In [86]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.rc('figure',figsize=(10,6))
```

In [87]:

```
df=pd.read_csv('incheon_streetlight.csv',engine='python')
df
```

Out[87]:

	순번	분전함	등주	위도	경도	데이터기준일자
0	1	월미로21(YB1)	좌103	37.472676	126.606904	2020-04-27
1	2	월미로21(YB1)	좌104	37.473054	126.606874	2020-04-27
2	3	월미로21(YB1)	좌105	37.473430	126.606698	2020-04-27
3	4	월미로21(YB1)	좌106	37.473816	126.606510	2020-04-27
4	5	월미로21(YB1)	좌107	37.474174	126.606356	2020-04-27
...	...	...	...	...	...	...
15725	15726	흰바위로LP_14	26	37.478098	126.502607	2020-04-27
15726	15727	흰바위로LP_14	27	37.477987	126.502791	2020-04-27
15727	15728	흰바위로LP_14	28	37.477861	126.503011	2020-04-27
15728	15729	흰바위로LP_14	29	37.477743	126.503217	2020-04-27
15729	15730	흰바위로LP_14	30	37.477630	126.503417	2020-04-27

15730 rows × 6 columns

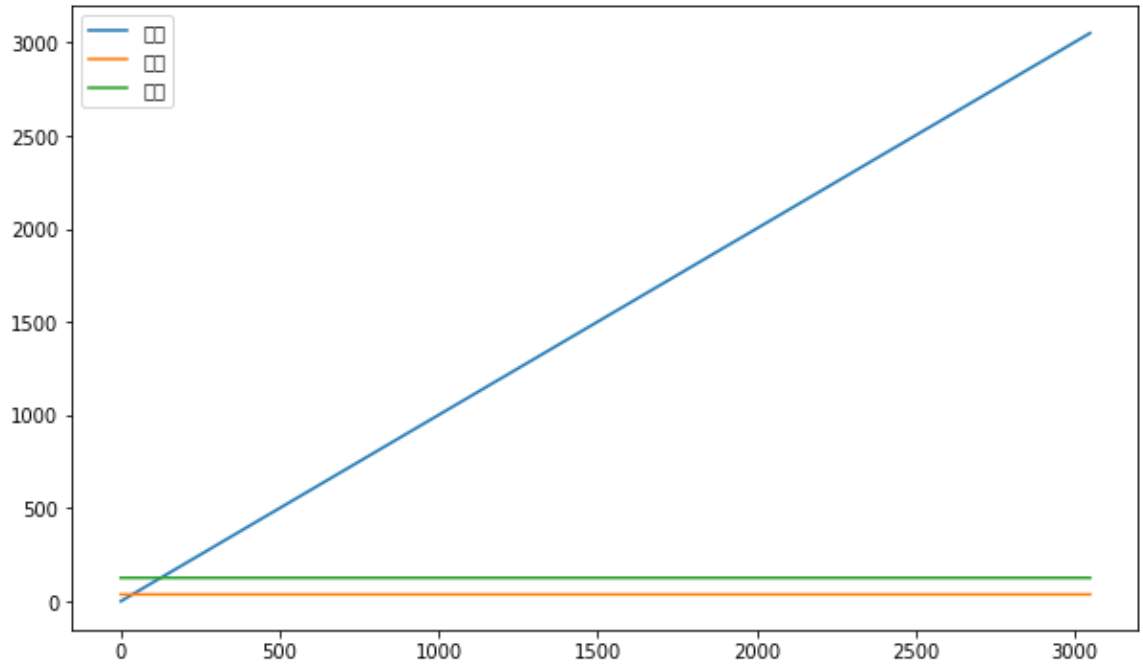
In [88]:

```
df[df['분전함']=='월미로21(YB1)'].plot()
```

Out[88]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x191a34eba48>

C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:211: RuntimeWarning: Glyph 49692 missing from current font.
  font.set_text(s, 0.0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:211: RuntimeWarning: Glyph 48264 missing from current font.
  font.set_text(s, 0.0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:211: RuntimeWarning: Glyph 50948 missing from current font.
  font.set_text(s, 0.0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:211: RuntimeWarning: Glyph 46020 missing from current font.
  font.set_text(s, 0.0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:211: RuntimeWarning: Glyph 44221 missing from current font.
  font.set_text(s, 0.0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:180: RuntimeWarning: Glyph 49692 missing from current font.
  font.set_text(s, 0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:180: RuntimeWarning: Glyph 48264 missing from current font.
  font.set_text(s, 0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:180: RuntimeWarning: Glyph 50948 missing from current font.
  font.set_text(s, 0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:180: RuntimeWarning: Glyph 46020 missing from current font.
  font.set_text(s, 0, flags=flags)
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩matplotlib₩backends₩backend_agg.p
y:180: RuntimeWarning: Glyph 44221 missing from current font.
  font.set_text(s, 0, flags=flags)
```

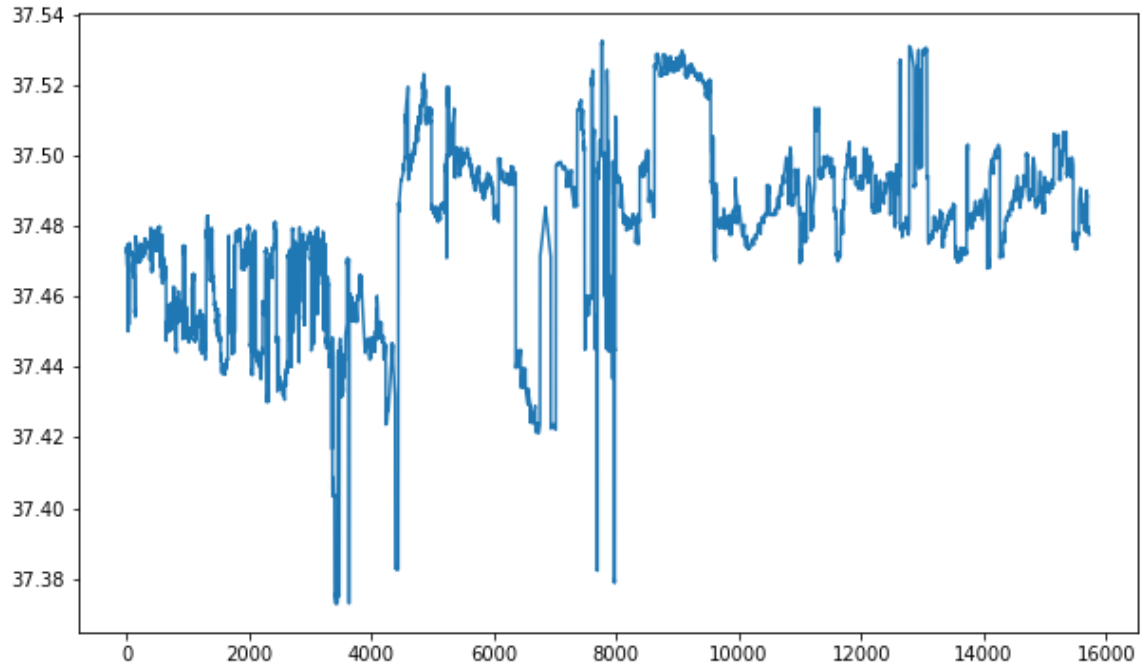


In [11]:

```
df['위도'].plot()
```

Out[11]:

<matplotlib.axes._subplots.AxesSubplot at 0x191a00cba08>

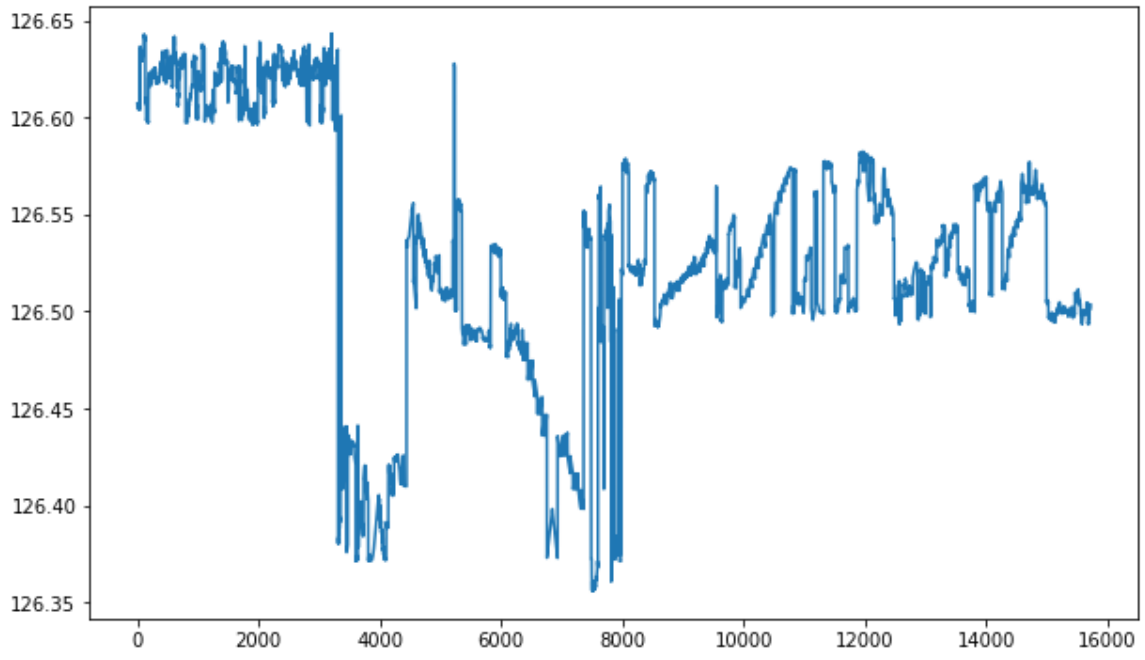


In [12]:

```
df['경도'].plot()
```

Out[12]:

<matplotlib.axes._subplots.AxesSubplot at 0x1919ff3ff48>



In [20]:

```
from pandas_datareader import wb as pdr
```

In [60]:

```
df = pdr.search('gdp.*capita.*const')
df
```

Out[60]:

	id	name	unit	source	sourceNote	sourceOrganization
646	6.0.GDPpc_constant	GDP per capita, PPP (constant 2011 internation...		LAC Equity Lab	GDP per capita based on purchasing power parit...	b'World Development Indicators (World Bank)'
9212	NY.GDP.PCAP.KD	GDP per capita (constant 2010 US$)		World Development Indicators	GDP per capita is gross domestic product divid...	b'World Bank national accounts data, and OECD ...
9214	NY.GDP.PCAP.KN	GDP per capita (constant LCU)		World Development Indicators	GDP per capita is gross domestic product divid...	b'World Bank national accounts data, and OECD ...
9216	NY.GDP.PCAP.PP.KD	GDP per capita, PPP (constant 2011 internation...		World Development Indicators	GDP per capita based on purchasing power parit...	b'World Bank, International Comparison Program...
9217	NY.GDP.PCAP.PP.KD.87	GDP per capita, PPP (constant 1987 internation...		WDI Database Archives		b''

In [28]:

```
dat = pdr.download(indicator='NY.GDP.PCAP.KD', country=['US', 'CA', 'MX'], start=2015, end=2018)
dat
```

Out[28]:

		NY.GDP.PCAP.KD
country	**year**	
Canada	**2018**	51391.708417
	2017	51150.754618
	2016	50263.834327
	2015	50279.585839
Mexico	**2018**	10403.540397
	2017	10301.357885
	2016	10205.795753
	2015	10037.201490
United States	**2018**	54579.016837
	2017	53356.236236
	2016	52534.365284
	2015	52099.269760

In [32]:

```
pdr.country_codes
```

Out[32]:

```
['AD',
 'AE',
 'AF',
 'AG',
 'AI',
 'AL',
 'AM',
 'AO',
 'AQ',
 'AR',
 'AS',
 'AT',
 'AU',
 'AW',
 'AX',
 'AZ',
 'BA',
 'BB',
 'BD',
 'BE',
 'BF',
 'BG',
 'BH',
 'BI',
 'BJ',
 'BL',
 'BM',
 'BN',
 'BO',
 'BQ',
 'BR',
 'BS',
 'BT',
 'BV',
 'BW',
 'BY',
 'BZ',
 'CA',
 'CC',
 'CD',
 'CF',
 'CG',
 'CH',
 'CI',
 'CK',
 'CL',
 'CM',
 'CN',
 'CO',
 'CR',
 'CU',
 'CV',
 'CW',
 'CX',
 'CY',
 'CZ',
 'DE',
 'DJ',
 'DK',
```

```
    'DM',
    'DO',
    'DZ',
    'EC',
    'EE',
    'EG',
    'EH',
    'ER',
    'ES',
    'ET',
    'FI',
    'FJ',
    'FK',
    'FM',
    'FO',
    'FR',
    'GA',
    'GB',
    'GD',
    'GE',
    'GF',
    'GG',
    'GH',
    'GI',
    'GL',
    'GM',
    'GN',
    'GP',
    'GQ',
    'GR',
    'GS',
    'GT',
    'GU',
    'GW',
    'GY',
    'HK',
    'HM',
    'HN',
    'HR',
    'HT',
    'HU',
    'ID',
    'IE',
    'IL',
    'IM',
    'IN',
    'IO',
    'IQ',
    'IR',
    'IS',
    'IT',
    'JE',
    'JM',
    'JO',
    'JP',
    'KE',
    'KG',
    'KH',
    'KI',
    'KM',
    'KN',
```

```
    'KP',
    'KR',
    'KW',
    'KY',
    'KZ',
    'LA',
    'LB',
    'LC',
    'LI',
    'LK',
    'LR',
    'LS',
    'LT',
    'LU',
    'LV',
    'LY',
    'MA',
    'MC',
    'MD',
    'ME',
    'MF',
    'MG',
    'MH',
    'MK',
    'ML',
    'MM',
    'MN',
    'MO',
    'MP',
    'MQ',
    'MR',
    'MS',
    'MT',
    'MU',
    'MV',
    'MW',
    'MX',
    'MY',
    'MZ',
    'NA',
    'NC',
    'NE',
    'NF',
    'NG',
    'NI',
    'NL',
    'NO',
    'NP',
    'NR',
    'NU',
    'NZ',
    'OM',
    'PA',
    'PE',
    'PF',
    'PG',
    'PH',
    'PK',
    'PL',
    'PM',
    'PN',
```

```
    'PR',
    'PS',
    'PT',
    'PW',
    'PY',
    'QA',
    'RE',
    'RO',
    'RS',
    'RU',
    'RW',
    'SA',
    'SB',
    'SC',
    'SD',
    'SE',
    'SG',
    'SH',
    'SI',
    'SJ',
    'SK',
    'SL',
    'SM',
    'SN',
    'SO',
    'SR',
    'SS',
    'ST',
    'SV',
    'SX',
    'SY',
    'SZ',
    'TC',
    'TD',
    'TF',
    'TG',
    'TH',
    'TJ',
    'TK',
    'TL',
    'TM',
    'TN',
    'TO',
    'TR',
    'TT',
    'TV',
    'TW',
    'TZ',
    'UA',
    'UG',
    'UM',
    'US',
    'UY',
    'UZ',
    'VA',
    'VC',
    'VE',
    'VG',
    'VI',
    'VN',
    'VU',
```

```
    'WF',
    'WS',
    'YE',
    'YT',
    'ZA',
    'ZM',
    'ZW',
    'ABW',
    'AFG',
    'AGO',
    'AIA',
    'ALA',
    'ALB',
    'AND',
    'ARE',
    'ARG',
    'ARM',
    'ASM',
    'ATA',
    'ATF',
    'ATG',
    'AUS',
    'AUT',
    'AZE',
    'BDI',
    'BEL',
    'BEN',
    'BES',
    'BFA',
    'BGD',
    'BGR',
    'BHR',
    'BHS',
    'BIH',
    'BLM',
    'BLR',
    'BLZ',
    'BMU',
    'BOL',
    'BRA',
    'BRB',
    'BRN',
    'BTN',
    'BVT',
    'BWA',
    'CAF',
    'CAN',
    'CCK',
    'CHE',
    'CHL',
    'CHN',
    'CIV',
    'CMR',
    'COD',
    'COG',
    'COK',
    'COL',
    'COM',
    'CPV',
    'CRI',
    'CUB',
```

```
    'CUW',
    'CXR',
    'CYM',
    'CYP',
    'CZE',
    'DEU',
    'DJI',
    'DMA',
    'DNK',
    'DOM',
    'DZA',
    'ECU',
    'EGY',
    'ERI',
    'ESH',
    'ESP',
    'EST',
    'ETH',
    'FIN',
    'FJI',
    'FLK',
    'FRA',
    'FRO',
    'FSM',
    'GAB',
    'GBR',
    'GEO',
    'GGY',
    'GHA',
    'GIB',
    'GIN',
    'GLP',
    'GMB',
    'GNB',
    'GNQ',
    'GRC',
    'GRD',
    'GRL',
    'GTM',
    'GUF',
    'GUM',
    'GUY',
    'HKG',
    'HMD',
    'HND',
    'HRV',
    'HTI',
    'HUN',
    'IDN',
    'IMN',
    'IND',
    'IOT',
    'IRL',
    'IRN',
    'IRQ',
    'ISL',
    'ISR',
    'ITA',
    'JAM',
    'JEY',
    'JOR',
```

```
    'JPN',
    'KAZ',
    'KEN',
    'KGZ',
    'KHM',
    'KIR',
    'KNA',
    'KOR',
    'KWT',
    'LAO',
    'LBN',
    'LBR',
    'LBY',
    'LCA',
    'LIE',
    'LKA',
    'LSO',
    'LTU',
    'LUX',
    'LVA',
    'MAC',
    'MAF',
    'MAR',
    'MCO',
    'MDA',
    'MDG',
    'MDV',
    'MEX',
    'MHL',
    'MKD',
    'MLI',
    'MLT',
    'MMR',
    'MNE',
    'MNG',
    'MNP',
    'MOZ',
    'MRT',
    'MSR',
    'MTQ',
    'MUS',
    'MWI',
    'MYS',
    'MYT',
    'NAM',
    'NCL',
    'NER',
    'NFK',
    'NGA',
    'NIC',
    'NIU',
    'NLD',
    'NOR',
    'NPL',
    'NRU',
    'NZL',
    'OMN',
    'PAK',
    'PAN',
    'PCN',
    'PER',
```

```
    'PHL',
    'PLW',
    'PNG',
    'POL',
    'PRI',
    'PRK',
    'PRT',
    'PRY',
    'PSE',
    'PYF',
    'QAT',
    'REU',
    'ROU',
    'RUS',
    'RWA',
    'SAU',
    'SDN',
    'SEN',
    'SGP',
    'SGS',
    'SHN',
    'SJM',
    'SLB',
    'SLE',
    'SLV',
    'SMR',
    'SOM',
    'SPM',
    'SRB',
    'SSD',
    'STP',
    'SUR',
    'SVK',
    'SVN',
    'SWE',
    'SWZ',
    'SXM',
    'SYC',
    'SYR',
    'TCA',
    'TCD',
    'TGO',
    'THA',
    'TJK',
    'TKL',
    'TKM',
    'TLS',
    'TON',
    'TTO',
    'TUN',
    'TUR',
    'TUV',
    'TWN',
    'TZA',
    'UGA',
    'UKR',
    'UMI',
    'URY',
    'USA',
    'UZB',
    'VAT',
```

```
 'VCT',
 'VEN',
 'VGB',
 'VIR',
 'VNM',
 'VUT',
 'WLF',
 'WSM',
 'YEM',
 'ZAF',
 'ZMB',
 'ZWE',
 'all',
 'ALL',
 'All']
```

In [33]:

```
df=dat['NY.GDP.PCAP.KD'].groupby(level=0).mean()
df
```

Out[33]:

```
country
Canada           50771.470800
Mexico           10236.973881
United States    53142.222029
Name: NY.GDP.PCAP.KD, dtype: float64
```

In [39]:

```
pdr.search('cell.*%').iloc[:,:2]
```

Out[39]:

	id	name
7516	IT.CEL.COVR.ZS	Population covered by mobile cellular network (%)
7571	IT.MOB.COV.ZS	Population coverage of mobile cellular telepho...

In [81]:

```
ind = ['NY.GDP.PCAP.KD']
df = pdr.download(indicator=ind, country='all', start=2018, end=2018).dropna()
df.columns = ['gdp']
print(df.tail())
```

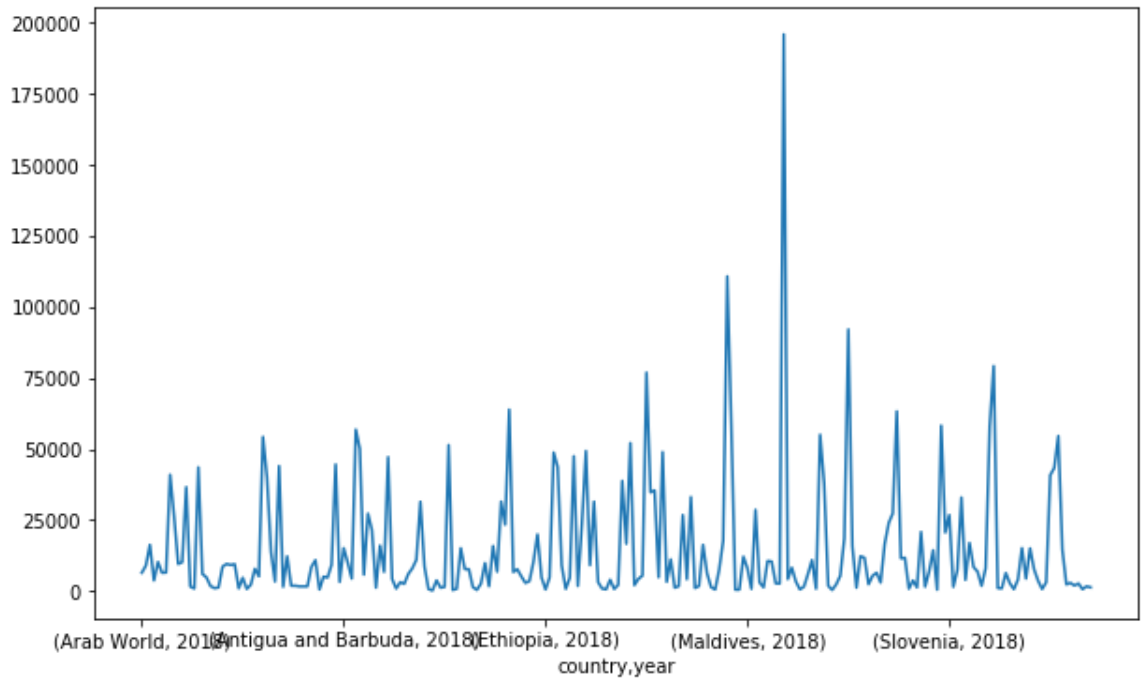
```
                              gdp
country            year
Vietnam            2018  1964.475991
West Bank and Gaza 2018  2680.318123
Yemen, Rep.        2018   667.945437
Zambia             2018  1672.345428
Zimbabwe           2018  1322.344063
```

In [82]:

```
df['gdp'].plot()
```

Out[82]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x191a33de0c8>
```



In [66]:

```
#Json
```

In [76]:

```
obj="""
{"brand":"braun",
"production_country":["United States","Spain","Germany","Canada"],
"product_kind":"Shaver",
"products":[{"name":"Series 3","features":["waterproof","safe"]},
{"name":"Series 5", "features":["flexible","comfortable","safe","deliberate"]},
{"name":"Series 7", "features":["flexible","comfortable","safe","deliberate","smart","efficien
t"]}]
}
"""
obj
```

Out[76]:

```
'₩n{"brand":"braun",₩n"production_country":["United States","Spain","Germany","Can
ada"],₩n"product_kind":"Shaver",₩n"products":[{"name":"Series 3","features":["wate
rproof","safe"]},₩n{"name":"Series 5", "features":["flexible","comfortable","saf
e","deliberate"]},₩n{"name":"Series 7", "features":["flexible","comfortable","saf
e","deliberate","smart","efficient"]}]₩n}₩n'
```

In [77]:

```
import json
result=json.loads(obj)
result
```

Out[77]:

```
{'brand': 'braun',
 'production_country': ['United States', 'Spain', 'Germany', 'Canada'],
 'product_kind': 'Shaver',
 'products': [{'name': 'Series 3', 'features': ['waterproof', 'safe']},
  {'name': 'Series 5',
   'features': ['flexible', 'comfortable', 'safe', 'deliberate']},
  {'name': 'Series 7',
   'features': ['flexible',
    'comfortable',
    'safe',
    'deliberate',
    'smart',
    'efficient']}]}
```

In [78]:

```
dff=pd.DataFrame(result['production_country'],columns=['country'])
dff
```

Out[78]:

	country
0	United States
1	Spain
2	Germany
3	Canada

In [79]:

```
df=pd.DataFrame(result['products'],columns=['name','features'])
df
```

Out[79]:

	name	features
0	Series 3	[waterproof, safe]
1	Series 5	[flexible, comfortable, safe, deliberate]
2	Series 7	[flexible, comfortable, safe, deliberate, smar...

In [ ]:

```

```