main

March 25, 2020

In [52]:
```
from ipyleaflet import Map, Marker, MarkerCluster
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

%matplotlib inline
```

In [53]:
```
def load_data(filename):
    df = pd.read_json(filename)
    df['crash_date'] = pd.to_datetime(df['crash_date'], format='%Y%m%d')
    return df
```

In [54]:
```
data_2018 = load_data('data/dtp-data-2018.json')
data_2017 = load_data('data/dtp-data-2017.json')
data_2017
```

Out[54]:

	reg_code	reg_name	road_code	\
0	83	-		
1	20			
2	79		-4	
3	3			
4	82		-215	
...	...		...	...
161132	33			
161133	40	-		
161134	36			
161135	80			
161136	118			

	road_name	\
0	-.	
1	- (), ...	
2	"" - - -- - ...	
3	.	
4	- - -	
...		...
161132		
161133		

```
161134
161135
161136

                                                road_type        oktmo  \
0           ...     83648415
1           ...     20631000
2                           79703000
3           ...     03703000
4                           82607000
...                                                   ...           ...
161132                                    33630154
161133
161134
161135                                 80647460101
161136                                    11851000

                                                  address  \
0       , - , ...
1            ,  ,
2                        ,  ,
3          ,  ,  ,
4          ,  ,
...                                                   ...
161132   ,  ,  ...
161133  -,  -, - ...
161134     ,  ,   , 21
161135   ,  ,  ...
161136    ,  -,  ...

             crash_type_name crash_date          crash_time  \
0                2018-01-01 2020-03-25 02:50:00
1               2018-01-01 2020-03-25 02:35:00
2          2018-01-01 2020-03-25 02:35:00
3                2018-01-01 2020-03-25 02:30:00
4             2018-01-01 2020-03-25 02:30:00
...                       ...        ...                 ...
161132           2017-01-01 2020-03-25 03:20:00
161133     2017-01-01 2020-03-25 03:10:00
161134        2017-01-01 2020-03-25 03:00:00
161135        2017-01-01 2020-03-25 03:00:00
161136     2017-01-01 2020-03-25 03:00:00

                                             crash_reason  fatalities_amount  \
0                                                1
1             ...                    1
2                                              2
3                                      1
```

```
4              ...                   0
...                                                ...                ...
161132                                                                    2
161133                                                                    3
161134                                                                    1
161135                                                                    1
161136                                                                    2

        victims_amount  vehicles_amount  participants_amount   latitude  \
0                    0                2                    6  43.589500
1                    0                1                    2  50.818300
2                    0                1                    3  44.825800
3                    0                2                    2  44.905600
4                    2                1                    3  43.315800
...                ...              ...                  ...        ...
161132               0                3                    4  58.577538
161133               0                1                    4  59.931700
161134               0                1                    2  53.346007
161135               0                1                    2  55.082500
161136               0                1                    2  67.639495

        longitude
0       43.200277
1       39.133100
2       39.224700
3       37.333100
4       47.430800
...           ...
161132  49.609588
161133  30.354700
161134  50.221799
161135  58.641100
161136  53.038521

[161137 rows x 17 columns]
```

In [55]:
```
def get_meta(data, year, df=None):
    if df is None:
        df = pd.DataFrame(columns=['Year', 'Accident count', 'Fatalities count', 'Vict
    df = df.append({
        'Year': year,
        'Accident count': len(data),
        'Fatalities count': sum(data['fatalities_amount']),
        'Victims amount': sum(data['victims_amount'])
    }, ignore_index=True)
    return df
```

In [56]:
```
meta = get_meta(data_2018, 2018)
meta = get_meta(data_2017, 2017, meta)
```

```
In [57]: def show_vics_count_histogram(data):
             vic = data.groupby(data['crash_date'].dt.month_name()).sum()['victims_amount'].re
         #     fat = data.groupby(data['crash_date'].dt.month_name()).sum()['fatalities_amount
             cnt = data.groupby(data['crash_date'].dt.month_name()).size().rename('Accidents c
             df = pd.concat([vic, cnt], axis=1)
             order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August'
             df = df.reindex(order, axis=0)
             df.plot(
                 kind='bar',
                 title='Accidents count and victims count by months',
                 figsize=(20,10),
                 grid=True
             )

In [58]: def show_count_daily_histogram(data):
             data.groupby(data['crash_time'].dt.hour).size().plot(
                 kind='bar',
                 title='Accidents count by hours',
                 figsize=(20,10),
                 grid=True
             )

In [59]: def show_region_count_histogram(data):
             df = data.groupby(data['reg_name']).size().sort_values()
             df.plot(
                 kind='bar',
                 title='Accidents count by regions',
                 figsize=(20,10),
                 grid=True
             )

In [60]: def show_crash_type_pie(data):
             df = data.groupby(data['crash_type_name']).size()
             ax = df.plot(
                 kind='pie',
                 figsize=(20,10),
                 autopct='%1.0f%%',
                 labels=None,
                 legend=True
             )
             ax.set_ylabel('')

In [61]: def show_accidents_with_vic_perc_pie(data):
             non_zero_count = np.count_nonzero(data['victims_amount'])
             zero_count = len(data['victims_amount']) - non_zero_count
             df = pd.Series([non_zero_count, zero_count], index=['With victims', 'Without vict
             ax = df.plot(
                 kind='pie',
                 figsize=(20,10),
```

```
            autopct='%1.0f%%',
            legend=True,
            title='Accidents with victims percentage'
        )
        ax.set_ylabel('')
```

```
In [62]: def show_map(data):
            center = (65.5240097, 105.3187561)
            m = Map(center=center, zoom=3)
            markers = []
            for x in data.itertuples():
                markers.append(Marker(location=(x.latitude, x.longitude)))
            marker_cluster = MarkerCluster(markers=markers)
            m.add_layer(marker_cluster);
            display(m)
```

Car accidents in 2017 and 2018 Data was taken from .. There are no data for 2019 and 2020 and it is why I took data for 2018 and 2017.

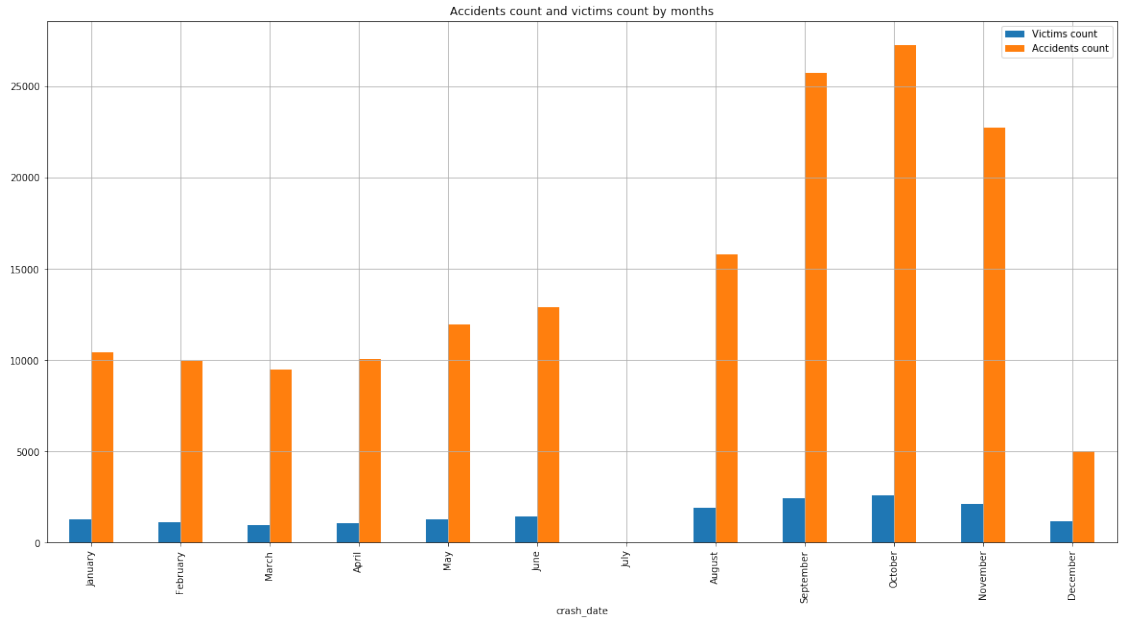
```
In [63]: meta
```

```
Out[63]:    Year Accident count Fatalities count Victims amount
         0  2018           48028             75072          11255
         1  2017          161137            207815          17244
```

In 2017 and 2018, 209,165 accidents occurred with 28,499 casualties, which is frightening.

0.0.1 Year 2017

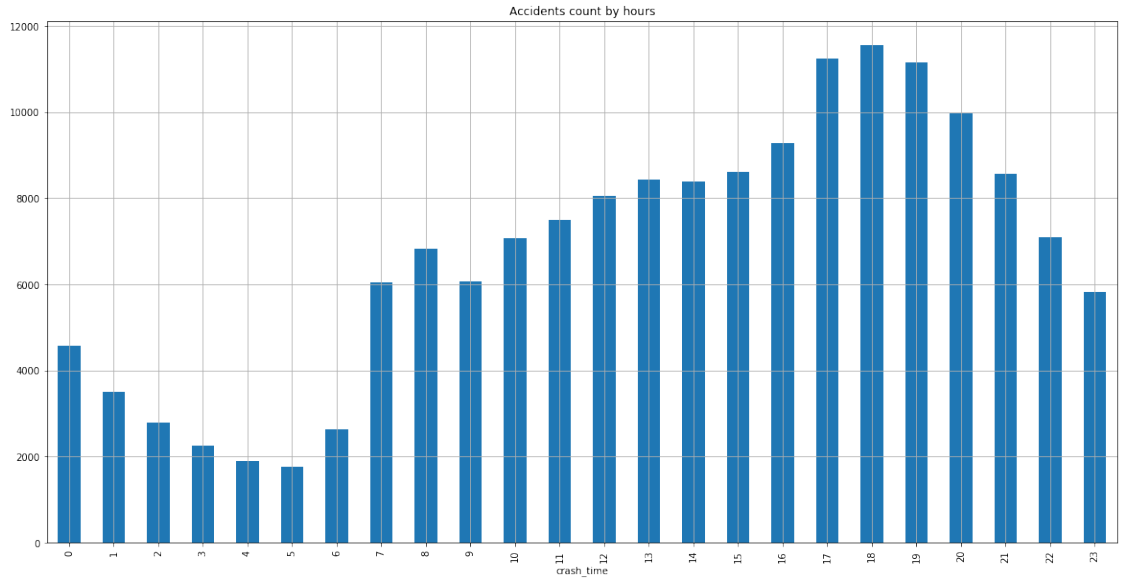
```
In [64]: show_vics_count_histogram(data_2017)
```



There is no info about July :(

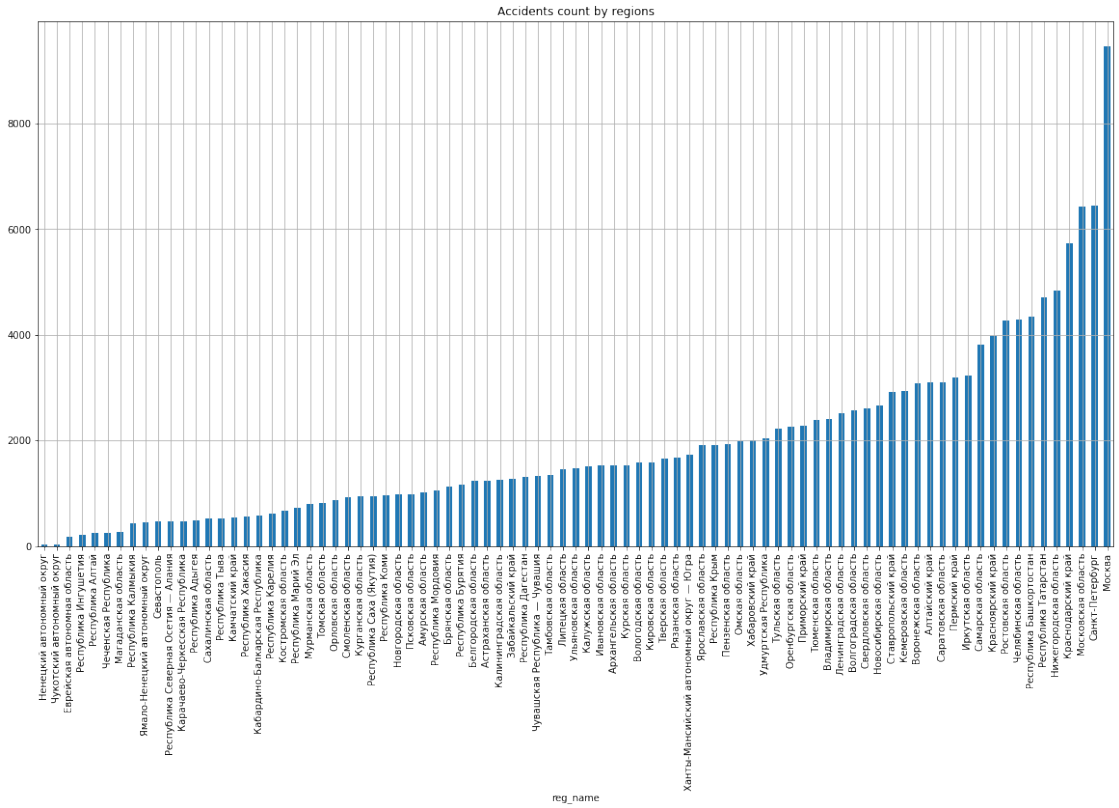
We can see that the most of accidents happens at Autumn (~25k/month), and ~10k/month accidens in other months, except December with ~5k accidents.

```
In [65]: show_count_daily_histogram(data_2017)
```



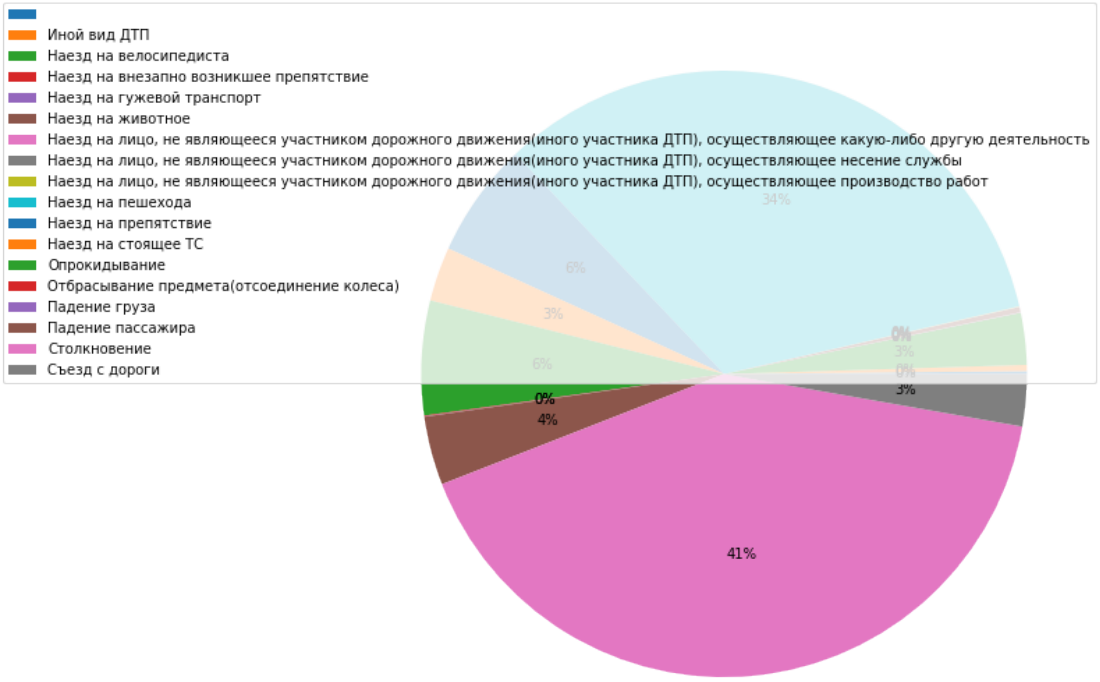
The most of accidents happens around 17:00-20:00 (when people return from work), then it slowly goes down and reaches a minimum at 05:00. We can see a huge difference between 06:00 and 07:00 (people wake up and go to work). Then it slowly grows from 06:00 to 19:00.

```
In [66]: show_region_count_histogram(data_2017)
```



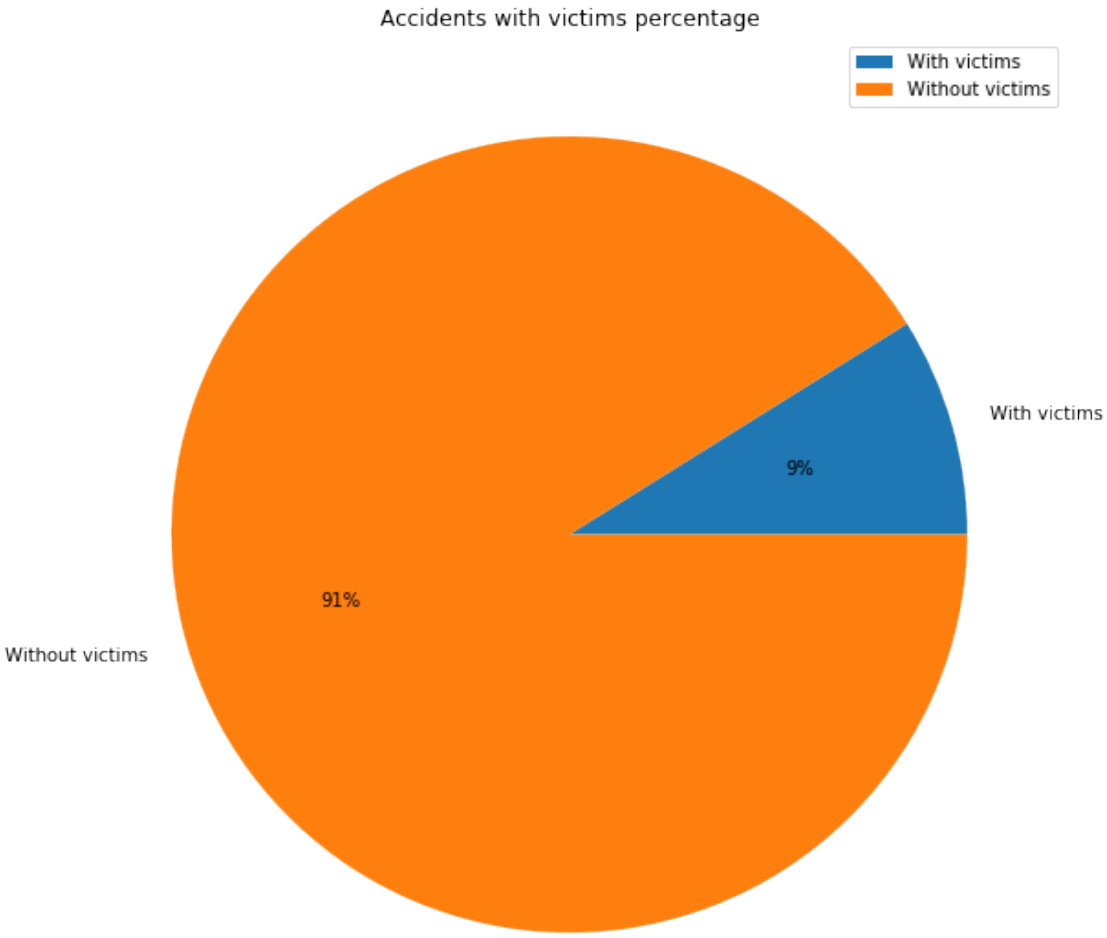
Most of acccidents happens in Moscow (Moscow region) and in Saint Petersburg. In these regions, the largest population density and the largest number of vehicles.

```
In [67]: show_crash_type_pie(data_2017)
```



We can highlight the most common types of accidents: car collision (41%) and pedestrian collision (34%).

```
In [68]: show_accidents_with_vic_perc_pie(data_2017)
```



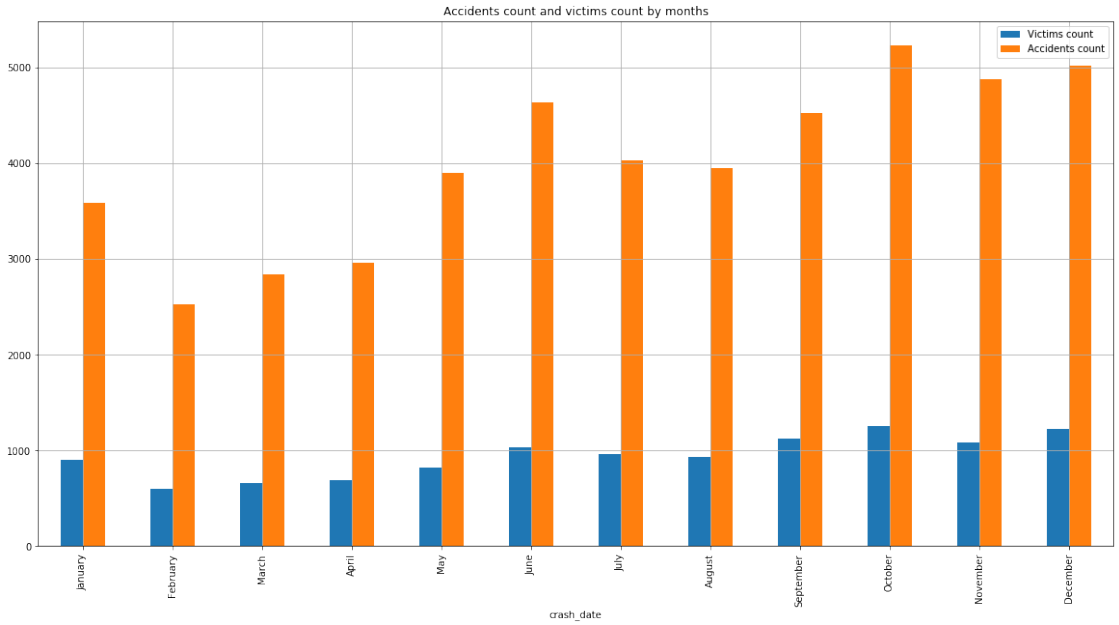
Every tenth (9%) accident has victims.

```
In [ ]: # show_map(data_2017) // works too slow with 160k markers
```

Map with accidents

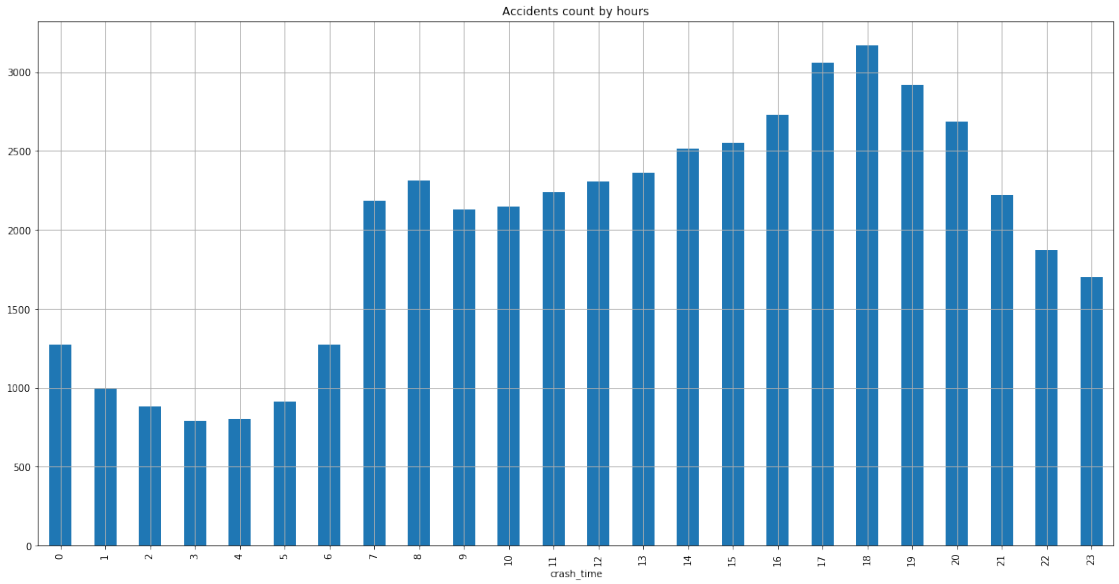
0.0.2 Year 2018

```
In [69]: show_vics_count_histogram(data_2018)
```



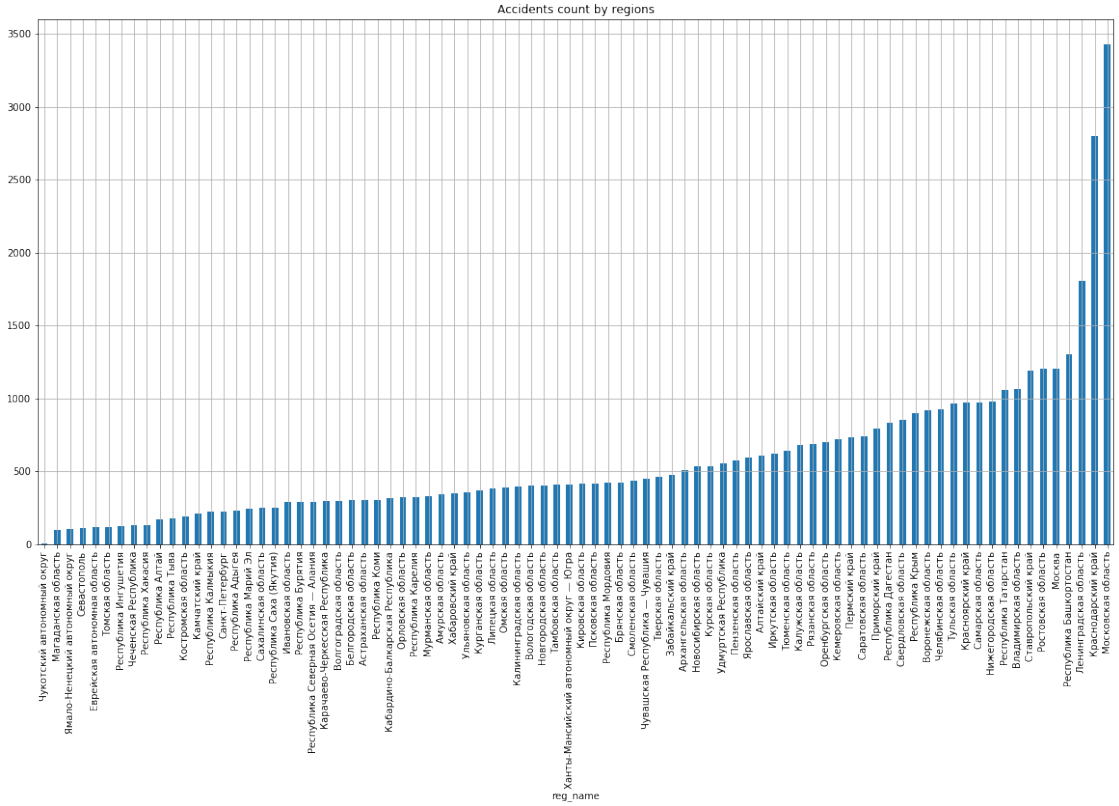
Most of accidents happpens in Autumn (as well as in 2017),

```
In [70]: show_count_daily_histogram(data_2018)
```



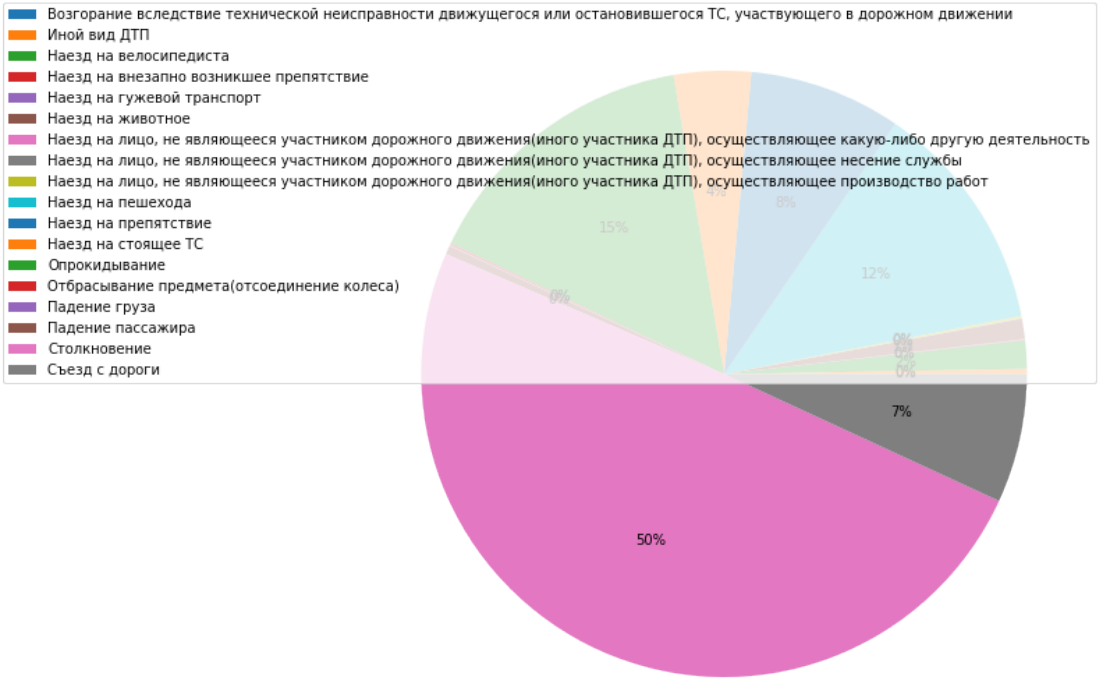
Almost the same as in 2017, most of accidents happens in 17:00-20:00, then it goes down until 05:00, and starts to grow after 05:00 with a big gap at 07:00.

```
In [71]: show_region_count_histogram(data_2018)
```

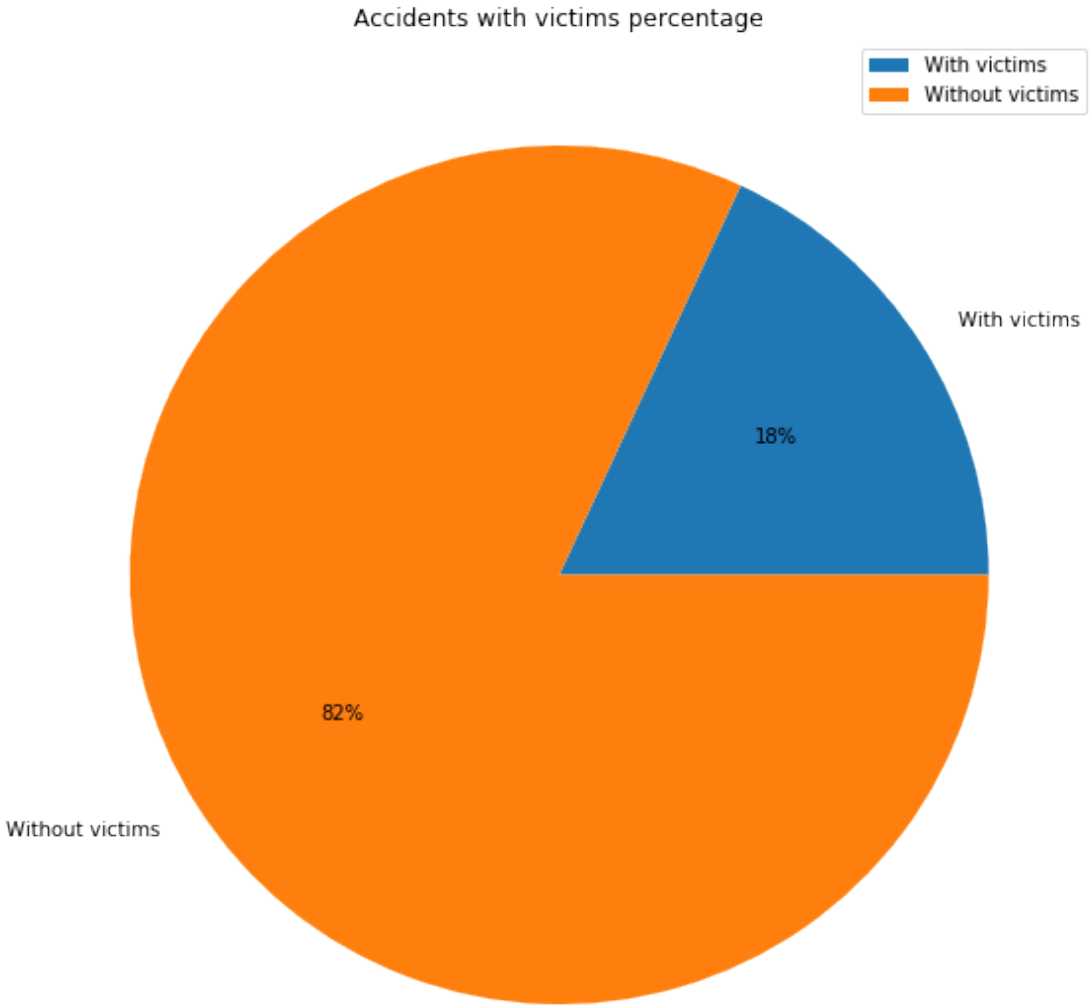


Most of accidents happens in Moscow, Krasnadar and Saint Petersburg.

```
In [72]: show_crash_type_pie(data_2018)
```



In [73]: show_accidents_with_vic_perc_pie(data_2018)



Every 5th (18%) accident has victims.

```
In [74]: show_map(data_2018)
```

```
Map(center=[65.5240097, 105.3187561], controls=(ZoomControl(options=['position', 'zoom_in_text
```

Map on which markers are applied corresponding to the accidents in 2018, it can be seen that the vast majority occurred in the European part of the country.

```
In [ ]:
```