', 'полотенце кухонное',
           'полотенце махровое', 'полотенце прессованное', 'нетканые салфетки','шприц', 'стеклянная крышка', 'бензин',
           'доска разделочная', 'пьезозажигалка', 'масленка', 'салатник', 'просеиватель', 'емкость для соуса',
           'рассекатель пламени на газовую плиту', 'ополаскиватель для посудомоечных машин', 'отделитель косточек',
           'крышка оцинкованная', 'скребок кондитерский', 'кипятильник']
storage = ['кофр', 'декоративная', 'короб', 'коробка', 'складная', 'чехол', 'корзина', 'корзинка', 'этажерка', 'ящик',
           'обувница', 'вакуумный', 'бак', 'стеллаж', 'лоток knit a4', 'подставка для обуви']
tools = ['крепеж', 'крючок', 'линейка', 'инструмент', 'напильник', 'сверел', 'петля', 'сверло',
         'стяжка', 'уголок', 'фиксатор', 'ручка-скоба', 'сварка', 'шило', 'шнур', 'штангенциркуль', 'решетка вентиляционная',
         'угольник', 'шпагат полипропиленовый', 'лестница', 'стремянк', 'фал', 'шпингалет', 'ручка мебельная',
         'насадка на валик']
bedroom = ['постельного', 'двуспальное', 'наматра', 'наволочка', 'простыня', 'одеяло', 'плед', 'пододеяльник', 'подушка',
           'покрывало', 'простынь']
homemade = ['весы', 'вешалк', 'вкладыши', 'гладильная', 'ключница', 'ковер', 'ковр', 'комод', 'комплект махровых', 'вешалок',
            'сушилка', 'гладильн', 'плечики', 'подрукавник', 'полк', 'пылесос', 'рукав', 'сетка', 'пуф', 'сушилка для белья',
            'термометр', 'ткань', 'утюг', 'фоторамка',  'штора', 'светильник', 'прищеп']
bathroom = ['ванн', 'дозатор', 'соль', 'зубная', 'карниз', 'кольца для штор', 'комплект для мытья', 'полотенц',
            'мыльница', 'основание для пробки', 'пена', 'подголовник', 'увлажняющая маска', 'мыло', 'кондиционер для белья',
            'фен', 'штора со встроенными','халат','мыльница', 'держатель для туалетной бумаги', 'унитаз', 'таз', 'ведро']
cleaning = ['антижир', 'антижир', 'антинакипин', 'ароматизированное', 'бальзам', 'веник', 'губка', 'стирки', 'ерш', 'ёрш',
            'концентрат', 'мешок', 'насадка-отжим', 'мусорный', 'перчатки', 'отжим для ведра', 'салфеток', 'салфетки', 'сметка', 'окномойка',
```

```
        'ополаскиватель', 'ролик', 'сиденье', 'совок', 'средство', 'стиральный', 'универсальное', 'урна', 'насадка-моп',
        'урна-пепельница', 'тряпка', 'чистящий', 'швабра', 'швабр', 'щетка', 'скребок 44 см', 'насадка для мытья',
        'скребок для окон', 'паста для полировки']
```

In [23]:
```
# Создаем функцию для создания категорий
def category(product):
    for word in storage:
        if word in product:
            return 'хранение'
    for word in kitchen:
        if word in product:
            return 'кухонная утварь'
    for word in tools:
        if word in product:
            return 'инструменты и товары для ремонта'
    for word in bedroom:
        if word in product:
            return 'товары для спальни'
    for word in bathroom:
        if word in product:
            return 'ванная комната'
    for word in cleaning:
        if word in product:
            return 'уборка'
    for word in homemade:
        if word in product:
            return 'товары для дома'
    for word in bags:
        if word in product:
            return 'сумки'
    for word in artificial:
        if word in product:
            return 'муляжи'
    for word in seeds:
        if word in product:
            return 'рассада'
    for word in plants:
        if word in product:
            return 'растения'
    return 'другое'

# Применяем функцию к столбцу датасета
df['category'] = df['product'].apply(category)
df['category'].unique()
```

```
C:\Users\dimch\AppData\Local\Temp\ipykernel_13212\1730354170.py:39: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[23]:
```
array(['растения', 'товары для дома', 'ванная комната', 'хранение',
       'инструменты и товары для ремонта', 'сумки', 'рассада', 'муляжи',
       'уборка', 'кухонная утварь', 'товары для спальни'], dtype=object)
```

In [24]:
```
df.head()
```

Out[24]:

| | date | customer_id | order_id | product | quantity | price | datetime | hour | weekday | revenue | category |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | комнатное растение в горшке алоэ вера d12 h30 | 1 | 142.0 | 2018-10-01 | 0 | 0 | 142.0 | растения |
| **1** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | комнатное растение в горшке кофе арабика d12 h25 | 1 | 194.0 | 2018-10-01 | 0 | 0 | 194.0 | растения |
| **2** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | радермахера d-12 см h-20 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 | растения |
| **3** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | хризолидокарпус лутесценс d-9 см | 1 | 179.0 | 2018-10-01 | 0 | 0 | 179.0 | растения |
| **4** | 2018-10-01 | ee47d746-6d2f-4d3c-9622-c31412542920 | 68477 | циперус зумула d-12 см h-25 см | 1 | 112.0 | 2018-10-01 | 0 | 0 | 112.0 | растения |

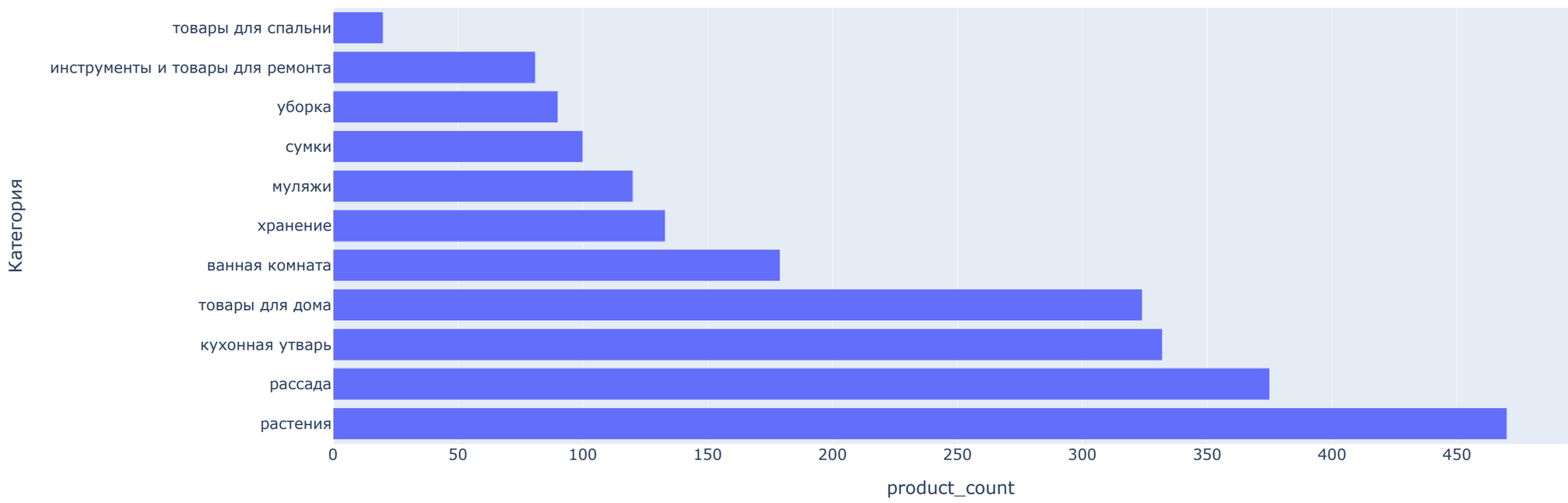
In [25]:
```
# Посчитаем количество уникальных товаров в категориях
category_pivot = df.pivot_table(index='category',
                                values='product',
                                aggfunc='nunique').sort_values(by='product',
                                                               ascending=False).reset_index().rename(columns={'product':'product_count'})

category_pivot
```

Out[25]:

| | category | product_count |
|---|---|---|
| **0** | растения | 470 |
| **1** | рассада | 375 |
| **2** | кухонная утварь | 332 |
| **3** | товары для дома | 324 |
| **4** | ванная комната | 179 |
| **5** | хранение | 133 |
| **6** | муляжи | 120 |
| **7** | сумки | 100 |
| **8** | уборка | 90 |
| **9** | инструменты и товары для ремонта | 81 |
| **10** | товары для спальни | 20 |

In [26]:
```
# Визуализируем распределение товаров по категориям
fig = px.bar(category_pivot,
             x='product_count',
             y='category',
             title='Распределение количества товаров по категориям',
             labels={'order_count':'Количество товаров', 'category':'Категория'})
fig.show()
```




Самой большой получилась категория растений, рассада и кухонная утварь на втором и третьем местах.

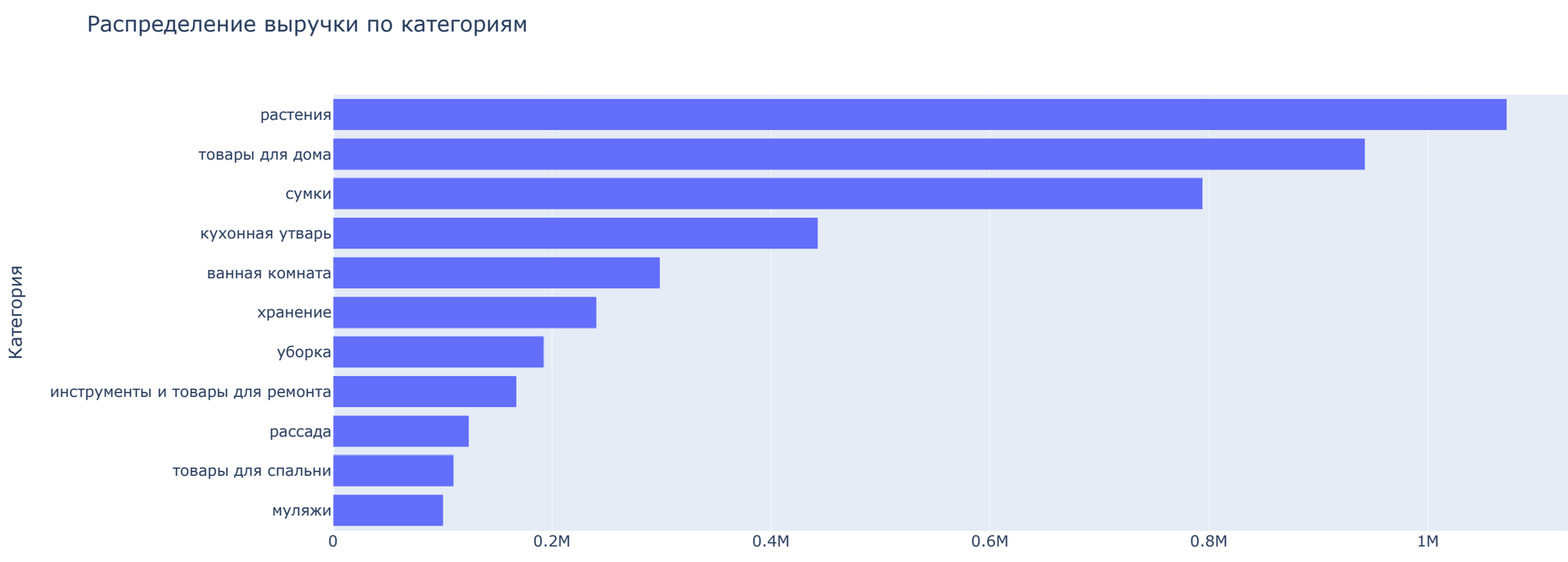
### Топ категорий по продажам

In [27]:
```
# Посчитаем выручку и количество проданных товаров по категориям
top_category = df.pivot_table(index='category',
                              values=['quantity', 'revenue'],
                              aggfunc='sum').sort_values(by='revenue').reset_index()
top_category
```

Out[27]:

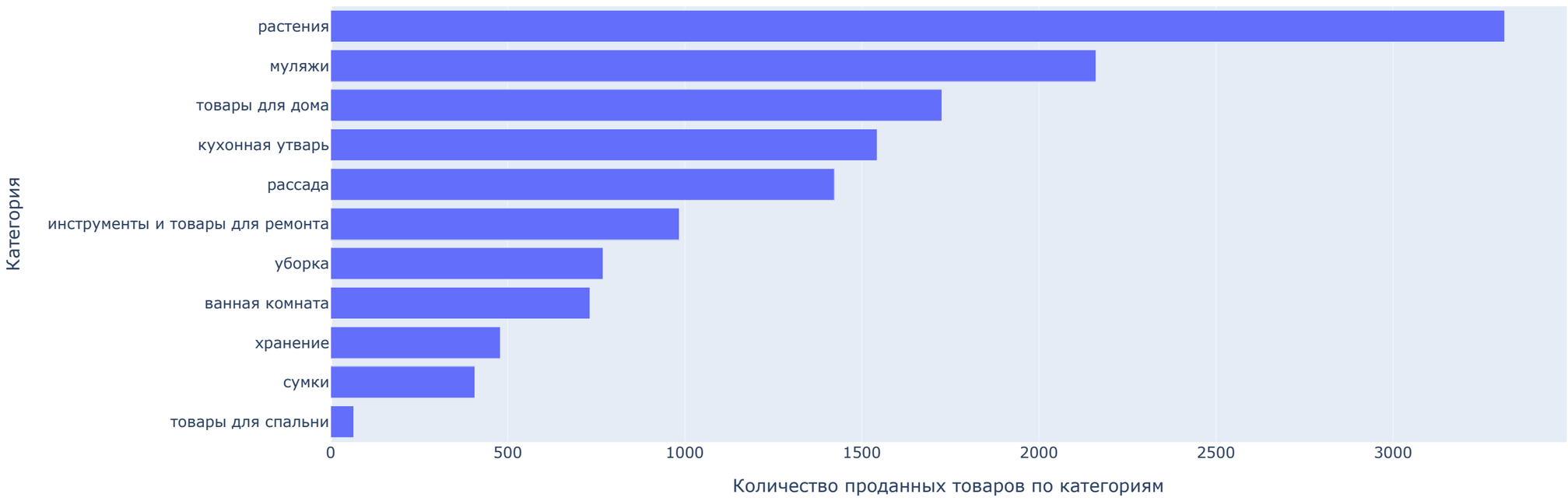
| | category | quantity | revenue |
|---|---|---|---|
| **0** | муляжи | 2161 | 1.007313e+05 |
| **1** | товары для спальни | 65 | 1.101600e+05 |
| **2** | рассада | 1422 | 1.242140e+05 |
| **3** | инструменты и товары для ремонта | 984 | 1.676673e+05 |
| **4** | уборка | 769 | 1.924915e+05 |
| **5** | хранение | 479 | 2.407382e+05 |
| **6** | ванная комната | 732 | 2.986300e+05 |
| **7** | кухонная утварь | 1543 | 4.429470e+05 |
| **8** | сумки | 407 | 7.943032e+05 |
| **9** | товары для дома | 1726 | 9.426285e+05 |
| **10** | растения | 3315 | 1.072149e+06 |

In [28]:
```
# Визуализируем распределение выручки по категориям
fig = px.bar(top_category,
             x='revenue',
             y='category',
             title='Распределение выручки по категориям',
             labels={'revenue':'Суммарная выручка по категориям', 'category':'Категория'})
fig.show()
```




Больше всего денег приносят товары категорий растения(1,04 млн), товары для дома(905,67 тыс) и сумки(739,82 тыс). Самые неприбыльными оказались рассада(121,03 тыс), товары для спальни(110,16 тыс) и муляжи(98,24 тыс).

In [29]:
```
# Визуализируем распределение количества проданных товаров по категориям
fig = px.bar(top_category,
             x='quantity',
             y='category',
             title='Распределение количества проданных товаров по категориям',
             labels={'quantity':'Количество проданных товаров по категориям', 'category':'Категория'})
fig.update_layout(yaxis={'categoryorder':'total ascending'})
fig.show()
```




По числу проданных товаров в лидерах растения(3172), муляжи(2093) и товары для дома(1632). Меньше всего было продано товаров для хранения(464), сумки(381) и товары для спальни(65).

### Проверка сезонности продаж по пяти самым продаваемым категориям

In [30]:
```
# Создадим функцию для построения визуализаций
def seasonal_visualization(dataframe, categories):
    # Выделение месяца и года
    dataframe['year_month'] = pd.to_datetime(df['date']).dt.strftime("%Y-%m")
    # Создание распределения количества продаж по выбранным категориям и времени
    sales_distribution = dataframe[dataframe['category'].isin(categories)].groupby(['year_month','category'])['quantity'].sum().reset_index()

    # Построение распределений с использованием библиотеки seaborn
    fig = px.bar(sales_distribution,
                 x='year_month',
                 y='quantity',
                 color='category',
                 title='Распределение количества проданных товаров по времени',
                 labels={'quantity':'Количество проданных товаров', 'year_month':'Дата'},
                 barmode='group')
    fig.show()

# Применим функцию
selected_categories = ['растения', 'муляжи', 'товары для дома', 'кухонная утварь', 'рассада']
seasonal_visualization(df, selected_categories)
```

```
C:\Users\dimch\AppData\Local\Temp\ipykernel_13212\407822895.py:4: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```


Наблюдается сезонность у категорий растения и рассада. Рассаду покупают к началу дачного сезона в апреле и мае, а сезон покупки растений с марта по июнь, в июне пик покупки растений.

### Средняя цена продукта по категориям

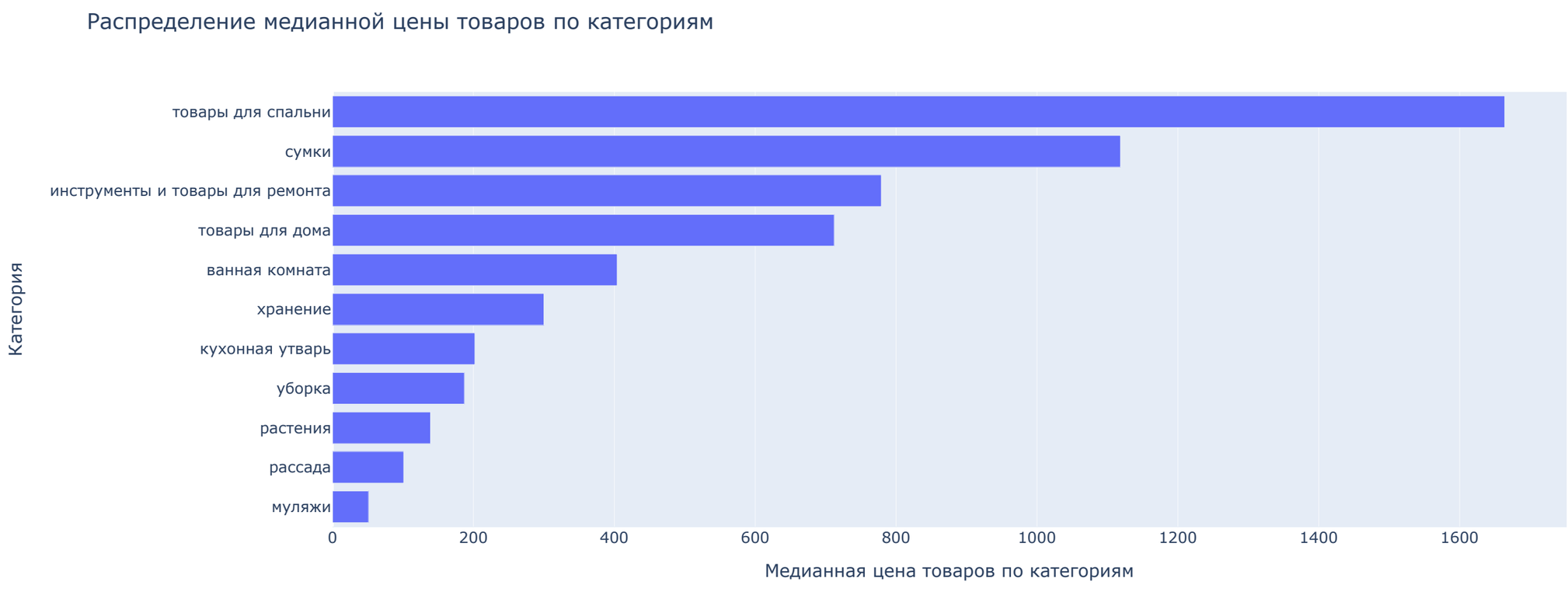
In [31]:
```
# Посчитаем медианную цену товаров по категориям
avg_price = df.pivot_table(index='category',
                           values='price',
                           aggfunc='median').sort_values(by='price').reset_index()
avg_price
```

Out[31]:

| | category | price |
|---|---|---|
| **0** | муляжи | 51.0 |
| **1** | рассада | 101.0 |
| **2** | растения | 139.0 |
| **3** | уборка | 187.0 |
| **4** | кухонная утварь | 202.0 |
| **5** | хранение | 300.0 |
| **6** | ванная комната | 404.0 |
| **7** | товары для дома | 712.0 |
| **8** | инструменты и товары для ремонта | 779.0 |
| **9** | сумки | 1118.5 |
| **10** | товары для спальни | 1664.0 |

In [32]:
```
# Визуализируем распределение медианной цены товаров по категориям
fig = px.bar(avg_price,
             x='price',
```

```
            y='category',
            title='Распределение медианной цены товаров по категориям',
            labels={'price':'Медианная цена товаров по категориям', 'category':'Категория'})
fig.show()
```




Самые дорогие товары в категориях товары для дома, сумки и инструменты и товары для ремонта. Дешевле всего в среднем обходяться растения, рассада и муляжи.

**Вывод:**

- Большинство пользователей совершило лишь один заказ(1407), два заказа совершило 832 клиента, больше двух заказов всего у нескольких десятков;
- Больше всего заказов пришлось на январь 2020(269), февраль 2019(266) и декабрь 2018(265),меньше всего в июне 2019(148);
- Больше всего транзакций пришлось на апрель(655) и май(651) 2019, в январе 2019(170) их было меньше всего;
- Было выделено 11 категорий товаров;
- Самая крупная категория- растения;
- Больше всего денег приносят товары категорий растения(1,04 млн), товары для дома(905,67 тыс) и сумки(739,82 тыс);
- Самые неприбыльными оказались рассада(121,03 тыс), товары для спальни(110,16 тыс) и муляжи(98,24 тыс);
- По числу проданных товаров в лидерах растения(3172), муляжи(2093) и товары для дома(1632);
- Меньше всего было продано товаров для хранения(464), сумки(381) и товары для спальни(65);
- Наблюдается сезонность у категорий растения и рассада. Рассаду покупают к началу дачного сезона в апреле и мае, а сезон покупки растений с марта по июнь, в июне пик покупки растений.

## Шаг. Сегментация пользователей

Для сегментации пользователей используем RFM-анализ, для этого сначала нужно категоризировать пользователей по тратам, частоте заказов и дате последнего заказа.

### Категоризация пользователей по тратам

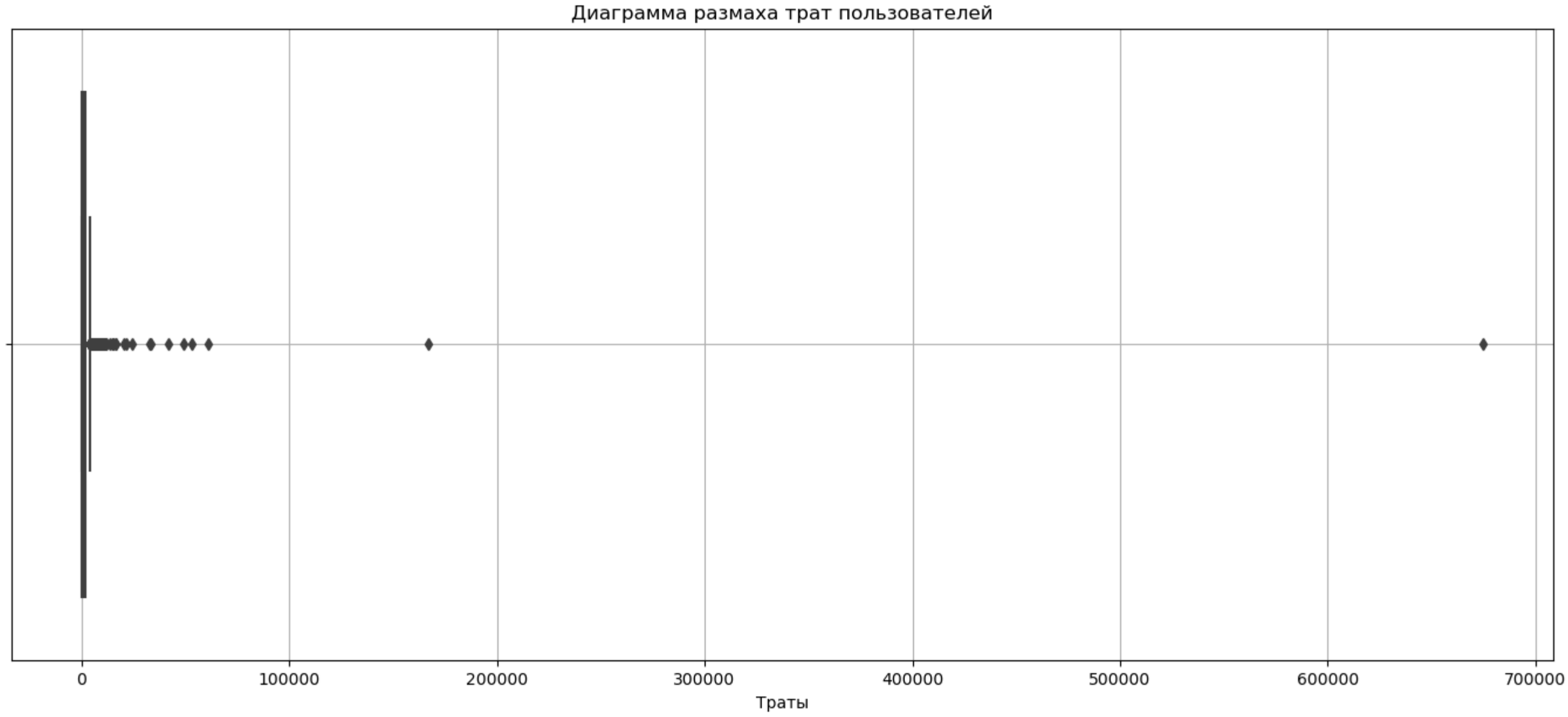
In [33]:
```
# Посчитаем траты по пользователям
user_spending = df.pivot_table(index='customer_id',
                               values='revenue',
                               aggfunc='sum').sort_values(by='revenue',
                                                          ascending=False).reset_index().rename(columns={'revenue':'spending'})
user_spending
```

Out[33]:

| | customer_id | spending |
|---|---|---|
| **0** | 312e9a3e-5fca-43ff-a6a1-892d2b2d5ba6 | 675000.0 |
| **1** | c971fb21-d54c-4134-938f-16b62ee86d3b | 166988.0 |
| **2** | 4d93d3f6-8b24-403b-a74b-f5173e40d7db | 60828.0 |
| **3** | 58a4c3cc-504f-43ea-a74a-bae19e665552 | 53232.0 |
| **4** | 146cd9bf-a95c-4afb-915b-5f6684b17444 | 49432.0 |
| **...** | ... | ... |
| **2260** | 2b6439c9-1ae1-4785-9509-ca4348b3d39a | 22.0 |
| **2261** | 10f79846-2640-4c43-8392-4e76ff5455ef | 22.0 |
| **2262** | f420bf6d-9985-47bc-95a7-5c640ad6d001 | 22.0 |
| **2263** | 9777b839-4212-41bb-94c2-87de3658248a | 15.0 |
| **2264** | 2330d859-e9cb-4c8f-abd0-55f9e27e6745 | 15.0 |

2265 rows × 2 columns

In [34]:
```
# Посмотрим на диаграмме размаха распределение трат
plt.figure(figsize=(17, 7))
ax = sns.boxplot(x=user_spending['spending'])
plt.title('Диаграмма размаха трат пользователей')
plt.xlabel('Траты')
plt.grid()
plt.show()
```



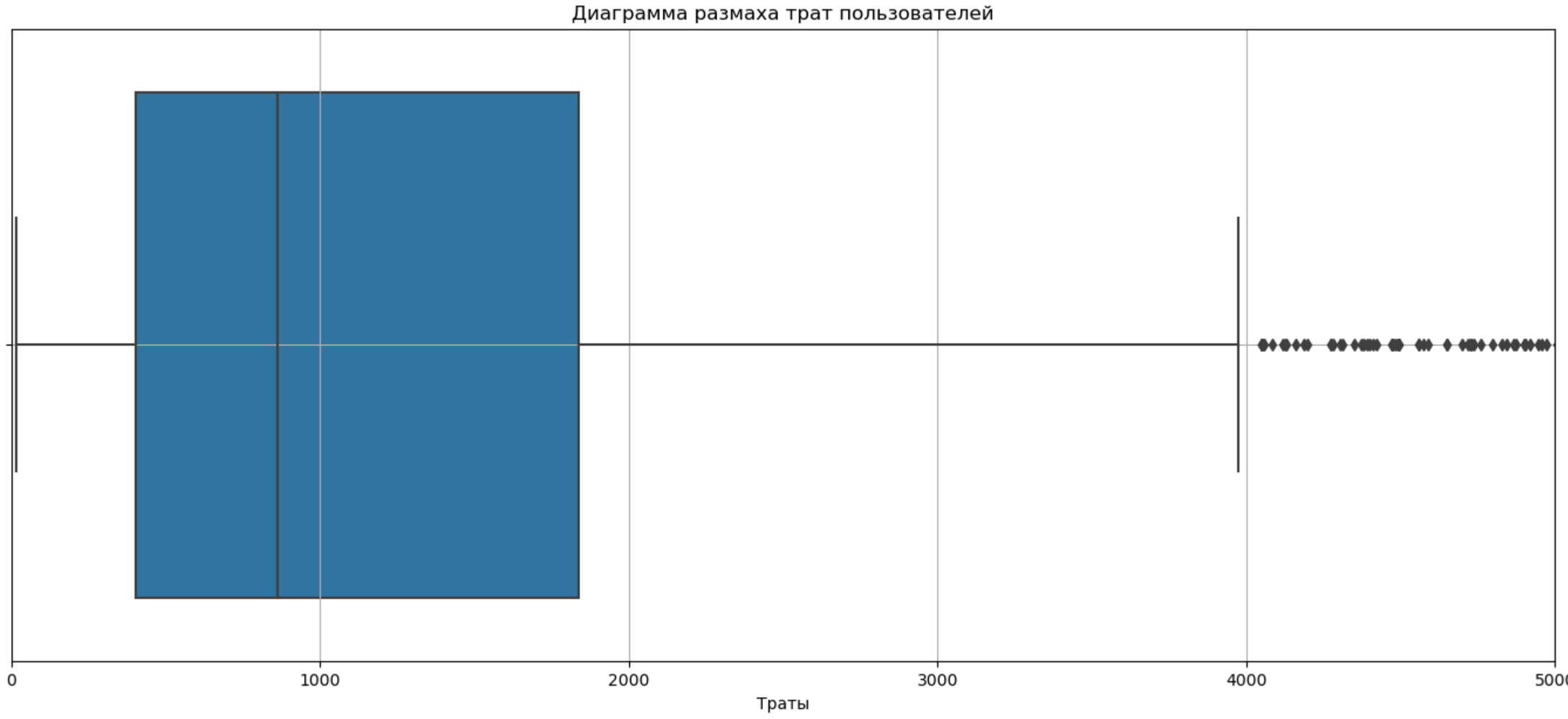

In [35]:
```
user_spending.describe()
```

Out[35]:

| | spending |
|---|---|
| **count** | 2265.000000 |
| **mean** | 1980.865526 |
| **std** | 14876.258666 |
| **min** | 15.000000 |
| **25%** | 404.000000 |
| **50%** | 861.000000 |
| **75%** | 1837.000000 |
| **max** | 675000.000000 |

Видно, что в основном пользователи тратят сильно меньше 10000. Необходимо изменить масштаб диаграммы, чтобы лучше рассмотреть.

In [36]:
```
plt.figure(figsize=(17, 7))
ax = sns.boxplot(x=user_spending['spending'])
plt.title('Диаграмма размаха трат пользователей')
plt.xlabel('Траты')
plt.grid()
plt.xlim(0, 5000)
plt.show()
```




Разделим пользователей по тратам на 3 сегмента:1- 15-845,5; 2- 845,59-1799;3- больше 1799.

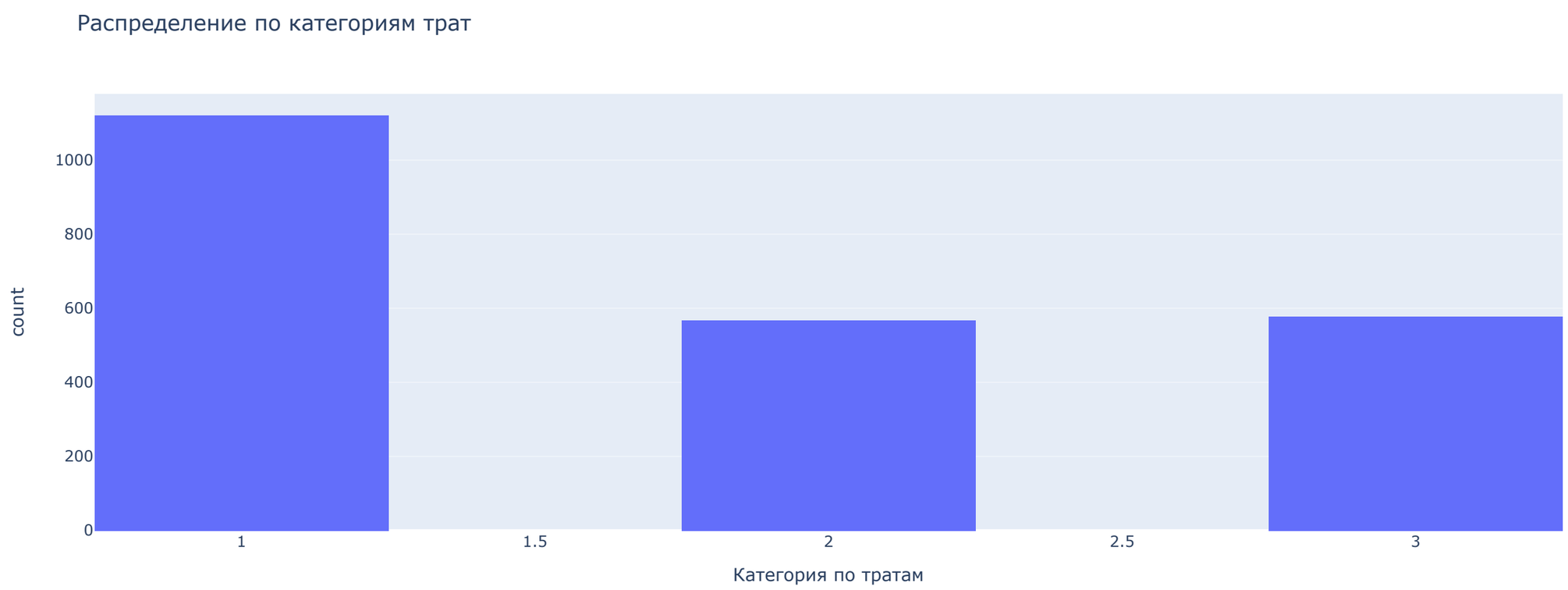
In [37]:
```
# Создадим функцию для определения категории по тратам
def categorize_spending(x):
    if x <= 845.5:
        return 1
    elif x <= 1799:
        return 2
    else:
        return 3

# добавляем столбец с категориями
user_spending['category_spending'] = user_spending['spending'].apply(categorize_spending)
user_spending.head(20)
```

Out[37]:

| | customer_id | spending | category_spending |
|---|---|---|---|
| **0** | 312e9a3e-5fca-43ff-a6a1-892d2b2d5ba6 | 675000.0 | 3 |
| **1** | c971fb21-d54c-4134-938f-16b62ee86d3b | 166988.0 | 3 |
| **2** | 4d93d3f6-8b24-403b-a74b-f5173e40d7db | 60828.0 | 3 |
| **3** | 58a4c3cc-504f-43ea-a74a-bae19e665552 | 53232.0 | 3 |
| **4** | 146cd9bf-a95c-4afb-915b-5f6684b17444 | 49432.0 | 3 |
| **5** | 498f12a4-6a62-4725-8516-cf5dc9ab8a3a | 41900.0 | 3 |
| **6** | 8fba3604-ef57-4b9f-b2fe-3402fa8825c8 | 33680.0 | 3 |
| **7** | 6987e6d6-a63a-4ce2-a2d0-f424092d235e | 32718.0 | 3 |
| **8** | 1b2764ad-9151-4051-a46a-9b01b62e6335 | 24370.0 | 3 |
| **9** | 73d1cd35-5e5f-4629-8cf2-3fda829d4e58 | 21713.0 | 3 |
| **10** | 940c175f-ea87-44e0-9e16-0a3d0a9abecd | 20232.0 | 3 |
| **11** | f279d50f-a508-40b4-bde5-5cb4a1be3ad0 | 16557.0 | 3 |
| **12** | 909564b8-3a5c-4d3e-8310-5ba1c837bbd7 | 16536.0 | 3 |
| **13** | 5d189e88-d4d6-4eac-ab43-fa65a3c4d106 | 15300.0 | 3 |
| **14** | 0d87f4ae-465a-4fac-81e6-5d629761783e | 14917.0 | 3 |
| **15** | ad66d870-22f5-43bc-958f-73420822586b | 13731.0 | 3 |
| **16** | 639c4989-b0ab-412a-b7ec-be394cb2d372 | 12095.0 | 3 |
| **17** | 86c97bf1-c834-423e-9e38-8acda68f97e8 | 11548.0 | 3 |
| **18** | a9089b7e-e6a5-48f9-9b76-48693b63a092 | 11495.0 | 3 |
| **19** | 6be74251-7159-4cc0-99fb-d034a17c61b0 | 11250.0 | 3 |

In [38]:
```
# Визуализируем распределение пользователей по категориям трат
fig = px.histogram(user_spending,
             x='category_spending',
             title='Распределение по категориям трат',
             labels={'category_spending':'Категория по тратам'},
             nbins=5)
fig.show()
```




По гистограмме видно, что больше всего пользователей с низкими тратами, примерно поровну пользователей со средними и высокими тратами.

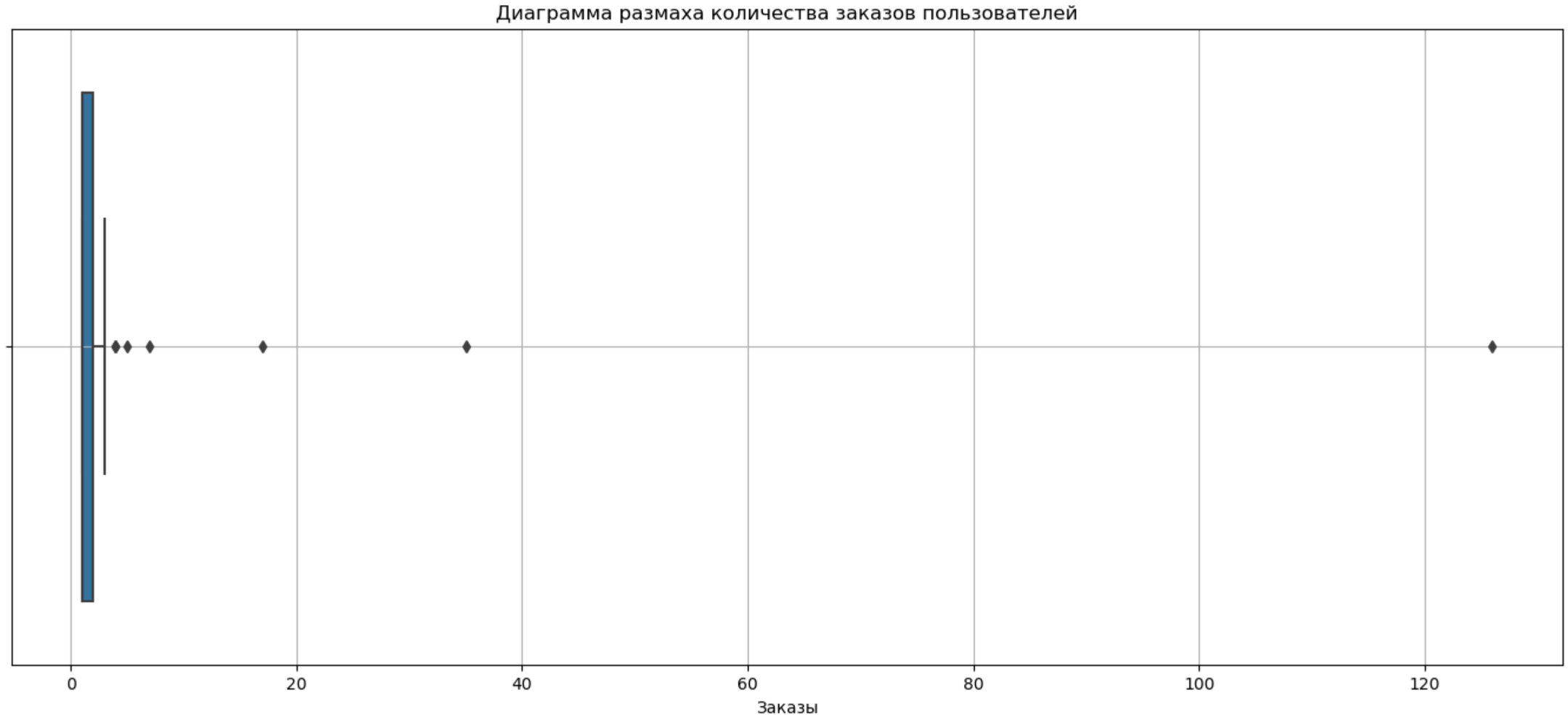
### Категоризация пользователей по количеству заказов

In [39]:
```
# Посчитаем траты по пользователям
user_orders = df.pivot_table(index='customer_id',
                             values='order_id',
                             aggfunc='nunique').sort_values(by='order_id',
                                                            ascending=False).reset_index().rename(columns={'order_id':'orders_count'})
user_orders.head(10)
```

Out[39]:

| | customer_id | orders_count |
|---|---|---|
| **0** | c971fb21-d54c-4134-938f-16b62ee86d3b | 126 |
| **1** | 4d93d3f6-8b24-403b-a74b-f5173e40d7db | 35 |
| **2** | 73d1cd35-5e5f-4629-8cf2-3fda829d4e58 | 17 |
| **3** | b7b865ab-0735-407f-8d0c-31f74d2806cc | 7 |
| **4** | 0184f535-b60a-4914-a982-231e3f615206 | 5 |
| **5** | 498f12a4-6a62-4725-8516-cf5dc9ab8a3a | 4 |
| **6** | e0535076-6270-4df2-8621-cb06264a94fa | 4 |
| **7** | bea7a833-2074-42db-bc49-4457abd3c930 | 4 |
| **8** | 552e17df-ba16-4e66-84fb-55a5557a6bea | 3 |
| **9** | 62952c5b-e5ef-4009-a2f9-1ebff401c514 | 3 |

In [40]:
```
# Посмотрим на диаграмме размаха распределение трат
plt.figure(figsize=(17, 7))
ax = sns.boxplot(x=user_orders['orders_count'])
plt.title('Диаграмма размаха количества заказов пользователей')
plt.xlabel('Заказы')
plt.grid()
plt.show()
```



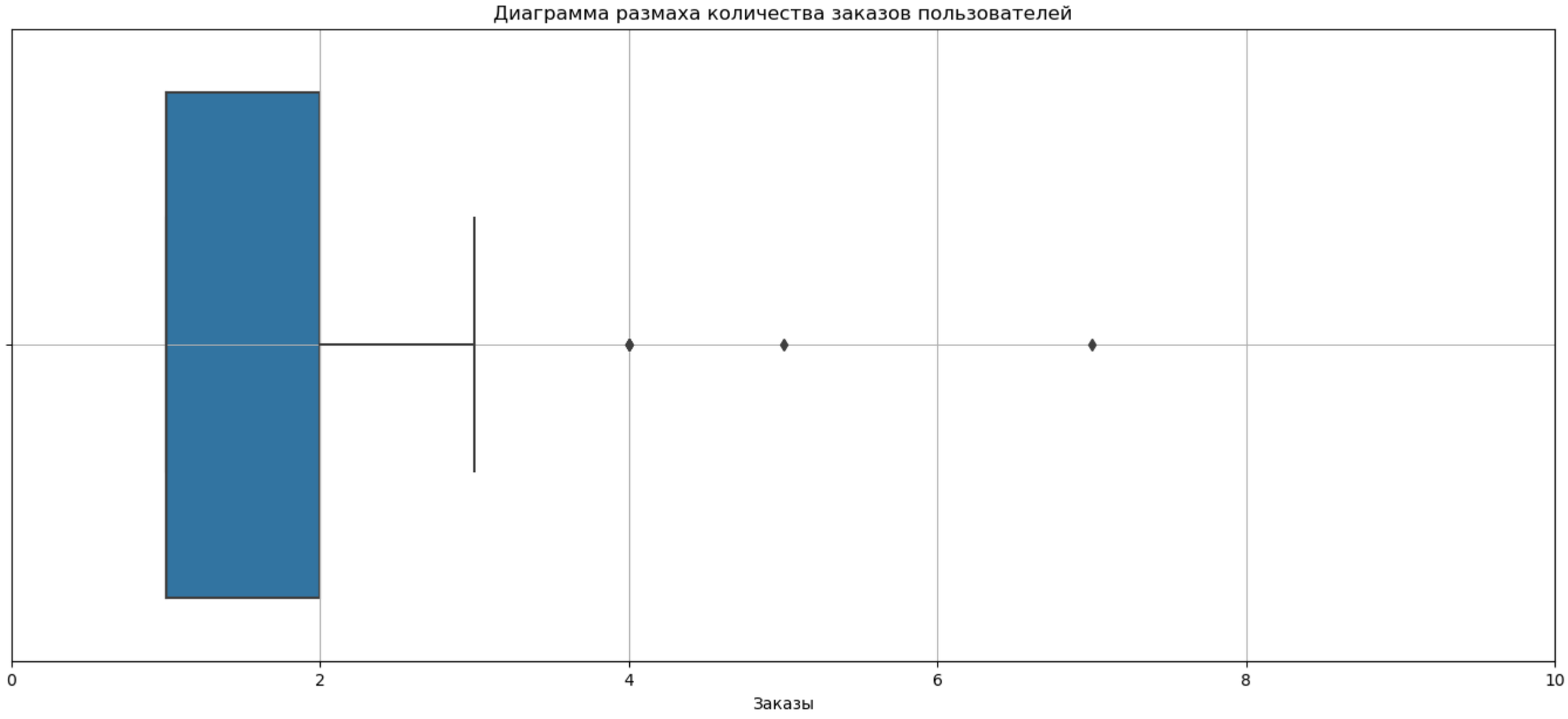

In [41]:
```
user_orders.describe()
```

Out[41]:

| | orders_count |
|---|---|
| **count** | 2265.000000 |
| **mean** | 1.468874 |
| **std** | 2.781982 |
| **min** | 1.000000 |
| **25%** | 1.000000 |
| **50%** | 1.000000 |
| **75%** | 2.000000 |
| **max** | 126.000000 |

Большинство пользователей совершили лишь один заказ, также изменим масштаб диаграммы, чтобы лучше рассмотреть.

In [42]:
```
plt.figure(figsize=(17, 7))
ax = sns.boxplot(x=user_orders['orders_count'])
plt.title('Диаграмма размаха количества заказов пользователей')
plt.xlabel('Заказы')
plt.grid()
plt.xlim(0, 10)
plt.show()
```




Также разделим пользователей на 3 сегмента: 1- 1 заказ; 2- 2 заказа; 3- больше 2 заказов.

In [43]:
```
# Создадим функцию для определения категории по количеству заказов пользователей
def categorize_orders(x):
    if x == 1:
        return 1
    elif x == 2:
        return 2
    else:
        return 3

# добавляем столбец с категориями
user_orders['category_orders'] = user_orders['orders_count'].apply(categorize_orders)
user_orders.head(30)
```

Out[43]:

| | customer_id | orders_count | category_orders |
|---|---|---|---|
| **0** | c971fb21-d54c-4134-938f-16b62ee86d3b | 126 | 3 |
| **1** | 4d93d3f6-8b24-403b-a74b-f5173e40d7db | 35 | 3 |
| **2** | 73d1cd35-5e5f-4629-8cf2-3fda829d4e58 | 17 | 3 |
| **3** | b7b865ab-0735-407f-8d0c-31f74d2806cc | 7 | 3 |
| **4** | 0184f535-b60a-4914-a982-231e3f615206 | 5 | 3 |
| **5** | 498f12a4-6a62-4725-8516-cf5dc9ab8a3a | 4 | 3 |
| **6** | e0535076-6270-4df2-8621-cb06264a94fa | 4 | 3 |
| **7** | bea7a833-2074-42db-bc49-4457abd3c930 | 4 | 3 |
| **8** | 552e17df-ba16-4e66-84fb-55a5557a6bea | 3 | 3 |
| **9** | 62952c5b-e5ef-4009-a2f9-1ebff401c514 | 3 | 3 |
| **10** | d02429ab-22e0-4ff2-9465-3082befde444 | 3 | 3 |
| **11** | eb6521ae-56e3-4a72-9ea2-e9c69701ff3f | 3 | 3 |
| **12** | d16fbc13-50a6-4dea-aafc-bc197aafc9e4 | 3 | 3 |
| **13** | 4856a2a7-b9d2-4243-b8d9-a96ec1425bbe | 3 | 3 |
| **14** | f163e581-59ba-4022-99db-e0973c7497c0 | 3 | 3 |
| **15** | 41117d9d-94f7-4145-a8c9-cb6675ce7674 | 3 | 3 |
| **16** | 639c4989-b0ab-412a-b7ec-be394cb2d372 | 3 | 3 |
| **17** | e8204583-4d55-4724-ad3f-049c7db43bdd | 3 | 3 |
| **18** | 7d64b4ea-d03f-4c3a-b283-21b3d0d237f1 | 3 | 3 |
| **19** | 5d566073-92e8-41d1-a2e6-d301ee5ab6d8 | 3 | 3 |
| **20** | ff422162-fc4a-4b65-a0e2-17f5095ea2c6 | 3 | 3 |
| **21** | dfbcfde5-21de-4504-aff4-453e617d81c1 | 3 | 3 |
| **22** | a9089b7e-e6a5-48f9-9b76-48693b63a092 | 3 | 3 |
| **23** | 0d1b15b6-9cf3-4642-8bc3-74c7dee7b40e | 3 | 3 |
| **24** | 7d0641a6-e043-487d-b356-38895fe7df84 | 3 | 3 |
| **25** | 6b0c6cfb-7717-4c34-8535-bbc6e2b2c758 | 3 | 3 |
| **26** | d31b819d-1dfc-42dc-83f3-52551313fdc5 | 2 | 2 |
| **27** | d35119ba-71ea-4c01-bb80-1e936c2ce0f7 | 2 | 2 |
| **28** | 4a7fe822-c617-460a-a477-33be987babbf | 2 | 2 |
| **29** | 4a6df0eb-6675-4285-a121-3db91e6bb02b | 2 | 2 |

In [44]:
```
# Визуализируем распределение пользователей по категориям количества заказов пользователей
fig = px.histogram(user_orders,
             x='category_orders',
             title='Распределение по категориям количества заказов пользователей',
             labels={'category_orders':'Категория по количеству заказов'},
             nbins=5)
fig.show()
```


Как видно по гистограмме больше всего пользователей, совершивших лишь один заказ(1333), два заказа совершило почти в два раза меньше(826), от трёх и больше лишь 26 человек.

### Категоризация пользователей по давности последнего заказа

In [45]:
```
# Посчитаем траты по пользователям
user_last_order = df.pivot_table(index='customer_id',
                                 values='date',
                                 aggfunc='max').sort_values(by='date',
                                                            ascending=False).reset_index().rename(columns={'date':'last_order'})
user_last_order.head(10)
```

Out[45]:

| | customer_id | last_order |
|---|---|---|
| **0** | 2e460a26-35af-453d-a369-a036e95a40e0 | 2020-01-31 |
| **1** | cdd17932-623e-415f-a577-3b31312fd0e2 | 2020-01-31 |
| **2** | 2c9bd08d-8c55-4e7a-9bfb-8c56ba42c6d6 | 2020-01-31 |
| **3** | d99d25f1-4017-4fcd-8d29-c580cc695a1a | 2020-01-30 |
| **4** | 63208953-a8e4-4f77-9b47-3a46e7b72eee | 2020-01-30 |
| **5** | 370ed405-57f6-4eff-ab2e-a0bacab6e982 | 2020-01-30 |
| **6** | 4228e34b-dcba-4df8-ae70-b282e84a1edb | 2020-01-29 |
| **7** | 0b2157e5-101e-4e0e-bfaf-7340ed23e574 | 2020-01-29 |
| **8** | 28437f82-c2a8-41ea-a7c1-bcedece59d8b | 2020-01-29 |
| **9** | 9777b839-4212-41bb-94c2-87de3658248a | 2020-01-29 |

Разделим на 3 категории: 1- больше двух месяцев назад; 2- до двух месяцев; 3- до месяца. Будем считать, что данные у нас актуальные, т.е. считать от 31 января 2020 года, за месяц возьмём 30 дней.

In [46]:
```
# Определение даты референса 30.10.2020
referencedate = df['date'].max()

# Создадим функцию категории для каждой строки
def categorizedate(last_date):
    timedifference = referencedate - last_date
```

```
    if timedifference.days <= 30:
        return 3
    elif timedifference.days <= 60:
        return 2
    else:
        return 1

# Добавление столбца с категориями
user_last_order['category_last_order'] = user_last_order['last_order'].apply(categorizedate)
user_last_order
```

Out[46]:

| | customer_id | last_order | category_last_order |
|---|---|---|---|
| **0** | 2e460a26-35af-453d-a369-a036e95a40e0 | 2020-01-31 | 3 |
| **1** | cdd17932-623e-415f-a577-3b31312fd0e2 | 2020-01-31 | 3 |
| **2** | 2c9bd08d-8c55-4e7a-9bfb-8c56ba42c6d6 | 2020-01-31 | 3 |
| **3** | d99d25f1-4017-4fcd-8d29-c580cc695a1a | 2020-01-30 | 3 |
| **4** | 63208953-a8e4-4f77-9b47-3a46e7b72eee | 2020-01-30 | 3 |
| **...** | ... | ... | ... |
| **2260** | 649c4398-a4c4-4d01-b29b-f14704ce6e17 | 2018-10-02 | 1 |
| **2261** | 161e1b98-45ba-4b4e-8236-e6e3e70f6f7c | 2018-10-01 | 1 |
| **2262** | ee47d746-6d2f-4d3c-9622-c31412542920 | 2018-10-01 | 1 |
| **2263** | 0948b0c2-990b-4a11-b835-69ac4714b21d | 2018-10-01 | 1 |
| **2264** | 375e0724-f033-4c76-b579-84969cf38ee2 | 2018-10-01 | 1 |

2265 rows × 3 columns

In [47]:
```
# Визуализируем распределение пользователей по категориям последнего заказа пользователей
fig = px.histogram(user_last_order,
             x='category_last_order',
             title='Распределение по категориям последнего заказа пользователей',
             labels={'category_orders':'Категория по последнему заказу пользователей'},
             nbins=5)
fig.show()
```


Большинство заказов относятся к категории 1(1674), в третьей и второй категориях лишь 242 и 269 заказа. Получается, что большинство пользователей сделали свой последний заказ больше двух месяцев назад.

Подытожим разбивку на категории:

**Категоризация пользователей по тратам:**

- 1 категория- низкие траты;
- 2 категория- средние траты;
- 3 категория- высокие траты.

**Категоризация пользователей по количеству заказов:**

- 1 категория- разовые покупки;
- 2 категория- редкие покупки;
- 3 категория- частые покупки.

**Категоризация пользователей по давности последнего заказа:**

- 1 категория- давние заказы;
- 2 категория- не очень давно;
- 3 категория- недавние заказы.

В таком порядке будем обозначать сочетания категорий для сегментации, например, 123-низкие траты, редкие покупки и недавний заказ.

### Итоговая сегментация

В данном пункте сведём полученные по категориям данные, чтобы выделить сегменты пользователей. Выделим идеальных, новых, лояльных, крупных, спящих и потерянных покупателей.

In [48]:
```
# Объединим три полученных датасета с категориями
merged_category = user_spending.merge(user_orders,
                                      on='customer_id',
                                      how='outer').merge(user_last_order,
                                                         on='customer_id',
                                                         how='outer')
merged_category.head(15)
```

Out[48]:

| | customer_id | spending | category_spending | orders_count | category_orders | last_order | category_last_order |
|---|---|---|---|---|---|---|---|
| **0** | 312e9a3e-5fca-43ff-a6a1-892d2b2d5ba6 | 675000.0 | 3 | 1 | 1 | 2019-06-18 | 1 |
| **1** | c971fb21-d54c-4134-938f-16b62ee86d3b | 166988.0 | 3 | 126 | 3 | 2019-03-06 | 1 |
| **2** | 4d93d3f6-8b24-403b-a74b-f5173e40d7db | 60828.0 | 3 | 35 | 3 | 2018-10-24 | 1 |
| **3** | 58a4c3cc-504f-43ea-a74a-bae19e665552 | 53232.0 | 3 | 2 | 2 | 2019-01-15 | 1 |
| **4** | 146cd9bf-a95c-4afb-915b-5f6684b17444 | 49432.0 | 3 | 1 | 1 | 2019-06-11 | 1 |
| **5** | 498f12a4-6a62-4725-8516-cf5dc9ab8a3a | 41900.0 | 3 | 4 | 3 | 2019-04-19 | 1 |
| **6** | 8fba3604-ef57-4b9f-b2fe-3402fa8825c8 | 33680.0 | 3 | 1 | 1 | 2018-11-29 | 1 |
| **7** | 6987e6d6-a63a-4ce2-a2d0-f424092d235e | 32718.0 | 3 | 1 | 1 | 2018-12-21 | 1 |
| **8** | 1b2764ad-9151-4051-a46a-9b01b62e6335 | 24370.0 | 3 | 1 | 1 | 2018-11-06 | 1 |
| **9** | 73d1cd35-5e5f-4629-8cf2-3fda829d4e58 | 21713.0 | 3 | 17 | 3 | 2019-10-31 | 1 |
| **10** | 940c175f-ea87-44e0-9e16-0a3d0a9abecd | 20232.0 | 3 | 2 | 2 | 2019-06-12 | 1 |
| **11** | f279d50f-a508-40b4-bde5-5cb4a1be3ad0 | 16557.0 | 3 | 2 | 2 | 2019-12-31 | 2 |
| **12** | 909564b8-3a5c-4d3e-8310-5ba1c837bbd7 | 16536.0 | 3 | 1 | 1 | 2019-02-04 | 1 |
| **13** | 5d189e88-d4d6-4eac-ab43-fa65a3c4d106 | 15300.0 | 3 | 1 | 1 | 2019-05-20 | 1 |
| **14** | 0d87f4ae-465a-4fac-81e6-5d629761783e | 14917.0 | 3 | 1 | 1 | 2019-07-29 | 1 |

- ***Идеальные покупатели:*** заказывают часто, тратят много, последний заказ недавно(333).
- ***Новые покупатели:*** один недавний заказ(113, 213, 313).
- ***Лояльные покупатели:*** не разовые заказы, последний недавно(122, 123, 132, 133, 222, 232, 233, 223).
- ***Крупные покупатели:*** тратят много, но нерегулярно(311, 312, 322, 323, 332).
- ***Спящие покупатели:*** не разовые заказы больше двух месяцев назад(112, 121, 131, 212, 221, 231, 321, 331).
- ***Потерянные покупатели:*** давно и немного заказывали(111, 211).

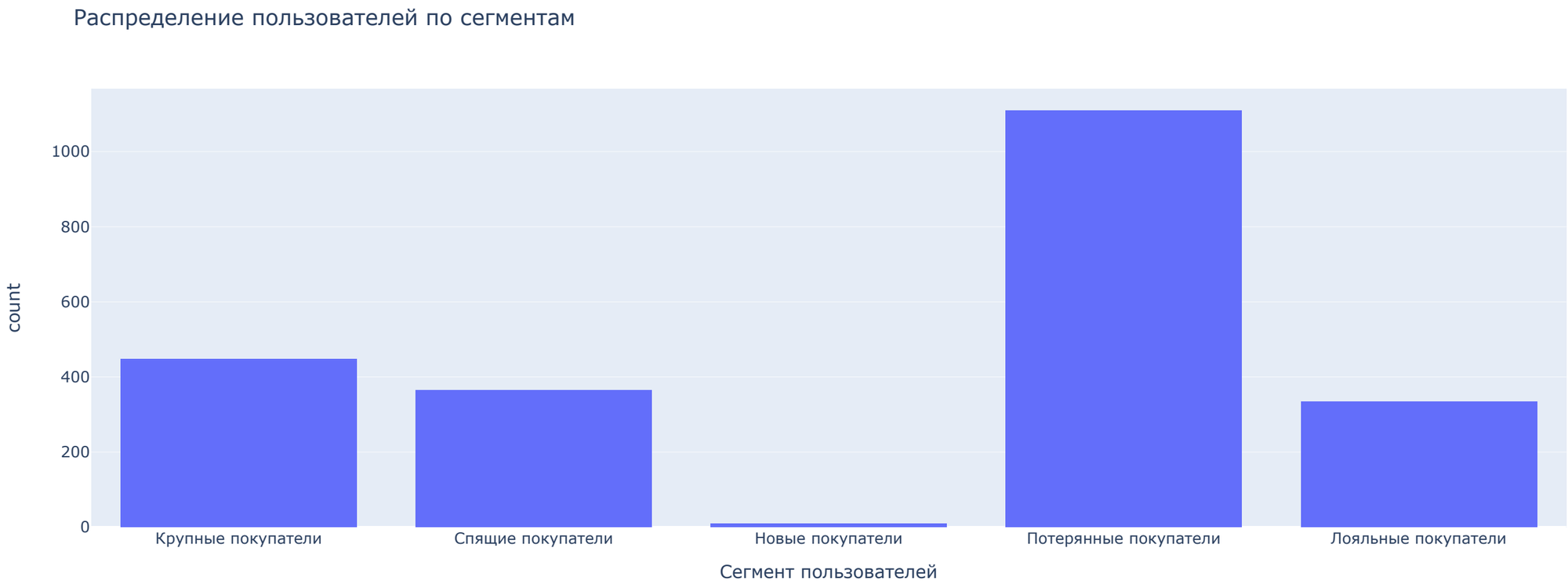
In [49]:
```
# Создадим функцию для определения сегмента пользователя
def segment_users(x):
    if x['category_spending'] == 3 and x['category_orders'] == 3 and x['category_last_order'] == 3:
        return 'Идеальные покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 1 and x['category_last_order'] == 3:
        return 'Новые покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 1 and x['category_last_order'] == 3:
        return 'Новые покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 1 and x['category_last_order'] == 3:
        return 'Новые покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 2 and x['category_last_order'] == 2:
        return 'Лояльные покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 2 and x['category_last_order'] == 3:
        return 'Лояльные покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 3 and x['category_last_order'] == 2:
        return 'Лояльные покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 3 and x['category_last_order'] == 3:
        return 'Лояльные покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 2 and x['category_last_order'] == 2:
        return 'Лояльные покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 3 and x['category_last_order'] == 2:
        return 'Лояльные покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 3 and x['category_last_order'] == 3:
        return 'Лояльные покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 2 and x['category_last_order'] == 3:
        return 'Лояльные покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 1 and x['category_last_order'] == 1:
        return 'Крупные покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 1 and x['category_last_order'] == 2:
        return 'Крупные покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 1 and x['category_last_order'] == 3:
        return 'Крупные покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 2 and x['category_last_order'] == 2:
        return 'Крупные покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 3 and x['category_last_order'] == 2:
        return 'Крупные покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 2 and x['category_last_order'] == 3:
        return 'Крупные покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 1 and x['category_last_order'] == 2:
        return 'Спящие покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 2 and x['category_last_order'] == 1:
        return 'Спящие покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 3 and x['category_last_order'] == 1:
        return 'Спящие покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 1 and x['category_last_order'] == 2:
        return 'Спящие покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 2 and x['category_last_order'] == 1:
        return 'Спящие покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 3 and x['category_last_order'] == 1:
        return 'Спящие покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 2 and x['category_last_order'] == 1:
        return 'Спящие покупатели'
    elif x['category_spending'] == 3 and x['category_orders'] == 3 and x['category_last_order'] == 1:
        return 'Спящие покупатели'
    elif x['category_spending'] == 1 and x['category_orders'] == 1 and x['category_last_order'] == 1:
        return 'Потерянные покупатели'
    elif x['category_spending'] == 2 and x['category_orders'] == 1 and x['category_last_order'] == 1:
        return 'Потерянные покупатели'
    else:
        return 'segment_N'

# Добавим столбец с сегментами пользователей
merged_category['segment'] = merged_category.apply(segment_users, axis=1)
merged_category.head(10)
```

Out[49]:

| | customer_id | spending | category_spending | orders_count | category_orders | last_order | category_last_order | segment |
|---|---|---|---|---|---|---|---|---|
| **0** | 312e9a3e-5fca-43ff-a6a1-892d2b2d5ba6 | 675000.0 | 3 | 1 | 1 | 2019-06-18 | 1 | Крупные покупатели |
| **1** | c971fb21-d54c-4134-938f-16b62ee86d3b | 166988.0 | 3 | 126 | 3 | 2019-03-06 | 1 | Спящие покупатели |
| **2** | 4d93d3f6-8b24-403b-a74b-f5173e40d7db | 60828.0 | 3 | 35 | 3 | 2018-10-24 | 1 | Спящие покупатели |
| **3** | 58a4c3cc-504f-43ea-a74a-bae19e665552 | 53232.0 | 3 | 2 | 2 | 2019-01-15 | 1 | Спящие покупатели |
| **4** | 146cd9bf-a95c-4afb-915b-5f6684b17444 | 49432.0 | 3 | 1 | 1 | 2019-06-11 | 1 | Крупные покупатели |
| **5** | 498f12a4-6a62-4725-8516-cf5dc9ab8a3a | 41900.0 | 3 | 4 | 3 | 2019-04-19 | 1 | Спящие покупатели |
| **6** | 8fba3604-ef57-4b9f-b2fe-3402fa8825c8 | 33680.0 | 3 | 1 | 1 | 2018-11-29 | 1 | Крупные покупатели |
| **7** | 6987e6d6-a63a-4ce2-a2d0-f424092d235e | 32718.0 | 3 | 1 | 1 | 2018-12-21 | 1 | Крупные покупатели |
| **8** | 1b2764ad-9151-4051-a46a-9b01b62e6335 | 24370.0 | 3 | 1 | 1 | 2018-11-06 | 1 | Крупные покупатели |
| **9** | 73d1cd35-5e5f-4629-8cf2-3fda829d4e58 | 21713.0 | 3 | 17 | 3 | 2019-10-31 | 1 | Спящие покупатели |

In [50]:
```
# Визуализируем распределение пользователей по сегментам
fig = px.histogram(merged_category,
             x='segment',
             title='Распределение пользователей по сегментам',
             labels={'segment':'Сегмент пользователей'},
             nbins=5)
fig.show()
```




Новых покупателей всего 9, а потерянных 1068, на втором месте крупные(415), спящие(359) и лояльные(334) примерно на одном уровне.

**Вывод:** Для сегментации клиентов был использован RFM-анализ, позволяющий сегментировать клиентов по частоте и сумме покупок и выявлять тех, которые приносят больше денег.

Можно сформулировать рекомендации для работы с каждым из сегментов для стимуляции к новым покупкам.

- ***Идеальные покупатели:*** к сожалению, таких клиентов у нас нет, но если бы были, то их необходимо было бы удерживать индивидуальными предложениями и условиями обслуживания, важно не надоедать таким клиентам рассылкой, сообщаая только важную информацию.
- ***Новые покупатели:*** у нас 9 таких клиентов, в рассылке можно поздравить их с первым заказом, предложив скидку на следующий или другие выгодные акции, важно добиться их лояльности.
- ***Лояльные покупатели:*** в данном сегменте 334 клиента, можно попробовать повысить их активность, предлагая интересные им категории.
- ***Крупные покупатели:*** второй по величине сегмент с 415 клиентами, таким клиентам можно предложить индивидуальные скидки и предоставлить специальные условий для крупных заказов.
- ***Спящие покупатели:*** в сегменте 359 клиентов, важно вернуть их интерес, можно предложить им принять участие в программе лояльности, предложить акции и скидки на основе их предпочтений, попробовать провести опрос, чтобы узнать причины пропажи интереса.
- ***Потерянные покупатели:*** самый крупный сегмент с 1068 клиентами, можно попробовать узнать причины их ухода и предложить решение возможных проблем, предложить скидки или акции в рассылке.

## Шаг. Проверка статистических гипотез

### Средние траты лояльных клиентов больше, чем у спящих

Сформулируем гипотезу:

- $H_0$ : Средние траты лояльных и спящих клиентов равны.
- $H_1$ : Средние траты лояльных клиентов больше, чем у спящих.

Уровень статистической значимости возьмём 5%.

In [51]:
```
# Выделим траты спящих и лояльных клиентов
loyal_customers = merged_category.query('segment == "Лояльные покупатели"')['spending']
sleeping_customers = merged_category.query('segment == "Спящие покупатели"')['spending']

alpha = 0.05

# Проверим гипотезу
results = st.ttest_ind(loyal_customers, sleeping_customers, alternative='greater', equal_var=False)
print('p-значение:', results.pvalue)
if (results.pvalue < alpha):
    print('Отвергаю нулевую гипотезу')
else:
    print('Не получилось отвергнуть нулевую гипотезу')
```

```
p-значение: 0.9999309349973169
Не получилось отвергнуть нулевую гипотезу
```

*Не получучилось отвергнуть нулевую гипотезу, а это значит, что средние траты лояльных и спящих клиентов одинаковы.*

### Средние траты новых и потерянных клиентов не отличаются

Сформулируем гипотезу:

- $H_0$ : Средние траты новых и потерянных клиентов не отличаются.
- $H_1$ : Средние траты новых и потерянных клиентов отличаются.

Уровень статистической значимости возьмём 5%.

In [52]:
```
# Выделим траты спящих и лояльных клиентов
new_customers = merged_category.query('segment == "Новые покупатели"')['spending']
lost_customers = merged_category.query('segment == "Потерянные покупатели"')['spending']

alpha = 0.05

# Проверим гипотезу
results = st.ttest_ind(new_customers, lost_customers, equal_var=False)
print('p-значение:', results.pvalue)
if (results.pvalue < alpha):
    print('Отвергаю нулевую гипотезу')
else:
    print('Не получилось отвергнуть нулевую гипотезу')
```

```
p-значение: 0.4769855184925792
Не получилось отвергнуть нулевую гипотезу
```

*Не получучилось отвергнуть нулевую гипотезу, а это значит, что средние траты новых и потерянных клиентов не отличаются.*

## Шаг. Итоговый вывод:

Целью данного проекта было узнать больше информации о пользователях, чтобы улучшить показатели интернет-магазина «Пока все ещё тут».

Для этого был проведён исследовательский анализ в ходе, которого мы выяснили:

- Большинство пользователей совершило лишь один заказ(1407), два заказа совершило 832 клиента, больше двух заказов всего у нескольких десятков;
- Больше всего заказов пришлось на январь 2020(269), февраль 2019(266) и декабрь 2018(265),меньше всего в июне 2019(148);
- Больше всего транзакций пришлось на апрель(655) и май(651) 2019, в январе 2019(170) их было меньше всего;
- Было выделено 11 категорий товаров;
- Самая крупная категория- растения;
- Больше всего денег приносят товары категорий растения(1,04 млн), товары для дома(905,67 тыс) и сумки(739,82 тыс);
- Самые неприбыльными оказались рассада(121,03 тыс), товары для спальни(110,16 тыс) и муляжи(98,24 тыс);

- По числу проданных товаров в лидерах растения(3172), муляжи(2093) и товары для дома(1632);
- Меньше всего было продано товаров для хранения(464), сумки(381) и товары для спальни(65);
- Наблюдается сезонность у категорий растения и рассада. Рассаду покупают к началу дачного сезона в апреле и мае, а сезон покупки растений с марта по июнь, в июне пик покупки растений.

Проведена сегментация пользователей при помощи RFM-анализа, позволяющего сегментировать клиентов по частоте и сумме покупок и выявлять тех, которые приносят больше денег. В ходе сегментации были выделены пять сегментов пользователей интернет-магазина «Пока все ещё тут» и даны рекомендации для работы с каждым из сегментов для стимуляции к новым покупкам:

- ***Новые покупатели:*** у нас 9 таких клиентов, в рассылке можно поздравить их с первым заказом, предложив скидку на следующий или другие выгодные акции, важно добиться их лояльности.
- ***Лояльные покупатели:*** в данном сегменте 334 клиента, можно попробовать повысить их активность, предлагая интересные им категории.
- ***Крупные покупатели:*** второй по величине сегмент с 415 клиентами, таким клиентам можно предложить индивидуальные скидки и предоставлить специальные условий для крупных заказов.
- ***Спящие покупатели:*** в сегменте 359 клиентов, важно вернуть их интерес, можно предложить им принять участие в программе лояльности, предложить акции и скидки на основе их предпочтений, попробовать провести опрос, чтобы узнать причины пропажи интереса.
- ***Потерянные покупатели:*** самый крупный сегмент с 1068 клиентами, можно попробовать узнать причины их ухода и предложить решение возможных проблем, предложить скидки или акции в рассылке.

А также была проведена проверка гипотез, из которых следуют следующие выводы:

- Средние траты лояльных и спящих клиентов одинаковы;
- Средние траты новых и потерянных клиентов не отличаются.

***Презентация-*** https://drive.google.com/file/d/1vjUZCdjfv9POMMiG5ZNYE_PabhOsqbFN/view?usp=sharing