Table of Contents

Проект "Принятие решений в бизнесе"

Вместе с отделом маркетинга подготовлен список гипотез для увеличения выручки онлайн-магазина.

Необходимо приоритезировать гипотезы, запустить A/B-тест и проанализировать его результаты.

План работы:

Предобработка данных

Часть 1. Приоритизация гипотез.

- Применение фреймворков ICE и RICE, выбор приоритетных гипотез

Часть 2. Анализ A/B-теста

1. Построение следующих графиков:

- график кумулятивной выручки по группам
- график кумулятивного среднего чека по группам
- график относительного изменения кумулятивного среднего чека группы B к группе A
- график кумулятивного среднего количества заказов на посетителя по группам
- график относительного изменения кумулятивного среднего количества заказов на посетителя группы B к группе A
- точечный график количества заказов по пользователям
- точечный график стоимостей заказов

Выводы и предположения по каждому графику.

1. Определение границ для определения аномальных пользователей:

- подсчет 95-й и 99-й перцентили количества заказов на пользователя, стоимости заказов

Далее будем работать с "сырыми", исходными данными и отфильтрованными от аномалий

1. Расчет статистической значимости:

- различий в среднем количестве заказов на посетителя между группами по «сырым» данным
- различий в среднем чеке заказа между группами по «сырым» данным
- различий в среднем количестве заказов на посетителя между группами по «очищенным» данным
- различий в среднем чеке заказа между группами по «очищенным» данным

Принятие решения на основании результатов теста:

1. Остановить тест, зафиксировать победу одной из групп.
2. Остановить тест, зафиксировать отсутствие различий между группами.
3. Продолжить тест.

Описание данных:

Датасет *hypothesis*:

- 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.

Датасет *orders*:

- 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.

Датасет *visitors*:

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

Знакомство с данными. Предобработка

In [1]:
```
# импорт библиотек
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('ggplot')
%config InlineBackend.figure_format = 'retina'
from scipy import stats as st
import datetime as dt
from pandas.plotting import register_matplotlib_converters
import warnings
register_matplotlib_converters()
pd.options.display.max_colwidth = 130
```

In [2]:
```
# сохранение датасетов в одноименные переменные:
try:
    hypothesis = pd.read_csv(r'\Users\Hp\Desktop\Практикум\AB\hypothesis.csv')
    orders = pd.read_csv(r'\Users\Hp\Desktop\Практикум\AB\orders.csv')
    visitors = pd.read_csv(r'\Users\Hp\Desktop\Практикум\AB\visitors.csv')
except:
    hypothesis = pd.read_csv('https://code.s3.yandex.net/datasets/hypothesis.csv')
    orders = pd.read_csv('https://code.s3.yandex.net/datasets/orders.csv')
    visitors = pd.read_csv('https://code.s3.yandex.net/datasets/visitors.csv')
```

Таблица hypothesis

In [3]:
```
display(hypothesis)
hypothesis.info()
```

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

```
<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
```

В таблице 9 гипотез, данные хранятся в корректном виде. Приведем названия столбцов к snake_case:

In [4]:
```
# приведение названий столбцов к нижнему регистру:
hypothesis.columns = [x.lower() for x in hypothesis.columns]
```

Таблица hypothesis готова к работе

Таблица orders

In [5]:
```
display(orders.head())
orders.info()
```

	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

```
<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
```

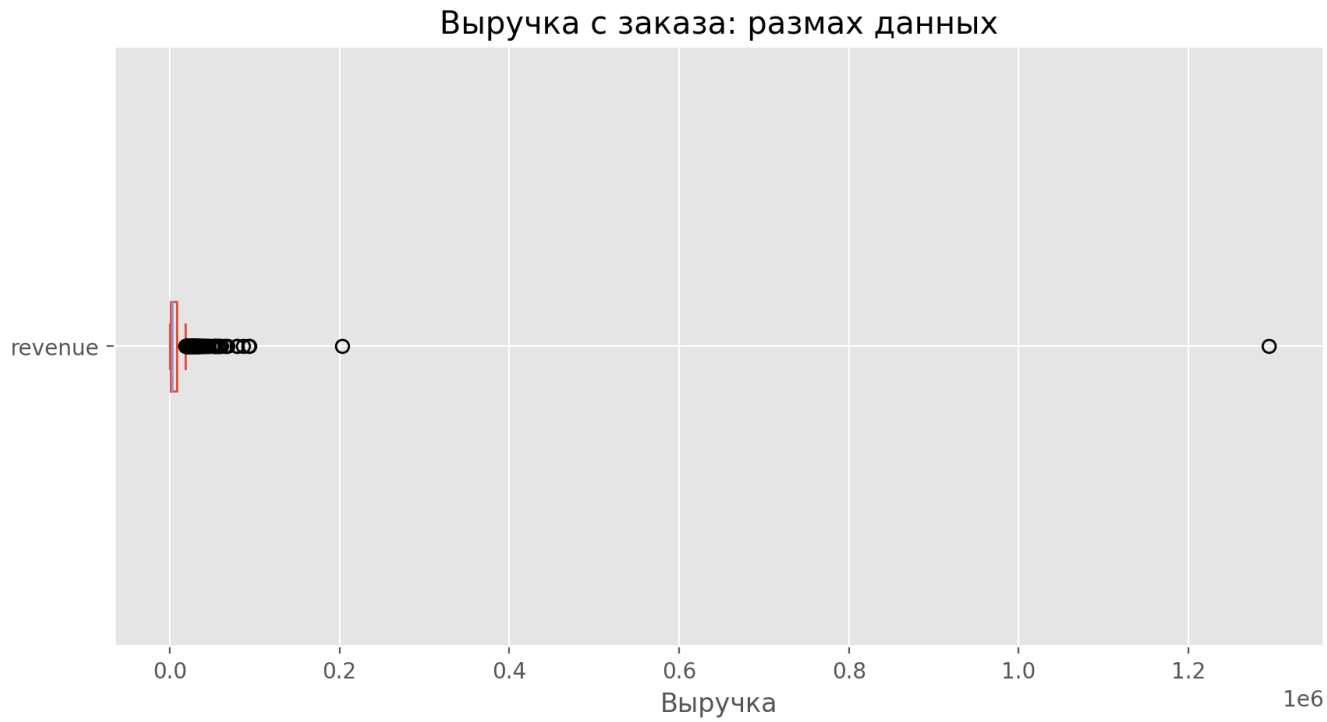
Таблица содержит 1197 записей, пропусков нет. Приведем дату к корректному типу, проверим наличие дубликатов, выявим начальную и конечную даты, бегло взглянем на распределение в столбце '*revenue*':

In [6]:
```
#изменение типа данных в стобце 'date':
orders["date"] = orders["date"].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))

print('Число дубликатов в таблице orders:', orders.duplicated().sum())
print('Дата начала теста:', orders['date'].min())
print('Дата окончания теста:',orders['date'].max())
```

```
Число дубликатов в таблице orders: 0
Дата начала теста: 2019-08-01 00:00:00
Дата окончания теста: 2019-08-31 00:00:00
```

In [7]:
```
plt.figure(figsize=(10, 5))
orders['revenue'].plot(kind='box', vert=False);
plt.title('Выручка с заказа: размах данных')
plt.xlabel('Выручка')
plt.ylabel('')
plt.show()
```



Исследуемый период: 1-31 августа 2019. Столбец revenue очевидно содержит выбросы - встречается как минимум одно значение > 1.2 млн, рассмотрим их ближе при проведении статистического анализа.

Таблица orders готова к работе.

Таблица visitors

In [8]:
```
display(visitors.head())
visitors.info()
```

	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

```
<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
```

Таблица содержит 62 наблюдения, пропусков нет. Изменим тип данных в столбце *date*, выявим дату старта и окончания, посмотрим на распределение в столбце visitors

In [9]:
```
#изменение типа данных в стобце 'date':
visitors['date'] = visitors['date'].map(lambda x: dt.datetime.strptime(x, '%Y-%m-%d'))

print('Число дубликатов в таблице visitors:', visitors.duplicated().sum())
print('Дата начала теста:', visitors['date'].min())
print('Дата окончания теста:',visitors['date'].max())
```

```
Число дубликатов в таблице visitors: 0
Дата начала теста: 2019-08-01 00:00:00
Дата окончания теста: 2019-08-31 00:00:00
```

Исследуемый период совпадает с периодом в таблице orders

Дополнительная предобработка для анализи A/B-теста

Проверим на какие группы деляться пользователи в обеих таблицах:

In [10]:
```
print('Группы в таблице orders:', orders['group'].sort_values().unique())
print('Группы в таблице visitors:', visitors['group'].sort_values().unique())
```

```
Группы в таблице orders: ['A' 'B']
Группы в таблице visitors: ['A' 'B']
```

В обеих таблицах встречаются только две группы A и B, никаких сюрпризов нет. Проверим, `могли ли покупатели попасть в обе группы одновременно`

In [11]:
```
a_id = orders.query('group == "A"')['visitorId'].unique().tolist()
b_id = orders.query('group == "B"')['visitorId'].unique().tolist()

double_id_orders = []

for a in a_id:
    if a in b_id:
        double_id_orders.append(a)
print('Количество покупателей, попавших в обе группы:', len(double_id_orders))
```

```
Количество покупателей, попавших в обе группы: 58
```

`Встречается 58 покупателей, попавших в обе группы одновременно.` Если пользователь видит разные версии исследуемой страницы в ходе одного исследования, неизвестно, какая именно повлияла на его решения. Значит, и результаты такого теста нельзя интерпретировать однозначно. `Удалим ID таких пользователей.`

In [12]:
```
orders = orders.query('visitorId not in @double_id_orders')
```

Теперь посмотрим как распределены `покупатели по группам:`

In [13]:
```
orders.groupby('group')['transactionId'].count()
```

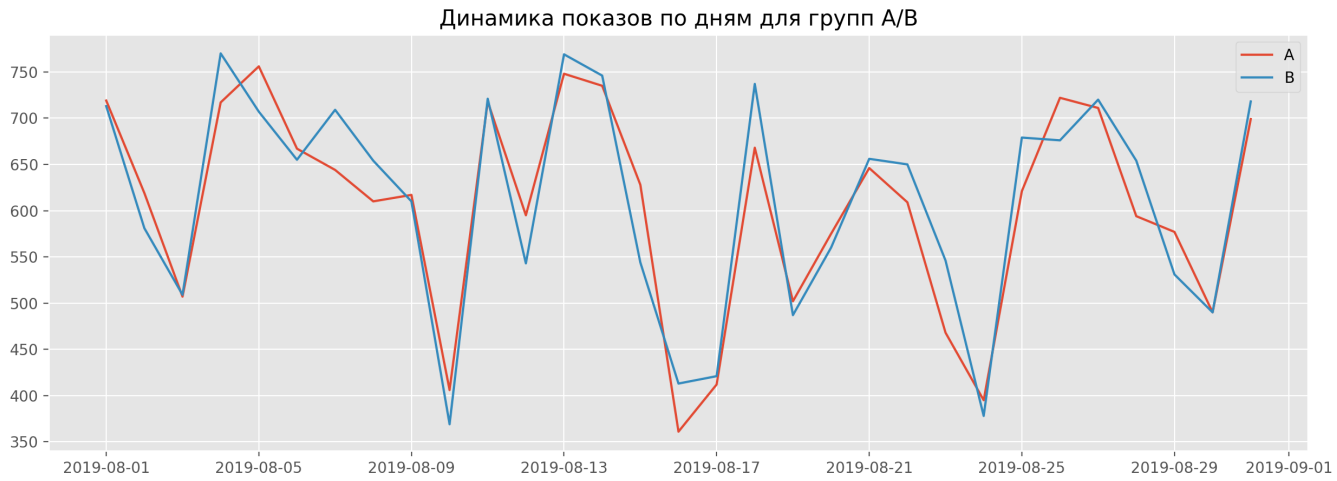
Out[13]:
```
group
A    468
B    548
Name: transactionId, dtype: int64
```

`Покупателей больше в группе B`. При анализе A/B-теста узнаем, как именно это отображается на конверсии.

Посмотрим на динамику посетителей по дням в каждой группе.

In [14]:
```
a_visitors = visitors.query('group =="A"')
b_visitors = visitors.query('group =="B"')

plt.figure(figsize=(15, 5))
plt.plot(a_visitors['date'], a_visitors['visitors'], label='A')
plt.plot(b_visitors['date'], b_visitors['visitors'], label='B')
plt.legend()
#plt.xticks(b_visitors['date'])
plt.title('Динамика показов по дням для групп A/B');
```



У обеих групп число показов сильно "скачет" по дням, но главное, что `существенных различий между группами нет`, обе группы имеют схожее число показов по дням

Приоритезация гипотез

- Применим фреймворк ICE для приоритизации гипотез. Отсортируем их по убыванию приоритета.
- Применим фреймворк RICE для приоритизации гипотез. Отсортируем их по убыванию приоритета.
- Посмотрим, как изменилась приоритизация гипотез при применении RICE вместо ICE

Фреймфорк ICE

In [15]:
```
# вычесления по формуле ICE, сортировка по приоритету:
hypothesis['ICE'] = round((hypothesis['impact']*hypothesis['confidence']) / hypothesis['effor
hypothesis[['hypothesis', 'ICE']].sort_values('ICE', ascending=False)
```

Out[15]:

	hypothesis	ICE
8	Запустить акцию, дающую скидку на товар в день рождения	16.20
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	13.33
7	Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок	11.20
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	8.00
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	7.00
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2.00
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	1.33
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	1.12
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	1.00

Приоритетные гипотезы: под индексами 8, 0, 7:

- Запустить акцию, дающую скидку на товар в день рождения
- Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей
- Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок

Фреймфорк RICE

In [16]:
```
# вычесления по формуле RICE, сортировка по приоритету:
hypothesis['RICE'] = (hypothesis['reach'] * hypothesis['impact'] * hypothesis['confidence'])
hypothesis[['hypothesis', 'RICE']].sort_values('RICE', ascending=False)
```

Out[16]:

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

Приоритетные гипотезы: под индексами 7, 2, 0, 6:

- Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок
- Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа
- Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей
- Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию

При применении RICE изменилась приоретизация гипотез: 7-я становится самой приоритетной, 8-я, самая приоритетная по ICE, не входит в топ-3, зато на втором месте по значимости 2-я гипотеза. Выведем всю таблицу:

In [17]:
```
hypothesis.sort_values('RICE', ascending=False)
```

Out[17]:

	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	11.20	112.0
2	Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа	8	3	7	3	7.00	56.0
0	Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей	3	10	8	6	13.33	40.0
6	Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию	5	3	8	3	8.00	40.0
8	Запустить акцию, дающую скидку на товар в день рождения	1	9	9	5	16.20	16.2
3	Изменить структура категорий, что увеличит конверсию, т.к. пользователи быстрее найдут нужный товар	8	3	3	8	1.12	9.0
1	Запустить собственную службу доставки, что сократит срок доставки заказов	2	5	4	10	2.00	4.0
5	Добавить страницу отзывов клиентов о магазине, что позволит увеличить количество заказов	3	2	2	3	1.33	4.0
4	Изменить цвет фона главной страницы, чтобы увеличить вовлеченность пользователей	3	1	1	1	1.00	3.0

Приоритетность, полученная фреймворком RICE, напрямую зависит от параметра **reach**. У гипотез с индексом 7 и 2 этот параметр оценен в 10 и 8 баллов соответственно, отсюда и вытекает их высокий приоритет. У гипотезы с индексом 8, лидирующей в фреймфорке ICE, параметр reach самый низкий, поэтому в RICE приоритет этой гипотезы опускается на 5-ю строку.

Помимо силы гипотез, важно учитывать, скольких пользователей затронет изменение, которое мы хотим внести. Будем опираться на фреймфорк RICE и определим **следующие приоритетные гипотезы:**

- Добавить форму подписки на все основные страницы, чтобы собрать базу клиентов для email-рассылок
- Добавить блоки рекомендаций товаров на сайт интернет магазина, чтобы повысить конверсию и средний чек заказа
- Добавить два новых канала привлечения трафика, что позволит привлекать на 30% больше пользователей

- Показать на главной странице баннеры с актуальными акциями и распродажами, чтобы увеличить конверсию

Анализ A/B-теста

Перед анализом A/B-теста убедимся, что:

- количество пользователей в различных группах различается не более, чем на 1%;
- применим критерий Манна-Уитни и оценим различия между двумя группами
- проверим совпадают ли даты в таблицах orders и visitors, чтобы в дальнейшем их объединить

In [18]:
```
# подсчет визитов в каждой группе, вычисление разницы между ними:
visitros_sum = visitors.pivot_table(index='group', values='visitors', aggfunc='sum')
display(visitros_sum)
diff = (visitros_sum['visitors'].max()-visitros_sum['visitors'].min())/visitros_sum['visitors
print('Количество пользователей в группах различается на', "{0:.4f}%".format(diff))
print()

# применение критерия Манна-Уитни:
visitors_a = visitors.query('group == "A"')['visitors'].tolist ()
visitors_b = visitors.query('group == "B"')['visitors'].tolist ()

alpha = 0.05
results = st.mannwhitneyu(visitors_a, visitors_b, True, 'less')

print('p-значение: ', "{0:.4f}".format(results.pvalue))
if results.pvalue < alpha:
      print('Разница между группами статистически значима')
else:
    print('Разница между группами статистически не значима')
print()

# сравнение дат в visitors и orders
if visitors['date'].drop_duplicates().tolist().sort() == orders['date'].drop_duplicates().tol
    print('Даты в таблицах visitors и orders совпадают')
else:
    print('Даты в таблицах visitors и orders НЕ совпадают')
```

group	visitors
A	18736
B	18916

```
Количество пользователей в группах различается на 0.0095%

p-значение:  0.3651
Разница между группами статистически не значима

Даты в таблицах visitors и orders совпадают
```

Можем приступать к анализу A/B-теста.

Кумулятивные метрики

Чтобы построить графики, нужно собрать кумулятивные данные. Объявим датафрейм `cumulativeData` со столбцами:

- date — дата;
- group — группа A/B-теста (A или B);

- orders — кумулятивное количество заказов на указанную дату в указанной группе;
- buyers — кумулятивное количество пользователей, совершивших хотя бы один заказ, на указанную дату в указанной группе;
- revenue — кумулятивная выручка на указанную дату в указанной группе (средний чек);
- visitors — кумулятивное количество посетителей интернет-магазина на указанную дату в определённой группе.

In [19]:
```
# массив уникальных пар значений дат и групп теста:
datesGroups = orders[['date','group']].drop_duplicates()

# агрегированные кумулятивные по дням данные о заказах:
ordersAggregated = datesGroups.apply(
    lambda x: orders[np.logical_and(orders['date'] <= x['date'], orders['group'] == x['group'
    'date' : 'max',
    'group' : 'max',
    'transactionId' : pd.Series.nunique,
    'visitorId' : pd.Series.nunique,
    'revenue' : 'sum'}), axis=1).sort_values(by=['date','group']
)

#  агрегированные кумулятивные по дням данные о посетителях:
visitorsAggregated = datesGroups.apply(lambda x: visitors[np.logical_and(visitors['date'] <=

# объединение кумулятивных данных в одной таблице, присвоение названий столбцам:
cumulativeData = ordersAggregated.merge(visitorsAggregated, left_on=['date', 'group'], right_
cumulativeData.columns = ['date', 'group', 'orders', 'buyers', 'revenue', 'visitors']

display(cumulativeData.head())
print('Минимальная дата:', cumulativeData['date'].min())
print('Минимальная дата:', cumulativeData['date'].max())
```

	date	group	orders	buyers	revenue	visitors
0	2019-08-01	A	23	19	142779	719
1	2019-08-01	B	17	17	59758	713
2	2019-08-02	A	42	36	234381	1338
3	2019-08-02	B	40	39	221801	1294
4	2019-08-03	A	66	60	346854	1845

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

Таблица `cumulativeData` сформирована, минимальная и максимальная даты совпадают с исходными, приступаем к построению и анализу графиков

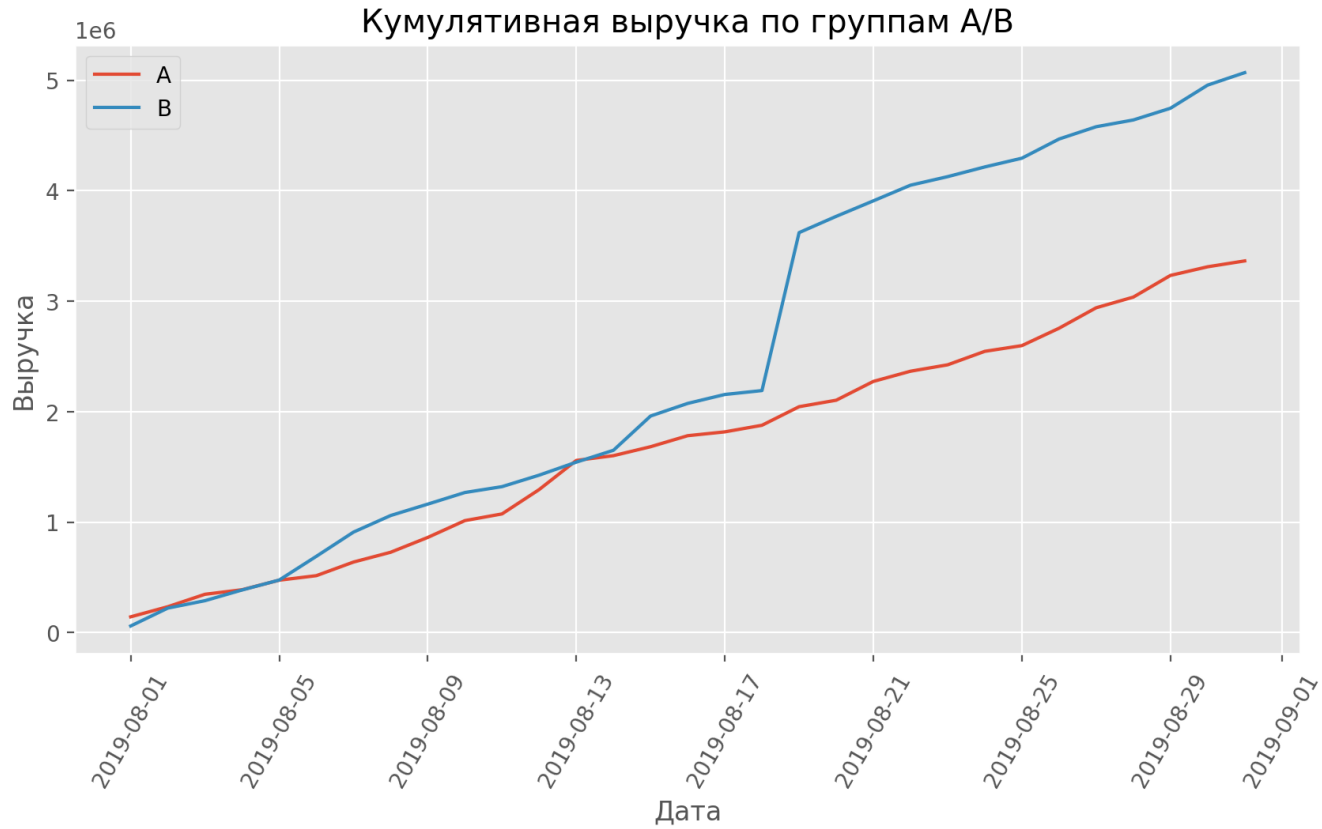
Кумулятивная выручка по группам

In [20]:
```
# датафрейм с кумулятивным количеством заказов и кумулятивной выручкой по дням в группе A
cumulativeRevenueA  = cumulativeData[cumulativeData['group']=='A'][['date','revenue', 'orders

# датафрейм с кумулятивным количеством заказов и кумулятивной выручкой по дням в группе B
cumulativeRevenueB = cumulativeData[cumulativeData['group']=='B'][['date','revenue', 'orders'

plt.figure(figsize=(10, 5))
plt.plot(cumulativeRevenueA['date'], cumulativeRevenueA['revenue'], label='A')
plt.plot(cumulativeRevenueB['date'], cumulativeRevenueB['revenue'], label='B')
plt.grid(True)
plt.xticks(rotation=60)
plt.ylabel('Выручка')
plt.xlabel('Дата')
```

```
plt.title('Кумулятивная выручка по группам A/B')
plt.legend();
```



Практически за весь исследуемый период `кумулятивная выручка группы B выше, чем у A.`

Обратим внимание на резкий скачок выручки группы B на отрезке 17-21 марта. Такой резкий рост может быть связан либо со всплеском числа заказов, либо, что более вероятно, `кумулятивная выручка повысилась из-за очень дорогого заказа/заказов.` На этапе предобработки мы бегло смотрели на распределение сумм заказов и обнаружили аномальные суммы > 1.2 млн.

Посмотрим на самые дорогие заказы в исходной таблице orders:

In [21]:
```
orders.sort_values(by='revenue', ascending=False).head()
```

Out[21]:

	transactionId	visitorId	date	revenue	group
425	590470918	1920142716	2019-08-19	1294500	B
1196	3936777065	2108080724	2019-08-15	202740	B
1136	666610489	1307669133	2019-08-13	92550	A
744	3668308183	888512513	2019-08-27	86620	B
743	3603576309	4133034833	2019-08-09	67990	A

Гипотеза подтвердилась: в группе B 19.08 был оформлен `аномально дорогой заказ стоимостью 1.294.500, именно он "вытягивает" кумулятивную выручку группы вверх.`

Заметим, что второй по дороговизне заказ на сумму > 200 тыс. также приходится на группу B. Заказ совершен 15.08, но при этом мы не наблюдаем на графике такого же резкого скачка, как 19.08.

После аномально дорогого заказа кумулятивная выручка растет плавно, без резкого роста.

В целом, за исключением 13.08, **кумулятивная выручка группы B стабильно выше, чем у группы A**

Кумулятивный средний чек по группам

In [22]:
```
plt.figure(figsize=(10,4))
plt.plot(cumulativeRevenueA['date'], cumulativeRevenueA['revenue']/cumulativeRevenueA['orders
plt.plot(cumulativeRevenueB['date'], cumulativeRevenueB['revenue']/cumulativeRevenueB['orders
plt.grid(True)
plt.xticks(rotation=50)
plt.title('Кумулятивный средный чек по группам A/B')
plt.ylabel('Средний чек')
plt.xlabel('Дата')
plt.legend();
```

Аномальный заказ в группе B ожидаемо отражается и на среднем чеке группы, также видим, что после пикового значения 19.08 `кумулятивный средний чек у B снижается` - очевидно, настолько дорогих заказов больше не совершается, соответственно, средний чек не может оставаться хотя бы на том же уровне.

Ранее мы отмечали, что 13.08 кумулятивная выручка у обоих групп одинаковая, можем объяснить это ростом среднего чека в группе A: в этот день в группе были сделаны более дорогие покупки.

Если рассматривать последние две недели, то в целом `средний чек группы A держится на одном и том же уровне и менее подвержен колебаниям`

Нельзя сделать каких-то однозначных выводов, **кумулятивный средний чек группы B сильно зависит от аномального заказа**.

Убедимся в этом, изучив средний чек по группам без накопления:

In [23]:
```
mean_b = orders.query('group == "B"').groupby('date').agg({'revenue':'mean'})
mean_a = orders.query('group == "A"').groupby('date').agg({'revenue':'mean'})

plt.figure(figsize=(10,4))
plt.plot(mean_a['revenue'], label='A')
plt.plot(mean_b['revenue'], label='B')
plt.grid(True)
plt.xticks(rotation=50)
plt.title('Средний чек по группам A/B (без накопления)')
plt.ylabel('Средний чек')
plt.xlabel('Дата')
plt.legend();
```

Рассматривая средний чек без накопления, видим, что `средний чек чаще выше у группы A`. Однако нельзя сказать, особенно по последним датам, что разница существенна. Позже мы оценим статистическую значимость разницы в средних чеках между группами.

Относительное изменение кумулятивного среднего чека группы B к группе A

In [24]:
```
# собираем данные в одном датафрейме
mergedCumulativeRevenue = cumulativeRevenueA.merge(cumulativeRevenueB, left_on='date', right_

# строим отношение средних чеков
plt.figure(figsize=(10,4))
plt.plot(mergedCumulativeRevenue['date'], (mergedCumulativeRevenue['revenueB']/mergedCumulati
plt.grid(True)
plt.xticks(rotation=50)
plt.title('Относительного изменения кумулятивного среднего чека группы B к группе A')
plt.ylabel('Прирост')
plt.xlabel('Дата')
# добавляем ось X
plt.axhline(y=0, color='black', linestyle='--');
```

В целом, график выглядит ожидаемо, т.к. является производным от графика кумулятивного среднего чека по группам. Видим отрицательную точку 13.08, когда средний чек у A выше, чем у B,

видим все тот же резкий рост 19.08.

Но можем отметить, что `более наглядно продемонстрировано падение среднего чека в группе B` после аномального чека - начиная с 25.08 разница между группами заметно снижается.

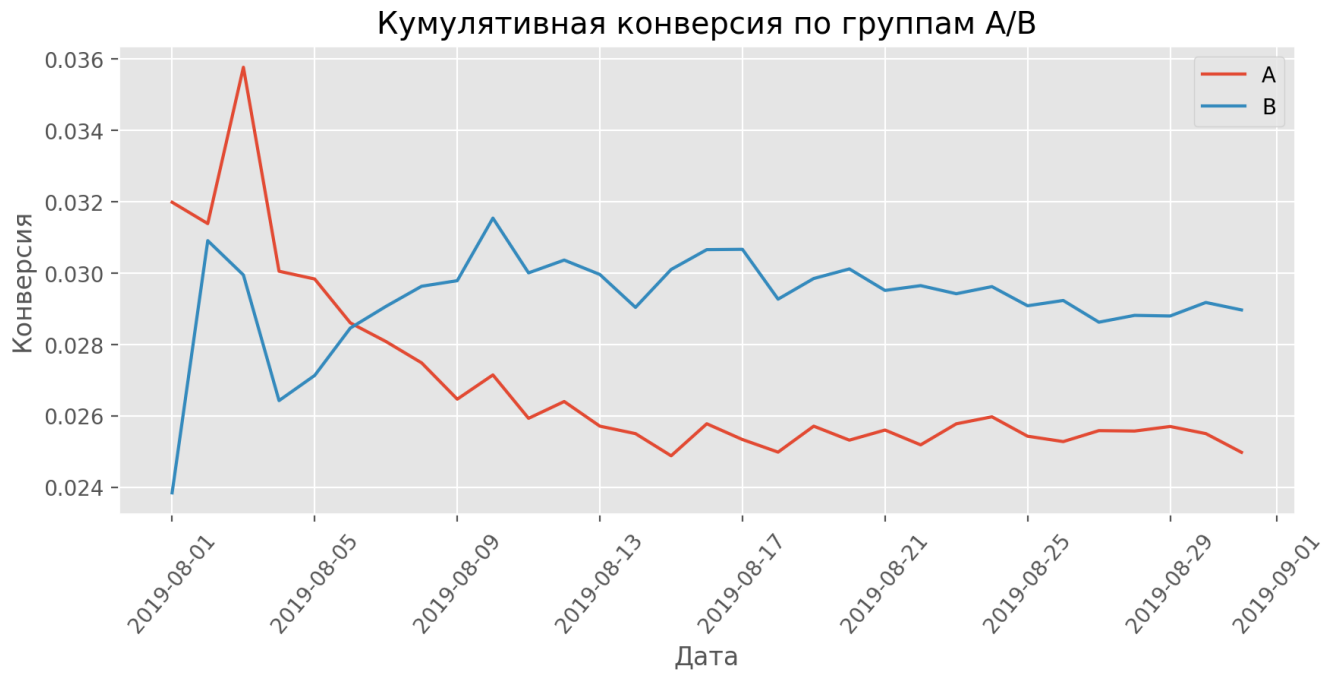
Как и в предыдущем пункте, **на этом этапе не стоит делать каких-то однозначных выводов в разнице кумулятивного среднего чека между группами.**

Кумулятивная конверсия по группам

In [25]:
```
plt.figure(figsize=(10,4))
# считаем кумулятивную конверсию
cumulativeData['conversion'] = cumulativeData['orders']/cumulativeData['visitors']
# отделяем данные по группе A
cumulativeDataA = cumulativeData[cumulativeData['group']=='A']

# отделяем данные по группе B
cumulativeDataB = cumulativeData[cumulativeData['group']=='B']

# строим графики
plt.plot(cumulativeDataA['date'], cumulativeDataA['conversion'], label='A')
plt.plot(cumulativeDataB['date'], cumulativeDataB['conversion'], label='B')
plt.grid(True)
plt.xticks(rotation=50)
plt.title('Кумулятивная конверсия по группам A/B')
plt.ylabel('Конверсия')
plt.xlabel('Дата')
plt.legend();
```



Конверсия уже не зависит от аномалий и по ней можно сделать некоторые выводы.

`Определенно лидирует группа B:` начиная с 6.08 конверсия этой группы стабильно выше, чем в группе A. Конверсия B неравномерна, под конец периода сдает свои позиции, но с 6.08 до конца периода:

- кумулятивная конверсия группы B располагается в промежутке 2.8%-3.2%
- кумулятивная конверсия группы A не превышает 2.7% (за исключением первых дней)

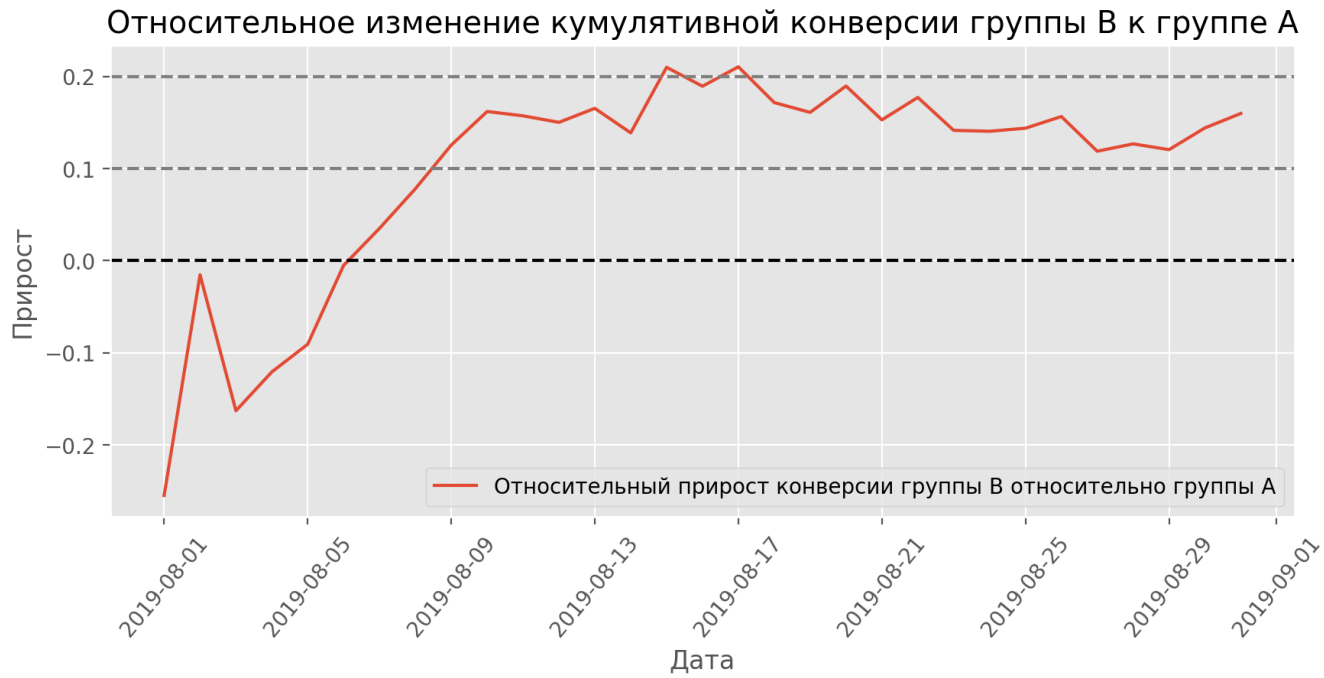
Клиенты из группы B лучше конвертируются в покупателей, чем клиенты группы A

Относительное изменение кумулятивной конверсии группы B к группе A

In [26]:
```
# собираем данные в одном датафрейме
mergedCumulativeConversions = cumulativeDataA[['date','conversion']]\
    .merge(cumulativeDataB[['date','conversion']], left_on='date', right_on='date', how='left

plt.figure(figsize=(10,4))
plt.plot(mergedCumulativeConversions['date'], \
         mergedCumulativeConversions['conversionB']/mergedCumulativeConversions['conversionA'
         label="Относительный прирост конверсии группы B относительно группы A")

plt.axhline(y=0, color='black', linestyle='--')
plt.axhline(y=0.2, color='grey', linestyle='--')
plt.axhline(y=0.1, color='grey', linestyle='--')
plt.grid(True)
plt.xticks(rotation=50)
plt.title('Относительное изменение кумулятивной конверсии группы B к группе A')
plt.ylabel('Прирост')
plt.xlabel('Дата')
plt.legend();
```



В целом, результаты не удивляют: мы уже выявили, что начиная с 6.08 конверсия группы B стабильно выше, отсюда и относительный прирост группы B к A.

Можем отметить, что начиная с 8.08, **конверсия группы B стабильно выше конверсии группы A не менее, чем на 10%.**

Стоимость заказов и количество заказов по пользователям. Границы для определения аномалий

Мы уже сталкивались с аномально дорогими заказами, искажающими выводы о среднем чеке, также нам могут помешать и "аномальные покупатели", совершающие покупки чаще, чем остальные.

Для получения более точного представления о статистической значимости `необходимо установить границы для определени таких аномалий`, чтобы была возможность провести расчеты и для "сырых", исходных данных, и для данных, отфильтрованных от выбросов.

Точечный график числа заказов. Граница для определения аномальных пользователей

In [27]:
```
ordersByUsers = (
    orders.groupby('visitorId', as_index=False)
    .agg({'transactionId': 'nunique'})
)
ordersByUsers.columns = ['userId', 'orders']

plt.figure(figsize=(10,4))
x_values = pd.Series(range(0, len(ordersByUsers)))
plt.scatter(x_values, ordersByUsers['orders'], alpha=0.5)
plt.grid(True)
plt.xticks(rotation=50)
plt.title('Распределение числа заказов по пользователям')
plt.ylabel('Число заказов');
```

Абсолютное большинство пользователей ограничивается одним заказом в месяц, больше трех заказов - редкость.

Встречаются пользователи, успевшие за календарный месяц сделать больше 10 заказов.

Определим границы для аномальных пользователей: посчитаем 95-99 перцентили количества заказов на пользователя.

In [28]:
```
print('95, 96, 97, 98, 99 перцентили:', np.percentile(ordersByUsers['orders'], [95,96,97,98,9
```

```
95, 96, 97, 98, 99 перцентили: [1. 1. 2. 2. 2.]
```

98% пользователей совершают до двух заказов, можем **считать аномалией 3 и более заказов на одного пользователя**

Точечный график стоимостей заказов. Граница для определения аномальных заказов

In [29]:
```
plt.figure(figsize=(10,4))
x_values = pd.Series(range(0, len(orders['revenue'])))
plt.scatter(x_values, orders['revenue'], alpha=0.5)
plt.grid(True)
plt.xticks(rotation=50)
plt.title('Распределение стоимости заказов')
plt.ylabel('Сумма заказа');
```

Снова мешает самый дорогой заказ в выборке, из-за которого мы не можем более детально рассмотреть распределение. Посмотрим на тот же график без этого заказа

In [30]:
```
plt.figure(figsize=(10,4))
x_values = pd.Series(range(0, len(orders.query('revenue < 1_000_000')['revenue'])))
plt.scatter(x_values, orders.query('revenue < 1_000_000')['revenue'], alpha=0.5)
plt.grid(True)
plt.xticks(rotation=50)
plt.title('Распределение стоимости заказов')
plt.ylabel('Сумма заказа');
```

`Стоимость большинства заказов не превышает суммы в 25.000`. Более дорогие заказы встречаются намного реже, заказы, дороже 50.000 - единицы

Определим границы для определения аномальных сумм заказов: посчитаем 95-99 перцентили сумм на пользователя.

In [31]:
```
print('95, 96, 97, 98, 99 перцентили:', np.percentile(orders['revenue'], [95,96,97,98,99]))
```

```
95, 96, 97, 98, 99 перцентили: [26785. 30649. 34792. 42353. 53904.]
```

Остановимся на той же границе, что определели для количества заказов: **только 2% покупателей совершают заказы дороже 44133.2, определим эту сумму как верхнюю границу.**

Статистическая значимость

Рассчитаем статистическую значимость различий в среднем количестве заказов на посетителя (конверсии) и различий в среднем чеке заказа между группами A/B. Сначала проанализируем "сырые" данные, затем отфильтрованные от аномалий.

Создадим таблицу `data` для расчета статистической значимости:

In [32]:
```
visitorsADaily = visitors[visitors['group'] == 'A'][['date', 'visitors']]
visitorsADaily.columns = ['date', 'visitorsPerDateA']

visitorsACummulative = visitorsADaily.apply(
    lambda x: visitorsADaily[visitorsADaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateA': 'sum'}
    ),
    axis=1,
)
visitorsACummulative.columns = ['date', 'visitorsCummulativeA']

visitorsBDaily = visitors[visitors['group'] == 'B'][['date', 'visitors']]
visitorsBDaily.columns = ['date', 'visitorsPerDateB']

visitorsBCummulative = visitorsBDaily.apply(
    lambda x: visitorsBDaily[visitorsBDaily['date'] <= x['date']].agg(
        {'date': 'max', 'visitorsPerDateB': 'sum'}
    ),
    axis=1,
)
visitorsBCummulative.columns = ['date', 'visitorsCummulativeB']

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']

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',
]

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']

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',
]
```

```
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.head()
```

Out[32]:

	date	ordersPerDateA	revenuePerDateA	ordersPerDateB	revenuePerDateB	ordersCummulativeA	revenueC
0	2019-08-01	23	142779	17	59758	23	
1	2019-08-02	19	91602	23	162043	42	
2	2019-08-03	24	112473	14	67049	66	
3	2019-08-04	11	41176	14	96890	77	
4	2019-08-05	22	86383	21	89908	99	

Таблица для рассчета статистической значимости готова. Устройство 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.

Далее создадим переменные `ordersByUsersA` и `ordersByUsersB`. В них для пользователей, которые заказывали хотя бы 1 раз, укажем число совершённых заказов.

In [33]:
```
ordersByUsersA = (
    orders[orders['group'] == 'A']
    .groupby('visitorId', as_index=False)
    .agg({'transactionId': pd.Series.nunique})
)
ordersByUsersA.columns = ['visitorId', 'orders']
```

```
ordersByUsersB = (
    orders[orders['group'] == 'B']
    .groupby('visitorId', as_index=False)
    .agg({'transactionId': pd.Series.nunique})
)
ordersByUsersB.columns = ['visitorId', 'orders']
```

Теперь можем приступать к расчету статистической значимости

Статистическая значимость различий конверсии между группами по «сырым» данным

- **Нулевая гипотеза H0:** Между группами нет статистически значимых различий **(группы равны)**
- **Альтернативная гипотеза H1:** Между группами есть статистически значимые различия **(группы не равны)**
- **Уровень статистической значимости:** 0.05

In [34]:
```
sampleA = pd.concat([ordersByUsersA['orders'],pd.Series(0, index=np.arange(data['visitorsPerD
    .sum() - len(ordersByUsersA['orders'])), name='orders')],axis=0)

sampleB = pd.concat([ordersByUsersB['orders'],pd.Series(0, index=np.arange(data['visitorsPerD
    .sum() - len(ordersByUsersB['orders'])), name='orders')],axis=0)

alpha = 0.05

p_value = st.mannwhitneyu(sampleA, sampleB)[1]
print('p_value:', "{0:.4f}".format(p_value))
print('Относительное различие в конверсии между группами',"{0:.4f}".format(sampleB.mean() / s
if p_value < alpha:
    print('Отвергаем нулевую гипотезу, между группами есть статистически значимые различия')
else:
    print('Не отвергаем нулеву гипотезу, между группами нет статистически значимых различий')
```

```
p_value: 0.0110
Относительное различие в конверсии между группами 0.1598
Отвергаем нулевую гипотезу, между группами есть статистически значимые различия
```

Ранее мы уже замечали, что с определенного периода конверсия группы B стабильно выше, чем конверсия группы A минимум на 10%. Теперь подтвердили это статистически.

Относительный прирост конверсии группы B к группе A: ~16%%. P-value значительно ниже установленной статистической значимости

статистическая значимость различий в среднем чеке заказа между группами по «сырым» данным

- **Нулевая гипотеза H0:** Между группами нет статистически значимых различий **(группы равны)**
- **Альтернативная гипотеза H1:** Между группами есть статистически значимые различия **(группы не равны)**
- **Уровень статистической значимости:** 0.05

In [35]:
```
p_value = st.mannwhitneyu(orders[orders['group']=='A']['revenue'], orders[orders['group']=='B

print('p_value:', '{0:.3f}'.format(p_value))
print('Относительное различие в среднем чеке между группами:',
      '{0:.3f}'.format(orders[orders['group']=='B']['revenue'].mean()/orders[orders['group']=
```

```
if p_value < alpha:
    print('Отвергаем нулевую гипотезу, между группами есть статистически значимые различия')
else:
    print('Не отвергаем нулевую гипотезу, между группами нет статистически значимых различий'
```

```
p_value: 0.829
Относительное различие в среднем чеке между группами: 0.287
Не отвергаем нулевую гипотезу, между группами нет статистически значимых различий
```

Несмотря на то, что средний чек группы B значительно выше (на 28,7%), **статистически значимых различий между группами нет**, P-value сильно превышает установленный уровень статистической значимости.

Аномально дорогой чек не дает оснований полагать, что глобально средние чеки в группах ощутимо различаются.

Статистическая значимость различий конверсии между группами по «очищенным» данным

Исходя из определенных ранее границ, сохраним аномальные данные в соответствующих переменных:

- `usersWithManyOrders` - число заказов > 2
- `usersWithExpensiveOrders` - сумма заказа > 44133.2

In [36]:
```
extra_order = 3
extra_revenue = 44133.2

usersWithManyOrders = pd.concat(
    [
        ordersByUsersA[ordersByUsersA['orders'] >= extra_order]['visitorId'],
        ordersByUsersB[ordersByUsersB['orders'] >=extra_order]['visitorId'],
    ],
    axis=0,
)
usersWithExpensiveOrders = orders[orders['revenue'] > extra_revenue]['visitorId']
abnormalUsers = (
    pd.concat([usersWithManyOrders, usersWithExpensiveOrders], axis=0)
    .drop_duplicates()
    .sort_values()
)

print('Число "аномальных" пользователей:', len(abnormalUsers))
```

```
Число "аномальных" пользователей: 25
```

Исключим таких пользователей и рассчитаем статистическую значимость для "очищенных" данных

In [37]:
```
sampleAFiltered = pd.concat(
    [
        ordersByUsersA[
            np.logical_not(ordersByUsersA['visitorId'].isin(abnormalUsers))
        ]['orders'],
        pd.Series(
            0,
            index=np.arange(
                data['visitorsPerDateA'].sum() - len(ordersByUsersA['orders'])
            ),
            name='orders',
        ),
    ],
    axis=0,
```

```
)

sampleBFiltered = pd.concat(
    [
        ordersByUsersB[
            np.logical_not(ordersByUsersB['visitorId'].isin(abnormalUsers))
        ]['orders'],
        pd.Series(
            0,
            index=np.arange(
                data['visitorsPerDateB'].sum() - len(ordersByUsersB['orders'])
            ),
            name='orders',
        ),
    ],
    axis=0,
)
```

- **Нулевая гипотеза H0:** Между группами нет статистически значимых различий **(группы равны)**
- **Альтернативная гипотеза H1:** Между группами есть статистически значимые различия **(группы не равны)**
- **Уровень статистической значимости:** 0.05

In [38]:
```
p_value = st.mannwhitneyu(sampleAFiltered, sampleBFiltered)[1]

print('p_value:', '{0:.3f}'.format(p_value))
print('Относительное различие в конверсии между группами:', '{0:.3f}'.format(sampleBFiltered.

if p_value < alpha:
    print('Отвергаем нулевую гипотезу, между группами есть статистически значимые различия')
else:
    print('Не отвергаем нулевую гипотезу, между группами нет статистически значимых различий'
```

```
p_value: 0.005
Относительное различие в конверсии между группами: 0.201
Отвергаем нулевую гипотезу, между группами есть статистически значимые различия
```

По очищенным данным `конверсия в группе B выше, чем в группе A на 20.1%,` результат лучше, чем по "сырым" данным.

Статистическая значимость различий в среднем чеке заказа между группами по «очищенным» данным

In [39]:
```
p_value = st.mannwhitneyu(
            orders[
                np.logical_and(
                    orders['group'] == 'A',
                    np.logical_not(orders['visitorId'].isin(abnormalUsers)),
                )
            ]['revenue'],
            orders[
                np.logical_and(
                    orders['group'] == 'B',
                    np.logical_not(orders['visitorId'].isin(abnormalUsers)),
                )
            ]['revenue'],
        )[1]



print('p_value:', '{0:.3f}'.format(p_value))

print('Относительное различие в в среднем чеке между группами:',
```

```
    "{0:.3f}".format(
        orders[
            np.logical_and(
                orders['group'] == 'B',
                np.logical_not(orders['visitorId'].isin(abnormalUsers)),
            )
        ]['revenue'].mean()
        / orders[
            np.logical_and(
                orders['group'] == 'A',
                np.logical_not(orders['visitorId'].isin(abnormalUsers)),
            )
        ]['revenue'].mean()
        - 1
    )
)

if p_value < alpha:
    print('Отвергаем нулевую гипотезу, между группами есть статистически значимые различия')
else:
    print('Не отвергаем нулевую гипотезу, между группами нет статистически значимых различий'
```

```
p_value: 0.996
Относительное различие в в среднем чеке между группами: 0.031
Не отвергаем нулевую гипотезу, между группами нет статистически значимых различий
```

Избавившись от аномально дорогих заказов, отмечаем, что `между группами разница всего в 3,7%.`

Нет оснований полагать, что **средний чек в группах значимо отличается**

Решение по результатам теста

Конверсия

Между группами A и B **есть** статистически значимые различия. **Относительный выигрыш группы B над A**:

- по сырым данным: 13,81%
- по очищенным данным: 18,5%

Средний чек

Между группами **нет** статистически значимых различий. Относительные различия между группами:

- по сырым данным: 25,9% (за счет одного аномально дорогого чека)
- по очищенным: 3,7%

Иными словами, группы ощутимо различаются по конверсии, но имеют близкий по значению средний чек.

На основании значимого преимущества в конверсии, останавливаем тест и фиксируем победу группы B