```
import pandas as pd
import glob
import re
from functools import reduce
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from collections import Counter
import matplotlib.pyplot as plt
from wordcloud import STOPWORDS, WordCloud
```

```
pd.__version__
```

```
all_files = glob.glob('/content/drive/MyDrive/Colab Notebooks/riss_data/myCabinetExcelData (*).xlsx
all_files
```

10개의 엑셀 파일을 읽어서 하나의 데이터 프레임으로 병합하고 csv 파일에 저장한다.

```
all_files_data = []

for file in all_files:
    df = pd.read_excel(file, engine='openpyxl')
    all_files_data.append(df)

all_files_data[0]
```

```
all_files_data_concat = pd.concat(all_files_data, axis=0, ignore_index=True)
all_files_data_concat
```

```
all_files_data_concat.to_csv('riss_bigdata.csv', encoding='utf-8', index=False)
```

1000개의 데이터를 하나의 csv 파일로 만들었다.

이제는 데이터 전처리

```
#제목을 추출하여  어떻게 작성되었는지 살펴보자
all_title = all_files_data_concat['제목']
all_title
```

```
#불용어   stopwords
import nltk
nltk.download('stopwords')

# stopwords.words("english")는  nltk.corpus에서 제공하는 영어 불용어 리스트를 리턴

stopWords = set(stopwords.words('english'))

#표제어 추출 작업을 제공하는 WordNetLemmatizer 객체를 만든다.

lemma = WordNetLemmatizer()


nltk.download('punkt')
nltk.download('wordnet')
nltk.download('omw-1.4')

words = []

for title in all_title :
  print(title)
  EnWords = re.sub(r"[^a-zA-Z]+", " ", str(title))
  print(EnWords)
  EnWordsToken = word_tokenize(EnWords.lower())
  print(EnWordsToken)
  EnWordsTokenStop = [w for w in EnWordsToken if w not in stopWords]
  EnWordsTokenStopLemma = [lemma.lemmatize(w) for w in EnWordsTokenStop]
  words.append(EnWordsTokenStopLemma)
  print("---"*3)



print(words)


words2 = list(reduce(lambda x, y: x+y, words))
words2
```

제목에 대한 데이터 전처리가 끝났다.

이제부터 분석을 한다.

단어 빈도...를 구해보자.

```
count = Counter(words2)   #딕셔너리객체 생성
count


word_count = dict()
for tag, counts in count.most_common(50):
  if(len(str(tag)) > 1) :
```

```
        word_count[tag]= counts
        print("%s : %d" %(tag, counts))

    # 히스토그램 그리기
    sorted_Keys = sorted(word_count, key = word_count.get, reverse = True)
    sorted_Values = sorted(word_count.values(), reverse = True)


    plt.bar(range(len(word_count)), sorted_Values, align = 'center')
    plt.xticks(range(len(word_count)), list(sorted_Keys), rotation='85')

    plt.show()



    all_files_data_concat['doc_count'] = 0
    summary_year = all_files_data_concat.groupby('출판일', as_index=False)['doc_count'].count()
    summary_year


    plt.figure(figsize = (12,5))
    plt.xlabel("year")
    plt.ylabel("doc_count")
    plt.grid(True)
    plt.plot(range(len(summary_year)), summary_year['doc_count'])
    plt.xticks(range(len(summary_year)), [text for text in summary_year['출판일']])
    plt.show()


    stopwords = set(STOPWORDS)
    wc = WordCloud(stopwords = stopwords, width = 800, height=600)
    cloud = wc.generate_from_frequencies(word_count)
    plt.figure(figsize = (8,8))
    plt.imshow(cloud)
    plt.axis('off')
    plt.show()



    wc = WordCloud(stopwords = stopwords, width = 800, height=600, background_color = 'ivory')
    cloud = wc.generate_from_frequencies(word_count)
    plt.figure(figsize = (8,8))
    plt.imshow(cloud)
    plt.axis('off')
    plt.show()


    #@title
    cloud.to_file('riss_bigdata_wordClould.jpg')
```

Colab 유료 제품 - 여기에서 계약 취소

✓ 0초 오후 9:45에 완료됨