In [1]:
```
import pandas as pd
import numpy as np
# import matplotlib.pyplot as plt
# import seaborn as sns

from statsmodels.formula.api import ols
from scipy import stats
```

Задание:

1. Проведите разведочный анализ данных. Сделайте основные выводы про полноту датасета.
2. Нужно ли увеличить количество полицейских, чтобы снизить количество краж? Поясни свой ответ.

In [2]:
```
"""
first_part
https://docs.google.com/spreadsheets/d/1JkevCnJrPzzVnhC4FMRunBVWs_g5q9mqM

Date: Дата.

District: Район города.

Warehouse_Name: Название склада.

Number_of_Guards: Количество охраны на складе.

Police_Units: Количество полицейских на район.
"""

first_part = pd.read_excel('first_part.xlsx')
first_part.head()
```

Out[2]:

	Date	District	Warehouse_Name	Number_of_Guards	Police_Units
0	2024-01-26	Мышеостровский	Колбасовы	7	6
1	2024-01-26	Мышеостровский	Молочковы	10	9
2	2024-01-26	Мышеостровский	Мятновы	8	8
3	2024-01-26	Мышеостровский	Сметанинковы	8	7
4	2024-01-26	Краснокотейский	Колбасовы	7	6

In [3]:
```
"""
```

```
second_part
https://docs.google.com/spreadsheets/d/12ENNpZ18t2-aBX2nAIjYSwV3hOeaW6Sjd

Date: Дата.

District: Район города.

Warehouse_Name: Название склада.

Percent_of_Crime_Solved: Процент раскрытых преступлений по уничтожению фо

Number_of_Lights: Количество фонарей в районе.
"""

second_part = pd.read_excel('second_part.xlsx')
second_part.head()
```

Out[3]:

	Date	District	Warehouse_Name	Percent_of_Crime_Solved	Number_
0	2024-01-26	Мышеостровский	Колбасовы	83.160959	
1	2024-01-26	Мышеостровский	Молочковы	87.756487	
2	2024-01-26	Мышеостровский	Мятновы	87.010121	
3	2024-01-26	Мышеостровский	Сметанинковы	82.207203	
4	2024-01-26	Краснокотейский	Колбасовы	86.684716	

Проверим полноту датасета:

In [4]:
```
print(first_part.shape)
first_part.info()
```

```
(17280, 5)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17280 entries, 0 to 17279
Data columns (total 5 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Date              17280 non-null  datetime64[ns]
 1   District          17280 non-null  object
 2   Warehouse_Name    17280 non-null  object
 3   Number_of_Guards  17280 non-null  int64
 4   Police_Units      17280 non-null  int64
dtypes: datetime64[ns](1), int64(2), object(2)
memory usage: 675.1+ KB
```

In [5]:
```
print(second_part.shape)
```

```
second_part.info()
```

```
(17280, 5)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17280 entries, 0 to 17279
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Date                    17280 non-null  datetime64[ns]
 1   District                17280 non-null  object
 2   Warehouse_Name          17280 non-null  object
 3   Percent_of_Crime_Solved  17280 non-null  float64
 4   Number_of_Lights        17280 non-null  int64
dtypes: datetime64[ns](1), float64(1), int64(1), object(2)
memory usage: 675.1+ KB
```

В обоих датасетах не существует ни одного `null` значения.

In [6]:
```
print(first_part.columns)
print((first_part
     .groupby(['Date', 'District', 'Warehouse_Name'])
     .value_counts()
).max())

print(second_part.columns)
print((second_part
     .groupby(['Date', 'District', 'Warehouse_Name'])
     .value_counts()
).max())
```

```
Index(['Date', 'District', 'Warehouse_Name', 'Number_of_Guards',
       'Police_Units'],
      dtype='object')
1
Index(['Date', 'District', 'Warehouse_Name', 'Percent_of_Crime_Solved',
       'Number_of_Lights'],
      dtype='object')
1
```

Триплет `['Date', 'District', 'Warehouse_Name']` является уникальным ключом для обеих частей, так что можно попробовать соединить их:

In [7]:
```
full_data = first_part.merge(
    right=second_part,
    on=['Date', 'District', 'Warehouse_Name'],
    how='inner'
)
print(full_data.columns)
print(full_data.shape)
full_data.head()
```

```
Index(['Date', 'District', 'Warehouse_Name', 'Number_of_Guards',
       'Police_Units', 'Percent_of_Crime_Solved', 'Number_of_Lights'],
      dtype='object')
(17280, 7)
```

Out[7]:

	Date	District	Warehouse_Name	Number_of_Guards	Police_Units	P
0	2024-01-26	Мышеостровский	Колбасовы	7	6	
1	2024-01-26	Мышеостровский	Молочковы	10	9	
2	2024-01-26	Мышеостровский	Мятновы	8	8	
3	2024-01-26	Мышеостровский	Сметанинковы	8	7	
4	2024-01-26	Краснокотейский	Колбасовы	7	6	

Соединение прошло удачно. Посмотрим, как распределены данные:

In [8]:
```
(full_data
    ['District']
    .value_counts()
)
```

Out[8]:
```
District
Мышеостровский     2880
Краснокотейский    2880
Мышесельский       2880
Петрокотский       2880
Приморский         2880
Невский            2880
Name: count, dtype: int64
```

In [9]:
```
(full_data
    ['Warehouse_Name']
    .value_counts()
)
```

Out[9]:
```
Warehouse_Name
Колбасовы       4320
Молочковы       4320
Мятновы         4320
Сметанинковы    4320
Name: count, dtype: int64
```

In [10]:
```
(full_data
    [['District', 'Warehouse_Name']]
    .value_counts()
)
```

Out[10]:

```
District         Warehouse_Name
Краснокотейский  Колбасовы         720
                 Молочковы         720
Приморский       Мятновы           720
                 Молочковы         720
                 Колбасовы         720
Петрокотский     Сметанинковы      720
                 Мятновы           720
                 Молочковы         720
                 Колбасовы         720
Невский          Сметанинковы      720
                 Мятновы           720
                 Молочковы         720
                 Колбасовы         720
Мышесельский     Сметанинковы      720
                 Мятновы           720
                 Молочковы         720
                 Колбасовы         720
Мышеостровский   Сметанинковы      720
                 Мятновы           720
                 Молочковы         720
                 Колбасовы         720
Краснокотейский  Сметанинковы      720
                 Мятновы           720
Приморский       Сметанинковы      720
Name: count, dtype: int64
```

Помимо того, что названия округов *подозрительно напоминают названия районов Санкт-Петербурга,* заметим, что данные по ним распределены **чрезвычайно ровно.** Их визуализация не даёт ничего полезного, так что приводить её здесь я не буду. Однако, я бы хотел оценить, насколько их распределения совпадают по среднему значению и СКО.

In [11]:

```
metric_list = ('Number_of_Guards', 'Police_Units', 'Percent_of_Crime_Solv

def test_data(data, metrics_to_test, field_to_test, test_to_use):
    res_str = ">>> {}\n".format(test_to_use)

    field_values = data[field_to_test].unique()

    for metric in metric_list:
        res_str += '>> {}\n'.format(metric)
        for fv in field_values[1:]:
            ttest_res = stats.ttest_ind(
                data[data[field_to_test] == field_values[0]][metric],
                data[data[field_to_test] == fv][metric],
            )

            eq_string = ''
            if ttest_res.pvalue <= 0.05:
                eq_string = 'НЕ '
```

```
            res_str += """> {}-{} {}равны по показателю {}:\n{}\n\n""".fo
                field_values[0],
                fv,
                eq_string,
                metric,
                ttest_res
            )
        res_str += '\n'

    return res_str
```

In [12]:
```
print(test_data(
    data=full_data,
    metrics_to_test=metric_list,
    field_to_test='District',
    test_to_use=stats.ttest_ind,
))
```

```
>>> <function ttest_ind at 0x127a3d9d0>
>> Number_of_Guards
> Мышеостровский-Краснокотейский равны по показателю Number_of_Guards:
TtestResult(statistic=0.8715041607006833, pvalue=0.38351527024669474, df=5
758.0)

> Мышеостровский-Мышесельский равны по показателю Number_of_Guards:
TtestResult(statistic=0.4825399632092622, pvalue=0.6294407596917191, df=57
58.0)

> Мышеостровский-Петрокотский равны по показателю Number_of_Guards:
TtestResult(statistic=-0.30863835552401464, pvalue=0.7576078063723227, df=
5758.0)

> Мышеостровский-Приморский равны по показателю Number_of_Guards:
TtestResult(statistic=0.32882672248510725, pvalue=0.7422986221006729, df=5
758.0)

> Мышеостровский-Невский равны по показателю Number_of_Guards:
TtestResult(statistic=0.03905434517787736, pvalue=0.9688484143849871, df=5
758.0)


>> Police_Units
> Мышеостровский-Краснокотейский равны по показателю Police_Units:
TtestResult(statistic=1.1454536349606426, pvalue=0.2520688965348275, df=57
58.0)

> Мышеостровский-Мышесельский равны по показателю Police_Units:
TtestResult(statistic=0.6022231939766808, pvalue=0.5470492918713652, df=57
58.0)

> Мышеостровский-Петрокотский равны по показателю Police_Units:
TtestResult(statistic=-0.3067142118950546, pvalue=0.7590720018236087, df=5
758.0)
```

> Мышеостровский–Приморский равны по показателю Police_Units:
TtestResult(statistic=0.08678638459973438, pvalue=0.930844327986379, df=57
58.0)

> Мышеостровский–Невский равны по показателю Police_Units:
TtestResult(statistic=0.1332212004697219, pvalue=0.8940231001557188, df=57
58.0)

>> Percent_of_Crime_Solved
> Мышеостровский–Краснокотейский равны по показателю Percent_of_Crime_Solv
ed:
TtestResult(statistic=0.23448439779536875, pvalue=0.8146172815196955, df=5
758.0)

> Мышеостровский–Мышесельский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=-0.1532499802938289, pvalue=0.8782065146470208, df=5
758.0)

> Мышеостровский–Петрокотский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.30068319613350686, pvalue=0.7636669395827532, df=5
758.0)

> Мышеостровский–Приморский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=-0.010761776050321675, pvalue=0.9914138836124542, df
=5758.0)

> Мышеостровский–Невский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.15121383403814914, pvalue=0.8798123349622053, df=5
758.0)

>> Number_of_Lights
> Мышеостровский–Краснокотейский равны по показателю Number_of_Lights:
TtestResult(statistic=-0.8476611501080906, pvalue=0.3966619408712536, df=5
758.0)

> Мышеостровский–Мышесельский равны по показателю Number_of_Lights:
TtestResult(statistic=0.8444025888717446, pvalue=0.39847955730860873, df=5
758.0)

> Мышеостровский–Петрокотский равны по показателю Number_of_Lights:
TtestResult(statistic=0.2980367122362964, pvalue=0.7656858970387669, df=57
58.0)

> Мышеостровский–Приморский равны по показателю Number_of_Lights:
TtestResult(statistic=0.6269577833380775, pvalue=0.5307117658183869, df=57
58.0)

> Мышеостровский–Невский равны по показателю Number_of_Lights:
TtestResult(statistic=0.6081620018366941, pvalue=0.5431040206796555, df=57
58.0)

In [13]:
```
print(test_data(
    data=full_data,
    metrics_to_test=metric_list,
    field_to_test='Warehouse_Name',
    test_to_use=stats.ttest_ind,
))
```

```
>>> <function ttest_ind at 0x127a3d9d0>
>> Number_of_Guards
> Колбасовы-Молочковы равны по показателю Number_of_Guards:
TtestResult(statistic=-0.05987771723013768, pvalue=0.9522544122845377, df=
8638.0)

> Колбасовы-Мятновы равны по показателю Number_of_Guards:
TtestResult(statistic=-0.9886615638226333, pvalue=0.3228564463875877, df=8
638.0)

> Колбасовы-Сметанинковы равны по показателю Number_of_Guards:
TtestResult(statistic=0.19579113250347852, pvalue=0.8447782563751698, df=8
638.0)


>> Police_Units
> Колбасовы-Молочковы равны по показателю Police_Units:
TtestResult(statistic=0.2044125312672294, pvalue=0.8380359542352203, df=86
38.0)

> Колбасовы-Мятновы равны по показателю Police_Units:
TtestResult(statistic=-0.46918340065064884, pvalue=0.6389503909404572, df=
8638.0)

> Колбасовы-Сметанинковы равны по показателю Police_Units:
TtestResult(statistic=0.7209938018615215, pvalue=0.4709328403159081, df=86
38.0)


>> Percent_of_Crime_Solved
> Колбасовы-Молочковы равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.32859944819108805, pvalue=0.7424664306651769, df=8
638.0)

> Колбасовы-Мятновы равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=-0.10869818088869354, pvalue=0.9134444094976287, df=
8638.0)

> Колбасовы-Сметанинковы равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.09716204633791999, pvalue=0.9225999622041798, df=8
638.0)


>> Number_of_Lights
```

```
> Колбасовы-Молочковы равны по показателю Number_of_Lights:
TtestResult(statistic=1.471064906416363, pvalue=0.1413099908568232, df=863
8.0)

> Колбасовы-Мятновы равны по показателю Number_of_Lights:
TtestResult(statistic=0.5695557582658989, pvalue=0.5689938531140089, df=86
38.0)

> Колбасовы-Сметанинковы равны по показателю Number_of_Lights:
TtestResult(statistic=1.1114495872055554, pvalue=0.2664058122338372, df=86
38.0)
```

In [14]:
```
print(test_data(
    data=full_data,
    metrics_to_test=metric_list,
    field_to_test='District',
    test_to_use=stats.levene,
))
```

```
>>> <function levene at 0x1278dc550>
>> Number_of_Guards
> Мышеостровский-Краснокотейский равны по показателю Number_of_Guards:
TtestResult(statistic=0.8715041607006833, pvalue=0.38351527024669474, df=5
758.0)

> Мышеостровский-Мышесельский равны по показателю Number_of_Guards:
TtestResult(statistic=0.4825399632092622, pvalue=0.6294407596917191, df=57
58.0)

> Мышеостровский-Петрокотский равны по показателю Number_of_Guards:
TtestResult(statistic=-0.30863835552401464, pvalue=0.7576078063723227, df=
5758.0)

> Мышеостровский-Приморский равны по показателю Number_of_Guards:
TtestResult(statistic=0.32882672248510725, pvalue=0.7422986221006729, df=5
758.0)

> Мышеостровский-Невский равны по показателю Number_of_Guards:
TtestResult(statistic=0.03905434517787736, pvalue=0.9688484143849871, df=5
758.0)


>> Police_Units
> Мышеостровский-Краснокотейский равны по показателю Police_Units:
TtestResult(statistic=1.1454536349606426, pvalue=0.2520688965348275, df=57
58.0)

> Мышеостровский-Мышесельский равны по показателю Police_Units:
TtestResult(statistic=0.6022231939766808, pvalue=0.5470492918713652, df=57
58.0)
```

> Мышеостровский–Петрокотский равны по показателю Police_Units:
TtestResult(statistic=-0.3067142118950546, pvalue=0.7590720018236087, df=5758.0)

> Мышеостровский–Приморский равны по показателю Police_Units:
TtestResult(statistic=0.08678638459973438, pvalue=0.930844327986379, df=5758.0)

> Мышеостровский–Невский равны по показателю Police_Units:
TtestResult(statistic=0.1332212004697219, pvalue=0.8940231001557188, df=5758.0)

>> Percent_of_Crime_Solved
> Мышеостровский–Краснокотейский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.23448439779536875, pvalue=0.8146172815196955, df=5758.0)

> Мышеостровский–Мышесельский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=-0.1532499802938289, pvalue=0.8782065146470208, df=5758.0)

> Мышеостровский–Петрокотский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.30068319613350686, pvalue=0.7636669395827532, df=5758.0)

> Мышеостровский–Приморский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=-0.010761776050321675, pvalue=0.9914138836124542, df=5758.0)

> Мышеостровский–Невский равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.15121383403814914, pvalue=0.8798123349622053, df=5758.0)

>> Number_of_Lights
> Мышеостровский–Краснокотейский равны по показателю Number_of_Lights:
TtestResult(statistic=-0.8476611501080906, pvalue=0.3966619408712536, df=5758.0)

> Мышеостровский–Мышесельский равны по показателю Number_of_Lights:
TtestResult(statistic=0.8444025888717446, pvalue=0.39847955730860873, df=5758.0)

> Мышеостровский–Петрокотский равны по показателю Number_of_Lights:
TtestResult(statistic=0.2980367122362964, pvalue=0.7656858970387669, df=5758.0)

> Мышеостровский–Приморский равны по показателю Number_of_Lights:
TtestResult(statistic=0.6269577833380775, pvalue=0.5307117658183869, df=5758.0)

```
> Мышеостровский–Невский равны по показателю Number_of_Lights:
TtestResult(statistic=0.6081620018366941, pvalue=0.5431040206796555, df=57
58.0)
```

In [15]:
```
print(test_data(
    data=full_data,
    metrics_to_test=metric_list,
    field_to_test='Warehouse_Name',
    test_to_use=stats.levene,
))
```

```
>>> <function levene at 0x1278dc550>
>> Number_of_Guards
> Колбасовы–Молочковы равны по показателю Number_of_Guards:
TtestResult(statistic=-0.05987771723013768, pvalue=0.9522544122845377, df=
8638.0)

> Колбасовы–Мятновы равны по показателю Number_of_Guards:
TtestResult(statistic=-0.9886615638226333, pvalue=0.3228564463875877, df=8
638.0)

> Колбасовы–Сметанинковы равны по показателю Number_of_Guards:
TtestResult(statistic=0.19579113250347852, pvalue=0.8447782563751698, df=8
638.0)


>> Police_Units
> Колбасовы–Молочковы равны по показателю Police_Units:
TtestResult(statistic=0.2044125312672294, pvalue=0.8380359542352203, df=86
38.0)

> Колбасовы–Мятновы равны по показателю Police_Units:
TtestResult(statistic=-0.46918340065064884, pvalue=0.6389503909404572, df=
8638.0)

> Колбасовы–Сметанинковы равны по показателю Police_Units:
TtestResult(statistic=0.7209938018615215, pvalue=0.4709328403159081, df=86
38.0)


>> Percent_of_Crime_Solved
> Колбасовы–Молочковы равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.32859944819108805, pvalue=0.7424664306651769, df=8
638.0)

> Колбасовы–Мятновы равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=-0.10869818088869354, pvalue=0.9134444094976287, df=
8638.0)

> Колбасовы–Сметанинковы равны по показателю Percent_of_Crime_Solved:
TtestResult(statistic=0.09716204633791999, pvalue=0.9225999622041798, df=8
```

```
638.0)


>> Number_of_Lights
> Колбасовы–Молочковы равны по показателю Number_of_Lights:
TtestResult(statistic=1.471064906416363, pvalue=0.1413099908568232, df=863
8.0)

> Колбасовы–Мятновы равны по показателю Number_of_Lights:
TtestResult(statistic=0.5695557582658989, pvalue=0.5689938531140089, df=86
38.0)

> Колбасовы–Сметанинковы равны по показателю Number_of_Lights:
TtestResult(statistic=1.1114495872055554, pvalue=0.2664058122338372, df=86
38.0)
```

100%! Распределения по `District` и `Warehouse_Name` равны между собой по значениям среднего и СКО для всех показателей датасета (`'Number_of_Guards', 'Police_Units', 'Percent_of_Crime_Solved', 'Number_of_Lights'`).

Показывать результат для 5 полей по 24 парам `District, Warehouse_Name` я не буду, потому что он очень громоздкий (а код менее красивый), но и там несоответствия наблюдаются только в **6.52%** сравнений.

Замечание. Да, статистически корректно писать не "равны / не равны по показателю", а "не обнаружено доказательств / обнаружены доказательства того, что средние/СКО в группах различаются", но для быстроты чтения в выводе использован первый вариант.

In [16]:
```
def test_data_2(data, metrics_to_test, fields_to_test, test_to_use):
    res_str = ">>> {}\n".format(test_to_use)

    field_values_0 = data[fields_to_test[0]].unique()
    field_values_1 = data[fields_to_test[1]].unique()

    for metric in metric_list:
        res_str += '>> {}\n'.format(metric)
        for fv_0 in field_values_0:
            for fv_1 in field_values_1:
                if fv_0 == field_values_0[0] and field_values_1[0] == fv_
                    continue

                ttest_res = stats.ttest_ind(
                    data[(data[fields_to_test[0]] == field_values_0[0]) &
                    data[(data[fields_to_test[0]] == fv_0) & (data[fields
                )
```

```
                eq_string = ''
                if ttest_res.pvalue <= 0.05:
                    eq_string = 'НЕ '

                res_str += """> {}/{}-{}/{} {}равны по показателю {}:\n{}
                    field_values_0[0], field_values_1[0],
                    fv_0, fv_1,
                    eq_string,
                    metric,
                    ttest_res
                )
        res_str += '\n'

    return res_str

(test_data_2(
    data=full_data,
    metrics_to_test=metric_list,
    fields_to_test=('District', 'Warehouse_Name'),
    test_to_use=stats.ttest_ind
))

(test_data_2(
    data=full_data,
    metrics_to_test=metric_list,
    fields_to_test=('District', 'Warehouse_Name'),
    test_to_use=stats.levene
))
print(6.0 / 92.0)
```

```
0.06521739130434782
```

Попробуем вывести статистическую зависимость между процентом раскрытых преступлений и всеми остальными показателями с помощью линейной регрессии.

Для этого сначала сгененерируем one-hot encoding на строковые колонки `Warehouse_Name` и `District`:

In [17]:
```
warehouse_aliases = {
    'Колбасовы': 'KO',
    'Молочковы': 'MO',
    'Мятновы': 'MA',
    'Сметанинковы': 'SM'
}

for (k, v) in warehouse_aliases.items():
    full_data['is_W{}'.format(v)] = full_data['Warehouse_Name'].map(lambd
```

In [18]:
```
district_aliases = {
    'Мышеостровский': 'MO',
    'Краснокотейский': 'KK',
```

```
    'Мышесельский': 'MS',
    'Петрокотский': 'PK',
    'Приморский': 'PR',
    'Невский': 'NE'
}

for (k, v) in district_aliases.items():
    full_data['is_D{}'.format(v)] = full_data['District'].map(lambda x: i
    #Префикс D нужен для избежания пересечения с Warehouse_Name 'Молочков
```

In [19]:
```
(full_data
     [['is_WKO', 'is_WMO', 'is_WMA', 'is_WSM',
       'is_DMO', 'is_DKK', 'is_DMS', 'is_DPK', 'is_DPR', 'is_DNE']]
    .value_counts()
    .sort_index()
)
```

Out[19]:

is_WKO	is_WMO	is_WMA	is_WSM	is_DMO	is_DKK	is_DMS	is_DPK	is_DPR	is_DNE	
0	0	0	1	0	0	0	0	0	1	720
0	0	0	1	0	0	0	0	1	0	720
0	0	0	1	0	0	0	1	0	0	720
0	0	0	1	0	0	1	0	0	0	720
0	0	0	1	0	1	0	0	0	0	720
0	0	0	1	1	0	0	0	0	0	720
0	0	1	0	0	0	0	0	0	1	720
0	0	1	0	0	0	0	0	1	0	720
0	0	1	0	0	0	0	1	0	0	720
0	0	1	0	0	0	1	0	0	0	720
0	0	1	0	0	1	0	0	0	0	720
0	0	1	0	1	0	0	0	0	0	720
0	1	0	0	0	0	0	0	0	1	720
0	1	0	0	0	0	0	0	1	0	720
0	1	0	0	0	0	0	1	0	0	720
0	1	0	0	0	0	1	0	0	0	720
0	1	0	0	0	1	0	0	0	0	720
0	1	0	0	1	0	0	0	0	0	720
1	0	0	0	0	0	0	0	0	1	720
1	0	0	0	0	0	0	0	1	0	720
1	0	0	0	0	0	0	1	0	0	720
1	0	0	0	0	0	1	0	0	0	720
1	0	0	0	0	1	0	0	0	0	720
1	0	0	0	1	0	0	0	0	0	720

Name: count, dtype: int64

In [20]:
```
#Проверим все переменные.
```

```
model = ols(
    """Percent_of_Crime_Solved ~
        Number_of_Guards
        + Police_Units
        + Number_of_Lights
        + is_WKO
        + is_WMO
        + is_WMA
        + is_WSM
        + is_DMO
        + is_DKK
        + is_DMS
        + is_DPK
        + is_DPR
        + is_DNE""",
    data = full_data
).fit()

print(model.params)
model.summary()
```

```
Intercept           32.751650
Number_of_Guards    -0.126136
Police_Units         0.050563
Number_of_Lights     0.033262
is_WKO               8.107703
is_WMO               8.204614
is_WMA               8.216475
is_WSM               8.222859
is_DMO               5.449876
is_DKK               5.235693
is_DMS               5.628037
is_DPK               5.397317
is_DPR               5.547008
is_DNE               5.493719
dtype: float64
```

Out[20]:

OLS Regression Results

Dep. Variable:	Percent_of_Crime_Solved	**R-squared:**	0.217
Model:	OLS	**Adj. R-squared:**	0.217
Method:	Least Squares	**F-statistic:**	435.4
Date:	Fri, 09 Feb 2024	**Prob (F-statistic):**	0.00
Time:	22:48:34	**Log-Likelihood:**	-66288.
No. Observations:	17280	**AIC:**	1.326e+05
Df Residuals:	17268	**BIC:**	1.327e+05
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	32.7517	0.281	116.439	0.000	32.200	33.303
Number_of_Guards	-0.1261	0.104	-1.208	0.227	-0.331	0.078
Police_Units	0.0506	0.109	0.463	0.643	-0.163	0.265
Number_of_Lights	0.0333	0.000	69.143	0.000	0.032	0.034
is_WKO	8.1077	0.164	49.428	0.000	7.786	8.429
is_WMO	8.2046	0.163	50.286	0.000	7.885	8.524
is_WMA	8.2165	0.164	50.047	0.000	7.895	8.538
is_WSM	8.2229	0.163	50.324	0.000	7.903	8.543
is_DMO	5.4499	0.197	27.713	0.000	5.064	5.835
is_DKK	5.2357	0.197	26.588	0.000	4.850	5.622
is_DMS	5.6280	0.196	28.686	0.000	5.243	6.013
is_DPK	5.3973	0.197	27.454	0.000	5.012	5.783
is_DPR	5.5470	0.196	28.265	0.000	5.162	5.932
is_DNE	5.4937	0.196	27.973	0.000	5.109	5.879

Omnibus:	7680.994	**Durbin-Watson:**	0.475
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	92041.290
Skew:	-1.815	**Prob(JB):**	0.00
Kurtosis:	13.708	**Cond. No.**	7.32e+17

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.67e-26. This might indicate that there are strong multicollinearity problems or that the design matrix is singular.

In [21]:
```
#Согласно правилам ступенчатой регрессии, уберём Police_Units, так как св
model = ols(
    """Percent_of_Crime_Solved ~
        Number_of_Guards
        + Number_of_Lights
        + is_WKO
        + is_WMO
        + is_WMA
        + is_WSM
        + is_DMO
```

```
        + is_DKK
        + is_DMS
        + is_DPK
        + is_DPR
        + is_DNE""",
    data = full_data
).fit()

print(model.params)
model.summary()
```

```
Intercept           32.733842
Number_of_Guards    -0.080061
Number_of_Lights     0.033264
is_WKO               8.104134
is_WMO               8.200324
is_WMA               8.211587
is_WSM               8.217797
is_DMO               5.447083
is_DKK               5.231824
is_DMS               5.624764
is_DPK               5.394573
is_DPR               5.544994
is_DNE               5.490604
dtype: float64
```

Out[21]:

OLS Regression Results

Dep. Variable:	Percent_of_Crime_Solved	**R-squared:**	0.217
Model:	OLS	**Adj. R-squared:**	0.217
Method:	Least Squares	**F-statistic:**	478.9
Date:	Fri, 09 Feb 2024	**Prob (F-statistic):**	0.00
Time:	22:48:34	**Log-Likelihood:**	-66288.
No. Observations:	17280	**AIC:**	1.326e+05
Df Residuals:	17269	**BIC:**	1.327e+05
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	32.7338	0.279	117.481	0.000	32.188	33.280
Number_of_Guards	-0.0801	0.032	-2.534	0.011	-0.142	-0.018
Number_of_Lights	0.0333	0.000	69.151	0.000	0.032	0.034
is_WKO	8.1041	0.164	49.462	0.000	7.783	8.425
is_WMO	8.2003	0.163	50.342	0.000	7.881	8.520
is_WMA	8.2116	0.164	50.122	0.000	7.890	8.533

is_WSM	8.2178	0.163	50.407	0.000	7.898	8.537
is_DMO	5.4471	0.197	27.712	0.000	5.062	5.832
is_DKK	5.2318	0.197	26.593	0.000	4.846	5.617
is_DMS	5.6248	0.196	28.689	0.000	5.240	6.009
is_DPK	5.3946	0.197	27.453	0.000	5.009	5.780
is_DPR	5.5450	0.196	28.263	0.000	5.160	5.930
is_DNE	5.4906	0.196	27.974	0.000	5.106	5.875

Omnibus:	7681.189	**Durbin-Watson:**	0.475
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	92046.917
Skew:	-1.815	**Prob(JB):**	0.00
Kurtosis:	13.708	**Cond. No.**	7.58e+17

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.56e-26. This might indicate that there are strong multicollinearity problems or that the design matrix is singular.

In [22]:
```
#Уберём Number_of_Guards по той же причине.
model = ols(
    """Percent_of_Crime_Solved ~
        Number_of_Lights
        + is_WKO
        + is_WMO
        + is_WMA
        + is_WSM
        + is_DMO
        + is_DKK
        + is_DMS
        + is_DPK
        + is_DPR
        + is_DNE""",
    data = full_data
).fit()

print(model.params)
model.summary()
```

```
Intercept           32.395446
Number_of_Lights     0.033265
is_WKO               8.020521
is_WMO               8.116435
is_WMA               8.123397
is_WSM               8.135093
is_DMO               5.389344
is_DKK               5.179032
is_DMS               5.569779
is_DPK               5.335083
is_DPR               5.489119
is_DNE               5.433088
dtype: float64
```

Out[22]:

OLS Regression Results

Dep. Variable:	Percent_of_Crime_Solved	**R-squared:**	0.217
Model:	OLS	**Adj. R-squared:**	0.216
Method:	Least Squares	**F-statistic:**	531.2
Date:	Fri, 09 Feb 2024	**Prob (F-statistic):**	0.00
Time:	22:48:35	**Log-Likelihood:**	-66291.
No. Observations:	17280	**AIC:**	1.326e+05
Df Residuals:	17270	**BIC:**	1.327e+05
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	32.3954	0.245	132.458	0.000	31.916	32.875
Number_of_Lights	0.0333	0.000	69.141	0.000	0.032	0.034
is_WKO	8.0205	0.161	49.968	0.000	7.706	8.335
is_WMO	8.1164	0.160	50.881	0.000	7.804	8.429
is_WMA	8.1234	0.160	50.735	0.000	7.810	8.437
is_WSM	8.1351	0.160	50.923	0.000	7.822	8.448
is_DMO	5.3893	0.195	27.601	0.000	5.007	5.772
is_DKK	5.1790	0.196	26.469	0.000	4.796	5.563
is_DMS	5.5698	0.195	28.580	0.000	5.188	5.952
is_DPK	5.3351	0.195	27.342	0.000	4.953	5.718
is_DPR	5.4891	0.195	28.152	0.000	5.107	5.871
is_DNE	5.4331	0.195	27.864	0.000	5.051	5.815

Omnibus:	7685.622	**Durbin-Watson:**	0.474
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	92137.601
Skew:	-1.816	**Prob(JB):**	0.00
Kurtosis:	13.713	**Cond. No.**	9.22e+17

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 1.05e-26. This might indicate that there are strong multicollinearity problems or that the design matrix is singular.

In [23]:
```
#Всё "здорово", но что если мы уберём по одной dummy variable?
model = ols(
    """Percent_of_Crime_Solved ~
        Number_of_Lights
        + is_WMO
        + is_WMA
        + is_WSM
        + is_DKK
        + is_DMS
        + is_DPK
        + is_DPR
        + is_DNE""",
    data = full_data
).fit()

print(model.params)
model.summary()
```

```
Intercept          45.805311
Number_of_Lights    0.033265
is_WMO              0.095914
is_WMA              0.102876
is_WSM              0.114572
is_DKK             -0.210312
is_DMS              0.180435
is_DPK             -0.054261
is_DPR              0.099775
is_DNE              0.043744
dtype: float64
```

Out[23]:

OLS Regression Results

Dep. Variable:	Percent_of_Crime_Solved	**R-squared:**	0.217
Model:	OLS	**Adj. R-squared:**	0.216
Method:	Least Squares	**F-statistic:**	531.2
Date:	Fri, 09 Feb 2024	**Prob (F-statistic):**	0.00
Time:	22:48:35	**Log-Likelihood:**	-66291.
No. Observations:	17280	**AIC:**	1.326e+05
Df Residuals:	17270	**BIC:**	1.327e+05
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	45.8053	0.424	108.062	0.000	44.974	46.636
Number_of_Lights	0.0333	0.000	69.141	0.000	0.032	0.034
is_WMO	0.0959	0.241	0.397	0.691	-0.377	0.569
is_WMA	0.1029	0.241	0.426	0.670	-0.370	0.576
is_WSM	0.1146	0.241	0.475	0.635	-0.359	0.588
is_DKK	-0.2103	0.296	-0.711	0.477	-0.790	0.369
is_DMS	0.1804	0.296	0.610	0.542	-0.399	0.760
is_DPK	-0.0543	0.296	-0.184	0.854	-0.634	0.525
is_DPR	0.0998	0.296	0.337	0.736	-0.480	0.679
is_DNE	0.0437	0.296	0.148	0.882	-0.536	0.623

Omnibus:	7685.622	**Durbin-Watson:**	0.474
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	92137.601
Skew:	-1.816	**Prob(JB):**	0.00
Kurtosis:	13.713	**Cond. No.**	4.86e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 4.86e+03. This might indicate that there are strong multicollinearity or other numerical problems.

Всё посыпалось: dummy variables имеют статистическую значимость только если все 4 склада и 6 районов присутствуют в регрессии **одновременно**. Тот факт, что данные расределены по этим *46=24 парам равномерно создаёт ситуацию, в которой знание о том, в каком именно районе и на каком именно складе произошло преступление не даёт нам ровным счётом ничего. В регрессии они без какой-либо пользы перетягивают на себя части* `Intercept` *'а *(и гордо получают статистически значимую связь)*.

Попробуем самую простую регрессию из возможных:

In [24]:
```
model = ols(
    """Percent_of_Crime_Solved ~ Number_of_Lights""",
    data = full_data
).fit()

print(model.params)
model.summary()
```

```
Intercept           45.901904
Number_of_Lights     0.033253
dtype: float64
```

Out[24]:

OLS Regression Results

Dep. Variable:	Percent_of_Crime_Solved	**R-squared:**	0.217
Model:	OLS	**Adj. R-squared:**	0.217
Method:	Least Squares	**F-statistic:**	4780.
Date:	Fri, 09 Feb 2024	**Prob (F-statistic):**	0.00
Time:	22:48:35	**Log-Likelihood:**	-66292.
No. Observations:	17280	**AIC:**	1.326e+05
Df Residuals:	17278	**BIC:**	1.326e+05
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	45.9019	0.346	132.526	0.000	45.223	46.581
Number_of_Lights	0.0333	0.000	69.140	0.000	0.032	0.034

Omnibus:	7683.743	**Durbin-Watson:**	0.474
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	92078.626
Skew:	-1.816	**Prob(JB):**	0.00
Kurtosis:	13.710	**Cond. No.**	2.92e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.92e+03. This might indicate that there are strong multicollinearity or other numerical problems.

Чем богаты: количество фонарей объясняет 21.7% изменения процента раскрытых преступлений по уничтожению фонарей — каждый 1 добавленный фонарь ведёт к увеличению процента раскрытых преступлений на (целых) 0.033.

Ответы:

1. Проведённый разведочный анализ данных показал, что **датасет обладает абсолютной полнотой** (нет ни одного `null` значения), а данные в нём распределены равномерно и так, будто они были сгенерированы с одинаковыми показателями: проведённые t- и F-тесты не находят

доказательств неравенства средних и СКО между 4 складами и 6 районами, а количество тестов, которые нашли такие доказательства для 24 пар склад-район не превышает 6.52%.

(Также хочется заметить, что приведённое в условии наблюдение "в одном районе краж больше, чем в другом" не подтверждается при условии, что мы используем процент раскрытых преступлений по уничтожению фонарей как косвенный показатель краж.)

2. Увеличивать количество полицейских, чтобы снизить количество краж, **не нужно**, потому что не обнаружено статистической зависимости между их количеством и процентом раскрытых преступлений по уничтожению фонарей (которые, согласно гипотезе полицейских, связаны с кражами — проверить это мы не можем по причине отсутствия необходимых данных*, так что на данный момент согласимся с ними).

В качестве шуточной рекомендации хочется предложить *провести оптимизацию, уволив некоторое количество полицейских и охранников и потратив освободившиеся средства на установку фонарей* (так как это единственный показатель, с которым у процента раскрытых преступлений есть статистически значимая связь). Однако это остаётся шуткой, так как корреляция не есть каузация (и это предложение выглядит довольно смешно).

*Для того, чтобы ответить на этот вопрос точнее, мне бы хотелось иметь данные о количестве краж и проценте их раскрытия — я полагаю, если у полицейских есть данные по фонарям, то и по кражам что-то найдётся.