In [1]:

```
#-*coding:utf-8 -*-

import pathlib
import random
from functools import reduce
from collections import defaultdict

import pandas as pd
import geopandas as gpd
import folium
import shapely
import numpy as np
from IPython.display import display
import matplotlib.pyplot as plt
from tqdm.notebook import tqdm
import xgboost
import sklearn.cluster
import tensorflow as tf
import matplotlib as mpl
import seaborn as sns

from pandas import DataFrame
from geoband import API
from folium import Map, CircleMarker, Vega, Popup
from vincent import Bar
import math
```

In [2]:

```
input_path = pathlib.Path('./input')
if not input_path.is_dir():
    input_path.mkdir()

from geoband.API import *
GetCompasData('SBJ_2102_001', '1', '1.  _  .csv')
GetCompasData('SBJ_2102_001', '2', '2.  _  .csv')
GetCompasData('SBJ_2102_001', '3', '3.  _ ( )_  .csv')
GetCompasData('SBJ_2102_001', '4', '4.  _   ( )_  .csv')
GetCompasData('SBJ_2102_001', '5', '5.  _   ( )_  .csv')
GetCompasData('SBJ_2102_001', '6', '6.  _  ( )_  .csv')
GetCompasData('SBJ_2102_001', '7', '7.  _ (  )_  .csv')
GetCompasData('SBJ_2102_001', '8', '8.  _   (  )_  .csv')
GetCompasData('SBJ_2102_001', '9', '9.  _   (  )_  .csv')
GetCompasData('SBJ_2102_001', '10', '10.  _  (  )_  .csv')
GetCompasData('SBJ_2102_001', '11', '11.  _   ( )_  .csv')
GetCompasData('SBJ_2102_001', '12', '12.  _ ( )_  .csv')
GetCompasData('SBJ_2102_001', '13', '13.  _ _  .csv')
GetCompasData('SBJ_2102_001', '14', '14.  _  .csv')
GetCompasData('SBJ_2102_001', '15', '15.  _  _  .csv')
GetCompasData('SBJ_2102_001', '16', '16.  _ _  .geojson')
GetCompasData('SBJ_2102_001', '17', '17.  _ _  .csv')
GetCompasData('SBJ_2102_001', '18', '18.  _   (2017~2020).csv')
GetCompasData('SBJ_2102_001', '19', '19.  _ _   _ .geojson')
GetCompasData('SBJ_2102_001', '20', '20.  _  .csv')
GetCompasData('SBJ_2102_001', '21', '21.  _  .csv')
GetCompasData('SBJ_2102_001', '22', '22.  _ _  .csv')
GetCompasData('SBJ_2102_001', '23', '23.  _   _ .geojson')
GetCompasData('SBJ_2102_001', '24', '24.  _    _ .geojson')
GetCompasData('SBJ_2102_001', '25', '25.  _ _ _   .csv')
GetCompasData('SBJ_2102_001', '26', '26.  _  .csv')
GetCompasData('SBJ_2102_001', '27', '27.  _ _  .csv')
GetCompasData('SBJ_2102_001', '28', '28.  _ _   _ .csv')
GetCompasData('SBJ_2102_001', '29', '29.  _  ( ).csv')
GetCompasData('SBJ_2102_001', '30', '30.  _  (  ).geojson')
GetCompasData('SBJ_2102_001', '31', '31.  _  (  ).geojson')
GetCompasData('SBJ_2102_001', '32', '32.  _  (  ).geojson')
GetCompasData('SBJ_2102_001', '33', '33.  _ .geojson')
GetCompasData('SBJ_2102_001', '34', '34.  _  _  .zip')

for path in list(input_path.glob('*.csv')) + list(input_path.glob('*.geojson')):
    print(path)
```

```
input/1.   _   .csv
input/10.   _    (   )_    .csv
input/11.   _     (  )_    .csv
input/12.   _  (  )_    .csv
input/13.   _   _    .csv
input/14.   _    .csv
input/15.   _    _    .csv
input/17.   _   _    .csv
input/18.   _      (2017~2020).csv
input/2.   _   .csv
input/20.   _    .csv
input/21.   _    .csv
input/22.   _   _    .csv
input/28.   _   _     _   .csv
input/25.   _  _   _     .csv
input/26.   _    .csv
input/27.   _   _    .csv
input/29.   _    (  ).csv
input/3.   _   (  )_    .csv
input/4.   _     (  )_    .csv
input/5.   _     (  )_    .csv
input/6.   _    (  )_    .csv
input/7.   _   (   )_    .csv
input/8.   _     (   )_    .csv
input/9.   _     (   )_    .csv
input/16.   _   _    .geojson
input/19.   _   _      _  .geojson
input/23.   _     _  .geojson
input/24.   _      _  .geojson
input/30.   _    (   ).geojson
input/31.   _    (   ).geojson
input/32.   _    (   ).geojson
input/33.   _   .geojson
```

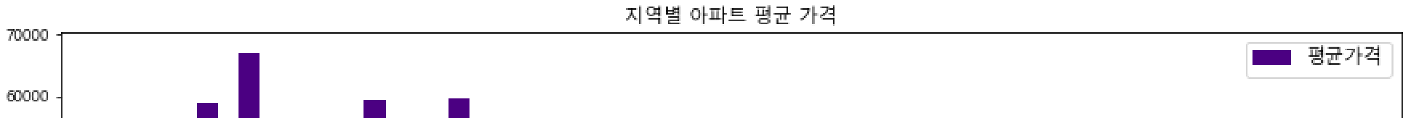
In [3]:

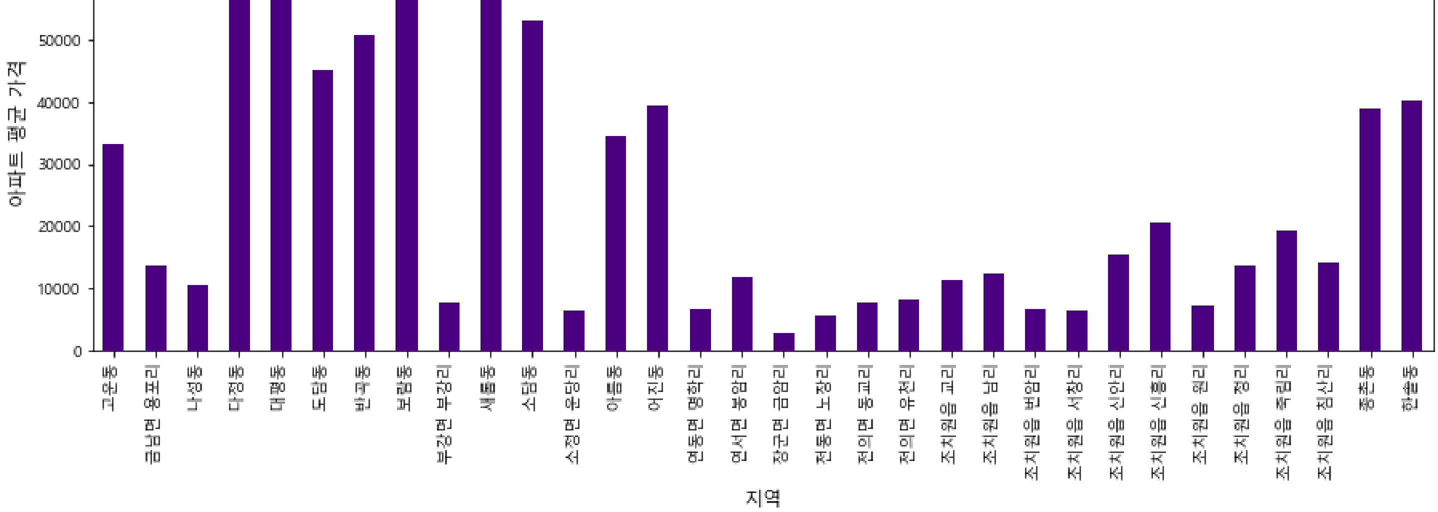
```
#
  _ = pd.read_csv(input_path.joinpath('3.   _   (  )_    .csv'), encoding='cp949' )
  _

#
for i in range(len(   _ ['   '])):
      _ ['   '][i] =     _ ['   '][i][7:]

# col
  _  .columns

#
data_    =   _ .groupby('    ')['    (  )'].mean()
data_

#
import matplotlib
matplotlib.font_manager._rebuild()
plt.rc("font", family="Malgun Gothic")

#
ax = data_   .plot(kind ='bar', title ='             ', figsize =(15, 5), legend=True, fontsize = 10, color='indigo')
ax.set_xlabel('   ', fontsize = 12)
ax.set_ylabel('         ', fontsize = 12)
ax.legend(['     '], fontsize=12)
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  import sys
```

Out[3]:

```
<matplotlib.legend.Legend at 0x7f062813a6a0>
```





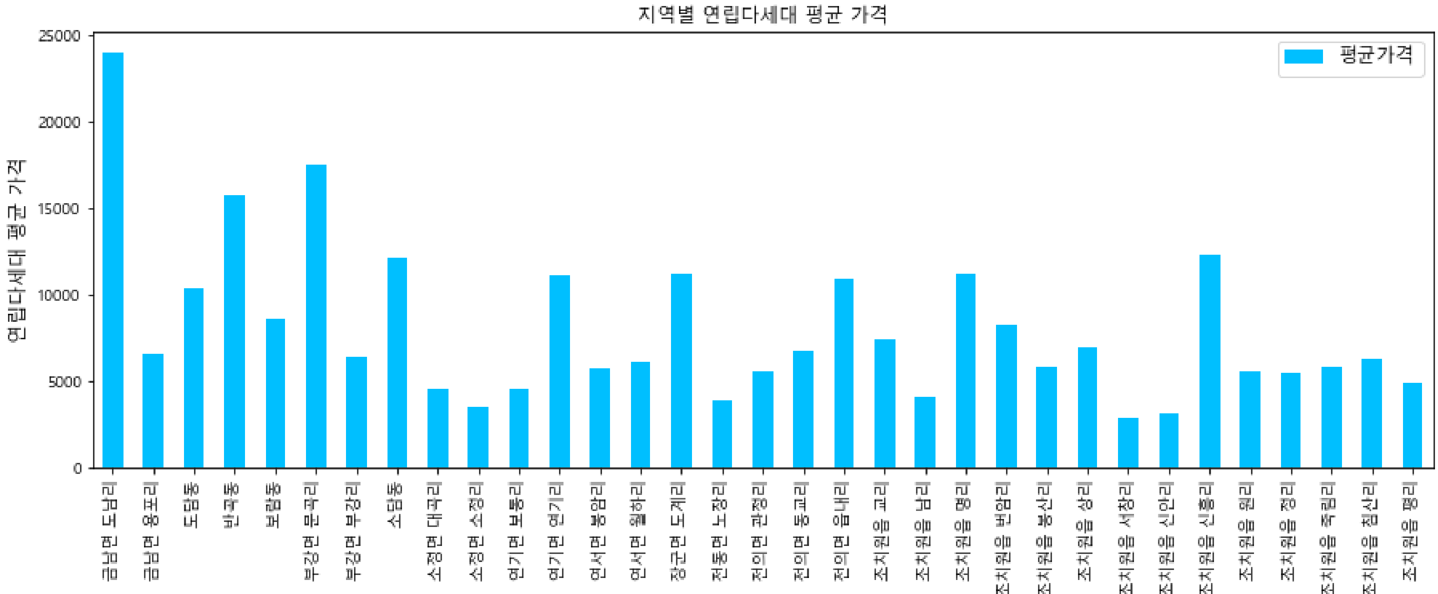
In [4]:

```
#
    _  = pd.read_csv(input_path.joinpath('4.   _     (  )_     .csv'), thousands = ',',encoding='cp949' )
    _

#
for i in range(len(     _  ['    '])):
        _  ['    '][i] =       _  ['    '][i][7:]

# col
    _  .columns

#
data_      =       _  .groupby('    ')['    (  )'].mean()
data_

#
import matplotlib
matplotlib.font_manager._rebuild()
plt.rc("font", family="Malgun Gothic")

#
ax = data_     .plot(kind ='bar', title ='                ', figsize =(15, 5), legend=True, fontsize = 10, color='deepskyblue')
ax.set_xlabel('    ', fontsize = 12)
ax.set_ylabel('            ', fontsize = 12)
ax.legend(['     '], fontsize=12)
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:7: SettingWithCopyWarning: 
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  import sys
```

Out[4]:

```
<matplotlib.legend.Legend at 0x7f057f778f60>
```



지역

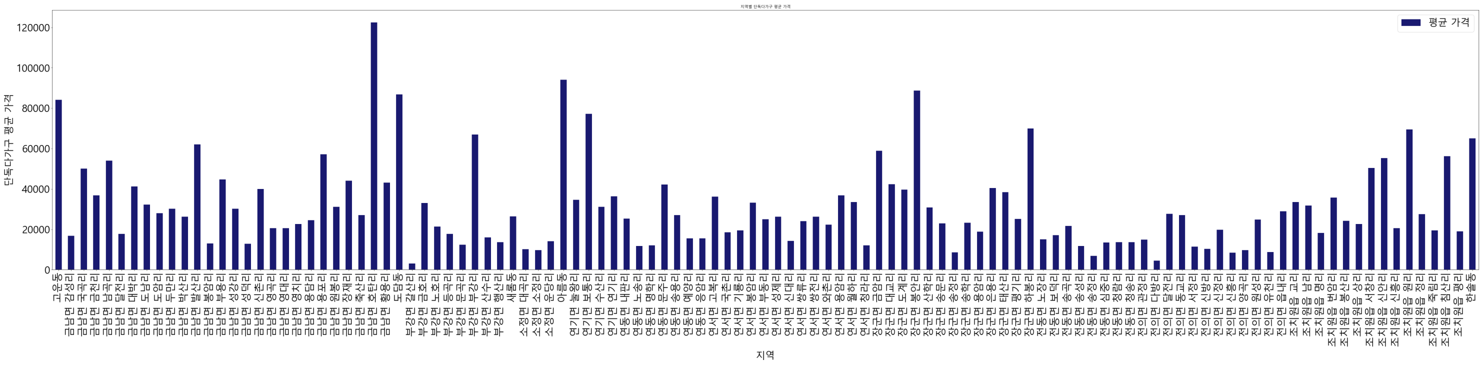
In [5]:

```
#
  _  = pd.read_csv(input_path.joinpath('5.   _      (  )_     .csv'),encoding='cp949' )
  _

#
for i in range(len(     _  ['    '])):
        _  ['    '][i] =       _  ['    '][i][7:]

# col
  _   .columns

#
data_      =      _   .groupby('    ')['     (   )'].mean()
data_

#
import matplotlib
matplotlib.font_manager._rebuild()
plt.rc("font", family="Malgun Gothic")

#
ax = data_     .plot(kind='bar', title='                ', figsize=(80,15), legend=True, fontsize=32, color='midnightblue')
ax.set_xlabel('   ',fontsize=30)
ax.set_ylabel('            ', fontsize=30)
ax.legend(['     '],fontsize=30)
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:7: SettingWithCopyWarning: 
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  import sys
```

Out[5]:

```
<matplotlib.legend.Legend at 0x7f057f4d6400>
```



In [6]:

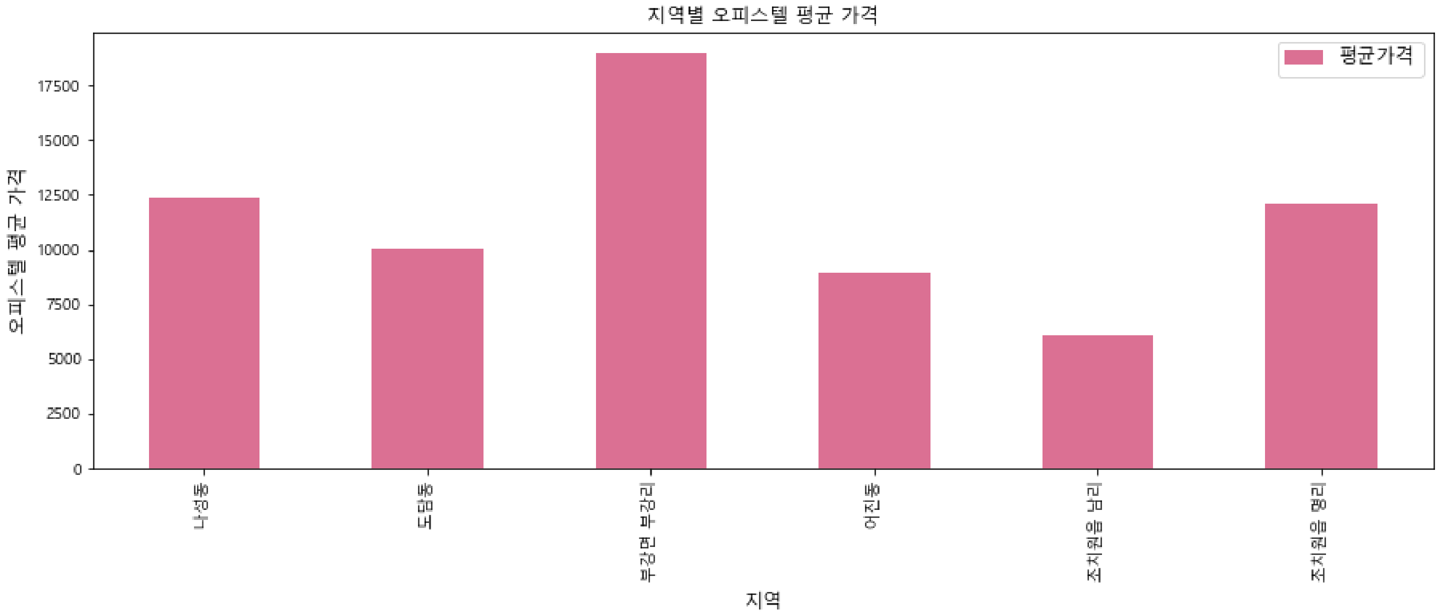
```
#
  _  = pd.read_csv(input_path.joinpath('6.   _      (  )_     .csv'),encoding='cp949' )
  _

#
for i in range(len(     _  ['    '])):
        _  ['    '][i] =       _  ['    '][i][7:]

# col
  _   .columns

#
data_      =      _   .groupby('    ')['     (   )'].mean()
data_

#
import matplotlib
matplotlib.font_manager._rebuild()
plt.rc("font", family="Malgun Gothic")

#
ax = data_     .plot(kind ='bar', title ='                ', figsize =(15, 5), legend=True, fontsize = 10, color='palevioletred')
```

```
ax.set_xlabel('  ', fontsize = 12)
ax.set_ylabel('         ', fontsize = 12)
ax.legend(['    '], fontsize=12)
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:7: SettingWithCopyWarning: 
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  import sys
```

Out[6]:

```
<matplotlib.legend.Legend at 0x7f057f4db7f0>
```



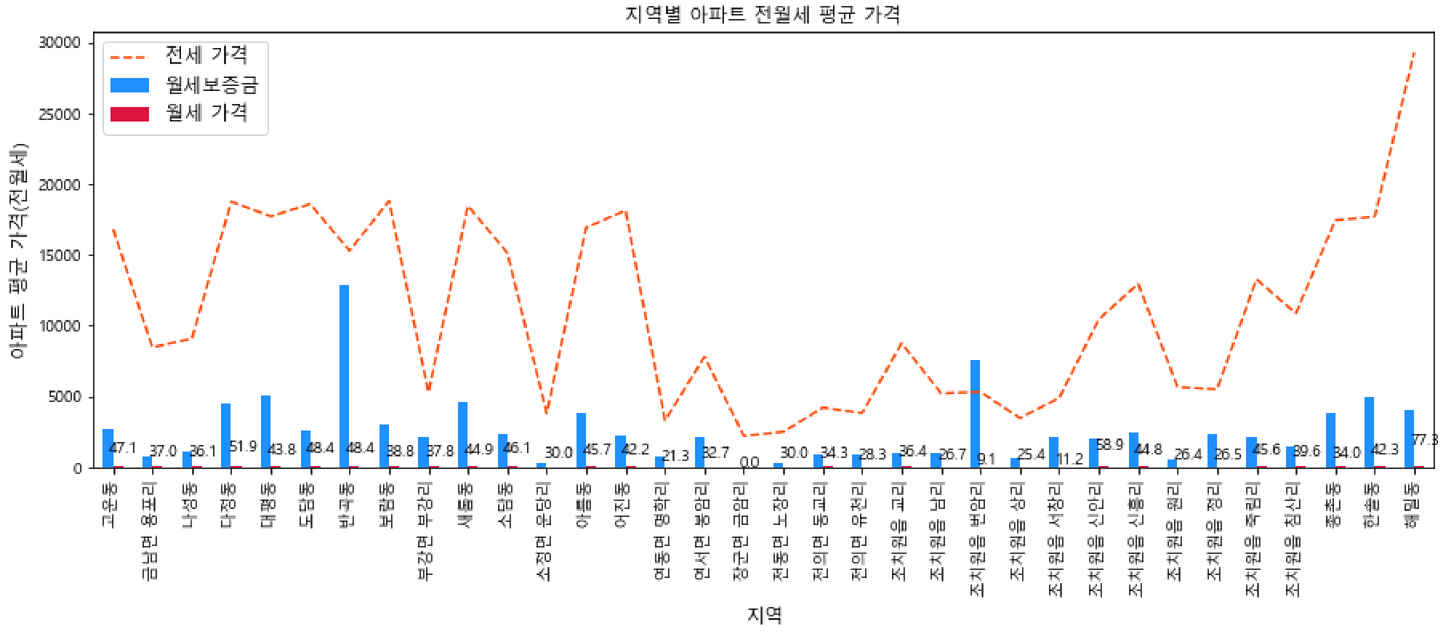
In [ ]:

,

In [7]:

```
#
  _   = pd.read_csv(input_path.joinpath('7.   _   (   )_    .csv'),thousands = ',',encoding='cp949' )
  _

#
  _   .columns

#
for i in range(len(   _   ['   '])):
      _   ['   '][i] =    _   ['   '][i][7:]

#      ,
data_   _      =    _   .groupby(['    ', '     '])[['   (  )', '  (  )']].mean()
data_   _      = data_   _    .unstack().fillna(0)
data_   _

data_   _   _       = data_   _    ['   (  )','  ']
data_   _   _    = data_   _    ['  (  )','  ']
data_   _    = data_   _    ['   (  )','  ']

#data_   _   ['  (  )','  '] = data_   _
del(data_   _   ['  (  )','  '])
del(data_   _   ['   (  )','  '])

round(data_   _    ,2)

#
matplotlib.font_manager._rebuild()
plt.rc("font", family="Malgun Gothic")

#
ax = data          .plot(kind ='bar', title ='                ', figsize =(15, 5), legend=True, fontsize = 10, color=['dodgerblue', 'crimson'])
ax = data_  _  .plot(kind ='bar', title = '               ', figsize =(15, 5), legend=True, fontsize = 10, color=['dodgerblue', 'crimson'])
plt.plot(data_  _  , linestyle='--', color = 'orangered')
ax.set_xlabel('  ', fontsize = 12)
ax.set_ylabel('        (   )', fontsize = 12)
ax.legend(['     ','     ', '     '], fontsize=12)

for i, v in enumerate(data_  _   ['  (  )','  ']):
    ax.text(i-0.1, v+v*20, str(round(v, 1)))
```

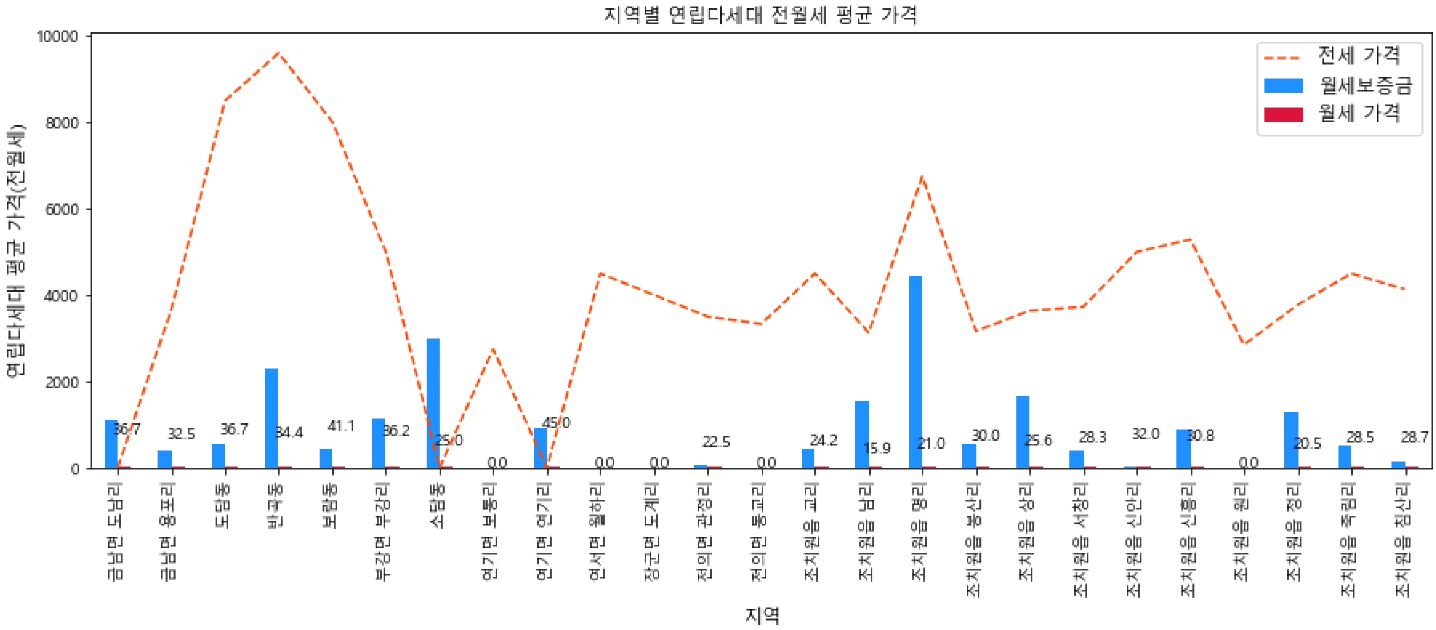
```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # Remove the CWD from sys.path while we load stuff.
```



In [8]:

```
#
    _    = pd.read_csv(input_path.joinpath('8.   _     (   )_     .csv'),thousands = ',',encoding='cp949' )
    _

#
    _   .columns

#
for i in range(len(     _   ['   '])):
        _   ['   '][i] =       _   ['   '][i][7:]

#      ,
data_    _     =      _    .groupby(['    ', '    '])[['   (  )', '   (  )']].mean()
data_    _     = data_    _     .unstack().fillna(0)
data_    _

data_    _   _       = data_     _    ['   (  )','   ']
data_    _   _    = data_     _   ['  (  )','   ']
data_    _    = data_     _    ['   (  )','   ']

#data_      _   ['   (  )','   '] = data_       _
del(data_      _   ['  (  )','   '])
del(data_      _   ['   (  )','   '])

round(data_      _     ,2)

#
matplotlib.font_manager._rebuild()
plt.rc("font", family="Malgun Gothic")

#
ax = data_     _    .plot(kind ='bar', title ='                  ', figsize =(15, 5), legend=True, fontsize = 10, color=[ 'dodgerblue', 'crimson'])
plt.plot(data_     _    , linestyle='--', color = 'orangered')
ax.set_xlabel('  ', fontsize = 12)
ax.set_ylabel('          (    )', fontsize = 12)
ax.legend(['     ','     ', '     '], fontsize=12)

for i, v in enumerate(data_      _    ['  (  )','  ']):
    ax.text(i-0.1, v+v*20, str(round(v, 1)))
```

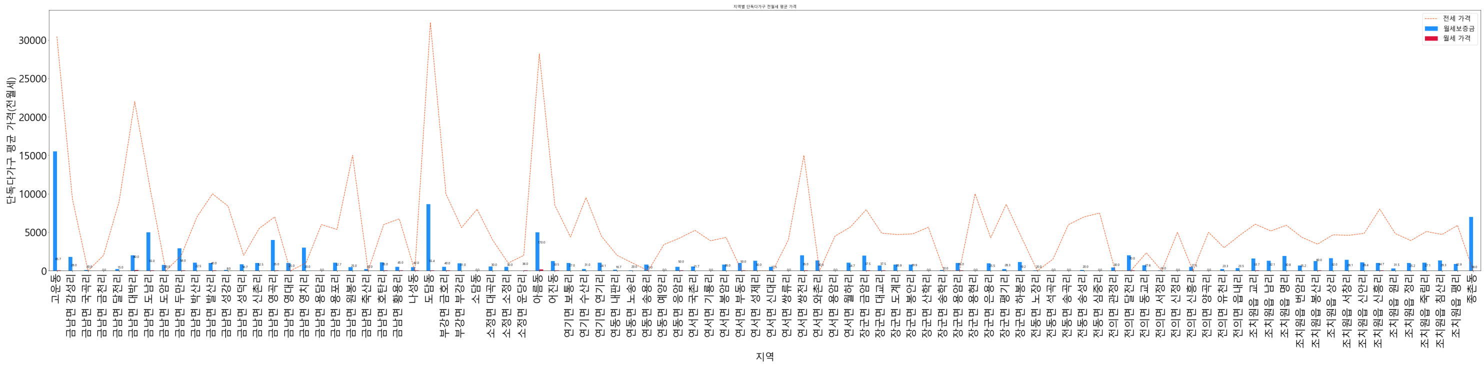
```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # Remove the CWD from sys.path while we load stuff.
```



In [9]:

```
#
    _   = pd.read_csv(input_path.joinpath('9.   _    (   )_    .csv'),thousands = ',',encoding='cp949' )
    _

#
    _   .columns

#
for i in range(len(      _   ['   '])):
        _   ['   '][i] =       _   ['   '][i][7:]

#       ,
data_    _    =      _   .groupby(['    ', '     '])[['   (  )', '   (  )']].mean()
data_    _    = data_    _    .unstack().fillna(0)
data_    _

data_    _  _      = data_     _   ['   (  )','   ']
data_    _  _   = data_     _   ['  (  )','   ']
data_    _   = data_     _   ['   (  )','   ']

#data_     _   ['   (  )','   '] = data_     _
del(data_      _   ['  (  )','   '])
del(data_      _   ['   (  )','   '])

round(data_     _    ,2)

#
matplotlib.font_manager._rebuild()
plt.rc("font", family="Malgun Gothic")

#
ax = data_     _    .plot(kind ='bar', title ='                    ', figsize =(80, 15), legend=True, fontsize = 32, color=[ 'dodgerblue', 'crimson'])
plt.plot(data_     _   , linestyle='--', color = 'orangered')
ax.set_xlabel('   ', fontsize = 30)
ax.set_ylabel('            (   )', fontsize = 30)
ax.legend(['     ','      ', '      '], fontsize=20)

for i, v in enumerate(data_      _   ['  (  )','   ']):
    ax.text(i-0.1, v+v*20, str(round(v, 1)))
```

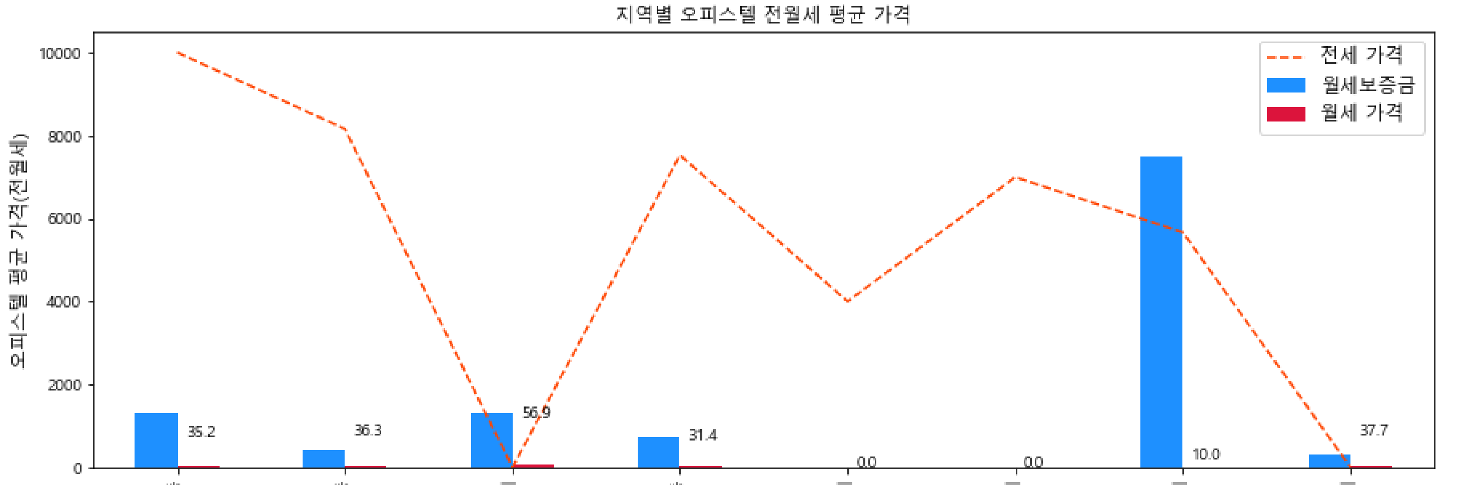
```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # Remove the CWD from sys.path while we load stuff.
```

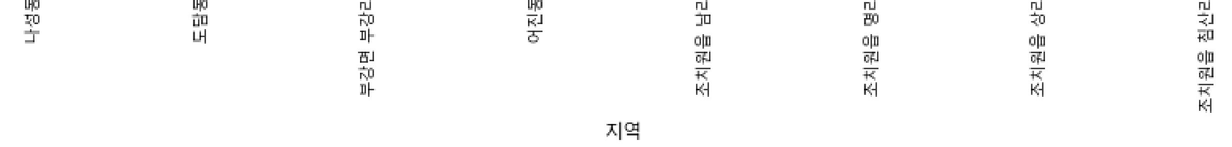


In [10]:

```
#
    _    = pd.read_csv(input_path.joinpath('10.   _   (  )_    .csv'),thousands = ',',encoding='cp949' )
    _

#
    _    .columns

#
for i in range(len(    _   ['   '])):
        _   ['   '][i] =     _   ['   '][i][7:]

#      ,
data_    _     =    _   .groupby(['   ', '     '])[['   (  )', '  (  )']].mean()
data_    _     = data_    _   .unstack().fillna(0)
data_    _

data_    _    _      = data_    _    ['   (  )',' ']
data_    _    _    = data_    _    ['  (  )',' ']
data_    _    = data_    _    ['  (  )',' ']

#data_    _   ['  (  )','  '] = data_    _
del(data_    _   ['  (  )','  '])
del(data_    _   ['   (  )','  '])

round(data_    _    ,2)

#
matplotlib.font_manager._rebuild()
plt.rc("font", family="Malgun Gothic")

#
ax = data_    _   .plot(kind ='bar', title ='                    ', figsize =(15, 5), legend=True, fontsize = 10, color=[ 'dodgerblue', 'crimson'])
plt.plot(data_    _   , linestyle='--', color = 'orangered')
ax.set_xlabel('   ', fontsize = 12)
ax.set_ylabel('          (   )', fontsize = 12)
ax.legend(['     ','      ', '     '], fontsize=12)

for i, v in enumerate(data_    _   ['  (  )','  ']):
    ax.text(i+0.05, v+v*20, str(round(v, 1)))
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # Remove the CWD from sys.path while we load stuff.
```





In [ ]:

```

```

geojson

In [11]:

```
  _    = gpd.read_file(input_path.joinpath('19.  _  _     _ .geojson'))
  _
```

Out[11]:

	gid	201710_20 _	201810_20 _	201910_20 _	202010_20 _	201710_30 _	201810_30 _	201910_30 _	202010_30 _	201710_40 _	...	202010_80 _	201710_90 _	20
0	667567	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
1	667568	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
2	668566	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
3	668567	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
4	668568	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
47391	919332	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
47392	919333	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
47393	919334	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
47394	920332	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	
47395	920333	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	

47396 rows × 38 columns

In [12]:

```
    = gpd.read_file(input_path.joinpath('23.  _     _ .geojson'))
```

Out[12]:

	BDTYP_CD	BULD_NM	BULD_NM_DC	BULD_SE_CD	BUL_MAN_NO	EMD_CD	GRO_FLO_CO	LNBR_MNNM	LNBR_SLNO	UND_FLO_CO	
											MUL

```
       BDTYP_CD BULD_NM BULD_NM_DC BULD_SE_CD BUL_MAN_NO EMD_CD GRO_FLO_CO LNBR_MNNM LNBR_SLNO UND_FLO_CO
    0     10299    None       None          0      55874    350          1       177         4          0
                                                                                                          MUL
    1     10299    None       None          0      55795    340          1        42        23          0
                                                                                                          MUL
    2     04401    None       None          0      55266    107          7       723         0          2
                                                                                                          MUL
    3     04001    None       None          0      60761    107          9       722         0          3
                                                                                                          MUL
    4     03999    None       None          0      55331    107          8       721         0          3
  ...       ...     ...        ...        ...        ...    ...        ...       ...       ...        ...
                                                                                                          MUL
54952     01001    None       None          0      36537    340          1       135         0          0
                                                                                                          MUL
54953     14001    None       None          0      40633    340          1       135         0          0
                                                                                                          MUL
54954     01001    None       None          0      19892    340          1       135         0          0
                                                                                                          MUL
54955     14001    None       None          0      19893    340          1       135         0          0
                                                                                                          MUL
54956     01001   119-1       None          0      54174    340          1       119         1          0

54957 rows × 11 columns
```

In [13]:

```
   = gpd.read_file(input_path.joinpath('24.  _    _ .geojson'))
```

Out[13]:

```
          gid   val                                         geometry
    0  667567   NaN  MULTIPOLYGON (((127.12716 36.70773, 127.12716
                                                                 ...
    1  667568   NaN  MULTIPOLYGON (((127.12716 36.70863, 127.12716
                                                                 ...
    2  668566   NaN  MULTIPOLYGON (((127.12829 36.70683, 127.12828
                                                                 ...
    3  668567   NaN  MULTIPOLYGON (((127.12828 36.70773, 127.12828
                                                                 ...
    4  668568   NaN  MULTIPOLYGON (((127.12828 36.70863, 127.12828
                                                                 ...
  ...     ...   ...                                              ...
47391  919332   NaN  MULTIPOLYGON (((127.40956 36.49642, 127.40956
                                                                 ...
47392  919333   NaN  MULTIPOLYGON (((127.40956 36.49732, 127.40956
                                                                 ...
47393  919334   NaN  MULTIPOLYGON (((127.40956 36.49823, 127.40955
                                                                 ...
47394  920332   NaN  MULTIPOLYGON (((127.41067 36.49642, 127.41067
                                                                 ...
47395  920333   NaN  MULTIPOLYGON (((127.41067 36.49732, 127.41067
                                                                 ...

47396 rows × 3 columns
```

In [14]:

```
 _    = gpd.read_file(input_path.joinpath('30.  _   (  ).geojson'))
 _
```

Out[14]:

	SIG_CD	SIG_KOR_NM	geometry
0	36110		MULTIPOLYGON (((127.17841 36.59687, 127.17839 ...

In [15]:

```
 _    = gpd.read_file(input_path.joinpath('31.  _   (  ).geojson'))
 _
```

Out[15]:

	EMD_CD	EMD_KOR_NM	geometry
0	36110101		MULTIPOLYGON (((127.30833 36.47977, 127.30818 ...
1	36110102		MULTIPOLYGON (((127.30480 36.48845, 127.30480 ...
2	36110103		MULTIPOLYGON (((127.29383 36.48508, 127.29381 ...
3	36110104		MULTIPOLYGON (((127.27215 36.46388, 127.27197 ...
4	36110105		MULTIPOLYGON (((127.24631 36.46300, 127.24626 ...
5	36110106		MULTIPOLYGON (((127.25661 36.47033, 127.25653 ...
6	36110107		MULTIPOLYGON (((127.26603 36.49182, 127.26744 ...
7	36110108		MULTIPOLYGON (((127.23978 36.48556, 127.23966 ...
8	36110109		MULTIPOLYGON (((127.24003 36.48809, 127.24004 ...
9	36110110		MULTIPOLYGON (((127.25544 36.49534, 127.25543 ...
10	36110111		MULTIPOLYGON (((127.25357 36.50688, 127.25357 ...
11	36110112		MULTIPOLYGON (((127.23895 36.53520, 127.23960 ...
12	36110113		MULTIPOLYGON (((127.25238 36.51029, 127.25218 ...
13	36110114		MULTIPOLYGON (((127.25357 36.50688, 127.25348 ...
14	36110115		MULTIPOLYGON (((127.26859 36.53949, 127.26858 ...
15	36110116		MULTIPOLYGON (((127.27076 36.52337, 127.27005 ...
16	36110117		MULTIPOLYGON (((127.34569 36.51897, 127.34628 ...
17	36110118		MULTIPOLYGON (((127.31732 36.48175, 127.31730 ...
18	36110250		MULTIPOLYGON (((127.30663 36.60062, 127.30662 ...
19	36110310		MULTIPOLYGON (((127.28524 36.55518, 127.28605 ...
20	36110320		MULTIPOLYGON (((127.32202 36.58302, 127.32209 ...
21	36110330		MULTIPOLYGON (((127.36891 36.48959, 127.36891 ...
22	36110340		MULTIPOLYGON (((127.36098 36.49088, 127.36059 ...
23	36110350		MULTIPOLYGON (((127.23952 36.48582, 127.23966 ...
24	36110360		MULTIPOLYGON (((127.19288 36.60110, 127.19298 ...

	EMD_CD	EMD_KOR_NM	geometry
25	36110370		MULTIPOLYGON (((127.19609 36.60448, 127.19604 ...
26	36110380		MULTIPOLYGON (((127.27948 36.63350, 127.27948 ...
27	36110390		MULTIPOLYGON (((127.20791 36.71901, 127.20781 ...

In [16]:

```
_    = gpd.read_file(input_path.joinpath('32.  _   (   ).geojson'))
_
```

Out[16]:

	ADM_DR_CD	ADM_DR_NM	geometry
0	2901011		MULTIPOLYGON (((127.29172 36.63614, 127.29203 ...
1	2901031		MULTIPOLYGON (((127.24092 36.46741, 127.24087 ...
2	2901032		MULTIPOLYGON (((127.35572 36.55450, 127.35568 ...
3	2901033		MULTIPOLYGON (((127.37706 36.56788, 127.37711 ...
4	2901034		MULTIPOLYGON (((127.36424 36.51353, 127.36375 ...
5	2901035		MULTIPOLYGON (((127.21187 36.57860, 127.21233 ...
6	2901036		MULTIPOLYGON (((127.21134 36.62035, 127.21169 ...
7	2901037		MULTIPOLYGON (((127.20998 36.71848, 127.21068 ...
8	2901038		MULTIPOLYGON (((127.25304 36.69594, 127.25410 ...
9	2901039		MULTIPOLYGON (((127.14980 36.72605, 127.14981 ...
10	2901053		MULTIPOLYGON (((127.26461 36.52069, 127.26459 ...
11	2901060		MULTIPOLYGON (((127.23853 36.53540, 127.23895 ...
12	2901056		MULTIPOLYGON (((127.24233 36.50869, 127.24234 ...
13	2901065		MULTIPOLYGON (((127.31207 36.50321, 127.31209 ...
14	2901062		MULTIPOLYGON (((127.26244 36.47868, 127.26244 ...
15	2901059		MULTIPOLYGON (((127.25182 36.52672, 127.25193 ...
16	2901061		MULTIPOLYGON (((127.26244 36.47868, 127.26181 ...
17	2901064		MULTIPOLYGON (((127.27743 36.47628, 127.27743 ...
18	2901066		MULTIPOLYGON (((127.29381 36.48496, 127.29381 ...

In [17]:

```
   = gpd.read_file(input_path.joinpath('33.   _   .geojson'))
```

Out[17]:

	PNU	JIBUN	geometry
0	3611037031101950001	195-1	MULTIPOLYGON (((127.17929 36.65234, 127.17930 ...
1	3611034029102230001	223-1	MULTIPOLYGON (((127.30726 36.46622, 127.30716 ...
2	3611034028101540000	154	MULTIPOLYGON (((127.30510 36.45196, 127.30502 ...
[illegible]	[illegible]	[illegible]	MULTIPOLYGON (((127.31121 36.47124, 127.31127

	PNU	JIBUN	geometry
3	3611034029200630000	63	MULTIPOLYGON (((
4	3611037031104690001	469-1	MULTIPOLYGON (((127.16223 36.64483, 127.16220 ...
...	...	...	...
198804	3611038024200170001	17-1	MULTIPOLYGON (((127.25132 36.65408, 127.25186 ...
198805	3611038024200170008	17-8	MULTIPOLYGON (((127.25186 36.65394, 127.25191 ...
198806	3611038024200170009	17-9	MULTIPOLYGON (((127.25189 36.65385, 127.25221 ...
198807	3611038024200170002	17-2	MULTIPOLYGON (((127.24888 36.65086, 127.24884 ...
198808	3611038024200170010	17-10	MULTIPOLYGON (((127.25175 36.65324, 127.25172 ...

198809 rows × 3 columns

In [18]:

```
_    = gpd.read_file(input_path.joinpath('16.  _  _   .geojson'))
_
```

Out[18]:

	gid	ws_cnt	found_age_1	found_age_2	found_age_3	found_age_4	found_age_5	found_age_6	runoutmon	smbiz_yn	...	indcd_m_yn	in
0	667567	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
1	667568	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
2	668566	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
3	668567	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
4	668568	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
...	...	...	...	...	...	...	...	...	...	...	...	...	
47391	919332	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
47392	919333	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
47393	919334	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
47394	920332	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	
47395	920333	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	None	

47396 rows × 69 columns

In [ ]:

In [19]:

```
#
    = pd.read_csv(input_path.joinpath('14.   _    .csv'))
```

Out[19]:

												lon	lat
0	D		D21	/ /	D21A06	/	G47511		3611025000		90-1, ( )	127.298552	36.599920
1	D		D21	/ /	D21A02	/	G47511		3611037000		40, ( )	127.204298	36.682526
2	F		F15		F15A03		F42201	·	3611036000		322, ( )	127.280990	36.560523
3	Q		Q04		Q04A01		I56194		3611034000		61-3, ( )	127.270561	36.426637
4	D		D11		D11A03		G47599		3611025000		159, (, 2)	127.294177	36.602922
...	...	...	...	...	...	...	...	...	...	...	...	...	...
10768	D		D23	/	D23A04		G45211		3611025000		48-11, ( )	127.299415	36.593481
10769	L		L01		L01A01		L68221		3611034000		529, ( )	127.333858	36.480396
10770	Q		Q06		Q06A02		I56114		3611055000		13, ( )	127.235897	36.500933
10771	N	/ /	N02	/ /	N02A01		R91223		3611056000		51, ( )	127.289353	36.477949
10772	R	/	R05	-	R05A02	/ /	P85620		3611054000		3 160, (, 4)	127.243716	36.506156

10773 rows × 13 columns

In [20]:

```
#
data_    = pd.crosstab(    .    ,     .         , margins=True)
data_
```

Out[20]:

	/ /							/	All
	7	25	74	61	0	3	136	47	353
	13	98	102	184	2	1	284	15	699
	2	2	32	25	0	1	38	11	111
	13	51	83	217	0	3	388	48	803
	5	19	69	61	0	5	230	31	420
	11	14	69	135	8	0	189	10	436
	21	157	159	313	0	4	349	115	1118
	0	11	44	38	0	0	81	38	212
	1	4	22	43	1	0	87	7	165
	3	13	38	45	0	6	115	57	277
	1	28	37	56	3	0	67	2	194
	0	15	14	46	3	0	71	7	156
	5	40	50	162	7	0	164	16	444
	10	78	40	128	3	1	203	2	465
	2	5	22	53	6	0	51	4	143
	7	9	61	139	5	0	182	14	417
	97	105	574	1133	36	3	1402	203	3553
	1	20	48	87	0	3	131	56	346
	1	89	49	77	0	0	174	71	461
All	200	783	1587	3003	74	30	4342	754	10773

In [21]:

```
#
```

```
data_    _all = data_    ['All']
data_    _all
```

Out[21]:

```
       353
       699
       111
       803
       420
       436
      1118
       212
       165
       277
       194
       156
       444
       465
       143
       417
      3553
       346
       461
All    10773
Name: All, dtype: int64
```

In [22]:

```
#
   _    = gpd.read_file(input_path.joinpath('32.   _    (   ).geojson'))
   _

   _   ['lon'] = (    _   ['geometry'].bounds['maxx'] +     _   ['geometry'].bounds['minx'])/2
   _   ['lat'] = (    _   ['geometry'].bounds['maxy'] +     _   ['geometry'].bounds['miny'])/2
data_    _lon_lat =     _   [['ADM_DR_NM', 'lon', 'lat']]
data_    _lon_lat=data_    _lon_lat.rename(columns={'ADM_DR_NM':'    '})
data_    _lon_lat
```

Out[22]:

	lon	lat
0	127.282863	36.607199
1	127.279812	36.507771
2	127.330061	36.550175
3	127.381030	36.528753
4	127.290863	36.462920
5	127.207249	36.507231
6	127.247954	36.583444
7	127.199439	36.655099
8	127.253810	36.656332
9	127.167851	36.711410
10	127.265374	36.508349
11	127.236059	36.517374
12	127.247568	36.503792
13	127.307823	36.490650
14	127.252152	36.488832
15	127.251252	36.517544
16	127.249586	36.473352
17	127.274502	36.470178
18	127.288813	36.478255

In [23]:

```
#         ,
data_    _info = pd.merge( data_    _all,data_    _lon_lat, on ="    ")
data_    _info
```

Out[23]:

	All	lon	lat
0	353	127.236059	36.517374
1	699	127.290863	36.462920
2	111	127.274502	36.470178
3	803	127.265374	36.508349
4	420	127.288813	36.478255
5	436	127.381030	36.528753
6	1118	127.252152	36.488832
7	212	127.307823	36.490650
8	165	127.167851	36.711410
9	277	127.251252	36.517544
10	194	127.279812	36.507771
11	156	127.330061	36.550175
12	444	127.247954	36.583444
13	465	127.207249	36.507231
14	143	127.253810	36.656332
15	417	127.199439	36.655099
16	3553	127.282863	36.607199
17	346	127.247568	36.503792
18	461	127.249586	36.473352

In [24]:

```
#
data_     =
data_     = data_     .sort_values(by='        ')
data_

data_     = data_     [['        ', '      ', 'lon', 'lat']]

#
data_     .rename(columns={'        ':'    '}, inplace = True)
data_
```

Out[24]:

			lon	lat
6091	/ /	16, ( )	127.236487	36.501126
153	/ /	565, ( )	127.220519	36.499410
9564	/ /	19, ( )	127.233984	36.505851
2597	/ /	233, ( )	127.296533	36.601936
3054	/ /	385-3, ( )	127.203729	36.496271
...	...	...	...	...
8580	/	157, ( )	127.262479	36.479318
1639	/	17, ( )	127.294190	36.595325
1642	/	2 64, ( )	127.289099	36.602117
8606	/	62-15, ( )	127.251201	36.487016
10772	/	3 160, (, 4)	127.243716	36.506156

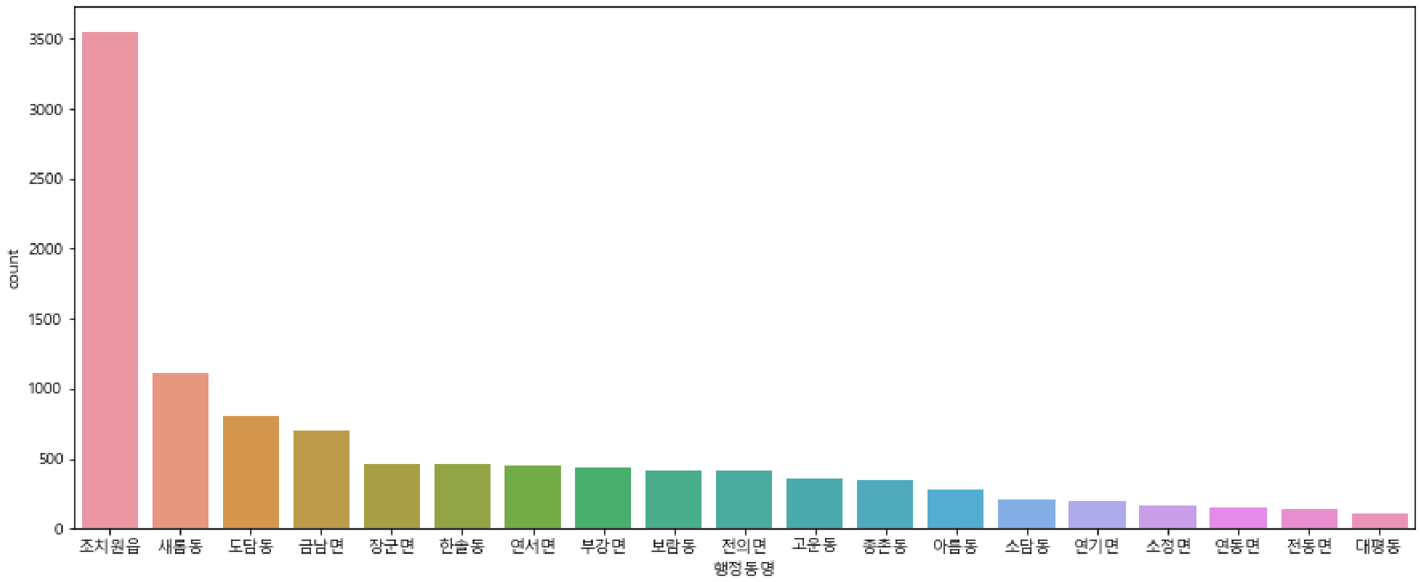
10773 rows × 4 columns

In [25]:

```
#
plt.figure(figsize= (15,6))
sns.countplot(data =      , x="      ", order=    ['    '].value_counts().index)
```

Out[25]:

```
<AxesSubplot:xlabel='    ', ylabel='count'>
```



In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```

In [26]:

```
#  ,   ,          float
data_    _info['lon'] = data_    _info.lon.astype(float)
data_    _info['lat'] = data_    _info.lat.astype(float)
data_    _info['All'] = data_    _info.All.astype(float)
data_    _info

#
  _map = folium.Map(location = [data_    _info['lat'].mean(), data_    _info['lon'].mean()],zoom_start = 11)

for item in data_    _info.index:
    latitude = data_    _info.loc[item,'lat']
    longtitude = data_    _info.loc[item,'lon']
    popups = folium.Popup(data_    _info.loc[item, '    '], max_width=100)
    folium.CircleMarker([latitude, longtitude],
                        radius = data_    _info.loc[item,'All']/100,
                        popup = popups,
                        color = 'red',
                        fill = True).add_to(    _map)
  _map
```

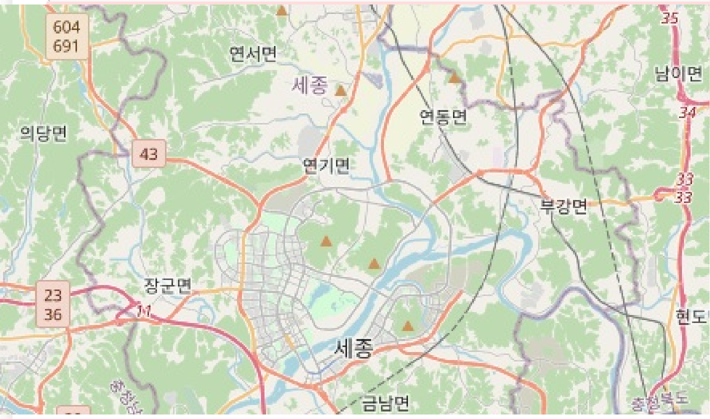
Out[26]:

```
Make this Notebook Trusted to load map: File -> Trust Notebook
```

2

In [27]:

```
#
   _    = gpd.read_file(input_path.joinpath('32.   _    (   ).geojson'))
   _

data = data_    _info[['    ', 'All']]
data

#
  _map2 = folium.Map(location = [data_    _info['lat'].mean(), data_    _info['lon'].mean()],zoom_start = 11)

  _map2.choropleth(
   geo_data =      _   ,
   data = data,
   columns=['    ', 'All'],
   key_on = 'feature.properties.ADM_DR_NM',
   fill_color = 'BuPu',
   legend_name ='    ',
   )

  _map2
```



```
ureWarning: The choropleth  method has been deprecated. Instead use the new C
 notebook 'GeoJSON_and_choropleth' for how to do this.
```

In [ ]:

```
```

In [ ]:

```
```

In [28]:

```
data_     =
data_     = data_    [['        ', '      ', 'lon', 'lat']]

#
data_    .rename(columns={'        ':'    '}, inplace = True)
data_


#   ,
data_    ['lon'] = data_    .lon.astype(float)
data_    ['lat'] = data_    .lat.astype(float)


#

    _map = folium.Map(location = [data_    ['lat'].mean(), data_    ['lon'].mean()],zoom_start = 11)

for item in data_    .index:
    latitude = data_    .loc[item,'lat']
    longtitude = data_    .loc[item,'lon']

    if data_    .loc[item, '    '] == '  /  /  ':
        colors = 'dodgerblue'

    elif data_    .loc[item, '    '] == '    ':
        colors = 'burlywood'

    elif data_    .loc[item, '    '] == '      ':
        colors = 'gold'

    elif data_    .loc[item, '    '] == '  ':
        colors = 'darkolivegreen'

    elif data_    .loc[item, '    '] == '  /  ':
        colors = 'slategrey'

    elif data_    .loc[item, '    '] == '  ' :
        colors = 'salmon'

    elif data_    .loc[item, '    '] == '    ' :
        colors = 'blueviolet'

    elif data_    .loc[item, '    '] == '  ':
        colors = 'lihgtpink'
    popups = folium.Popup(data_    .loc[item, '      '], max_width=150)
    folium.CircleMarker([latitude, longtitude],
                      popup = popups,
                      color = colors,
                      fill = True).add_to(    _map)
    _map
```

```
/opt/app-root/lib/python3.6/site-packages/pandas/core/frame.py:4308: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  errors=errors,
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # Remove the CWD from sys.path while we load stuff.
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  # This is added back by InteractiveShellApp.init_path()
```

Out[28]:

Make this Notebook Trusted to load map: File -> Trust Notebook

In [ ]:

In [29]:

```
  _

#

data_  _   =   _ [['   ', '    (  )']]
for i in range(len(data_  _  ['   '])):
    if len(data_  _  .loc[i, '   ']) == 9:
        data_  _  ['   '][i] = data_  _  ['   '][i][:5]
    elif len(data_  _  .loc[i, '   ']) == 8:

        if data_  _  .loc[i, '   '][1] == ' ':
            data_  _  ['   '][i] = data_  _  ['   '][i][:5]
        else:
            data_  _  ['   '][i] = data_  _  ['   '][i][:5]


data_  _
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:14: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/opt/app-root/lib/python3.6/site-packages/IPython/core/interactiveshell.py:3343: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  exec(code_obj, self.user_global_ns, self.user_ns)
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:12: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  if sys.path[0] == '':
```

Out[29]:

		( )
0		8600
1		27300
2		32000
3		30500
4		31300
...	...	...
21135		50000
21136		23500
21137		68000
21138		6500
21139		37000

21140 rows × 2 columns

In [30]:

```
#        ( )
data_  _    = data_  _  .groupby(data_  _  .   )['   ( )'].mean()
data_  _

data_  _    = data_  _  .reset_index()
data_  _

for i in range(len(data_  _   ['   '])):
    data_  _  ['   '][i] = data_  _  ['   '][i][1:]

data_  _
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  if __name__ == '__main__':
```

Out[30]:

		( )
0		33226.134474
1		13544.158965
2		10566.632948
3		58885.662667
4		66912.200820
5		45208.741341
6		50896.721893
7		59299.212299
8		7824.732143
9		59670.444348
10		53189.771694
11		6329.310345

12	34519.918834
13	39438.960880
14	6573.083333
15	11860.162602
16	2718.142857
17	5652.380952
18	7884.500000
19	16883.180214
20	38950.013474
21	40368.280587

In [31]:

```
#
  _    = gpd.read_file(input_path.joinpath('31.   _    (   ).geojson'))
  _
```

Out[31]:

	EMD_CD	EMD_KOR_NM	geometry
0	36110101		MULTIPOLYGON (((127.30833 36.47977, 127.30818 ...
1	36110102		MULTIPOLYGON (((127.30480 36.48845, 127.30480 ...
2	36110103		MULTIPOLYGON (((127.29383 36.48508, 127.29381 ...
3	36110104		MULTIPOLYGON (((127.27215 36.46388, 127.27197 ...
4	36110105		MULTIPOLYGON (((127.24631 36.46300, 127.24626 ...
5	36110106		MULTIPOLYGON (((127.25661 36.47033, 127.25653 ...
6	36110107		MULTIPOLYGON (((127.26603 36.49182, 127.26744 ...
7	36110108		MULTIPOLYGON (((127.23978 36.48556, 127.23966 ...
8	36110109		MULTIPOLYGON (((127.24003 36.48809, 127.24004 ...
9	36110110		MULTIPOLYGON (((127.25544 36.49534, 127.25543 ...
10	36110111		MULTIPOLYGON (((127.25357 36.50688, 127.25357 ...
11	36110112		MULTIPOLYGON (((127.23895 36.53520, 127.23960 ...
12	36110113		MULTIPOLYGON (((127.25238 36.51029, 127.25218 ...
13	36110114		MULTIPOLYGON (((127.25357 36.50688, 127.25348 ...
14	36110115		MULTIPOLYGON (((127.26859 36.53949, 127.26858 ...
15	36110116		MULTIPOLYGON (((127.27076 36.52337, 127.27005 ...
16	36110117		MULTIPOLYGON (((127.34569 36.51897, 127.34628 ...
17	36110118		MULTIPOLYGON (((127.31732 36.48175, 127.31730 ...
18	36110250		MULTIPOLYGON (((127.30663 36.60062, 127.30662 ...
19	36110310		MULTIPOLYGON (((127.28524 36.55518, 127.28605 ...
20	36110320		MULTIPOLYGON (((127.32202 36.58302, 127.32209 ...
21	36110330		MULTIPOLYGON (((127.36891 36.48959, 127.36891 ...
22	36110340		MULTIPOLYGON (((127.36098 36.49088, 127.36059 ...
23	36110350		MULTIPOLYGON (((127.23952 36.48582, 127.23966

	EMD_CD	EMD_KOR_NM	geometry
23	36110350		
24	36110360		MULTIPOLYGON (((127.19288 36.60110, 127.19298 ...
25	36110370		MULTIPOLYGON (((127.19609 36.60448, 127.19604 ...
26	36110380		MULTIPOLYGON (((127.27948 36.63350, 127.27948 ...
27	36110390		MULTIPOLYGON (((127.20791 36.71901, 127.20781 ...

In [32]:

```
#
  _   ['lon']= (   _  ['geometry'].bounds['maxx'] +     _  ['geometry'].bounds['minx'])/2
  _   ['lat'] = (   _  ['geometry'].bounds['maxy'] +     _  ['geometry'].bounds['miny'])/2

data_   _lon_lat =     _   [['EMD_KOR_NM', 'lon', 'lat']]
data_   _lon_lat = data_   _lon_lat.rename(columns={'EMD_KOR_NM':'   '})
data_   _lon_lat
```

Out[32]:

		lon	lat
0		127.310382	36.491414
1		127.301194	36.485012
2		127.288813	36.478255
3		127.274502	36.470178
4		127.246671	36.471256
5		127.254392	36.477017
6		127.264276	36.486736
7		127.249013	36.484422
8		127.243697	36.493832
9		127.265374	36.500819
10		127.247569	36.503792
11		127.236093	36.517375
12		127.251275	36.517544
13		127.262456	36.515875
14		127.253918	36.534591
15		127.271144	36.527016
16		127.333018	36.525097
17		127.329770	36.496421
18		127.282863	36.607327
19		127.279813	36.507784
20		127.330061	36.550718
21		127.381030	36.528753
22		127.290864	36.462920
23		127.207249	36.507231
24		127.247954	36.583444
25		127.199440	36.655088
26		127.253811	36.656332
27		127.167851	36.711410

In [33]:

```
#

data_  _  .loc[len(data_  _  )] = ['   ', 0]
data_  _  .loc[len(data_  _  )] = ['   ', 0]
data_  _  .loc[len(data_  _  )] = ['   ', 0]
data_  _  .loc[len(data_  _  )] = ['   ', 0]
data_  _  .loc[len(data_  _  )] = ['   ', 0]
data_  _  .loc[len(data_  _  )] = ['   ', 0]
data
```

```
data_  _
```

Out[33]:

	( )
0	33226.134474
1	13544.158965
2	10566.632948
3	58885.662667
4	66912.200820
5	45208.741341
6	50896.721893
7	59299.212299
8	7824.732143
9	59670.444348
10	53189.771694
11	6329.310345
12	34519.918834
13	39438.960880
14	6573.083333
15	11860.162602
16	2718.142857
17	5652.380952
18	7884.500000
19	16883.180214
20	38950.013474
21	40368.280587
22	0.000000
23	0.000000
24	0.000000
25	0.000000
26	0.000000
27	0.000000

In [34]:

```
data_  _  ['   '][1] = '   '
data_  _  ['   '][8] = '   '
data_  _  ['   '][11] ='   '
data_  _  ['   '][14] ='   '
data_  _  ['   '][15] ='   '
data_  _  ['   '][16] ='   '
data_  _  ['   '][17]='   '
data_  _  ['   '][18]='   '
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  This is separate from the ipykernel package so we can avoid doing imports until
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  after removing the cwd from sys.path.
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  import sys
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

In [35]:

```
data_  _  _info = pd.merge(data_  _lon_lat, data_  _  , on ='  ', how="outer")
data_  _  _info
```

Out[35]:

	lon	lat	( )
0	127.310382	36.491414	50896.721893
1	127.301194	36.485012	53189.771694
2	127.288813	36.478255	59299.212299
3	127.274502	36.470178	66912.200820
4	127.246671	36.471256	0.000000
5	127.254392	36.477017	40368.280587
6	127.264276	36.486736	10566.632948
7	127.249013	36.484422	59670.444348
8	127.243697	36.493832	58885.662667
9	127.265374	36.500819	39438.960880
10	127.247569	36.503792	38950.013474
11	127.236093	36.517375	33226.134474
12	127.251275	36.517544	34519.918834
13	127.262456	36.515875	45208.741341
14	127.253918	36.534591	0.000000
15	127.271144	36.527016	0.000000
16	127.333018	36.525097	0.000000
17	127.329770	36.496421	0.000000
18	127.282863	36.607327	16883.180214
19	127.279813	36.507784	0.000000
20	127.330061	36.550718	6573.083333
21	127.381030	36.528753	7824.732143
22	127.290864	36.462920	13544.158965
23	127.207249	36.507231	2718.142857
24	127.247954	36.583444	11860.162602
25	127.199440	36.655088	7884.500000
26	127.253811	36.656332	5652.380952
27	127.167851	36.711410	6329.310345

In [36]:

```
#   ,      float
data_  _  _info['lon'] = data_  _  _info.lon.astype(float)
data_  _  _info['lat'] = data_  _  _info.lat.astype(float)
```

In [37]:

```
#
  _    = gpd.read_file(input_path.joinpath('31.  _    (   ).geojson'))
  _

   _map = folium.Map(location = [data_   _   _info['lat'].mean(), data_   _   _info['lon'].mean()] ,zoom_start = 11)

data = data_   _   _info[['   ','   (  )']]

folium.Choropleth(
    geo_data =      _   ,
    data = data,
    columns=['   ','   (  )'],
    key_on = 'feature.properties.EMD_KOR_NM',
    fill_color = 'BuPu',
    legend_name ='     ',
    ).add_to(     _map)


   _map
```

Out[37]:

Make this Notebook Trusted to load map: File -> Trust Notebook

In [ ]:

```

```

,

In [38]:

```
#
  = pd.read_csv(input_path.joinpath('20.   _     .csv') )


#
  = pd.read_csv(input_path.joinpath('21.   _     .csv') )
```

Out[38]:

0	201701	3

0	201701	5				
1	201701	9				
2	201701	13				
3	201701	3				
4	201701	1				1
...	...	...	...	...	...	...
80987	202101	22				
80988	202101	2				1
80989	202101	35				
80990	202101	13				1
80991	202101	84				

80992 rows × 6 columns

In [39]:

```
#
data_    =  .groupby('       ')['    '].agg('sum')
data_

#
data_    =  .groupby('       ')['    '].agg('sum')
data_
```

Out[39]:

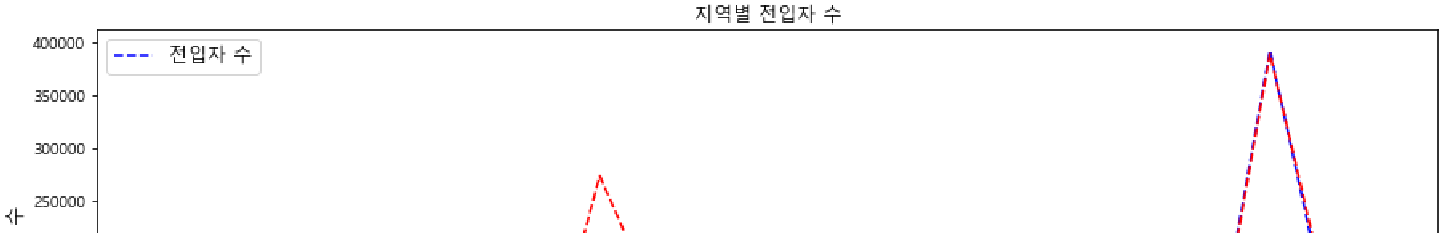
```
        104194
         77692
         11042
         24195
        150364
         63403
         36692
        131275
         66903
         10559
         77777
         19753
         19127
         54553
         50845
         17857
         35215
        393101
        101396
         83269
Name:     , dtype: int64
```

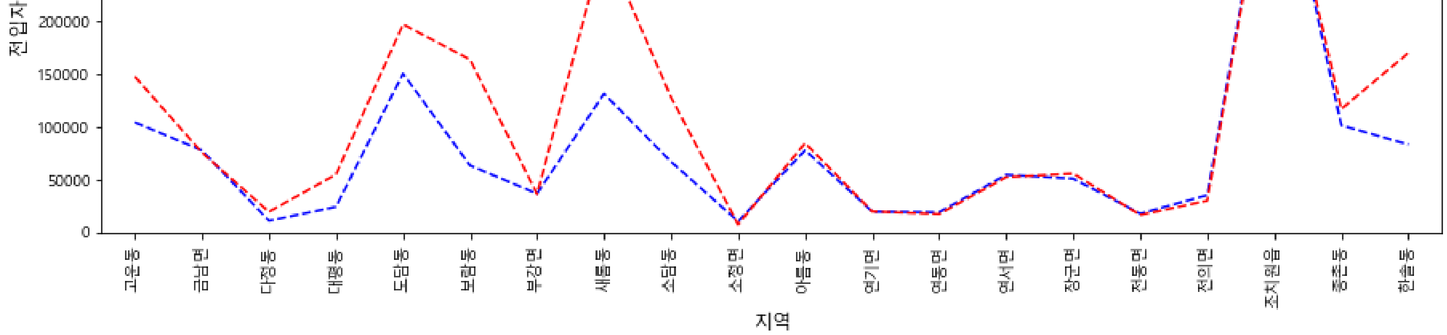
In [40]:

```
ax = data_    .plot(kind ='bar',title ='          ', figsize =(15, 5), legend=True, fontsize = 10, color='white')
ax.plot(data_     , linestyle='--', color ='blue')
ax.plot(data_     , linestyle='--', color='red')
#ax = data_   .plot(kind ='bar', title ='          ', figsize =(15, 5), legend=True, fontsize = 10)
#plt.plot(data_   , linestyle='--', color = 'orangered')
ax.set_xlabel('  ', fontsize = 12)
ax.set_ylabel('     ', fontsize = 12)
ax.legend(['     '], fontsize=12)
```

Out[40]:

```
<matplotlib.legend.Legend at 0x7f0575f3c550>
```





In [ ]:

In [41]:

```
#
  = pd.read_csv(input_path.joinpath('12.  _ ( )_   .csv') , encoding='cp949')

#
for i in range(len(  ['   '])):
     ['   '][i] =  ['   '][i][8:12]
```

```
/opt/app-root/lib/python3.6/site-packages/ipykernel_launcher.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[41]:

							( )	( )	
0	201701	1				8m	793.0	16302	
1	201701	1				8m	1833.0	37682	
2	201701	1				8m	2055.0	42246	
3	201701	2				8m	235.0	1600	
4	201701	2				-	396.0	1604	
...	...	...	...	...	...	...	...	...	...
32440	202010	28				-	320.0	2910	
32441	202010	28				-	148.0	4925	
32442	202010	28				8m	102.0	900	
32443	202010	29				12m	390.0	10000	
32444	202010	29				-	1448.0	5068	

32445 rows × 9 columns

In [42]:

```
#
pd.set_option('display.max_rows', 700)
data_   = round(  .groupby(['   ',' '])[['    ( )', '    ( )']].mean(),3)
data_
```

Out[42]:

		( )	( )
		32.935	4341.750
		466.026	48160.316
		89.330	7507.750

()	( )
10.837	769.556
1397.900	190000.000
1501.956	86040.455
950.865	25800.870
105.000	11078.000
1193.933	25209.415
277.281	15032.036
38.145	1041.616
385.500	12802.222
273.420	27482.000
2.364	129.000
371.333	28333.333
1337.664	8894.499
392.375	20762.500
808.516	17339.125
300.000	30000.000
2816.000	11386.667
1722.143	55305.214
5105.667	422455.500
359.877	41993.230
50.667	283.000
261.667	9333.667
4588.286	404578.429
36.000	567.000
296.324	44642.538
563.240	57472.187
522.000	2532.000
2959.750	539130.750
732.852	28797.290
2403.833	23802.500
1191.015	24110.720
283.331	8829.132
131.254	2138.119
514.526	8729.368
635.000	800.000
1073.914	5836.211
746.700	52329.250
774.593	15349.158
684.000	8300.000
405.667	18665.667
134.176	4619.471
9060.000	526990.000
220.125	671.375
4211.000	1446476.000
2905.133	104983.727
1311.652	9633.288
298.115	8237.415
204.469	1577.429
787.000	3758.000
850.889	2866.175
1171.250	21670.500
1283.413	14388.170
1665.650	43028.000
707.667	3735.333

()	( )
336.748	42198.622
2707.002	1376042.500
832.543	27465.824
1301.698	45647.311
1178.264	62206.619
49.356	1117.036
125.999	8823.889
676.848	7752.190
1859.885	24657.154
684.182	26279.714
15.000	363.000
203.555	17910.000
28.230	1025.000
963.000	45785.000
1739.625	29095.375
334.400	3146.800
1980.671	30568.041
232.464	19803.109
84.969	2704.984
1587.000	24000.000
972.928	14188.310
217.000	4583.000
666.639	20659.524
247.000	5039.000
649.667	17666.667
33.000	1342.500
179.943	16237.714
1360.432	25300.192
365.250	7266.750
1399.424	27314.750
341.912	11587.603
67.351	1429.604
564.278	14524.833
564.500	7250.000
134.250	3090.000
1450.667	32773.333
1082.925	5860.402
435.615	14520.308
874.370	18303.290
364.000	22833.333
394.111	4530.000
20.411	409.431
18138.714	430506.714
688.591	20293.091
509.000	1500.000
1189.714	26457.425
361.199	15014.671
59.326	1380.489
300.098	9164.679
626.000	6500.000
1060.003	12090.879
58.888	2272.020
779.989	22006.768
165.667	5841.333

()	()
579.000	14012.000
163.153	3797.778
72.000	4867.000
14230.750	264583.000
1977.978	15145.978
281.000	988.333
1623.510	15810.489
413.795	6263.018
105.303	780.048
497.667	4237.733
986.000	1756.000
352.000	2630.000
1288.797	3035.566
1987.834	29630.400
1191.096	11714.125
323.200	3089.000
422.333	5197.333
754.000	2959.125
6413.433	172625.556
127.220	1156.525
2202.000	12151.500
893.654	11283.918
353.401	8897.252
83.785	1300.452
501.444	3467.081
443.000	1920.000
1052.295	3314.531
338.804	8397.500
611.511	8904.316
168.667	1556.667
200.000	5687.000
995.000	22000.000
1996.400	67493.333
568.800	26120.400
440.000	4322.062
724.500	32883.000
701.992	27411.211
769.839	29615.781
154.503	11796.762
25.190	967.486
52.000	6500.000
897.500	19583.500
420.751	6324.491
497.375	23897.750
812.117	33631.755
30.000	4047.000
101.000	5090.000
150.500	11500.000
9120.551	344089.408
329.168	34361.772
2152.292	299406.257

In [ ]:

In [43]:

```
# 31.   _     (   ).geojson              ,
  _lon_lat = data_   _lon_lat
  _lon_lat
```

Out[43]:

	lon	lat
0	127.310382	36.491414
1	127.301194	36.485012
2	127.288813	36.478255
3	127.274502	36.470178
4	127.246671	36.471256
5	127.254392	36.477017
6	127.264276	36.486736
7	127.249013	36.484422
8	127.243697	36.493832
9	127.265374	36.500819
10	127.247569	36.503792
11	127.236093	36.517375
12	127.251275	36.517544
13	127.262456	36.515875
14	127.253918	36.534591
15	127.271144	36.527016
16	127.333018	36.525097
17	127.329770	36.496421
18	127.282863	36.607327
19	127.279813	36.507784
20	127.330061	36.550718
21	127.381030	36.528753
22	127.290864	36.462920
23	127.207249	36.507231
24	127.247954	36.583444
25	127.199440	36.655088
26	127.253811	36.656332
27	127.167851	36.711410

In [44]:

```
data_    = data_   .reset_index()
data_
```

Out[44]:

	( )	( )
0	32.935	4341.750
1	466.026	48160.316
2	89.330	7507.750
3	10.837	769.556
4	1397.900	190000.000
5	1501.956	86040.455
6	950.865	25800.870

7	105.000	11078.000
8	1193.933	25209.415
9	277.281	15032.036
10	38.145	1041.616
11	385.500	12802.222
12	273.420	27482.000
13	2.364	129.000
14	371.333	28333.333
15	1337.664	8894.499
16	392.375	20762.500
17	808.516	17339.125
18	300.000	30000.000
19	2816.000	11386.667
20	1722.143	55305.214
21	5105.667	422455.500
22	359.877	41993.230
23	50.667	283.000
24	261.667	9333.667
25	4588.286	404578.429
26	36.000	567.000
27	296.324	44642.538
28	563.240	57472.187
29	522.000	2532.000
30	2959.750	539130.750
31	732.852	28797.290
32	2403.833	23802.500
33	1191.015	24110.720
34	283.331	8829.132
35	131.254	2138.119
36	514.526	8729.368
37	635.000	800.000
38	1073.914	5836.211
39	746.700	52329.250
40	774.593	15349.158
41	684.000	8300.000
42	405.667	18665.667
43	134.176	4619.471
44	9060.000	526990.000
45	220.125	671.375
46	4211.000	1446476.000
47	2905.133	104983.727
48	1311.652	9633.288
49	298.115	8237.415
50	204.469	1577.429
51	787.000	3758.000
52	850.889	2866.175
53	1171.250	21670.500
54	1283.413	14388.170
55	1665.650	43028.000
56	707.667	3735.333
57	336.748	42198.622
58	2707.002	1376042.500
59	832.543	27465.824
60	1301.698	45647.311

61	1178.264	62206.619
62	49.356	1117.036
63	125.999	8823.889
64	676.848	7752.190
65	1859.885	24657.154
66	684.182	26279.714
67	15.000	363.000
68	203.555	17910.000
69	28.230	1025.000
70	963.000	45785.000
71	1739.625	29095.375
72	334.400	3146.800
73	1980.671	30568.041
74	232.464	19803.109
75	84.969	2704.984
76	1587.000	24000.000
77	972.928	14188.310
78	217.000	4583.000
79	666.639	20659.524
80	247.000	5039.000
81	649.667	17666.667
82	33.000	1342.500
83	179.943	16237.714
84	1360.432	25300.192
85	365.250	7266.750
86	1399.424	27314.750
87	341.912	11587.603
88	67.351	1429.604
89	564.278	14524.833
90	564.500	7250.000
91	134.250	3090.000
92	1450.667	32773.333
93	1082.925	5860.402
94	435.615	14520.308
95	874.370	18303.290
96	364.000	22833.333
97	394.111	4530.000
98	20.411	409.431
99	18138.714	430506.714
100	688.591	20293.091
101	509.000	1500.000
102	1189.714	26457.425
103	361.199	15014.671
104	59.326	1380.489
105	300.098	9164.679
106	626.000	6500.000
107	1060.003	12090.879
108	58.888	2272.020
109	779.989	22006.768
110	165.667	5841.333
111	579.000	14012.000
112	163.153	3797.778
113	72.000	4867.000
114	14230.750	264583.000

115	1977.978	15145.978
116	281.000	988.333
117	1623.510	15810.489
118	413.795	6263.018
119	105.303	780.048
120	497.667	4237.733
121	986.000	1756.000
122	352.000	2630.000
123	1288.797	3035.566
124	1987.834	29630.400
125	1191.096	11714.125
126	323.200	3089.000
127	422.333	5197.333
128	754.000	2959.125
129	6413.433	172625.556
130	127.220	1156.525
131	2202.000	12151.500
132	893.654	11283.918
133	353.401	8897.252
134	83.785	1300.452
135	501.444	3467.081
136	443.000	1920.000
137	1052.295	3314.531
138	338.804	8397.500
139	611.511	8904.316
140	168.667	1556.667
141	200.000	5687.000
142	995.000	22000.000
143	1996.400	67493.333
144	568.800	26120.400
145	440.000	4322.062
146	724.500	32883.000
147	701.992	27411.211
148	769.839	29615.781
149	154.503	11796.762
150	25.190	967.486
151	52.000	6500.000
152	897.500	19583.500
153	420.751	6324.491
154	497.375	23897.750
155	812.117	33631.755
156	30.000	4047.000
157	101.000	5090.000
158	150.500	11500.000
159	9120.551	344089.408
160	329.168	34361.772
161	2152.292	299406.257

In [45]:

```
#
data_ _   = data_ .iloc[0:5]
data_ _  1=data_ _  [[' ',' ( )']]
data_ _  1 = data_ _  1.set_index(' ')
data_ _  1
```

Out[45]:

Out[45]:

( )
32.935
466.026
89.330
10.837
1397.900

In [46]:

```
#
data_ _  2=data_ _  [['  ',  '   (  )']]
data_ _  2 = data_ _  2.set_index('  ')
data_ _  2
```

Out[46]:

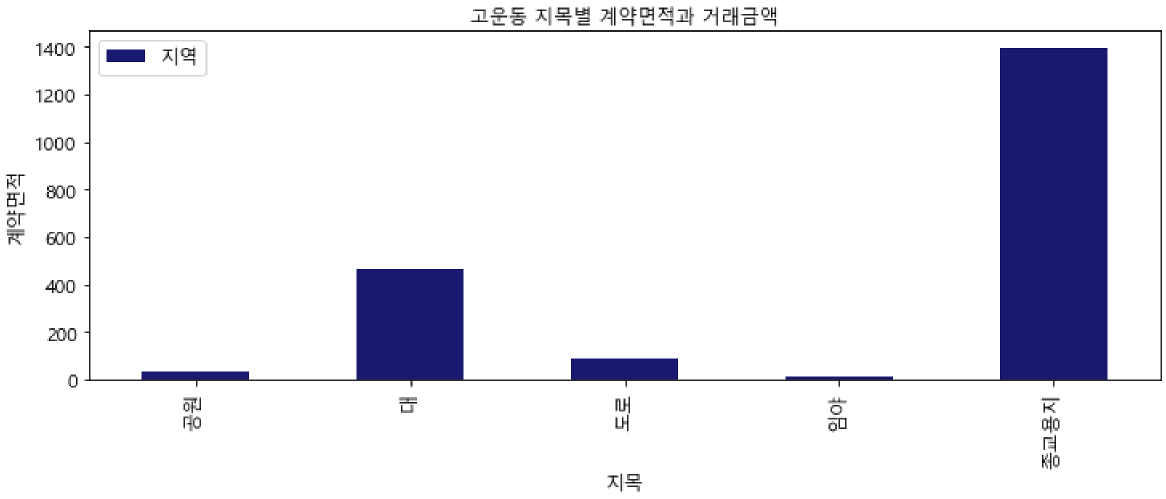
( )
4341.750
48160.316
7507.750
769.556
190000.000

In [47]:

```
ax = data_ _  1.plot(kind='bar', title='                 ', figsize=(12,4), legend=True, fontsize=12, color = 'midnightblue')
#graph = ax.plot(data_ _  1, color = 'midnightblue')
#ax.plot(data_ _  2, linestyle='--', color = 'palevioletred')

ax.set_xlabel('  ',fontsize=12)
ax.set_ylabel('    ', fontsize=12)
ax.legend(['  ','    '],fontsize=12)
```

Out[47]:

```
<matplotlib.legend.Legend at 0x7f0575ad4b00>
```



In [ ]:

```

```

In [48]:

```
#

vegabar = Bar(data_ _  1, width = 400, height = 200).axis_titles(x = '  ', y = '    ')
```

```
vegagraph = Vega(vegabar.to_json(), width = vegabar.width+100, height = vegabar.height + 50)
vegabar2 = Bar(data_  _   2, width = 400, height = 200).axis_titles(x = '  ', y = '    ')
vegagraph2 = Vega(vegabar2.to_json(), width = vegabar2.width+100, height = vegabar2.height + 50)

  _    = gpd.read_file(input_path.joinpath('31.   _    (   ).geojson'))
  _

 _map = folium.Map(location = [  _lon_lat['lat'].mean(),   _lon_lat['lon'].mean()], zoom_start = 11)
 _map

#
 _map.choropleth(
    geo_data =      _   ,
    key_on = 'feature.properties.EMD_KOR_NM',
    fill_color = 'White',
    fill_opacity = 0.3,
    legend_name ='     ',
    )

#
for i in range(len(  _lon_lat['   '])):
    latitude =    _lon_lat['lat'][i]
    longtitude =    _lon_lat['lon'][i]
    popups = folium.Popup(  _lon_lat['   '][i], max_width=100)

    if(  _lon_lat['   '][i] == '   '):

        popups = folium.Popup(max_width=500).add_child(vegagraph)


    folium.CircleMarker([latitude, longtitude],
                        popup = popups,
                        color = 'midnightblue',
                        fill = True).add_to(  _map)


#
for i in range(len(  _lon_lat['   '])):
    latitude = (  _lon_lat['lat'][i]+0.001)
    longtitude = (  _lon_lat['lon'][i]+0.001)

    popups2 = folium.Popup(  _lon_lat['   '][i], max_width=100)
    if(  _lon_lat['   '][i] == '   '):
        popups2 = folium.Popup(max_width=500).add_child(vegagraph2)

    folium.CircleMarker([latitude, longtitude],
                        popup = popups2,
                        color = 'crimson',
                        fill = True).add_to(  _map)




 _map
```

```
/opt/app-root/lib/python3.6/site-packages/folium/folium.py:415: FutureWarning: The choropleth  method has been deprecated. Instead use the new C
horopleth class, which has the same arguments. See the example notebook 'GeoJSON_and_choropleth' for how to do this.
  FutureWarning
```

Out[48]:

Make this Notebook Trusted to load map: File -> Trust Notebook

In [ ]:

In [ ]:

In [ ]: