Содержание

Ввод [1]:
```
import pandas as pd
from tqdm import tqdm
import folium
from shapely.geometry import Point
from shapely.geometry.polygon import Polygon
import seaborn as sns
import matplotlib.pyplot as plt
from datetime import datetime
import plotly.graph_objects as go
import numpy as np
import plotly.express as px
import ipywidgets as widgets
from IPython.display import display
from ipywidgets import interact
```

1 Скачай библиотеки

Ввод [2]:
```
#!pip install tqdm
#!pip install folium
#!pip install shapely
#!pip install plotly
```

2 Импорты

Ввод [3]:
```
wr  = pd.read_csv('wifi_routers.csv', sep = ';')
```

Ввод [4]:
```
wr
```

Out[4]:

	guid	geom	address_json
0	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	{"rus": "г. Тула, Октябрьская ул. – ул. Демидо...
1	6422a0a5-2c2d-4610-bebc-91722ea37827	POINT (37.5826629190378 54.1688958982062)	{"rus": "г. Тула, ул. Болдина - Оружейная ул."...
2	b17aefd3-8431-4054-a0b5-b0a26eeb9f14	POINT (37.5931054621157 54.1854456631672)	{"rus": "г. Тула, Первомайская ул. – ул. Фрунз...
3	92c1cc9e-cfa4-4ef0-91f0-c0a158f547e7	POINT (37.5726628595483 54.1691301221863)	{"rus": "г. Тула, ул. 9-мая – ул. Макаренко", ...
4	f0058c02-034f-429a-b932-8638089d8718	POINT (37.6230252453024 54.2347433816421)	{"rus": "г. Тула, Аэропорт (Октябрьская ул., 2...
5	37cea6a6-eaaa-4e12-9b4b-b444163a2cc8	POINT (37.616479 54.191903)	{"rus": "г. Тула, Советская ул. – пр-т Ленина"...
6	07190dec-be71-44a2-8d64-c24fb51ebc7b	POINT (37.622509 54.2128)	{"rus": "г. Тула, Октябрьская ул. – ул. Максим...

Ввод [5]:
```
rn = pd.read_csv('road_network.csv', sep = ';')
```

Ввод [6]:
```
rn
```

Out[6]:

	geom	from_vertex_id	to_vertex_id	weight	was_one_way	group_ic
0	LINESTRING (37.081866 54.504539,37.081866 54.5...	3652	3650	17.841263	False	42993.(
1	LINESTRING (38.207016 53.963469,38.207077 53.9...	58925	69671	46.891587	False	NaN
2	LINESTRING (38.207016 53.963469,38.207077 53.9...	58925	69510	60.111670	False	NaN
3	LINESTRING (37.101248 54.581554,37.101109 54.5...	248317	248315	19.512979	False	NaN
4	LINESTRING (37.113535 54.50601,37.11351 54.506...	24102	24100	6.462459	False	41418.(
...	...	...	...	...	...	..
349748	LINESTRING (37.658182 54.237703,37.658616 54.2...	298532	298534	90.974444	False	38000.(
349749	LINESTRING (38.73984 54.273241,38.740602 54.27...	138623	138621	172.788779	False	NaN
349750	LINESTRING (38.446588 54.061748,38.44644 54.06...	100230	100232	33.078981	False	NaN
349751	LINESTRING (37.982225 53.900182,37.98195 53.90...	256839	256837	92.553325	False	NaN
349752	LINESTRING (37.887381 54.542572,37.887218 54.5...	194509	194507	22.617343	False	NaN

349753 rows × 6 columns

3 Визуализация координат

3.1 Создание точек роутеров на карте

Ввод [7]:
```
wr['clean_geom'] = wr['geom'].str.replace('POINT', '')
wr['clean_geom']
```

Out[7]:
```
0                    (37.618886 54.204617)
1      (37.5826629190378 54.1688958982062)
2      (37.5931054621157 54.1854456631672)
3      (37.5726628595483 54.1691301221863)
4      (37.6230252453024 54.2347433816421)
5                    (37.616479 54.191903)
6                      (37.622509 54.2128)
7                    (37.611688 54.195984)
8                    (37.625018 54.218076)
9                    (37.622566 54.189654)
10                    (37.604794 54.18133)
11                   (37.614096 54.193204)
12     (37.6286864157148 54.2276569624141)
13     (37.6245072514748 54.2333462654037)
14                    (37.67935 54.219052)
15     (37.6182365086987 54.2413214831852)
16                   (37.627888 54.191341)
17     (37.6119371506259 54.2491761841739)
18                   (37.625265 54.190082)
```

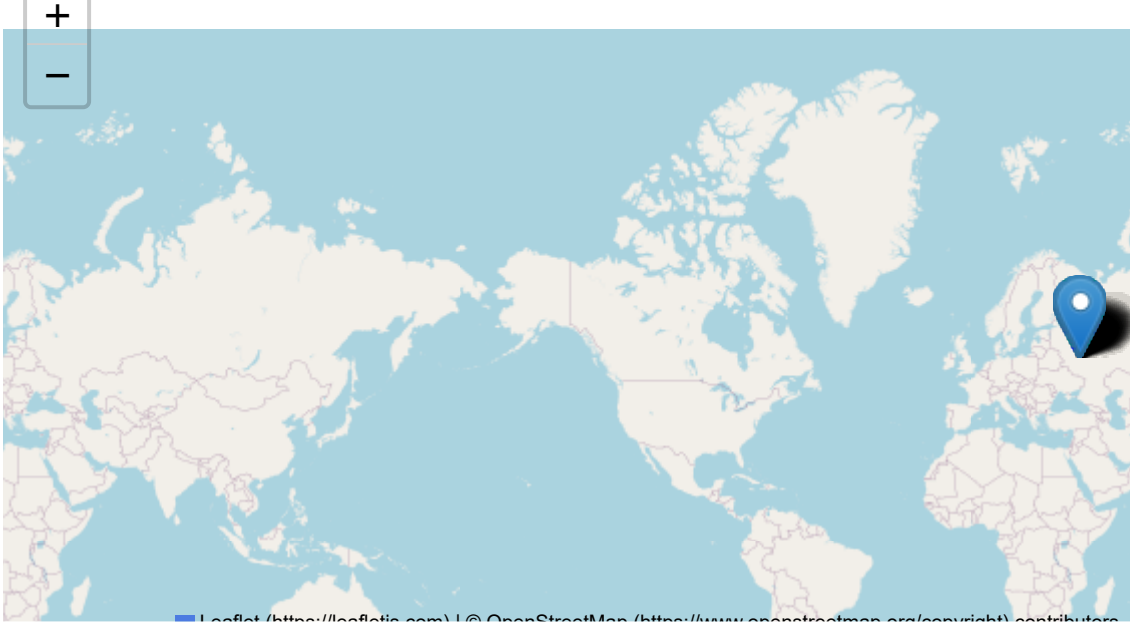
Ввод [8]:
```
wr['clean_geom'] = wr['clean_geom'].replace({'\(': ' ', '\)': ' '}, regex=Tr
```

Ввод [9]:
```
wr['clean_geom']
```

Out[9]:
```
0                    37.618886 54.204617
1       37.5826629190378 54.1688958982062
2       37.5931054621157 54.1854456631672
3       37.5726628595483 54.1691301221863
4       37.6230252453024 54.2347433816421
5                    37.616479 54.191903
6                      37.622509 54.2128
7                    37.611688 54.195984
8                    37.625018 54.218076
9                    37.622566 54.189654
10                    37.604794 54.18133
11                   37.614096 54.193204
12      37.6286864157148 54.2276569624141
13      37.6245072514748 54.2333462654037
14                    37.67935 54.219052
15      37.6182365086987 54.2413214831852
16                   37.627888 54.191341
17      37.6119371506259 54.2491761841739
18                   37.625265 54.190082
19                     37.620844 54.2101
20      37.5855121948686 54.1714546891764
21      37.5799264475165 54.1666267574838
22      37.5754334510529 54.1718766491252
23                   37.619014 54.191034
24      37.5946593824111 54.1646166190679
25      37.5974766750285 54.1839287967087
26                   37.619966 54.208412
27                   37.614644 54.200674
28                   37.619182 54.205679
Name: clean_geom, dtype: object
```

Ввод [10]:
```
map = folium.Map()

for index, row in wr.iterrows():
    location = [float(coord) for coord in row['clean_geom'].strip('[]').spli
    location.reverse()  # Изменяем порядок координат
    marker = folium.Marker(location=location, popup=f'Location {index}')
    marker.add_to(map)

map
```

Out[10]:
Make this Notebook Trusted to load map: File -> Trust Notebook



3.2 Создание линий на карте

Ввод [11]:
```
import re

# Преобразуем каждую строку в колонке geom в список координат
rn['coords'] = rn['geom'].apply(lambda x: [[float(i) for i in coord.split()]

# Получаем список всех координат из столбца coords
line_coords = [coord for sublist in rn['coords'] for coord in sublist]
```

Ввод [12]:
```
rn['flipped_geom'] = rn['geom'].apply(lambda x: ','.join([' '.join(coord.spl

rn['flipped_geom']
```

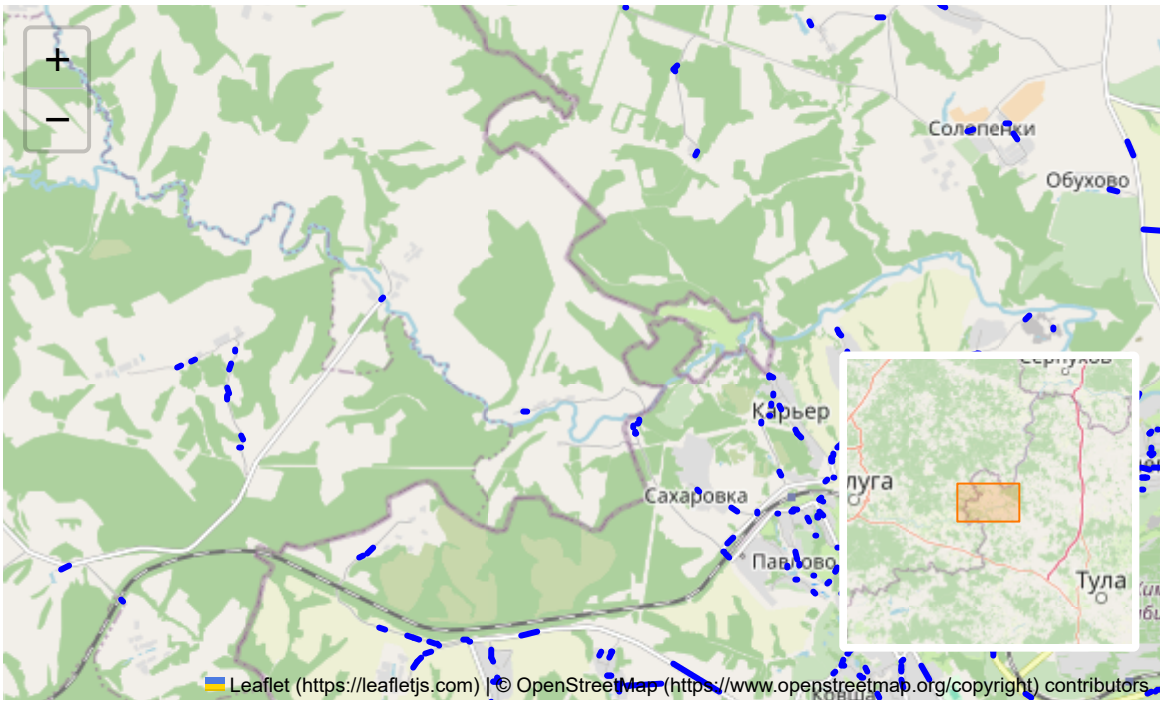
Out[12]:
```
0         54.504539 37.081866,54.50443 37.081866,54.5043...
1         53.963469 38.207016,53.963733 38.207077,53.963...
2         53.963469 38.207016,53.963733 38.207077,53.963...
3         54.581554 37.101248,54.581603 37.101109,54.581...
4         54.50601 37.113535,54.506036 37.11351,54.50605...
                                ...
349748    54.237703 37.658182,54.238107 37.658616,54.238...
349749    54.273241 38.73984,54.2746 38.740602,54.274707...
349750    54.061748 38.446588,54.061882 38.44644,54.0619...
349751    53.900182 37.982225,53.900393 37.98195,53.9008...
349752    54.542572 37.887381,54.542527 37.887218,54.542...
Name: flipped_geom, Length: 349753, dtype: object
```

Ввод [13]:
```
del map
```

Ввод [14]:
```
# import folium
from folium.plugins import MiniMap
from tqdm import tqdm

# Создание основной карты
m = folium.Map(location=[54.5, 37], zoom_start=12)

# Добавление линий на основную карту с прогрессбаром
for i in tqdm(range(len(rn['flipped_geom'])), desc='Добавление линий'):
    if i % 20 == 0:
        line = rn['flipped_geom'][i]
        coords = [list(map(float, point.split(' '))) for point in line.lstri
        folium.PolyLine(locations=coords, color='blue').add_to(m)

# Создание мини-карты
mini_map = MiniMap()
m.add_child(mini_map)

# Отображение основной карты
m
```

```
Добавление линий: 100%|██████████████████████████████████████████████
| 349753/349753 [00:01<00:00, 314992.15it/s]
```

Out[14]:



3.3 Очистка router network от координат, не входящих в площадь

Ввод [15]:
```
rn['cleaned_geom'] = [item.split(",")[0].strip() for item in rn['flipped_geo
rn['cleaned_geom']
```

Out[15]:
```
0         54.504539 37.081866
1         53.963469 38.207016
2         53.963469 38.207016
3         54.581554 37.101248
4          54.50601 37.113535
                  ...
349748    54.237703 37.658182
349749     54.273241 38.73984
349750    54.061748 38.446588
349751    53.900182 37.982225
349752    54.542572 37.887381
Name: cleaned_geom, Length: 349753, dtype: object
```

Ввод [16]:
```
rn['cleaned_geom'] = rn['cleaned_geom'].str.replace(' ', ',')
```

Ввод [17]:
```
del m
```

Ввод [18]:
```
rn['cleaned_geom']
```

Out[18]:
```
0         54.504539,37.081866
1         53.963469,38.207016
2         53.963469,38.207016
3         54.581554,37.101248
4          54.50601,37.113535
                  ...
349748    54.237703,37.658182
349749     54.273241,38.73984
349750    54.061748,38.446588
349751    53.900182,37.982225
349752    54.542572,37.887381
Name: cleaned_geom, Length: 349753, dtype: object
```

Ввод [19]:
```
# Создание полигона для квадрата
square_coords = [(37.70902904943893, 54.25855193504911),
                 (37.56590025352726, 54.25855193504911),
                 (37.56590025352726, 54.167450582590305),
                 (37.70902904943893, 54.167450582590305)]
polygon = Polygon(square_coords)

# Функция для проверки, находится ли точка внутри полигона
def point_inside_square(coord):

        lat, lon = map(float, coord.split(','))  # Разделяем строку координа
        point = Point(lon, lat)  # Создаем объект точки
        return polygon.contains(point)  # Проверяем, находится ли точка внут

# Применение функции к столбцу cleaned_geom
rn['is_inside_square'] = rn['cleaned_geom'].apply(point_inside_square)

# Удаление строк, для которых is_inside_square равно False
rn = rn[rn['is_inside_square']]
```

Ввод [20]:
```
rn
```

Out[20]:

	geom	from_vertex_id	to_vertex_id	weight	was_one_way	group
15	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	24602	158.805103	False	N
16	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	62722	207.237527	False	N
17	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	24601	158.519475	False	N
25	LINESTRING (37.599466 54.211929,37.59949 54.21...	129918	129920	7.188256	False	4034

Ввод [21]:
```
rn['is_inside_square'] = rn['cleaned_geom'].apply(point_inside_square)

# Удаление строк, для которых is_inside_square равно False
rn = rn[rn['is_inside_square']]
```

```
C:\Users\Ruslan\AppData\Local\Temp\ipykernel_15448\4179812677.py:1: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://p
andas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-vi
ew-versus-a-copy)
  rn['is_inside_square'] = rn['cleaned_geom'].apply(point_inside_square)
```

Ввод [22]:
```
rn.reset_index()
```

Out[22]:

	index	geom	from_vertex_id	to_vertex_id	weight	was_one_way	g
0	15	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	24602	158.805103	False	
1	16	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	62722	207.237527	False	
2	17	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	24601	158.519475	False	
3	25	LINESTRING (37.599466 54.211929,37.59949 54.21...	129918	129920	7.188256	False	
4	62	LINESTRING (37.590964 54.192815,37.591059 54.1...	262461	262459	16.193512	False	
...	...	...	...	...	...	...	
25562	349711	LINESTRING (37.660305 54.204487,37.660334 54.2...	16501	16503	16.716789	False	
25563	349742	LINESTRING (37.585576 54.187689,37.584933 54.1...	1518	1520	96.022967	False	
25564	349743	LINESTRING (37.585576 54.187689,37.584933 54.1...	1518	1530	106.381193	False	
25565	349745	LINESTRING (37.688395 54.234531,37.688325 54.2...	288258	288260	20.798847	False	
25566	349748	LINESTRING (37.658182 54.237703,37.658616 54.2...	298532	298534	90.974444	False	

25567 rows × 11 columns

Ввод [23]:
```
rn['flipped_geom'] = rn['geom'].apply(lambda x: ','.join([' '.join(coord.spl

rn['flipped_geom'].reset_index()
```

Out[23]:

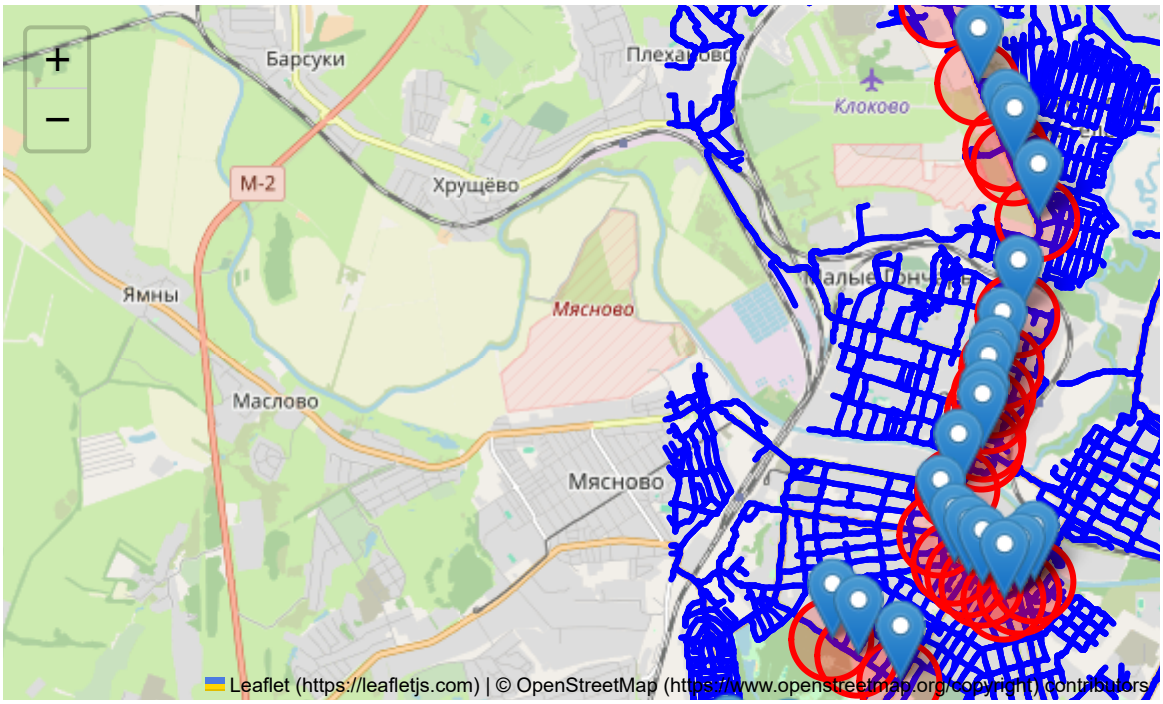
	index	flipped_geom
0	15	54.200393 37.686139,54.20105 37.687477,54.2007...
1	16	54.200393 37.686139,54.20105 37.687477,54.2015...
2	17	54.200393 37.686139,54.20105 37.687477,54.2013...
3	25	54.211929 37.599466,54.211924 37.59949,54.2119...
4	62	54.192815 37.590964,54.192822 37.591059,54.192...
...	...	...
25562	349711	54.204487 37.660305,54.204468 37.660334,54.204...
25563	349742	54.187689 37.585576,54.187356 37.584933,54.187...
25564	349743	54.187689 37.585576,54.187356 37.584933,54.187...
25565	349745	54.234531 37.688395,54.234653 37.688325,54.234...
25566	349748	54.237703 37.658182,54.238107 37.658616,54.238...

Ввод [24]:
```
rn = rn.reset_index(drop=True)
```

Ввод [25]:
```
m = folium.Map(location=[54.19, 37.62], zoom_start=12)

# Добавление линий на карту с прогрессбаром
for i in tqdm(range(len(rn['flipped_geom'])), desc='Добавление линий'):
    #if i % 20 == 0:
        line = rn['flipped_geom'][i]
        coords = [list(map(float, point.split(' '))) for point in line.lstri
        folium.PolyLine(locations=coords, color='blue').add_to(m)

        #_____________________
for index, row in wr.iterrows():
    location = [float(coord) for coord in row['clean_geom'].strip('[]').spli
    location.reverse()  # Изменяем порядок координат
    marker = folium.Marker(location=location, popup=f'Location {index}')
    circle = folium.CircleMarker(location=location, radius=20, color='red',
    marker.add_to(m)
    circle.add_to(m)
# Отображение карты
m
```

```
Добавление линий: 100%|██████████████████████████████████████████████
███| 25567/25567 [00:01<00:00, 17388.23it/s]
```

Out[25]:



3.4 Инициализация точек и дорог на одной карте

```
m = folium.Map(location=[54.19, 37.62], zoom_start=12)

# Добавление линий на карту с прогрессбаром
for i in tqdm(range(len(rn['flipped_geom'])), desc='Добавление линий'):
    #if i % 20 == 0:
        line = rn['flipped_geom'][i]
        coords = [list(map(float, point.split(' '))) for point in
line.lstrip('LINESTRING (').rstrip(')').split(',')]
        folium.PolyLine(locations=coords, color='blue').add_to(m)

        #_____________________
for index, row in wr.iterrows():
```

```
    location = [float(coord) for coord in
row['clean_geom'].strip('[]').split()]
    location.reverse()  # Изменяем порядок координат
    marker = folium.Marker(location=location, popup=f'Location {index}')
    marker.add_to(m)
# Отображение карты
m
```

4 Прогонка небольших логов wifi_logs_2022_12_01_202312081829

Ввод [26]:
```
df1 = pd.read_csv('wifi_logs_2022_12_01_202312081829.csv', sep = ';')
```

Ввод [27]:
```
df1
```

Out[27]:

	guid	tm	router_mac	user_mac	signal	router_id
0	1a25652b-f346-4ffb-aec3-295ecf08fd97	2022-12-01 03:00:03.000 +0300	CC:2D:E0:82:B8:DD	0E:AC:4A:34:2A:F1	-68.0	cdcab1(cc3a-46 ae! 024216190a
1	f250e9af-498e-42e7-9439-7a82473b829a	2022-12-01 03:00:09.000 +0300	CC:2D:E0:82:B9:07	56:36:9B:28:28:F8	-67.0	8a823f(771f-42(89 928cba314a
2	525c9f06-fc70-426e-81b3-ed96497b6f01	2022-12-01 03:00:10.000 +0300	CC:2D:E0:82:B9:07	E0:63:DA:DC:D8:49	-61.0	8a823f(771f-42(89 928cba314a
3	cf66e62d-a621-4aed-a46a-b6bc135ce766	2022-12-01 03:00:14.000 +0300	CC:2D:E0:82:B9:07	1C:15:1F:CA:88:EE	-70.0	8a823f(771f-42(89 928cba314a
4	d3554553-5555-4cfd-8d01-3f56fea0a589	2022-12-01 03:00:17.000 +0300	CC:2D:E0:82:B9:40	D0:37:45:C1:E1:1E	-66.0	3d949(fb08-4c 80 ca744a6d9a
...	...	...	...	...	...	
463699	a9e0a25f-0ae0-46c1-8dba-6d9c7b52eb9e	2022-12-02 02:59:57.000 +0300	48:8F:5A:AC:81:1D	3C:84:6A:4F:1A:54	-72.0	8a7346 e451-4f 8b 3b3b46e334
463700	79af3fc4-f500-46b0-8e3c-66787e31b158	2022-12-02 02:59:57.000 +0300	48:8F:5A:AC:81:1D	C8:3A:35:EA:DD:31	-75.0	8a7346 e451-4f 8b 3b3b46e334
463701	fcc9465e-c40f-491d-8de4-36a7da784a20	2022-12-02 02:59:59.000 +0300	48:8F:5A:AC:81:1D	B0:95:75:DA:D6:00	-70.0	8a7346 e451-4f 8b 3b3b46e334
463702	2f287ac9-0581-4103-9028-f37d47dc7263	2022-12-02 02:59:23.000 +0300	CC:2D:E0:F3:8C:6E	F8:34:41:CD:71:07	-69.0	0b91ef 3b4f-4a b3 9683c6fd78
463703	03595db5-4efb-40a1-8851-82652a5c6d3d	2022-12-02 02:59:26.000 +0300	CC:2D:E0:F3:8C:6E	A6:4C:84:74:9A:1D	-73.0	0b91ef 3b4f-4a b3 9683c6fd78

463704 rows × 6 columns

Ввод [28]:
```
df1.dtypes
```

Out[28]:
```
guid           object
tm             object
router_mac     object
user_mac       object
signal        float64
router_id      object
dtype: object
```

Ввод [29]:
```
df1['tm'] = pd.to_datetime(df1['tm'], format='%Y-%m-%d %H:%M:%S.%f %z')
df1['tm'] = df1['tm'].dt.round('H')
```

Ввод [30]:
```
df1
```

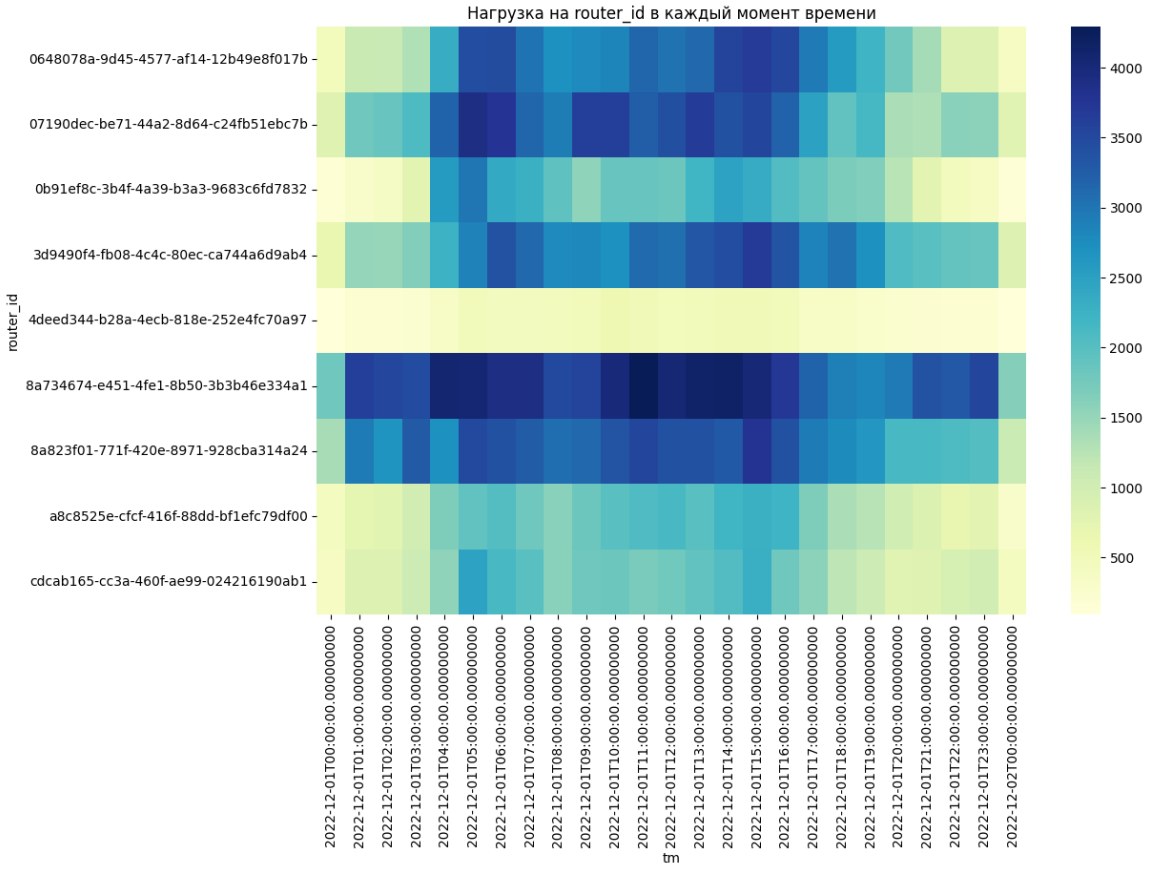
Out[30]:

	guid	tm	router_mac	user_mac	signal	router_id
0	1a25652b-f346-4ffb-aec3-295ecf08fd97	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B8:DD	0E:AC:4A:34:2A:F1	-68.0	cdca cc3a- 02421619
1	f250e9af-498e-42e7-9439-7a82473b829a	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B9:07	56:36:9B:28:28:F8	-67.0	8a82 771f- 928cba31
2	525c9f06-fc70-426e-81b3-ed96497b6f01	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B9:07	E0:63:DA:DC:D8:49	-61.0	8a82 771f- 928cba31
3	cf66e62d-a621-4aed-a46a-b6bc135ce766	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B9:07	1C:15:1F:CA:88:EE	-70.0	8a82 771f- 928cba31
4	d3554553-5555-4cfd-8d01-3f56fea0a589	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B9:40	D0:37:45:C1:E1:1E	-66.0	3d94 fb08- ca744a6c
...	...	...	...	...	...	
463699	a9e0a25f-0ae0-46c1-8dba-6d9c7b52eb9e	2022-12-02 03:00:00+03:00	48:8F:5A:AC:81:1D	3C:84:6A:4F:1A:54	-72.0	8a73- e451- 3b3b46e3
463700	79af3fc4-f500-46b0-8e3c-66787e31b158	2022-12-02 03:00:00+03:00	48:8F:5A:AC:81:1D	C8:3A:35:EA:DD:31	-75.0	8a73- e451- 3b3b46e3
463701	fcc9465e-c40f-491d-8de4-36a7da784a20	2022-12-02 03:00:00+03:00	48:8F:5A:AC:81:1D	B0:95:75:DA:D6:00	-70.0	8a73- e451- 3b3b46e3
463702	2f287ac9-0581-4103-9028-f37d47dc7263	2022-12-02 03:00:00+03:00	CC:2D:E0:F3:8C:6E	F8:34:41:CD:71:07	-69.0	0b91 3b4f- 9683c6fc
463703	03595db5-4efb-40a1-8851-82652a5c6d3d	2022-12-02 03:00:00+03:00	CC:2D:E0:F3:8C:6E	A6:4C:84:74:9A:1D	-73.0	0b91 3b4f- 9683c6fc

463704 rows × 6 columns

4.1 Heatmap лога

Ввод [31]:
```
# Сначала сгруппируем данные
grouped_data = df1.groupby(['router_id', 'tm'])['user_mac'].count().reset_in
pivot_table = grouped_data.pivot(index='router_id', columns='tm', values='us

# Теперь построим тепловую карту
plt.figure(figsize=(12, 8))
sns.heatmap(pivot_table, cmap='YlGnBu')
plt.title('Нагрузка на router_id в каждый момент времени')
plt.show()
```

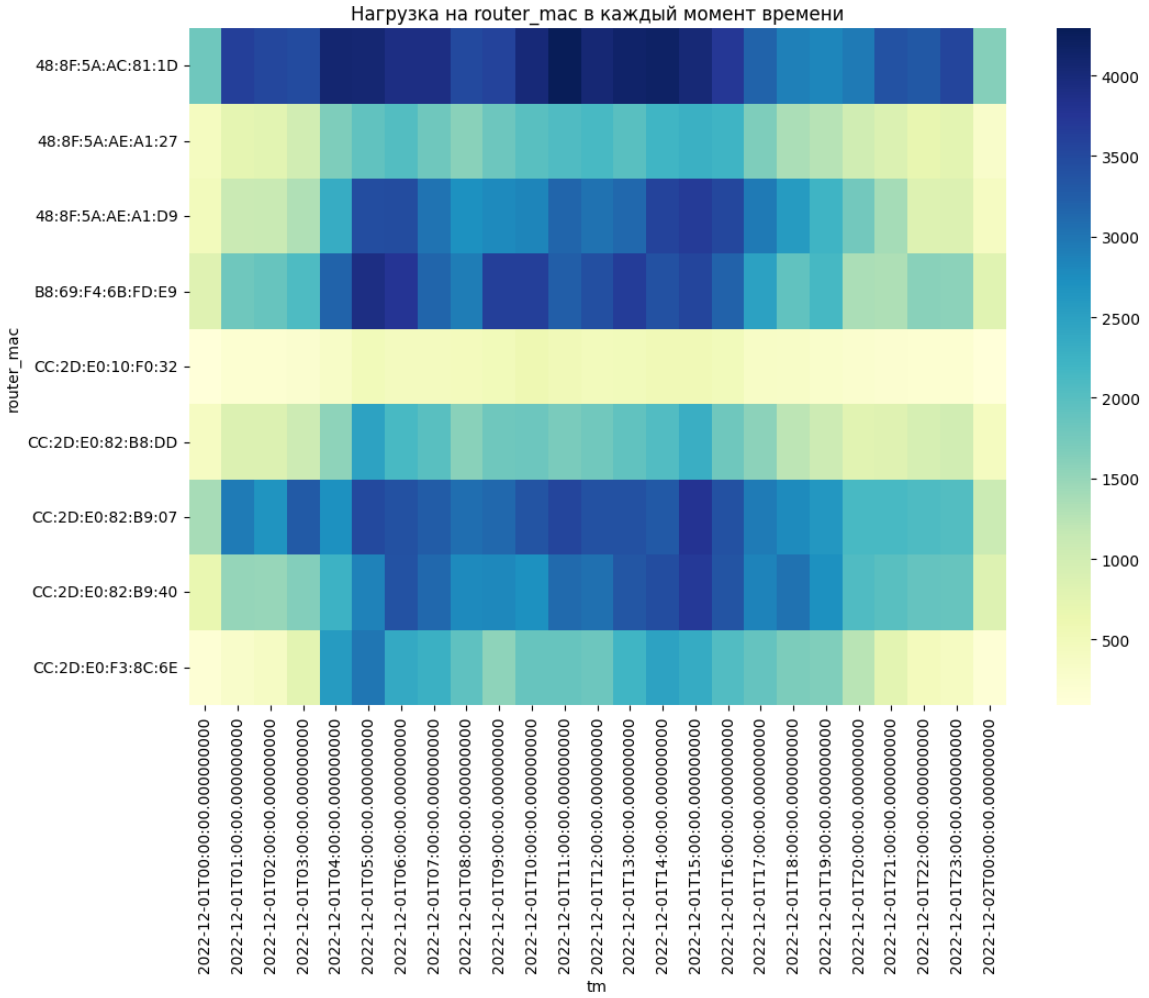


Ввод [32]:
```
df1.groupby(['router_mac', 'tm'])['user_mac'].count()
```

Out[32]:
```
router_mac         tm
48:8F:5A:AC:81:1D  2022-12-01 03:00:00+03:00    1813
                   2022-12-01 04:00:00+03:00    3628
                   2022-12-01 05:00:00+03:00    3526
                   2022-12-01 06:00:00+03:00    3488
                   2022-12-01 07:00:00+03:00    4078
                                                ...
CC:2D:E0:F3:8C:6E  2022-12-01 23:00:00+03:00    1243
                   2022-12-02 00:00:00+03:00     756
                   2022-12-02 01:00:00+03:00     455
                   2022-12-02 02:00:00+03:00     403
                   2022-12-02 03:00:00+03:00     143
Name: user_mac, Length: 225, dtype: int64
```

4.2 Проверка картины на соответствие с id & mac

Ввод [33]:
```
# Сначала сгруппируем данные
grouped_data = df1.groupby(['router_mac', 'tm'])['user_mac'].count().reset_i
pivot_table = grouped_data.pivot(index='router_mac', columns='tm', values='u

# Теперь построим тепловую карту
plt.figure(figsize=(12, 8))
sns.heatmap(pivot_table, cmap='YlGnBu')
plt.title('Нагрузка на router_mac в каждый момент времени')
plt.show()
```



Ввод [34]:
```
wr['address_json'].loc[0]
```

Out[34]:
```
'{"rus": "г. Тула, Октябрьская ул. – ул. Демидовская плотина (Оружейный пе
р.)", "tur": "No TUR", "usa": "No USA"}'
```

Ввод [35]:
```
wr['address_json'] = wr['address_json'].str.split('", "').str[0]
```

Ввод [36]:
```
wr['address_json'] = wr['address_json'].str.split('": "').str[1]
```

Ввод [37]:
```
wr['address_json'].unique()
```

Out[37]:
```
array(['г. Тула, Октябрьская ул. – ул. Демидовская плотина (Оружейный пе
р.)',
       'г. Тула, ул. Болдина - Оружейная ул.',
       'г. Тула, Первомайская ул. – ул. Фрунзе',
       'г. Тула, ул. 9-мая – ул. Макаренко',
       'г. Тула, Аэропорт (Октябрьская ул., 250)',
       'г. Тула, Советская ул. – пр-т Ленина',
       'г. Тула, Октябрьская ул. – ул. Максима Горького',
       'г. Тула, Советская ул. – Красноармейский пр-т',
       'г. Тула, Октябрьская ул. – ул. Пузакова',
       'г. Тула, Советская ул. – Оборонная ул.',
       'г. Тула, пр-т Ленина – Первомайская ул.',
       'г. Тула, Советская ул. – ул. Фридриха Энгельса',
       'г. Тула, Октябрьская ул. – ул. Неман Н.',
       'г. Тула, Октябрьская ул. – ул. Судейского',
       'г. Тула, ул. Щегловская Засека, д. 5',
       'г. Тула, Октябрьская ул. – ул. Карпова',
       'г. Тула, Советская ул. – ул. Дзержинского',
       'г. Тула, Октябрьская ул. – Большая ул.',
```

Ввод [38]:
```
wr.groupby('geom')['address_json'].count()
```

Out[38]:
```
geom
POINT (37.5726628595483 54.1691301221863)    1
POINT (37.5754334510529 54.1718766491252)    1
POINT (37.5799264475165 54.1666267574838)    1
POINT (37.5826629190378 54.1688958982062)    1
POINT (37.5855121948686 54.1714546891764)    1
POINT (37.5931054621157 54.1854456631672)    1
POINT (37.5946593824111 54.1646166190679)    1
POINT (37.5974766750285 54.1839287967087)    1
POINT (37.604794 54.18133)                   1
POINT (37.611688 54.195984)                  1
POINT (37.6119371506259 54.2491761841739)    1
POINT (37.614096 54.193204)                  1
POINT (37.614644 54.200674)                  1
POINT (37.616479 54.191903)                  1
POINT (37.6182365086987 54.2413214831852)    1
POINT (37.618886 54.204617)                  1
POINT (37.619014 54.191034)                  1
POINT (37.619182 54.205679)                  1
```

Ввод [39]:
```
wr['guid']
```

Out[39]:
```
0     0648078a-9d45-4577-af14-12b49e8f017b
1     6422a0a5-2c2d-4610-bebc-91722ea37827
2     b17aefd3-8431-4054-a0b5-b0a26eeb9f14
3     92c1cc9e-cfa4-4ef0-91f0-c0a158f547e7
4     f0058c02-034f-429a-b932-8638089d8718
5     37cea6a6-eaaa-4e12-9b4b-b444163a2cc8
6     07190dec-be71-44a2-8d64-c24fb51ebc7b
7     8a823f01-771f-420e-8971-928cba314a24
8     8a734674-e451-4fe1-8b50-3b3b46e334a1
9     cdcab165-cc3a-460f-ae99-024216190ab1
10    64642101-76de-4d38-9a09-c595b0c310d5
11    b42c687a-0dc2-49d1-9d6e-0d9e514d29bf
12    4cf88651-6c76-46a4-80d0-5dea3e5bcce2
13    7a04763d-e3ba-4e45-b1df-1d322a449030
14    4deed344-b28a-4ecb-818e-252e4fc70a97
15    86b9b151-ae96-45b1-97b2-20558726245c
16    3d9490f4-fb08-4c4c-80ec-ca744a6d9ab4
17    be579fae-23ff-48ed-a400-0691c6075faa
18    7080fd33-a510-4b1a-af7e-99fb5abc29d7
```

Ввод [40]:
```
df1.groupby('router_mac')['router_id'].nunique()
```

Out[40]:
```
router_mac
48:8F:5A:AC:81:1D    1
48:8F:5A:AE:A1:27    1
48:8F:5A:AE:A1:D9    1
B8:69:F4:6B:FD:E9    1
CC:2D:E0:10:F0:32    1
CC:2D:E0:82:B8:DD    1
CC:2D:E0:82:B9:07    1
CC:2D:E0:82:B9:40    1
CC:2D:E0:F3:8C:6E    1
Name: router_id, dtype: int64
```

Ввод [41]:
```
df1[df1['router_id'] == '0648078a-9d45-4577-af14-12b49e8f017b']
```

Out[41]:

	guid	tm	router_mac	user_mac	signal	router_id
34	605990ad-a4a3-4327-ac2e-793500068a07	2022-12-01 03:00:00+03:00	48:8F:5A:AE:A1:D9	68:D7:9A:A2:74:60	-69.0	06480 9d45-4 a 12b49e8f0
35	d7bad818-1471-4db6-a29f-62cb44de8f36	2022-12-01 03:00:00+03:00	48:8F:5A:AE:A1:D9	46:D5:F2:29:ED:8C	-64.0	06480 9d45-4 a 12b49e8f0
36	31334b4f-e6d3-47a3-84aa-13d03effa3f5	2022-12-01 03:00:00+03:00	48:8F:5A:AE:A1:D9	6C:5A:B0:A1:DB:8E	-68.0	06480 9d45-4 a 12b49e8f0
95	651a0be7-e685-4d91-b14c-4c7e69af2072	2022-12-01 03:00:00+03:00	48:8F:5A:AE:A1:D9	68:D7:9A:A2:74:60	-69.0	06480 9d45-4 a 12b49e8f0
96	0066ae6d-f742-4c84-800e-7f235ab01053	2022-12-01 03:00:00+03:00	48:8F:5A:AE:A1:D9	46:D5:F2:29:ED:8C	-64.0	06480 9d45-4 a 12b49e8f0
...	...	...	...	...	...	
463658	14f3caf8-f4c3-40fa-a05c-a196039f9a0b	2022-12-02 03:00:00+03:00	48:8F:5A:AE:A1:D9	6C:5A:B0:A1:DB:8E	-68.0	06480 9d45-4 a 12b49e8f0
463659	b7776026-6c56-4498-a344-0205b3b3e82c	2022-12-02 03:00:00+03:00	48:8F:5A:AE:A1:D9	68:D7:9A:A2:74:60	-69.0	06480 9d45-4 a 12b49e8f0
463688	c3f30413-6388-4c07-acd1-d79dbd165a3c	2022-12-02 03:00:00+03:00	48:8F:5A:AE:A1:D9	68:D7:9A:A2:74:60	-71.0	06480 9d45-4 a 12b49e8f0
463689	1ffcb435-be6b-4b25-ac2a-2833572c24d6	2022-12-02 03:00:00+03:00	48:8F:5A:AE:A1:D9	6C:5A:B0:A1:DB:8E	-69.0	06480 9d45-4 a 12b49e8f0
463690	9cadfc89-e791-4b60-aa9b-cb88ca37a72e	2022-12-02 03:00:00+03:00	48:8F:5A:AE:A1:D9	C4:5D:83:FD:E5:01	-73.0	06480 9d45-4 a 12b49e8f0

57752 rows × 6 columns

Заджойнить df1 & wr по колонке router id. После этого у нас к каждому роутеру будет соотнесена улица, что позволит раскрасить её в соответствии с баллом загруженности дорог

Ввод [42]:
```
wr1 = wr.copy()
```

Ввод [43]:
```
wr1.columns = ['router_id', 'geom', 'address_json', 'clean_geom']
```

Ввод [44]:
```
df2 = pd.merge(wr1, df1, on='router_id')
```

Ввод [45]:
```
df2.head()
```

Out[45]:

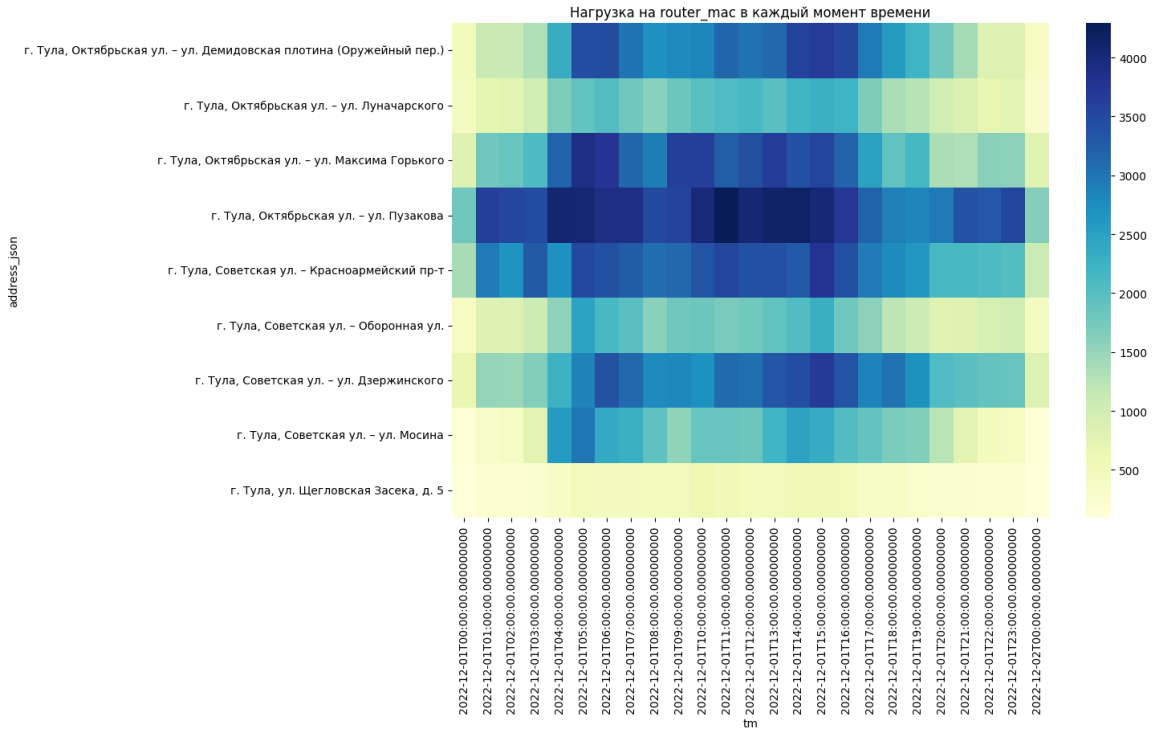
	router_id	geom	address_json	clean_geom	guid	tm	router_
0	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	605990ad-a4a3-4327-ac2e-793500068a07	2022-12-01 03:00:00+03:00	48:8F:5
1	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	d7bad818-1471-4db6-a29f-62cb44de8f36	2022-12-01 03:00:00+03:00	48:8F:5
2	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	31334b4f-e6d3-47a3-84aa-13d03effa3f5	2022-12-01 03:00:00+03:00	48:8F:5
3	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	651a0be7-e685-4d91-b14c-4c7e69af2072	2022-12-01 03:00:00+03:00	48:8F:5
4	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	0066ae6d-f742-4c84-800e-7f235ab01053	2022-12-01 03:00:00+03:00	48:8F:5

Ввод [46]:
```
df2.groupby('router_mac')['address_json'].unique()
```

Out[46]:
```
router_mac
48:8F:5A:AC:81:1D            [г. Тула, Октябрьская ул. – ул. Пузакова]
48:8F:5A:AE:A1:27         [г. Тула, Октябрьская ул. – ул. Луначарского]
48:8F:5A:AE:A1:D9     [г. Тула, Октябрьская ул. – ул. Демидовская пл...
B8:69:F4:6B:FD:E9     [г. Тула, Октябрьская ул. – ул. Максима Горького]
CC:2D:E0:10:F0:32               [г. Тула, ул. Щегловская Засека, д. 5]
CC:2D:E0:82:B8:DD              [г. Тула, Советская ул. – Оборонная ул.]
CC:2D:E0:82:B9:07       [г. Тула, Советская ул. – Красноармейский пр-т]
CC:2D:E0:82:B9:40           [г. Тула, Советская ул. – ул. Дзержинского]
CC:2D:E0:F3:8C:6E                [г. Тула, Советская ул. – ул. Мосина]
Name: address_json, dtype: object
```

4.3 Дадим каждому роутеру название улицы

Ввод [47]:
```
# Сначала сгруппируем данные
grouped_data = df2.groupby(['address_json', 'tm'])['user_mac'].count().reset
pivot_table = grouped_data.pivot(index='address_json', columns='tm', values=

# Теперь построим тепловую карту
plt.figure(figsize=(12, 8))
sns.heatmap(pivot_table, cmap='YlGnBu')
plt.title('Нагрузка на router_mac в каждый момент времени')
plt.show()
```



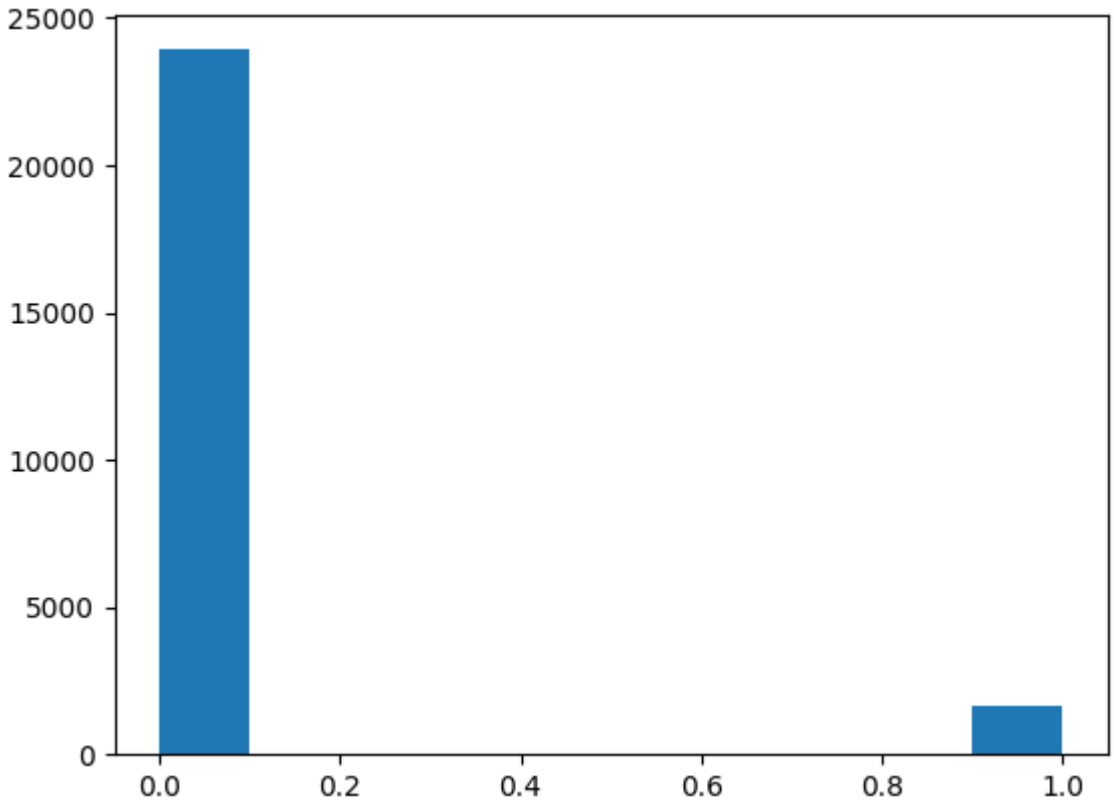
Ввод [48]:
```
rn.head()
```

Out[48]:

	geom	from_vertex_id	to_vertex_id	weight	was_one_way	group_id	coc
0	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	24602	158.805103	False	NaN	[[37 [54 [37
1	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	62722	207.237527	False	NaN	[[37 [54 [37
2	LINESTRING (37.686139 54.200393,37.687477 54.2...	62724	24601	158.519475	False	NaN	[[37 [54 [37
3	LINESTRING (37.599466 54.211929,37.59949 54.21...	129918	129920	7.188256	False	40342.0	[[37 [54 [3
4	LINESTRING (37.590964 54.192815,37.591059 54.1...	262461	262459	16.193512	False	28339.0	[[37 [54 [37

5 Отношение дорог одностороннего движение к двустороннему

Ввод [49]:
```
# Преобразование булевых значений в целочисленный формат
rn['was_one_way_int'] = rn['was_one_way'].astype(int)

# Построение гистограммы
plt.hist(rn['was_one_way_int'])
plt.show()
```



Ввод [50]:
```
rn_oneway = rn[rn['was_one_way']==True]
rn_oneway.head()
```

Out[50]:

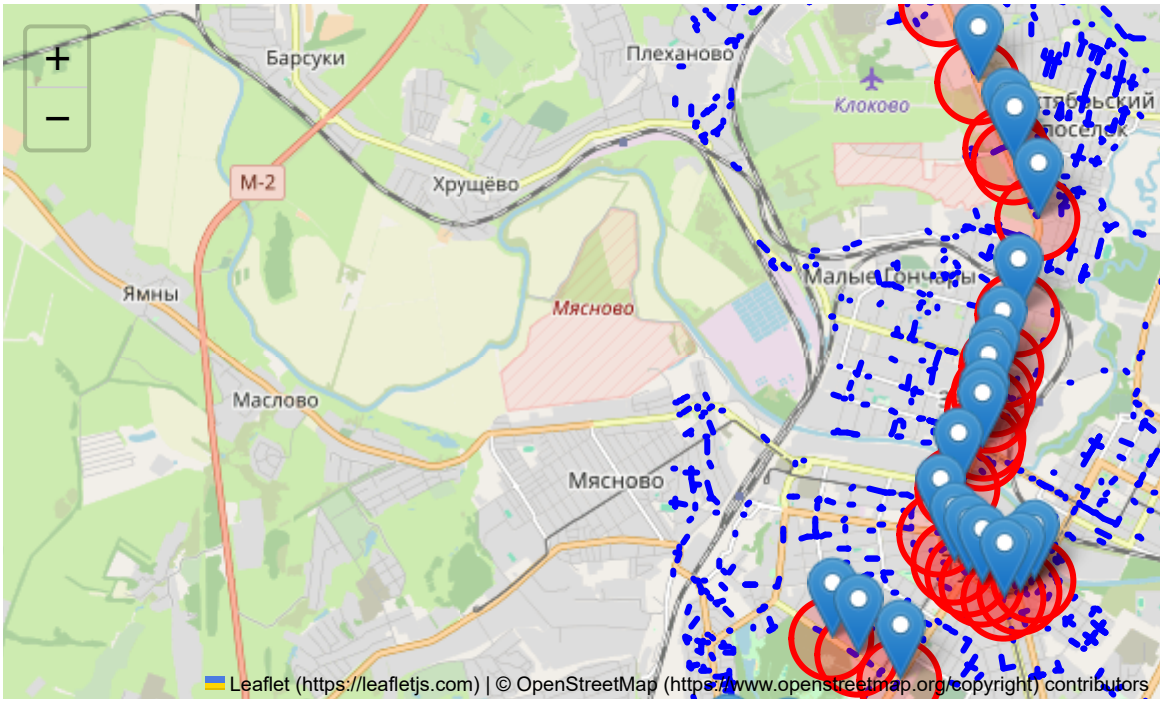
	geom	from_vertex_id	to_vertex_id	weight	was_one_way	group_id	coc
5	LINESTRING (37.606554 54.186595,37.606625 54.1...	232135	232136	50.292525	True	17740.0	[[37 [54 [37
13	LINESTRING (37.586232 54.167799,37.586266 54.1...	174056	174057	53.602831	True	18878.0	[[37 [54 [37
14	LINESTRING (37.616011 54.201864,37.615922 54.2...	187622	187623	31.109309	True	1874.0	[[37 [54 [37
15	LINESTRING (37.665038 54.212276,37.664805 54.2...	165727	165728	23.419885	True	27145.0	[[37 [54 [37
37	LINESTRING (37.616462 54.182934,37.616894 54.1...	8984	8985	99.713604	True	17061.0	[[37 [54 [37

5.1 Инициализация дорог одностороннего движения

Ввод [51]:
```
m = folium.Map(location=[54.19, 37.62], zoom_start=12)

# Добавление линий на карту с прогрессбаром
for i in tqdm(range(len(rn_oneway['flipped_geom'])), desc='Добавление линий'
    #if i % 20 == 0:
        line = rn['flipped_geom'][i]
        coords = [list(map(float, point.split(' '))) for point in line.lstri
        folium.PolyLine(locations=coords, color='blue').add_to(m)
for index, row in wr.iterrows():
    location = [float(coord) for coord in row['clean_geom'].strip('[]').spli
    location.reverse()  # Изменяем порядок координат
    marker = folium.Marker(location=location, popup=f'Location {index}')
    circle = folium.CircleMarker(location=location, radius=20, color='red',
    marker.add_to(m)
    circle.add_to(m)
# Отображение карты
m
m
```

```
Добавление линий: 100%|██████████████████████████████████████████████
█████| 1645/1645 [00:00<00:00, 13595.17it/s]
```

Out[51]:



6 Чтение файла с логами за 2023-03-07 : 2023-03-13

Ввод [52]:
```
df_march = pd.read_csv('march-07.csv')
```

Ввод [53]:
```
df_march.columns
```

Out[53]:
```
Index(['Unnamed: 0', 'guid', 'tm', 'router_mac', 'user_mac', 'signal',
       'router_id'],
      dtype='object')
```

Ввод [54]:
```
df_march = df_march.drop('Unnamed: 0', axis=1)
```

Ввод [55]:
```
df_march['tm'] = pd.to_datetime(df_march['tm'], format='%Y-%m-%d %H:%M:%S.%f
df_march['tm'] = df_march['tm'].dt.round('H')
```

Ввод [56]:
```
df_march['tm'].head()
```

Out[56]:
```
0   2023-03-07 03:00:00+03:00
1   2023-03-07 03:00:00+03:00
2   2023-03-07 03:00:00+03:00
3   2023-03-07 03:00:00+03:00
4   2023-03-07 03:00:00+03:00
Name: tm, dtype: datetime64[ns, UTC+03:00]
```

Ввод [57]:
```
df_march['tm'] = df_march['tm'].apply(lambda x: x.strftime('%Y-%m-%d %H'))
```

7 График загруженности по последнему мартовскому логу

Ввод [58]:
```
# Сначала сгруппируем данные
grouped_data = df_march.groupby(['router_id', 'tm'])['user_mac'].count().res
pivot_table = grouped_data.pivot(index='router_id', columns='tm', values='us
```

7.1 Замена mac на название улицы

Ввод [59]:
```
wr2 = wr.copy()
```

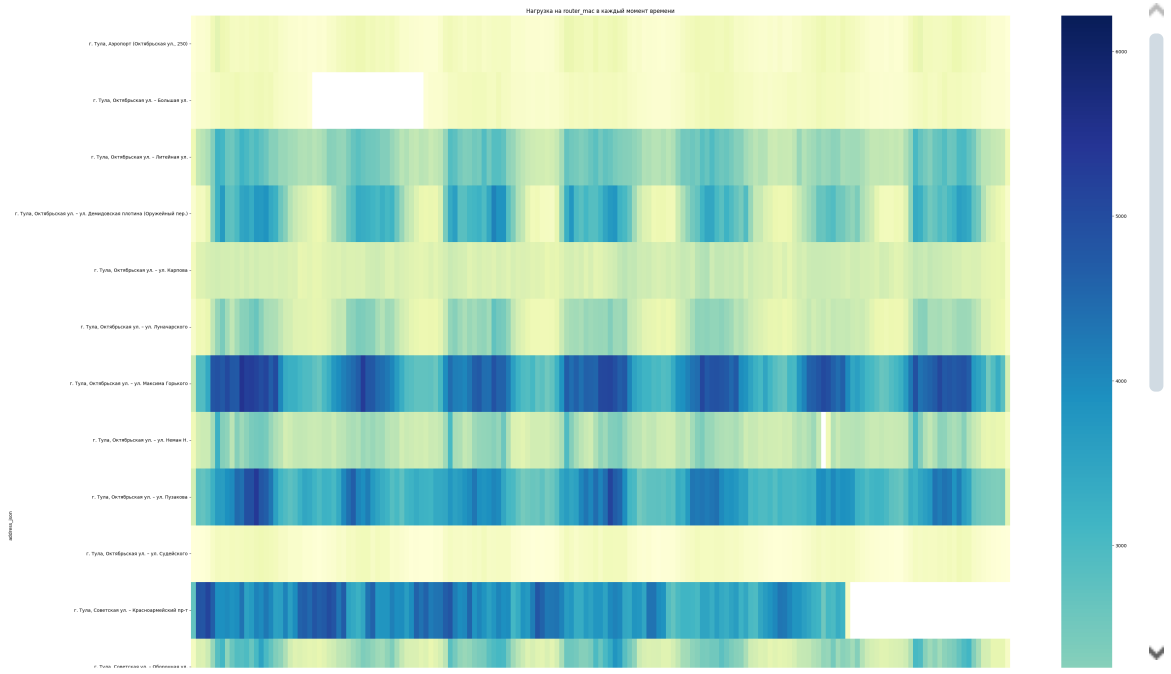
Ввод [60]:
```
wr2.columns = ['router_id', 'geom', 'address_json', 'clean_geom']
```

Ввод [61]:
```
df_name_march = pd.merge(wr2, df_march, on='router_id')
```

Ввод [62]:
```
df_name_march.groupby('router_mac')['address_json'].unique()
```

Out[62]:
```
router_mac
08:55:31:2A:7F:0D              [г. Тула, Октябрьская ул. – ул. Неман Н.]
08:55:31:2A:81:09            [г. Тула, Аэропорт (Октябрьская ул., 250)]
08:55:31:2A:81:1D              [г. Тула, Октябрьская ул. – ул. Карпова]
48:8F:5A:AC:7E:D8              [г. Тула, пр-т Ленина – Первомайская ул.]
48:8F:5A:AC:81:1D              [г. Тула, Октябрьская ул. – ул. Пузакова]
48:8F:5A:AE:A1:27          [г. Тула, Октябрьская ул. – ул. Луначарского]
48:8F:5A:AE:A1:D9    [г. Тула, Октябрьская ул. – ул. Демидовская пл...
48:8F:5A:AE:A2:2A              [г. Тула, Октябрьская ул. – Литейная ул.]
48:8F:5A:B4:91:50            [г. Тула, Октябрьская ул. – ул. Судейского]
B8:69:F4:6B:FD:B6     [г. Тула, Советская ул. – ул. Фридриха Энгельса]
B8:69:F4:6B:FD:E9    [г. Тула, Октябрьская ул. – ул. Максима Горького]
CC:2D:E0:10:F0:32              [г. Тула, Октябрьская ул. – Большая ул.]
CC:2D:E0:82:B8:DD              [г. Тула, Советская ул. – Оборонная ул.]
CC:2D:E0:82:B9:07      [г. Тула, Советская ул. – Красноармейский пр-т]
CC:2D:E0:82:B9:40          [г. Тула, Советская ул. – ул. Дзержинского]
CC:2D:E0:F3:8C:6E                [г. Тула, Советская ул. – ул. Мосина]
CC:2D:E0:F3:8C:EF              [г. Тула, Советская ул. – пр-т Ленина]
CC:2D:E0:F3:8C:FB            [г. Тула, Советская ул. – Тургеневская ул.]
Name: address_json, dtype: object
```

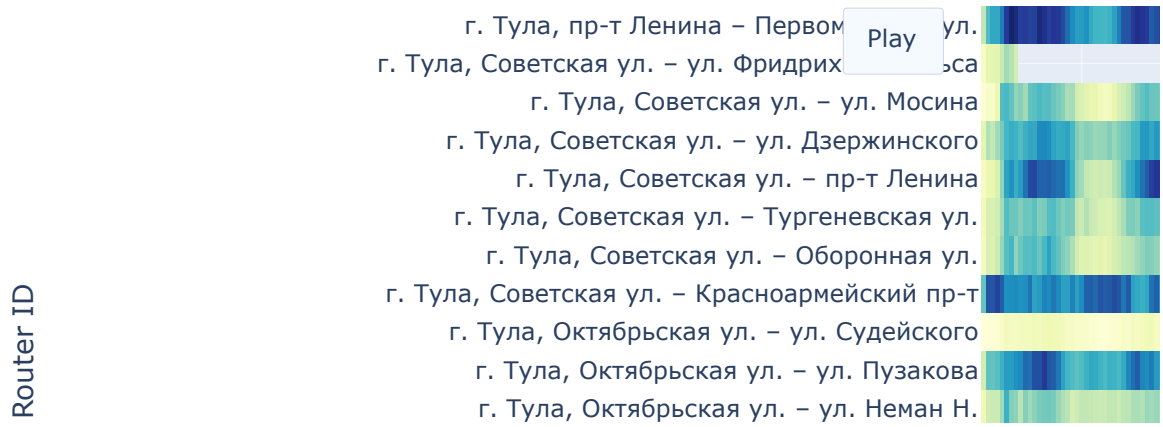
Ввод [63]:
```
# Сначала сгруппируем данные
grouped_data = df_name_march.groupby(['address_json', 'tm'])['user_mac'].cou
pivot_table = grouped_data.pivot(index='address_json', columns='tm', values=

# Теперь построим тепловую карту
plt.figure(figsize=(40, 40))
sns.heatmap(pivot_table, cmap='YlGnBu')
plt.title('Нагрузка на router_mac в каждый момент времени')
plt.show()
```



7.2 heatmap в динамике

Ввод [64]:

```
# Filtering the DataFrame based on the condition df_march['tm'] < 2023.03.08
filtered_df = df_march[df_march['tm'] < '2023-03-08 00']

# Creating a pivot table to get the counts of user_mac for each router_id an
pivot_table = filtered_df.groupby(['router_id', 'tm'])['user_mac'].count().u


grouped_data = df_name_march.groupby(['address_json', 'tm'])['user_mac'].cou
pivot_table = grouped_data.pivot(index='address_json', columns='tm', values=

# Building the heatmap using the pivot table
fig = go.Figure(data=go.Heatmap(
    z=pivot_table.values,
    x=pivot_table.columns,
    y=pivot_table.index,
    colorscale='YlGnBu'))

# Configuring the slider to change the date range
fig.update_layout(
    title='Dynamic Heatmap with a Slider for Date Range Selection',
    xaxis=dict(title='Date'),
    yaxis=dict(title='Router ID'),
    updatemenus=[dict(
        type="buttons",
        buttons=[dict(label="Play",
                      method="animate",
                      args=[None])])])
```



Ввод [65]:
```
pivot_table
```

Out[65]:

tm	2023-03-07 03	2023-03-07 04	2023-03-07 05	2023-03-07 06	2023-03-07 07	2023-03-07 08	2023-03-07 09	2023-03-07 10	2023-03-07 11	201
address_json										
г. Тула, Аэропорт (Октябрьская ул., 250)	77.0	197.0	201.0	284.0	657.0	1008.0	762.0	685.0	489.0	
г. Тула, Октябрьская ул. – Большая ул.	114.0	291.0	273.0	317.0	442.0	506.0	651.0	625.0	654.0	
г. Тула, Октябрьская ул. – Литейная ул.	749.0	1613.0	1732.0	1870.0	2307.0	3172.0	3078.0	2612.0	2545.0	2
г. Тула, Октябрьская ул. – ул. Демидовская	237.0	581.0	555.0	970.0	2280.0	3192.0	3683.0	2806.0	2756.0	3

Ввод [66]:
```
grouped_data
```

Out[66]:

	address_json	tm	user_mac
0	г. Тула, Аэропорт (Октябрьская ул., 250)	2023-03-07 03	77
1	г. Тула, Аэропорт (Октябрьская ул., 250)	2023-03-07 04	197
2	г. Тула, Аэропорт (Октябрьская ул., 250)	2023-03-07 05	201
3	г. Тула, Аэропорт (Октябрьская ул., 250)	2023-03-07 06	284
4	г. Тула, Аэропорт (Октябрьская ул., 250)	2023-03-07 07	657
...	...	...	...
2819	г. Тула, пр-т Ленина – Первомайская ул.	2023-03-13 23	3587
2820	г. Тула, пр-т Ленина – Первомайская ул.	2023-03-14 00	3472
2821	г. Тула, пр-т Ленина – Первомайская ул.	2023-03-14 01	3470
2822	г. Тула, пр-т Ленина – Первомайская ул.	2023-03-14 02	3433
2823	г. Тула, пр-т Ленина – Первомайская ул.	2023-03-14 03	1697

7.3 Изменения дорожно-транспортной ситуации с течением времени на основе перемещений между роутерами

Ввод [67]:
```
fig = px.histogram(grouped_data,
                   x='address_json',
                   y= 'user_mac',
                   range_x=[0, 15],
                   animation_frame='tm',
                   range_y=[0, 6000],
                   color='tm')

fig.update_layout(title="Изменения дорожно-транспортной ситуации с течением
```

Ввод [68]:
```
df_march.head()
```

Out[68]:

	guid	tm	router_mac	user_mac	signal	router_id
0	9cee7337-103a-4e3e-9d6a-d4fa2ba9fd90	2023-03-07 03	CC:2D:E0:82:B9:07	80:2A:A8:76:08:1C	-74.0	8a823f01-771f-420e-8971-928cba314a24
1	f77d6f20-fac1-4fd3-879c-454b52bf92f2	2023-03-07 03	CC:2D:E0:82:B9:07	56:36:9B:28:28:F8	-63.0	8a823f01-771f-420e-8971-928cba314a24
2	281ae006-577a-4e7d-afb7-6b88f7f7f889	2023-03-07 03	CC:2D:E0:82:B9:07	98:54:1B:8A:20:A5	-74.0	8a823f01-771f-420e-8971-928cba314a24
3	0059a476-d01f-4e55-a78a-b2ba9c7389c3	2023-03-07 03	CC:2D:E0:82:B9:07	0C:0E:76:D3:6D:F8	-65.0	8a823f01-771f-420e-8971-928cba314a24
4	6e6122d4-34e9-4360-9e63-0d55778d8faf	2023-03-07 03	CC:2D:E0:82:B9:07	0C:D9:96:4D:80:D3	-74.0	8a823f01-771f-420e-8971-928cba314a24

Ввод [69]:
```
df_march.groupby(['router_mac','user_mac'])['signal'].count()
```

Out[69]:
```
router_mac         user_mac
08:55:31:2A:7F:0D  00:00:00:8D:08:2B     1
                   00:00:21:14:32:34     1
                   00:00:73:27:F3:01     1
                   00:02:5B:1E:00:1B     1
                   00:02:5B:D6:10:07    12
                                        ..
CC:2D:E0:F3:8C:FB  FE:FF:FF:53:A8:90     1
                   FE:FF:FF:FF:DC:2A     1
                   FF:6B:48:CF:09:15     1
                   FF:E3:90:85:A3:ED     1
                   FF:FF:FF:FF:FF:FF     2
Name: signal, Length: 2369569, dtype: int64
```

Ввод [70]:
```
wr3 = wr.copy()
```

Ввод [71]:
```
wr3.columns = ['router_id', 'geom', 'address_json', 'clean_geom']
```

Ввод [72]:
```
df_name_march3 = pd.merge(wr3, df_march, on='router_id')
```

Ввод [73]:
```
df_name_march3.groupby('router_mac')['address_json'].unique()
```

Out[73]:
```
router_mac
08:55:31:2A:7F:0D            [г. Тула, Октябрьская ул. – ул. Неман Н.]
08:55:31:2A:81:09           [г. Тула, Аэропорт (Октябрьская ул., 250)]
08:55:31:2A:81:1D             [г. Тула, Октябрьская ул. – ул. Карпова]
48:8F:5A:AC:7E:D8            [г. Тула, пр-т Ленина – Первомайская ул.]
48:8F:5A:AC:81:1D            [г. Тула, Октябрьская ул. – ул. Пузакова]
48:8F:5A:AE:A1:27        [г. Тула, Октябрьская ул. – ул. Луначарского]
48:8F:5A:AE:A1:D9    [г. Тула, Октябрьская ул. – ул. Демидовская пл...
48:8F:5A:AE:A2:2A            [г. Тула, Октябрьская ул. – Литейная ул.]
48:8F:5A:B4:91:50          [г. Тула, Октябрьская ул. – ул. Судейского]
B8:69:F4:6B:FD:B6     [г. Тула, Советская ул. – ул. Фридриха Энгельса]
B8:69:F4:6B:FD:E9    [г. Тула, Октябрьская ул. – ул. Максима Горького]
CC:2D:E0:10:F0:32             [г. Тула, Октябрьская ул. – Большая ул.]
CC:2D:E0:82:B8:DD             [г. Тула, Советская ул. – Оборонная ул.]
CC:2D:E0:82:B9:07      [г. Тула, Советская ул. – Красноармейский пр-т]
CC:2D:E0:82:B9:40          [г. Тула, Советская ул. – ул. Дзержинского]
CC:2D:E0:F3:8C:6E                [г. Тула, Советская ул. – ул. Мосина]
CC:2D:E0:F3:8C:EF               [г. Тула, Советская ул. – пр-т Ленина]
CC:2D:E0:F3:8C:FB          [г. Тула, Советская ул. – Тургеневская ул.]
Name: address_json, dtype: object
```

Ввод [74]:
```
df_name_march3
```

Out[74]:

	router_id	geom	address_json	clean_geom	guid	tm	router_m
0	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	0390ae84-f7ef-4675-a6e2-189836f44af1	2023-03-07 03	48:8F:5A:
1	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	3c65597e-9d06-4a3f-bbb3-da71de9c0c3f	2023-03-07 03	48:8F:5A:
2	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	851d14f3-11ed-47f0-8b91-824ccb233629	2023-03-07 03	48:8F:5A:
3	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	93f20c99-b29b-4525-82df-93d69c45dc53	2023-03-07 03	48:8F:5A:
4	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	9671c147-ef4d-4d7f-8c40-3d752a957e1c	2023-03-07 03	48:8F:5A:
...	...	...	...	...	...	...	
6177220	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	8945b49c-8089-496b-a22f-988f82772624	2023-03-14 03	48:8F:5A
6177221	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	fa2b6d88-a1d4-4254-95fa-7c80f13aeecd	2023-03-14 03	48:8F:5A
6177222	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	cd22c223-0932-4bdb-9ed7-8fd16082f72a	2023-03-14 03	48:8F:5A
6177223	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	0e8c30f8-eb97-447f-a7b4-bbae733ebbe2	2023-03-14 03	48:8F:5A
6177224	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	f124ecb1-a640-478b-821d-5a0c7f116452	2023-03-14 03	48:8F:5A

6177225 rows × 9 columns

7.4 Разбивка загруженности по категориальным данным

Ввод [75]:
```
time_categories = pd.cut(pd.to_datetime(df_name_march3['tm']).dt.hour,
                         bins=[7, 10, 16, 19],
                         labels=['утренний_час_пик', 'день', 'вечерний_час_п


df_name_march3['time_category'] = time_categories

# Создаем матрицу спроса по категориям времени
demand_matrix = df_name_march3.pivot_table(index='address_json', columns='ti
```

Ввод [76]:
```
demand_matrix
```

Out[76]:

time_category address_json	утренний_час_пик	день	вечерний_час_пик
г. Тула, Аэропорт (Октябрьская ул., 250)	13461	25694	12933
г. Тула, Октябрьская ул. – Большая ул.	9426	21789	10332
г. Тула, Октябрьская ул. – Литейная ул.	52742	107823	52528
г. Тула, Октябрьская ул. – ул. Демидовская плотина (Оружейный пер.)	56953	126779	68998
г. Тула, Октябрьская ул. – ул. Карпова	29163	57762	28261
г. Тула, Октябрьская ул. – ул. Луначарского	39472	84908	42452
г. Тула, Октябрьская ул. – ул. Максима Горького	91620	200123	97150
г. Тула, Октябрьская ул. – ул. Неман Н.	44416	79268	43561
г. Тула, Октябрьская ул. – ул. Пузакова	74129	175022	83850
г. Тула, Октябрьская ул. – ул. Судейского	8778	22134	11825
г. Тула, Советская ул. – Красноармейский пр-т	69615	130210	61740
г. Тула, Советская ул. – Оборонная ул.	47197	99964	57603
г. Тула, Советская ул. – Тургеневская ул.	49665	107839	56660
г. Тула, Советская ул. – пр-т Ленина	68867	182258	102163
г. Тула, Советская ул. – ул. Дзержинского	57866	131628	71375
г. Тула, Советская ул. – ул. Мосина	46894	98000	53772
г. Тула, Советская ул. – ул. Фридриха Энгельса	5037	0	0
г. Тула, пр-т Ленина – Первомайская ул.	104518	218518	107363

Ввод [77]:
```
time_categories = pd.cut(pd.to_datetime(df_name_march3['tm']).dt.hour,
                         bins=[7, 10, 16, 19],
                         labels=['утренний_час_пик', 'день', 'вечерний_час_п

# Добавляем категории дней
day_categories = pd.cut(pd.to_datetime(df_name_march3['tm']).dt.dayofweek,
                        bins=[0, 4, 6],
                        labels=['рабочий_день', 'выходной'])

# Создаем новую колонку для категорий дней
df_name_march3['day_category'] = day_categories

# Создаем матрицу спроса по категориям времени и дня
demand_matrix = df_name_march3.pivot_table(index='address_json', columns=['d
```

Ввод [78]:
```
demand_matrix.reset_index()
```

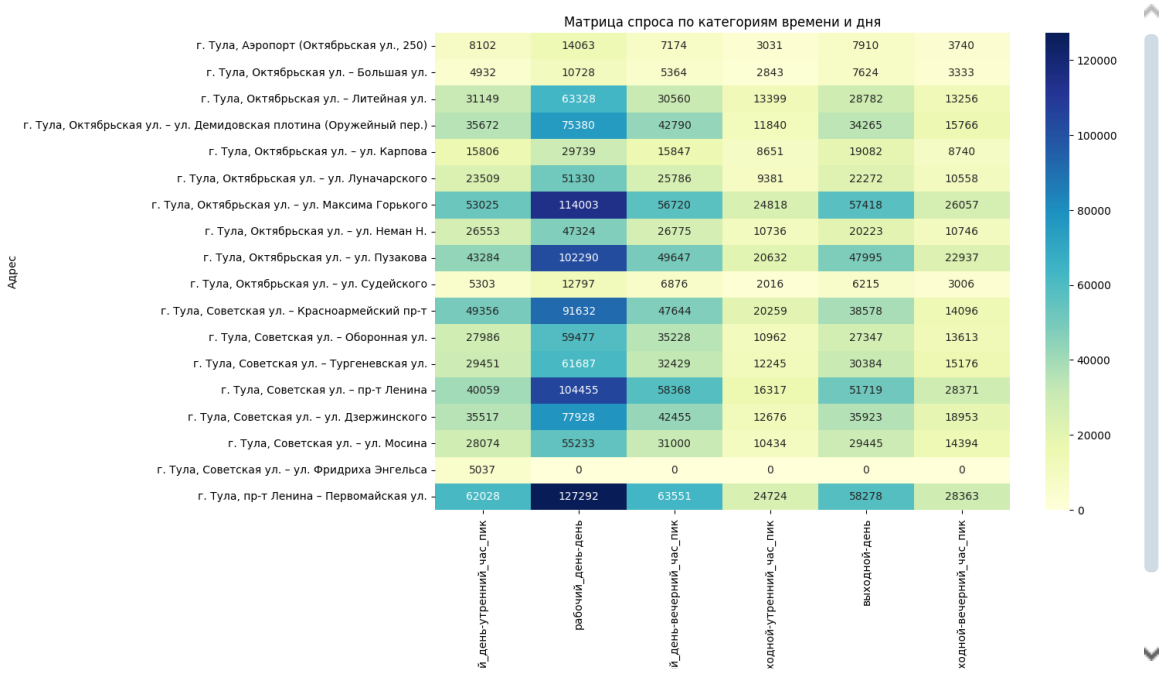
Out[78]:

day_category	address_json	рабочий_день			выходной
time_category		утренний_час_пик	день	вечерний_час_пик	утренний_
0	г. Тула, Аэропорт (Октябрьская ул., 250)	8102	14063	7174	
1	г. Тула, Октябрьская ул. – Большая ул.	4932	10728	5364	
2	г. Тула, Октябрьская ул. – Литейная ул.	31149	63328	30560	
3	г. Тула, Октябрьская ул. – ул. Демидовская пло...	35672	75380	42790	

Ввод [79]:
```
demand_matrix.columns
```

Out[79]:
```
MultiIndex([('рабочий_день', 'утренний_час_пик'),
            ('рабочий_день',             'день'),
            ('рабочий_день', 'вечерний_час_пик'),
            (    'выходной', 'утренний_час_пик'),
            (    'выходной',             'день'),
            (    'выходной', 'вечерний_час_пик')],
           names=['day_category', 'time_category'])
```

Ввод [80]:
```
# Создаем тепловую карту для матрицы спроса
plt.figure(figsize=(12, 8))
sns.heatmap(demand_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.title('Матрица спроса по категориям времени и дня')
plt.xlabel('Категория времени и дня')
plt.ylabel('Адрес')
plt.show()
```



Ввод [81]:
```
df1.head()
```

Out[81]:

	guid	tm	router_mac	user_mac	signal	router_id
0	1a25652b-f346-4ffb-aec3-295ecf08fd97	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B8:DD	0E:AC:4A:34:2A:F1	-68.0	cdcab165-cc3a-460f-ae99-024216190ab1
1	f250e9af-498e-42e7-9439-7a82473b829a	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B9:07	56:36:9B:28:28:F8	-67.0	8a823f01-771f-420e-8971-928cba314a24
2	525c9f06-fc70-426e-81b3-ed96497b6f01	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B9:07	E0:63:DA:DC:D8:49	-61.0	8a823f01-771f-420e-8971-928cba314a24
3	cf66e62d-a621-4aed-a46a-b6bc135ce766	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B9:07	1C:15:1F:CA:88:EE	-70.0	8a823f01-771f-420e-8971-928cba314a24
4	d3554553-5555-4cfd-8d01-3f56fea0a589	2022-12-01 03:00:00+03:00	CC:2D:E0:82:B9:40	D0:37:45:C1:E1:1E	-66.0	3d9490f4-fb08-4c4c-80ec-ca744a6d9ab4

Ввод [82]:
```
df1.groupby(['router_mac','user_mac'])['signal'].count()
```

Out[82]:
```
router_mac         user_mac
48:8F:5A:AC:81:1D  00:00:FA:17:DB:FA    4
                   00:08:22:0A:D8:2C    1
                   00:08:22:16:29:F7    1
                   00:08:22:20:E9:FB    5
                   00:08:22:2A:4B:A7    1
                                       ..
CC:2D:E0:F3:8C:6E  FE:FF:0C:68:6E:6E    1
                   FE:FF:3A:1D:85:DE    2
                   FE:FF:60:31:2E:63    1
                   FE:FF:E7:EA:19:B6    1
                   FF:D4:DE:BA:72:46    1
Name: signal, Length: 192140, dtype: int64
```

7.5 Проверка среднего времени пребывания на роутерах

Ввод [83]:
```
df1[(df1['router_mac'] == '48:8F:5A:AC:81:1D') & (df1['user_mac'] == '00:00:
```

Out[83]:

	guid	tm	router_mac	user_mac	signal	router_id
194951	aa0a778e-fc12-4918-b460-fb7d2bc83e70	2022-12-01 13:00:00+03:00	48:8F:5A:AC:81:1D	00:00:FA:17:DB:FA	-72.0	8a7346 e451-4 8b 3b3b46e33
236598	af6bbed6-f7f4-4b2b-8bfb-03510d7402a5	2022-12-01 15:00:00+03:00	48:8F:5A:AC:81:1D	00:00:FA:17:DB:FA	-68.0	8a7346 e451-4 8b 3b3b46e33
283852	c4ee8dc8-40b2-48f7-b360-7d312327ad8d	2022-12-01 17:00:00+03:00	48:8F:5A:AC:81:1D	00:00:FA:17:DB:FA	-73.0	8a7346 e451-4 8b 3b3b46e33
311862	41464573-16fa-44a8-99f0-b423dd32a43b	2022-12-01 18:00:00+03:00	48:8F:5A:AC:81:1D	00:00:FA:17:DB:FA	-75.0	8a7346 e451-4 8b 3b3b46e33

Ввод [84]:
```
df1.groupby(['router_mac', 'user_mac', 'tm'])['signal'].mean()
```

Out[84]:
```
router_mac         user_mac           tm
48:8F:5A:AC:81:1D  00:00:FA:17:DB:FA  2022-12-01 13:00:00+03:00   -72.0
                                      2022-12-01 15:00:00+03:00   -68.0
                                      2022-12-01 17:00:00+03:00   -73.0
                                      2022-12-01 18:00:00+03:00   -75.0
                   00:08:22:0A:D8:2C  2022-12-01 12:00:00+03:00   -67.0
                                                                   ...
CC:2D:E0:F3:8C:6E  FE:FF:3A:1D:85:DE  2022-12-01 09:00:00+03:00   -70.0
                                      2022-12-01 10:00:00+03:00   -66.0
                   FE:FF:60:31:2E:63  2022-12-01 06:00:00+03:00   -65.0
                   FE:FF:E7:EA:19:B6  2022-12-01 11:00:00+03:00   -71.0
                   FF:D4:DE:BA:72:46  2022-12-01 12:00:00+03:00   -74.0
Name: signal, Length: 202164, dtype: float64
```

Ввод [85]:
```
t_mean = df1.groupby(['router_mac', 'user_mac','router_id'])['tm'].agg(lambd
t_mean
```

Out[85]:
```
router_mac         user_mac           router_id
48:8F:5A:AC:81:1D  00:00:FA:17:DB:FA  8a734674-e451-4fe1-8b50-3b3b46e334a1
0 days 05:00:00
                   00:08:22:0A:D8:2C  8a734674-e451-4fe1-8b50-3b3b46e334a1
0 days 00:00:00
                   00:08:22:16:29:F7  8a734674-e451-4fe1-8b50-3b3b46e334a1
0 days 00:00:00
                   00:08:22:20:E9:FB  8a734674-e451-4fe1-8b50-3b3b46e334a1
0 days 11:00:00
                   00:08:22:2A:4B:A7  8a734674-e451-4fe1-8b50-3b3b46e334a1
0 days 00:00:00

...
CC:2D:E0:F3:8C:6E  FE:FF:0C:68:6E:6E  0b91ef8c-3b4f-4a39-b3a3-9683c6fd7832
0 days 00:00:00
                   FE:FF:3A:1D:85:DE  0b91ef8c-3b4f-4a39-b3a3-9683c6fd7832
0 days 01:00:00
                   FE:FF:60:31:2E:63  0b91ef8c-3b4f-4a39-b3a3-9683c6fd7832
0 days 00:00:00
                   FE:FF:E7:EA:19:B6  0b91ef8c-3b4f-4a39-b3a3-9683c6fd7832
0 days 00:00:00
                   FF:D4:DE:BA:72:46  0b91ef8c-3b4f-4a39-b3a3-9683c6fd7832
0 days 00:00:00
Name: tm, Length: 192140, dtype: timedelta64[ns]
```

Ввод [86]:
```
t_mean = t_mean.reset_index()
```

Ввод [87]:
```
t_mean.columns
```

Out[87]:
```
Index(['router_mac', 'user_mac', 'router_id', 'tm'], dtype='object')
```

Ввод [88]:
```
t_mean.head()
```

Out[88]:

	router_mac	user_mac	router_id	tm
0	48:8F:5A:AC:81:1D	00:00:FA:17:DB:FA	8a734674-e451-4fe1-8b50-3b3b46e334a1	0 days 05:00:00
1	48:8F:5A:AC:81:1D	00:08:22:0A:D8:2C	8a734674-e451-4fe1-8b50-3b3b46e334a1	0 days 00:00:00
2	48:8F:5A:AC:81:1D	00:08:22:16:29:F7	8a734674-e451-4fe1-8b50-3b3b46e334a1	0 days 00:00:00
3	48:8F:5A:AC:81:1D	00:08:22:20:E9:FB	8a734674-e451-4fe1-8b50-3b3b46e334a1	0 days 11:00:00
4	48:8F:5A:AC:81:1D	00:08:22:2A:4B:A7	8a734674-e451-4fe1-8b50-3b3b46e334a1	0 days 00:00:00

Ввод [89]:
```
t_mean['tm'] = pd.to_timedelta(t_mean['tm'], errors='coerce')
```

Ввод [90]:
```
t_mean['tm'] = t_mean['tm'].apply(lambda x: x.total_seconds() / 3600 if not
```

Ввод [91]:
```
t_mean['tm']
```

Out[91]:
```
0          5.0
1          0.0
2          0.0
3         11.0
4          0.0
          ...
192135     0.0
192136     1.0
192137     0.0
192138     0.0
192139     0.0
Name: tm, Length: 192140, dtype: float64
```

```
t_mean.plot(x='user_mac', y='tm', kind='line', figsize=(30, 30))  # Adjust
the figsize parameter here

plt.xlabel('User MAC')
plt.ylabel('Time (hours)')
plt.title('Time Spent by User MAC')

plt.show()
```

```
average_time_per_router = t_mean.groupby('router_mac')['tm'].mean()

# Создаем линейный график
plt.figure(figsize=(12, 6))
average_time_per_router.plot(kind='line', marker='o')
plt.xlabel('Router MAC')
plt.ylabel('Average Time (hours)')
plt.title('Average Time Spent on Each Router')

plt.show()
```

Ввод [92]:
```
t_mean_name = pd.merge(wr3, t_mean, on='router_id')
t_mean_name.groupby('router_mac')['address_json'].unique()
```

Out[92]:
```
router_mac
48:8F:5A:AC:81:1D            [г. Тула, Октябрьская ул. – ул. Пузакова]
48:8F:5A:AE:A1:27        [г. Тула, Октябрьская ул. – ул. Луначарского]
48:8F:5A:AE:A1:D9    [г. Тула, Октябрьская ул. – ул. Демидовская пл...
B8:69:F4:6B:FD:E9    [г. Тула, Октябрьская ул. – ул. Максима Горького]
CC:2D:E0:10:F0:32               [г. Тула, ул. Щегловская Засека, д. 5]
CC:2D:E0:82:B8:DD             [г. Тула, Советская ул. – Оборонная ул.]
CC:2D:E0:82:B9:07      [г. Тула, Советская ул. – Красноармейский пр-т]
CC:2D:E0:82:B9:40          [г. Тула, Советская ул. – ул. Дзержинского]
CC:2D:E0:F3:8C:6E                [г. Тула, Советская ул. – ул. Мосина]
Name: address_json, dtype: object
```

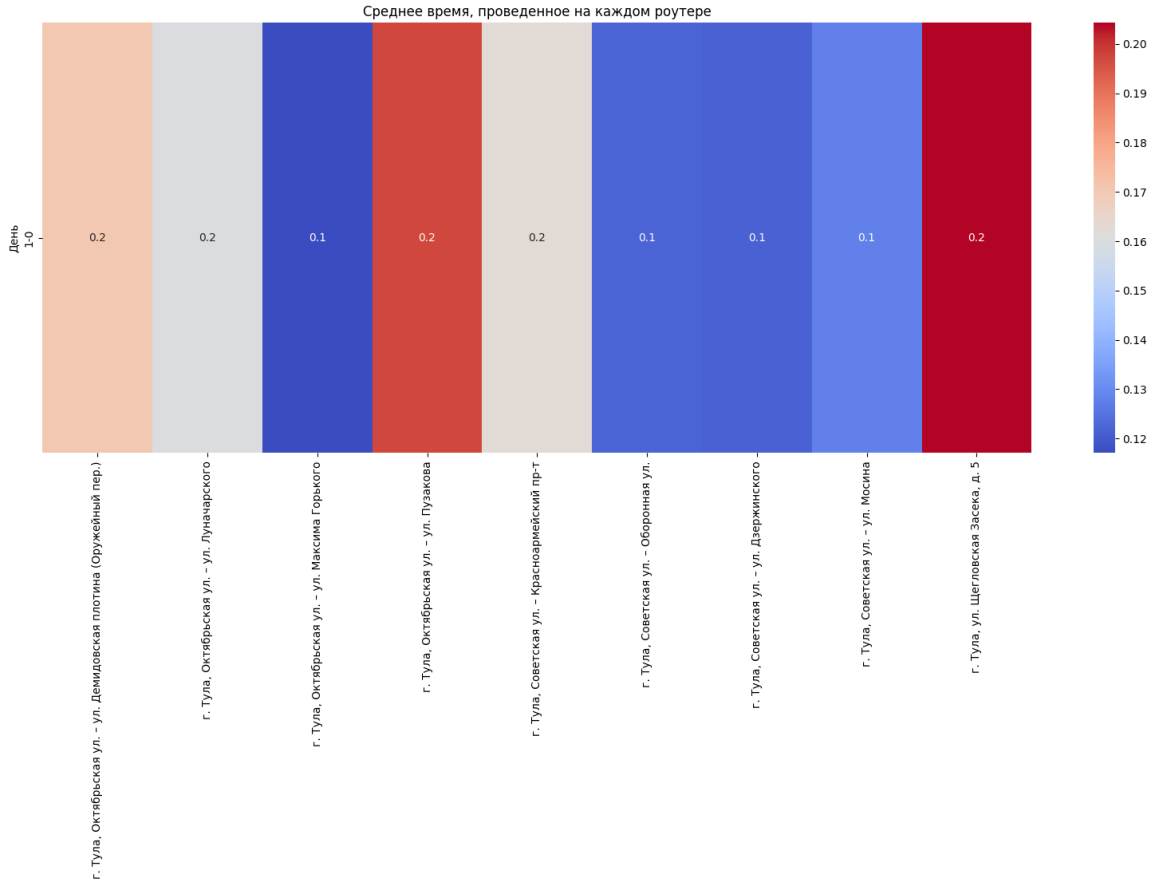
Ввод [93]:
```
t_mean_name
```

Out[93]:

	router_id	geom	address_json	clean_geom	router_mac	user_mac
0	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	48:8F:5A:AE:A1:D9	00:00:73:27
1	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	48:8F:5A:AE:A1:D9	00:01:12:42
2	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	48:8F:5A:AE:A1:D9	00:05:B5:AE:(
3	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	48:8F:5A:AE:A1:D9	00:07:22:C2:
4	0648078a-9d45-4577-af14-12b49e8f017b	POINT (37.618886 54.204617)	г. Тула, Октябрьская ул. – ул. Демидовская пло...	37.618886 54.204617	48:8F:5A:AE:A1:D9	00:08:22:00:
...	...	...	...	...	...	
192135	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	48:8F:5A:AE:A1:27	FE:FC:BA:18:
192136	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	48:8F:5A:AE:A1:27	FE:FD:CF:1E:I
192137	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	48:8F:5A:AE:A1:27	FE:FF:3A:1D:
192138	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	48:8F:5A:AE:A1:27	FE:FF:B6:63:
192139	a8c8525e-cfcf-416f-88dd-bf1efc79df00	POINT (37.619182 54.205679)	г. Тула, Октябрьская ул. – ул. Луначарского	37.619182 54.205679	48:8F:5A:AE:A1:27	FE:FF:C8:73:

192140 rows × 7 columns

Ввод [94]:
```
t_mean.index = pd.to_datetime(t_mean.index)

# Создаем сводную таблицу для среднего времени, проведенного всеми пользовате
average_time_per_router = t_mean_name.groupby([t_mean.index.day, t_mean.inde

# Создаем тепловую карту
plt.figure(figsize=(20, 7))
sns.heatmap(average_time_per_router, cmap='coolwarm', annot=True, fmt=".1f")
plt.title('Среднее время, проведенное на каждом роутере')
plt.xlabel(' ')

plt.ylabel('День')

plt.show()
```



Задачи:

1. Провести разведочный анализ данных (EDA - Exploratory data analysis). Проанализировать изменения дорожно-транспортной ситуации с течением времени на основе перемещений между роутерами. Построить временную шкалу по неделям
2. Составить матрицу перемещений/спроса с расчётом среднего времени поездки по трём видам интервалов: утренний час пик, день, вечерний час пик.
3. Визуализировать результат на диаграммах/картограммах
4. Предоставить замечания/комментарии, если такие появятся, к расположение WiFi роутеров для качественного улучшения охвата города

8 Итоги

8.1 Провести разведочный анализ данных (EDA - Exploratory data analysis). Проанализировать изменения дорожно-транспортной ситуации с течением времени на основе перемещений между роутерами. Построить временную шкалу по неделям

- Проведен разведочный анализ данных всех файлов, которые были предоставлены.

В процессе EDA определено,что файл wifi_routers содержит 3 столбца и 29 строк. Все три столбца содержат категориальные данные, определяющие место положение роутера, а строки - отдельный роутер, расположенный в городе Тула. Пропуски в данных отсутствуют.

Файл road_network содержит данные о дорожной сети - графы и узлы. Пропуски имеются в столбце group_id. Данный столбец несет служебную информацию и не имеет значения для последующего анализа. Также содержится столбец, несущий в себе информацию, является ли улица односторонней.

Также для анализа представлены данные с каждого роутера по дням в течение одного года - с 1 декабря 2022 года по 30 ноября 2023 года. Нами был взят за основу и изучен один период времени - с 7 по 14 марта 2023 года.

Данные за неделю были объединены в один файл, содержащий 7 столбцов и 6 177 225 строк. В файле содержится информация о том, какие и сколько машин проехали через тот или иной перекресток. Пропуски отсутствуют, в файл попало 18 роутеров из 29, представленных в первом файле.

Временную шкалу по неделям построить не удалось, так как изучался всего один период - одна неделя в году. Была построена временная шкала перемещений между роутерами за каждый час выше указанной недели.

8.2 Составить матрицу перемещений/спроса с расчётом среднего времени поездки по трём видам интервалов: утренний час пик, день, вечерний час пик

Матрица перемещений составлена без расчета среднего времени поездки. При попадании в финал у нас будет стимул построить данную матрицу)

Данные по спросу с тремя видами интервалов представлены выше. Самыми загруженными перекрестками оказались Ленина - Первомайская, Октябрьская - Максима Горького, Октябрьская - Пузакова. Эти перекрестки загружены в любое время суток.

На перекрестках Советская - Энгельса, Советская - Красноармейский проспект и Октябрьская - Большая часть информации по передвижениям отсутствует. Мы предполагаем, что данное обстоятельство связано с техническими сбоями роутеров.

8.3 Предоставить замечания/комментарии, если такие появятся, к расположение WiFi роутеров для качественного улучшения охвата города

Город Тула является старинным городом, которому в этом году исполнилось 877 лет. При этом население составляет всего полмиллиона человек. Роутеры, представленные в третьем файле и анализируемые нами, расположены в центре города - в Советском и Центральном районах, в основе своей - на центральных улицах, а с учетом возраста города - довольно узких, что объясняет большую загрузку в дневное время суток.

В связи с выше изложенным, матрица перемещений значительно отличается от матрицы, которая была бы при анализе данных в спальных районах.

Для качественного улучшения охвата города, рекомендую установить роутеры на таких перекрестках, как Кутузова - Кирова, Металлургов - Ложевая, Некрасова - Перекопская, Некрасова - Оборонная, Одоевское шоссе - Тихмянова, Ликбеза - Чмутова, Дульная - Максима Горького и иные выезды из спальных районов.

Ввод [ ]: