Набор данных о продажах электронной коммерции через Amazon

https://www.kaggle.com/datasets/thedevastator/unlock-profits-with-e-commerce-sales-data

Анализ и максимизация эффективности онлайн-бизнеса

In [160…
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.dates import DateFormatter
```

In [161…
```
# Переменные

a = 1000000 # Размер строк для head

b = 20 # Переменная для размера графиков
```

In [162…
```
# Укажите путь к файлу CSV
file_path = './5_Набор_данных_о_продажах_электронной_коммерции.csv'

# Исключить определенные столбцы при чтении файла CSV
input_raw = pd.read_csv(file_path, usecols=lambda col: col not in ['Unnamed: 22'])

# Преобразование колонки "Date" в формат datetime
input_raw['Date'] = pd.to_datetime(input_raw['Date'], format='%m-%d-%y')

input_raw.head(a)
```

Out[162]:

	index	Order ID	Date	Status	Fulfilment	Sales Channel	ship-service-level	Style	SKU	Category	...	Qty	currency	Amour
0	0	405-8078784-5731545	2022-04-30	Cancelled	Merchant	Amazon.in	Standard	SET389	SET389-KR-NP-S	Set	...	0	INR	647.6
1	1	171-9198151-1101146	2022-04-30	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	JNE3781	JNE3781-KR-XXXL	kurta	...	1	INR	406.0
		404-	2022-											

2	2	0687676-7273146	04-30	Shipped	Amazon	Amazon.in	Expedited	JNE3371	JNE3371-KR-XL	kurta	...	1	INR	329.0
3	3	403-9615377-8133951	2022-04-30	Cancelled	Merchant	Amazon.in	Standard	J0341	J0341-DR-L	Western Dress	...	0	INR	753.3
4	4	407-1069790-7240320	2022-04-30	Shipped	Amazon	Amazon.in	Expedited	JNE3671	JNE3671-TU-XXXL	Top	...	1	INR	574.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
128970	128970	406-6001380-7673107	2022-05-31	Shipped	Amazon	Amazon.in	Expedited	JNE3697	JNE3697-KR-XL	kurta	...	1	INR	517.0
128971	128971	402-9551604-7544318	2022-05-31	Shipped	Amazon	Amazon.in	Expedited	SET401	SET401-KR-NP-M	Set	...	1	INR	999.0
128972	128972	407-9547469-3152358	2022-05-31	Shipped	Amazon	Amazon.in	Expedited	J0157	J0157-DR-XXL	Western Dress	...	1	INR	690.0
128973	128973	402-6184140-0545956	2022-05-31	Shipped	Amazon	Amazon.in	Expedited	J0012	J0012-SKD-XS	Set	...	1	INR	1199.0
128974	128974	408-7436540-8728312	2022-05-31	Shipped	Amazon	Amazon.in	Expedited	J0003	J0003-SET-S	Set	...	1	INR	696.0

128975 rows × 23 columns

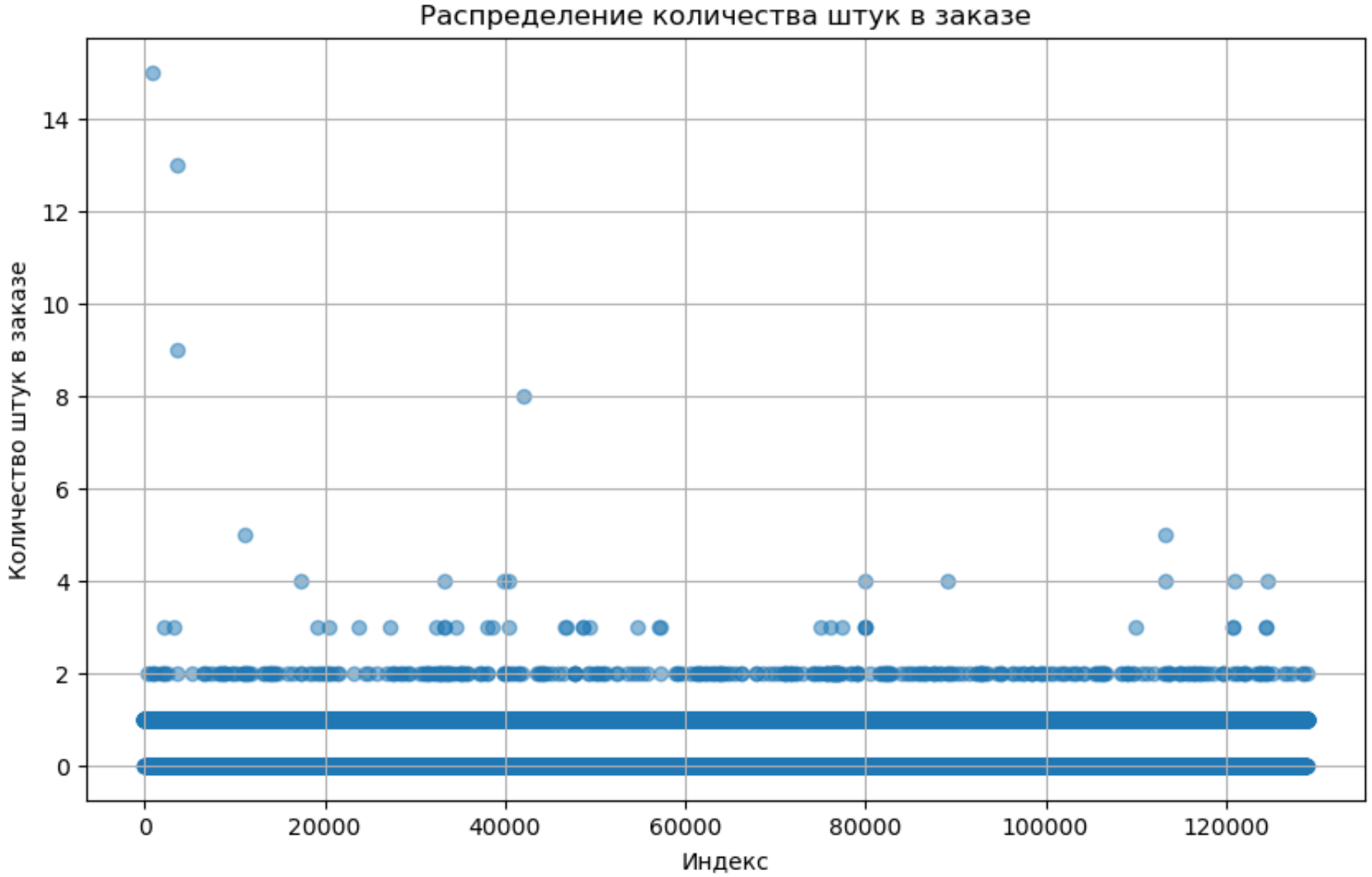
In [163…
```
#input_raw.dtypes
```

In [164…
```
input_raw.describe()
```

Out[164]:

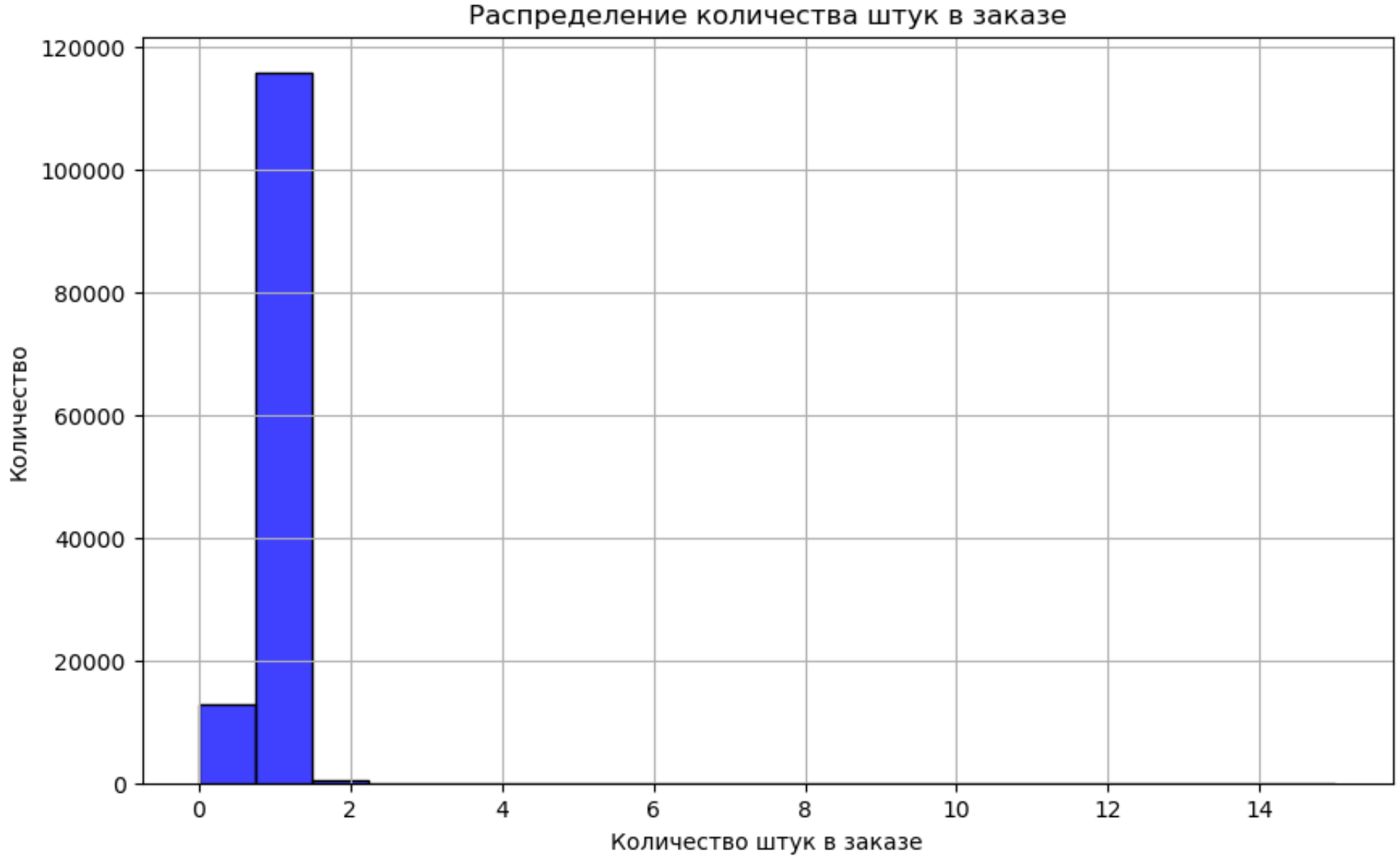
	index	Date	Qty	Amount	ship-postal-code
count	128975.000000	128975	128975.000000	121180.000000	128942.000000
mean	64487.000000	2022-05-12 11:49:27.951928576	0.904431	648.561465	463966.236509
min	0.000000	2022-03-31 00:00:00	0.000000	0.000000	110001.000000
25%	32243.500000	2022-04-20 00:00:00	1.000000	449.000000	382421.000000
50%	64487.000000	2022-05-10 00:00:00	1.000000	605.000000	500033.000000
75%	96730.500000	2022-06-04 00:00:00	1.000000	788.000000	600024.000000
max	128974.000000	2022-06-29 00:00:00	15.000000	5584.000000	989898.000000
std	37232.019822	NaN	0.313354	281.211687	191476.764941

In [165…
```
df = input_raw.copy(deep=True)

plt.figure(figsize=(10, 6))
plt.scatter(df.index, df['Qty'], alpha=0.5)

plt.title('Распределение количества штук в заказе')
plt.xlabel('Индекс')
plt.ylabel('Количество штук в заказе')
plt.grid(True)
plt.show()
```


Распределение количества штук в заказе
Количество штук в заказе
Индекс
0
2
4
6
8
10
12
14
0
20000
40000
60000
80000
100000
120000

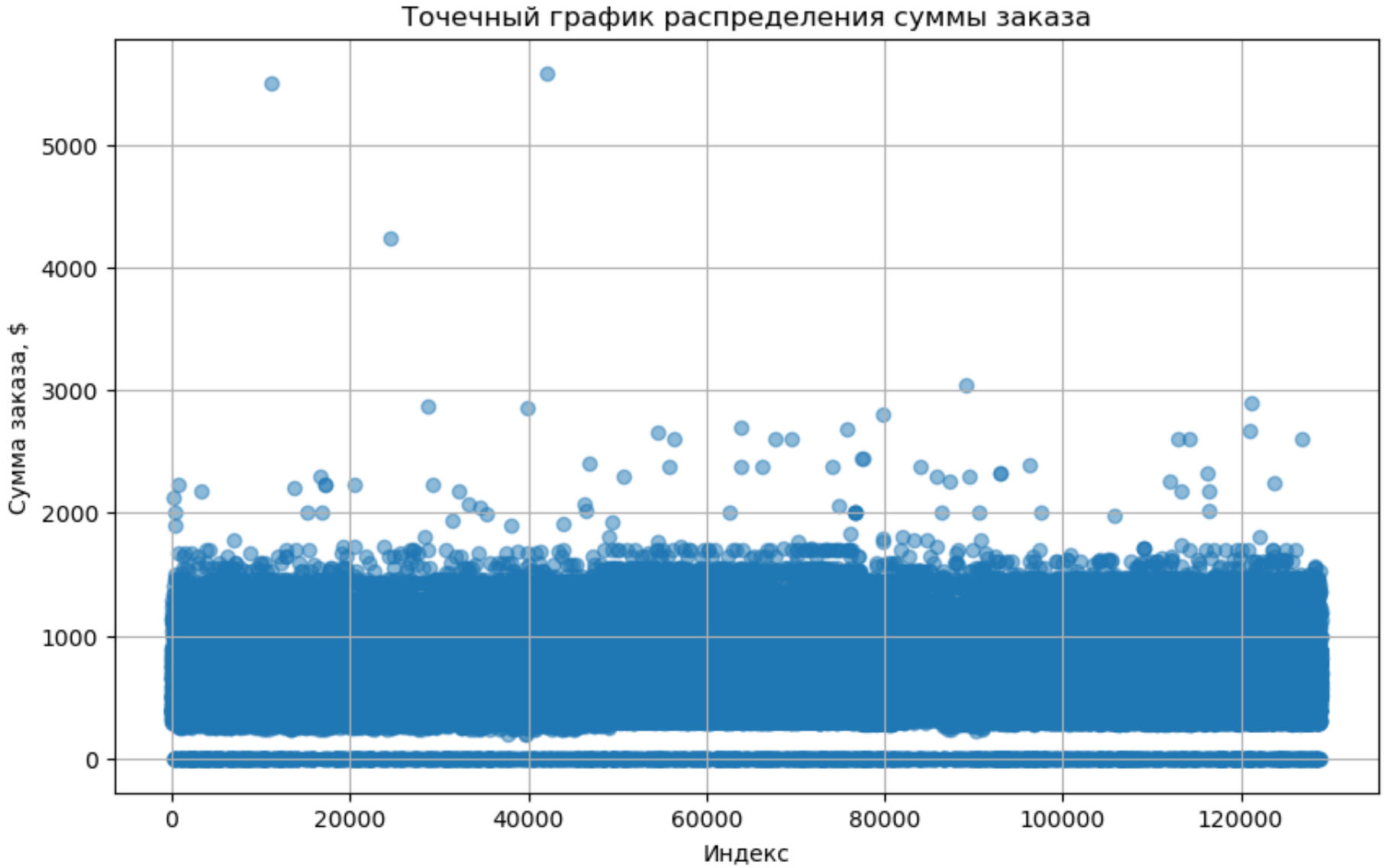
In [166…

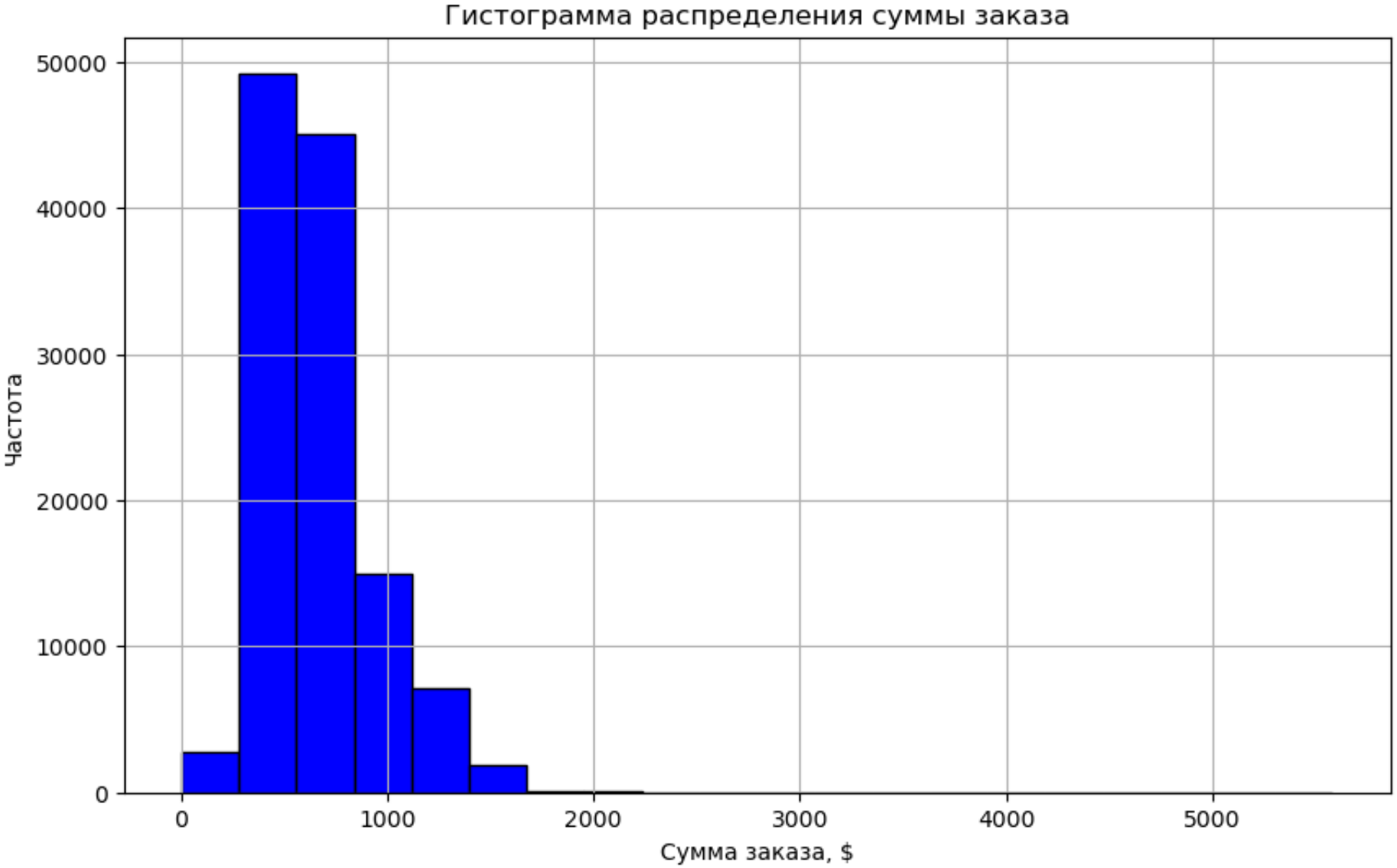
```
df = input_raw.copy(deep=True)

# Построение гистограммы
plt.figure(figsize=(10, 6))
sns.histplot(df['Qty'], bins=20, kde=False, color='blue')
plt.title('Распределение количества штук в заказе')
plt.xlabel('Количество штук в заказе')
plt.ylabel('Количество')
plt.grid(True)
plt.show()
```


Распределение количества штук в заказе
Количество
120000
100000
80000
60000
40000
20000
0
0
2
4
6
8
10
12
14
Количество штук в заказе

In [167…

```
df = input_raw.copy(deep=True)
plt.figure(figsize=(10, 6))
plt.scatter(df.index, df['Amount'], alpha=0.5)
plt.title('Точечный график распределения суммы заказа')
plt.xlabel('Индекс')
plt.ylabel('Сумма заказа, $')
plt.grid(True)
plt.show()

# Гистограмма
plt.figure(figsize=(10, 6))
plt.hist(df['Amount'], bins=20, color='blue', edgecolor='black')
plt.title('Гистограмма распределения суммы заказа')
plt.xlabel('Сумма заказа, $')
plt.ylabel('Частота')
plt.grid(True)
plt.show()
```


Точечный график распределения суммы заказа
Сумма заказа, $
5000
4000
3000
2000
1000
0
0
20000
40000
60000
80000
100000
120000
Индекс


Гистограмма распределения суммы заказа
Частота
50000
40000
30000
20000
10000
0
0
1000
2000
3000
4000
5000
Сумма заказа, $

In [168…

```python
df = input_raw.copy(deep=True)

# Убираем строки, где в колонке 'Status' есть указанные значения
df_filtered = df[df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Подсчет количества строк
count_rows = len(df_filtered)

# Форматирование строки с использованием f-строк
formatted_count_rows = f'{count_rows:,}'

# Вывод результата
print(f"Количество заказов после фильтрации: {formatted_count_rows}")
```

```
Количество заказов после фильтрации: 107,546
```

In [169…

```python
DF = input_raw.copy(deep=True)

statuses_to_remove = ['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up']
DF_filtered = DF[DF['Status'].isin(statuses_to_remove)]

total_amount = DF_filtered['Amount'].sum()

# Форматирование строки с использованием f-строк
formatted_total_amount = f'{total_amount:,.2f}'

print(f'Общая сумма продаж: {formatted_total_amount} $')
```

```
Общая сумма продаж: 69,636,322.00 $
```

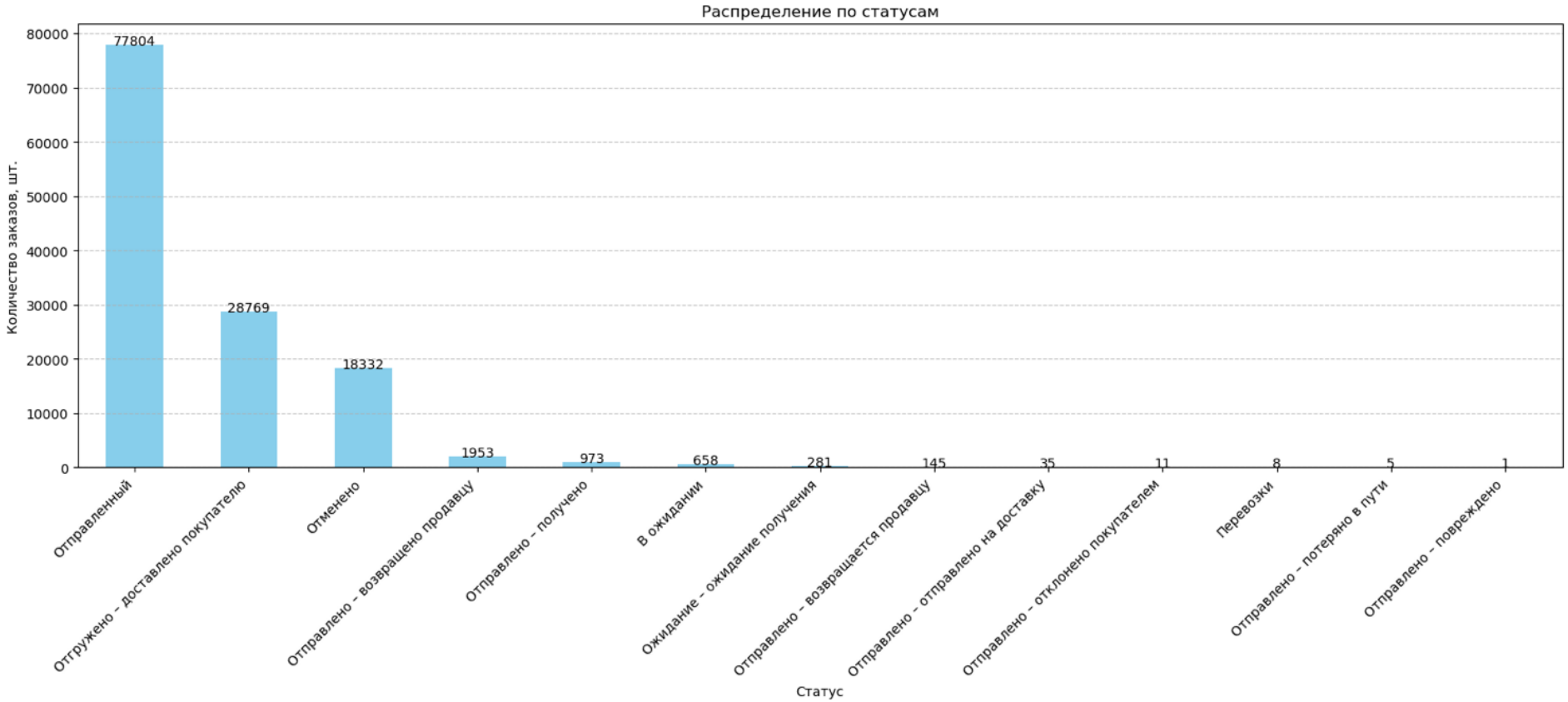
In [170…

```python
DF = input_raw.copy(deep=True)

statuses_to_remove = ['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up']
DF_filtered = DF[DF['Status'].isin(statuses_to_remove)]

# Используйте метод 'mean' для вычисления среднего значения
total_amount = DF_filtered['Amount'].mean()

# Округление до двух знаков после точки
rounded_total_amount = round(total_amount, 2)

print('Средняя сумма продаж: ', rounded_total_amount, ' $')
```

```
Средняя сумма продаж:  648.81  $
```

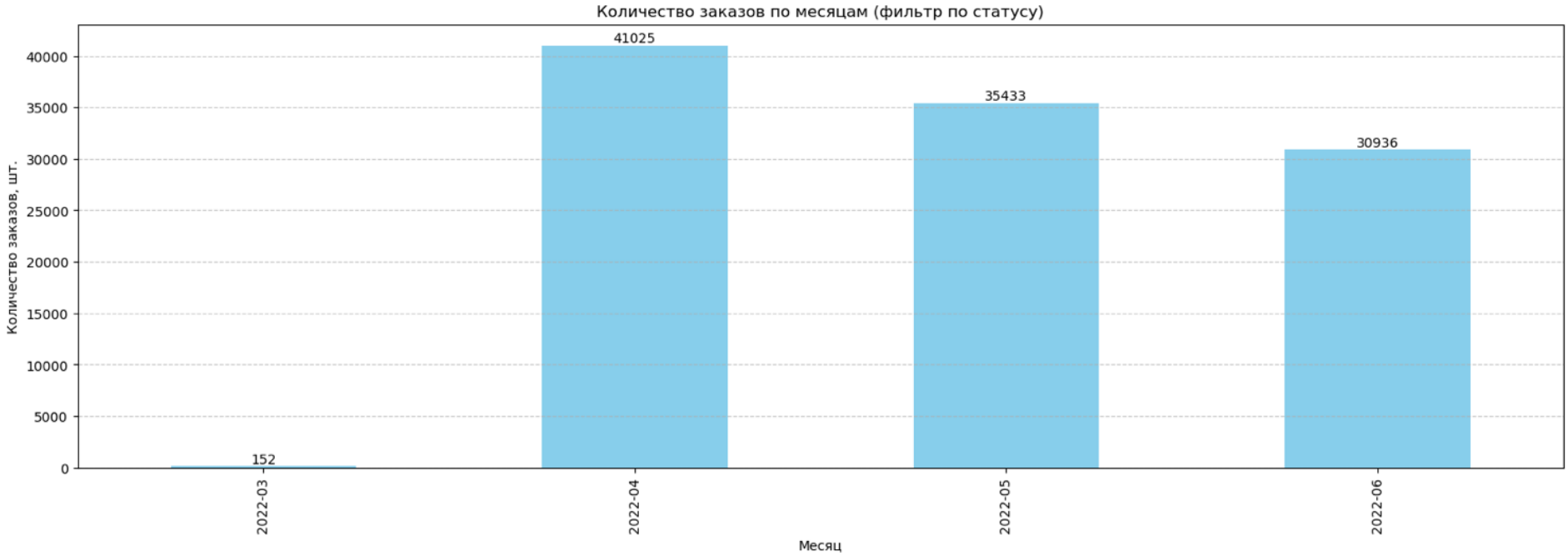
In [171…

```python
df = input_raw.copy(deep=True)

# Словарь соответствия английских и русских статусов
status_translation = {
    'Cancelled': 'Отменено',
    'Pending': 'В ожидании',
    'Pending - Waiting for Pick Up': 'Ожидание — ожидание получения',
    'Shipped': 'Отправленный',
    'Shipped - Damaged': 'Отправлено — повреждено',
    'Shipped - Delivered to Buyer': 'Отгружено — доставлено покупателю',
    'Shipped - Lost in Transit': 'Отправлено — потеряно в пути',
    'Shipped - Out for Delivery': 'Отправлено — отправлено на доставку',
    'Shipped - Picked Up': 'Отправлено — получено',
    'Shipped - Rejected by Buyer': 'Отправлено — отклонено покупателем',
    'Shipped - Returned to Seller': 'Отправлено — возвращено продавцу',
    'Shipped - Returning to Seller': 'Отправлено — возвращается продавцу',
    'Shipping': 'Перевозки'
}

# Преобразование статусов в русские
df['Status_Russian'] = df['Status'].map(status_translation)

# Группировка данных по статусам
status_counts = df['Status_Russian'].value_counts()

# Построение столбчатой диаграммы
plt.figure(figsize=(b, b * 0.3))
bars = status_counts.plot(kind='bar', color='skyblue')

# Добавление аннотаций над столбиками
for bar in bars.patches:
    plt.text(bar.get_x() + bar.get_width() / 2 , bar.get_height() + 0.1, str(bar.get_height()), ha='center', colo

plt.title('Распределение по статусам')
plt.xlabel('Статус')
plt.ylabel('Количество заказов, шт.')
plt.xticks(rotation=45, ha='right')  # Поворот подписей оси x для лучшей читаемости
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```


Распределение по статусам
Количество заказов, шт.
80000
70000
60000
50000
40000
30000
20000
10000
0
77804
28769
18332
1953
973
658
281
145
35
11
8
5
1
Отправленный
Отгружено – доставлено покупателю
Отменено
Отправлено – возвращено продавцу
Отправлено – получено
В ожидании
Ожидание – ожидание получения
Отправлено – возвращается продавцу
Отправлено – отправлено на доставку
Отправлено – отклонено покупателем
Перевозки
Отправлено – потеряно в пути
Отправлено – повреждено
Статус

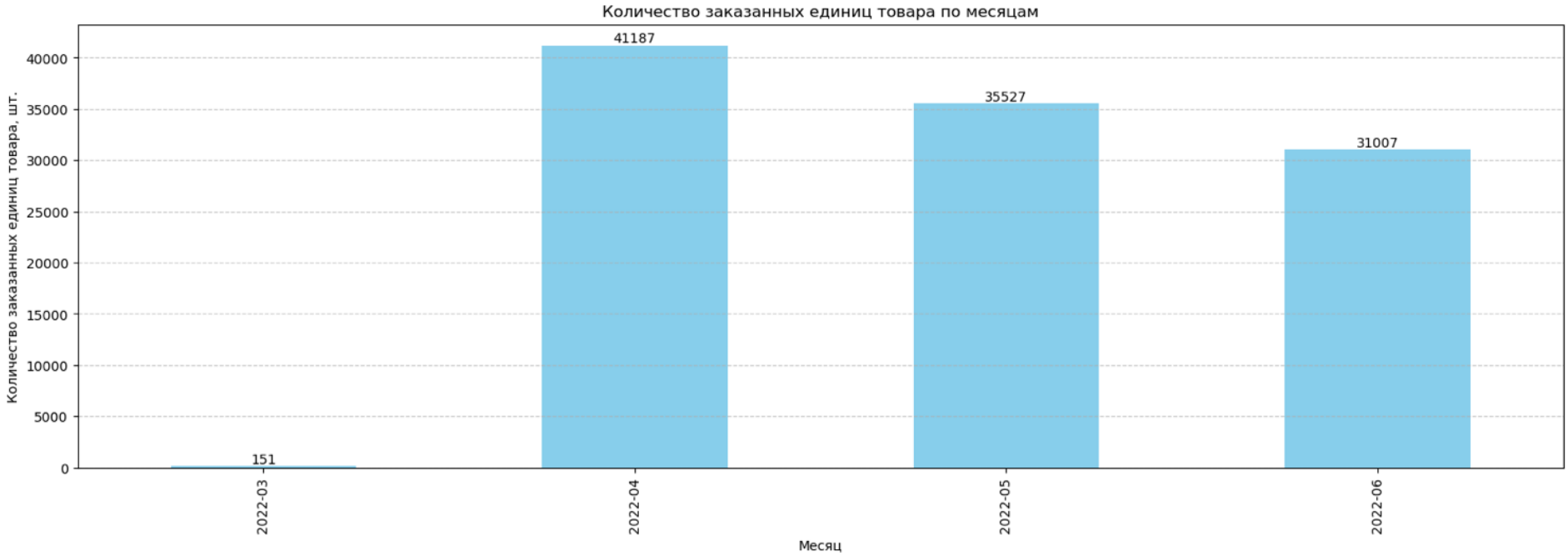
In [172…

```
DF = input_raw.copy(deep=True)

DF['Month'] = DF['Date'].dt.to_period('M')
monthly_orders = DF.groupby('Month').size()

# Фильтр для столбца 'Status'
allowed_statuses = ['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up']
filtered_DF = DF[DF['Status'].isin(allowed_statuses)]

# Постройте столбчатую диаграмму для отфильтрованных данных
plt.figure(figsize=(b, b * 0.3))
bar_plot = filtered_DF.groupby('Month').size().plot(kind='bar', color='skyblue')
plt.title('Количество заказов по месяцам (фильтр по статусу)')
plt.xlabel('Месяц')
plt.ylabel('Количество заказов, шт.')
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Добавьте аннотации (количество заказов) над каждым столбиком
for i, value in enumerate(filtered_DF.groupby('Month').size()):
    bar_plot.text(i, value + 0.1, str(value), ha='center', va='bottom')

plt.show()
```


Количество заказов по месяцам (фильтр по статусу)
Количество заказов, шт.
0
5000
10000
15000
20000
25000
30000
35000
40000
152
41025
35433
30936
2022-03
2022-04
2022-05
2022-06
Месяц

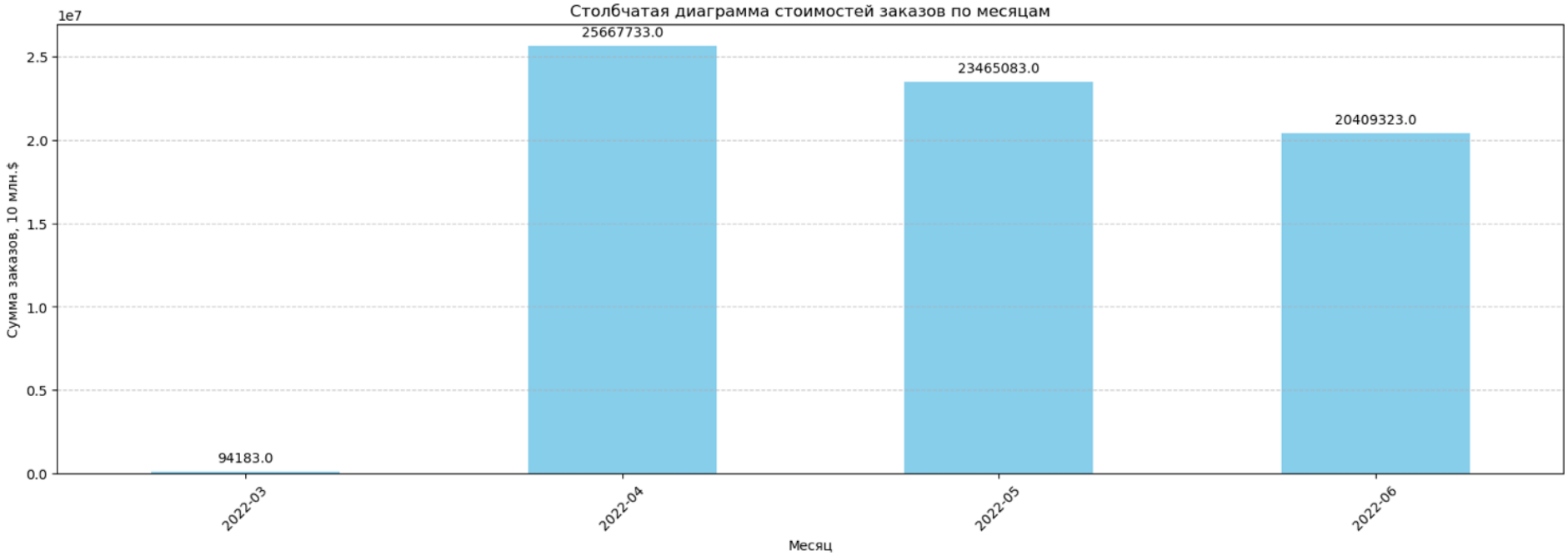
In [173…

```
DF = input_raw.copy(deep=True)

# Фильтрация данных по значениям в колонке 'Status'
DF = DF[DF['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Создание нового столбца 'Month' с месяцем
DF['Month'] = DF['Date'].dt.to_period('M')

# Группировка данных по месяцам и суммирование количества заказов
monthly_orders = DF.groupby('Month')['Qty'].sum()

# Построение столбчатой диаграммы с добавлением значений над столбиками
plt.figure(figsize=(b, b * 0.3))
ax = monthly_orders.plot(kind='bar', color='skyblue')

# Добавление значений над столбиками
for i, v in enumerate(monthly_orders):
    ax.text(i, v + 0.1, str(v), ha='center', va='bottom')

plt.title('Количество заказанных единиц товара по месяцам')
plt.xlabel('Месяц')
plt.ylabel('Количество заказанных единиц товара, шт.')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```


Количество заказанных единиц товара по месяцам
Количество заказанных единиц товара, шт.
0
5000
10000
15000
20000
25000
30000
35000
40000
151
41187
35527
31007
2022-03
2022-04
2022-05
2022-06
Месяц

In [174…

```
df = input_raw.copy(deep=True)

# Фильтрация данных по значениям в колонке 'Status'
filtered_df = df[df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])].copy()

# Добавление нового столбца 'Month' для хранения месяца из столбца 'Date'
filtered_df['Month'] = filtered_df['Date'].dt.to_period('M')

# Группировка данных по месяцам и суммирование стоимости заказов
monthly_sum = filtered_df.groupby('Month')['Amount'].sum()

# Построение столбчатой диаграммы
plt.figure(figsize=(b, b * 0.3))
ax = monthly_sum.plot(kind='bar', color='skyblue')

# Добавление значений суммы заказов над столбиками
for p in ax.patches:
    ax.annotate(str(p.get_height()), (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', xytext=(0, 10), textcoords='offset points')

plt.title('Столбчатая диаграмма стоимостей заказов по месяцам')
plt.xlabel('Месяц')
plt.ylabel('Сумма заказов, 10 млн.$')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```


Столбчатая диаграмма стоимостей заказов по месяцам
1e7
25667733.0
23465083.0
20409323.0
94183.0
2.5
2.0
1.5
1.0
0.5
0.0
Сумма заказов, 10 млн.$
2022-03
2022-04
2022-05
2022-06
Месяц

In [175…

```python
df = input_raw.copy(deep=True)

# Фильтрация строк по условию в колонке Status
filtered_df = df[~df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Группировка по категориям и суммирование по колонке Amount
grouped_df = filtered_df.groupby('Category')['Amount'].sum().reset_index()

# Сортировка по убыванию суммы
grouped_df = grouped_df.sort_values(by='Amount', ascending=False)

# Построение столбчатого графика
plt.figure(figsize=(b, b * 0.3))
plt.bar(grouped_df['Category'], grouped_df['Amount'])
plt.xlabel('Категория')
plt.ylabel('Сумма продаж, млн.')
plt.title('График суммы продаж по категориям')
plt.xticks(rotation=45, ha='right')  # Подписи оси x под углом 45 градусов
plt.grid(axis='y')  # Включаем горизонтальную сетку

# Добавляем значения суммы над столбиками
for i, value in enumerate(grouped_df['Amount']):
    plt.text(i, value + 5, str(value), ha='center', va='bottom')

plt.show()
```

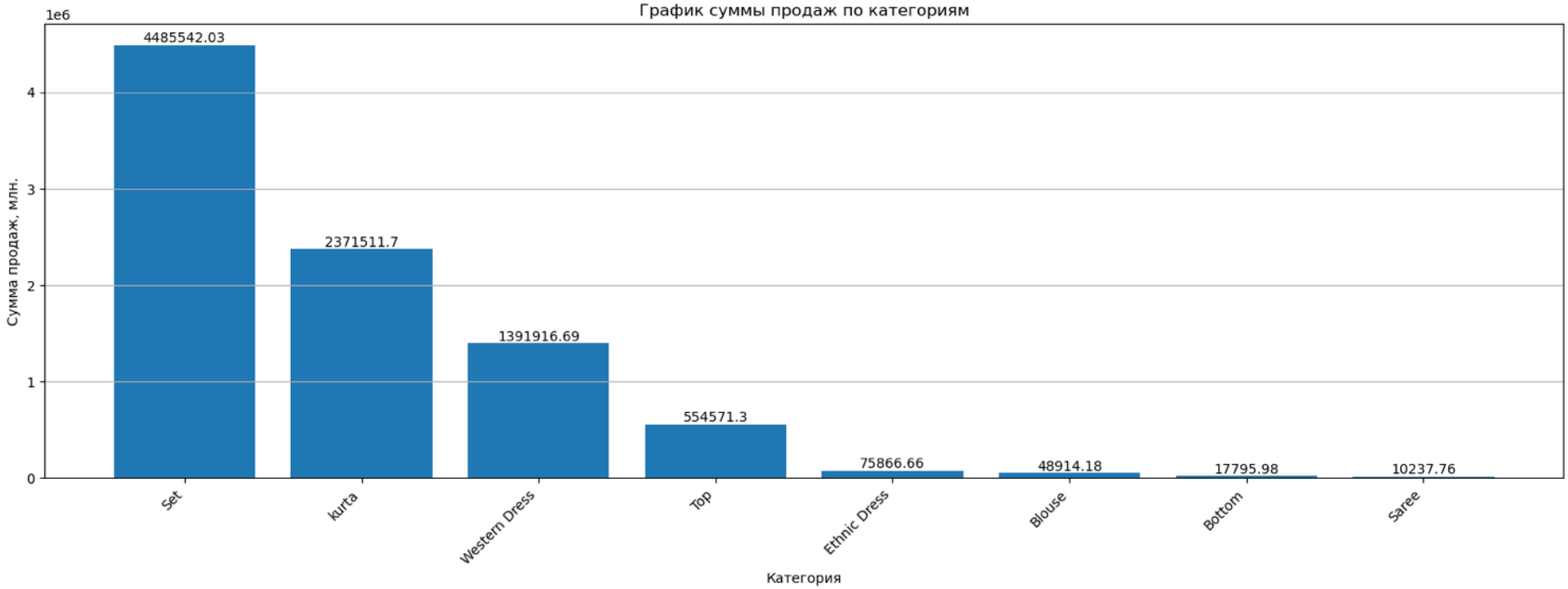


График суммы продаж по категориям
1e6
Сумма продаж, млн.
0
1
2
3
4
4485542.03
2371511.7
1391916.69
554571.3
75866.66
48914.18
17795.98
10237.76
Set
kurta
Western Dress
Top
Ethnic Dress
Blouse
Bottom
Saree
Категория

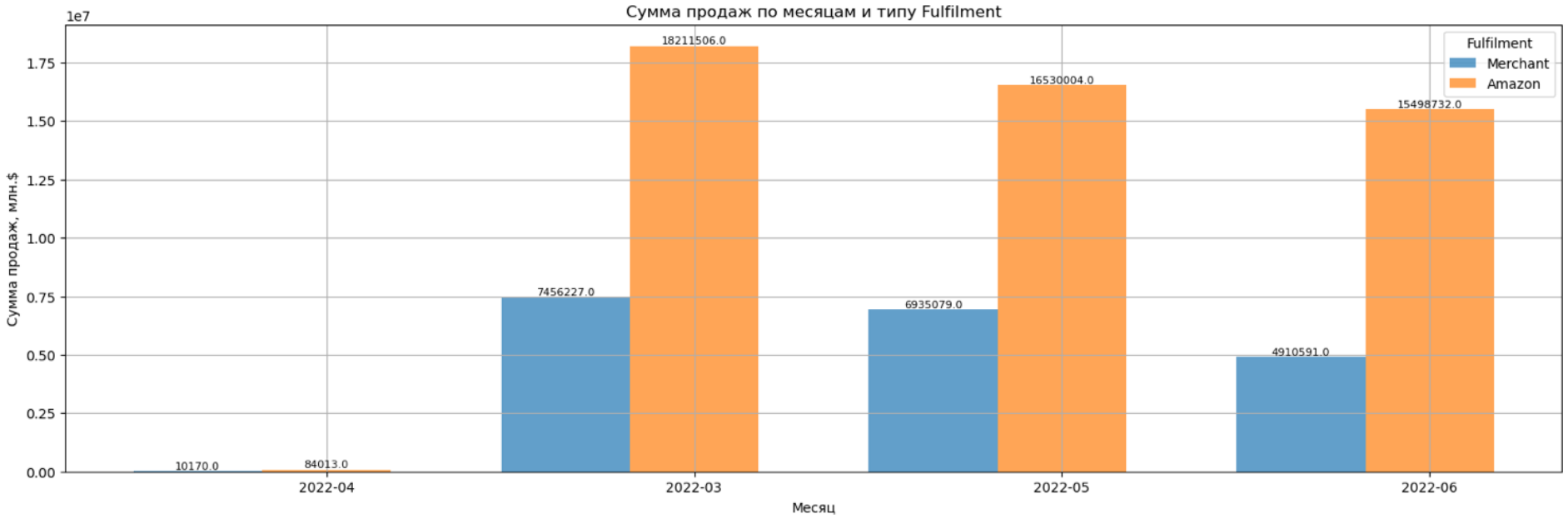
In [176…

```
df = input_raw.copy(deep=True)

# Фильтруем строки по условию в колонке Status
filtered_df = df[df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Создаем область для двух диаграмм
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(12, 6))

# Первая круговая диаграмма (по количеству)
fulfilment_count = filtered_df['Fulfilment'].value_counts()
fulfilment_count.plot.pie(autopct='%1.1f%%', startangle=90, ax=axes[0])
axes[0].set_title('Распределение по количеству заказов')

# Вторая круговая диаграмма (по сумме в колонке Amount)
fulfilment_sum = filtered_df.groupby('Fulfilment')['Amount'].sum()
fulfilment_sum.plot.pie(autopct='%1.1f%%', startangle=90, ax=axes[1])
axes[1].set_title('Распределение по сумме заказов')

# Отображение диаграмм
plt.show()
```

In [177…

```python
df = input_raw.copy(deep=True)

df['Date'] = pd.to_datetime(df['Date'])

# Фильтрация строк по условиям
filtered_df = df[df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Построение столбчатой диаграммы
fig, ax = plt.subplots(figsize=(b, b * 0.3))

# Создание уникального списка месяцев
months = filtered_df['Date'].dt.to_period("M").unique()

# Ширина столбиков
```

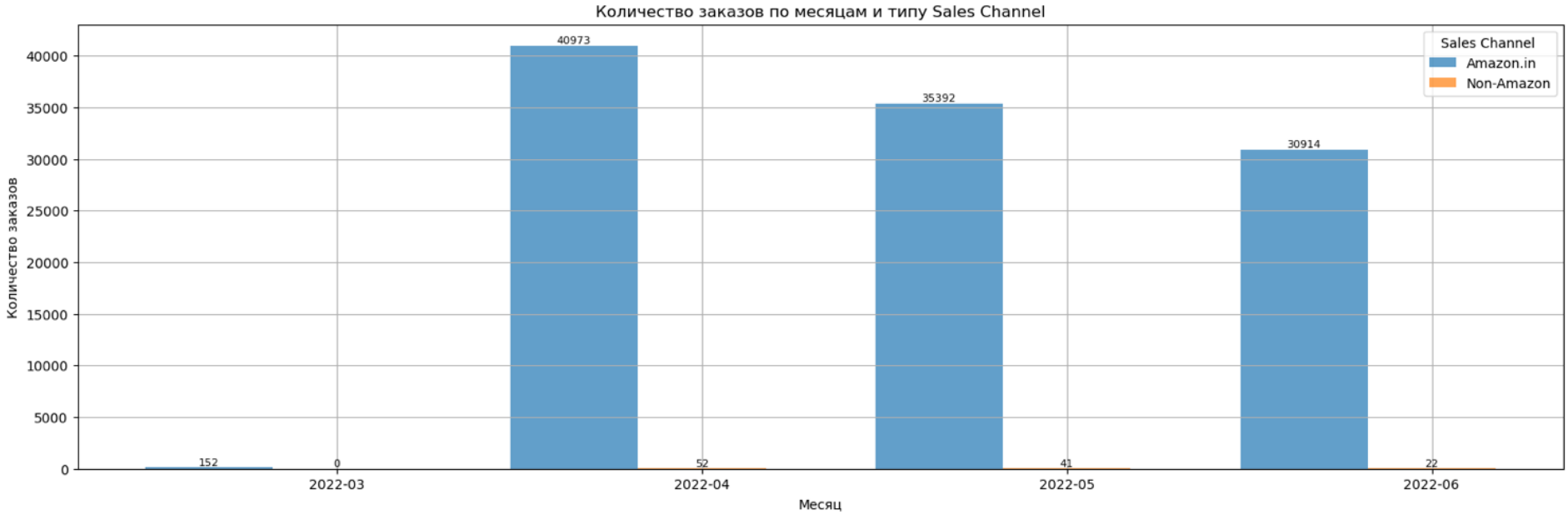
```
bar_width = 0.35

for i, fulfillment in enumerate(filtered_df['Fulfilment'].unique()):
    subset = filtered_df[filtered_df['Fulfilment'] == fulfillment]
    subset_grouped = subset.groupby(subset['Date'].dt.to_period("M"))['Amount'].sum()

    # Перемещение каждого столбика по x на ширину столбика
    x_pos = [pos + i * bar_width for pos in range(len(months))]

    bars = ax.bar(x_pos, subset_grouped, width=bar_width, label=fulfillment, alpha=0.7)

    # Добавление значений над столбиками
    for bar, value in zip(bars, subset_grouped):
        ax.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), round(value, 2),
                ha='center', va='bottom', fontsize=8, color='black')

# Настройка внешнего вида диаграммы
ax.set_title('Сумма продаж по месяцам и типу Fulfilment')
ax.set_xlabel('Месяц')
ax.set_ylabel('Сумма продаж, млн.$')
ax.grid(True)
ax.legend(title='Fulfilment')

# Форматирование дат на оси x
date_form = DateFormatter("%Y-%m")
ax.xaxis.set_major_formatter(date_form)

# Расстановка меток по x и их названия (месяцы)
ax.set_xticks([pos + bar_width for pos in range(len(months))])
ax.set_xticklabels(months)

# Отображение диаграммы
plt.show()
```



In [178…

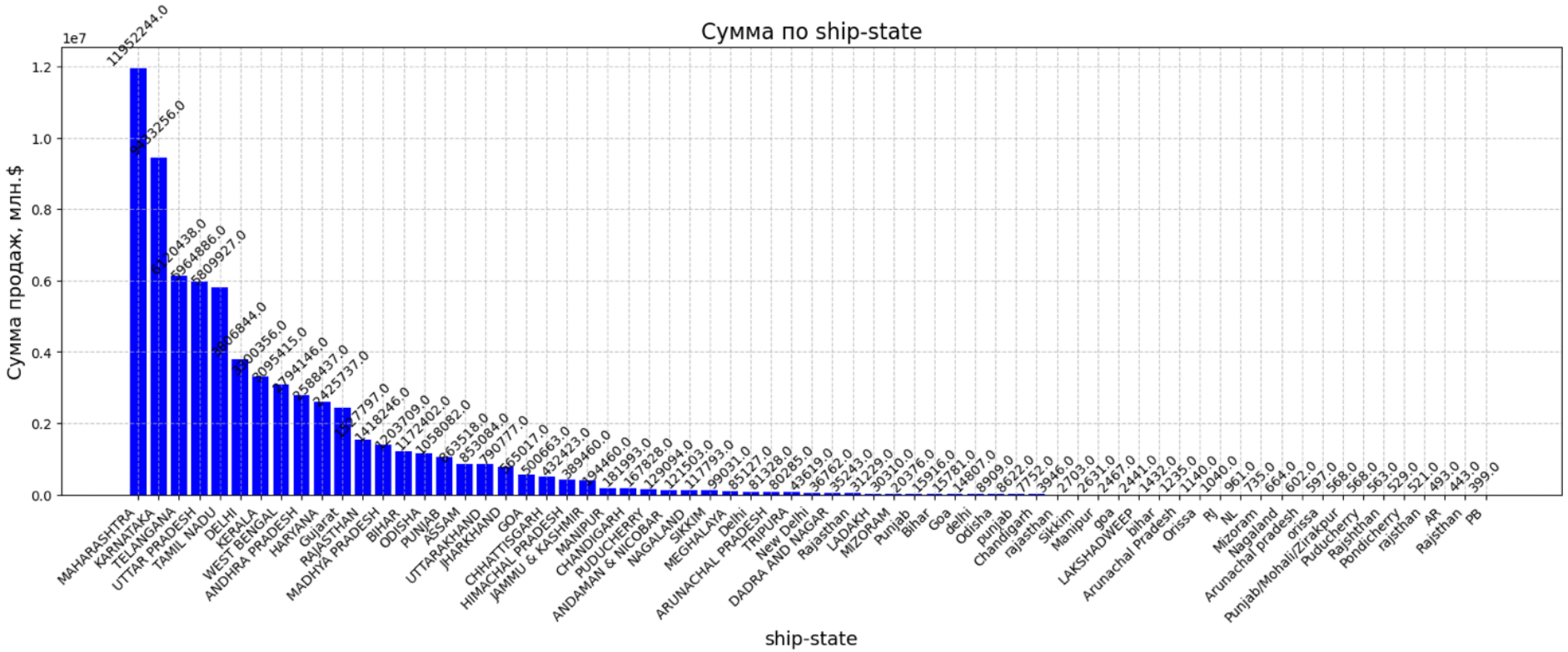
```
df = input_raw.copy(deep=True)

df['Date'] = pd.to_datetime(df['Date'])

# Фильтрация строк по условиям
filtered_df = df[df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Построение столбчатой диаграммы
fig, ax = plt.subplots(figsize=(b, b * 0.3))

# Создание уникального списка месяцев и сортировка их по возрастанию
months = sorted(filtered_df['Date'].dt.to_period("M").unique())

# Ширина столбиков
bar_width = 0.35

for i, channel in enumerate(filtered_df['Sales Channel '].unique()):
    subset = filtered_df[filtered_df['Sales Channel '] == channel]
    subset_grouped = subset.groupby(subset['Date'].dt.to_period("M"))['Qty'].size()

    # Перемещение каждого столбика по x на ширину столбика
    x_pos = [pos + i * bar_width for pos in range(len(months))]
```

```
    bars = ax.bar(x_pos, subset_grouped.reindex(months, fill_value=0), width=bar_width, label=channel, alpha=0.7)

    # Добавление значений над столбиками
    for bar, value in zip(bars, subset_grouped.reindex(months, fill_value=0)):
        ax.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), round(value, 2),
                ha='center', va='bottom', fontsize=8, color='black')

# Настройка внешнего вида диаграммы
ax.set_title('Количество заказов по месяцам и типу Sales Channel')
ax.set_xlabel('Месяц')
ax.set_ylabel('Количество заказов')
ax.grid(True)
ax.legend(title='Sales Channel ')

# Форматирование дат на оси x
date_form = DateFormatter("%Y-%m")
ax.xaxis.set_major_formatter(date_form)

# Расстановка меток по x и их названия (месяцы)
ax.set_xticks([pos + bar_width for pos in range(len(months))])
ax.set_xticklabels(months)

# Отображение диаграммы
plt.show()
```


Количество заказов по месяцам и типу Sales Channel
Sales Channel
Amazon.in
Non-Amazon
Количество заказов
Месяц
2022-03
2022-04
2022-05
2022-06
152
0
40973
52
35392
41
30914
22
0
5000
10000
15000
20000
25000
30000
35000
40000

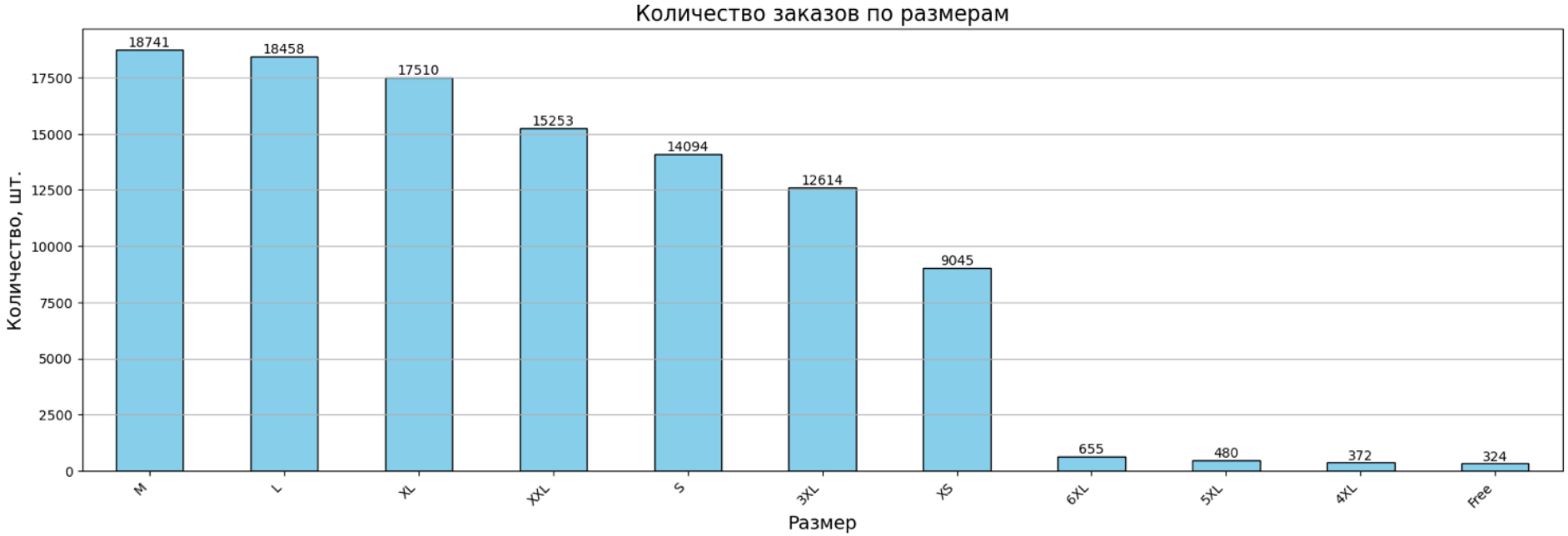
In [179…

```python
df = input_raw.copy(deep=True)

# Фильтрация строк по значению в колонке 'Status'
filtered_df = df[df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Группировка по 'ship-state' и суммирование 'Amount' для каждой группы
grouped_df = filtered_df.groupby('ship-state')['Amount'].sum().reset_index()

# Сортировка по уменьшению 'Amount'
grouped_df = grouped_df.sort_values(by='Amount', ascending=False)

# Построение столбчатой диаграммы
fig, ax = plt.subplots(figsize=(b, b * 0.3))
bars = ax.bar(grouped_df['ship-state'], grouped_df['Amount'], color='blue')

# Добавление названий и подписей на русском языке
plt.title('Сумма по ship-state', fontsize=16)
plt.xlabel('ship-state', fontsize=14)
plt.ylabel('Сумма продаж, млн.$', fontsize=14)

# Добавление сетки
plt.grid(True, linestyle='--', alpha=0.7)

# Добавление подписей к столбцам
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval, round(yval, 2), ha='center', va='bottom', rotation=45)

# Поворот подписей оси x на 45 градусов
plt.xticks(rotation=45, ha='right')

# Отображение графика
plt.show()
```


Сумма по ship-state
1e7
Сумма продаж, млн.$
ship-state
11952244.0
9453256.0
6120438.0
5964886.0
5809927.0
3906844.0
3300356.0
3095415.0
2794146.0
2588437.0
2425737.0
1577797.0
1418246.0
1203709.0
1172402.0
1058082.0
863518.0
853084.0
790777.0
565017.0
500663.0
432423.0
389460.0
194460.0
181993.0
167828.0
129094.0
121503.0
117793.0
99031.0
85127.0
81328.0
80285.0
43619.0
36762.0
35243.0
31229.0
30310.0
20376.0
15916.0
15781.0
14807.0
8909.0
8622.0
7752.0
3946.0
2703.0
2631.0
2467.0
2441.0
1432.0
1235.0
1140.0
1040.0
961.0
735.0
664.0
602.0
597.0
568.0
568.0
563.0
529.0
521.0
493.0
443.0
399.0
MAHARASHTRA
KARNATAKA
TELANGANA
UTTAR PRADESH
TAMIL NADU
DELHI
KERALA
WEST BENGAL
ANDHRA PRADESH
HARYANA
Gujarat
RAJASTHAN
MADHYA PRADESH
BIHAR
ODISHA
PUNJAB
ASSAM
UTTARAKHAND
JHARKHAND
GOA
CHHATTISGARH
HIMACHAL PRADESH
JAMMU & KASHMIR
MANIPUR
CHANDIGARH
PUDUCHERRY
ANDAMAN & NICOBAR
NAGALAND
SIKKIM
MEGHALAYA
Delhi
ARUNACHAL PRADESH
TRIPURA
New Delhi
DADRA AND NAGAR
Rajasthan
LADAKH
MIZORAM
Punjab
Bihar
Goa
delhi
Odisha
punjab
Chandigarh
rajasthan
Sikkim
Manipur
goa
LAKSHADWEEP
bihar
Arunachal Pradesh
Orissa
RJ
NL
Mizoram
Nagaland
Arunachal pradesh
orissa
Punjab/Mohali/Zirakpur
Puducherry
Rajshthan
Pondicherry
rajsthan
AR
Rajsthan
PB

In [180…

```
df = input_raw.copy(deep=True)

# Отфильтруем строки по заданным значениям в колонке Status
filtered_df = df[df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Построим столбчатую диаграмму для колонки Size
plt.figure(figsize=(b, b * 0.3))
size_counts = filtered_df['Size'].value_counts().sort_values(ascending=False)
size_counts.plot(kind='bar', color='skyblue', edgecolor='black')

# Настройка графика
plt.title('Количество заказов по размерам', fontsize=16)
plt.xlabel('Размер', fontsize=14)
plt.ylabel('Количество, шт.', fontsize=14)
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y')

# Добавление значений над столбиками
for i, v in enumerate(size_counts):
    plt.text(i, v + 0.2, str(v), ha='center', va='bottom', fontsize=10)

plt.show()
```



In [181…

```
df = input_raw.copy(deep=True)

df['Date'] = pd.to_datetime(df['Date'])

# Фильтруем строки
filtered_df = df[df['Status'].isin(['Shipped', 'Shipped - Delivered to Buyer', 'Shipped - Picked Up'])]

# Строим круговую диаграмму
b2b_sum = filtered_df.groupby('B2B')['Amount'].sum()
b2b_sum.plot(kind='pie', autopct='%1.1f%%', labels=['B2B', 'Non-B2B'], colors=['lightblue', 'lightcoral'])
plt.title('Продажи B2B по сумме')
plt.show()
```


Продажи B2B по сумме
Amount
B2B
99.2%
0.8%
Non-B2B