Исследование рынка заведений общественного питания Москвы

Цель проекта:

- найти интересные особенности рынка заведений общественного питания Москвы, которые помогут в выборе подходящего инвесторам места открытия нового заведения.

Задачи проекта:

- подготовить исследование рынка Москвы
- подготовить структурированную, информативную и лаконичную презентацию на основе исследования.

План выполнения проекта.

1. Загрузка данных и изучение общей информации
    - Импорт библиотек
    - загрузка данных о заведениях общественного питания Москвы. Путь к файлу: /datasets/moscow_places.csv
    - Изучить общую информацию о датасете. Сколько заведений представлено? Что можно сказать о каждом столбце? Значения какого типа они хранят?
    - описание данных
2. Предобработка данных
    - поиск дубликатов в данных
    - поиск пропусков: встречаются ли они, в каких столбцах? Можно ли их обработать или оставить как есть?
    - Выполнить предобработку данных:
    - Добавить столбец 'street' с названиями улиц из столбца с адресом.
    - Добавить столбец 'is_24_7' с обозначением, что заведение работает ежедневно и круглосуточно (24/7):
        - логическое значение True — если заведение работает ежедневно и круглосуточно;
        - логическое значение False — в противоположном случае.
3. Анализ данных
    - Какие категории заведений представлены в данных? Исследовать количество объектов общественного питания по категориям: рестораны, кофейни, пиццерии, бары и так далее. Построить визуализации. Ответить на вопрос о распределении заведений по категориям.
    - Исследовать количество посадочных мест в местах по категориям: рестораны, кофейни, пиццерии, бары и так далее. Построить визуализации. Проанализировать результаты и сделать выводы.
    - Рассмотреть и изобразить соотношение сетевых и несетевых заведений в датасете. Каких заведений больше?
    - Какие категории заведений чаще являются сетевыми? Исследовать данные и ответить на вопрос графиком.
    - Сгруппировать данные по названиям заведений и найти топ-15 популярных сетей в Москве. Под популярностью понимается количество заведений этой сети в регионе. Построить подходящую для такой информации визуализацию. Знакомы ли нам эти сети? Есть ли какой-то признак, который их объединяет? К какой категории заведений они относятся?
    - Какие административные районы Москвы присутствуют в датасете? Отобразить общее количество заведений и количество заведений каждой категории по районам. Попробовать проиллюстрировать эту информацию одним графиком.
    - Визуализировать распределение средних рейтингов по категориям заведений. Сильно ли различаются усреднённые рейтинги в разных типах общепита?
    - Построить фоновую картограмму (хороплет) со средним рейтингом заведений каждого района. Границы районов Москвы, которые встречаются в датасете, хранятся в файле. Путь к файлу'admin_level_geomap.geojson', внешняя ссылка.
    - Отобразить все заведения датасета на карте с помощью кластеров средствами библиотеки folium.

    - Найти топ-15 улиц по количеству заведений. Построить график распределения количества заведений и их категорий по этим улицам. Попробовать проиллюстрировать эту информацию одним графиком.
    - Найти улицы, на которых находится только один объект общепита. Что можно сказать об этих заведениях?
    - Посчитать медиану столбца middle_avg_bill для каждого района. В этом столбце хранятся значения средних чеков заведений х. Эти числа показывают примерную стоимость заказа в рублях, которая чаще всего выражена диапазоном. Использовать это значение в качестве ценового индикатора района.
        - Построить фоновую картограмму (хороплет) с полученными значениями для каждого района.
        - Проанализируйте цены в центральном административном округе и других. Как удалённость от центра влияет на цены в заведениях?
4. Собрать наблюдения по анализу данных в один общий вывод.
5. Детализация исследования
    - Сколько всего кофеен в датасете? В каких районах их больше всего, каковы особенности их расположения?
    - Есть ли круглосуточные кофейни?
    - Какие у кофеен рейтинги? Как они распределяются по районам?
    - На какую стоимость чашки капучино стоит ориентироваться при открытии и почему?
    - расширить список вопросов для исследования исходя из бизнеса
    - Построить визуализации.
    - Дать рекомендацию для открытия нового заведения. Решение должно быть чем-то обосновано: текстом с описанием или маркерами на географической карте.

1 Загрузка данных и изучение общей информации

1.1 Импорт библиотек

In [1]:

```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
from plotly import graph_objects as go
import plotly.express as px



#from folium.plugins import MarkerCluster # импортируем кластер
from folium import Map, Choropleth # импортируем карту, хороплет
from folium import Marker, Map
from folium.plugins import MarkerCluster # импортируем кластер
from folium.features import CustomIcon # для кастомизации иконок на карте

import re # Импортируем модуль «re»
#from re import search #

# снимаем ограничение на количество столбцов
pd.set_option('display.max_columns', None)

# снимаем ограничение на ширину столбцов
pd.set_option('display.max_colwidth', None)

# игнорируем предупреждения
pd.set_option('chained_assignment', None)
```

1.2 Загрузка данных

Датасет содержит информацию о заведениях общественного питания Москвы, составленный на основе данных сервисов Яндекс Карты и Яндекс Бизнес на лето 2022 года.

Информация, размещённая в сервисе Яндекс Бизнес, могла быть добавлена пользователями или найдена в общедоступных источниках.

Информация носит исключительно справочный характер.

Загрузим датасет '/datasets/moscow_places.csv' и сохраним его в переменную 'df'

In [2]:

```
try:
    df = pd.read_csv('/datasets/moscow_places.csv', sep=',')
except:
    df = pd.read_csv('https://code.s3.yandex.net/datasets/moscow_places.csv', sep=',')
```

1.3 Общая информация о датасете

Изучим общую информацию о датасете. Для начала убедимся, что данные загружены, а также посмотрим что именно находится в датасете. Для этого посмотрим на несколько случайных строк в загруженном датасете 'df'.

In [3]:

```
display(df.sample(n=5, random_state=2))
```

	name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_avg_bill	middle_coffee_cup	chain	seats
4073	Кафе &Бар	кафе	Москва, площадь Киевского Вокзала, 1	Западный административный округ	ежедневно, 08:00–22:00	55.743343	37.565115	3.4	NaN	NaN	NaN	NaN	0	10.0
5784	Кафе	кафе	Москва, Рябиновая улица, 26, стр. 1	Западный административный округ	пн-пт 08:30–19:00	55.695122	37.423049	3.1	NaN	NaN	NaN	NaN	0	80.0
7022	I like wine	кафе	Москва, Севастопольский проспект, 49	Юго-Западный административный округ	NaN	55.668904	37.585943	3.4	NaN	NaN	NaN	NaN	0	NaN
4150	КОФЕПОРТ	кофейня	Москва, Усачёва улица, 2, стр. 1	Центральный административный округ	пн-пт 08:00–18:30	55.730609	37.576470	4.2	NaN	Цена чашки капучино:90–100 ₽	NaN	95.0	1	96.0
1128	ЛапшаWOK	кафе	Москва, проспект Мира, 119, стр. 501	Северо-Восточный административный округ	ежедневно, 10:00–21:00	55.836063	37.626229	3.9	средние	Средний счёт:300–500 ₽	400.0	NaN	0	NaN

Данные загружены успешно.

Изучим детальнее информацию:

- Сколько заведений представлено?
- Значения какого типа они хранят?
- Что можно сказать о каждом столбце?

In [4]:

```
# Get the number of rows and columns
rows = len(df.axes[0])
cols = len(df.axes[1])
print(f"Датасет содержит информацию о {str(rows)} заведениях г. Москвы.\n\
\nДатасет содержит {str(rows)} строк и {str(cols)} колонок.\n\
\nОбщее количество ненулевых значений в столбцах и\n\
типы данных каждого столбца выведем методом df.info():\n")
print(df.info())
```

Датасет содержит информацию о 8406 заведениях г. Москвы.

Датасет содержит 8406 строк и 14 колонок.

Общее количество ненулевых значений в столбцах и
типы данных каждого столбца выведем методом df.info():

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8406 entries, 0 to 8405
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   name              8406 non-null   object 
 1   category          8406 non-null   object 
 2   address           8406 non-null   object 
 3   district          8406 non-null   object 
 4   hours             7870 non-null   object 
 5   lat               8406 non-null   float64
 6   lng               8406 non-null   float64
 7   rating            8406 non-null   float64
 8   price             3315 non-null   object 
 9   avg_bill          3816 non-null   object 
 10  middle_avg_bill   3149 non-null   float64
 11  middle_coffee_cup  535 non-null   float64
 12  chain             8406 non-null   int64  
 13  seats             4795 non-null   float64
dtypes: float64(6), int64(1), object(7)
memory usage: 919.5+ KB
None
```

1.3.1 Описание колонок

Датафрейм 'df' содержит 14 столбцов:

- 'name' — название заведения;
- 'category' — категория заведения, например «кафе», «пиццерия» или «кофейня»;
- 'address' — адрес заведения;
- 'district' — административный район, в котором находится заведение, например Центральный административный округ;
- 'hours' — информация о днях и часах работы;
- 'lat' — широта географической точки, в которой находится заведение;
- 'lng' — долгота географической точки, в которой находится заведение;
- 'rating' — рейтинг заведения по оценкам пользователей в Яндекс Картах (высшая оценка — 5.0);
- 'price' — категория цен в заведении, например «средние», «ниже среднего», «выше среднего» и так далее;
- 'avg_bill' — строка, которая хранит среднюю стоимость заказа в виде диапазона, например:
  - «Средний счёт: 1000–1500 ₽»;
  - «Цена чашки капучино: 130–220 ₽»;
  - «Цена бокала пива: 400–600 ₽».
  - и так далее;
- 'middle_avg_bill' — число с оценкой среднего чека, которое указано только для значений из столбца avg_bill, начинающихся с подстроки «Средний счёт»:
  - Если в строке указан ценовой диапазон из двух значений, в столбец войдёт медиана этих двух значений.
  - Если в строке указано одно число — цена без диапазона, то в столбец войдёт это число.

    - Если значения нет или оно не начинается с подстроки «Средний счёт», то в столбец ничего не войдёт.
- 'middle_coffee_cup' — число с оценкой одной чашки капучино, которое указано только для значений из столбца avg_bill, начинающихся с подстроки «Цена чашки капучино»:
    - Если в строке указан ценовой диапазон из двух значений, в столбец войдёт медиана этих двух значений.
    - Если в строке указано одно число — цена без диапазона, то в столбец войдёт это число.
    - Если значения нет или оно не начинается с подстроки «Цена одной чашки капучино», то в столбец ничего не войдёт.
- 'chain' — число, выраженное 0 или 1, которое показывает, является ли заведение сетевым (для маленьких сетей могут встречаться ошибки):
    - 0 — заведение не является сетевым
    - 1 — заведение является сетевым
- 'seats' — количество посадочных мест.

1.3.2 Описательная статистика датасета

Исследование качественных переменных удобно начать с метода .describe(). Его применение к категориальным столбцам выдаст:

- общее количество значений (count);
- количество уникальных значений (unique);
- наиболее часто встречающееся значение (top);
- и количество таких значений (freq). Так как есть пропуски, то укажем аргумент include = 'all'.

In [5]:

```
df.describe(include='all')
```

Out[5]:

	name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_avg_bill	middle_coffee_cup	chain	seats
count	8406	8406	8406	8406	7870	8406.000000	8406.000000	8406.000000	3315	3816	3149.000000	535.000000	8406.000000	4795.000000
unique	5614	8	5753	9	1307	NaN	NaN	NaN	4	897	NaN	NaN	NaN	NaN
top	Кафе	кафе	Москва, проспект Вернадского, 86В	Центральный административный округ	ежедневно, 10:00–22:00	NaN	NaN	NaN	средние	Средний счёт:1000–1500 ₽	NaN	NaN	NaN	NaN
freq	189	2378	28	2242	759	NaN	NaN	NaN	2117	241	NaN	NaN	NaN	NaN
mean	NaN	NaN	NaN	NaN	NaN	55.750109	37.608570	4.229895	NaN	NaN	958.053668	174.721495	0.381275	108.421689
std	NaN	NaN	NaN	NaN	NaN	0.069658	0.098597	0.470348	NaN	NaN	1009.732845	88.951103	0.485729	122.833396
min	NaN	NaN	NaN	NaN	NaN	55.573942	37.355651	1.000000	NaN	NaN	0.000000	60.000000	0.000000	0.000000
25%	NaN	NaN	NaN	NaN	NaN	55.705155	37.538583	4.100000	NaN	NaN	375.000000	124.500000	0.000000	40.000000
50%	NaN	NaN	NaN	NaN	NaN	55.753425	37.605246	4.300000	NaN	NaN	750.000000	169.000000	0.000000	75.000000
75%	NaN	NaN	NaN	NaN	NaN	55.795041	37.664792	4.400000	NaN	NaN	1250.000000	225.000000	1.000000	140.000000
max	NaN	NaN	NaN	NaN	NaN	55.928943	37.874466	5.000000	NaN	NaN	35000.000000	1568.000000	1.000000	1288.000000

Категорий заведений 8 и самым распространенным является тип - кафе. Административных округов -9. Адресов и уникальных значений заведений меньше, чем самих заведений на треть. Большое количество уникальных графиков работы заведений - анализировать 1,3тыс графиков нецелесообразно.

1.3.3 Итоги раздела

Данные загружены. Датасет содержит информацию о 8406 заведениях г. Москвы. в 14-ти столбцах.
При первом взгляде уже можно выделить такие моменты:

- столбцы поименованы с соблюдением стиля, значит корректировка не требуется;
- колонка 'hours' - нужно категоризировать графики работы для уменьшения их количества, чтобы использовать эту информацию в анализе;
- колонка 'seats' - количество посадочных мест представлено вещественным числом, так как в столбце имеются пропуски и такой тип данных был присвоен автоматически. Место исчисляется в целых числах. Необходимо решить вопрос с пропусками и привести тип данных к целочисленному типу;
- Колонки 'lat' и 'lng' предоставляют данные о географических координатах заведения, тип данных float64 указан корректно;
- есть пропуски в столбцах:
    - 'hours',
    - 'price',
    - 'avg_bill',
    - 'middle_avg_bill',
    - 'middle_coffee_cup',
    - 'seats'.

Предварительно можно утверждать, что предоставленного объема данных достаточно для исследования.
Проведем предобработку данных.

2 Предобработка данных

- Выполним предобработку данных:
- поищем пропуски в данных: встречаются ли они, в каких столбцах? Можно ли их обработать или оставить как есть?
- Добавим столбец 'street' с названиями улиц из столбца с адресом.
- Добавим столбец 'is_24_7' с обозначением, что заведение работает ежедневно и круглосуточно (24/7):
    - логическое значение True — если заведение работает ежедневно и круглосуточно;
    - логическое значение False — в противоположном случае.
- поищем дубликаты в данных

2.1 Обработка пропусков

При обзоре данных было выявлено, что в некоторых столбцах содержатся пропуски.
Посмотрим можно ли их обработать или оставить как есть. Посчитаем сколько всего пропусков содержится в датасете, а также сколько пропусков в каждом столбце.

In [6]:

```
print(f"Всего датасет содержит {df.isnull().values.sum()} пропусков.")
```

Всего датасет содержит 26956 пропусков.

Для подсчета пропусков в каждом столбце напишем функцию, которая выведет на печать название столбца и количество пропусков в нем.

In [7]:

```
#function to show the nulls total values per column
colum_name = np.array(df.columns.values)
def iter_columns_name(colum_name):
 for k in colum_name:
   if pd.isnull(df[k]).values.ravel().sum() != 0:
       print("Пропусков строк в столбце \'{}\' =
".format(k),pd.isnull(df[k]).values.ravel().sum(),\
             f"шт. Это {round((pd.isnull(df[k]).values.ravel().sum())/ len(df) *100,
2)} %")
#call the function
iter_columns_name(colum_name)
```

Пропусков строк в столбце 'hours' = 536 шт. Это 6.38 %
Пропусков строк в столбце 'price' = 5091 шт. Это 60.56 %
Пропусков строк в столбце 'avg_bill' = 4590 шт. Это 54.6 %
Пропусков строк в столбце 'middle_avg_bill' = 5257 шт. Это 62.54 %

Пропусков строк в столбце 'middle_coffee_cup' = 7871 шт. Это 93.64 %
Пропусков строк в столбце 'seats' = 3611 шт. Это 42.96 %

Пропусков в данных большое количество. Просто удалить их нельзя, т.к. будут искажения в исследовании. Исследуем эти значения и посмотрим чем их можно заменить.

2.1.1 Пропуски в столбце hours

Пропусков в столбцах hours не много чуть более 6%. Посмотрим на заведения, которые содержат пропуски с часами работы.

In [8]:

```
df_hours_null = df[df['hours'].isna()]
df_hours_null['category'].value_counts()
```

Out[8]:

```
кафе              375
ресторан           74
быстрое питание    33
бар,паб            18
кофейня            15
столовая            9
булочная            7
пиццерия            5
Name: category, dtype: int64
```

Больше всего пропусков у категории кафе. Посмотрим, какую долю эти заведения занимают в среди тех, которые работают ежедневно и круглосуточно и примем решение, можно ли заменить пропуски на что-то. Остальные виды графиков рассматривать нецелесообразно, слишком их много.

In [9]:

```
ratio_24_7 = round(len(df.query('hours == "ежедневно, круглосуточно"'))\
             / len(df) * 100)
ratio_cafe_24_7 = round(len(df.query('category == "кафе" & hours == "ежедневно, круглосуточно"'))\
             / len(df.query('category == "кафе"')) * 100)
print(f'Доля работающих круглосуточно и ежедневно заведений равна: {ratio_24_7} %,\n\
и доля кафе среди работающих круглосуточно составляет: {ratio_cafe_24_7} %.')
```

```
Доля работающих круглосуточно и ежедневно заведений равна: 9 %,
и доля кафе среди работающих круглосуточно составляет: 11 %.
```

Чтож, заведений, которые работают 24/7 составляет менее 10%, доля кафе среди таких заведений не лидирующая - 11%, если сравнивать совокупную долю всех остальных заведений, что и требуется в этом конкретном случае.

Для заполнения пропусков в категориальных данных подойдет метод заполнения наиболее часто встречающимся значением (модой). Если пропусков немного, этот метод вполне обоснован. В нашем случае достаточно было убедиться, что значения столбца 'hours' с графиком работы "ежедневно, круглосуточно" не являются модой среди всех вариантов графиков работы заведений. Нам это важно, так как в дальнейшем будет добавлен столбец с булевым значением 'True' если заведение работает по графику "ежедневно, круглосуточно". Остальные графики работы мы рассматривать не будем.

Поэтому в данном случае смело заменим все пропуски в столбце 'hours' значением "unknown". Сохраним результат и выведем несколько случайных строк.

In [10]:

```
df['hours'] = df['hours'].fillna('unknown')
df.sample(n=3, random_state=6)
```

Out[10]:

	name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_avg_bill	middle_coffee_cup	chain	seats

5628	Кафе	кафе	Москва, Аминьевское шоссе, 3А	Западный административный округ	ежедневно, 10:00–22:00	55.702685	37.457738	4.0	NaN	NaN	NaN	NaN	0	NaN
1269	Трактир	кафе	Москва, улица Врубеля, 13А	Северный административный округ	ежедневно, 10:00–23:00	55.804015	37.498888	4.3	NaN	NaN	NaN	NaN	1	20.0
2548	Доброд ар	бар,паб	Москва, Комсомольская площадь, 5	Центральный административный округ	пн-пт 08:30–20:00; сб,вс 09:00–20:00	55.777056	37.656708	2.9	NaN	NaN	NaN	NaN	0	NaN

In [11]:

```
print('Пропусков в столбце "hours" после заполнения:',
df['hours'].isnull().values.sum())
```

Пропусков в столбце "hours" после заполнения: 0

2.1.2 Пропуски в столбцах price, avg_bill и seats

Пропусков строк в столбце 'price' - 5091 шт. Это 60.56 %.
Пропусков строк в столбце 'avg_bill' - 4590 шт. Это 54.6 %.
Пропусков строк в столбце 'seats' = 3611 шт. Это 42.96 %.

- 'price' содержит информацию о категории цен в заведении, например «средние», «ниже среднего», «выше среднего» и так далее.
- 'avg_bill' — строка, которая хранит среднюю стоимость заказа в виде диапазона, например:
  - «Средний счёт: 1000–1500 ₽»;
  - «Цена чашки капучино: 130–220 ₽»;
  - «Цена бокала пива: 400–600 ₽»
  - и так далее;
- 'seats' - строка содержит информацию о количество посадочных мест.
  Пропусков очень много, если заполнить эти пропуски медианными или средними значениями по группам, или же модой - это сильно исказит данные.

Оставим пропуски в столбцах 'price', avg_bill и 'seats' как есть, без изменений. Отсутствие информации - тоже информация и в данном случае заполнение пропусков не обоснованно.

2.1.3 Пропуски в столбцах middle_avg_bill и middle_coffee_cup

Эти столбцы содержат число с оценкой среднего чека из столбца avg_bill. Заполняются значения только если есть определенные условия,а именно:

- 'middle_avg_bill' — число с оценкой среднего чека, которое указано только для значений из столбца avg_bill, начинающихся с подстроки «Средний счёт». Если значения нет или оно не начинается с подстроки «Средний счёт», то в столбец ничего не войдёт.
- 'middle_coffee_cup' — число с оценкой одной чашки капучино, которое указано только для значений из столбца avg_bill, начинающихся с подстроки «Цена одной чашки капучино». Если значения нет или оно не начинается с подстроки «Цена одной чашки капучино», то в столбец ничего не войдёт.

Проверим число строк в колонке 'avg_bill', которые начинаются с подстроки «Средний счёт» и сравним с числом строк с заполненными значениями в столбце 'middle_avg_bill'.

In [12]:

```
print(f"Число строк в колонке 'avg_bill', которые начинаются с подстроки «Средний
счёт»: \
{df['avg_bill'].str.contains('Средний счёт').value_counts()[True]} шт.")
print(f"Число заполненных строк в колонке 'middle_avg_bill': \
{df['middle_avg_bill'].count()} шт.")
if df['avg_bill'].str.contains('Средний счёт').value_counts()[True] -
df['middle_avg_bill'].count() == 0:
   print('Пропуски в колонке middle_avg_bill обоснованы. Оставляем без изменений.')
```

Число строк в колонке 'avg_bill', которые начинаются с подстроки «Средний счёт»: 3149 шт.

Число заполненных строк в колонке 'middle_avg_bill': 3149 шт.
Пропуски в колонке middle_avg_bill обоснованы. Оставляем без изменений.

Аналогично сравним число строк в колонке 'avg_bill', которые начинаются с подстроки «Цена одной чашки капучино» и сравним с числом строк с заполненными значениями в столбце 'middle_coffee_cup'.
Поиск "Цена одной чашки капучино" ничего не дал. Посмотрим какие уникальные значения есть в столбце 'avg_bill'

In [13]:

```
df['avg_bill'].unique()
```

Out[13]:

array([nan, 'Средний счёт:1500–1600 ₽', 'Средний счёт:от 1000 ₽',
'Цена чашки капучино:155–185 ₽', 'Средний счёт:400–600 ₽',
'Средний счёт:199 ₽', 'Средний счёт:200–300 ₽',
… ВЫРЕЗАНО…
'Цена бокала пива:160–390 ₽', 'Средний счёт:120–130 ₽',
'Цена чашки капучино:120–220 ₽', 'Цена чашки капучино:100–220 ₽',
'Цена чашки капучино:80–120 ₽'], dtype=object)

Видим есть значения, начинающиеся с «Цена чашки капучино». Видимо в описании условий слово "одной" лишнее. Проверим эту гипотезу и осуществим поиск в колонке 'avg_bill', строк которые начинаются с подстроки «Цена чашки капучино».

In [14]:

```
print(f"Число строк в колонке 'avg_bill', которые начинаются с подстроки «Цена чашки капучино»: \
{df['avg_bill'].str.contains('Цена чашки капучино').value_counts()[True]} шт.")
print(f"Число заполненных строк в колонке 'middle_coffee_cup': \
{df['middle_coffee_cup'].count()} шт.")
if df['avg_bill'].str.contains('Цена чашки капучино').value_counts()[True] -
df['middle_coffee_cup'].count() == 0:
   print('Пропуски в колонке middle_coffee_cup обоснованы. Оставляем без изменений.')
```

Число строк в колонке 'avg_bill', которые начинаются с подстроки «Цена чашки капучино»: 535 шт.
Число заполненных строк в колонке 'middle_coffee_cup': 535 шт.
Пропуски в колонке middle_coffee_cup обоснованы. Оставляем без изменений.

2.1.4 Итоги обработки пропусков

- Оставили пропуски в столбцах 'price', avg_bill и 'seats' как есть, без изменений. В данном случае отсутствие информации - тоже информация и заполнение пропусков заглушкой не обосновано. Заполнить их автоматизированно без существенных затрат времени на поиск информации по каждому заведению затруднительно.
- Пропуски в колонке middle_coffee_cup и в колонке middle_avg_bill обоснованы. Оставляем без изменений.

2.2 Добавим столбец 'street'

Выделим из полного адреса заведения название улицы из колонки с адресом и запишем его в столбец 'street'.
Посмотрим сначала на уникальные адреса в столбце 'address'

In [15]:

```
print(list(df['address'].unique()))
```

['Москва, улица Дыбенко, 7/1', 'Москва, улица Дыбенко, 36, корп. 1', 'Москва, Клязьминская улица, 15',
'Москва, улица Маршала Федоренко, 12', 'Москва, Правобережная улица, 1Б', 'Москва, Ижорская улица,
… ВЫРЕЗАНО…
'Москва, улица Лобачевского, 52, корп. 1', 'Москва, Болотниковская улица, 52, корп. 2', 'Москва, Чонгарский бульвар, 26А, корп. 1']

Видим, что адрес это строка, в которой служит разделителем запятая между названием города, улицы, номером дома, строением. Уверенности, что сразу после города Москва будет название улицы нет. Поэтому вместо того, чтобы разделить строку на список подстрок по разделителю метода str.split(', ') используем более универсальный способ с помощью регулярных выражений.

Полный адрес разделим на компоненты и сохраним в столбец нужную нам часть адреса-улицу в столбец 'street'.

Составим список типов улиц и сохраним ее в переменной street_list. Типы улиц взяты из открытых источников в интернете.

Тип улицы(улица, бульвар шоссе и т.д.) может стоять как перед названием, так и после. Учтем это при составлении регулярного выражения. В поиске участвуют типы улиц из сохраненного списка в street_list. Для удобства сохраним в переменную search_street наше регулярное выражение.

Передадим регулярное выражение в .str.extract(), чтобы «извлечь» части каждой ячейки в Series. В .str.extract(), .str — это аксессор, а .str.extract() — метод аксессора.

Чтобы регулярные выражения не рассматривали заглавные и прописные буквы как разные символы, передадим re-функции дополнительный аргумент flags=flags=re.IGNORECASE.

In [16]:

```
# список типов улиц
street_list = ['аллея', 'бульвар',
               'взвоз', 'дорожка',
               'квартал', 'линия',
               'магистраль', 'мост',
               'микрорайон', 'микро-район',
               'микро район', 'мкад',
               'набережная', 'площадь',
               'парк',
               'природно-исторический парк',
               'парк культуры и отдыха',
               'просек',
               'переулок', 'просека',
               'проспект', 'проезд',
               'пр', 'пр.',
               'сквер', 'тупик',
               'ул', 'ул.',
               'улица',
               'улица ',
               'улица'
               'линия',
               'территория', 'шоссе']

search_street = r".*,\s*\b([^,]*?(?:{})\b[^,]*)[,$]+".format("|".join(street_list))
```

In [17]:

```
df['street'] = df['address'].str.extract(search_street, flags=re.IGNORECASE)
```

In [18]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8406 entries, 0 to 8405
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   name             8406 non-null   object
 1   category         8406 non-null   object
 2   address          8406 non-null   object
 3   district         8406 non-null   object
 4   hours            8406 non-null   object
```

```
 5   lat               8406 non-null   float64
 6   lng               8406 non-null   float64
 7   rating            8406 non-null   float64
 8   price             3315 non-null   object
 9   avg_bill          3816 non-null   object
 10  middle_avg_bill   3149 non-null   float64
 11  middle_coffee_cup  535 non-null   float64
 12  chain             8406 non-null   int64
 13  seats             4795 non-null   float64
 14  street            8273 non-null   object
dtypes: float64(6), int64(1), object(8)
memory usage: 985.2+ KB
```

Видим, что столбец 'street' содержит пропуски. Выведем на экран уникальные адреса данных с пропущенными значениями в столбце 'street' применив метод isna().

In [19]:

```
df.query('street.isna()').address.unique()
```

Out[19]:

```
array(['Москва, парк Левобережный',
       'Москва, ландшафтный заказник Лианозовский',
       'Москва, Лианозовский парк культуры и отдыха',
       'Москва, парк Алтуфьево', 'Москва, парк Ангарские пруды',
       'Москва, Проектируемый проезд № 5265',
       'Москва, парк Ангарские Пруды',
       'Москва, парк Этнографическая деревня Бибирево',
       'Москва, Северный административный округ, Головинский район',
       'Москва, парк культуры и отдыха Северное Тушино',
       'Москва, Северо-Западный административный округ, район Северное Тушино',
       'Москва, парк Дружбы', 'Москва, Ленинградское шоссе',
       'Москва, Северный административный округ, район Левобережный, территория парка Дружбы',
       'Москва, Коптевский бульвар д 18 А стр 1',
       'Москва, Грузинский сквер',
       'Москва, Северо-Восточный административный округ, район Отрадное',
       'Москва, Северо-Восточный административный округ, Останкинский район, Выставка достижений народного хозяйства, Кольцевая дорога',
       'Москва, парк Останкино',
       'Москва, Главный ботанический сад имени Н.В. Цицина Российской академии наук',
       'Москва, Северо-Восточный административный округ, район Ростокино',
       'Москва, Северо-Восточный административный округ, Останкинский район, Выставка достижений народного хозяйства, Главная аллея',
       'Москва, парк Сад будущего',
       'Москва, Северо-Западный административный округ, район Строгино',
       'Москва, памятник природы Серебряный бор',
       'Москва, улица Черняховского',
       'Москва, Северный административный округ, Хорошёвский район, микрорайон Ходынское Поле',
       'Москва, 3-й Лучевой просек',
       'Москва, Восточный административный округ, район Сокольники, территория парка Сокольники',
       'Москва, Третье транспортное кольцо', 'Москва, Рижский проезд',
       'Москва, парк Сокольники', 'Москва, проезд Сокольнического Круга',
       'Москва, Песочная аллея', 'Москва, улица Гиляровского',
       'Москва, Восточный административный округ, район Богородское, жилой комплекс Богородский',
       'Москва, природно-исторический парк Измайлово',
       'Москва, Измайловский парк культуры и отдыха',
```

'Москва, Западный административный округ, район Крылатское',
'Москва, парк культуры и отдыха Фили',
'Москва, 2-я Филёвская улица', 'Москва, Ворошиловский парк',
'Москва, 2-й Силикатный проезд',
'Москва, Центральный административный округ, Пресненский район, жилой комплекс Редсайд, 37',
'Москва, парк Красная Пресня', 'Москва, сад Эрмитаж',
'Москва, Пушкинская набережная', 'Москва, парк искусств Музеон',
'Москва, Центральный административный округ, район Якиманка',
'Москва, Таганский парк культуры и отдыха',
'Москва, Коммунистический переулок', 'Москва, Павелецкая площадь',
'Москва, Перовский парк культуры и отдыха', 'Москва, парк Радуга',
'Москва, Восточный административный округ, район Измайлово',
'Москва, Терлецкий лесопарк', 'Москва, Авиамоторная улица',
'Москва, сквер имени М.И. Калинина',
'Москва, Западный административный округ, район Проспект Вернадского',
'Москва, Ботанический сад Московского государственного университета',
'Москва, парк Олимпийской Деревни',
'Москва, Ленинский проспект (дублёр)',
'Москва, Андреевская набережная',
'Москва, Воробьёвская набережная',
'Москва, Андреевский пешеходный мост',
'Москва, Гагаринский тоннель', 'Москва, № 7',
'Москва, Центральный парк культуры и отдыха имени М. Горького',
'Москва, Северный ландшафтный парк', 'Москва, Рязанский проспект',
'Москва, улица Шкулёва 4',
'Москва, Западный административный округ, район Тропарёво-Никулино, Центральная аллея',
'Москва, ландшафтный заказник Тёплый Стан',
'Москва, Воронцовский парк', 'Москва, парк Зюзино',
'Москва, Юго-Западный административный округ, Академический район',
'Москва, музей-заповедник Коломенское',
'Москва, парк Технических видов спорта', 'Москва, Большая улица',
'Москва, парк Братеевская набережная',
'Москва, парк имени 850-летия города Москвы',
'Москва, Юго-Восточный административный округ, район Капотня',
'Москва, Братиславский парк', 'Москва, парк имени Артёма Боровика',
'Москва, парк Борисовские пруды', 'Москва, Братеевский парк',
'Москва, Сумская, 2/12', 'Москва, Варшавское шоссе',
'Москва, Липецкая улица (дублёр)',
'Москва, Южный административный округ, район Орехово-Борисово Северное, Воздушная улица',
'Москва, 2-й Павелецкий проезд', 'Москва, парк Тюфелева роща',
'Москва, Юго-Восточный административный округ, район Кузьминки',
'Москва, Юго-Восточный административный округ, Нижегородский район'],
dtype=object)

Регулярное выражение сработало не на все адреса. Какое-то количество адресом можно довнести вручную. Составим словарь с адресами и далее с помощью метода map() заменим пустые значения на те, что можем поменять: нефрматный адрес. а также стоит внести парки Москвы, их много и много точек общепита в них.

In [20]:

```
df_map_street = df[df['street'].isna() & df['address'].str.contains(r'\улица')]
print(f"Улицы с нестандартным названием для замену
вручную:\n{list(df_map_street['address'])}")
```

Улицы с нестандартным названием для замену вручную:

['Москва, улица Черняховского', 'Москва, улица Гиляровского', 'Москва, 2-я Филёвская улица', 'Москва, Авиамоторная улица', 'Москва, улица Шкулёва 4', 'Москва, Большая улица', 'Москва, Липецкая улица (дублёр)', 'Москва, Южный административный округ, район Орехово-Борисово Северное, Воздушная улица']

In [21]:

```
 #словарь для замены ключ - название парка для замены
df_map_park = df[df['street'].isna() & df['address'].str.contains(r'\парк')]
print(f"Парки Москвы для внесения в адрес
вручную:\n{list(df_map_park['address'].unique())}")
```

Парки Москвы для внесения в адрес вручную:
['Москва, парк Левобережный', 'Москва, Лианозовский парк культуры и отдыха', 'Москва, парк Алтуфьево', 'Москва, парк Ангарские пруды', 'Москва, парк Ангарские Пруды', 'Москва, парк Этнографическая деревня Бибирево', 'Москва, парк культуры и отдыха Северное Тушино', 'Москва, парк Дружбы', 'Москва, Северный административный округ, район Левобережный, территория парка Дружбы', 'Москва, парк Останкино', 'Москва, парк Сад будущего', 'Москва, Восточный административный округ, район Сокольники, территория парка Сокольники', 'Москва, парк Сокольники', 'Москва, природно-исторический парк Измайлово', 'Москва, Измайловский парк культуры и отдыха', 'Москва, парк культуры и отдыха Фили', 'Москва, Ворошиловский парк', 'Москва, парк Красная Пресня', 'Москва, парк искусств Музеон', 'Москва, Таганский парк культуры и отдыха', 'Москва, Перовский парк культуры и отдыха', 'Москва, парк Радуга', 'Москва, Терлецкий лесопарк', 'Москва, парк Олимпийской Деревни', 'Москва, Центральный парк культуры и отдыха имени М. Горького', 'Москва, Северный ландшафтный парк', 'Москва, Воронцовский парк', 'Москва, парк Зюзино', 'Москва, парк Технических видов спорта', 'Москва, парк Братеевская набережная', 'Москва, парк имени 850-летия города Москвы', 'Москва, Братиславский парк', 'Москва, парк имени Артёма Боровика', 'Москва, парк Борисовские пруды', 'Москва, Братеевский парк', 'Москва, парк Тюфелева роща']

In [22]:

```
df_map_other = df[df['street'].isna() & df['address']\
                  .str.contains(r'\пешеходный
мост|набережная|сад|сквер|бульвар|проезд|площадь|шоссе|жилой комплекс|аллея')]
print(f"Остальные адреса с нестандартным названием для замену
вручную:\n{list(df_map_other['address'].unique())}")
```

Остальные адреса с нестандартным названием для замену вручную:
['Москва, Проектируемый проезд № 5265', 'Москва, Ленинградское шоссе', 'Москва, Коптевский бульвар д 18 А стр 1', 'Москва, Грузинский сквер', 'Москва, Главный ботанический сад имени Н.В. Цицина Российской академии наук', 'Москва, Северо-Восточный административный округ, Останкинский район, Выставка достижений народного хозяйства, Главная аллея', 'Москва, Рижский проезд', 'Москва, проезд Сокольнического Круга', 'Москва, Песочная аллея', 'Москва, Восточный административный округ, район Богородское, жилой комплекс Богородский', 'Москва, 2-й Силикатный проезд', 'Москва, Центральный административный округ, Пресненский район, жилой комплекс Редсайд, 37', 'Москва, сад Эрмитаж', 'Москва, Пушкинская набережная', 'Москва, Павелецкая площадь', 'Москва, сквер имени М.И. Калинина', 'Москва, Ботанический сад Московского государственного университета', 'Москва, Андреевская набережная', 'Москва, Воробьёвская набережная', 'Москва, Андреевский пешеходный мост', 'Москва, Западный административный округ, район Тропарёво-Никулино, Центральная аллея', 'Москва, парк Братеевская набережная', 'Москва, Варшавское шоссе', 'Москва, 2-й Павелецкий проезд']

Создадим словарь для замены пропусков в столбце street. Где ключом будет полный адрес из столбца 'address'. Если улицы нет, то данные будут заполнены на unknown. Если будет указан парк г. Москвы, то он послужит вместо названия улицы. В парках могут быть точки общепита.

In [23]:

```
 #словарь для замены ключ - название улицы для замены
map_street = {'Москва, улица Черняховского': 'Черняховского улица',
              'Москва, улица Гиляровского': 'Гиляровского улица',
              'Москва, 2-я Филёвская улица': '2-я Филёвская улица',
              'Москва, Авиамоторная улица': 'Авиамоторная улица',
              'Москва, улица Шкулёва 4': ' улица Шкулёва',
```

```
            'Москва, Большая улица': 'Большая улица',
            'Москва, Липецкая улица (дублёр)': 'Липецкая улица (дублёр)',
            'Москва, Южный административный округ, район Орехово-Борисово Северное,
Воздушная улица': 'Воздушная улица',
            'Москва, парк Левобережный': 'парк Левобережный',
          'Москва, Лианозовский парк культуры и отдыха': 'Лианозовский парк культуры
и отдыха',
          'Москва, парк Алтуфьево': 'парк Алтуфьево',
          'Москва, парк Ангарские пруды': 'парк Ангарские пруды',
          'Москва, парк Ангарские Пруды': 'парк Ангарские пруды',
          'Москва, парк Этнографическая деревня Бибирево': 'парк Этнографическая
деревня Бибирево',
          'Москва, парк культуры и отдыха Северное Тушино': 'парк культуры и отдыха
Северное Тушино',
          'Москва, парк Дружбы': 'территория парка Дружбы',
          'Москва, Северный административный округ, район Левобережный, территория
парка Дружбы': 'территория парка Дружбы',
          'Москва, парк Останкино': 'парк Останкино',
          'Москва, парк Сад будущего': 'парк Сад будущего',
          'Москва, Восточный административный округ, район Сокольники, территория
парка Сокольники': 'территория парка Сокольники',
          'Москва, парк Сокольники': 'территория парка Сокольники',
          'Москва, природно-исторический парк Измайлово': 'природно-исторический
парк Измайлово',
          'Москва, Измайловский парк культуры и отдыха': 'Измайловский парк культуры
и отдыха',
          'Москва, парк культуры и отдыха Фили': 'парк культуры и отдыха Фили',
          'Москва, Ворошиловский парк': 'Ворошиловский парк',
          'Москва, парк Красная Пресня': 'парк Красная Пресня',
          'Москва, парк искусств Музеон': 'парк искусств Музеон',
          'Москва, Таганский парк культуры и отдыха': 'Таганский парк культуры и
отдыха',
          'Москва, Перовский парк культуры и отдыха': 'Перовский парк культуры и
отдыха',
          'Москва, парк Радуга': 'парк Радуга',
          'Москва, Терлецкий лесопарк': 'ерлецкий лесопарк',
          'Москва, парк Олимпийской Деревни': 'парк Олимпийской Деревни',
          'Москва, Центральный парк культуры и отдыха имени М. Горького':
'Центральный парк культуры и отдыха имени М. Горького',
          'Москва, Северный ландшафтный парк': 'Северный ландшафтный парк',
          'Москва, Воронцовский парк': 'Воронцовский парк',
          'Москва, парк Зюзино': 'парк Зюзино',
          'Москва, парк Технических видов спорта': 'парк Технических видов спорта',
          'Москва, парк Братеевская набережная': 'парк Братеевская набережная',
          'Москва, парк имени 850-летия города Москвы': 'парк имени 850-летия города
Москвы',
          'Москва, Братиславский парк': 'Братиславский парк',
          'Москва, парк имени Артёма Боровика': 'парк имени Артёма Боровика',
          'Москва, парк Борисовские пруды': 'парк Борисовские пруды',
          'Москва, Братеевский парк': 'Братеевский парк',
          'Москва, парк Тюфелева роща': 'парк Тюфелева роща',
        'Москва, Проектируемый проезд № 5265': 'Проектируемый проезд № 5265',
  'Москва, Ленинградское шоссе': 'Ленинградское шоссе',
  'Москва, Коптевский бульвар д 18 А стр 1': 'Коптевский бульвар, д 18 А, стр 1',
```

```
   'Москва, Грузинский сквер': 'Грузинский сквер',
   'Москва, Главный ботанический сад имени Н.В. Цицина Российской академии наук':
'Главный ботанический сад имени Н.В. Цицина Российской академии наук',
   'Москва, Северо-Восточный административный округ, Останкинский район, Выставка
достижений народного хозяйства, Главная аллея': 'Главная аллея',
   'Москва, Рижский проезд': 'Рижский проезд',
   'Москва, проезд Сокольнического Круга': 'проезд Сокольнического круга',
   'Москва, Песочная аллея': 'Песочная аллея',
   'Москва, Восточный административный округ, район Богородское, жилой комплекс
Богородский': 'жилой комплекс Богородский',
   'Москва, 2-й Силикатный проезд': '2-й Силикатный проезд',
   'Москва, Центральный административный округ, Пресненский район, жилой комплекс
Редсайд, 37': 'жилой комплекс Редсайд, 37',
   'Москва, сад Эрмитаж': 'сад Эрмитаж',
   'Москва, Павелецкая площадь': 'Павелецкая площадь',
   'Москва, сквер имени М.И. Калинина': 'сквер имени М.И. Калинина',
   'Москва, Ботанический сад Московского государственного университета':
'Ботанический сад Московского государственного университета',
   'Москва, Андреевская набережная': 'Андреевская набережная',
   'Москва, Воробьёвская набережная': 'Воробьёвская набережная',
   'Москва, Андреевский пешеходный мост': 'Андреевский пешеходный мост',
   'Москва, Пушкинская набережная': 'Пушкинская набережная',
   'Москва, Западный административный округ, район Тропарёво-Никулино, Центральная
аллея': 'Центральная аллея',
   'Москва, Варшавское шоссе': 'Варшавское шоссе',
   'Москва, парк Братеевская набережная': 'парк Братеевская набережная',
   'Москва, 2-й Павелецкий проезд': '2-й Павелецкий проезд',
   'Москва, парк Левобережный': 'парк Левобережный',
      'Москва, ландшафтный заказник Лианозовский': 'ландшафтный заказник
Лианозовский',
      'Москва, Лианозовский парк культуры и отдыха': 'Лианозовский парк культуры и
отдыха',
      'Москва, парк Алтуфьево':'парк Алтуфьево',
   'Москва, парк Ангарские пруды': 'парк Ангарские пруды',
      'Москва, парк Ангарские Пруды': 'Ангарские Пруды',
      'Москва, парк Этнографическая деревня Бибирево': 'Этнографическая деревня
Бибирево',
      'Москва, Северный административный округ, Головинский район': 'unknown',
      'Москва, парк культуры и отдыха Северное Тушино': 'парк культуры и отдыха
Северное Тушино',
      'Москва, Северо-Западный административный округ, район Северное Тушино':
'unknown',
      'Москва, парк Дружбы': 'парк Дружбы',
      'Москва, Северный административный округ, район Левобережный, территория парка
Дружбы': 'территория парка Дружбы',
      'Москва, Северо-Восточный административный округ, район Отрадное': 'unknown',
      'Москва, Северо-Восточный административный округ, Останкинский район, Выставка
достижений народного хозяйства, Кольцевая дорога': 'Кольцевая дорога',
      'Москва, парк Останкино': 'парк Останкино',
      'Москва, Северо-Восточный административный округ, район Ростокино': 'unknown',
      'Москва, парк Сад будущего': 'парк Сад будущего',
      'Москва, Северо-Западный административный округ, район Строгино': 'unknown',
      'Москва, памятник природы Серебряный бор': 'памятник природы Серебряный бор',
      'Москва, улица Черняховского': 'Черняховского улица',
```

```
      'Москва, Северный административный округ, Хорошёвский район, микрорайон
Ходынское Поле': 'микрорайон Ходынское Поле',
      'Москва, 3-й Лучевой просек': '3-й Лучевой просек',
      'Москва, Восточный административный округ, район Сокольники, территория парка
Сокольники': 'парк Сокольники',
      'Москва, Третье транспортное кольцо': 'unknown',
   'Москва, парк Сокольники': 'парк Сокольники',
      'Москва, улица Гиляровского': 'Гиляровского улица',
      'Москва, природно-исторический парк Измайлово': 'природно-исторический парк
Измайлово',
      'Москва, Измайловский парк культуры и отдыха': 'Измайловский парк культуры и
отдыха',
      'Москва, Западный административный округ, район Крылатское': 'unknown',
      'Москва, парк культуры и отдыха Фили': 'парк культуры и отдыха Фили',
      'Москва, 2-я Филёвская улица':'2-я Филёвская улица',
   'Москва, Ворошиловский парк': 'Ворошиловский парк',
      'Москва, парк Красная Пресня': 'парк Красная Пресня',
   'Москва, парк искусств Музеон': 'парк искусств Музеон',
      'Москва, Центральный административный округ, район Якиманка': 'unknown',
      'Москва, Таганский парк культуры и отдыха': 'Таганский парк культуры и отдыха',
      'Москва, Коммунистический переулок': 'Коммунистический переулок',
      'Москва, Перовский парк культуры и отдыха': 'Перовский парк культуры и отдыха',
   'Москва, парк Радуга': 'парк Радуга',
      'Москва, Восточный административный округ, район Измайлово': 'unknown',
      'Москва, Терлецкий лесопарк': 'Терлецкий лесопарк',
   'Москва, Авиамоторная улица': 'Авиамоторная улица',
      'Москва, Западный административный округ, район Проспект Вернадского':
'unknown',
      'Москва, парк Олимпийской Деревни': 'парк Олимпийской Деревни',
      'Москва, Ленинский проспект (дублёр)': 'Ленинский проспект (дублёр)',
      'Москва, Гагаринский тоннель': 'Гагаринский тоннель',
   'Москва, № 7': 'unknown',
      'Москва, Центральный парк культуры и отдыха имени М. Горького': 'Центральный
парк культуры и отдыха имени М. Горького',
      'Москва, Северный ландшафтный парк': 'Северный ландшафтный парк',
   'Москва, Рязанский проспект': 'Рязанский проспект',
      'Москва, улица Шкулёва 4': 'Шкулёва улица, 4',
      'Москва, ландшафтный заказник Тёплый Стан': 'ландшафтный заказник Тёплый Стан',
      'Москва, Воронцовский парк': 'Воронцовский парк',
   'Москва, парк Зюзино': 'парк Зюзино',
      'Москва, Юго-Западный административный округ, Академический район': 'unknown',
      'Москва, музей-заповедник Коломенское': 'музей-заповедник Коломенское',
      'Москва, парк Технических видов спорта': 'парк Технических видов спорта',
   'Москва, Большая улица': 'Большая улица',
      'Москва, парк имени 850-летия города Москвы': 'парк имени 850-летия города
Москвы',
      'Москва, Юго-Восточный административный округ, район Капотня': 'unknown',
      'Москва, Братиславский парк': 'Братиславский парк',
   'Москва, парк имени Артёма Боровика': 'парк имени Артёма Боровика',
      'Москва, парк Борисовские пруды': 'парк Борисовские пруды',
   'Москва, Братеевский парк': 'Братеевский парк',
      'Москва, Сумская, 2/12': 'Сумская улица, 2/12',
   'Москва, Липецкая улица (дублёр)': 'Липецкая улица (дублёр)',
```

```
      'Москва, Южный административный округ, район Орехово-Борисово Северное,
Воздушная улица': 'Воздушная улица',
      'Москва, парк Тюфелева роща': 'парк Тюфелева роща',
      'Москва, Юго-Восточный административный округ, район Кузьминки': 'unknown',
      'Москва, Юго-Восточный административный округ, Нижегородский район': 'unknown'
         }
```

Применим созданный словарь map_street с данными для замены Null в столбце street

In [24]:

```
mask = df['street'].astype('str').isin(['NaN','nan'])
 # заносим адреса улиц в столбец street по словарю -заменяем пропуски значением для
замены - ключ в столбце address
df.loc[mask, 'street'] = df.loc[mask, 'address'].map(map_street)
```

In [25]:

```
df.query('street.isna()').address.unique()
```

Out[25]:

array([], dtype=object)

Срез показал остутствие пропусков в столбце street. Так как в словарь было добавлено значение unknown для тех адресов, для которых не нашлось улиц, то выведем общую информацию по срезу с данными в колонке street == "unknown".

In [26]:

```
df.query('street == "unknown"').info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 28 entries, 316 to 8395
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   name             28 non-null     object
 1   category         28 non-null     object
 2   address          28 non-null     object
 3   district         28 non-null     object
 4   hours            28 non-null     object
 5   lat              28 non-null     float64
 6   lng              28 non-null     float64
 7   rating           28 non-null     float64
 8   price            1 non-null      object
 9   avg_bill         1 non-null      object
 10  middle_avg_bill  1 non-null      float64
 11  middle_coffee_cup  0 non-null    float64
 12  chain            28 non-null     int64
 13  seats            0 non-null      float64
 14  street           28 non-null     object
dtypes: float64(6), int64(1), object(8)
memory usage: 3.5+ KB
```

Заведений без улиц - 28 штук - для таких улиц применена заглушка "unknown".
Проведена замена адресов с применением регулярных выражений, также использована замена пропусков массивом данных из словаря. Получилось трудоемко, зато парки Москвы вошли в столбец с данными.

In [27]:

```
print(f"Всего уникальных названий улиц в столбце 'street': {df['street'].nunique()}
шт.")
```

```
Всего уникальных названий улиц в столбце 'street': 1462 шт.
```

Проверим изменится ли количество уникальных адресов, если мы произведем замену в наименовании улиц буквы ё на е.

In [28]:

```
df_street_e = df['street']
df_street_e = df_street_e.str.replace('ё', 'е', regex=True)
if df_street_e.nunique() - df['street'].nunique() > 0:
   print('Необходимо произвести замену в наименовании улиц буквы "ё" на "е".')
else:
   print('Замена в наименовании улиц буквы "ё" на "е" не требуется.\nОставляем букву "ё" в столбце "street"!')
```

Замена в наименовании улиц буквы "ё" на "е" не требуется.
Оставляем букву "ё" в столбце "street"!

2.3 Добавим столбец 'is_24_7'

Так как заведения имеют различный график работы, то сгруппируем их на те которые работают "ежедневно и круглосуточно" ,т.е. (24/7) и все остальные.
Для этого добавим столбец 'is_24_7' в котором:

- логическое значение True — если заведение работает ежедневно и круглосуточно;
- логическое значение False — в противоположном случае.

Посмотрим какое количество уникальных значений графиков работы у заведений.

In [29]:

```
print(f"Всего разновидностей графиков работы у заведений: {df['hours'].nunique()} шт.")
```

Всего разновидностей графиков работы у заведений: 1308 шт.

Ежедневно не значит круглосуточно, а круглосуточно - не значит ежедневно. Добавим оба этих условия для отбора в столбец 'is_24/7'

In [30]:

```
df['is_24/7'] = df['hours'].str.contains('ежедневно, круглосуточно')
df.sample(n=2, random_state=8)
```

Out[30]:

	name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_avg_bill	middle_coffee_cup	chain	seats	street	is_24/7
7298	Чайхона Анор	кафе	Москва, Тихорецкий бульвар, 1с29	Юго-Восточный административный округ	ежедневно, круглосуточно	55.67683	37.779148	4.7	NaN	NaN	NaN	NaN	0	NaN	Тихорецкий бульвар	True
7790	Айан	ресторан	Москва, Мелитопольская улица, 1, корп. 2, стр. 4	Южный административный округ	ежедневно, 09:00–05:00	55.58493	37.635817	4.4	средние	Средний счёт:700–1500 ₽	1100.0	NaN	0	NaN	Мелитопольская улица	False

In [31]:

```
print(f"Заведений, работающих ежедневно и круглосуточно оказалось: \
{df['is_24/7'].sum()}\
что составляет {round(df['is_24/7'].sum() / len(df) * 100, 1)} % от их общего числа.")
```

Заведений, работающих ежедневно и круглосуточно оказалось: 730 что составляет 8.7 % от их общего числа.

2.4 Поиск дубликатов в данных

In [32]:

```
print(f'Количество явных дубликатов в датафрейме "df": {df.duplicated().sum()} шт.')
```

Количество явных дубликатов в датафрейме "df": 0 шт.

Проверим датафрейм на "неявные" дубликаты. Поскольку в данных часто встречаются разного рода ошибки, полученные, например, при сборе из разных БД, использовании внешних данных. Поэтому следует сделать более тщательную проверку. Сравним данные,используя дополнительный параметр subset() по столбцам 'name', 'category', 'address'. Но перед этим приведем данные в столбцах к нижнему регистру.

In [33]:

```
# приводим столбцы к нижнему регистру
for col in ['name', 'address', 'avg_bill']:
    df[col] = df[col].str.lower()

# выводим случайную строку датафрейма
df.sample(n=1, random_state=2)
```

Out[33]:

	name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_avg_bill	middle_coffee_cup	chain	seats	street	is_24/7
4073	кафе&бар	кафе	москва, площадь киевского вокзала, 1	Западный административный округ	ежедневно, 08:00–22:00	55.74334	37.565115	3.4	NaN	NaN	NaN	NaN	0	10.0	площадь Киевского Вокзала	False

In [34]:

```
print(df[df.duplicated(['name', 'category', 'address'])].count())
df[df.duplicated(['name', 'category', 'address'])]
```

```
name              2
category          2
address           2
district          2
hours             2
lat               2
lng               2
rating            2
price             0
avg_bill          0
middle_avg_bill   0
middle_coffee_cup 0
chain             2
seats             1
street            2
is_24/7           2
dtype: int64
```

Out[34]:

	name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_avg_bill	middle_coffee_cup	chain	seats	street	is_24/7
215	кафе	кафе	москва, парк ангарские пруды	Северный административный округ	ежедневно, 10:00–22:00	55.881438	37.531848	3.2	NaN	NaN	NaN	NaN	0	NaN	парк Ангарские пруды	False
1511	more pokе	ресторан	москва, волоколамское шоссе, 11, стр. 2	Северный административный округ	пн-чт 09:00–18:00; пт,сб 09:00–21:00; вс	55.806307	37.497566	4.2	NaN	NaN	NaN	NaN	1	188.0	Волоколамское шоссе	False

					09:00–18:00											

Поиск обнаружил 2 строки с неявными дубликатами. Удалим их.

In [35]:

```
df = df.drop_duplicates(subset=['name', 'address'])
print(f"После удаления неявных дубликатов, в датасете осталось: {len(df)} строк.")
```

После удаления неявных дубликатов, в датасете осталось: 8402 строк.

Проверим изменится ли количество уникальных наименований заведений, если мы произведем замену в них буквы ё на е.

In [36]:

```
df_name_e = df['name']
df_name_e = df_name_e.str.replace('ё', 'е', regex=True)
if df_name_e.nunique() - df['name'].nunique() > 0:
    print('Необходимо произвести замену в наименовании заведений буквы "ё" на "е".')
else:
    print('Замена в наименовании заведений буквы "ё" на "е" не требуется.\nОставляем букву "ё" в столбце "name"!')
```

Замена в наименовании заведений буквы "ё" на "е" не требуется.
Оставляем букву "ё" в столбце "name"!

2.4.1 Итоги поиска дубликатов в данных

- Количество явных дубликатов в датафрейме "df" не найдено.
- Неявных дубликатов найдено 2 шт из 8404 строк. Несущественно и смело удалили эти строки.
- В названии заведений оставили букву "ё", так как замена на "е" не добавила уникальных значений. Буква "ё" доставляет много хлопот в данных, но без необходимости заменять ее не стоит!

2.5 Итоги раздела предобработки данных

- Изучили пропуски в датафрейме. В итоге пропуски в столбцах 'price', avg_bill и 'seats' оставили как есть, без изменений. В данном случае отсутствие информации - тоже информация и заполнение пропусков заглушкой не обосновано. Заполнить их автоматизировано без существенных затрат времени на поиск информации по каждому заведению затруднительно.
- Пропуски в колонке middle_coffee_cup и в колонке middle_avg_bill обоснованы. Оставляем без изменений.
- Количество явных дубликатов в датафрейме "df" не найдено. Неявных дубликатов найдено и удалено несущественное количество 2 шт из 8404 строк.
- В названии заведений и улиц оставили букву "ё". В замена на "е" нет необходимости.
- добавлен столбец 'is_24_7' - содержит булево значение, с True если заведение работает "круглосуточно и ежедневно",
- добавлен столбец 'street' - в нем содержится информация с названием улицы, где расположено заведение. При этом у 28 заведений не удалось выделить адрес по причине неполной информации в адресе.

Предобработка завершена. Можно приступать к анализу данных.

3 Анализ данных

3.1 Категории заведений общественного питания

3.1.1 Распределение заведений по виду

Посмотрим какие категории заведений представлены в данных. Исследуем количество объектов общественного питания по категориям: рестораны, кофейни, пиццерии, бары и так далее.

In [37]:

```
df_category = df.groupby('category')['name'].count().reset_index()
df_category.columns = ['category', 'count']
```

```
df_category['ratio_%'] = round(df_category['count'] *100 /
df_category['count'].sum(), 3)

df_category.sort_values(by='count', ascending=False).style.background_gradient()
```

Out[37]:

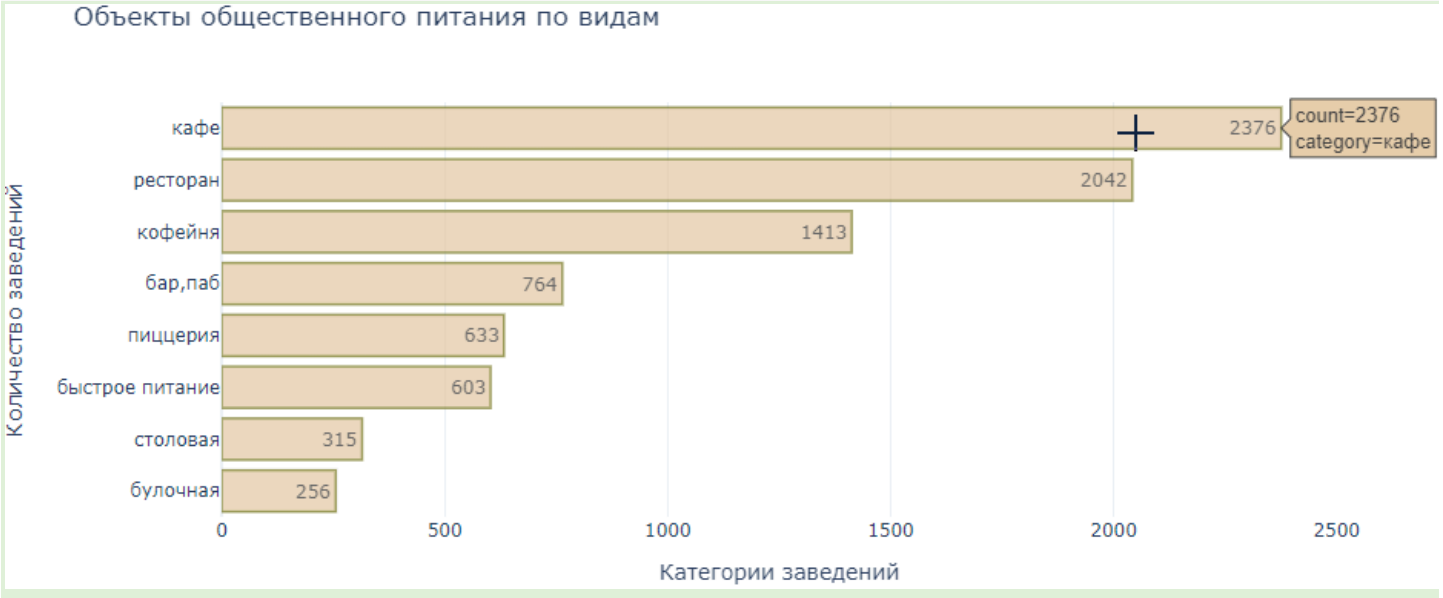
	category	count	ratio_%
3	кафе	2376	28.279000
6	ресторан	2042	24.304000
4	кофейня	1413	16.817000
0	бар,паб	764	9.093000
5	пиццерия	633	7.534000
2	быстрое питание	603	7.177000
7	столовая	315	3.749000
1	булочная	256	3.047000

3.1.2 График: Категории заведений общественного питания

Построим визуализации. Ответим на вопрос о распределении заведений по категориям.

In [38]:

```
fig = px.bar(df_category.sort_values(by='count', ascending=True), # датасет
дополнительно сортируем
             y='category', # указываем столбец с данными для оси X
             x='count', # указываем столбец с данными для оси y
             text='count', #подпись каждого столбца
             title='Объекты общественного питания по видам',
             template='plotly_white', #цвет подложки
             width=950, height=450)  #размер графика
             #color='category') # - легенда
# доп.оформление
fig.update_traces(marker_color = '#E7D0AE') # 'rgb(178,168,173)' - кастомизируем цвет
графика единый для всех столбцов

fig.update_layout(xaxis_title='Категории заведений',
                  yaxis_title='Количество заведений',
                  xaxis={'categoryorder':'total descending'})
fig.update_traces(marker_line_color='#858536',
                  marker_line_width=1.5, opacity=0.8)
fig.show()
```



Распределение самых распространенных видов заведений общественного питания в г. Москве по категориям в процентах выглядит так:

- кафе - 28.3 %
- рестораны - 24.3 %
- кофейня - 16.8 %

Численность этих заведений у каждого вида составляет менее 10% от общего числа всех заведений

- Бар, паб - 9.1 %
- пиццерия - 7.5%
- быстрое питание - 7.2 %
- столовая - 3.8 %
- булочная - 3 %

3.2 Посадочные места в заведениях общественного питания

3.2.1 Количество посадочных мест в заведениях общественного питания

Исследуем количество посадочных мест в местах по категориям: рестораны, кофейни, пиццерии, бары и так далее.

Столбец 'seats' — количество посадочных мест. Построим срез по тем данным. где указаны

Посмотрим на описательную статистику столбца 'seats' методом describe().

In [39]:

```
df['seats'].describe()
```

Out[39]:

```
count    4792.000000
mean      108.361436
std       122.841130
min         0.000000
25%        40.000000
50%        75.000000
75%       140.000000
max      1288.000000
Name: seats, dtype: float64
```

общее количество значений (count) 4792. Видим, что максимальное и минимальное значение в данных сильно отличаются. Максимальное количество посадочных мест 1288, а минимальное 0. Медиана и среднее значение отличаются почти в два раза: медиана 75 посадочных мест на заведение, а среднее значение 123 посадочных мест. Можно предположить наличие выбросов или ошибок в данных. Посмотрим на посадочные места на графике.

3.2.2 График Посадочные места в заведениях

Построим визуализацию на графике ящик с усами.

In [40]:

```
fig = px.box(df,
             x="seats",
             y="category",
             color_discrete_sequence=['#B2A8AD'],
             template='plotly_white', #цвет подложки
             width=950, height=450)# , #размер графика)
fig.update_layout(
   title={'text': "Количество посадочных мест в заведении", 'font': dict(size=14)},
   title_x=0.5,
   xaxis_title="Количество посадочных мест",
   yaxis_title="Тип заведения",
   font=dict(
       size=14
   ),
   yaxis = dict(tickfont = dict(size=14)),
   xaxis = dict(tickfont = dict(size=14))
)
fig.update_yaxes(categoryorder='array',
```

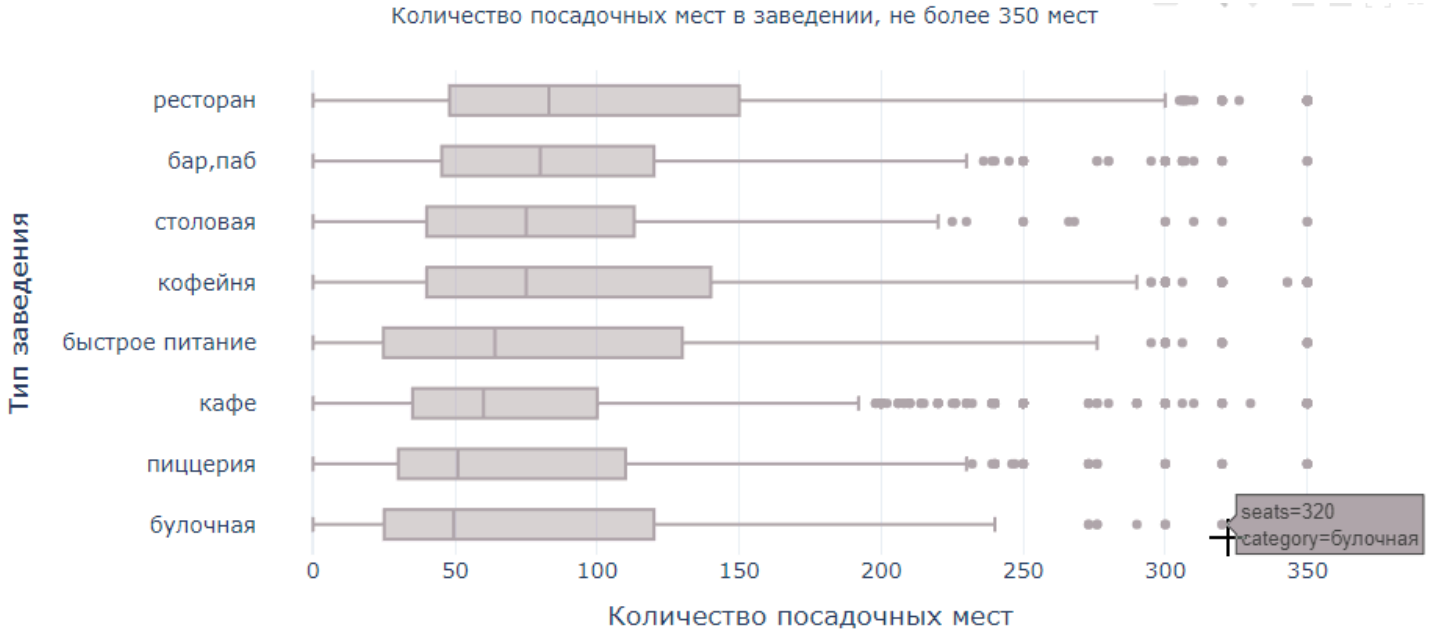
```
categoryarray=df.groupby('category').seats.median().sort_values().index)
fig.show()
```

In [41]:

Видно, что граница усов на диаграмме не превышает значение 350, остальное выбросы. В это легко поверить, так как заведения общественного питания с большим количеством посадочных мест всеже не норма, а отдельная история. Такие заведения специализируются на банкеты и мероприятия. Ограничим посадочные места 350 и посмотрим на диаграмму.

In [42]:

```
fig = px.box(df.query('seats <= 350'),
             x="seats",
             y="category",
             color_discrete_sequence=['#B2A8AD'],
             template='plotly_white', #цвет подложки
             width=950, height=450)# , #размер графика)
fig.update_layout(
   title={'text': "Количество посадочных мест в заведении, не более 350 мест",
'font': dict(size=14)},
   title_x=0.5,
   xaxis_title="Количество посадочных мест",
   yaxis_title="Тип заведения",
   font=dict(
       size=14
   ),
   yaxis = dict(tickfont = dict(size=14)),
   xaxis = dict(tickfont = dict(size=14))
)
fig.update_yaxes(categoryorder='array',
                 categoryarray=df.query('seats <=
350').groupby('category').seats.median().sort_values().index)
fig.show()
```



In [43]:

На таком графике лучше видно, что медиана по посадочным местам в заведениях от 50 до 80, а верхняя граница ящиков (или 75%) находится в диапазоне 100-150 посадочных мест. Максимальные значения, исключая выбросы(верхние границы усов) находятся в диапазоне от 190 до 300.

Ресторан и бар,паб - имеют самый высокий показатель медианы по количеству посадочных мест = 80. Ресторан также отличился по самому высокому уровню показателя третьего квартиля (75%) - 150 посадочных мест, а также самому высокому уровню максимального значения за исключением выбросов (граница верхнего уса) равного 300 посадочным местам. По этим параметрам второе место у кофейни - показатели третьего квартиля (75%) и максимального значения за исключением выбросов (граница верхнего уса) близки к показателям ресторана 140 и 290 соответственно.

Булочная и пиццерия имеют наименьшую медиану по количеству посадочных мест - 50. Кофейня и столовая имеют 72 медианных посадочных места. Заведения быстрого питания имеют 60 медианных посадочных места и широкий диапазон между первым и третьим квартилем от 25 до 130 посадочных места, а максимальное значение, за исключением выбросов 280 посадочных места.

3.2.3 График медианных значений посадочных мест на одно заведение

Визуализируем медианные значения количества посадочных мест на одно заведение общественного питания по видам.

In [44]:

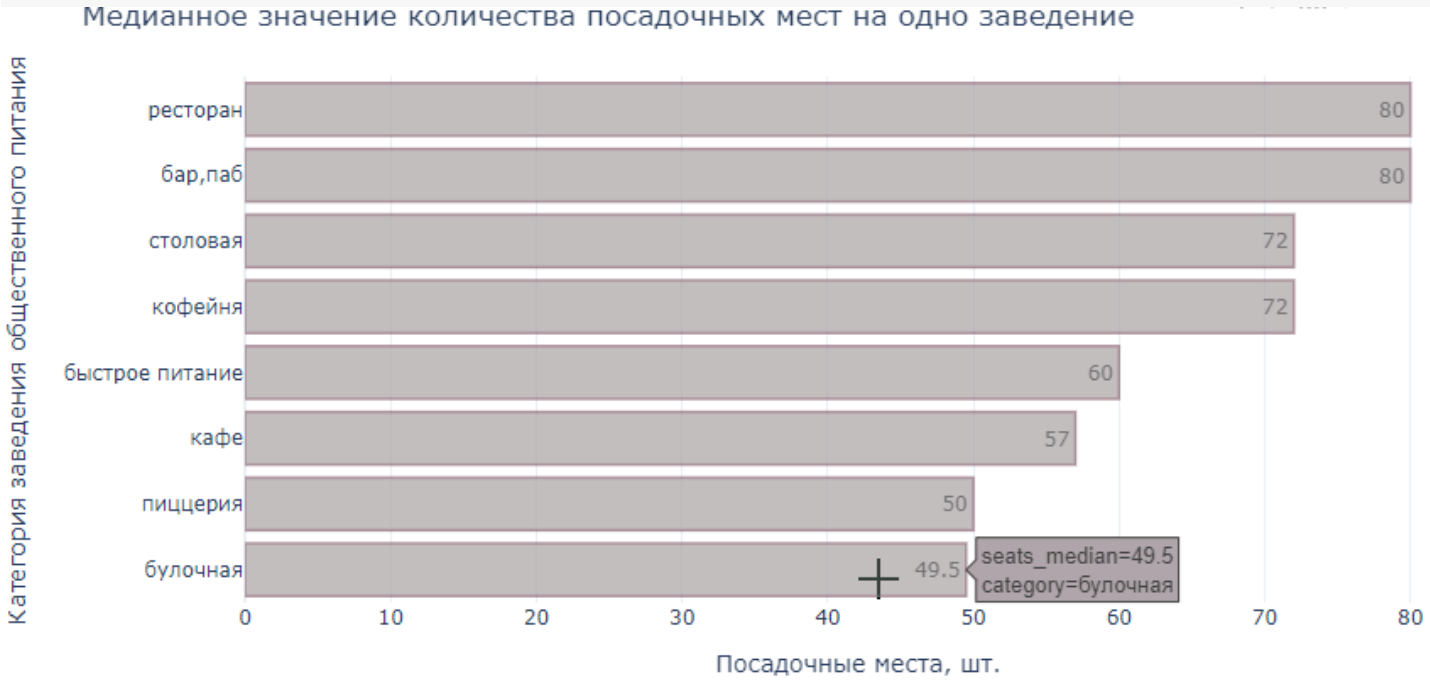
```
print(df.query('seats < 350').groupby('category') \
            .agg(seats_median=('seats', 'median')) \
            .sort_values(by='seats_median', ascending=False).reset_index())
```

	category	seats_median
0	бар,паб	80.0
1	ресторан	80.0
2	кофейня	72.0
3	столовая	72.0
4	быстрое питание	60.0
5	кафе	57.0
6	пиццерия	50.0
7	булочная	49.5

In [45]:

```
# строим столбчатую диаграмму
fig = px.bar((df.query('seats < 350').groupby('category') # загружаем данные,
сортируем по убыванию
            .agg(seats_median=('seats', 'median'))
            .sort_values(by='seats_median', ascending=True).reset_index()),
```

```
            x='seats_median', # указываем столбец с данными для оси X
            y='category', # указываем столбец с данными для оси Y
            text='seats_median', #подпись каждого столбца
            template='plotly_white', #цвет подложки
            width=950, height=450)# , #размер графика
            #color='category') # - легенда
# доп.оформление
fig.update_traces(marker_color = '#B2A8AD') # 'rgb(178,168,173)' - кастомизируем цвет графика единый для всех столбцов

fig.update_layout(title='Медианное значение количества посадочных мест на одно заведение',
                  xaxis_title='Посадочные места, шт.',
                  yaxis_title='Категория заведения общественного питания')
fig.update_traces(marker_line_color='#906C7B',
                  marker_line_width=1.5, opacity=0.75)
fig.show() # выводим график
```



3.2.4 Выводы по посадочным местам

По заведениям ресторан, кофейня и быстрое питание диапазон посадочных мест свыше 280 - 300 - это выбросы или ошибки в данных. Предположить ошибки в данных можно по наличию, например кофейни на 1,5тыс или 600 посадочных мест. Можно предполоить, что при заполнении было указано общее число посадочных мест по конкретному адресу, например на фудкоре, где десятки точек общепита имеют общие посадочные места.

Поэтому были выбраны медианные значения количества посадочных мест по виду заведения, они менее подвержены влиянию выбросов.

Медианные значения посадочных мест у заведений, шт:

- бар,паб 80
- ресторан 80
- кофейня 72
- столовая 72
- быстрое питание 60
- кафе 57
- пиццерия 50
- булочная 49

Некоторые аномальные значения, на подобии 1500 характерны конкретным адресам, может быть так заполнили число мест заведений, которые располагаются на фудкортах

3.3 Сетевые и несетевые заведения

Рассмотрим и изобразим соотношение сетевых и несетевых заведений в датасете. Каких заведений больше? Какие категории заведений чаще являются сетевыми? Исследуем данные и визуализируем результат графиком.

Столбец'chain' — число, выраженное 0 или 1, которое показывает, является ли заведение сетевым (для маленьких сетей могут встречаться ошибки):

- 0 — заведение не является сетевым
- 1 — заведение является сетевым

3.3.1 Таблица соотношения сетевых и несетевых заведений

Для удобства восприятия, число 0 и 1, в которых закодирован вид заведения(сетевое или не сетевое) заменим на явное указания вида заведения:

- "0" заменим на "не сетевое",
- "1" заменим на "сетевое".

И построим сводную таблицу chain, где сгруппируем заведения по виду- сетевое или не сетевое.

In [46]:

```
df['chain'] = df['chain'].replace([0, 1], ['не сетевое', 'сетевое'])
print(f"Замена значений [0, 1] в столбце chain на {list(df['chain'].unique())}
произошла успешно.")
```

Замена значений [0, 1] в столбце chain на ['не сетевое', 'сетевое'] произошла успешно.

In [47]:

```
chain = (
   df.pivot_table(
   index ='chain',
   values='name',
   aggfunc='count')
   .sort_values(by='name', ascending=True)
   .reset_index()
)
chain.columns = ['chain', 'count']
chain['ratio_%'] = round(chain['count'] * 100 / chain['count'].sum(), 2)
print('Таблица соотношения сетевых и не сетевых заведений')
chain.sort_values(by='count', ascending=False)
```

Таблица соотношения сетевых и не сетевых заведений

Out[47]:

	chain	count	ratio_%
1	не сетевое	5199	61.88
0	сетевое	3203	38.12

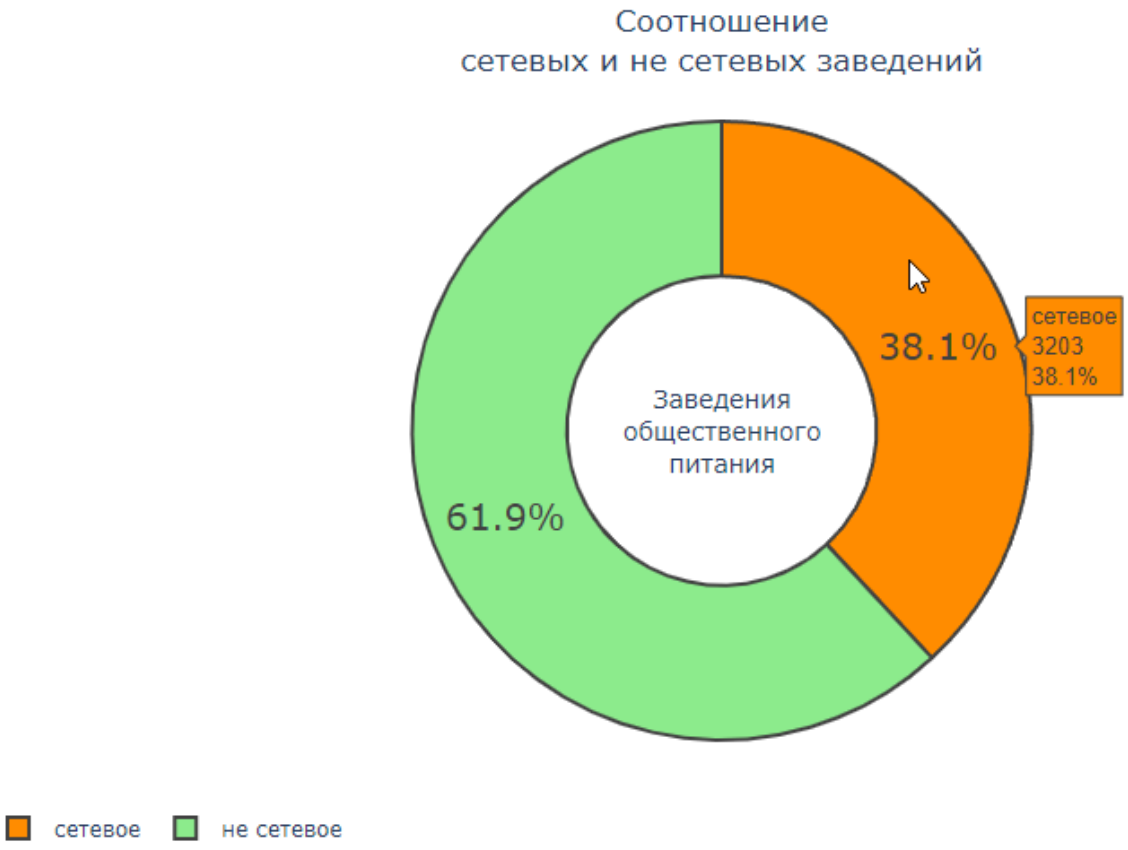
3.3.2 Диаграмма соотношения сетевых и несетевых заведений

Визуализируем полученный результат соотношения сетевых и несетевых заведений на круговой диаграмме.

In [48]:

```
   # назначаем цвета для круговой диаграммы
colors = ['darkorange','lightgreen']
#['gold','mediumturquoise','darkorange','lightgreen','cyan', '#00c600', '#66b3ff']
   # строим круговую диаграмму
fig=go.Figure()
fig.add_trace(go.Pie(
                  labels=chain['chain'],
                  values=chain['count'],
                  sort=False, # убираем встроенную сортировку по значениям
                  #pull=[0.05], # выдвигаем часть пирога
                  hole=0.5 # удаляем серединку и задаем размер
```

```
                    )
)

fig.update_layout(title='Соотношение<br>сетевых и не сетевых заведений', #
наименование графика
                  title_x=0.5, # выравниваем наименование
                  legend_orientation='h', # ориентация легенды
                  font_size=12)
   #делаем подпись внутри дырки- используем словарь т.к. аннотаций может быть много
fig.update_layout(
   annotations=[dict(text='Заведения<br>общественного<br>питания',x=0.5, y=0.5,
font_size=14, showarrow=False)])

   #дополняем настройки цвета бублика и линий
fig.update_traces(
   textposition='inside',
   #textinfo='percent+label',
   textfont_size=20,
   marker=dict(colors=colors,line=dict(color='# 000000',width=2)))

fig.show()
```



3.3.3 Выводы по диаграмме соотношения сетевых и несетевых заведений

Большинство заведений не сетевые, их доля 62%, соответственно доля сетевых заведений 38%. Нам известно, что данные по маленьким сетевым заведениям могут быть искажены. Поэтому доля сетевых заведений может быть и выше. Но и так понятно, что огромного перекоса среди заведений нет. Оба вида сетевые и не сетевые занимают большую долю на рынке заведений общественного питания г. Москвы.

3.4 Категории сетевых и несетевых заведений

Рассмотрим какие категории заведений чаще являются сетевыми. Исследуем данные и построим на их основании график.

3.4.1 Заведения по категориям и типу:сетевое/ не сетевое

In [49]:

```
# Посмотрим какие категории заведений чаще являются сетевыми
chain_type = df.groupby(['category', 'chain'])['name'].count().reset_index()
chain_type.columns = ['category', 'chain', 'count']
chain_type = chain_type.sort_values(['count', 'chain'])
chain_type
```

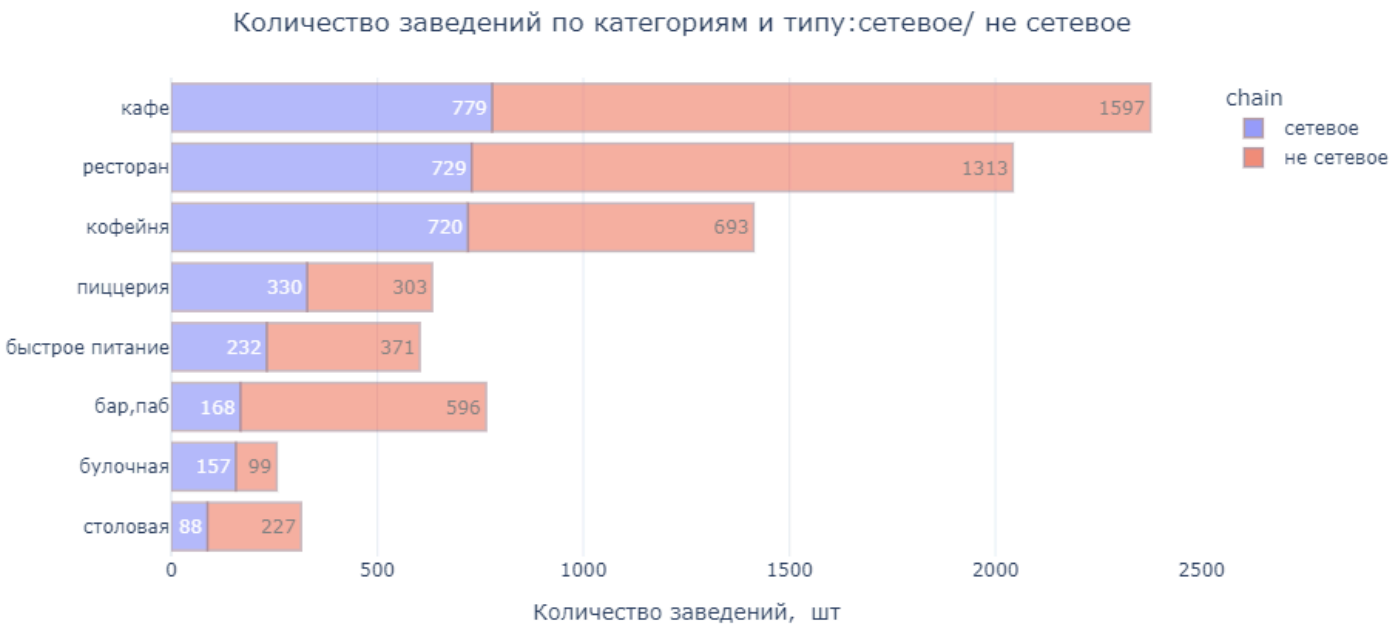
Out[49]:

	category	chain	count
15	столовая	сетевое	88
2	булочная	не сетевое	99
3	булочная	сетевое	157
1	бар,паб	сетевое	168
14	столовая	не сетевое	227
5	быстрое питание	сетевое	232
10	пиццерия	не сетевое	303
11	пиццерия	сетевое	330
4	быстрое питание	не сетевое	371
0	бар,паб	не сетевое	596
8	кофейня	не сетевое	693
9	кофейня	сетевое	720
13	ресторан	сетевое	729
7	кафе	сетевое	779
12	ресторан	не сетевое	1313
6	кафе	не сетевое	1597

3.4.2 Диаграмма количество заведений по категориям и типу: сетевое/ не сетевое

In [50]:

```
fig = px.bar(chain_type, x='count', y='category',
             text='count', template='plotly_white',
             opacity=0.7,
             color='chain', width=950, height=450, #размер графика
             category_orders={'chain': ['сетевое', 'не сетевое']}
            )
# оформляем график
#color = ['#E7D0AE', '#906C7B']
#fig.update_traces(marker_color=color) # 'rgb(178,168,173)' - кастомизируем цвет
графика единый для всех столбцов
fig.update_layout(title='Количество заведений по категориям и типу:сетевое/ не
сетевое', title_x=0.5,
                  xaxis_title='Количество заведений,  шт',
                  yaxis_title='')
fig.update_traces(marker_line_color='#906C7B',
                  marker_line_width=1.5, opacity=0.67)
fig.show()
```



3.4.3 Процент сетевых заведений по каждому типу.

Посмотрим нагляднее на графике, который покажет процент сетевых заведений по каждому типу. Для этого создадим таблицу только с сетевыми заведениями и добавим в нее столбец с процентами, которые показывают какую долю этого типа заведения составляют сетевые. Построим график.

In [51]:

```
category_count = df.pivot_table(
    index='category',
    values='name',
    aggfunc='count').sort_values(by='name', ascending=False)

chain_1 = df.query('chain == "сетевое"').pivot_table(
    index='category',
    values='name',
    aggfunc='count').sort_values(by = 'name', ascending=False)
chain_1 = category_count.merge(chain_1, how='left', on='category').reset_index()
chain_1.columns = ['category', 'count', 'chain']
chain_1['ratio_%'] = round(chain_1['chain'] * 100 / chain_1['count'], 3)
print('Процент сетевых заведений по каждому типу')
chain_1
```

Процент сетевых заведений по каждому типу

Out[51]:

	category	count	chain	ratio_%
0	кафе	2376	779	32.786
1	ресторан	2042	729	35.700
2	кофейня	1413	720	50.955
3	бар,паб	764	168	21.990
4	пиццерия	633	330	52.133
5	быстрое питание	603	232	38.474
6	столовая	315	88	27.937
7	булочная	256	157	61.328

3.4.4 График Рейтинг сетевых заведений по типу, %

In [52]:

```
#colors = ['#58B19F', '#BDC581','#CAD3C8', '#a4b0be', '#ffb8b8', '#c7ecee','#c8d6e5',
'#dff9fb']
```

```
plt.figure(figsize=(14,4))
sns.barplot(y='category', x='ratio_%', data=chain_1.sort_values(by='ratio_%',
ascending=False), palette='mako', alpha=0.6)
            #palette=colors) альтернативная палитра цветов

#plt.grid(True)
plt.title('Рейтинг сетевых заведений по типу, %',
          fontsize = 14, fontweight ='bold')
plt.xlabel('Доля сетевых заведений, %')
plt.ylabel('')
plt.show()
```

Про булочную и пиццерию любопытно, такое распространение связываю с нежеланием готовить в каждой булочной булки)

Плюс там где не нужна индивидуальность, а нужна эффективность (через стандартизацию) там больше сетей. Выпечка, пиццы, кофе, фастфуд - это зачастую стандартный продукт, а вот бары, пабы, рестораны наиболее индивидуальны, большинство как ручная работа.

3.4.5 Лидеры среди сетевых заведений

Булочная, пиццерия и кофейня чаще всего является сетевым заведением из всех представленных типов заведений.

Сетевыми у булочных являются - 60% от всех булочных заведений г. Москвы. Сетевых пиццерий 52%, а кофеен 51%.

Сеть быстрого питания имеет 38% сетевых заведений, остальные заведения быстрого питания являются не сетевыми. Сетевых ресторанов 36%, а кафе 33%. Сетевых столовых еще меньше - 28%. Самый малый показатель сетевых заведений у баров,пабов -22%.

3.5 Самые популярные сети Москвы

3.5.1 Рейтинг самых популярных сетевых заведений г. Москвы

Сгруппируем данные по названиям заведений и найдем топ-15 популярных сетей в Москве. Под популярностью понимается количество заведений этой сети в регионе.

In [53]:

```
top_chain = df.query('chain == "сетевое"')
top_chain = top_chain.groupby('name').agg({'category' : pd.Series.mode, 'rating':
'median', 'district' : 'count'})
top_chain = top_chain.rename(columns={'district':'count'})
top_chain = top_chain.sort_values('count', ascending = False).reset_index().head(15)

top_chain.style.background_gradient()
```

Out[53]:

	name	category	rating	count
0	шоколадница	кофейня	4.200000	120

1	домино'с пицца	пиццерия	4.200000	76
2	додо пицца	пиццерия	4.300000	74
3	one price coffee	кофейня	4.200000	71
4	яндекс лавка	ресторан	4.000000	69
5	cofix	кофейня	4.100000	65
6	prime	ресторан	4.200000	50
7	хинкальная	кафе	4.400000	44
8	кофепорт	кофейня	4.200000	42
9	кулинарная лавка братьев караваевых	кафе	4.400000	39
10	теремок	ресторан	4.100000	38
11	чайхана	кафе	4.100000	37
12	cofefest	кофейня	4.050000	32
13	буханка	булочная	4.400000	32
14	му-му	кафе	4.300000	27

In [54]:

```
print(f"Всего в топ-15 рейтинга самых популярных сетевых заведений\
г. Москвы вошли {top_chain['count'].sum()} заведений")
```

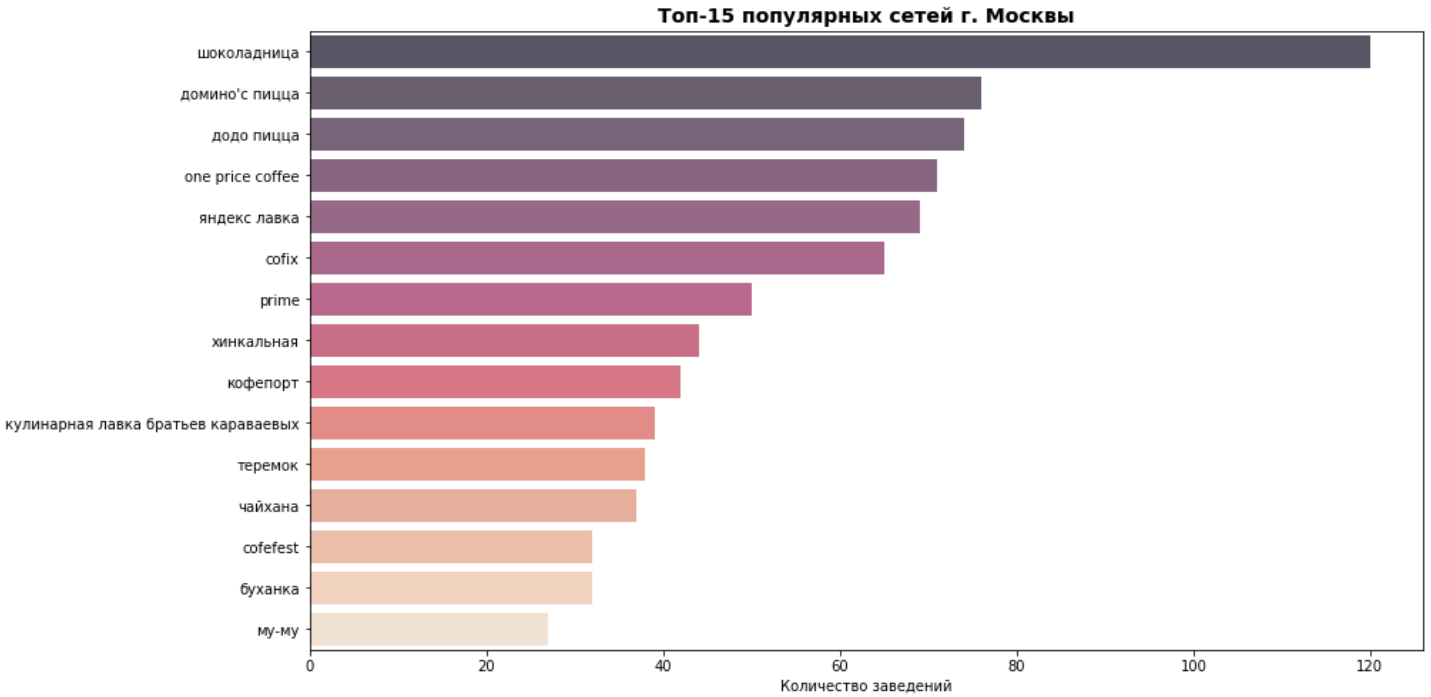
Всего в топ-15 рейтинга самых популярных сетевых заведенийг. Москвы вошли 816 заведений

3.5.2 Диаграмма топ-15 популярных сетей г. Москвы

Построим столбчатую диаграмму самых популярных сетевых заведений в г. Москва. Популярность определена по количеству заведений в сети.

In [55]:

```
plt.figure(figsize=(14,8))
sns.barplot(y='name', x='count', data=top_chain.sort_values(by='count',
ascending=False), palette='rocket', alpha=0.7)
           #palette=colors) palette='mako' альтернативная палитра цветов

#plt.grid(True)
plt.title('Топ-15 популярных сетей г. Москвы',
        fontsize = 14, fontweight ='bold')
plt.xlabel('Количество заведений')
plt.ylabel('')
plt.show()
```



В целом в топе много заведений, ориентированных на невысокий средний чек и большую проходимость
Можно было бы ещё подметить, что появление `Яндекс.Лавки` очень странное, ведь это не совсем ресторан, кафе)
Самая популярная сеть - Шоколадница - по данным из датасета их аж 120 на Москву. По данным из открытых источников: сеть кофеен «Шоколадница» — одна из крупнейших и самых динамично развивающихся компаний в сфере ресторанного бизнеса в Москве, регионах России и странах СНГ.
Второе место разделили между собой Домино'с пицца и Додо пицца 76 и 74 заведений у каждого соответственно. Это сети пиццерий.
Третье место с небольшим отрывом от пиццерий заняла сеть one price coffee. Замыкает топ-15 кафе Му-Му. По данным датасета в Москве у этого кафе есть 27 сетевых заведений.
В рейтинге все названия знакомы. В топ-15 вошло заведение яндекс лавка. Посмотрим к какой категории оно относится.

In [56]:

```
print('В датафрейме яндекс лавка относится к категории:')
print(df.query('name == "яндекс лавка"')['category'].unique())
```

В датафрейме яндекс лавка относится к категории:
['ресторан']

В датафрейме яндекс лавка относится к категории:'ресторан'. По факту эти заведения не являются классическим заведением общепита. Особенность работы сервиса яндекс лавка — в использовании небольших локальных складов формата «даркстор». Люди заказывают готовую еду с доставкой по адресу.
Это или ошибка сбора данных или такая категория не была выделена отдельно и ее отнесли в рестораны.
Правильнее было бы выделить для таких заведений отдельную категорию.
Посмотрим, какие категории заведений попали в топ-15.

3.5.3 Категории заведений в топ-15

In [57]:

```
category = list(df['category'].unique())
print('Количество заведений в топ-15 по категориям')
category_top =
top_chain.groupby(['category']).agg({'name':'count'}).sort_values(by='name',
ascending=False)
```

Количество заведений в топ-15 по категориям

In [58]:

```
category_top
```

Out[58]:

	name
category	
кофейня	5
кафе	4
ресторан	3
пиццерия	2
булочная	1

Кофейни и кафе - эти две категории самые популярные. Булочных всего одна, три ресторана. Пиццерий две.
Эти категории не попали в топ-15:

- 'бар,паб',
- 'быстрое питание',
- 'столовая'

3.6 Заведения по административным районам г. Москвы

Посмотрим какие административные районы Москвы присутствуют в датасете. Сгруппируем таблицу по округам с общим количеством заведений.

In [59]:

```
district_9 = df.groupby(['district']).agg({'name': 'count'}).sort_values('name',
ascending=False).reset_index()
district_9 = district_9.rename(columns={'name': 'count'})
district_9['ratio_%'] = round(district_9['count'] * 100 / district_9['count'].sum(),
3)
print('Количество заведений в каждом административном районе г. Москвы:')
district_9.style.background_gradient()
```

Количество заведений в каждом административном районе г. Москвы:

Out[59]:

	district	count	ratio_%
0	Центральный административный округ	2242	26.684000
1	Северный административный округ	898	10.688000
2	Южный административный округ	892	10.617000
3	Северо-Восточный административный округ	890	10.593000
4	Западный административный округ	850	10.117000
5	Восточный административный округ	798	9.498000
6	Юго-Восточный административный округ	714	8.498000
7	Юго-Западный административный округ	709	8.438000
8	Северо-Западный административный округ	409	4.868000

Всего представлено 9 административных округов Москвы.

3.6.1 Диаграмма с количеством заведений каждой категории по районам Москвы

Отобразим общее количество заведений и количество заведений каждой категории по районам на одном графике. Сгруппируем для начала таблицу по округам с категорией, рейтингом и количеством заведений.

In [60]:

```
district = df.groupby(['district', 'category']).agg({'rating': 'median', 'name':
'count'})
district = district.sort_values('rating', ascending=False).reset_index()
district = district.rename(columns={'name': 'count'})
print('Несколько строк из таблицы с группировкой всех заведений по округам')
district.sample(n=2, random_state=1)
```

Несколько строк из таблицы с группировкой всех заведений по округам

Out[60]:

	district	category	rating	count
19	Северо-Восточный административный округ	ресторан	4.3	182
55	Юго-Западный административный округ	кафе	4.2	238

Названия у округов длинные. Для удобства визуализации, сократим названия округов. Для Москвы аббревиатуры округов повсеместно используются.

- 'Центральный административный округ' - переименуем в 'ЦАО',
- 'Западный административный округ' - переименуем в 'ЗАО',
- 'Северо-Западный административный округ' - переименуем в 'СЗАО',
- 'Юго-Западный административный округ' - переименуем в 'ЮЗАО',
- 'Южный административный округ' - переименуем в 'ЮАО',
- 'Юго-Восточный административный округ' - переименуем в 'ЮВАО',
- 'Северный административный округ' - переименуем в 'САО',
- 'Восточный административный округ' - переименуем в 'ВАО',
- 'Северо-Восточный административный округ' - переименуем в 'СВАО'.

In [61]:

```
district['district'] = district['district'].replace(['Центральный административный
округ',
```

```
'Западный административный округ',
'Северо-Западный административный округ',
'Юго-Западный административный округ',
'Южный административный округ',
'Юго-Восточный административный округ',
'Северный административный округ',
'Восточный административный округ',
'Северо-Восточный административный округ'],
['ЦАО', 'ЗАО', 'СЗАО', 'ЮЗАО', 'ЮАО', 'ЮВАО', 'САО', 'ВАО', 'СВАО'])
print('Административные районы г. Москвы в датасете после переименования:')
list(district['district'].unique())
```

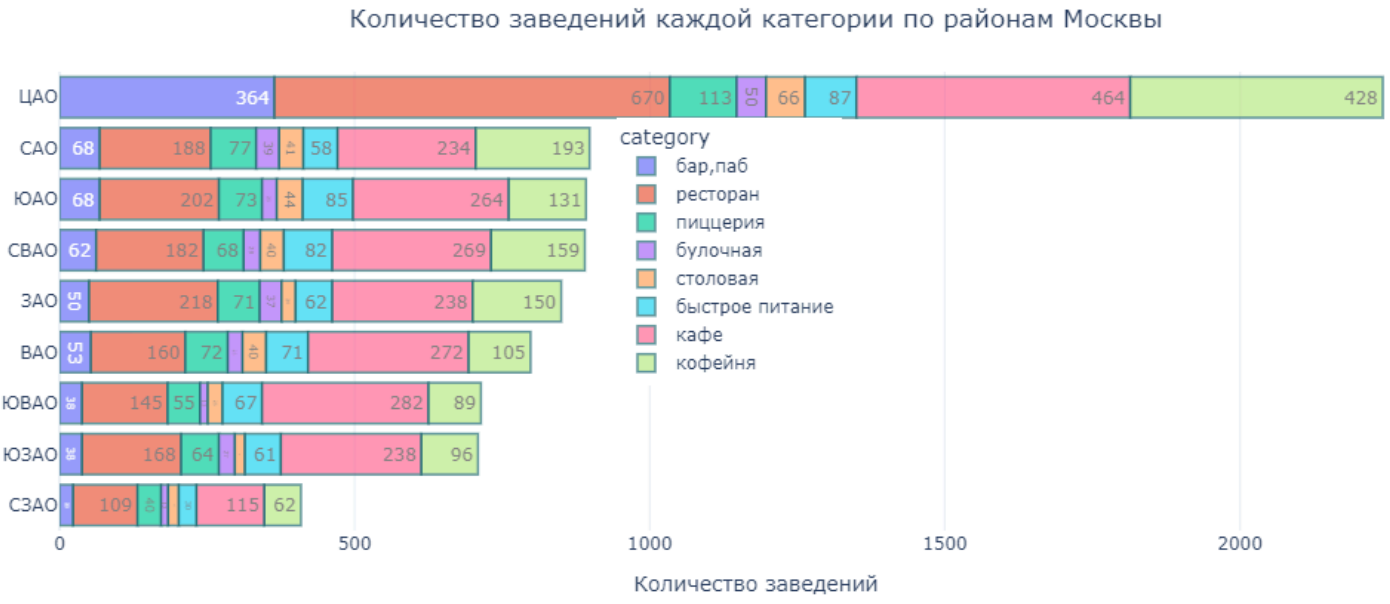
Административные районы г. Москвы в датасете после переименования:

Out[61]:

['ЦАО', 'ЗАО', 'СЗАО', 'ЮЗАО', 'ЮАО', 'ЮВАО', 'САО', 'ВАО', 'СВАО']

In [62]:

```
fig = px.bar(district,
             x='count',
             y='district',
             text='count',
             template='plotly_white',
             color='category',
             width=1100, height=450, #размер графика
            )
fig.update_layout(title = 'Количество заведений каждой категории по районам Москвы',
title_x=0.5,
                  xaxis_title = 'Количество заведений',
                  yaxis_title = '',
                  yaxis={'categoryorder': 'total ascending'}
)
fig.update_layout(legend=dict(yanchor="top", y=0.9, xanchor="left", x=0.4))
fig.update_traces(marker_line_color='rgb(18,98,107)',
                  marker_line_width=1.5, opacity=0.67)
fig.show()
```



3.6.2 Вывод по диаграмме количества заведений каждой категории по районам Москвы

Всего рассмотрено заведений в 9-ти районах г. Москвы: 'ЦАО', 'ЗАО', 'СЗАО', 'ЮЗАО', 'ЮАО', 'ЮВАО', 'САО', 'ВАО', 'СВАО'. ЦАО лидирует среди районов по количеству заведений- в этом округе аж 2242 заведений, почти 27% от всех рассматриваемых заведений. В ЦАО больше всего ресторанов, кафе, кофеен и баров/пабов, в остальных районах также много ресторанов, кафе, кофеен
В остальных районах показатели заведений колеблются в пределах 10,7-8.4%, за исключением СЗАО, в котором находится всего 4,9% от всех рассматриваемых в исследовании заведений г. Москвы.

3.6.3 Распределение среднего рейтинга заведений г. Москвы по категориям

Посмотрим как распределяется средний рейтинг заведений г. Москвы по категориям. Для этого сгруппируем данные в таблицу ratings.

In [63]:

```
ratings = (
    df.groupby('category')
    .agg(mean_rating=('rating', 'mean'))
    .sort_values(by='mean_rating', ascending=False)
    .reset_index()
)
print('Средний рейтинг заведений г. Москвы по категориям')
ratings.style.background_gradient()
```

Средний рейтинг заведений г. Москвы по категориям

Out[63]:

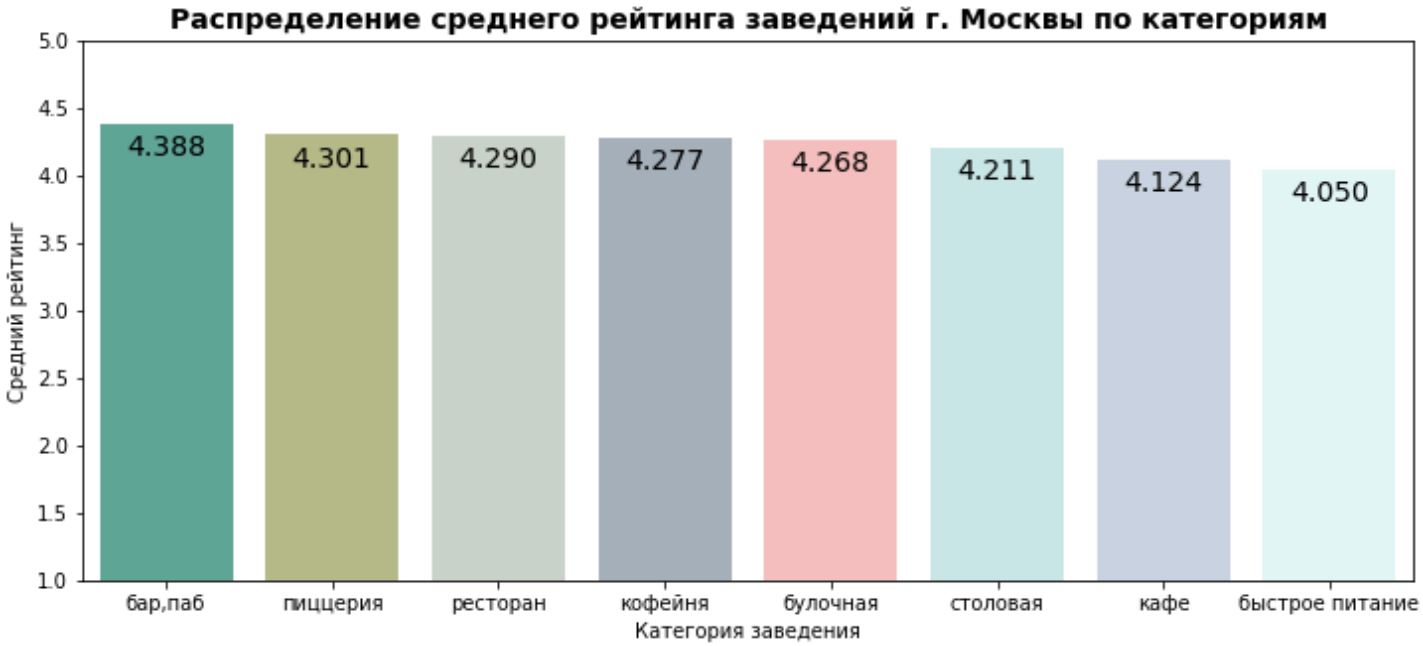
	category	mean_rating
0	бар,паб	4.387696
1	пиццерия	4.301264
2	ресторан	4.290402
3	кофейня	4.277282
4	булочная	4.268359
5	столовая	4.211429
6	кафе	4.124285
7	быстрое питание	4.050249

3.6.4 Диаграмма распредениея среднего рейтинга заведений г. Москвы по категориям

Визуализируем распределение средних рейтингов по категориям заведений.

In [64]:

```
colors = ['#58B19F', '#BDC581','#CAD3C8', '#a4b0be', '#ffb8b8', '#c7ecee','#c8d6e5',
'#dff9fb']

plt.figure(figsize=(12, 5))
plt.ylim(1, 5)
splot = sns.barplot(x="category", y="mean_rating", #, hue="sex",
                    data=ratings, palette=colors)
#добавляем подписи к столбцам со значением
for p in splot.patches:
    splot.annotate(format(round(p.get_height(), 3), '.3f'),
                   (p.get_x() + p.get_width() / 2., p.get_height()),
                   ha='center', va='center',
                   size=14,
                   xytext=(0, -12),
                   textcoords='offset points')
plt.title('Распределение среднего рейтинга заведений г. Москвы по категориям',
fontsize = 14, fontweight ='bold')
plt.xlabel('Категория заведения')
plt.ylabel('Средний рейтинг');
```



Видим, что сильных различий у усреднённых рейтингах в разных типах общепита нет.
Самый высокий рейтинг у заведений -бар/паб - 4.388. Самый маленький рейтинг у ресторанов быстрого питания - 4.050.

3.6.5 Фоновая картограмма (хороплет) со средним рейтингом заведений каждого района

Построим фоновую картограмму (хороплет) со средним рейтингом заведений каждого района. Границы районов Москвы, которые встречаются в датасете, хранятся в файле admin_level_geomap.geojson (скачать файл для локальной работы).

In [65]:

```
rating_ao = df.groupby('district', as_index=False)['rating'].agg('mean').round(3)
rating_ao
```

Out[65]:

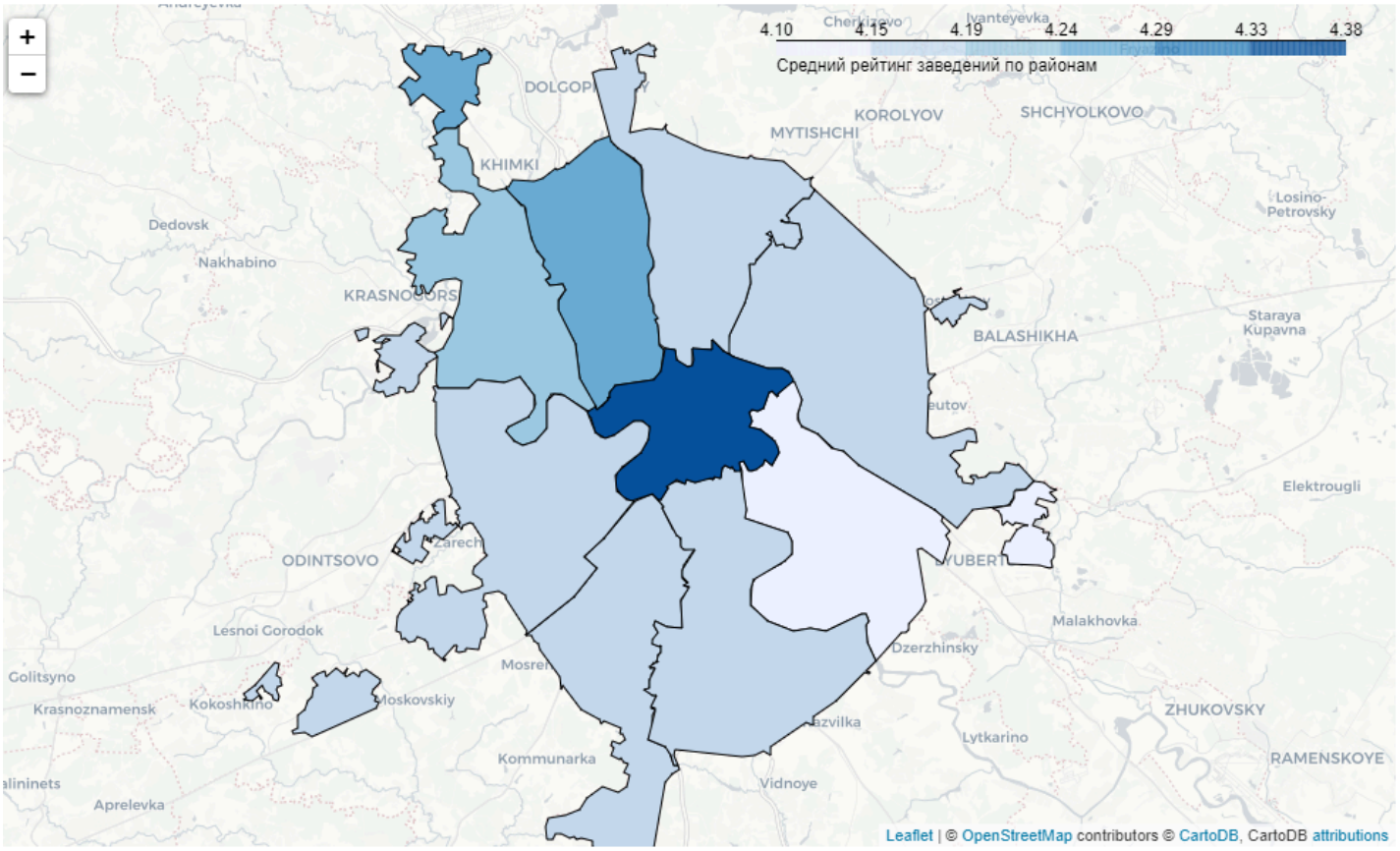
	district	rating
0	Восточный административный округ	4.174
1	Западный административный округ	4.182
2	Северный административный округ	4.241
3	Северо-Восточный административный округ	4.148
4	Северо-Западный административный округ	4.209
5	Центральный административный округ	4.378
6	Юго-Восточный административный округ	4.101
7	Юго-Западный административный округ	4.173
8	Южный административный округ	4.184

In [66]:

```
# загружаем JSON-файл с границами округов Москвы
try:
    state_geo = '/datasets/admin_level_geomap.geojson'
except:
    state_geo = 'https://code.s3.yandex.net/data-analyst/admin_level_geomap.geojson'

# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаём карту Москвы
m = Map(location=[moscow_lat, moscow_lng], zoom_start=10, tiles='Cartodb Positron')
```

```
# создаём хороплет с помощью конструктора Choropleth и добавляем его на карту
Choropleth(
    geo_data=state_geo,
    data=rating_ao,
    columns=['district', 'rating'],
    key_on='feature.name',
    fill_color='Blues',
    fill_opacity=1,
    legend_name='Средний рейтинг заведений по районам',
).add_to(m)

# выводим карту
m
```

Out[66]:



Самый высокий рейтинг в заведениях в Центральном административном округе - 4.38. Самый низкий - в Юго-Восточном административном округе - 4.1.

3.6.6 Все заведения на карте

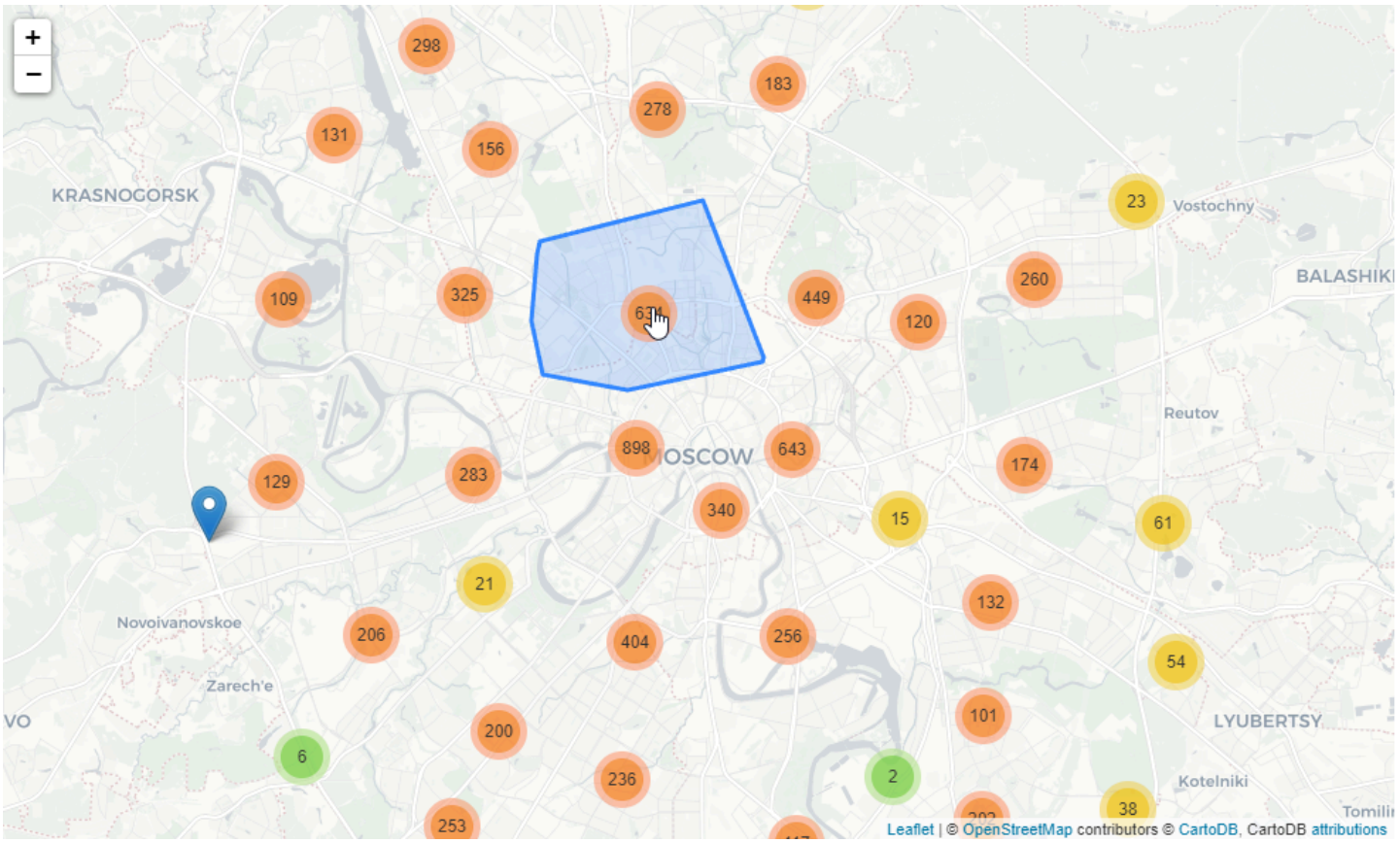
Отобразим все заведения датасета на карте с помощью кластеров средствами библиотеки folium.

In [67]:

```
# moscow_lat - широта центра Москвы, moscow_lng - долгота центра Москвы
moscow_lat, moscow_lng = 55.751244, 37.618423

# создаём карту Москвы
m = Map(location=[moscow_lat, moscow_lng], zoom_start=10, tiles="Cartodb Positron")
# создаём пустой кластер, добавляем его на карту
marker_cluster = MarkerCluster().add_to(m)

# пишем функцию, которая принимает строку датафрейма,
# создаёт маркер в текущей точке и добавляет его в кластер marker_cluster
def create_clusters(row):
```

```
    Marker(
        [row['lat'], row['lng']],
        popup=f"{row['name']} {row['rating']}",
    ).add_to(marker_cluster)

# применяем функцию create_clusters() к каждой строке датафрейма
df.apply(create_clusters, axis=1)

# выводим карту
m
```

Out[67]:



3.6.7 Итоги раздела

Выделяется из всех административных округов ЦАО. И высоким количеством заведений, и высоким средним рейтингом заведений.

Из открытых источников можно узнать, что площадь центрального округа составляет всего 6% от площади Москвы, но для многих туристов именно это и есть та самая Москва, которую они хотят увидеть.

Центральный округ – это наиболее «нафаршированный» из всех округов столицы:

- здесь сосредоточена большая часть памятников культуры Москвы. Это памятники всевозможным политическим деятелям, музеи и церкви. На территории ЦАО расположены 237 культовых учреждений разных конфессий и 8 из 10 московских монастырей.
    - здесь находятся 6 из 9 московских вокзалов – Курский, Казанский, Ярославский, Ленинградский, Белорусский и Павелецкий.
    - это один из округов, наиболее обеспеченных станциями метро (67 станций, 38% от общего количества). Кроме того, через ЦАО проходят почти все ветки метро, за исключением Каховской и Бутовской.
    - здесь расположены практически все правительственные учреждения (Кремль, Дом Правительства РФ, Совет Федераций, Госдума и большинство министерств)
    - здесь находятся тысячи развлекательных учреждений – баров, ресторанов, клубов, а также магазинов и торговых центров.

3.7 Топ-15 улиц по количеству заведений

Найдем топ-15 улиц по количеству заведений. Построим график распределения количества заведений и их категорий по этим улицам. Попробуйте проиллюстрировать эту информацию одним графиком. Сгруппируем для начала таблицу по улицам и количеству заведений, средним рейтингом этих заведений. Заведения не поделены на категории.

In [68]:

```
street = df.groupby('street').agg({'rating': 'median', 'name': 'count'})
street = street.rename(columns={'name': 'count'})
street = street.sort_values('count', ascending=False).reset_index().head(15)
street['ratio_%'] = round(street['count'] *100 / street['count'].sum(), 3)
print('Группировкой всех заведений по улицам')
street.style.background_gradient()
```

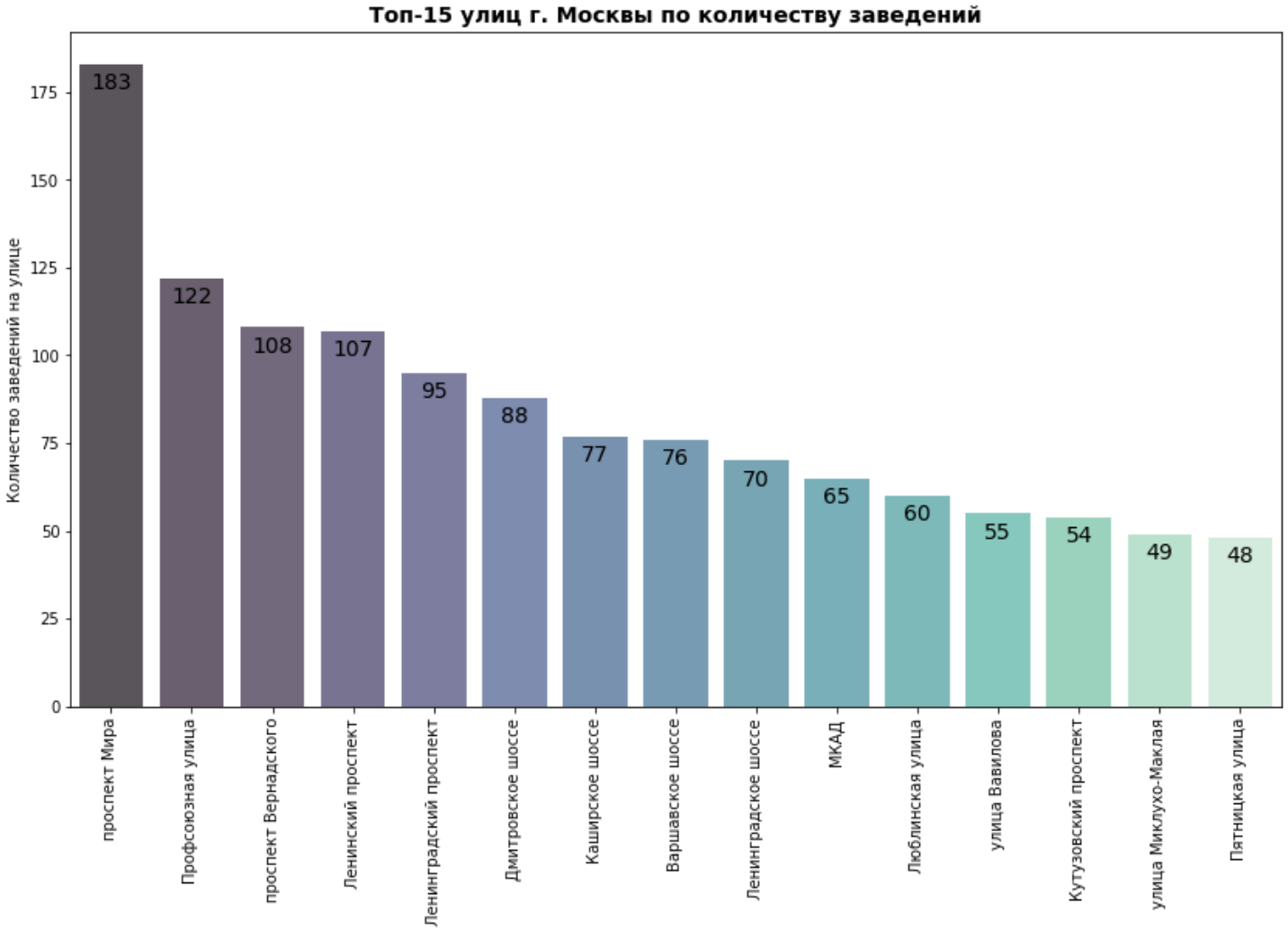
Группировкой всех заведений по улицам

Out[68]:

	street	rating	count	ratio_%
0	проспект Мира	4.200000	183	14.558000
1	Профсоюзная улица	4.300000	122	9.706000
2	проспект Вернадского	4.300000	108	8.592000
3	Ленинский проспект	4.300000	107	8.512000
4	Ленинградский проспект	4.300000	95	7.558000
5	Дмитровское шоссе	4.200000	88	7.001000
6	Каширское шоссе	4.200000	77	6.126000
7	Варшавское шоссе	4.200000	76	6.046000
8	Ленинградское шоссе	4.300000	70	5.569000
9	МКАД	4.100000	65	5.171000
10	Люблинская улица	4.300000	60	4.773000
11	улица Вавилова	4.300000	55	4.375000
12	Кутузовский проспект	4.350000	54	4.296000
13	улица Миклухо-Маклая	4.300000	49	3.898000
14	Пятницкая улица	4.400000	48	3.819000

In [69]:

```
plt.figure(figsize=(14,8))
splot = sns.barplot(x='street', y='count', data=street.sort_values(by='count',
ascending=False), palette='mako', alpha=0.7
                    )
#добавляем подписи к столбцам со значением
for p in splot.patches:
   splot.annotate(format(round(p.get_height(), 3), '.0f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha='center', va='center',
                  size=14,
                  xytext=(0, -12),
                  textcoords='offset points')
plt.title('Топ-15 улиц г. Москвы по количеству заведений',
        fontsize = 14, fontweight ='bold')
plt.xlabel('')
plt.xticks(rotation=90)
plt.ylabel('Количество заведений на улице')
plt.show()
```



запишем список улиц из топ-15 в переменную street_list

In [70]:

```
street_list = street['street']
print('Список топ-15 улиц г. Москвы, на которых расположено наибольшее число заведений')
list(street_list)
```

Список топ-15 улиц г. Москвы, на которых расположено наибольшее число заведений

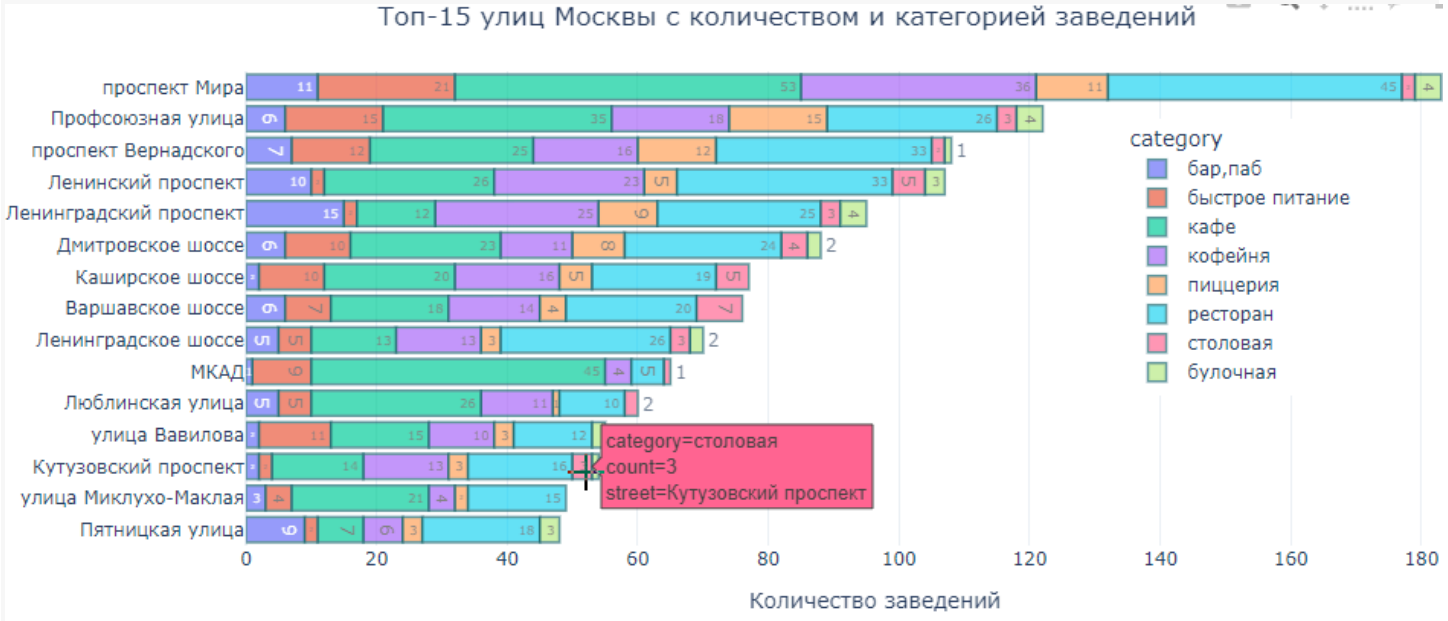
Out[70]:

['проспект Мира',
 'Профсоюзная улица',
 'проспект Вернадского',
 'Ленинский проспект',
 'Ленинградский проспект',
 'Дмитровское шоссе',
 'Каширское шоссе',
 'Варшавское шоссе',
 'Ленинградское шоссе',
 'МКАД',
 'Люблинская улица',
 'улица Вавилова',
 'Кутузовский проспект',
 'улица Миклухо-Маклая',
 'Пятницкая улица']

In [71]:

```
fig = px.bar(df.query("street in @street_list")
```

```
            .groupby(['street', 'category'])
            .agg(count=('category', 'count')).reset_index(),
            x='count',
            y='street',
            text='count',
            template='plotly_white',
            color='category',
            width=1100, height=450, #размер графика
           )
fig.update_layout(title = 'Топ-15 улиц Москвы с количеством и категорией заведений',
title_x=0.5,
                  xaxis_title = 'Количество заведений',
                  yaxis_title = '',
                  yaxis={'categoryorder': 'total ascending'}
)
fig.update_layout(legend=dict(yanchor="top", y=0.9, xanchor="left", x=0.7))
fig.update_traces(marker_line_color='rgb(18,98,107)',
                  marker_line_width=1.5, opacity=0.67)
fig.show()
```



Любопытные отличия некоторых улиц по составу заведений, так например на МКАДе не так много баров-пабов, что в принципе логично

Обращаю внимание, что дело в длине этих улиц, часть из них выдают такое большое число заведений именно за счёт длины

Посмотрим на протяженность нескольких улиц из ТОП-15. Информация взята из открытых источников:

Проспект Мира - Общая длина проспекта составляет около 9 км. Одна из самых известных и длинных улиц в городе Москва. Этот проспект начинается от Каланчевской площади и тянется на север вплоть до Останкинского района, превращаясь в Останкинскую улицу.

- Профсоюзная улица — ее протяженность составляет 14 км, ее принято считать самой длинной пешеходной улицей.
- Проспект Вернадского имеет протяженность 8 км.
- Ленинский проспект - самая длинная улица, названная в честь вождя мирового пролетариата, протяженность проспекта - 16 км.
- Ленинградский проспект - общая протяжённость улицы (шоссе) от Ленинградского проспекта до Шереметьевского шоссе — 19,7 км, из них 15,2 км приходится на Москву.
- Дмитровское шоссе - протяженность 15.5 км или 12 км согласно маршруту созданному по яндекс картам, от дома 1 на Дмитровской шоссе (м Дмитровская) до развязки МКАД.

- Каширское шоссе - дорога, расположенная в ЮАО. Ее протяженность (та часть, что располагается в Москве) составляет 10,5 км.
- Варшавское шоссе - его протяженность — 19,4 км.
- Ленинградское шоссе - протяженность проспекта составляет 5.6 км.
- Улица Миклухо-Маклая - протяженность 3.6 км.
- МКАД - длина окружной дороги составляет 108,9 км.
- Люблинская улица - 8,2 км.
- Улица Пятницкая одна из старейших улиц Замоскворечья Центрального округа Москвы протяженность - 1.8 км.
- По информации из интернета о Москве - средняя длина улиц в Москве — 934 метра.

Улицы из ТОП-15 намного превышают среднюю длину улиц Москвы, даже если учесть, что правильнее было бы найти медианную длину.
Вероятно поэтому и попали эти улицы в Топ-15. Для дальнейшего анализа пригодилась бы информация о протяженности улиц. Но мы такой не располагаем. Но можем предположить, что чем длиннее улица, тем больше на ней находится заведений. МКАД - не самая показательная улица - это авто-дорога без пешеходного трафика, и заведения там в основном расположены в торговых ценрах или заправках.

3.7.1 Вывод Топ-15 улиц Москвы с количеством и категорией заведений

Больше всего заведений расположено на проспекте Мира. Преобладают кафе, рестораны и кофейни.
Вторая по популярности улица - Профсоюзная улица, преобладают рестораны и кафе. На МКАД много кафе и нет пиццерий и булочных, а на Пятницкой улице и улице Вавилова нет столовых. На улице Миклухо-Маклая нет булочных, столовых.
Пятницкая улица - самая короткая в ТОП-15.
На улицах с большей протяженностью заведений расположено больше.

3.8 Взаимосвязь среднего чека и рейтинга заведения

Посмотрим. есть ли взаимосвязь между размером среднего чека и рейтингом заведения.

3.8.1 Добавим категории рейтинга

Чтобы сравнить - разобьем рейтинг на категории, рейтинг:

- 0 до 3.0 (не включая)- критически низкий
- от 3.0 до 4.0 (не включая) - низкий рейтинг
- 4.0 до 4.3 (включая) - средний рейтинг
- свыше 4.3 - высокий рейтинг.

Добавим категории в датафрейм в столбец 'rating_category'

In [72]:

```
# разобъем рейтинг на категории (значения-низкий, средний, высокий) и добавляем
столбец тип рейтинга
def cat_rating(row):
    if row['rating'] < 3:
        return 'критически низкий рейтинг'
    elif 3.0 <= row['rating'] < 4.0:
        return 'низкий рейтинг'
    elif 4.0 <= row['rating'] <= 4.3:
        return 'средний рейтинг'
    else:
        return 'высокий рейтинг'
df['rating_category'] = df.apply(cat_rating, axis=1)
df.sample(n=5, random_state=100)
```

Out[72]:

	name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_avg_bill	middle_coffee_cup	chain	seats	street	is_24/7	rating_category

526	живое пиво	кофейня	москва, смольная улица, 63б	Северный административный округ	ежедневно, 10:00–21:00	55.869735	37.470368	4.2	NaN	NaN	NaN	NaN	сетевое	25.0	Смольная улица	False	средний рейтинг
200	wild bean	кофейня	москва, дмитровское шоссе, 107е	Северный административный округ	ежедневно, круглосуточно	55.878477	37.543426	3.5	NaN	NaN	NaN	NaN	сетевое	20.0	Дмитровское шоссе	True	низкий рейтинг
1345	кулинарная лавка братьев караваевых	кафе	москва, балтийская улица, 5	Северный административный округ	ежедневно, 08:00–23:00	55.807679	37.511757	4.4	средние	средний счёт:400–800 ₽	600.0	NaN	сетевое	NaN	Балтийская улица	False	высокий рейтинг
938	славия	столовая	москва, ярославское шоссе, 44	Северо-Восточный административный округ	ежедневно, 07:00–19:00	55.863964	37.701587	4.4	NaN	NaN	NaN	NaN	несетевое	30.0	Ярославское шоссе	False	высокий рейтинг
6757	оморе море	ресторан	москва, ленинский проспект, 108, стр. 1	Западный административный округ	ежедневно, 09:00–23:00	55.669702	37.512464	1.0	NaN	NaN	NaN	NaN	сетевое	290.0	Ленинский проспект	False	критически низкий рейтинг

3.8.2 Средний чек в разрезе рейтинга

In [73]:

```
# посмотрим на средний чек в разрезе рейтинга
bill_rating = df.groupby(['category',
'rating_category'])['middle_avg_bill'].mean().reset_index()
bill_rating['middle_avg_bill'] = round(bill_rating['middle_avg_bill'])
bill_rating
```

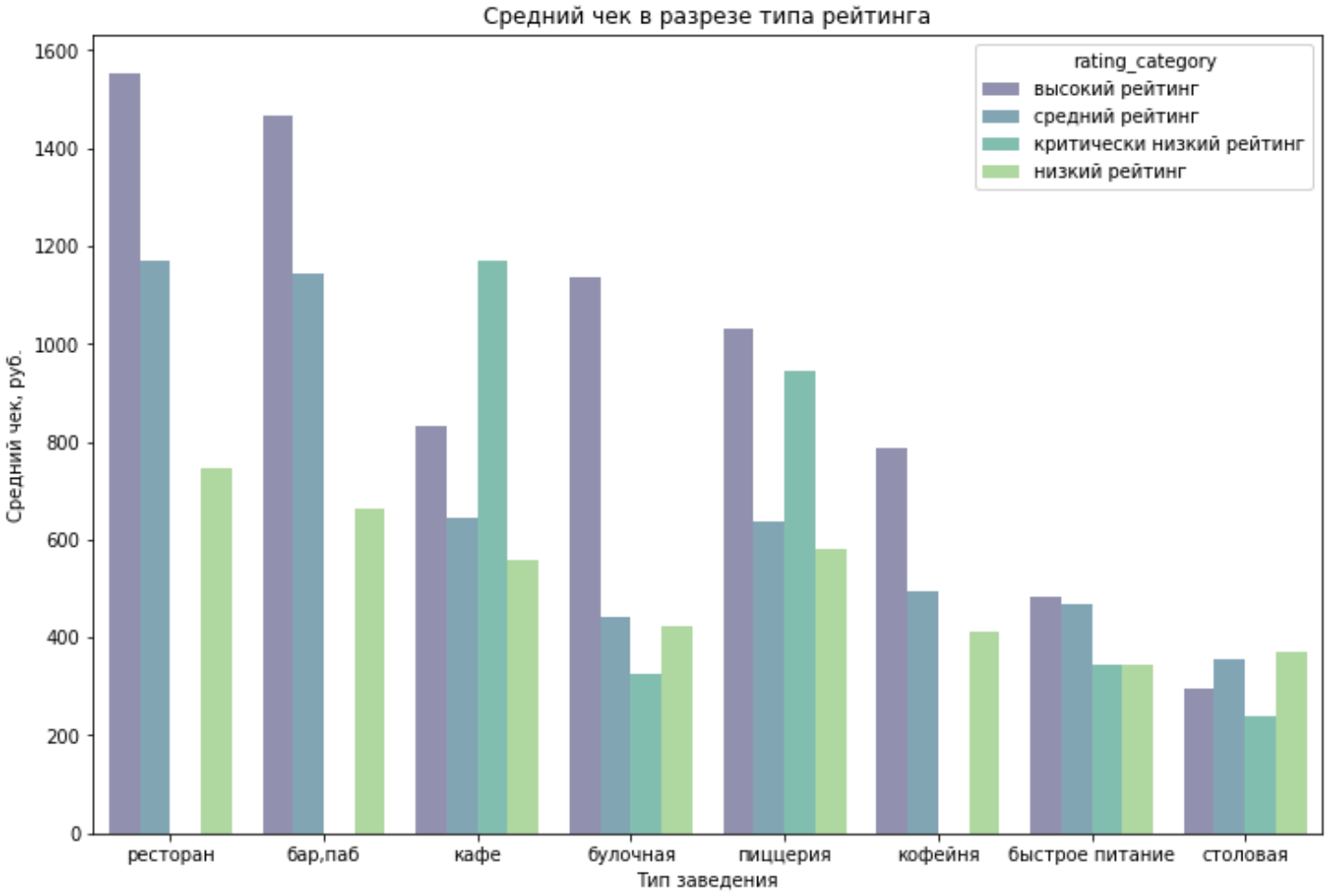
Out[73]:

	category	rating_category	middle_avg_bill
0	бар,паб	высокий рейтинг	1465.0
1	бар,паб	критически низкий рейтинг	NaN
2	бар,паб	низкий рейтинг	663.0
3	бар,паб	средний рейтинг	1142.0
4	булочная	высокий рейтинг	1136.0
5	булочная	критически низкий рейтинг	325.0
6	булочная	низкий рейтинг	425.0
7	булочная	средний рейтинг	441.0
8	быстрое питание	высокий рейтинг	482.0
9	быстрое питание	критически низкий рейтинг	343.0
10	быстрое питание	низкий рейтинг	345.0
11	быстрое питание	средний рейтинг	468.0
12	кафе	высокий рейтинг	831.0
13	кафе	критически низкий рейтинг	1169.0
14	кафе	низкий рейтинг	560.0
15	кафе	средний рейтинг	645.0
16	кофейня	высокий рейтинг	786.0
17	кофейня	критически низкий рейтинг	NaN
18	кофейня	низкий рейтинг	412.0
19	кофейня	средний рейтинг	494.0
20	пиццерия	высокий рейтинг	1031.0

21	пиццерия	критически низкий рейтинг	946.0
22	пиццерия	низкий рейтинг	579.0
23	пиццерия	средний рейтинг	636.0
24	ресторан	высокий рейтинг	1554.0
25	ресторан	критически низкий рейтинг	NaN
26	ресторан	низкий рейтинг	745.0
27	ресторан	средний рейтинг	1171.0
28	столовая	высокий рейтинг	294.0
29	столовая	критически низкий рейтинг	238.0
30	столовая	низкий рейтинг	372.0
31	столовая	средний рейтинг	355.0

In [74]:

```
plt.figure(figsize=(12, 8))
sns.barplot(
   x='category', y='middle_avg_bill',
   data=bill_rating.sort_values(by='middle_avg_bill', ascending=False),
   palette='viridis', alpha=0.6, hue='rating_category') # palette=rocket'mako'
'rocket' 'viridis' plasma


#  заголовок графика и подписи осей
plt.title('Средний чек в разрезе типа рейтинга')
plt.xlabel('Тип заведения')
plt.ylabel('Средний чек, руб.')


plt.show()
```



3.8.3 Вывод взаимосвязи среднего чека и рейтинга заведения

Кроме кафе и столовых, у остальных типов заведений - высокий рейтинг соответствует высокому уровню среднего чека. Особенно это заметно у ресторанов, баров/пабов - там средний чек колеблется в пределах 1500- 1600 с высоким рейтингом и средний чек в пределах 1200 руб со средним рейтингом. В булочных при среднем чеке в 1200 также высокий рейтинг.
В кафе наоборот, критически низкий рейтинг при высоком уровне среднего чека (в пределах 1200 руб), разве что у пиццерий еще есть такой критический уровень при среднем чеке около 1000 рублей. Но у пиццерий при среднем чеке слегка выше 1000 уровень высокого рейтинга всеже превышает критически низкую оценку.
Примечательно, что у ресторанов, баров и кофеен нет критически низкого рейтингах оценок.

3.9 Улицы с одним заведением общепита

Отфильтруем датасет и найдем все улици, на которых расположено только одно заведение общественного питания, сохраним список таких улиц в переменную onlyone_street. Для районов Москвы используем сокращения - аббервиатуры.

In [75]:

```
street_one_place = df.pivot_table(index = 'street', values = 'name', aggfunc =
'count').query("name == 1").reset_index()
onlyone_street = street_one_place['street']
print(f"В Москве {onlyone_street.count()} улиц, на которых расположено всего одно
заведение.")
```

В Москве 473 улиц, на которых расположено всего одно заведение.

In [76]:

```
df_onlyone_street = df.query("street in @onlyone_street")
df_onlyone_street['district'] = df_onlyone_street['district'].replace(['Центральный
административный округ',
'Западный административный округ',
'Северо-Западный административный округ',
'Юго-Западный административный округ',
'Южный административный округ',
'Юго-Восточный административный округ',
'Северный административный округ',
'Восточный административный округ',
'Северо-Восточный административный округ'],
['ЦАО', 'ЗАО', 'СЗАО', 'ЮЗАО', 'ЮАО', 'ЮВАО', 'САО', 'ВАО', 'СВАО'])
print('Административные районы г. Москвы в датасете после переименования:')
print(list(df_onlyone_street['district'].unique()))

df_onlyone_street.head(2)
```

Административные районы г. Москвы в датасете после переименования:
['САО', 'СВАО', 'СЗАО', 'ЗАО', 'ЦАО', 'ВАО', 'ЮВАО', 'ЮЗАО', 'ЮАО']

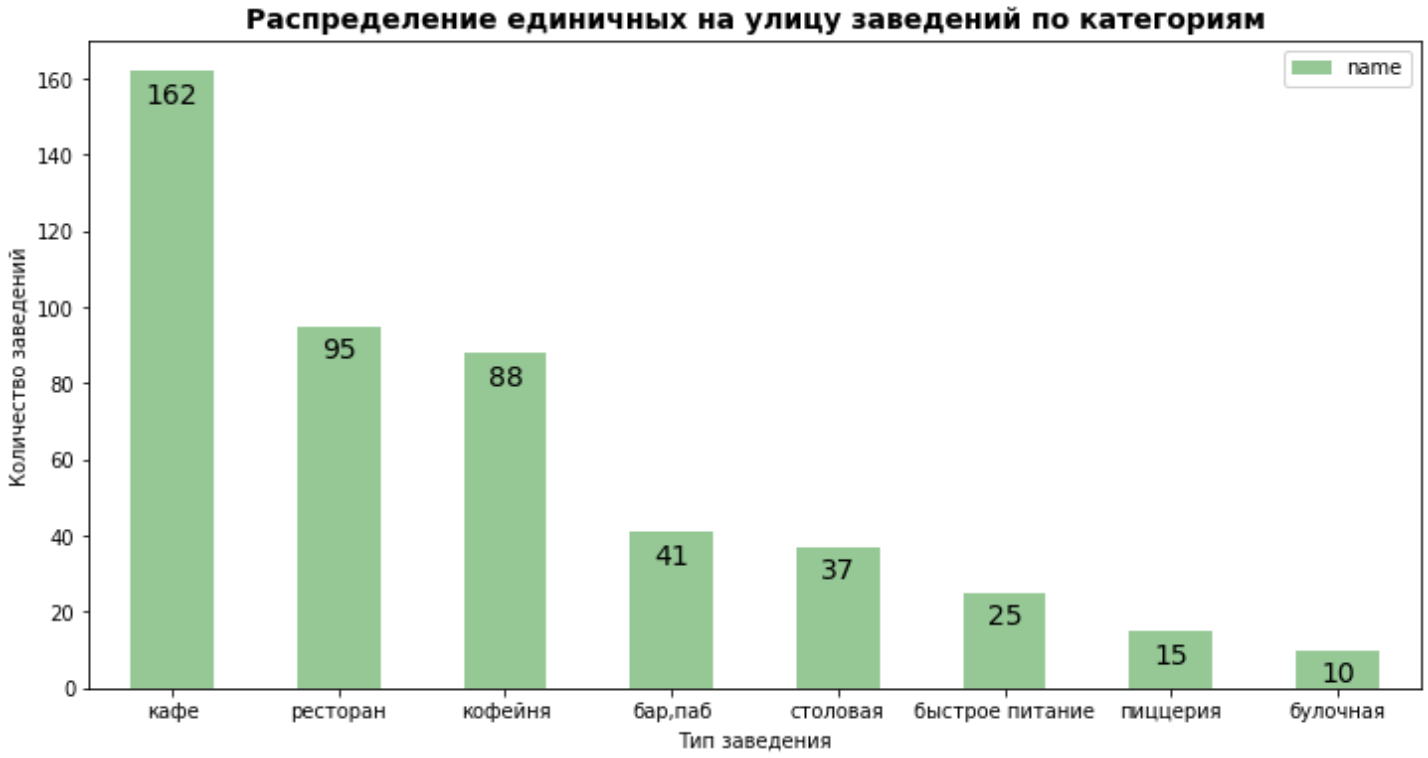
Out[76]:

	name	category	address	district	hours	lat	lng	rating	price	avg_bill	middle_avg_bill	middle_coffee_cup	chain	seats	street	is_24/7	rating_category
15	дом обеда	столовая	москва, улица бусиновская горка, 2	САО	пн-пт 08:30–18:30; сб 10:00–20:00	55.885890	37.493264	4.1	средние	средний счёт:300–500 ₽	400.0	NaN	не сетевое	180.0	улица Бусиновская Горка	False	средний рейтинг
21	7/12	кафе	москва, прибрежный проезд, 7	САО	ежедневно, 10:00–22:00	55.876805	37.464934	4.5	NaN	NaN	NaN	NaN	не сетевое	NaN	Прибрежный проезд	False	высокий рейтинг

3.9.1 Распределение единичных на улицу заведений по категориям

In [77]:

```
splot = df_onlyone_street.pivot_table(
       index='category',
       values='name',
       aggfunc='count').sort_values(by='name', ascending=False).plot(kind='bar',
figsize=(12, 6),
                                                                    color='green',
alpha=0.4)

#добавляем подписи к столбцам со значением
for p in splot.patches:
   splot.annotate(format(round(p.get_height(), 3), '.0f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha='center', va='center',
                  size=14,
                  xytext=(0, -12),
                  textcoords='offset points')

plt.title('Распределение единичных на улицу заведений по категориям', fontsize = 14,
fontweight ='bold')
plt.xlabel('Тип заведения')
plt.ylabel('Количество заведений')
plt.xticks(rotation=360)
#plt.grid()
plt.show()
```



3.9.2 Вывод взаимосвязь улиц с одним заведением и типа заведения

Заведением, которое расположено в единственном экземпляре на улице, скорее всего окажется кафе - их 160 одиноких. Одиноких булочных меньше, всего 8 на всю Москву. Посмотрим, в каком округе чаще всего расположены такие одиночные заведения.

3.9.3 Распределение единичных на улицу заведений по округам

In [78]:

```
district_only_one = df_onlyone_street.pivot_table(index = 'district', values =
'name', aggfunc = 'count').sort_values(by = 'name', ascending = False).reset_index()
```

```
plt.figure(figsize=(10, 6))
splot = sns.barplot(x='district', y='name', data=district_only_one, color='grey',
alpha=0.5)
#добавляем подписи к столбцам со значением
for p in splot.patches:
   splot.annotate(format(round(p.get_height(), 3), '.0f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha='center', va='center',
                  size=14,
                  xytext=(0, -12),
                  textcoords='offset points')

plt.title('Распределение единичных на улицу заведений по округам', fontsize = 14,
fontweight ='bold')
plt.xlabel('Административный округ г. Москвы')
plt.ylabel('Количество заведений')
plt.xticks(rotation=360)
#plt.grid()
plt.show()
```

3.9.4 Вывод взаимосвязь улиц с одним заведением и округа

Заведение, которое расположено в единственном экземпляре на улице, скорее всего окажется в ЦАО - 145 улиц, на котором расположено только одно заведение. В ЦАО много коротких улиц, а чем короче улица, тем меньше можно расположить на ней заведений. В ЮЗАО и в СЗАО - меньше всего улиц, на которых расположено всего одно заведение.

3.9.5 Итоги по одиночным заведениям

В Москве 459 улиц, на которых расположено всего одно заведение. Заведением, которое расположено в единственном экземпляре на улице, скорее всего окажется кафе - их 160 одиноких. Одиноких булочных меньше, всего 8 на всю Москву. Посмотрим, в каком округе чаще всего расположены такие одиночные заведения.

Заведение, которое расположено в единственном экземпляре на улице, скорее всего окажется в ЦАО - там 145 улиц, на котором расположено только одно заведение. В ЮЗАО и в СЗАО - меньше всего улиц, на которых расположено всего одно заведение.

3.10 Значения средних чеков заведений

Значения средних чеков заведений хранятся в столбце middle_avg_bill. Эти числа показывают примерную стоимость заказа, которая чаще всего выражена диапазоном.

Посчитаем медиану этого столбца для каждого района. Используем это значение в качестве ценового индикатора района. Построим фоновую картограмму (хороплет) с полученными значениями для каждого района.

In [79]:

```
median_bill = df.groupby('district')['middle_avg_bill'].median().reset_index()
median_bill.style.background_gradient()
```

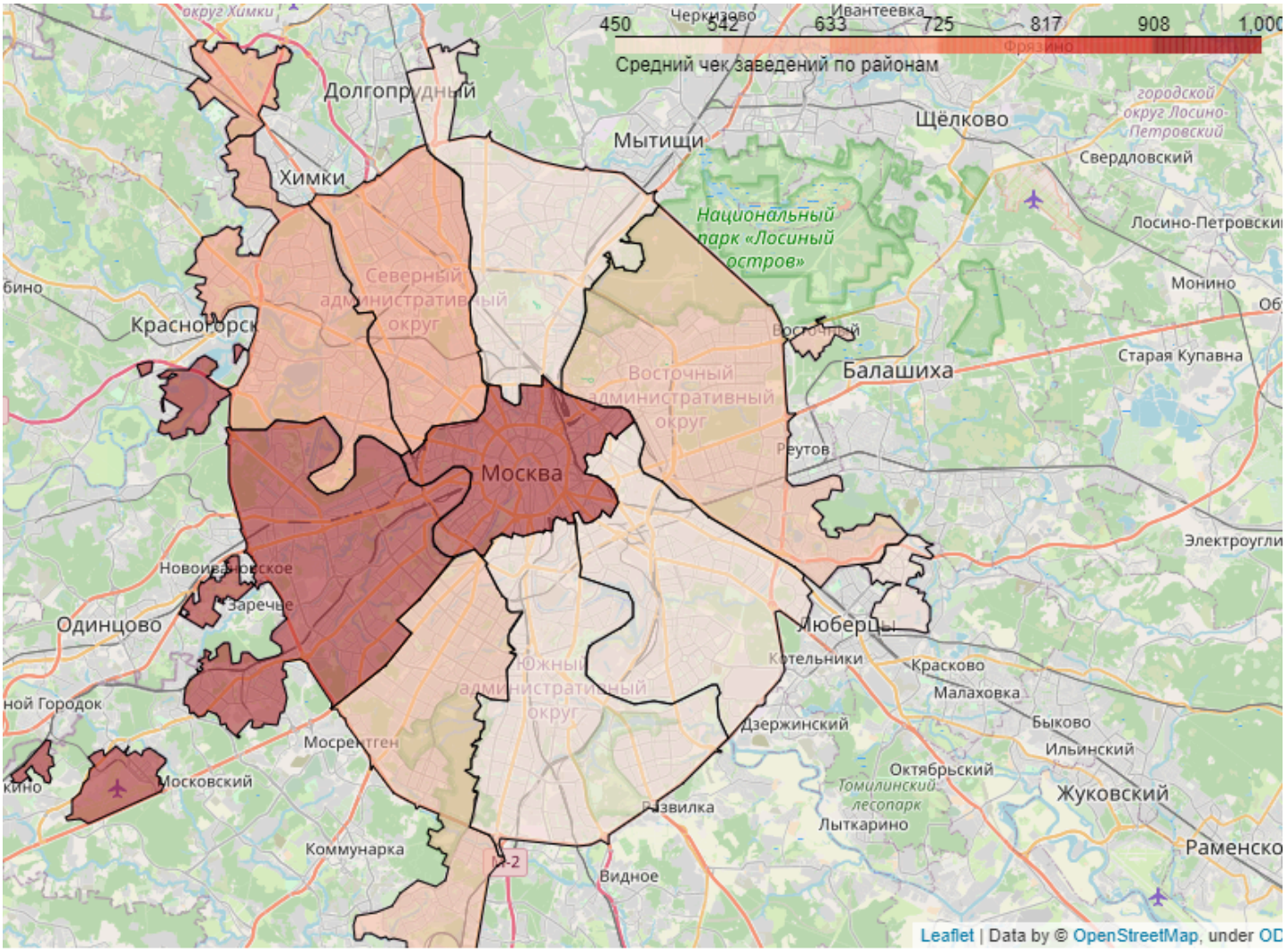
Out[79]:

	district	middle_avg_bill
0	Восточный административный округ	575.000000
1	Западный административный округ	1000.000000
2	Северный административный округ	650.000000
3	Северо-Восточный административный округ	500.000000
4	Северо-Западный административный округ	700.000000
5	Центральный административный округ	1000.000000
6	Юго-Восточный административный округ	450.000000
7	Юго-Западный административный округ	600.000000
8	Южный административный округ	500.000000

In [80]:

```
# создаем карту Москвы

mm = Map(location=[moscow_lat, moscow_lng], zoom_start=10)

# хороплет
Choropleth(
    geo_data=state_geo,
    data = median_bill,
    columns = ['district', 'middle_avg_bill'],
    key_on = 'feature.name',
    fill_color='Reds',
    fill_opacity=0.5,
    legend_name = 'Средний чек заведений по районам',
).add_to(mm)

mm
```

Out[80]:



3.10.1 Вывод по средним чекам

По данным выше мы видим, что самый высокий средний чек в Центральном и Западном округах (1000 руб.). Самый низкий средний чек в Юго-Восточном административном округе. Средний чек в Центральном административном округе и Западном административном округу в 1,5-2 раза выше, чем в остальных округах. Получается, что удаленность от центра не всегда влияет на величину среднего чека.

4 Выводы

1) Распределение самых распространенных видов заведений общественного питания в г. Москве по категориям в процентах выглядит так:

- кафе - 28.3 %
- рестораны - 24.3 %
- кофейня - 16.8 %

2) Медианные значения посадочных мест у заведений, шт:

- бар,паб 80
- ресторан 80
- кофейня 72
- столовая 72
- быстрое питание 60
- кафе 57
- пиццерия 50
- булочная 49

3) Большинство заведений не сетевые, их доля 62%, соответственно доля сетевых заведений - 38%.

4) Лидеры сетей - булочные, сетевыми у них являются - 60% от всех булочных г. Москвы. Сетевых - пиццерий 52%, а кофеен - 51%.

Сеть быстрого питания имеет 38% сетевых заведений, рестораны - 36%, а кафе 33%. Сетевых столовых еще меньше - 28%. Самый малый показатель сетевых заведений у баров,пабов -22% - они лидеры среди не сетевых заведений.

5) Самая популярная сеть - Шоколадница - по данным из датасета, у них 120 заведений на Москву. Второе место разделили между собой Домино'с пицца и Додо пицца 76 и 74 заведений у каждого соответственно. Это сети пиццерий.

Третье место с небольшим отрывом от пиццерий заняла сеть one price coffee. Замыкает топ-15 популярных заведений - кафе Му-Му. По данным датасета в Москве у этого кафе есть 27 сетевых заведений.

Среди самых популярных заведений оказалась яндекс лавка, в датафрейме это заведение относится к категории:'ресторан'. По факту же оно не являются классическим заведением общепита.

Эти категории не попали в топ-15: 'бар,паб', 'быстрое питание', 'столовая' 6) Всего рассмотрено заведений в 9-ти районах г. Москвы: 'ЦАО', 'ЗАО', 'СЗАО', 'ЮЗАО', 'ЮАО', 'ЮВАО', 'САО', 'ВАО', 'СВАО'.

ЦАО лидирует среди районов по количеству заведений- в этом округе аж 2242 заведений, почти 27% от всех рассматриваемых заведений. В ЦАО больше всего ресторанов, кафе, кофеен и баров/пабов, в остальных районах также много ресторанов, кафе, кофеен.

В остальных районах показатели заведений колеблются в пределах 10,7-8.4%, за исключением СЗАО, в котором находится всего 4,9% от всех рассматриваемых в исследовании заведений г. Москвы.

7) Сильных различий у усреднённых рейтингах в разных типах общепита нет. Самый высокий рейтинг у заведений -бар/паб - 4.388. Самый маленький рейтинг у ресторанов быстрого питания - 4.050.

8) Самый высокий рейтинг в заведениях в Центральном административном округе - 4.38. Самый низкий - в Юго-Восточном административном округе - 4.1.

9) Кроме кафе и столовых, у остальных типов заведений - высокий рейтинг соответствует высокому уровню среднего чека. Особенно это заметно у ресторанов, баров/пабов - там средний чек колеблется в районе 1500- 1600 с высоким рейтингом и средний чек в пределах 1200 со средним рейтингом. В булочных при среднем чеке в 1200 также высокий рейтинг.

В кафе наоборот, критически низкий рейтинг при высоком уровне среднего чека (в пределах 1200 руб). у ресторанов, баров и кофеен нет критически низкого рейтингах оценок.

10) Можем предположить, что чем длиннее улица, тем больше на ней находится заведений. А больше всего заведений расположено на проспекте Мира, там преобладают кафе, рестораны и кофейни. Вторая по популярности улица - Профсоюзная улица, там больше всего ресторанов и кафе.

На МКАД много кафе и нет пиццерий и булочных, а на Пятницкой улице и улице Вавилова нет столовых. На улице Миклухо-Маклая нет булочных, столовых.

11) В Москве 459 улиц, на которых расположено всего одно заведение. Заведением, которое расположено в единственном экземпляре на улице, скорее всего окажется кафе - их 160 одиноких. Одиноких булочных меньше, всего 8 на всю Москву. Посмотрим, в каком округе чаще всего расположены такие одиночные заведения.

Заведение, которое расположено в единственном экземпляре на улице, скорее всего окажется в ЦАО - 145 улиц, на котором расположено только одно заведение.

12) самый высокий средний чек в Центральном и Западном округах (1000 руб.). Самый низкий средний чек в Юго-Восточном административном округе. Средний чек в Центральном административном округе и Западном административном округу в 1,5-2 раза выше, чем в остальных округах. Получается, что удаленность от центра не всегда влияет на величину среднего чека.

5 Детализируем исследование: открытие кофейни

Цель: открыть доступную кофейню в Москве. Определим достижимость этой цели, для этого ответим на следующие вопросы:

- Сколько всего кофеен в датасете? В каких районах их больше всего, каковы особенности их расположения?
- Есть ли круглосуточные кофейни?
- Какие у кофеен рейтинги? Как они распределяются по районам?
- На какую стоимость чашки капучино стоит ориентироваться при открытии и почему?

- Построить визуализации.
- Дать рекомендацию для открытия нового заведения. Решение должно быть чем-то обосновано: текстом с описанием или маркерами на географической карте.

5.1 Сколько всего кофеен и в каких районах их больше всего.

Посмотрим сколько всего кофеен. В каких районах их больше всего и каковы особенности их расположения.

In [81]:

```
coffee_df = df[df['category'] == 'кофейня']
print(f"В Москве {len(coffee_df)} кофеен, {round(len(coffee_df) / len(df['name']) * 100)} % от всех заведений общественного питанияю")
```

В Москве 1413 кофеен, 17 % от всех заведений общественного питанияю

In [82]:

```
coffee_df_district = coffee_df.groupby('district', as_index=False)['name']\
.agg('count').round(3).sort_values('name', ascending=False)

coffee_df_district['ratio_%'] = round((coffee_df_district['name'] /
(coffee_df_district['name'].sum())) * 100)

coffee_df_district.style.background_gradient()
```

Out[82]:

	district	name	ratio_%
5	Центральный административный округ	428	30.000000
2	Северный административный округ	193	14.000000
3	Северо-Восточный административный округ	159	11.000000
1	Западный административный округ	150	11.000000
8	Южный административный округ	131	9.000000
0	Восточный административный округ	105	7.000000
7	Юго-Западный административный округ	96	7.000000
6	Юго-Восточный административный округ	89	6.000000
4	Северо-Западный административный округ	62	4.000000

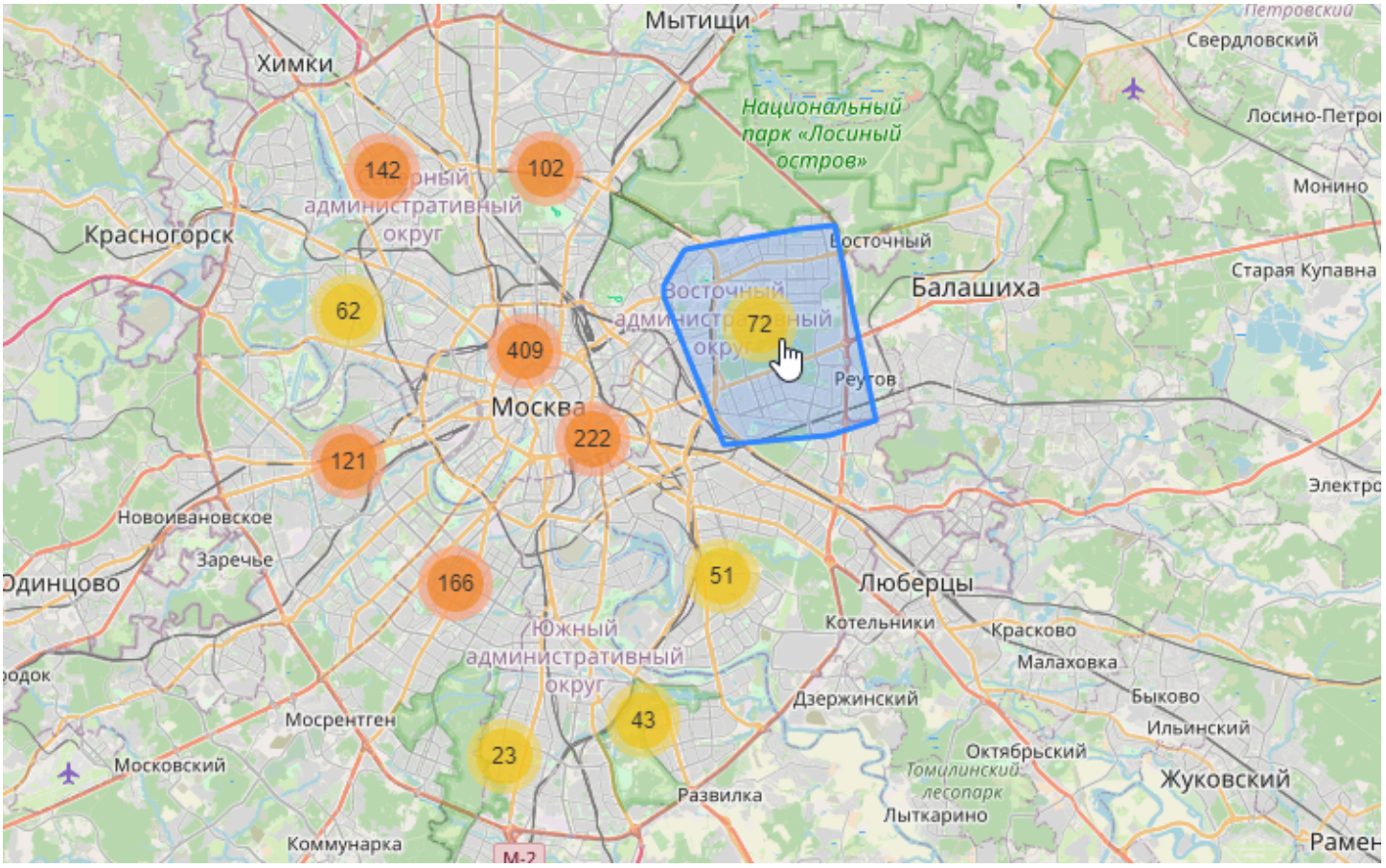
Больше всего кофеен в Центральном административном округе - 30% всех кофеен находятся именно там!
Посмотрим распределение кофеен на карте Москвы.

In [83]:

```
# создаем карту
m4 = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
# создаем пустой кластер и добавляем его на карту
marker_cluster = MarkerCluster().add_to(m4)

# функция, которая принимает строку датафрейма,
# создаёт маркер в текущей точке и добавляет его в кластер marker_cluster

def create_clusters(row):
    icon_url = 'https://img.icons8.com/?size=80&id=ocQcYk4Xz7Fc&format=png'
    #создаем объект с собственной иконкой размером 30x30
    icon = CustomIcon(icon_url, icon_size=(30, 30))
    #создаем маркер с иконкой icon и добавляем его в кластер
    Marker(
        [row['lat'], row['lng']],
        popup=f"{row['name']} {row['rating']}",
        icon=icon,
    ).add_to(marker_cluster)

# применяем функцию create_clusters() к каждой строке датафрейма
```

```
coffee_df.apply(create_clusters, axis=1)

# выводим карту
m4
```

Out[83]:



5.1.1 Итог по разделу:

В Москве 1413 кофеен. Большая часть кофеен располагается в Центральном административном округе, много кофеен также в Юго-Западном и Северном административных округах. Меньше всего кофеен - в Юговосточной и Южном административных округах города.

5.2 Круглосуточные кофейни в Москве

Посмотрим сколько всего круглосуточных кофеен в Москве, а также как они распределены по округам.

In [84]:

```
coffee_24_7 = coffee_df[coffee_df['is_24/7'] == True]
print(f"В Москве {len(coffee_24_7)} круглосуточных кофеен.")
```

В Москве 59 круглосуточных кофеен.

In [85]:

```
coffee_df_24_7 = coffee_24_7.groupby('district', as_index=False)['is_24/7']\
.agg('count').round(3).sort_values('is_24/7', ascending=False)

coffee_df_24_7['ratio_%'] = round((coffee_df_24_7['is_24/7'] /
(coffee_df_24_7['is_24/7'].sum())) * 100)

coffee_df_24_7.style.background_gradient()
```

Out[85]:

	district	is_24/7	ratio_%
5	Центральный административный округ	26	44.000000
1	Западный административный округ	9	15.000000

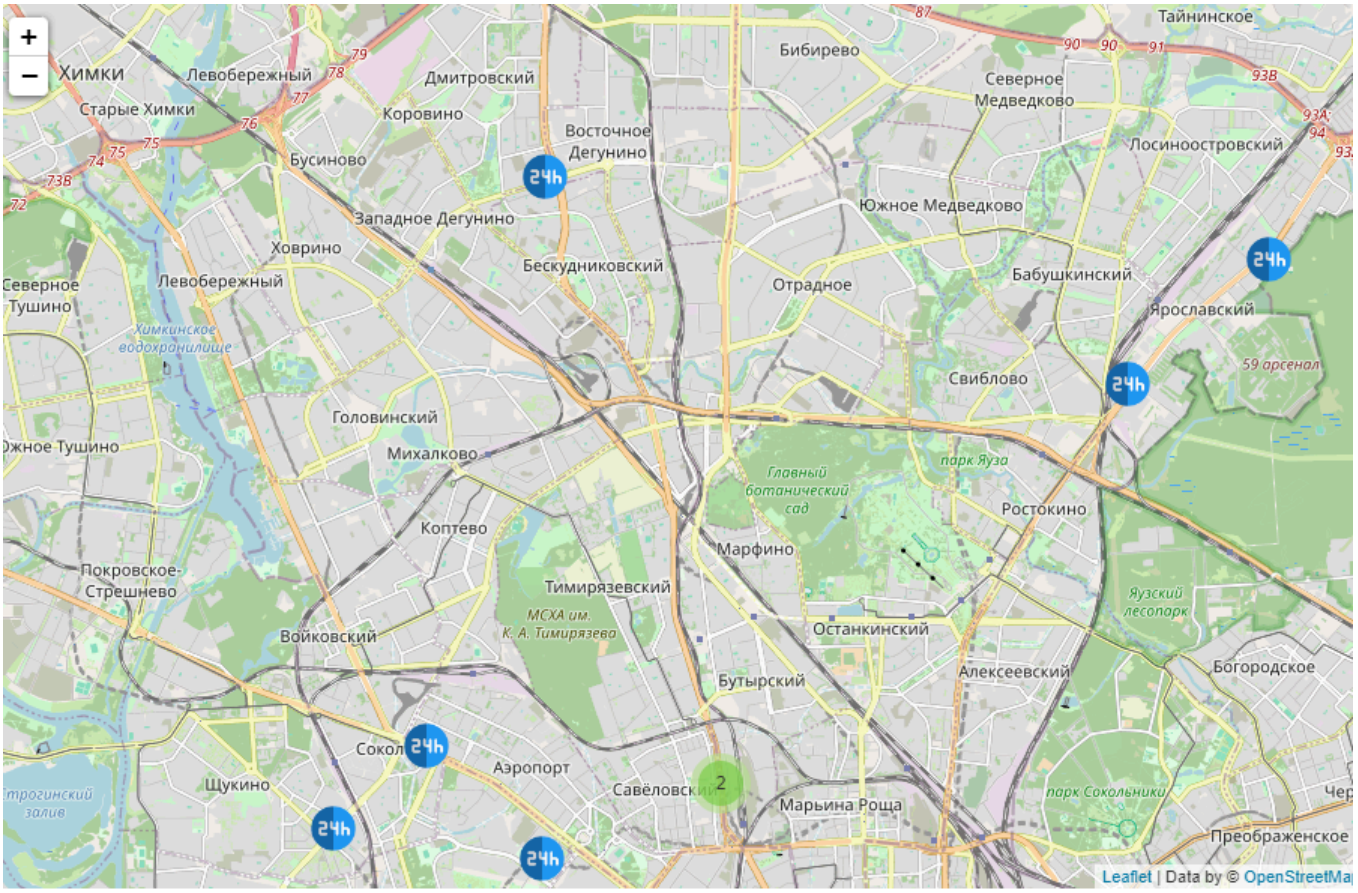
7	Юго-Западный административный округ	7	12.000000
0	Восточный административный округ	5	8.000000
2	Северный административный округ	5	8.000000
3	Северо-Восточный административный округ	3	5.000000
4	Северо-Западный административный округ	2	3.000000
6	Юго-Восточный административный округ	1	2.000000
8	Южный административный округ	1	2.000000

В Москве 59 круглосуточных кофеен, 44% из них (26шт.) расположены в пределах ЦАО.

In [86]:

```
# создаем карту
m5 = Map(location=[moscow_lat, moscow_lng], zoom_start=10)
# создаем пустой кластер и добавляем его на карту
marker_cluster = MarkerCluster().add_to(m5)

# функция, которая принимает строку датафрейма,
# создаёт маркер в текущей точке и добавляет его в кластер marker_cluster

def create_clusters(row):
    #ссылка на картинку
    icon_url =
'https://img.icons8.com/?size=100&id=4oOKWeSsi7dl&format=png&color=000000'
    #создаем объект с собственной иконкой размером 30x30
    icon = CustomIcon(icon_url, icon_size=(30, 30))
    #создаем маркер с иконкой icon и добавляем его в кластер
    Marker(
        [row['lat'], row['lng']],
        popup=f"{row['name']} {row['rating']}",
        icon=icon,
    ).add_to(marker_cluster)

# применяем функцию create_clusters() к каждой строке датафрейма
coffee_24_7.apply(create_clusters, axis=1)

# выводим карту
m5
```

Out[86]:



5.2.1 Итог по разделу:

В Москве 59 круглосуточных кофеен. Большинство круглосуточных кофеен находится в центре города. В других районах города их меньше, например в Северном Южном и Юго-Восточным административных округах всего по одной кофейне, работающей круглосуточно.

5.3 Рейтинги кофеен.

Посмотрим рейтинг кофеен по районам города. Сгруппируем таблицу с колонкой района и рейтингом по району, а также с отклонением от среднего рейтинга кофеен по Москве по всем округам, в процентах. Если отклонение положительное, то рейтинг в округе выше среднего по Москве, если отклонение отрицательно, то соответственно ниже.

In [87]:

```
coffee_rating = coffee_df.groupby('district', as_index=False)['rating']\
.agg('mean').round(3).sort_values('rating', ascending=False)
print(f"Средний рейтинг кофеен в Москве = {round(coffee_rating['rating'].mean())} из 5.0")
coffee_rating['deviation_%'] = round((coffee_rating['rating'] /
(coffee_rating['rating'].mean()) -1) * 100, 2)

coffee_rating.style.background_gradient()
```

Средний рейтинг кофеен в Москве = 4 из 5.0

Out[87]:

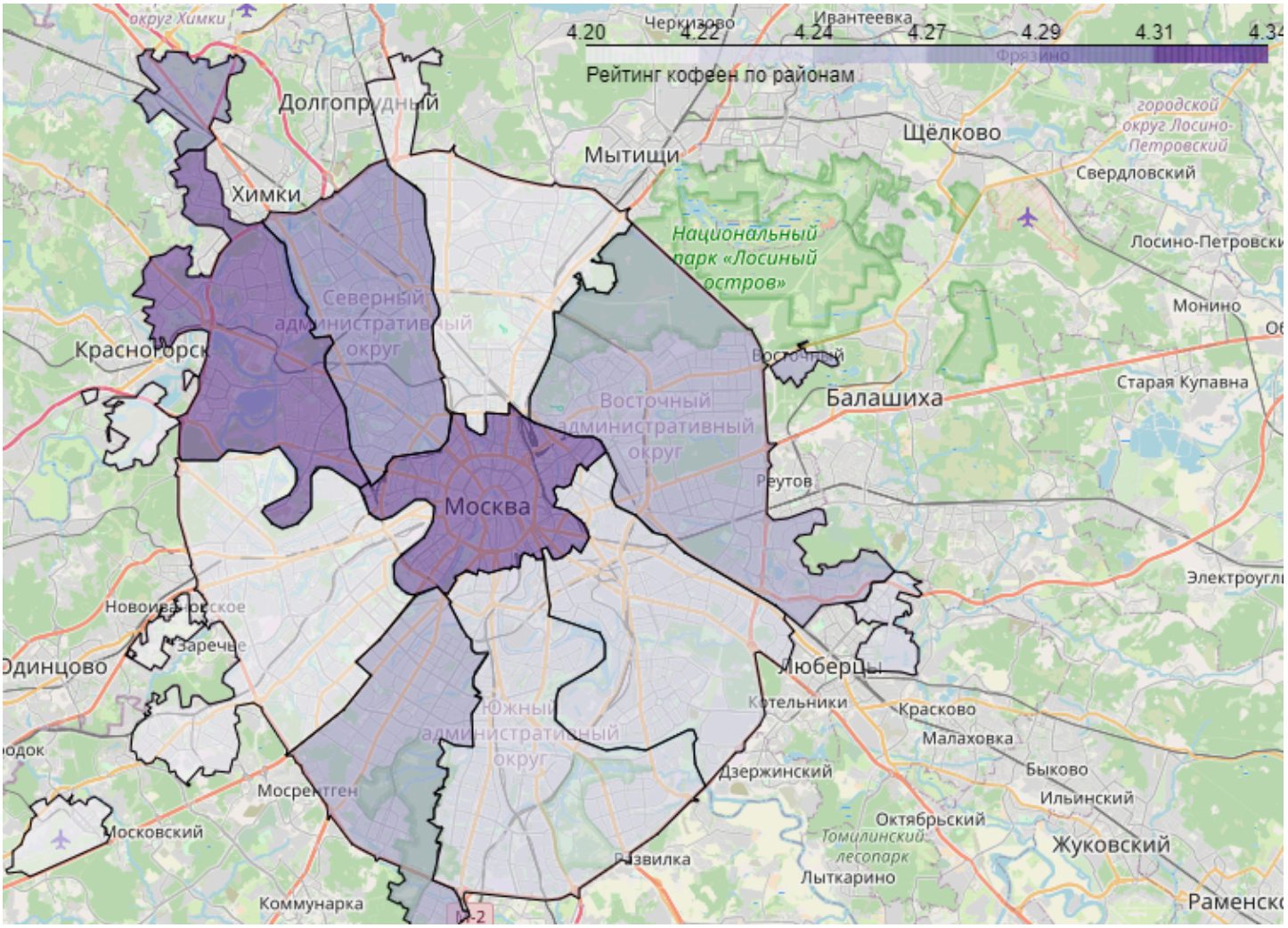
	district	rating	deviation_%
5	Центральный административный округ	4.336000	1.650000
4	Северо-Западный административный округ	4.326000	1.410000
2	Северный административный округ	4.292000	0.620000
0	Восточный административный округ	4.283000	0.410000
7	Юго-Западный административный округ	4.283000	0.410000
8	Южный административный округ	4.233000	-0.770000
6	Юго-Восточный административный округ	4.226000	-0.930000
3	Северо-Восточный административный округ	4.217000	-1.140000

1	Западный административный округ	4.195000	-1.660000

In [88]:

```
m6 = Map(location=[moscow_lat, moscow_lng], zoom_start=10)

# хороплет
Choropleth(
    geo_data=state_geo,
    data = coffee_rating,
    columns = ['district', 'rating'],
    key_on = 'feature.name',
    fill_color='Purples',
    fill_opacity=0.5,
    legend_name = 'Рейтинг кофеен по районам',
).add_to(m6)

m6
```

Out[88]:



5.3.1 Итог по разделу:

Самый высокий рейтинг в округах: Центральный административный округ и Северо-Западный административный округ - 4.34.
У Западного административного округа самый низкий рейтинг 4.20

5.4 Стоимость чашки капучино в кофейне.

Посмотрим стоимость чашки капучино в кофейне по округам города. Сгруппируем таблицу с колонкой округа, стоимостью чашки капучино и отклонением от стоимости средней чашки капучино по Москве, в процентах. Если отклонение положительное, то стоимость чашки капучино выше средней по Москве, если отклонение отрицательное, то соответственно ниже.

In [89]:

```
cup_of_coffee = coffee_df.groupby('district', as_index=False)['middle_coffee_cup']\
.agg('mean').round(2).sort_values('middle_coffee_cup', ascending=False)
print(f"Средняя стоимость одной чашки капучино в кофейне =
{round(cup_of_coffee['middle_coffee_cup'].mean())} руб.")
cup_of_coffee['deviation_%'] = round((cup_of_coffee['middle_coffee_cup'] /
(cup_of_coffee['middle_coffee_cup'].mean()) -1) * 100, 2)
cup_of_coffee.style.background_gradient()
```

Средняя стоимость одной чашки капучино в кофейне = 171 руб.

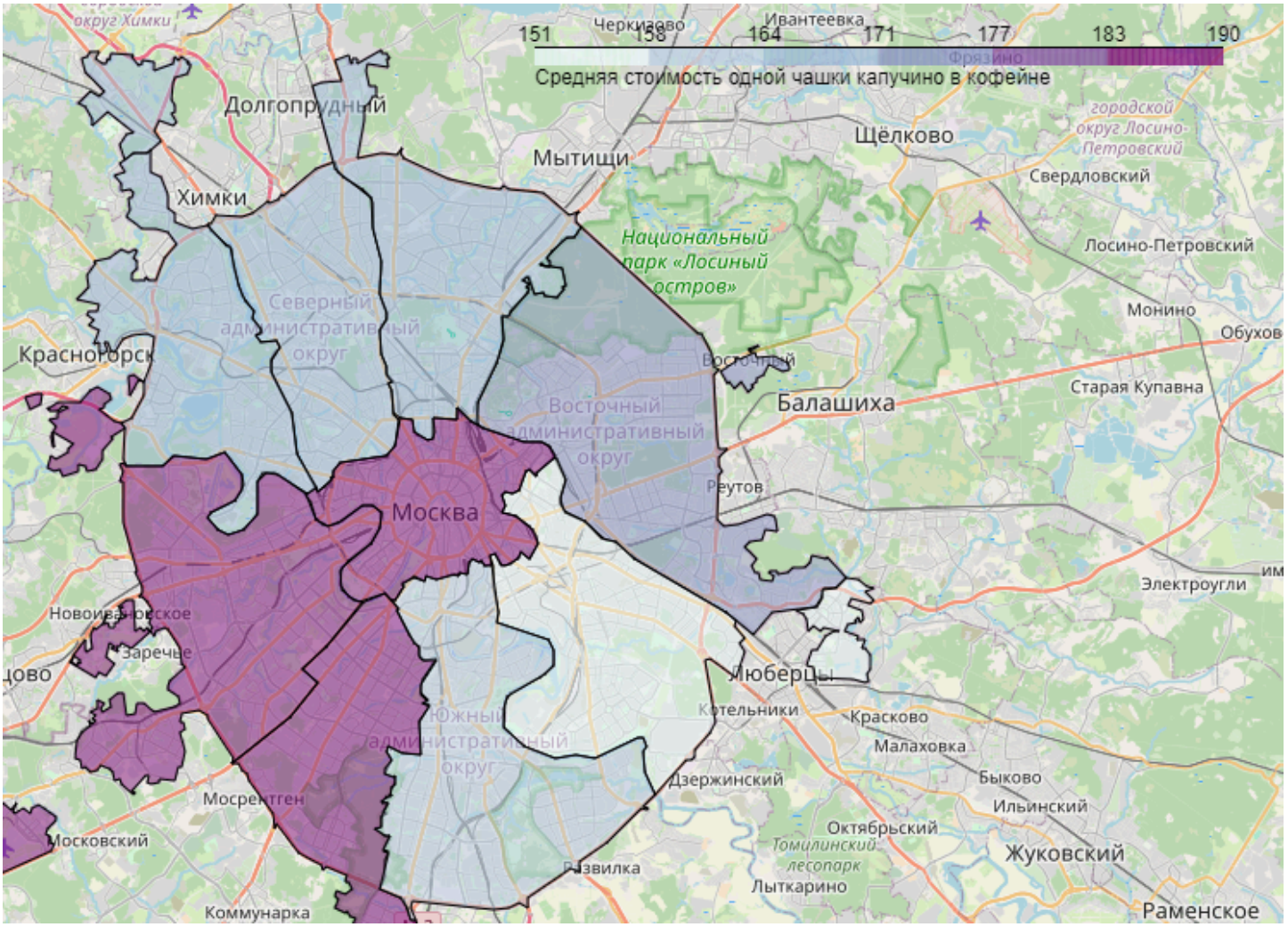
Out[89]:

	district	middle_coffee_cup	deviation_%
1	Западный административный округ	189.940000	10.870000
5	Центральный административный округ	187.520000	9.460000
7	Юго-Западный административный округ	184.180000	7.510000
0	Восточный административный округ	174.020000	1.580000
2	Северный административный округ	165.790000	-3.230000
4	Северо-Западный административный округ	165.520000	-3.390000
3	Северо-Восточный административный округ	165.330000	-3.500000
8	Южный административный округ	158.490000	-7.490000
6	Юго-Восточный административный округ	151.090000	-11.810000

In [90]:

```
m7 = Map(location=[moscow_lat, moscow_lng], zoom_start=10)

# хороплет
Choropleth(
    geo_data=state_geo,
    data = cup_of_coffee,
    columns = ['district', 'middle_coffee_cup'],
    key_on = 'feature.name',
    fill_color='BuPu',
    fill_opacity=0.5,
    legend_name = 'Средняя стоимость одной чашки капучино в кофейне',
).add_to(m7)

m7
```

Out[90]:



5.4.1 Итог по разделу:

В Западном (189.94 руб.) и Центральном (187.52 руб.) административных округах средняя стоимость чашки капучино выше, чем в других районах города. Средняя стоимость чашки капучино в Москве в кофейнях = 172 рубля. Средняя стоимость чашки капучино в Юго-Восточном административном округе г. Москвы самая низкая (151.09 руб.)

5.5 Выводы по дополнительному исследованию - открытие кофейни.

Кофейня «Central Perk» из сериала «Друзья» служит нам как основная идея открытия. Название «Central Perk» несет в себе несколько смыслов, как лежащих на поверхности, так и зашифрованных в игре слов и произношения. В нем зашифрованы ценности для клиента.

Вот основные мысли из этой статьи:

На русский его перевели довольно скучно, на мой взгляд, — просто “Центральное кафе”.

Что же скрыто в этом простом, на первый взгляд, названии. В сердце Манхеттена располагается Центральный Парк, по-английски — Central Park. Название кофейни созвучно названию парка, и первая ассоциация, которая приходит на ум — это именно Центральный Парк.

Вторая ассоциация связана со словом perk. Что такое perk? Это сокращение от percolate — “заваривать кофе”.

Словом perk в разговорной речи также обозначаются разного рода дополнительные бонусы к основному продукту, разного рода “плюшки”, “фишки”. Например, perks of adult life — “преимущества взрослой жизни”.

Так что в английском названии заложены и география (причем, угадывается даже город, где расположено кофейня, а не просто центральное расположение в произвольном населенном пункте), и тип заведения — кафе с кофейной специализацией. Причем в названии содержится игра слов и изрядная доля юмора, так как дается намеком присутствие некоторых “плюшек” или бонусов, ожидающих потенциальных посетителей кофейни.

Главной рекомендацией для открытия кофейни, которое может повторить успех киношного «Central Perk» - начать с истории или девиза, смысла, которое будет зашифровано в названии кофейни. История кофейни «Central Perk» построена, в том числе вокруг слова центральный.

Поэтому очевидной рекомендацией будет обратить внимание на Центральный административный округ. Там и центр Москвы и нулевой километр и еще сотни историй, которые можно обернуть вокруг слова центральный или другого слова с похожим смыслом/посылом.

Помимо идейной составляющей, в пользу ЦАО тот факт, что люди в этом округе пьют кофе. Об этом говорят цифры в ЦАО:

- находится 30% от общего числа кофеен в Москве.
- рейтинг кафе на 1,65% выше чем средний рейтинг кофеен по Москве, значит культура кофеен там поддерживается,
- стоимость чашки капучино на 9,46% выше средней стоимости по Москве
- 44 % круглосуточных кофеен (26шт.) расположены в пределах ЦАО.

Сетевых кофеен по Москве 51%.Концепция кофейни скорее всего подразумевает открытие на начальном этапе одной кофейни, а не сети. Если в дальнейшем планируется открывать сеть кофеен, то стоит заложить в историю что-то, что позволит масштабироваться, сохраняя идею.

Презентация