Анализ жд грузоперевозок.

План работ:

- Предобработка данных.
- Графическое отображение.
- Создание таблицы с ошибками.

Предобработка данных.

Откроем второй лист документа EXCEL:

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
#from pymystem3 import Mystem
```

In [2]:

```
data = pd.read_excel('datasets/Задача для аналитика.xlsx', sheet_name='Данные по перевозкам
```

Ознакомимся с данными:

In [3]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9680 entries, 0 to 9679
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   Название компаний   9680 non-null   object
 1   Год                 9680 non-null   object
 2   Месяц               9680 non-null   object
 3   Страна отправления  9680 non-null   object
 4   Тип вагона          9680 non-null   object
 5   Страна прибытия     9680 non-null   object
 6   Вагонов             9610 non-null   object
 7   Выручка             9665 non-null   object
dtypes: object(8)
memory usage: 605.1+ KB
```

Видны пропуски и некорректные типы данных.

In [4]:

```
display(data.head(5))
display(data.tail(5))
```

	Название компаний	Год	Месяц	Страна отправления	Тип вагона	Страна прибытия	Вагонов	Выручка
0	Компания 1	2019	1	АЗЕРБАЙДЖАН	Крытые	БЕЛАРУСЬ	8200	123000
1	Компания 1	2019	1	АЗЕРБАЙДЖАН	Крытые	РОССИЯ	200	3000
2	Компания 1	2019	1	АЗЕРБАЙДЖАН	Полувагоны	РОССИЯ	400	6000
3	Компания 1	2019	1	БЕЛАРУСЬ	Крытые	РОССИЯ	3800	57000
4	Компания 1	2019	1	БЕЛАРУСЬ	Полувагоны	РОССИЯ	300	4500

	Название компаний	Год	Месяц	Страна отправления	Тип вагона	Страна прибытия	Вагонов	Выручка
9675	Компания 16	2020	3	ТУРЦИЯ	Полувагоны	РОССИЯ	87	1305
9676	Компания 16	2020	3	УЗБЕКИСТАН	Полувагоны	КАЗАХСТАН	174	2610
9677	Компания 16	2020	3	УЗБЕКИСТАН	Полувагоны	РОССИЯ	783	11745
9678	Компания 16	2020	3	Украина	Полувагоны	КАЗАХСТАН	1479	22185
9679	Компания 16	2020	3	ШРИ-ЛАНКА	Полувагоны	КАЗАХСТАН	87	1305

Переименуем названия столбцов в более читабельные:

In [5]:

```
list_data_columns = data.columns.tolist()
for i in range(0, 8):
    list_data_columns[i] = list_data_columns[i].replace(' ', '_').lower()
list_data_columns[6] = 'количество_вагонов'
print(list_data_columns)
data.columns = list_data_columns
```

```
['название_компаний', 'год', 'месяц', 'страна_отправления', 'тип_вагона', 'с
трана_прибытия', 'количество_вагонов', 'выручка']
```

In [6]:

```
display(data.head(5))
display(data.tail(5))
```

	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия	ко
0	Компания 1	2019	1	АЗЕРБАЙДЖАН	Крытые	БЕЛАРУСЬ	
1	Компания 1	2019	1	АЗЕРБАЙДЖАН	Крытые	РОССИЯ	
2	Компания 1	2019	1	АЗЕРБАЙДЖАН	Полувагоны	РОССИЯ	
3	Компания 1	2019	1	БЕЛАРУСЬ	Крытые	РОССИЯ	
4	Компания 1	2019	1	БЕЛАРУСЬ	Полувагоны	РОССИЯ	

	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия
9675	Компания 16	2020	3	ТУРЦИЯ	Полувагоны	РОССИЯ
9676	Компания 16	2020	3	УЗБЕКИСТАН	Полувагоны	КАЗАХСТАН
9677	Компания 16	2020	3	УЗБЕКИСТАН	Полувагоны	РОССИЯ
9678	Компания 16	2020	3	Украина	Полувагоны	КАЗАХСТАН
9679	Компания 16	2020	3	ШРИ-ЛАНКА	Полувагоны	КАЗАХСТАН

Посмотрим на уникальные типы данных в таблице:

In [7]:

```
for i in data.columns:
    print(i)
    print(data[i].unique())
    print()
```

```
название_компаний
['Компания 1' 'Компания 2' 'Апельсин' 'Яблоки' 'Компания 3' 'Компания 4'
 'Компания 5' 'Компания 6' 'Компания 7' 2222 'Компания 8' 'Компания 9'
 'Компания 10' 'Компания 11' 'Компания 12' 'Компания 13' 'Компания 14'
 'Компания 15' 'Компания 16']

год
[2019 2020 '2O19' '2 019' '2O19']

месяц
[1 2 3 4 5 6 7 8 9 10 11 12 '07.01.1900' '07.01.1901' '07.01.1902'
 '07.01.1903' '07.01.1904' '07.01.1905' '07.01.1906' '07.01.1907'
 '07.01.1908' '07.01.1909' '07.01.1910' '07.01.1911' '07.01.1912'
 '07.01.1913' '07.01.1914' '07.01.1915' '07.01.1916' '07.01.1917'
 '07.01.1918' '07.01.1919' '07.01.1920' '07.01.1921' '07.01.1922'
 '07.01.1923' '07.01.1924' '07.01.1925' '07.01.1926' '07.01.1927'
 '07.01.1928' '07.01.1929' '07.01.1930' '07.01.1931' '07.01.1932'
 '07.01.1933' '07.01.1934' '07.01.1935' '07.01.1936' '07.01.1937'
 '07.01.1938' '07.01.1939' '07.01.1940' '07.01.1941' '07.01.1942'
 '07.01.1943' '07.01.1944' '07.01.1945' '07.01.1946' '07.01.1947'
 '07.01.1948' '07.01.1949' '07.01.1950' '07.01.1951' '07.01.1952'
 '07.01.1953' '07.01.1954' '07.01.1955' '07.01.1956' '07.01.1957'
 '07.01.1958' '07.01.1959' '07.01.1960']

страна_отправления
['АЗЕРБАЙДЖАН' 'БЕЛАРУСЬ' 'ГРУЗИЯ' 'ЕГИПЕТ' 'ИТАЛИЯ' 'КАЗАХСТАН'
 'КИРГИЗИЯ' 'КИТАЙ' 'КОРЕЯ, РЕСПУБЛИКА' 'ПОЛЬША' 'РОССИЯ'
 'СОЕДИНЕННЫЕ ШТАТЫ' 'ТАДЖИКИСТАН' 'Туркменистан' 'ТУРКМЕНИЯ' 'ТУРЦИЯ'
 'УЗБЕКИСТАН' 'Украина' 'ФИНЛЯНДИЯ' 'ГЕРМАНИЯ' 'ЛАТВИЯ' 'ЛИТВА' 'АЛЖИР'
 'БЕЛЬГИЯ' 'ИРАН' 'МЕКСИКА' 'МОЛДОВА, РЕСПУБЛИКА' 'МОНГОЛИЯ'
 'ЮЖНАЯ АФРИКА' 'НИДЕРЛАНДЫ' 'ОБЪЕДИНЕННЫЕ АРАБСКИЕ ЭМИРАТЫ' 'БОЛГАРИЯ'
 'ВЬЕТНАМ' 'ЛИВАН' 'ШРИ-ЛАНКА' 'ЭСТОНИЯ' 'ИНДИЯ' 'КАНАДА' 'АФГАНИСТАН'
 'ДАНИЯ' 'КАТАР' 'КУВЕЙТ' 'МАЛАЙЗИЯ' 'ПРОЧИЕ СТРАНЫ'
 'СОЕДИНЕННОЕ КОРОЛЕВСТВО' 'ШВЕЦИЯ' 'БАНГЛАДЕШ' 'ФИЛИППИНЫ' 'ВЕНГРИЯ'
 'КОРЕЯ,НАРОДНО-ДЕМОКРАТИЧЕСКАЯ' 'ЯПОНИЯ' 'ИСПАНИЯ' 'ФРАНЦИЯ' 'ШВЕЙЦАРИЯ'
 'ИЗРАИЛЬ' 'ПЕРУ' 'САУДОВСКАЯ АРАВИЯ' 7777777 'АВСТРАЛИЯ' 'КЕНИЯ' 'КИПР'
 'СИРИЙСКАЯ АРАБСКАЯ РЕСПУБЛИКА' 'ТУНИС' 'МАРОККО' 'ПОРТУГАЛИЯ' 'ТАИЛАНД'
 'ИНДОНЕЗИЯ' 'БРАЗИЛИЯ' 'ГОНКОНГ' 'ВИРГИНСКИЕ ОСТРОВА , БРИТАНСКИ'
 'МАЛЬТА' 'ГРЕЦИЯ' 'РУМЫНИЯ' 'СЕРБИЯ' 'РЕСПУБЛИКА МАКЕДОНИЯ' 'НОРВЕГИЯ'
 'Молдавия' 'СЛОВАКИЯ' 'ДОМИНИКАНСКАЯ РЕСПУБЛИКА'
 'ВИРГИНСКИЕ ОСТРОВА, США' 'ЭКВАДОР']

тип_вагона
['Крытые' 'Полувагоны' 'Цистерны' 'Цестерны' 'Полумагоны' 'Самолет'
 'Крысые' 'Полубагоны']

страна_прибытия
['БЕЛАРУСЬ' 'РОССИЯ' 'АФГАНИСТАН' 'КАЗАХСТАН' 'АЗЕРБАЙДЖАН' 'ГРУЗИЯ'
 'ДАНИЯ' 'СЛОВАКИЯ' 'ТАИЛАНД' 'Узбекистан' 'ИТАЛИЯ' 'КИРГИЗИЯ'
 'МОЛДОВА, РЕСПУБЛИКА' 'ПОЛЬША' 'ПРОЧИЕ СТРАНЫ' 'ТАДЖИКИСТАН' 'ТУНИС'
 'ТУРЦИЯ' 'ФРАНЦИЯ' 'ЛАТВИЯ' 'УКРАИНА' 'ЭСТОНИЯ' 'ГРЕЦИЯ' 'ИНДИЯ'
 'ИНДОНЕЗИЯ' 'МАЛАЙЗИЯ' 'МОНГОЛИЯ' 'ЯПОНИЯ' 'НИДЕРЛАНДЫ' 'ЛИТВА'
 'ГЕРМАНИЯ' 'БОЛГАРИЯ' 'ФИНЛЯНДИЯ' 'ВЕНГРИЯ' 'КИТАЙ' "КОТ-Д' ИВУАР"
 'ТУРКМЕНИЯ' 'ШРИ-ЛАНКА' 'ГАНА' 'ЕГИПЕТ' 'СОЕДИНЕННЫЕ ШТАТЫ'
```

```
 'ОБЪЕДИНЕННЫЕ АРАБСКИЕ ЭМИРАТЫ' 'БРАЗИЛИЯ' 'ХОРВАТИЯ' 'ШВЕЦИЯ' 'АВСТРИЯ'
 'ПОРТУГАЛИЯ' 'ЮЖНАЯ АФРИКА' 7777777 'КОРЕЯ,НАРОДНО-ДЕМОКРАТИЧЕСКАЯ'
 'ИСПАНИЯ' 'БЕЛЬГИЯ' 'ИРАН' 'СОЕДИНЕННОЕ КОРОЛЕВСТВО' 'АЛЖИР' 'МЕКСИКА'
 'СЕРБИЯ' 'НОРВЕГИЯ' 'ВЬЕТНАМ' 'КИПР' 'КАНАДА' 'ИЗРАИЛЬ'
 'САУДОВСКАЯ АРАВИЯ' 'МАЛЬТА' 'НИГЕРИЯ' 'ЭКВАДОР' 'КОРЕЯ, РЕСПУБЛИКА'
 'АВСТРАЛИЯ' 'КУВЕЙТ' 'НОВАЯ ЗЕЛАНДИЯ' 'АРМЕНИЯ' 'ШВЕЙЦАРИЯ'
 'ВИРГИНСКИЕ ОСТРОВА, США' 'Туркменистан' 'РФ']

количество_вагонов
[8200 200 400 ... 640668 105966 78561]

выручка
[123000 3000 6000 ... 9610020 1589490 1178415]
```

Начнём с названия компаний:

Подозрительное название 2222. Посмотрим, сколько таких значений в наших данных:

In [8]:

```
data.query('название_компаний == 2222')['название_компаний'].count()
```

Out[8]:

```
25
```

Всего 25. Посмотьрим на общую картину данных с этой ошибкой:

In [9]:

```
data.query('название_компаний == 2222')
```

Out[9]:

	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия	количество_в
4450	2222	2020	3	УЗБЕКИСТАН	Полувагоны	КАЗАХСТАН	
4451	2222	2020	3	УЗБЕКИСТАН	Полувагоны	РОССИЯ	
4452	2222	2020	3	Украина	Крытые	РОССИЯ	
4453	2222	2020	3	Украина	Полувагоны	КАЗАХСТАН	
4454	2222	2020	3	ФИНЛЯНДИЯ	Полувагоны	РОССИЯ	
4455	2222	2020	5	АЗЕРБАЙДЖАН	Крытые	КАЗАХСТАН	
4456	2222	2020	5	АЗЕРБАЙДЖАН	Крытые	РОССИЯ	
4457	2222	2020	5	АЗЕРБАЙДЖАН	Полувагоны	РОССИЯ	
4458	2222	2020	5	БЕЛАРУСЬ	Крытые	РОССИЯ	
4459	2222	2020	5	БЕЛАРУСЬ	Полувагоны	РОССИЯ	

Т.к. Компанию узнать не представляется возможным, заменим цифры 2222, выделяющейся строкой 'unknown'

In [10]:

```
data['название_компаний'].replace(2222, 'unknown', inplace=True)
```

In [11]:

```
data['название_компаний'].unique()
```

Out[11]:

```
array(['Компания 1', 'Компания 2', 'Апельсин', 'Яблоки', 'Компания 3',
       'Компания 4', 'Компания 5', 'Компания 6', 'Компания 7', 'unknown',
       'Компания 8', 'Компания 9', 'Компания 10', 'Компания 11',
       'Компания 12', 'Компания 13', 'Компания 14', 'Компания 15',
       'Компания 16'], dtype=object)
```

Рассмотрим столбец год. Здесь имеются строчные значения вместо int, а также значение года с пропуском (2 019).

In [12]:

```
data[data['год'] == '2 019']['год'].count()
```

Out[12]:

```
20
```

In [13]:

```
display(data[data['год'] == '2 019'])
```

	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия	количество_ва
8410	Компания 14	2 019	6	УЗБЕКИСТАН	Крытые	КИРГИЗИЯ	
8982	Компания 15	2 019	6	МАРОККО	Полувагоны	РОССИЯ	
8983	Компания 15	2 019	6	Молдавия	Полувагоны	Узбекистан	
8984	Компания 15	2 019	6	МОНГОЛИЯ	Крытые	РОССИЯ	
8985	Компания 15	2 019	6	МОНГОЛИЯ	Крытые	УКРАИНА	
8986	Компания 15	2 019	6	МОНГОЛИЯ	Полувагоны	РОССИЯ	
8987	Компания 15	2 019	6	ОБЪЕДИНЕННЫЕ АРАБСКИЕ ЭМИРАТЫ	Крытые	РОССИЯ	

Заменим на '2 019', 2O19 и 2O19 на 2019:

In [14]:

```
data['год'].replace('2 019', 2019, inplace=True)
```

In [15]:

```
data['год'].unique()
```

Out[15]:

```
array([2019, 2020, '2019', '2019'], dtype=object)
```

In [16]:

```
data['год'].replace('2019', 2019, inplace=True)
```

In [17]:

```
data['год'].unique()
```

Out[17]:

```
array([2019, 2020, '2019'], dtype=object)
```

In [18]:

```
data['год'].replace('2019', 2019, inplace=True)
```

In [19]:

```
data['год'].unique()
```

Out[19]:

```
array([2019, 2020])
```

Следующий - месяц.

Выбиваются строковые значения с полной датой типа '07.01.19**'. Преобразуем столбец в строку, заменим эти даты на -1, т.к. во все эти данные скорее всего сбой при записи базы либо при выгрузке давнных, затем преобразуем в int:

In [20]:

```
data['месяц'].unique()
```

Out[20]:

```
array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, '07.01.1900', '07.01.1901',
       '07.01.1902', '07.01.1903', '07.01.1904', '07.01.1905',
       '07.01.1906', '07.01.1907', '07.01.1908', '07.01.1909',
       '07.01.1910', '07.01.1911', '07.01.1912', '07.01.1913',
       '07.01.1914', '07.01.1915', '07.01.1916', '07.01.1917',
       '07.01.1918', '07.01.1919', '07.01.1920', '07.01.1921',
       '07.01.1922', '07.01.1923', '07.01.1924', '07.01.1925',
       '07.01.1926', '07.01.1927', '07.01.1928', '07.01.1929',
       '07.01.1930', '07.01.1931', '07.01.1932', '07.01.1933',
       '07.01.1934', '07.01.1935', '07.01.1936', '07.01.1937',
       '07.01.1938', '07.01.1939', '07.01.1940', '07.01.1941',
       '07.01.1942', '07.01.1943', '07.01.1944', '07.01.1945',
       '07.01.1946', '07.01.1947', '07.01.1948', '07.01.1949',
       '07.01.1950', '07.01.1951', '07.01.1952', '07.01.1953',
       '07.01.1954', '07.01.1955', '07.01.1956', '07.01.1957',
       '07.01.1958', '07.01.1959', '07.01.1960'], dtype=object)
```

In [21]:

```
list_12 = []
for i in range(1, 13):
    list_12.append(i)
```

In [22]:

```
data.query('месяц not in @list_12')['месяц'].count()
```

Out[22]:

```
72
```

In [23]:

```
data.query('месяц not in @list_12').head(15)
```

Out[23]:

	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия	количест
2692	Компания 5	2019	07.01.1900	РОССИЯ	Крытые	ПРОЧИЕ СТРАНЫ	
2758	Компания 5	2019	07.01.1900	УЗБЕКИСТАН	Крытые	КАЗАХСТАН	
2808	Компания 5	2019	07.01.1900	КАЗАХСТАН	Крытые	РОССИЯ	
2870	Компания 5	2019	07.01.1900	РОССИЯ	Крытые	РОССИЯ	
2895	Компания 5	2019	07.01.1900	ТУРКМЕНИЯ	Полувагоны	РОССИЯ	
4880	Компания 9	2019	07.01.1900	СОЕДИНЕННЫЕ ШТАТЫ	Полувагоны	РФ	
4881	Компания 9	2019	07.01.1901	ТАДЖИКИСТАН	Крытые	БЕЛАРУСЬ	
4882	Компания 9	2019	07.01.1902	ТАДЖИКИСТАН	Крытые	КАЗАХСТАН	
4883	Компания 9	2019	07.01.1903	ТАДЖИКИСТАН	Крытые	РФ	

In [24]:

```
data.query('месяц == "-1"')['месяц'].count()
```

Out[24]:

```
0
```

In [25]:

```
data['месяц'] = data['месяц'].astype(str).str.replace(r'\d\d.\d\d.\d{4}', '-1')
```

In [26]:

```
data['месяц'] = data['месяц'].astype('int64')
```

In [27]:

```
data['месяц'].replace(-1, data['месяц'].median(), inplace=True)
```

In [28]:

```
data['месяц'].unique()
```

Out[28]:

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

Рассмотрим столбец страна_отправления:

In [29]:

```
count_errors = 0
for i in data['страна_отправления']:
    try:
        int(i)
        count_errors += 1
    except:
        pass
print(f'Колличество ошибок: {count_errors}')
```

```
Колличество ошибок: 52
```

Заменим числовой тип данных на unknown, т.к. название страны не известно.

In [30]:

```
data['страна_отправления'].replace(7777777, 'unknown', inplace=True)
data['страна_отправления'] = data['страна_отправления'].str.lower()
```

In [31]:

```
#m = Mystem()
#for i in sorted_list:
#    lemmas = m.lemmatize(i)
#    lemmas = (''').join(lemmas[0:-1])
#    data['страна_отправления'].replace(i, lemmas, inplace=True)
#data['страна_отправления'].unique()
```

In [32]:

```
sorted_list = sorted(data['страна_отправления'].unique())
sorted_list
```

Out[32]:

```
['unknown',
 'австралия',
 'азербайджан',
 'алжир',
 'афганистан',
 'бангладеш',
 'беларусь',
 'бельгия',
 'болгария',
 'бразилия',
 'венгрия',
 'виргинские острова , британски',
 'виргинские острова, сша',
 'вьетнам',
 'германия',
 'гонконг',
 'греция',
 'грузия',
```

In [33]:

```
len(sorted_list)
```

Out[33]:

```
81
```

In [34]:

```
count_errors = data.query('страна_отправления == "молдавия"')['страна_отправления'].count()
count_errors
```

Out[34]:

```
12
```

In [35]:

```
data['страна_отправления'].replace('молдавия', 'молдова, республика', inplace=True)
```

In [36]:

```
count_errors = count_errors + data.query('страна_отправления == "корея,народно-демократичес
count_errors
```

Out[36]:

```
28
```

In [37]:

```
data['страна_отправления'].replace('корея,народно-демократическая', 'корея, республика', in
```

In [38]:

```
count_errors = count_errors + data.query('страна_отправления == "прочие страны"')['страна_о
count_errors
```

Out[38]:

```
80
```

In [39]:

```
data['страна_отправления'].replace('прочие страны', 'unknown', inplace=True)
#data['страна_отправления'].replace('перо', 'перу', inplace=True)
```

In [40]:

```
sorted_list = sorted(data['страна_отправления'].unique())
sorted_list
```

Out[40]:

```
['unknown',
 'австралия',
 'азербайджан',
 'алжир',
 'афганистан',
 'бангладеш',
 'беларусь',
 'бельгия',
 'болгария',
 'бразилия',
 'венгрия',
 'виргинские острова , британски',
 'виргинские острова, сша',
 'вьетнам',
 'германия',
 'гонконг',
 'греция',
 'грузия',
```

In [41]:

```
len(sorted_list)
```

Out[41]:

```
78
```

Посмотрим на столбцец тип_вагона:

In [42]:

```
data['тип_вагона'].unique()
```

Out[42]:

```
array(['Крытые', 'Полувагоны', 'Цистерны', 'Цестерны', 'Полумагоны',
       'Самолет', 'Крысые', 'Полубагоны'], dtype=object)
```

Посчитаем опечатки:

In [43]:

```
error_list = ['Цестерны', 'Полумагоны', 'Крысые', 'Полубагоны']
```

In [44]:

```
count_errors = 0
for i in error_list:
    tmp = i
    count = data.query('тип_вагона == @tmp')['тип_вагона'].count()
    count_errors += count
print(f'Колличество ошибок: {count_errors}')
```

```
Колличество ошибок: 72
```

Заменим опечатки на верные данные:

In [45]:

```
data['тип_вагона'].replace('Крысые', 'Крытые', inplace=True)
```

In [46]:

```
data['тип_вагона'].replace('Полубагоны', 'Полувагоны', inplace=True)
```

In [47]:

```
data['тип_вагона'].replace('Цестерны', 'Цистерны', inplace=True)
```

In [48]:

```
data['тип_вагона'].replace('Полумагоны', 'Полувагоны', inplace=True)
```

In [49]:

```
data['тип_вагона'] = data.тип_вагона.str.lower()
```

In [50]:

```
data['тип_вагона'].unique()
```

Out[50]:

```
array(['крытые', 'полувагоны', 'цистерны', 'самолет'], dtype=object)
```

Рассмотрим столбец страна_прибытия аналогично столбцу страна_отправления:

In [51]:

```
count_errors = 0
for i in data['страна_прибытия']:
    try:
        int(i)
        count_errors += 1
    except:
        pass
print(f'Колличество ошибок: {count_errors}')
```

```
Колличество ошибок: 16
```

In [52]:

```
data['страна_прибытия'].replace(7777777, 'unknown', inplace=True)
data['страна_прибытия'] = data['страна_прибытия'].str.lower()
```

In [53]:

```
sorted_list = sorted(data['страна_прибытия'].unique())
sorted_list
```

Out[53]:

```
['unknown',
 'австралия',
 'австрия',
 'азербайджан',
 'алжир',
 'армения',
 'афганистан',
 'беларусь',
 'бельгия',
 'болгария',
 'бразилия',
 'венгрия',
 'виргинские острова, сша',
 'вьетнам',
 'гана',
 'германия',
 'греция',
 'грузия',
```

In [54]:

```
len(sorted_list)
```

Out[54]:

```
75
```

In [55]:

```
count_errors = data.query('страна_прибытия == "корея,народно-демократическая"')['страна_при
count_errors
```

Out[55]:

28

In [56]:

```
data['страна_прибытия'].replace('корея,народно-демократическая', 'корея, республика', inpla
```

In [57]:

```
count_errors = count_errors + data.query('страна_прибытия == "рф"')['страна_прибытия'].coun
count_errors
```

Out[57]:

117

In [58]:

```
data['страна_прибытия'].replace('рф', 'россия', inplace=True)
```

In [59]:

```
count_errors = count_errors + data.query('страна_прибытия == "прочие страны"')['страна_приб
count_errors
```

Out[59]:

313

In [60]:

```
data['страна_прибытия'].replace('прочие страны', 'unknown', inplace=True)
```

In [61]:

```
sorted_list = sorted(data['страна_прибытия'].unique())
sorted_list
```

Out[61]:

```
['unknown',
 'австралия',
 'австрия',
 'азербайджан',
 'алжир',
 'армения',
 'афганистан',
 'беларусь',
 'бельгия',
 'болгария',
 'бразилия',
 'венгрия',
 'виргинские острова, сша',
 'вьетнам',
 'гана',
 'германия',
 'греция',
 'грузия',
```

In [62]:

```
len(sorted_list)
```

Out[62]:

```
72
```

Посмотрим на столбец количество вагонов:

In [63]:

```
data['количество_вагонов'].count()
```

Out[63]:

```
9610
```

Заполним пропуски звыбивающимся -1.

In [64]:

```
data['количество_вагонов'].fillna(-1, inplace=True)
```

Посчитаем строковые значения(ошибки) в столбце:

In [65]:

```
erorr_str = []
for i in data['количество_вагонов']:
    try:
        int(i)
    except:
        erorr_str.append(i)
print(erorr_str)
print(f'Колличество ошибок: {len(erorr_str)}')
```

```
['в', 'й', 'у', 'цй', 'к', 'е', 'й', 'цу', 'кцук', 'цу', 'пе', 'кке', 'йц',
'йцу', 'йуйцу']
Колличество ошибок: 15
```

In [66]:

```
erorr_str = set(erorr_str)
```

Заменим данные значения на выбивающееся -1, т.к. восстановить эти данные не представляется возможным:

In [67]:

```
for i in erorr_str:
    data['количество_вагонов'].replace(i, -1, inplace=True)
```

Заменим ошибки(-1) на среднее, т.к. их не много, влияния оказать в любом случае это не должно.

In [68]:

```
data['количество_вагонов'].replace(-1, data['количество_вагонов'].mean(), inplace=True)
```

Далее рассмотрим столобец выручка:

In [69]:

```
data['выручка'].count()
```

Out[69]:

```
9665
```

Заполним пропуски выбивающимся -1:

In [70]:

```
data['количество_вагонов'].describe()
```

Out[70]:

```
count    9.680000e+03
mean     1.264376e+04
std      8.125824e+04
min      8.700000e+01
25%      2.000000e+02
50%      7.200000e+02
75%      3.420000e+03
max      1.882100e+06
Name: количество_вагонов, dtype: float64
```

In [71]:

```
data['выручка'].fillna(-1, inplace=True)
```

In [72]:

```
count_errors = []
for i in data['выручка']:
    try:
        int(i)
    except:
        count_errors.append(i)
print(erorr_str)
print(f'Колличество ошибок: {len(count_errors)}')
```

```
{'цу', 'йц', 'пе', 'к', 'йуйцу', 'е', 'цй', 'у', 'кке', 'йцу', 'в', 'й', 'кц
ук'}
Колличество ошибок: 1
```

Заменим данное значение на выбивающееся -1, т.к. восстановить эту строку не представляется возможным:

In [73]:

```
data['выручка'].replace('цйцу', -1, inplace=True)
```

In [74]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9680 entries, 0 to 9679
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   название_компаний   9680 non-null   object 
 1   год                 9680 non-null   int64  
 2   месяц               9680 non-null   int64  
 3   страна_отправления  9680 non-null   object 
 4   тип_вагона          9680 non-null   object 
 5   страна_прибытия     9680 non-null   object 
 6   количество_вагонов  9680 non-null   float64
 7   выручка             9680 non-null   int64  
dtypes: float64(1), int64(3), object(4)
memory usage: 605.1+ KB
```

In [75]:

```
data['выручка'].replace(-1, data['выручка'].mean(), inplace=True)
```

In [76]:

```
display(data.head(5))
display(data.tail(5))
```

	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия	ко
0	Компания 1	2019	1	азербайджан	крытые	беларусь	
1	Компания 1	2019	1	азербайджан	крытые	россия	
2	Компания 1	2019	1	азербайджан	полувагоны	россия	
3	Компания 1	2019	1	беларусь	крытые	россия	
4	Компания 1	2019	1	беларусь	полувагоны	россия	

	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия
9675	Компания 16	2020	3	турция	полувагоны	россия
9676	Компания 16	2020	3	узбекистан	полувагоны	казахстан
9677	Компания 16	2020	3	узбекистан	полувагоны	россия
9678	Компания 16	2020	3	украина	полувагоны	казахстан
9679	Компания 16	2020	3	шри-ланка	полувагоны	казахстан

Проверм данные на дубликаты:

In [77]:

```
data.duplicated().sum()
```

Out[77]:

2

Дубликатов очень мало, но есть. избавимся от него:

In [78]:

```
data = data.drop_duplicates().reset_index(drop=True)
```

In [79]:

```
data.duplicated().sum()
```

Out[79]:

```
0
```

Предобработка данных завершена. Можно переходить к заданиям.

Графическое отображение.

Отобразим таблицу в разрезе месяц/год/компания/количество вагонов/ выручка:

In [80]:

```
display(data.loc[:, ['месяц', 'год', 'название_компаний', 'количество_вагонов', 'выручка']])
```

	месяц	год	название_компаний	количество_вагонов	выручка
5693	6	2020	Компания 9	90.0	1350.0
4111	4	2019	Компания 7	2156.0	32340.0
2379	2	2020	Компания 4	400.0	6000.0
9225	2	2020	Компания 15	261.0	3915.0
254	7	2019	Компания 1	2100.0	31500.0
8967	6	2019	Компания 15	957.0	14355.0
3197	5	2020	Компания 5	98.0	1470.0
5701	6	2020	Компания 9	180.0	2700.0
4095	4	2019	Компания 7	294.0	4410.0
6457	2	2019	Компания 11	2070.0	31050.0
3060	3	2020	Компания 5	60662.0	909930.0
7718	12	2019	Компания 13	203493.0	3052395.0

Построим линейный график помесячной динамики количества вагонов в разрезе направления (страна отправления-страна прибытия):

Добавим столбцец направление для выполнения задания:

In [81]:

```
data['направление'] = data['страна_отправления'] + ' - ' + data['страна_прибытия']
```

In [82]:

```
display(data.head(5))
display(data.tail(5))
```

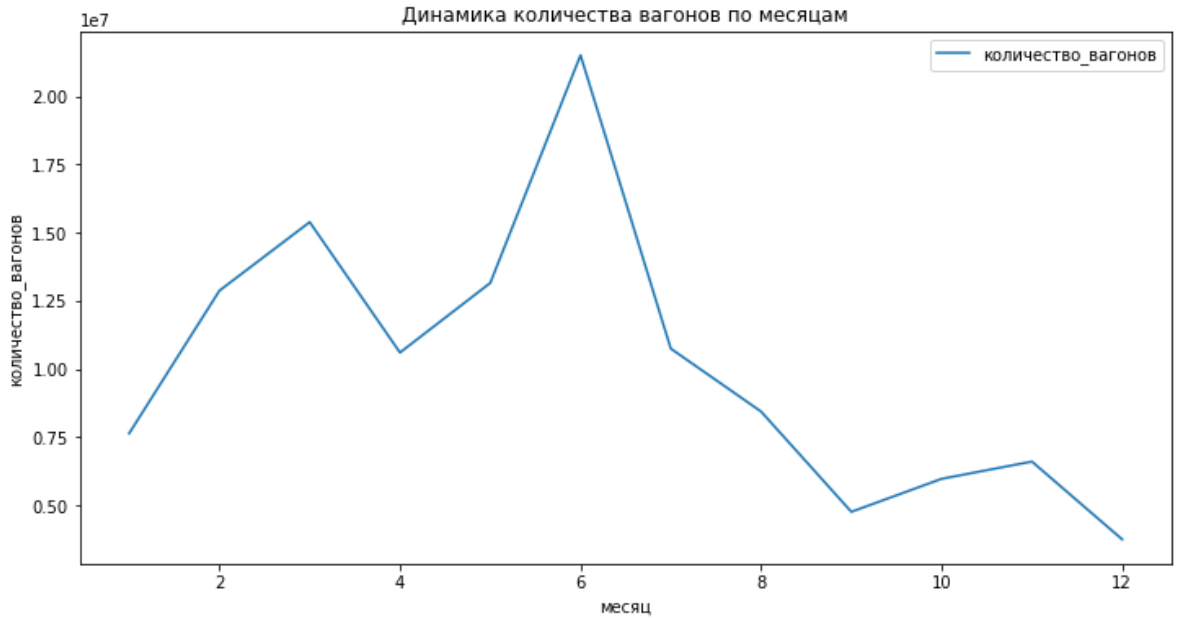
	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия	ко
0	Компания 1	2019	1	азербайджан	крытые	беларусь	
1	Компания 1	2019	1	азербайджан	крытые	россия	
2	Компания 1	2019	1	азербайджан	полувагоны	россия	
3	Компания 1	2019	1	беларусь	крытые	россия	
4	Компания 1	2019	1	беларусь	полувагоны	россия	

	название_компаний	год	месяц	страна_отправления	тип_вагона	страна_прибытия
9673	Компания 16	2020	3	турция	полувагоны	россия
9674	Компания 16	2020	3	узбекистан	полувагоны	казахстан
9675	Компания 16	2020	3	узбекистан	полувагоны	россия
9676	Компания 16	2020	3	украина	полувагоны	казахстан
9677	Компания 16	2020	3	шри-ланка	полувагоны	казахстан

Общий график динамики количества вогонов по месяцам:

In [83]:

```
(data.query('страна_отправления != "unknown" and страна_прибытия != "unknown"')
 .groupby('месяц')
 .agg({'количество_вагонов': 'sum'})
 .reset_index()
 .plot(x='месяц', y='количество_вагонов', figsize=(12, 6))
)
plt.xlabel('месяц')
plt.ylabel('количество_вагонов')
plt.title('Динамика количества вагонов по месяцам')
plt.show()
```



Отдельные графики каждого месяца с динамикой вагонов по направлениям:

In [84]:

```
for month in range(1, 13):
    (data.query('страна_отправления != "unknown" and страна_прибытия != "unknown" and месяц
     .groupby('направление')
     .agg({'количество_вагонов': 'sum'})
     .reset_index()
     .plot(x='направление', y='количество_вагонов', figsize=(12, 6))
    )
    plt.xlabel('направление')
    plt.xticks(rotation=45)
    plt.ylabel('количество_вагонов')
    plt.title(f'Динамика количества вагонов по направлениям в {month} месяце')
    plt.show()
```

Откинем выбросы с меньшей стороны и установим ограничение для с бОльшей стороны для наглядности:

In [85]:

```
data['количество_вагонов'].describe()
```

Out[85]:

```
count    9.678000e+03
mean     1.264635e+04
std      8.126644e+04
min      8.700000e+01
25%      2.000000e+02
50%      7.200000e+02
75%      3.420000e+03
max      1.882100e+06
Name: количество_вагонов, dtype: float64
```

In [86]:

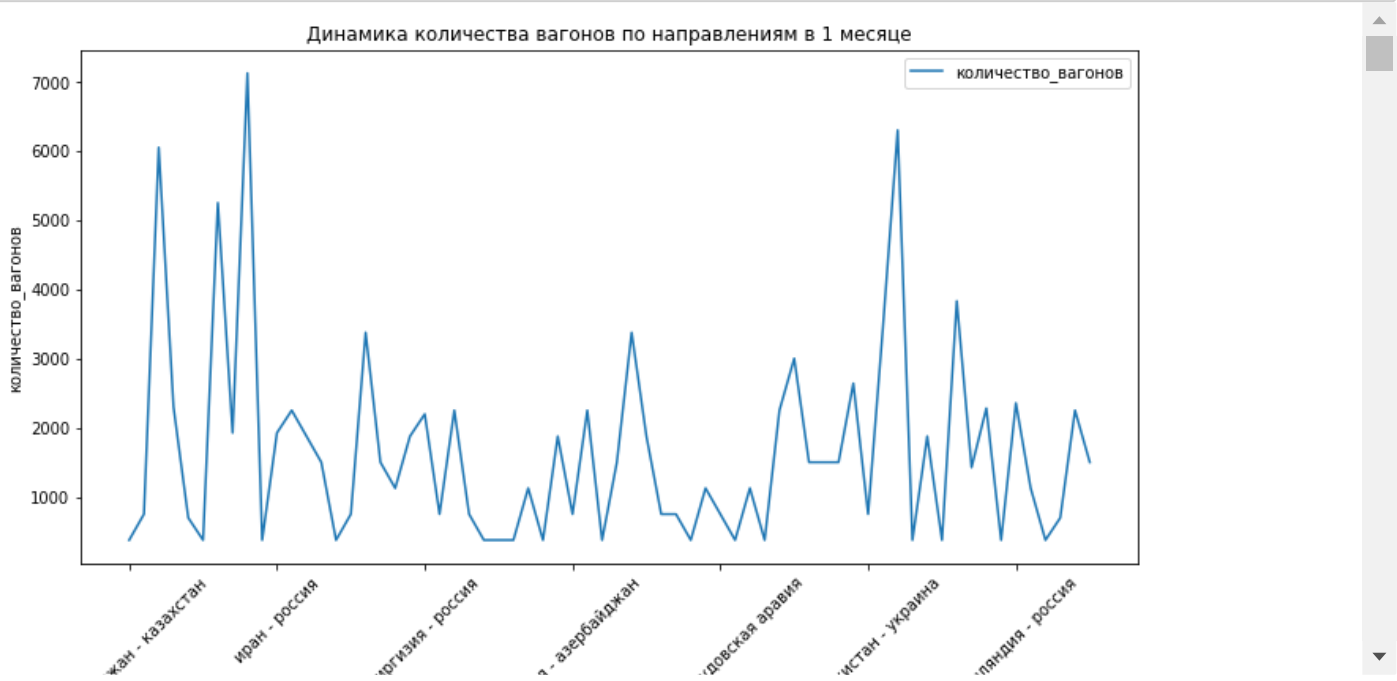
```
data['количество_вагонов'].describe()[5]
```

Out[86]:

```
720.0
```

In [87]:

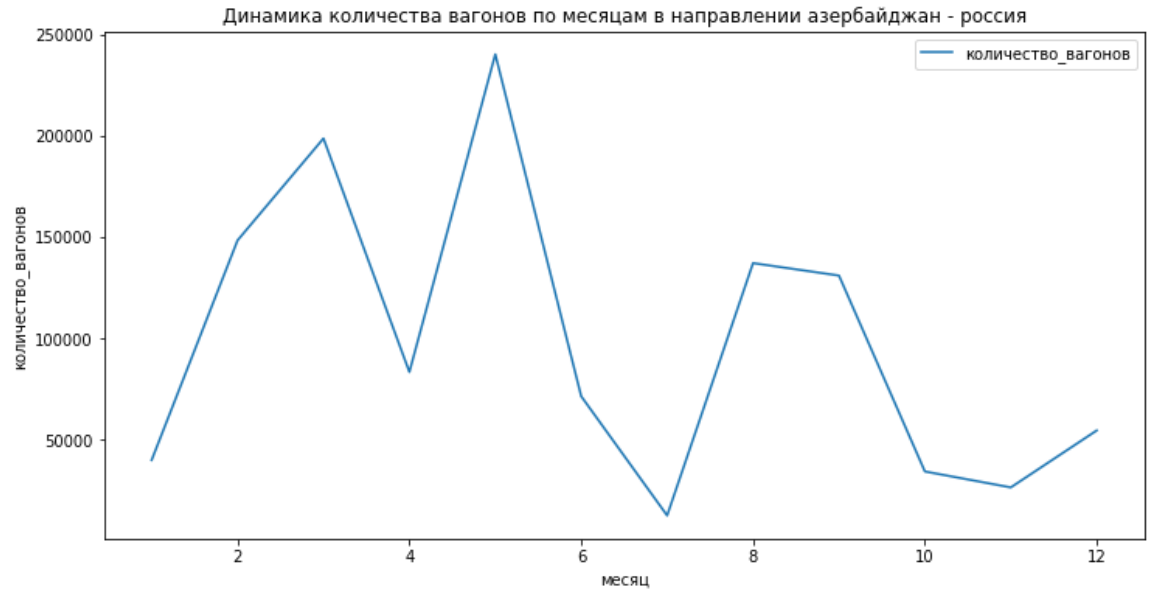
```
for month in range(1, 13):
    tmp = data.query('страна_отправления != "unknown" and страна_прибытия != "unknown" and
    #maximum = tmp['количество_вагонов'].mean() + (0.01 * np.std(tmp['количество_вагонов'])
    maximum = tmp['количество_вагонов'].describe()[5]
    minimum = tmp['количество_вагонов'].mean() - (3 * np.std(tmp['количество_вагонов']))
    three_sigma = tmp.query('@minimum < количество_вагонов < @maximum')
    (three_sigma
     .groupby('направление')
     .agg({'количество_вагонов': 'sum'})
     .reset_index()
     .plot(x='направление', y='количество_вагонов', figsize=(12, 6))
    )
    plt.xlabel('направление')
    plt.xticks(rotation=45)
    plt.ylabel('количество_вагонов')
    plt.title(f'Динамика количества вагонов по направлениям в {month} месяце')
    plt.show()
```



Посмотрим на динамику количества вагонов на отдельном графике каждого направления по месяцам:

In [88]:

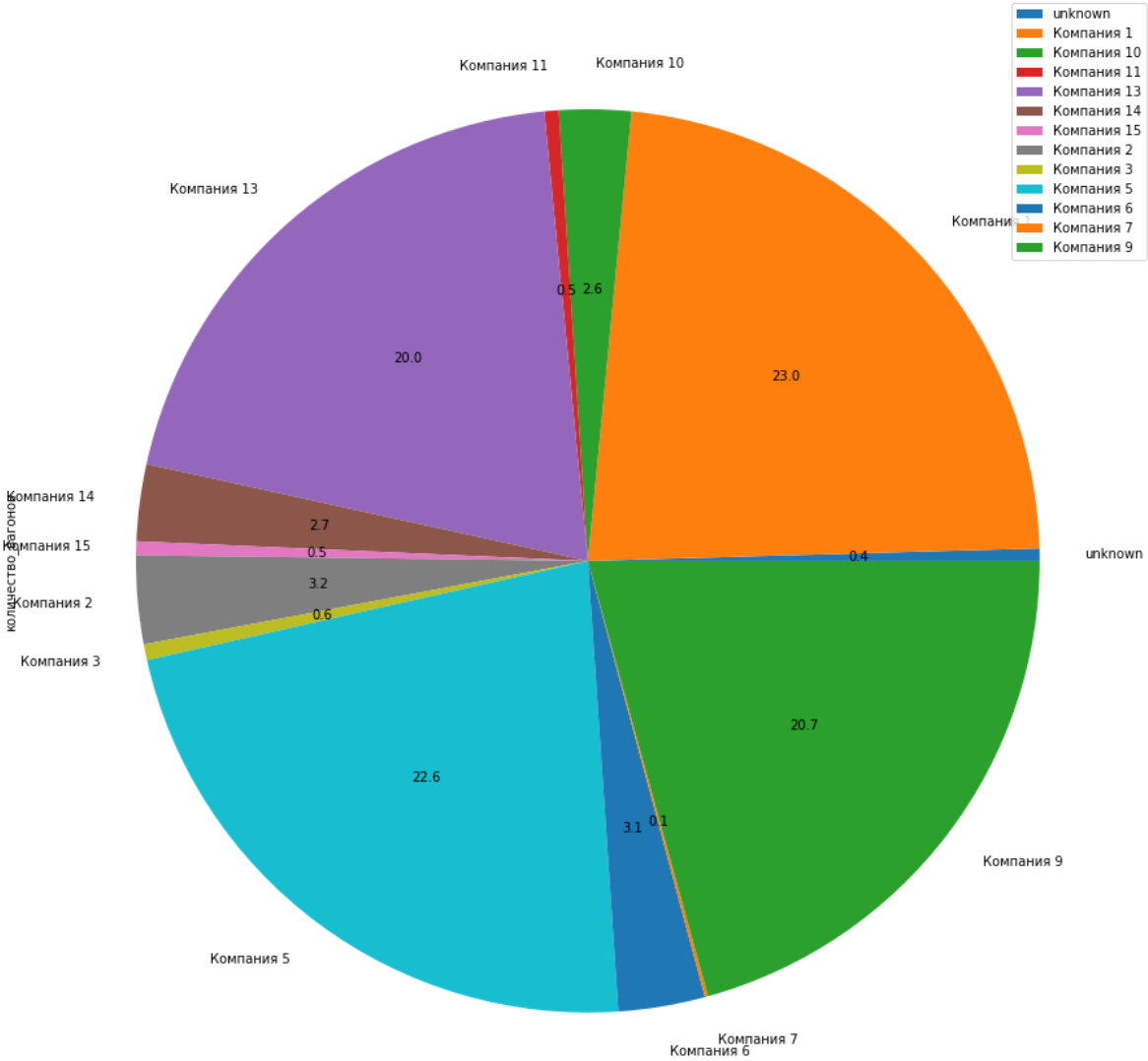
```
for direction in data['направление'].unique():
    min_num_month = len(data[data['направление'] == direction]['месяц'].unique())
    if min_num_month > 10:
        (data[(data['страна_прибытия'] != 'unknown') & (data['страна_отправления'] != 'unkn
         .groupby('месяц')
         .agg({'количество_вагонов': 'sum'})
         .reset_index()
         .plot(x='месяц', y='количество_вагонов', figsize=(12, 6))
        )
        plt.xlabel('месяц')
        plt.ylabel('количество_вагонов')
        plt.title(f'Динамика количества вагонов по месяцам в направлении {direction}')
        plt.show()
```



Построим график piechart распределения долей в количестве вагонов по компаниям за последний квартал по отправлениям из России:

In [89]:

```
(data.query('(месяц == 4 or месяц == 5 or месяц == 6) and страна_отправления == "россия" an
 .groupby('название_компаний')
 .agg({'количество_вагонов': 'sum'})
 .plot(kind='pie', y='количество_вагонов', figsize=(16, 16), autopct = '%.1f')
)
plt.show()
```



Оценим отклонение от нормы в данных по суммарному количеству вагонов в месяц, независимо от типа вагонов в виде распределения Гаусса

In [90]:

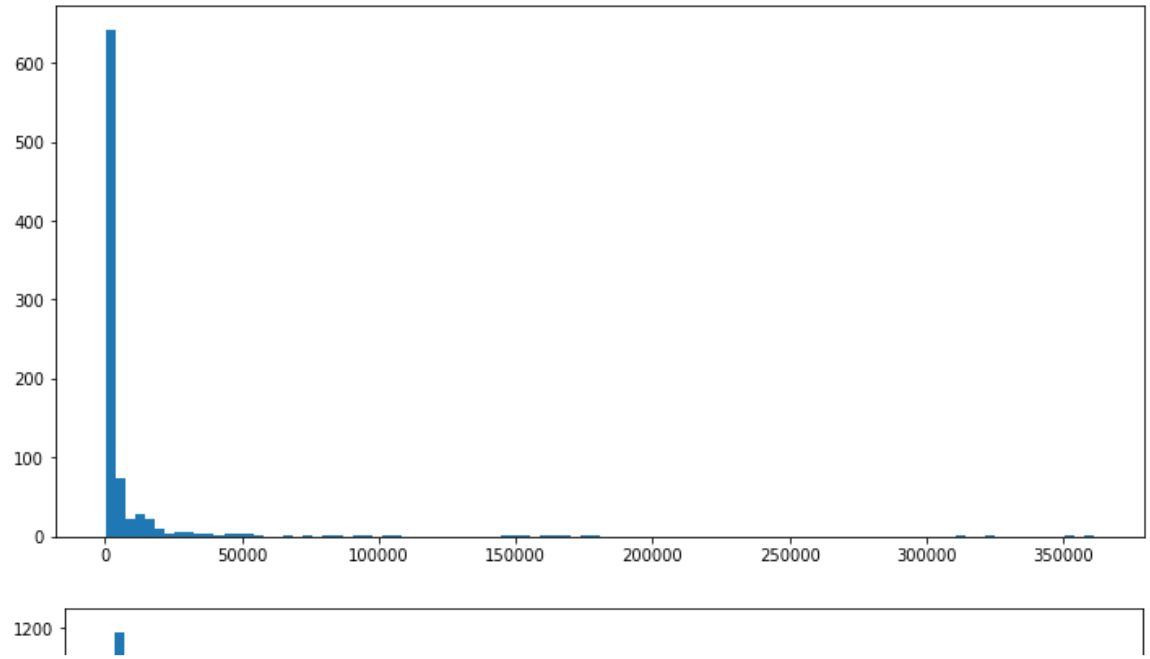
```
sum_of_wagons_months = data.groupby('месяц').agg({'количество_вагонов': 'sum'})
sum_of_wagons_months
```

Out[90]:

месяц	количество_вагонов
1	7.733940e+06
2	1.299435e+07
3	1.543764e+07
4	1.064751e+07
5	1.336421e+07
6	2.178060e+07
7	1.078847e+07
8	8.465872e+06
9	4.792832e+06
10	5.982000e+06
11	6.621918e+06
12	3.782034e+06

In [91]:

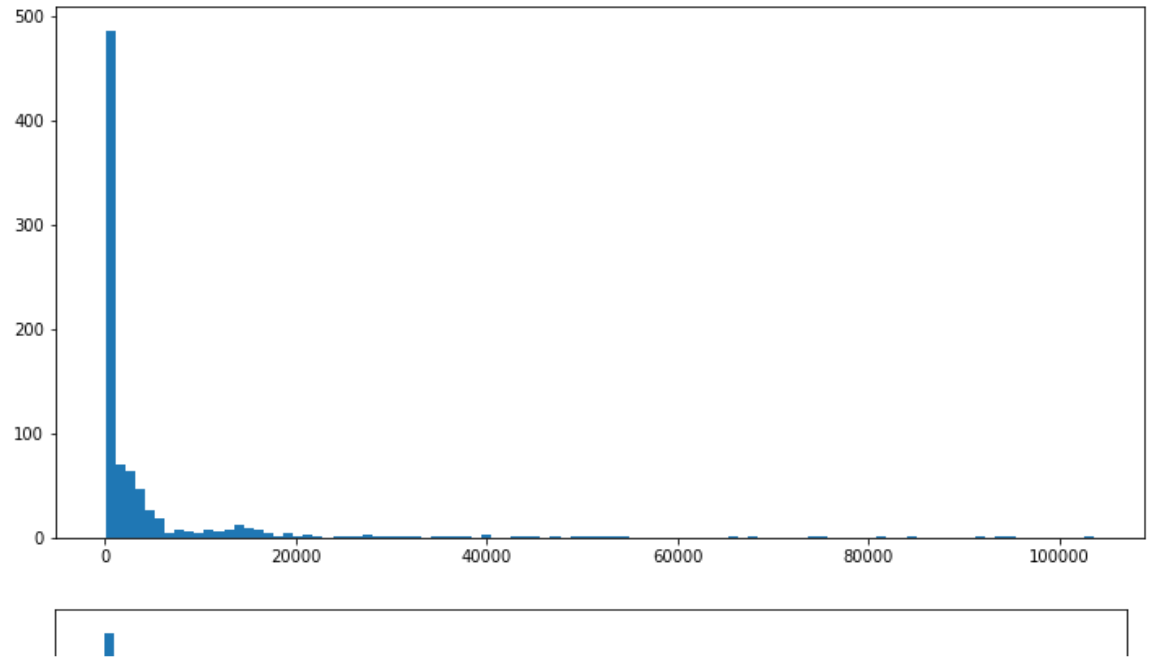
```
for month in range(1, 13):
    tmp = data.query('месяц == @month')
    plt.figure(figsize=(12, 6))
    plt.hist('количество_вагонов', bins=100, data=tmp)
    plt.show()
```



Откинем выбросы по правилу трёмх сигм и снова посмотрим на гистограммы:

In [92]:

```
for month in range(1, 13):
    tmp = data.query('месяц == @month')
    maximum = tmp['количество_вагонов'].mean() + (3 * np.std(tmp['количество_вагонов']))
    minimum = tmp['количество_вагонов'].mean() - (3 * np.std(tmp['количество_вагонов']))
    three_sigma = tmp.query('@minimum < количество_вагонов < @maximum')
    plt.figure(figsize=(12, 6))
    plt.hist('количество_вагонов', bins=100, data=three_sigma)
    plt.show()
```



На мой взгляд стоит откинуть хвосты побольше.

In [93]:

```
data['количество_вагонов'].describe()
```

Out[93]:

```
count    9.678000e+03
mean     1.264635e+04
std      8.126644e+04
min      8.700000e+01
25%      2.000000e+02
50%      7.200000e+02
75%      3.420000e+03
max      1.882100e+06
Name: количество_вагонов, dtype: float64
```

In [94]:

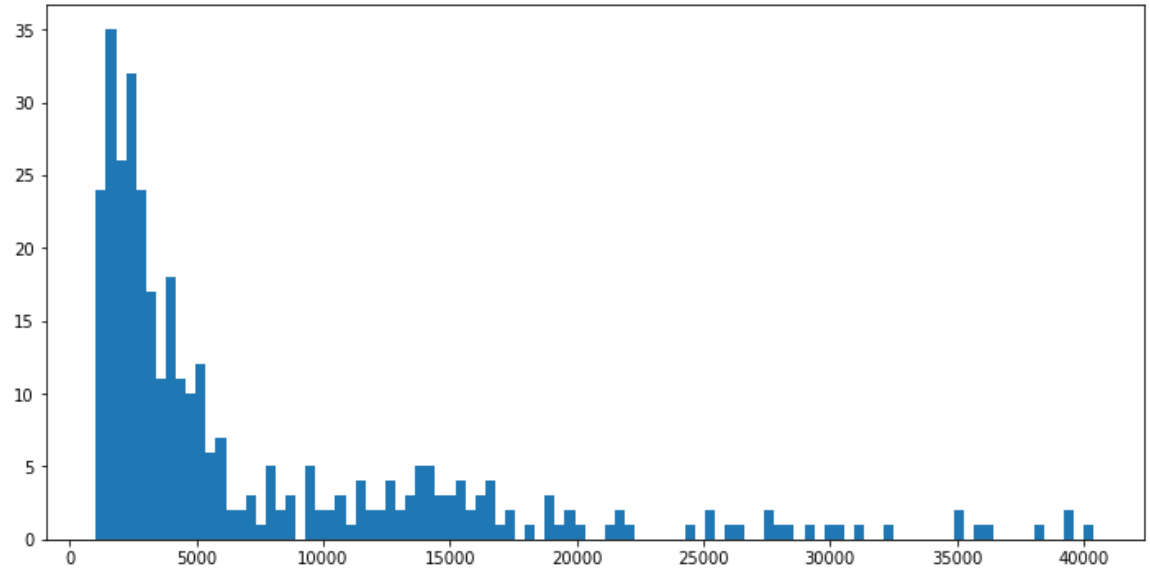
```
data['количество_вагонов'].describe()[4]
```

Out[94]:

```
200.0
```

In [95]:

```
for month in range(1, 13):
    tmp = data.query('месяц == @month')
    maximum = tmp['количество_вагонов'].mean() + (1 * np.std(tmp['количество_вагонов']))
    #minimum = tmp['количество_вагонов'].mean() - (1 * np.std(tmp['количество_вагонов']))
    #minimum = tmp['количество_вагонов'].describe()[4]
    three_sigma = tmp.query('1000 < количество_вагонов < @maximum')
    plt.figure(figsize=(12, 6))
    plt.hist('количество_вагонов', bins=100, data=three_sigma)
    plt.show()
```



Так их можно рассматривать.

Вид распределений похож на распределения Пуассона, а не нормальное распределение Гауса.

Создадим таблицу по посчитанным ошибкам:

Ошибки:

Пропуски:
Вагонов = 70
Выручка = 15

Название компаний(2222) = 25

Год(2 019, 2O19, 2O19) = 20

Месяц(07.01.19**) = 72

Страна отправления(7777777) = 52 Страна отправления(По разному написаны страны и прочие страны) = 80

Тип вагона('Цестерны', 'Полумагоны', 'Крысые', 'Полубагоны') = 72

Страна прибытия(7777777) = 16
Страна прибытия(По разному написаны страны и прочие страны) = 313

Вагонов ('в', 'й', 'у', 'цй', 'к', 'е', 'й', 'цу', 'кцук', 'цу', 'пе', 'кке', 'йц', 'йцу', 'йуйцу') = 15

In [96]:

```
columns = ['Название ошибки', 'Название компаний', 'Год', 'Месяц', 'Страна отправления', 'T
errors = [['Пропуски', 0, 0, 0, 0, 0, 0, 70, 15],
          ['int в string', 25, 0, 0, 52, 0, 16, 0, 0],
          ['Разное написание - один смысл и прочие страны', 0, 0, 0, 80, 0, 313, 0, 0],
          ['string в int', 0, 20, 0, 0, 0, 0, 0, 0],
          ['Предположительные сбои', 0, 0, 72, 0, 0, 0, 0, 0],
          ['Опечатки', 0, 0, 0, 0, 72, 0, 15, 0],
         ]
```

In [97]:

```
errors_tabl = pd.DataFrame(data=errors, columns=columns)
errors_tabl
```

Out[97]:

	Название ошибки	Название компаний	Год	Месяц	Страна отправления	Тип вагона	Страна прибытия	Вагонов	Вы
0	Пропуски	0	0	0	0	0	0	70	
1	int в string	25	0	0	52	0	16	0	
2	Разное написание - один смысл и прочие страны	0	0	0	80	0	313	0	
3	string в int	0	20	0	0	0	0	0	
4	Предположительные сбои	0	0	72	0	0	0	0	
5	Опечатки	0	0	0	0	72	0	15	