Table of Contents

Принятие решений в бизнесе

Вместе с отделом маркетинга подготовлен список гипотез для увеличения выручки.

Необходимо:

- приоритизировать гипотезы,
- запустить A/B-тест

- проанализировать результаты.

Загрузка данных и подготовка их к анализу

Импорт библиотек. Считывание данных из исходных файлов

In [1]:
```
# импортируем библиотеки
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as stats
import datetime as dt
import numpy as np
import seaborn as sns
```

In [2]:
```
# считываем  данные и сохраняем в переменные

hypothesis = pd.read_csv('hypothesis.csv')  # гипотезы
orders = pd.read_csv('orders.csv')  # заказы
visitors = pd.read_csv('visitors.csv')  # пользователи по датам и группам
```

Первичный обзор данных

In [3]:
```
# Функция первичного обзора данных.

def meet_dataset (dataset):
    #print('Первые 5 строк датасета')
    #print(dataset.head())
    #print('\n','\n')

    print('Общая информация о датасете')
    print(dataset.info(),'\n','\n')

    print('Общие статистические данные')
    print(dataset.describe(),'\n','\n')

    print('Общие гистограммы для всех столбцов датасета')
    dataset.hist (figsize=(15,5))
    plt.show()
    print('\n')

    print ('Количество дубликатов -', dataset.duplicated().sum(),'\n')
    print ('Количество пропусков -', dataset.isna().sum(),'\n','\n')
    print ('Датасет:','\n')

    #Приведение названий столбцов к нижнему регистру
    dataset.columns = [x.lower().replace(' ', '_') for x in dataset.columns.values]


    return dataset
```

Обзор датасета с гипотезами

In [4]:
```
meet_dataset(hypothesis)
```

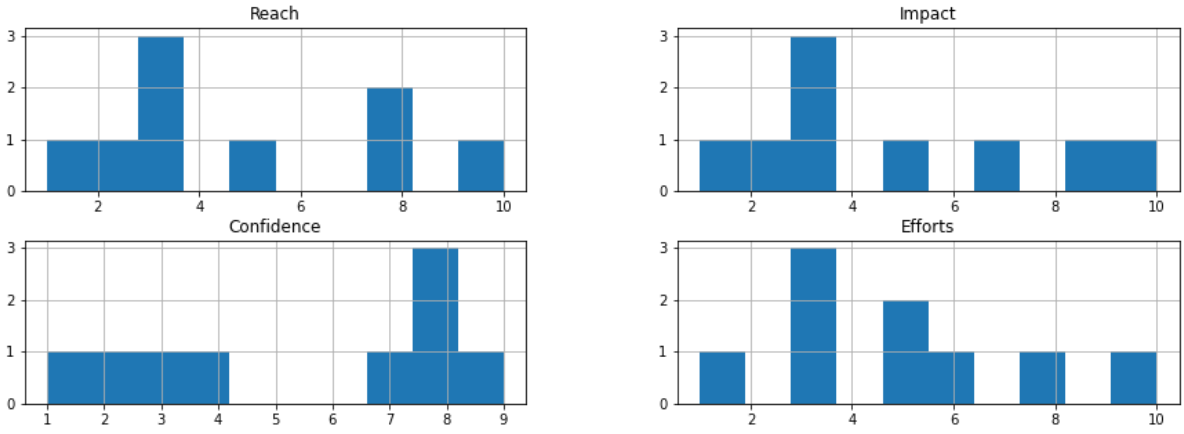
Общая информация о датасете

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Hypothesis  9 non-null      object
 1   Reach       9 non-null      int64 
 2   Impact      9 non-null      int64 
 3   Confidence  9 non-null      int64 
 4   Efforts     9 non-null      int64 
dtypes: int64(4), object(1)
memory usage: 488.0+ bytes
None
```

Общие статистические данные

	Reach	Impact	Confidence	Efforts
count	9.000000	9.000000	9.000000	9.000000
mean	4.777778	4.777778	5.555556	4.888889
std	3.153481	3.192874	3.045944	2.803767
min	1.000000	1.000000	1.000000	1.000000
25%	3.000000	3.000000	3.000000	3.000000
50%	3.000000	3.000000	7.000000	5.000000
75%	8.000000	7.000000	8.000000	6.000000
max	10.000000	10.000000	9.000000	10.000000

Общие гистограммы для всех столбцов датасета



Количество дубликатов - 0

```
Количество пропусков - Hypothesis    0
Reach         0
Impact        0
Confidence    0
Efforts       0
dtype: int64
```

Датасет:

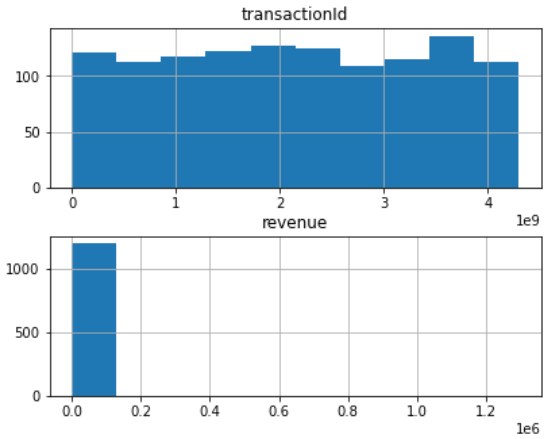
Out[4]:

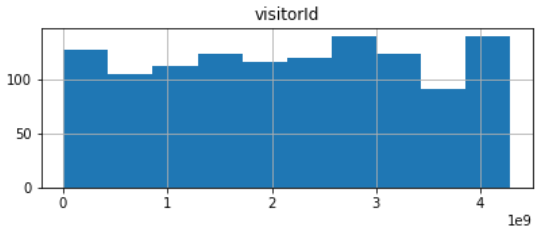
	hypothesis	reach	impact	confidence	efforts
0	Добавить два новых канала привлечения трафика,...	3	10	8	6
1	Запустить собственную службу доставки, что сок...	2	5	4	10
2	Добавить блоки рекомендаций товаров на сайт ин...	8	3	7	3
3	Изменить структура категорий, что увеличит кон...	8	3	3	8
4	Изменить цвет фона главной страницы, чтобы уве...	3	1	1	1
5	Добавить страницу отзывов клиентов о магазине,...	3	2	2	3
6	Показать на главной странице баннеры с актуаль...	5	3	8	3
7	Добавить форму подписки на все основные страни...	10	7	8	5
8	Запустить акцию, дающую скидку на товар в день...	1	9	9	5

Обзор датасета с заказами

In [5]:
```
meet_dataset(orders)
```

```
Общая информация о датасете
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1197 entries, 0 to 1196
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   transactionId  1197 non-null   int64 
 1   visitorId      1197 non-null   int64 
 2   date           1197 non-null   object
 3   revenue        1197 non-null   int64 
 4   group          1197 non-null   object
dtypes: int64(3), object(2)
memory usage: 46.9+ KB
None


Общие статистические данные
       transactionId     visitorId       revenue
count   1.197000e+03  1.197000e+03  1.197000e+03
mean    2.155621e+09  2.165960e+09  8.348006e+03
std     1.229085e+09  1.236014e+09  3.919113e+04
min     1.062393e+06  5.114589e+06  5.000000e+01
25%     1.166776e+09  1.111826e+09  1.220000e+03
50%     2.145194e+09  2.217985e+09  2.978000e+03
75%     3.237740e+09  3.177606e+09  8.290000e+03
max     4.293856e+09  4.283872e+09  1.294500e+06


Общие гистограммы для всех столбцов датасета
```






```
Количество дубликатов - 0

Количество пропусков - transactionId    0
visitorId        0
date             0
revenue          0
group            0
dtype: int64


Датасет:
```

Out[5]:

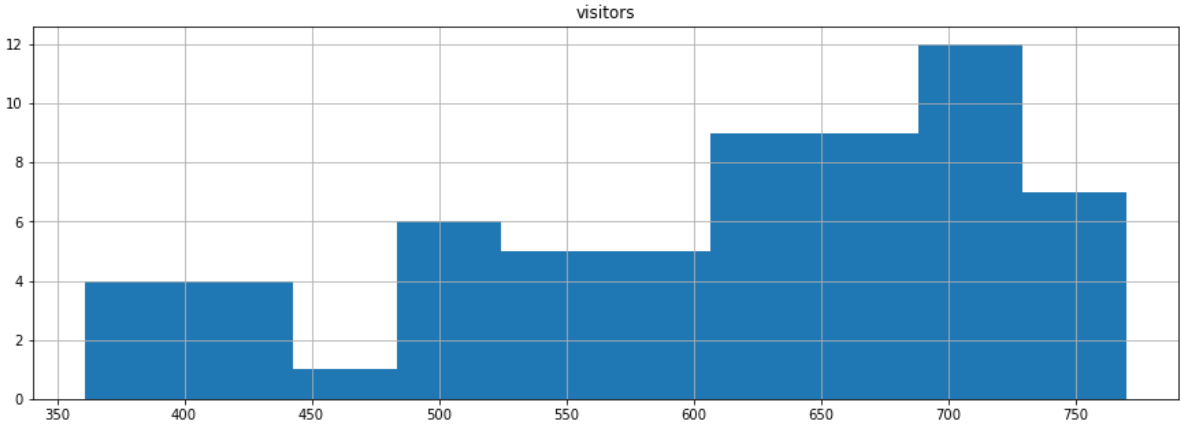
	transactionid	visitorid	date	revenue	group
0	3667963787	3312258926	2019-08-15	1650	B
1	2804400009	3642806036	2019-08-15	730	B
2	2961555356	4069496402	2019-08-15	400	A
3	3797467345	1196621759	2019-08-15	9759	B
4	2282983706	2322279887	2019-08-15	2308	B
...	...	...	...	...	...
1192	2662137336	3733762160	2019-08-14	6490	B
1193	2203539145	370388673	2019-08-14	3190	A
1194	1807773912	573423106	2019-08-14	10550	A
1195	1947021204	1614305549	2019-08-14	100	A
1196	3936777065	2108080724	2019-08-15	202740	B

1197 rows × 5 columns

Обзор датасета с пользователями

In [6]:
```
meet_dataset(visitors)
```

```
Общая информация о датасете
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62 entries, 0 to 61
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   date      62 non-null     object
 1   group     62 non-null     object
 2   visitors  62 non-null     int64 
dtypes: int64(1), object(2)
memory usage: 1.6+ KB
None


Общие статистические данные
         visitors
count   62.000000
mean   607.290323
std    114.400560
min    361.000000
25%    534.000000
50%    624.500000
75%    710.500000
max    770.000000


Общие гистограммы для всех столбцов датасета
```




```
Количество дубликатов - 0

Количество пропусков - date        0
group       0
visitors    0
dtype: int64


Датасет:
```

Out[6]:

	date	group	visitors
0	2019-08-01	A	719
1	2019-08-02	A	619
2	2019-08-03	A	507
3	2019-08-04	A	717
4	2019-08-05	A	756
...	...	...	...
57	2019-08-27	B	720
58	2019-08-28	B	654
59	2019-08-29	B	531
60	2019-08-30	B	490
61	2019-08-31	B	718

62 rows × 3 columns

Описание данных из задания

Гипотезы

Наименования столбцов:

- Hypothesis — краткое описание гипотезы;
- Reach — охват пользователей по 10-балльной шкале;
- Impact — влияние на пользователей по 10-балльной шкале;
- Confidence — уверенность в гипотезе по 10-балльной шкале;
- Efforts — затраты ресурсов на проверку гипотезы по 10-балльной шкале. Чем больше значение Efforts, тем дороже проверка гипотезы.

Заказы

Наименования столбцов:

- transactionId — идентификатор заказа;
- visitorId — идентификатор пользователя, совершившего заказ;
- date — дата, когда был совершён заказ;
- revenue — выручка заказа;
- group — группа A/B-теста, в которую попал заказ.

Пользователи

Наименования столбцов:

- date — дата;
- group — группа A/B-теста;
- visitors — количество пользователей в указанную дату в указанной группе A/B-теста

Комментарии к данным и предобработка

1. В датасетах нет пропусков и дубликатов.
2. В датасетах, в основном, типы данных корректны. Исключение- столбцы с датой в датасетах с гипотезами и заказами. Коррекция формата выполняется ниже.
3. В датасетах гипотез и заказов выполнено приведение наменований столбцов к нижнему регистру.

Приведение данных в столбцах 'date' к формату даты.

In [7]:
```
#orders['date'] = pd.to_datetime(orders['date'], format='%Y-%m-%d')
#visitors['date'] = pd.to_datetime(visitors['date'], format='%Y-%m-%d')

orders['date'] = orders['date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))
visitors['date'] = visitors['date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%
```

Минимальная и максимальная даты в датасете **заказов**:

In [8]:
```
print('Минимальная дата:', orders['date'].min())
print('Максимальная дата:', orders['date'].max(),'\n')
```

```
Минимальная дата: 2019-08-01 00:00:00
Максимальная дата: 2019-08-31 00:00:00
```

Минимальная и максимальная даты в датасете **пользователей**:

In [9]:
```
print('Минимальная дата:', visitors['date'].min())
print('Максимальная дата:', visitors['date'].max(),'\n')
```

```
Минимальная дата: 2019-08-01 00:00:00
Максимальная дата: 2019-08-31 00:00:00
```

Распределение записей по группам и датам в датасете "Пользователи"

In [10]:
```
print('Количество  записей дат для группы "A": ', visitors[visitors['group']=='A']
print('Количество уникальных записей дат для группы "A": ', visitors[visitors['grou
print()
print('Количество  записей дат для группы "B": ', visitors[visitors['group']=='B']
print('Количество уникальных записей дат для группы "B": ', visitors[visitors['grou
```

```
Количество  записей дат для группы "A":  31
Количество уникальных записей дат для группы "A":  31

Количество  записей дат для группы "B":  31
Количество уникальных записей дат для группы "B":  31
```

Количество групп в A/B тесте

In [11]:
```
print('Перечень групп в A/B тесте (датасет orders):')
print(orders['group'].unique())
print()
```

```
print('Перечень групп в A/B тесте (датасет visitors):')
print(visitors['group'].unique())
```

```
Перечень групп в A/B тесте (датасет orders):
['B' 'A']

Перечень групп в A/B тесте (датасет visitors):
['A' 'B']
```

Как имеем возможность наблюдать, в датасетах A/B-теста имеются группы только A и B. Все корректно.

Количество пользователей в каждой группе

In [12]:
```
# расчетный блок
visitorId_A =orders[orders['group']=='A']['visitorid'].count()
unique_visitorId_A = orders[orders['group']=='A']['visitorid'].nunique()

visitorId_B =orders[orders['group']=='B']['visitorid'].count()
unique_visitorId_B = orders[orders['group']=='B']['visitorid'].nunique()



print('Количество  заказов пользователей группы "A": ',visitorId_A )
print('Количество уникальных пользователей  группы "A": ', unique_visitorId_A)
print('Среднее количество заказов на пользователя для группы "A": ', round( visitor

print()
print('Количество  заказов пользователей группы "B": ', visitorId_B)
print('Количество уникальных пользователей  группы "B": ', unique_visitorId_B)
print('Среднее количество заказов на пользователя для группы "B": ', round( visitor

print()
print('Относительное увеличение количества заказов  группы "B" к "A" в процентах:
print('Относительное увеличение количества пользователей  группы "B" к "A" в процен
print('Относительное увеличение среднего количества пользователей  группы "B" к "A'
```

```
Количество  заказов пользователей группы "A":  557
Количество уникальных пользователей  группы "A":  503
Среднее количество заказов на пользователя для группы "A":  1.107

Количество  заказов пользователей группы "B":  640
Количество уникальных пользователей  группы "B":  586
Среднее количество заказов на пользователя для группы "B":  1.092

Относительное увеличение количества заказов  группы "B" к "A" в процентах:  114.90
1
Относительное увеличение количества пользователей  группы "B" к "A" в процентах:
116.501
Относительное увеличение среднего количества пользователей  группы "B" к "A" в про
центах:  98.627
```

1. В группе B стало больше пользователей (+ 14,9%) и заказов (+16,5%).
2. В группе B уменьшилось среднее количество заказов на одного пользователя.

Поверка групп на изолированность

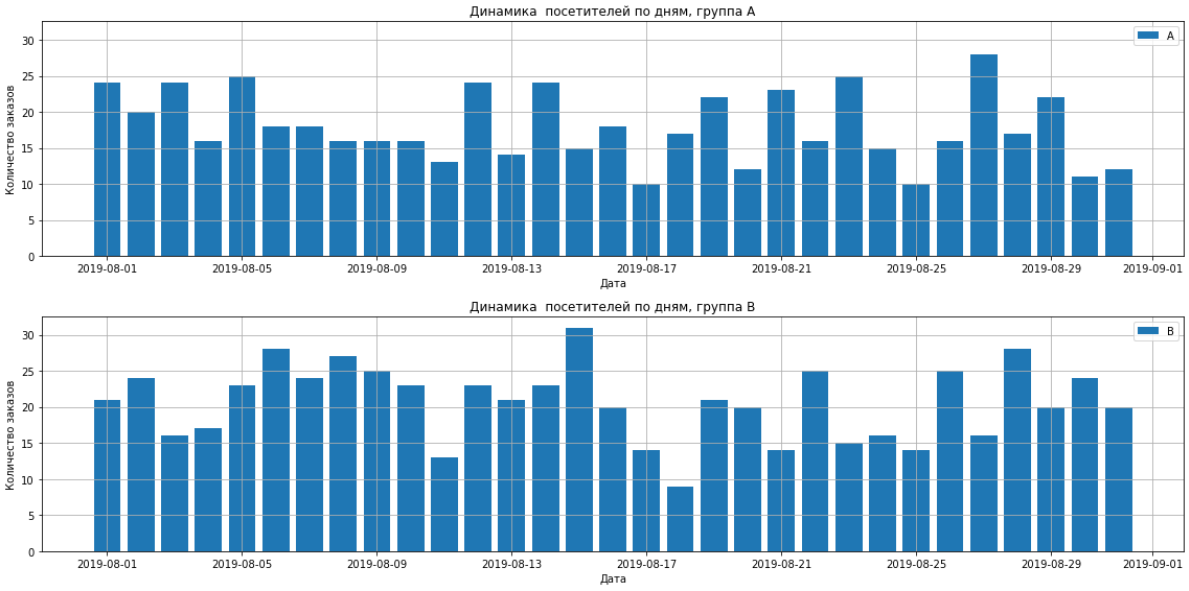
In [13]:
```
print('Количество пересекающихся пользователей в обеих группах:')
```

```
orders[(orders['group']=='A') & (orders['group']=='B') ]['visitorid'].nunique()
```

```
Количество пересекающихся пользователей в обеих группах:
0
```
Out[13]:

Динамика посетителей по дням по группам

In [14]:
```
#размер графика
plt.figure(figsize=(16,8))

dinamic = orders.groupby(by=['group','date']) ['visitorid'].count().reset_index()


# Строим график динамики посетителей по дням для группы A
ax1 = plt.subplot(2, 1, 1)
plt.bar( dinamic[dinamic['group']=='A']['date'], dinamic[dinamic['group']=='A']['v
plt.xlabel('Дата')
plt.ylabel('Количество заказов')
plt.title('Динамика  посетителей по дням, группа A')
plt.grid()
plt.legend()


# Строим график динамики посетителей по дням для группы B
ax2 = plt.subplot(2, 1, 2, sharey=ax1)
plt.bar(dinamic[dinamic['group']=='B']['date'], dinamic[dinamic['group']=='B']['vis
plt.xlabel('Дата')
plt.ylabel('Количество заказов')
plt.title('Динамика  посетителей по дням, группа B')
plt.grid()
plt.legend()

plt.tight_layout()
plt.show()
```



1. Характер взаимосвязи даты (числа месяца) и количества заказов для групп отличается.
2. Заметно в целом более высокое количество заказов для группы B.
3. В группе B есть дни с бОльшими просадками по количеству заказов.

Предварительные данные по датасетам:

1. Датасет с гипотезами: 9 гипотез, 4 критерия с оценками от 1 до 10.
2. Датасет с заказами: 1197 записей. Данные представлены за период с 1 по 31 августа 2019 года.
3. Датасет с пользователями: 62 записи. Записи распределены равномерно по датам и группам. Нет пересекающихся пользователей, одновременно представленных в обеих группах.
4. Данные представлены за период с 1 по 31 августа 2019 года.

Приоритизация гипотез

В датасете представлены 9 гипотез с уже выполненной оценкой по критериям

- Reach — охват пользователей по 10-балльной шкале;
- Impact — влияние на пользователей по 10-балльной шкале;
- Confidence — уверенность в гипотезе по 10-балльной шкале;
- Efforts — затраты ресурсов на проверку гипотезы по 10-балльной шкале. Чем больше значение Efforts, тем дороже проверка гипотезы.

Для корректной приоритизации применим фреймворки ICE и RICE, сравним результаты и сформулируем выводы.

Фреймворк ICE для приоритизации гипотез

В общем случае оценка по ICE = Impact x Confidence / Efforts, она учитывает влияние на пользователей, уверенность в гипотезе и затратность по ресурсам на реализацию идеи, отображенной в гипотезе.

Для оценки каждой гипотезы добавим столбец 'ice' в датасет 'hypothesis'. Для удобного чтения данных округлим результат до 3 знаков после запятой и отсортируем по оценке 'ice'

In [15]:
```
hypothesis['ice'] = round((hypothesis['impact'] * hypothesis['confidence'])/hypothe
pd.options.display.max_colwidth = 130
hypothesis.sort_values(by='ice', ascending = False)
```

Out[15]:

	hypothesis	reach	impact	confidence	efforts	ice
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.200
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6	13.333
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5	11.200
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.000
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.000
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.000
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.333
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.125
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.000

Расчет комплексной оценки ICE выявил три наиболее интересные гипотезы с номерами 8,0,7:

Фреймворк RICE для приоритизации гипотез

В общем случае оценка по RICE = Reach x Impact x Confidence / Efforts. Как и ICE, она учитывает влияние на пользователей, уверенность в гипотезе и затратность по ресурсам на реализацию идеи, отображенной в гипотезе. Дополнительно к ICE, оценка RICE учитывает охват пользователей.

Для оценки каждой гипотезы добавим столбец 'rice' в датасет 'hypothesis'. Для удобного чтения данных округлим результат до 3 знаков после запятой и отсортируем по оценке 'rice'

In [16]:
```
hypothesis['rice'] = round((hypothesis['reach'] * hypothesis['impact'] * hypothesis
pd.options.display.max_colwidth = 130
hypothesis.sort_values(by='rice', ascending = False)
```

Out[16]:

	hypothesis	reach	impact	confidence	efforts	ice	rice
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	10	7	8	5	11.200	112.0
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.000	56.0
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6	13.333	40.0
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.000	40.0
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.200	16.2
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.125	9.0
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.000	4.0
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.333	4.0
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.000	3.0

Расчет комплексной оценки RICE выявил три наиболее интересные гипотезы с номерами 7,2,0:

Выводы

1. По обоим методикам перечень топ-3 гипотез оказался отличным. 8-я гипотеза не прошла проверку охватом, а 7-я с третьего места по ICE вышла на 1-е место с учетом охвата пользователей.
2. Гипотеза номер 0 осталась в топ-3 и имеет шансы к реализаци, стоит проработать вопрос новых каналов привлечения пользователей
3. Гипотеза номер 2 с точки зрения RICE тоже имеет высокие шансы к реализации и, как следствие- к увеличению конверсии и среднего чека.
4. Первое место по RICE заняла гипотеза №7 "Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок". Реализация этой гипотезы затронет большинство пользователей и, возможно, принесет максимум пользы. По остальным критериям она занимает третье место с небольшим отрывом от первого.

Анализ A/B теста

По проектному заданию A/B-тест уже проведен и получены результаты, которые описаны в файлах 'orders.csv' и 'visitors.csv.

Проанализируем A/B - тест.

Предварительная обработка данных

In [17]:
```
# количество пользователей по датам для группы A
visitorsADaily = visitors[visitors['group'] == 'A'][['date', 'visitors']]
visitorsADaily.columns = ['date', 'visitorsPerDateA']

# кумулятивное количесто пользователей на  дату для группы A
visitorsACummulative = visitorsADaily.apply(
    lambda x: visitorsADaily[visitorsADaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateA': 'sum'}),     axis=1, )
visitorsACummulative.columns = ['date', 'visitorsCummulativeA']


# количесто пользователей по датам для группы B
visitorsBDaily = visitors[visitors['group'] == 'B'][['date', 'visitors']]
visitorsBDaily.columns = ['date', 'visitorsPerDateB']

# кумулятивное количество пользователей на  дату для группы И
visitorsBCummulative = visitorsBDaily.apply(
    lambda x: visitorsBDaily[visitorsBDaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateB': 'sum'} ),  axis=1, )

visitorsBCummulative.columns = ['date', 'visitorsCummulativeB']


# заказы группы A на дату
ordersADaily = (
    orders[orders['group'] == 'A'][['date', 'transactionid', 'visitorid', 'revenue
    .groupby('date', as_index=False)
    .agg({'transactionid': pd.Series.nunique, 'revenue': 'sum'})
)
ordersADaily.columns = ['date', 'ordersPerDateA', 'revenuePerDateA']


# заказы группы A кумулятивно на дату
ordersACummulative = ordersADaily.apply(
    lambda x: ordersADaily[ordersADaily['date'] <= x['date']].agg(
        {'date': 'max', 'ordersPerDateA': 'sum', 'revenuePerDateA': 'sum'}
    ),
    axis=1,
).sort_values(by=['date'])
ordersACummulative.columns = [
    'date',
    'ordersCummulativeA',
    'revenueCummulativeA',
]


# заказы группы B на дату
ordersBDaily = (
    orders[orders['group'] == 'B'][['date', 'transactionid', 'visitorid', 'revenue
    .groupby('date', as_index=False)
```

```
    .agg({'transactionid': pd.Series.nunique, 'revenue': 'sum'})
)
ordersBDaily.columns = ['date', 'ordersPerDateB', 'revenuePerDateB']


# заказы группы B кумулятивно на дату
ordersBCummulative = ordersBDaily.apply(
    lambda x: ordersBDaily[ordersBDaily['date'] <= x['date']].agg(
        {'date': 'max', 'ordersPerDateB': 'sum', 'revenuePerDateB': 'sum'}
    ),
    axis=1,
).sort_values(by=['date'])
ordersBCummulative.columns = [
    'date',
    'ordersCummulativeB',
    'revenueCummulativeB',
]


# собираем расчетные данные в одну таблицу.
data = (
    ordersADaily.merge(
        ordersBDaily, left_on='date', right_on='date', how='left'
    )
    .merge(ordersACummulative, left_on='date', right_on='date', how='left')
    .merge(ordersBCummulative, left_on='date', right_on='date', how='left')
    .merge(visitorsADaily, left_on='date', right_on='date', how='left')
    .merge(visitorsBDaily, left_on='date', right_on='date', how='left')
    .merge(visitorsACummulative, left_on='date', right_on='date', how='left')
    .merge(visitorsBCummulative, left_on='date', right_on='date', how='left')
)

data.sort_values(by='date', ascending=True).head()
```

Out[17]:

	date	ordersPerDateA	revenuePerDateA	ordersPerDateB	revenuePerDateB	ordersCummulativ
0	2019-08-01	24	148579	21	101217	
1	2019-08-02	20	93822	24	165531	
2	2019-08-03	24	112473	16	114248	
3	2019-08-04	16	70825	17	108571	
4	2019-08-05	25	124218	23	92428	

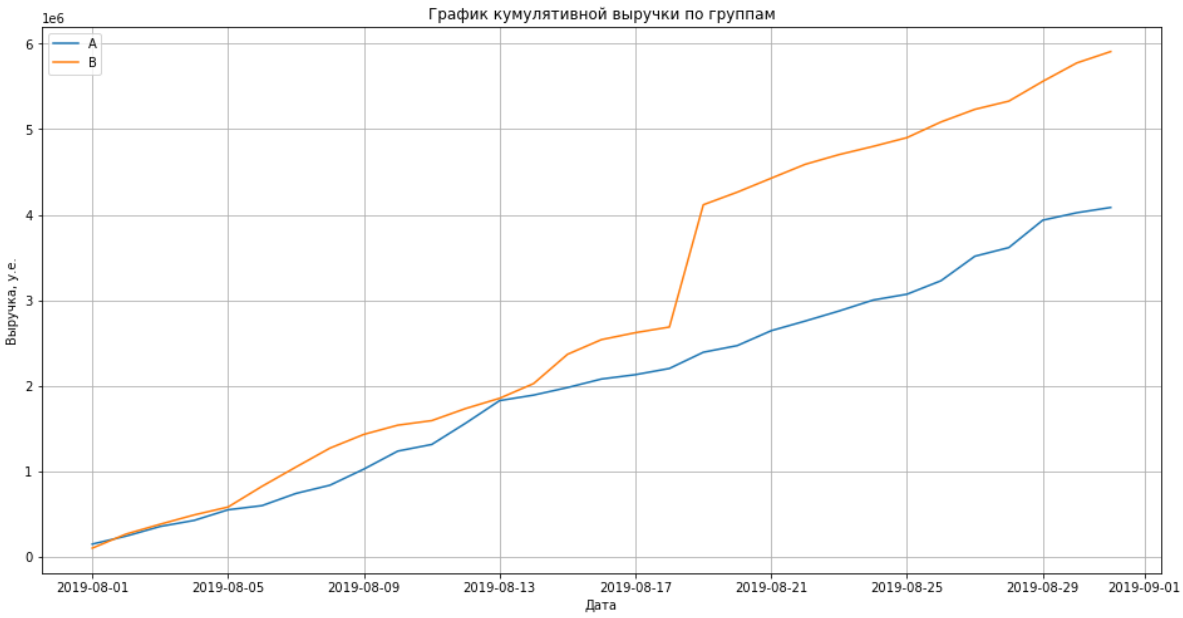
Названия столбцов данных сводной таблицы:

- date — дата;
- ordersPerDateA — количество заказов в выбранную дату в группе A;
- revenuePerDateA — суммарная выручка в выбранную дату в группе A;
- ordersPerDateB — количество заказов в выбранную дату в группе B;
- revenuePerDateB — суммарная выручка в выбранную дату в группе B;
- ordersCummulativeA — суммарное число заказов до выбранной даты включительно в группе A;

- revenueCummulativeA — суммарная выручка до выбранной даты включительно в группе A;
- ordersCummulativeB — суммарное количество заказов до выбранной даты включительно в группе B;
- revenueCummulativeB — суммарная выручка до выбранной даты включительно в группе B;
- visitorsPerDateA — количество пользователей в выбранную дату в группе A;
- visitorsPerDateB — количество пользователей в выбранную дату в группе B;
- visitorsCummulativeA — количество пользователей до выбранной даты включительно в группе A;
- visitorsCummulativeB — количество пользователей до выбранной даты включительно в группе B.

График кумулятивной выручки по группам.

In [18]:
```
#размер графика
plt.figure(figsize=(16, 8))

# Строим график кумулятивной выручки группы A
plt.plot(data['date'], data['revenueCummulativeA'], label='A')

# Строим график кумулятивной выручки группы B
plt.plot(data['date'], data['revenueCummulativeB'], label='B')

plt.xlabel('Дата')
plt.ylabel('Выручка, у.е.')
plt.title('График кумулятивной выручки по группам')
plt.grid()
plt.legend()
plt.show()
```



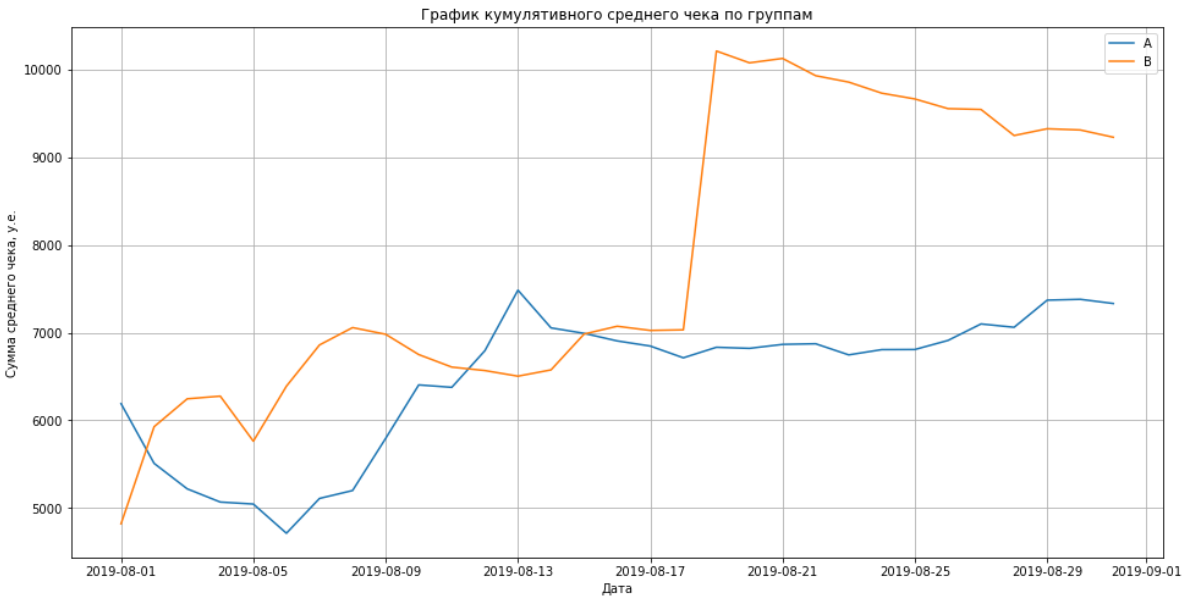
1. Группа A относительно стабильно растет со временем.
2. Группа B имеет резкий скачок вверх кумулятивной выручки во второй половине месяца. Но даже без него график идет выше контрольной группы (хотя и

примерно с тем же наклоном)- показатели группы В лучше.

3. Выручка зависит от количества заказов и среднего чека. Увеличение выручки группы В относительно контрольной группы А может свидетельствовать об увеличении количества заказов, увеличении среднего чека или о том и другом вместе.
4. Скорее всего резкий рост выручки по группе В обусловлен аномально дорогим единичным заказом- это более вероятно, чем резкое единократное увеличение колизества заказов (пользователей), так как в противном случае добавившиеся в таком количестве пользователи скорее всего разогнали бы поступление выручки и это было б не единократное событие, а тренд. Дальнейший анализ покажет картину изменений яснее.

График кумулятивного среднего чека по группам

In [19]:
```
#размер графика
plt.figure(figsize=(16, 8))

# Строим график кумулятивного среднего чека группы A
plt.plot(data['date'], data['revenueCummulativeA']/data['ordersCummulativeA'], labe

# Строим график кумулятивного среднего чека группы B
plt.plot(data['date'], data['revenueCummulativeB']/data['ordersCummulativeB'], labe

plt.xlabel('Дата')
plt.ylabel('Сумма среднего чека, у.е.')
plt.title('График кумулятивного среднего чека по группам')
plt.grid()
plt.legend()
plt.show()
```

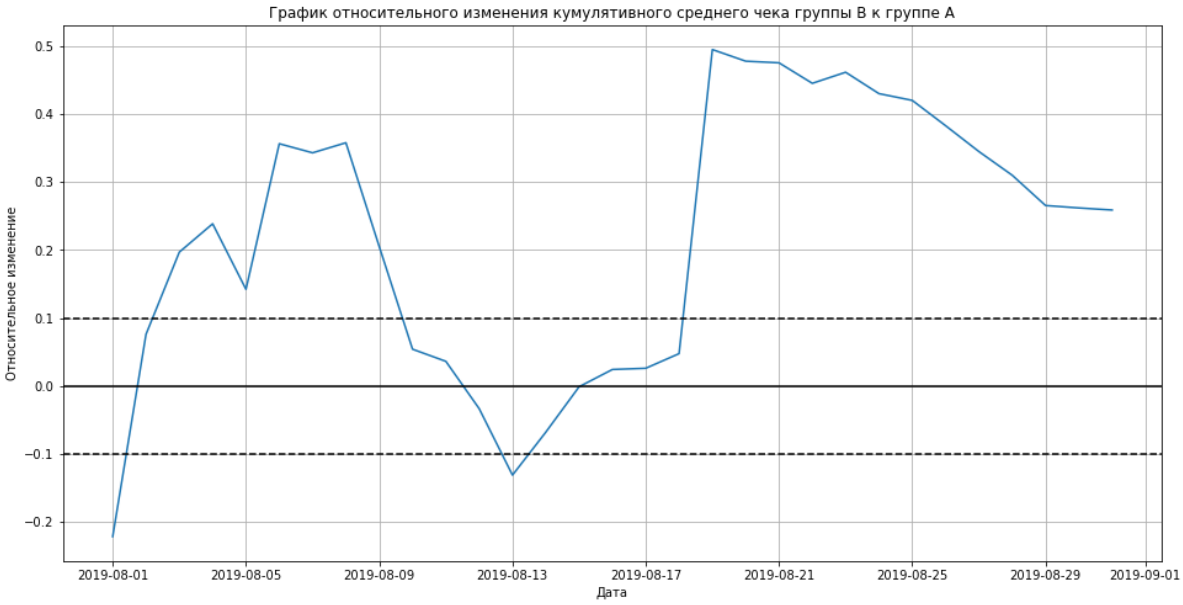


1. Очень непостоянные кривые в первой половине месяца для обеих групп. Во второй половине месяца группа А показывает относительно стабильные результаты.
2. Для группы В во второй половине месяца, как и на графике с выручкой, ориентировочно 18 августа наблюдается аномальная покупка, которая сдвигает

показатели группы радикально вверх с последующим снижением к концу месяца.

График относительного изменения кумулятивного среднего чека группы B к группе A

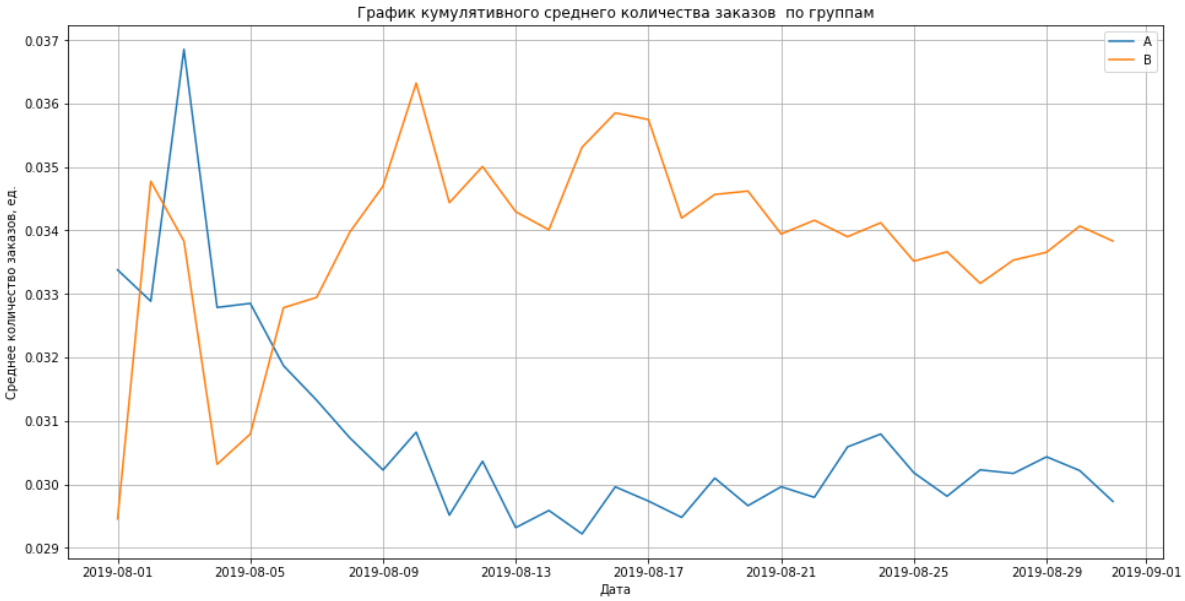
In [20]:
```
#размер графика
plt.figure(figsize=(16, 8))

# Строим график относительного изменения кумулятивного среднего чека группы B к гру

plt.plot(data['date'], (data['revenueCummulativeB']/data['ordersCummulativeB'])/(da


# добавляем ось X
plt.xlabel('Дата')
plt.ylabel('Относительное изменение')
plt.axhline(y=0, color='black', linestyle='-')
plt.axhline(y=0.1, color='black', linestyle='--')
plt.axhline(y=-0.1, color='black', linestyle='--')
plt.title('График относительного изменения кумулятивного среднего чека группы B к г
plt.grid()

plt.show()
```



1. Неоднозначные показатели. Первую неделю вторая группа B показывает результаты лучше контрольной группы A.
2. Вторая неделя характеризуется сильным падением группы B вплоть до отрицательных значений. Это связано, в том числе, с хорошими показателями группы A и трендом на снижение среднего чека для группы B, что приводит к просадке относительного показателя группы B.
3. Третья неделя для группы B положительна- намечается восходящий тренд, который резко усиливается единичной аномальной покупкой
4. Четвертая неделя показывает плавное снижение относительного показателя по средему чеку группы B. Если б не скачок, вызванный аномальной покупкой, то график ушел бы сильно в отрицательную зону.

График кумулятивного среднего количества заказов по группам

In [21]:
```
#размер графика
plt.figure(figsize=(16, 8))

# Строим график кумулятивного среднего чека группы A
plt.plot(data['date'], data['ordersCummulativeA']/data['visitorsCummulativeA'], lab

# Строим график кумулятивного среднего чека группы B
plt.plot(data['date'], data['ordersCummulativeB']/data['visitorsCummulativeB'], lab

plt.xlabel('Дата')
plt.ylabel('Среднее количество заказов, ед.')
plt.title('График кумулятивного среднего количества заказов  по группам')
plt.grid()
plt.legend()
plt.show()
```



1. Среднее количество заказов на первой неделе нестабильно для обеих групп.
2. К концу первой недели наблюдается два контртренда: среднее количество заказов для группы А падает, а для группы В- растет.
3. Во второй половине месяца наблюдает стабилизировавшиеся показатели для группы А и некоторый тренд на снижение для группы В. Причем аномальная покупка в группе В дает локальный прирост характеристики, но, в целом, не меняет характер зависимости.
4. Предположение, отмеченное выше в п.3.1, подтверждается: в группе В наблюдаем стабильно более высокий уровень количества заказов.

График относительного изменения кумулятивного среднего количества заказов группы B к группе A

In [22]:
```
#размер графика
plt.figure(figsize=(16, 8))

# Строим график относительного изменения кумулятивного среднего чека группы B к гру
```

```
plt.plot(data['date'], (data['ordersCummulativeB']/data['visitorsCummulativeB'])/ (

plt.xlabel('Дата')
plt.ylabel('Относительное изменение.')
# добавляем ось X
plt.axhline(y=0, color='black', linestyle='-')
plt.axhline(y=0.1, color='black', linestyle='--')
plt.title('График относительного изменения кумулятивного среднего количества заказо
plt.grid()

plt.show()
```

1. В первую неделю, скорее всего, изменения для группы B еще не начали действовать в полную силу. Поэтому в среднем количество заказов в контрольной группе A преобладает.
2. Начиная с 5 числа баланс сместился к группе B, а после 8 числа заказов у группы B минимум на 10% больше,чем у группы A (с одним микроисключением 27 августа).
3. Группа B со второй недели показывает стабильно лучшие результаты, чем контрольная A.

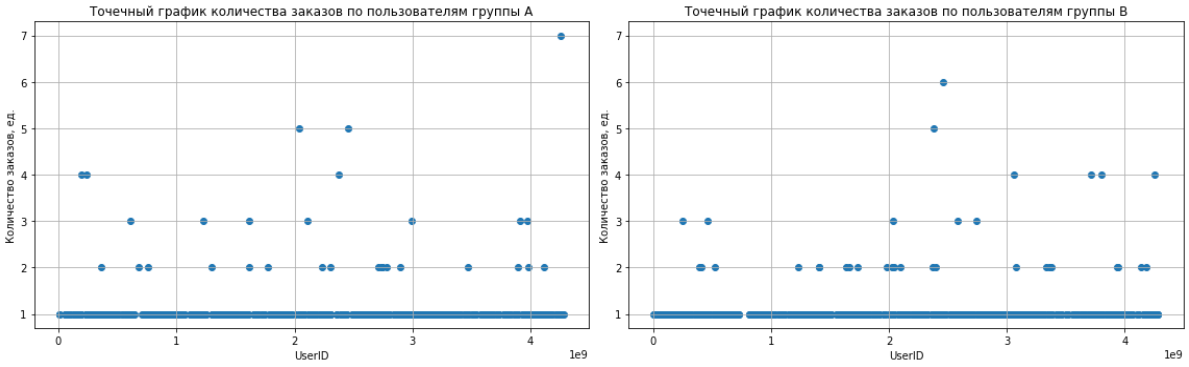
Точечный график количества заказов по пользователям

Формируем сводную таблицу по заказам для каждого пользователя в разбивке

In [23]:
```
# пользователи группы A
ordersByUsersA = ( orders[orders['group'] == 'A'].groupby('visitorid', as_index=Fa
ordersByUsersA.columns = ['userId', 'orders']

# пользователи группы B
ordersByUsersB = ( orders[orders['group'] == 'B'].groupby('visitorid', as_index=Fa
ordersByUsersB.columns = ['userId', 'orders']
```

In [24]:
```
#размер графика
plt.figure(figsize=(16, 5))
```

```
# Строим точечный график количества заказов по пользователям группы A
# так как UserId- обычное число ( внашем случае- уникальное),а интересует нас графи
ax1 = plt.subplot(1, 2, 1)
plt.xlabel('UserID')
plt.ylabel('Количество заказов, ед.')
plt.scatter(ordersByUsersA['userId'], ordersByUsersA['orders'])
plt.title('Точечный график количества заказов по пользователям группы A')
plt.grid()


# Строим точечный график количества заказов по пользователям группы B
# так как UserId- обычное число ( внашем случае- уникальное),а интересует нас графи
ax2 = plt.subplot(1, 2, 2, sharey=ax1)
plt.xlabel('UserID')
plt.ylabel('Количество заказов, ед.')
plt.scatter(ordersByUsersB['userId'], ordersByUsersB['orders'])
plt.title('Точечный график количества заказов по пользователям группы B')
plt.grid()

plt.tight_layout()
plt.show()
```



1. Основная масса пользователей делает по одному заказу.
2. Также есть некоторая часть пользователей с 2-3 заказами. Большее количество заказов уже не характерно для пользователей.

Расчет 95-го и 99-го перцентиля количества заказов на пользователя. Выбор границы для определения аномальных пользователей.

In [25]:
```
print('Расчет 95-го и 99-го перцентиля количества заказов на пользователя для групп
print(np.percentile(ordersByUsersA['orders'], [95, 99]))
print()
print('Расчет 95-го и 99-го перцентиля количества заказов на пользователя для групп
print(np.percentile(ordersByUsersB['orders'], [95, 99]))
```

```
Расчет 95-го и 99-го перцентиля количества заказов на пользователя для группы A
[2.   3.98]

Расчет 95-го и 99-го перцентиля количества заказов на пользователя для группы B
[2.   3.15]
```

1. 95% пользователей делают не более 2 заказов.
2. Менее 1% пользователей размещают более 3 заказов.

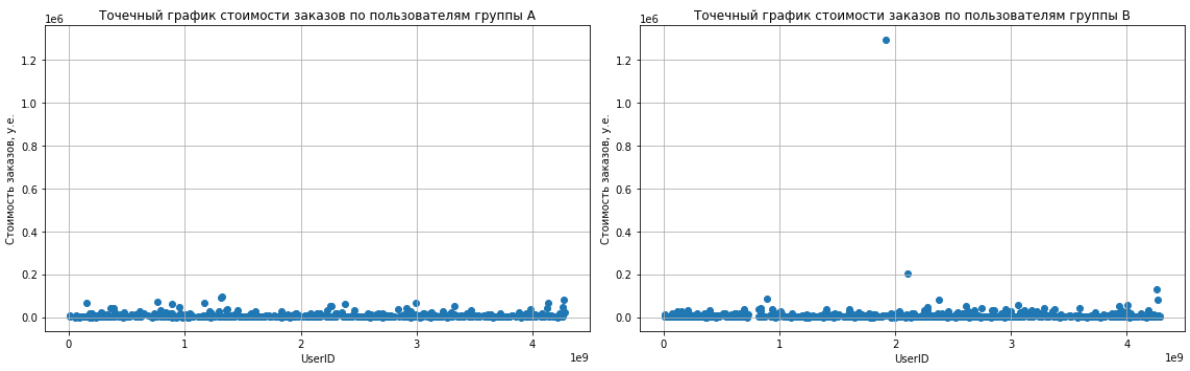
3. Учитывая расчет перцентилей и графическое распределение данных, определим границу аномального пользователя - 3 заказа в одни руки.

Точечный график стоимости заказов по пользователям

Формируем сводную таблицу по заказам для каждого пользователя

In [26]:
```
# пользователи группы A
revenueByUsersA = ( orders[orders['group'] == 'A'].groupby('visitorid', as_index=Fa
revenueByUsersA.columns = ['userId', 'revenue']

# пользователи группы B
revenueByUsersB = ( orders[orders['group'] == 'B'].groupby('visitorid', as_index=Fa
revenueByUsersB.columns = ['userId', 'revenue']
```

In [27]:
```
#размер графика
plt.figure(figsize=(16, 5))


# Строим точечный график количества заказов по пользователям группы A
# так как UserId- обычное число ( внашем случае- уникальное),а интересует нас графи
ax1 = plt.subplot(1, 2, 1)
plt.xlabel('UserID')
plt.ylabel('Стоимость заказов, у.е.')
plt.scatter(revenueByUsersA['userId'], revenueByUsersA['revenue'])
plt.title('Точечный график стоимости заказов по пользователям группы A')
plt.grid()


# Строим точечный график количества заказов по пользователям группы B
# так как UserId- обычное число ( внашем случае- уникальное),а интересует нас графи
ax2 = plt.subplot(1, 2, 2, sharey=ax1)
plt.xlabel('UserID')
plt.ylabel('Стоимость заказов, у.е.')

plt.scatter(revenueByUsersB['userId'], revenueByUsersB['revenue'])
plt.title('Точечный график стоимости заказов по пользователям группы B')
plt.grid()

plt.tight_layout()
plt.show()
```

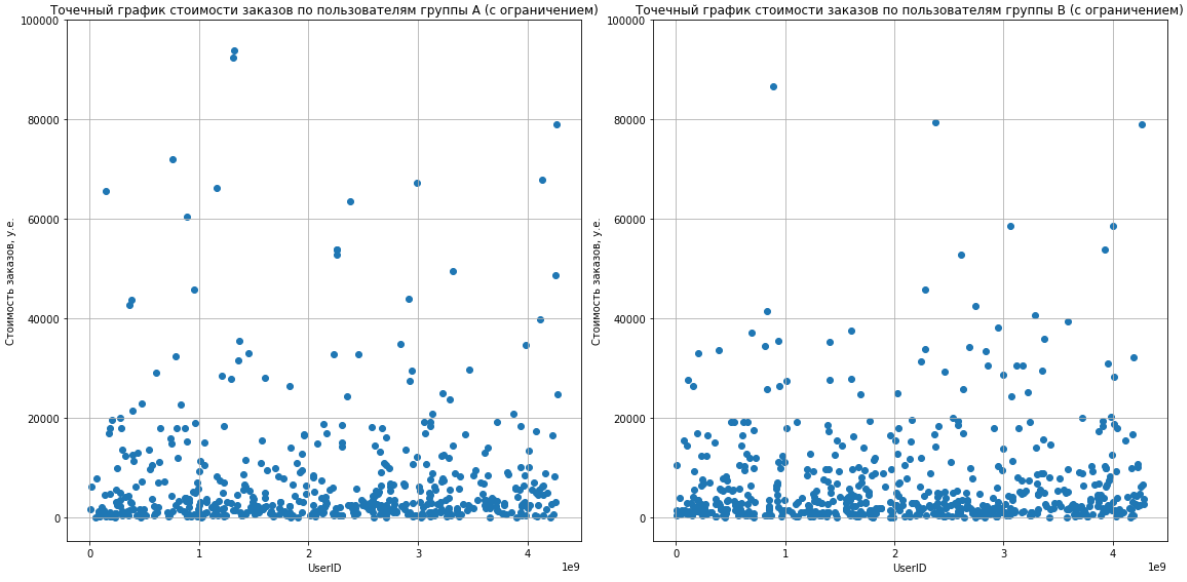


1. Уникальный пользователь с заказом в 1.3 млн у.е. для группы B.
2. Основная масса пользователей размещает заказы суммарно до 100 000 у.е.

Введем ограничение в 100 000 у.е. по верхнему значению стоимости заказов для обеих

групп для более детального рассмотрения характера распределения заказов.

In [28]:
```
#размер графика
plt.figure(figsize=(16, 8))


# Строим точечный график количества заказов по пользователям группы A
# так как UserId- обычное число ( внашем случае- уникальное),а интересует нас графи
ax1 = plt.subplot(1, 2, 1)
plt.xlabel('UserID')
plt.ylabel('Стоимость заказов, у.е.')
plt.scatter(revenueByUsersA['userId'], revenueByUsersA['revenue'])
plt.title('Точечный график стоимости заказов по пользователям группы A (с ограничен
plt.grid()
plt.ylim((None,100000))


# Строим точечный график количества заказов по пользователям группы B
# так как UserId- обычное число ( внашем случае- уникальное),а интересует нас графи
ax2 = plt.subplot(1, 2, 2, sharey=ax1)
plt.xlabel('UserID')
plt.ylabel('Стоимость заказов, у.е.')

plt.scatter(revenueByUsersB['userId'], revenueByUsersB['revenue'])
plt.title('Точечный график стоимости заказов по пользователям группы B (с ограничен
plt.grid()
plt.ylim((None,100000))

plt.tight_layout()
plt.show()
```



Наблюдаем примерно одинаковое распределение заказов по частоте и стоимости заказа как в группе A, так и в B.

Расчет 95-го и 99-го перцентиля стоимости заказов на пользователя. Выбор границы для определения аномальных пользователей.

In [29]:
```
print('Расчет 95-го и 99-го перцентиля стоимости заказов на пользователя для группы
print(np.percentile(revenueByUsersA['revenue'], [95, 99]))
```

```
print()
print('Расчет 95-го и 99-го перцентиля стоимости заказов на пользователя для группы
print(np.percentile(revenueByUsersB['revenue'], [95, 99]))
```

```
Расчет 95-го и 99-го перцентиля стоимости заказов на пользователя для группы A
[32763.1 67173.2]

Расчет 95-го и 99-го перцентиля стоимости заказов на пользователя для группы B
[30904.75 61616.  ]
```

1. 95% пользователей размещают заказы на сумму не более 32 763 (A) и 30904 (B) у.е.
2. 99% пользователей размещают заказы на сумму не более 67 173 (A) и 61616(B) у.е.
3. Учитывая расчет перцентилей и графическое распределение данных, определим границу аномального пользователя - 50 000 у.е.

Расчет статистической значимости различий в среднем количестве заказов между группами по «сырым» данным.

Объявим переменные sampleA и sampleB, в которых пользователям из разных групп будет соответствовать количество заказов. Тем, кто ничего не заказал, будут соответствовать нули.

Для проведения теста Уилкоксона-Манна-Уитни сформулируем гипотезы:

- Нулевая: различий в среднем количестве заказов между группами нет.
- Альтернативная: различия в среднем между группами есть.

Определим уровень значимости в 5%, расчетное p-value округлим до 3 знака после запятой.

In [30]:
```
sampleA = pd.concat([ordersByUsersA['orders'],pd.Series(0, index=np.arange(data['v

sampleB = pd.concat([ordersByUsersB['orders'],pd.Series(0, index=np.arange(data['v


alpha = 0.05
pvalue = stats.mannwhitneyu(sampleA, sampleB)[1]

print('Расчет статистической значимости различий в среднем количестве заказов межд

print('alpha = ', alpha)
print('Критерий Манна-Уитни:')
print("{0:.3f}".format(pvalue))

if pvalue < alpha:
    print("Статистически значимые различия между группами есть")
else:
    print("Статистически значимых различий между группами нет")

print()
print('Относительное различие между группами B и A:')
print("{0:.3f}".format(sampleB.mean() / sampleA.mean() - 1))
```

```
Расчет статистической значимости различий в среднем количестве заказов между групп
ами
alpha =  0.05
Критерий Манна-Уитни:
0.017
Статистически значимые различия между группами есть

Относительное различие между группами B и A:
0.138
```

1. Статистически значимая разница в среднем количестве заказов между группами с "сырыми" данными составляет 13.8%.
2. "Сырые" данные показывают преимущество группы B по количеству заказов.

Расчет статистической значимости различий в среднем чеке заказа между группами по «сырым» данным.

Для проведения теста Уилкоксона-Манна-Уитни сформулируем гипотезы:

- Нулевая: различий в среднем количестве заказов между группами нет.
- Альтернативная: различия в среднем между группами есть.

Определим уровень значимости в 5%, расчетное p-value округлим до 3 знака после запятой.

In [31]:
```
alpha = 0.05
pvalue = stats.mannwhitneyu(orders[orders['group']=='A']['revenue'], orders[orders

print('Расчет статистической значимости различий в среднем чеке заказа между группа

print('alpha = ', alpha)
print('Критерий Манна-Уитни:')
print("{0:.3f}".format(pvalue))

if pvalue < alpha:
    print("Статистически значимые различия между группами есть")
else:
    print("Статистически значимых различий между группами нет")

print()
print('Относительное различие между группами B и A:')
print("{0:.3f}".format(orders[orders['group']=='B']['revenue'].mean()/orders[orders
```

```
Расчет статистической значимости различий в среднем чеке заказа между группами
alpha =  0.05
Критерий Манна-Уитни:
0.729
Статистически значимых различий между группами нет

Относительное различие между группами B и A:
0.259
```

1. Не смотря на существенное относительное различие почти в 26%, статистически значимых различий между группами нет.
2. Такая значительная разница, учитывая расчет статзначимости, могла оказаться случайностью.

3. Но это для "сырых" данных с аномалими.

Расчет статистической значимости различий в среднем количестве заказов между группами по «очищенным» данным.

Определение аномальных пользователей

Примем за аномальных пользователей тех, кто совершил более 3 заказов, или разместил заказы на сумму свыше 50 000 у.е..

In [32]:
```
# введем лимитирующие переменные по аномальным пользователям
orders_limit_A = 3
orders_limit_B = 3
revenue_limit = 50000



# определим пользователей, которые размещают много заказов
usersWithManyOrders = pd.concat(
    [   ordersByUsersA[ordersByUsersA['orders'] > orders_limit_A]['userId'],
        ordersByUsersB[ordersByUsersB['orders'] > orders_limit_B]['userId'] ], axis

# определим пользователей, которые размещают дорогие заказы
usersWithExpensiveOrders = orders[orders['revenue'] > revenue_limit]['visitorid']

# объединим базы аномальных пользователей
abnormalUsers = (
    pd.concat([usersWithManyOrders, usersWithExpensiveOrders], axis=0)
    .drop_duplicates()
    .sort_values() )

print('Аномальные пользователи')
print(abnormalUsers.head())
```

```
Аномальные пользователи
1099    148427295
18      199603092
23      237748145
1137    759473111
949     887908475
dtype: int64
```

Объявим переменные sampleAFiltered и sampleBFiltered, в которых сохраним очищенные данные о заказах — не включая аномальных пользователей.

In [33]:
```
sampleAFiltered = pd.concat([ordersByUsersA[np.logical_not(ordersByUsersA['userId']
        pd.Series(0, index=np.arange(data['visitorsPerDateA'].sum() - len(ordersByU

sampleBFiltered = pd.concat([ordersByUsersB[np.logical_not(ordersByUsersB['userId']
        pd.Series(0, index=np.arange(data['visitorsPerDateB'].sum() - len(ordersByU
```

Для проведения теста Уилкоксона-Манна-Уитни сформулируем гипотезы:

- Нулевая: различий в среднем количестве заказов между группами нет.
- Альтернативная: различия в среднем между группами есть.

Определим уровень значимости в 5%, расчетное p-value округлим до 3 знака после запятой.

In [34]:
```
alpha = 0.05
pvalue = stats.mannwhitneyu(sampleAFiltered, sampleBFiltered)[1]

print('Расчет статистической значимости различий в среднем количестве заказов между

print('alpha = ', alpha)
print('Критерий Манна-Уитни:')
print("{0:.3f}".format(pvalue))

if pvalue < alpha:
    print("Статистически значимые различия между группами есть")
else:
    print("Статистически значимых различий между группами нет")

print()
print('Относительное различие между группами B и A:')
print("{0:.3f}".format(sampleBFiltered.mean() / sampleAFiltered.mean() - 1))
```

```
Расчет статистической значимости различий в среднем количестве заказов между групп
ами
alpha =  0.05
Критерий Манна-Уитни:
0.011
Статистически значимые различия между группами есть

Относительное различие между группами B и A:
0.158
```

1. Статистически значимое различие между группами по "очищенным" данным увеличилось до 15.8 %.
2. Пользователи группы B действительно размещают больше заказов, чем контрольная группа.

Расчет статистической значимости различий в среднем чеке заказа между группами по «очищенным» данным.

Для проведения теста Уилкоксона-Манна-Уитни сформулируем гипотезы:

- Нулевая: различий в среднем количестве заказов между группами нет.
- Альтернативная: различия в среднем между группами есть.

Определим уровень значимости в 5%, расчетное p-value округлим до 3 знака после запятой.

In [35]:
```
alpha = 0.05
pvalue = stats.mannwhitneyu(
                orders[np.logical_and(orders['group'] == 'A',np.logical_not(orders
                orders[np.logical_and(orders['group'] == 'B',np.logical_not(orders

print('Расчет статистической значимости различий в среднем чеке заказа между группа

print('alpha = ', alpha)
print('Критерий Манна-Уитни:')
```

```
print("{0:.3f}".format(pvalue))

if pvalue < alpha:
    print("Статистически значимые различия между группами есть")
else:
    print("Статистически значимых различий между группами нет")

print()
print('Относительное различие между группами B и A:')
print("{0:.3f}".format(orders[np.logical_and(orders['group'] == 'B',np.logical_not
```

```
Расчет статистической значимости различий в среднем чеке заказа между группами (оч
ищенные данные)
alpha =  0.05
Критерий Манна-Уитни:
0.819
Статистически значимых различий между группами нет

Относительное различие между группами B и A:
0.024
```

1. Для очищенных данных относительное различие уменьшилось до 2,4%, статистически значимых различий между группами нет.
2. Группа B генерирует средние чеки такие же, как группа A.

Решение по результатам теста

"Сырые" данные

Графичесий анализ

1. Кумулятивная средняя выручка группы B выше A.
2. Кумулятивный средний чек группы B выше A, но не стабильно и, если отбросить аномальный заказ, то период превышения группы B относительно A уменьшится.
3. Относительное изменение кумулятивного среднего чека для группы B выше A тоже не стабильно высокое. Есть периоды доминиирования группы A. А также картину превосходства гр. B изменит учет аномально дорогого заказа.
4. Кумулятивное среднее количество заказов по группе B выше, чем A.
5. Относительное изменение среднего количества заказов группы B выше A.

Статистический анализ

1. Количество заказов. Наблюдается статистически значимая разница 13,8 %. В группе B заказов больше, чем в A.
2. Средний чек. Расчетное превышение группы B в 25,9% не является статистически значимым, такое превышение могло оказаться случайно (что подтверждается очищением данных от аномальных записей)

"Очищенные" данные. Статистический анализ

1. Количество заказов. Наблюдается статистически значимая разница в 15,8 %. В группе B заказов больше, чем в A даже с учетом отсеивания аномальных заказов.

2. Средний чек. Расчетное превышение группы B в 2.4 % не является статистически значимым. Удаление аномальных пользователей вернуло проецент превышения в адекватный диапазон (при отсутстствии стат.значимости различий).

Вывод по результатам теста

1. Графический и статистический анализ сырых и очищенных данных показывает положительное отличие группы B от контрольной A.
2. Группа B генерирует больше заказов при примерно одинаковом среднем чеке.
3. Как следствие- выручка группы B выше, чем у контрольной A.
4. Учитывая данные, имеет смысл остановить тест.
5. Признать эксперимент по группе B успешным или нет- зависит от изначальной поставновки гипотезы: если гипотеза предполагала увеличение выручки или количества заказов, то эксперимент успешен. Если гипотеза предполагала увеличение среднего чека, то по результатам эксперимента статистически значимых изменений не зафиксировано.
6. Так как в целом группа B приносит больше выручки и заказов, то наблюдается положительный эффект при проведении A/B теста, поэтому его можно остановить и зафиксировать победу группы B.