(가제) 파이썬 데이터 분석

2.2) 트위터 API로 연관 키워드 분석하기

바로가기

- <Step1. API 호출> : 트위터 API로 데이터 가져오기
  - [API 데이터로 데이터 프레임 생성하기]
- <Step2. 추출> : 키워드 추출
  - [텍스트 데이터 전처리]
  - [nltk, konlpy를 이용한 키워드 추출]
- <Step3. 분석> : 연관 분석을 이용한 키워드 분석
  - [연관 키워드 추출하기]
  - [단어 빈도 추출하기]
- <Step4. 시각화> : 연관 키워드 네트워크 시각화
  - [연관 키워드 네트워크 시각화]

In [3]:

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

# 주피터 노트북을 실행한 브라우저에서 바로 그림을 볼 수 있게끔 만드는 것
#브라우저 내부(inline)에 바로 그려지도록 해주는 코드
%matplotlib inline

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings("ignore")
```

<Step1. API 호출> : 트위터 API로 데이터 가져오기

[API 데이터로 데이터 프레임 생성하기]

- API 사용법 참고 : https://github.com/yoonkt200/python-data-analysis/blob/master/chapter_text/(%EC%B0%B8%EA%B3%A0)%20using-twitter-api.ipynb (https://github.com/yoonkt200/python-data-analysis/blob/master/chapter_text/(%EC%B0%B8%EA%B3%A0)%20using-twitter-api.ipynb)
- 아래 코드 실행을 위해, anaconda prompt 혹은 Terminal에서 아래와 같은 패키지를 설치해 줍니다.
  - (env_name) `pip install tweepy`
- 혹은 아래의 코드로 라이브러리를 설치합니다.
- [2022.02.04] tweepy 4 버전 설치시 api.search에서 에러 발생 => tweepy 3 버전으로 설치
- pip install tweepy==3.10.0

In [4]:

```
!pip install tweepy
# tweepy 3 버전으로 설치
!pip install tweepy==3.10.0
```

```
Requirement already satisfied: tweepy in c:\users\yj\anaconda3\lib\site-packages (3.10.0)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\yj\anaconda3\lib\site-packages (from tweepy) (1.3.1)
Requirement already satisfied: six>=1.10.0 in c:\users\yj\anaconda3\lib\site-packages (from tweepy) (1.16.0)
Requirement already satisfied: requests[socks]>=2.11.1 in c:\users\yj\anaconda3\lib\site-packages (from tweepy) (2.26.0)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\yj\anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->tweepy) (3.2.0)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (3.2)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (1.26.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (2021.10.8)
Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy) (1.7.1)
Requirement already satisfied: tweepy==3.10.0 in c:\users\yj\anaconda3\lib\site-packages (3.10.0)
Requirement already satisfied: six>=1.10.0 in c:\users\yj\anaconda3\lib\site-packages (from tweepy==3.10.0) (1.16.0)
Requirement already satisfied: requests[socks]>=2.11.1 in c:\users\yj\anaconda3\lib\site-packages (from tweepy==3.10.0) (2.26.0)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\yj\anaconda3\lib\site-packages (from tweepy==3.10.0) (1.3.1)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\yj\anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->tweepy==3.10.0) (3.2.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy==3.10.0) (1.26.7)
Requirement already satisfied: idna<4,>=2.5 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy==3.10.0) (3.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->tweepy==3.10.0) (2021.10.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.
```

```
1->tweepy==3.10.0) (2.0.4)
Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in c:\users
\yj\anaconda3\lib\site-packages (from requests[socks]>=2.11.1->t
weepy==3.10.0) (1.7.1)
```

트위터 API 가져오기

- 트위터의 키워드 크롤링 기능을 사용하기 위해 트위터 앱에서 발급받은 KEY와 TOKEN 정보를 입력
- 총 4가지 정보를 입력하여 코드 실행
- tweepy의 OAuthHandler() 클래스가 자동으로 개인정보 인증 완료

In [5]:

```
import tweepy

# 발급 완료된 키 입력
CONSUMER_KEY = "LUCHdCDyarrHrZtB3TphjDPDX"
CONSUMER_SECRET = "cEVd1dTmutWmiISeXlIk7UNZ5qbRlRVJVMg9l4Elvz5GU8doAd
ACCESS_TOKEN_KEY = "169786640-GXkvSASVzzWCVgIgEtO91ENvZCEE4iGIFde9MDyW
ACCESS_TOKEN_SECRET = "xC6ghbPWdEKBmBE8Oi6cVdgVVsRzFpmFLINHLcOksa4Q

# 개인정보 인증을 요청하는 Handler
auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)

# 인증 요청을 수행
auth.set_access_token(ACCESS_TOKEN_KEY, ACCESS_TOKEN_SECRET)

# twitter API를 사용하기 위한 준비
api = tweepy.API(auth)
```

In [6]:

```
# tweepy 버전 출력
tweepy.__version__
```

Out[6]:

```
'3.10.0'
```

'손흥민' 키워드 검색

In [53]:

```
# Twitter Developer Platform의 Data dictionary 참조
# https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-mode

# twitter API를 사용하여 '손흥민' 이 포함된 트윗들을 크롤링한 뒤, 'user_mentions'와 'hasht
keyword = "개발자"

# 특정 키워드 크롤링은  search() 함수 수행
tweets = api.search(keyword)

# entities() 함수로 특정 속성의 메타데이터에 접근하여 정보 추출
for tweet in tweets:
    print(tweet.text)
    print(tweet.entities['user_mentions'])
    print(tweet.entities['hashtags'])
    print(tweet.created_at) # 트윗이 생성된 UTC 시간, 예) "2022-02-04 01:47:09"
```

```
RT @choiaemarket: [할머니의 손뜨개 포토카드 홀더 이벤트]
개발자의 할머니가 뜨개질로 만드신 포토카드 홀더예요!
[참여 방법]
1. #RT 해주세요!
2. 멘션으로 #최애마켓손뜨개최애자랑대회 해시태그를 달고 본인의 최애 자랑 + 받고…
[{'screen_name': 'choiaemarket', 'name': '최애마켓', 'id': 1430468644903944195, 'id_str': '1430468644903944195', 'indices': [3, 16]}]
[{'text': 'RT', 'indices': [82, 85]}, {'text': '최애마켓손뜨개최애자랑대회', 'indices': [100, 114]}]
2022-02-10 06:04:35
RT @kstartupceo: 저는 시니어 직원들에게도 실무를 직접 하라고 강조합니다. 시니어 개발자는 3인분 시스템을 직접 만들고, 시니어 기획자는 3인분 피쳐를 직접 설계하고, 마케터는 3인분 채널을 운영해야합니다. 매니징 하라고 데려온게 아니라…
[{'screen_name': 'kstartupceo', 'name': '스타트업 김대표', 'id': 1469635921511600134, 'id_str': '1469635921511600134', 'indices': [3, 15]}]
[]
2022-02-10 06:04:25
```

In [54]:

```
tweets
```

```
y_to_user_id_str=None, in_reply_to_screen_name=None, auth
or=User(_api=<tweepy.api.API object at 0x0000015E13621AC0
>, _json={'id': 1181578899970330625, 'id_str': '11815788999703
30625', 'name': '☒☒♡☒☒screen_name': 'ncityyoong', 'locatio
n': '엔city', 'description': '', 'url': None, 'entities': {'description':
 {'urls': []}}, 'protected': False, 'followers_count': 26, 'friends_co
unt': 62, 'listed_count': 0, 'created_at': 'Tue Oct 08 14:35:46 +00
00 2019', 'favourites_count': 8103, 'utc_offset': None, 'time_zon
e': None, 'geo_enabled': False, 'verified': False, 'statuses_coun
t': 5266, 'lang': None, 'contributors_enabled': False, 'is_translat
or': False, 'is_translation_enabled': False, 'profile_background_
color': 'F5F8FA', 'profile_background_image_url': None, 'profile
_background_image_url_https': None, 'profile_background_til
e': False, 'profile_image_url': 'http://pbs.twimg.com/profile_im
ages/1291656019731345408/u-fQD-ym_normal.jpg', 'profile_im
age_url_https': 'https://pbs.twimg.com/profile_images/129165
6019731345408/u-fQD-ym_normal.jpg', 'profile_banner_url': 'h
ttps://pbs.twimg.com/profile_banners/1181578899970330625/
1596789768', 'profile_link_color': '1DA1F2', 'profile_sidebar_bo
rder_color': 'C0DEED', 'profile_sidebar_fill_color': 'DDEEF6', 'pr
ofile_text_color': '333333', 'profile_use_background_image': Tr
```

데이터 프레임 형태로 수집

In [55]:

```
# 크롤링된 데이터를 저장할 데이터 프레임 생성
columns = ['created', 'tweet_text']
df = pd.DataFrame(columns=columns)

# 크롤링을 수행할 갯수를 입력
max_tweets = 1000

# Cursor 객체를 사용하여 크롤링 수행
# Cursor()로 keyword의 값에 해당하는 정보를  items()으로 지정한 갯수만큼 크롤링
searched_tweets = [status for status in tweepy.Cursor(api.search, q=keyword).items(

# '손흥민'이 포함된 1000개의 트윗들에서, 'text', 'created_at' 정보를 데이터 프레임으로 저장
for tweet in searched_tweets:
    tweet_json = tweet._json
    tweet_text = tweet_json['text']
    created = tweet_json['created_at']
    row = [created, tweet_text]
    series = pd.Series(row, index=df.columns) #  created, tweet_text로 시리즈를 생성
    df = df.append(series, ignore_index=True) # 생성된 시리즈를 데이터프레임에 추가
```

In [56]:

```
# 데이터 프레임 상위 5개 출력
df.head()
```

Out[56]:

	created	tweet_text
0	Thu Feb 10 06:04:35 +0000 2022	RT @choiaemarket: [할머니의 손뜨개 포토카드 홀더 이벤트]\n개발자의...
1	Thu Feb 10 06:04:25 +0000 2022	RT @kstartupceo: 저는 시니어 직원들에게도 실무를 직접 하라고 강조합니...
2	Thu Feb 10 06:04:20 +0000 2022	RT @Ghiblibli1: 미국에서 부트캠프 + 콜드이메일로 레퍼런스 받아서 개...
3	Thu Feb 10 06:03:54 +0000 2022	RT @choiaemarket: [할머니의 손뜨개 포토카드 홀더 이벤트]\n개발자의...
4	Thu Feb 10 06:03:44 +0000 2022	RT @espressivo_h: 나도 옷차림+성별 때문에 "개발자 아닌 줄 알았다....

In [57]:

```
# 데이터 프레임을 csv 로 저장
# df.to_csv("tweet_temp.csv", index=False)
```

<Step2. 추출> : 키워드 추출

[텍스트 데이터 전처리]

- 한글 문자열을 기준으로 키워드 추출 수행
- tweet_text 데이터에서 한글 문자열만을 추출하여 ko_text 생성

In [58]:

```
#df = pd.read_csv("tweet_temp.csv")
#df.head()
```

In [59]:

```
# 정규표현 라이브러리
import re

# 텍스트 정제 함수 : 한글 및 띄어쓰기 이외의 문자는 전부 제거
def text_cleaning(text):
  hangul = re.compile('[^ ㄱ-ㅣ가-힣]+') # 한글의 정규표현식
  result = hangul.sub('', text)
  return result
```

In [60]:

```
# 텍스트 정제 함수text_cleaning()를 'tweet_text' 피처에 이를 적용, 한글과 띄어쓰기만 추출하
df['ko_text'] = df['tweet_text'].apply(lambda x: text_cleaning(x))
df.head()
```

Out[60]:

	created	tweet_text	ko_text
0	Thu Feb 10 06:04:35 +0000 2022	RT @choiaemarket: [할머니의 손뜨개 포토카드 홀더 이벤트]\n개발자의...	할머니의 손뜨개 포토카드 홀더 이벤트개발자의 할머니가 뜨개질로 만드신 포토카드 ...
1	Thu Feb 10 06:04:25 +0000 2022	RT @kstartupceo: 저는 시니어 직원들에게도 실무를 직접 하라고 강조합니...	저는 시니어 직원들에게도 실무를 직접 하라고 강조합니다 시니어 개발자는 인분 시...
2	Thu Feb 10 06:04:20 +0000 2022	RT @Ghiblibli1: 미국에서 부트캠프 + 콜드이메일로 레퍼런스 받아서 개...	미국에서 부트캠프 콜드이메일로 레퍼런스 받아서 개발자 취업하신 분 인터뷰하는...
3	Thu Feb 10 06:03:54 +0000 2022	RT @choiaemarket: [할머니의 손뜨개 포토카드 홀더 이벤트]\n개발자의...	할머니의 손뜨개 포토카드 홀더 이벤트개발자의 할머니가 뜨개질로 만드신 포토카드 ...
4	Thu Feb 10 06:03:44 +0000 2022	RT @espressivo_h: 나도 옷차림+성별 때문에 "개발자 아닌 줄 알았다....	나도 옷차림성별 때문에 개발자 아닌 줄 알았다 라는 얘기를 정말 많이 들었는데 ...

[konlpy를 이용한 키워드 추출]

- 명사만 추출한 후 불용어 및 한글자 키워드 제거

In [61]:

```
from konlpy.tag import Okt
from collections import Counter

# 불용어 사전의 텍스트 문서 읽어 stopwords 리스트 생성
# 한국어 약식 불용어사전 예시 파일, 출처 - (https://www.ranks.nl/stopwords/korean)
korean_stopwords_path = "../data/korean_stopwords.txt"
with open(korean_stopwords_path, encoding='utf8') as f:
    stopwords = f.readlines()
stopwords = [x.strip() for x in stopwords] # strip()으로 공백 및 줄바꿈 제거

# 품사 중 명사만 추출하여 한글자 키워드 및 불용어 제거
# Okt(Open Korean Text)는 트위터에서 만든 오픈소스 한국어 처리기인 twitter-korean-text를

def get_nouns(x):
    # 명사만 추출
    nouns_tagger = Okt() # Okt()클래스 선언
    nouns = nouns_tagger.nouns(x)

    # 한글자 키워드 제거
    nouns = [noun for noun in nouns if len(noun) > 1]

    # 불용어를 제거
    nouns = [noun for noun in nouns if noun not in stopwords]

    return nouns
```

In [62]:

```
stopwords
```

Out[62]:

```
['아',
'휴',
'아이구',
'아이쿠',
'아이고',
'어',
'나',
'우리',
'저희',
'따라',
'의해',
'을',
'를',
'에',
'의',
'가',
'으로',
'로',
'에게',
'뿐이다'
```

In [63]:

```
# ‘ko_text’ 피처에 get_nouns() 함수를 적용하여 불용어 및 한글자를 제거한 명사 추출하여 'no
df['nouns'] = df['ko_text'].apply(lambda x: get_nouns(x))
print(df.shape) # 행과 열의 갯수를 튜플로 반환
df.head()
```

(1000, 4)

Out[63]:

	created	tweet_text	ko_text	nouns
0	Thu Feb 10 06:04:35 +0000 2022	RT @choiaemarket: [할머니의 손뜨개 포토카드 홀더 이벤트]\n개발자의...	할머니의 손뜨개 포토카드 홀더 이벤트개발자의 할머니가 뜨개질로 만드신 포토카드 ...	[할머니, 포토, 카드, 홀더, 이벤트, 개발자, 할머니, 뜨개질, 포토, 카드, ...
1	Thu Feb 10 06:04:25 +0000 2022	RT @kstartupceo: 저는 시니어 직원들에게도 실무를 직접 하라고 강조합니...	저는 시니어 직원들에게도 실무를 직접 하라고 강조합니다 시니어 개발자는 인분 시...	[시니어, 직원, 실무, 직접, 하라, 강조, 시니어, 개발자, 인분, 시스템, 직...
2	Thu Feb 10 06:04:20 +0000 2022	RT @Ghiblibli1: 미국에서 부트캠프 + 콜드이메일로 레퍼런스 받아서 개...	미국에서 부트캠프 콜드이메일로 레퍼런스 받아서 개발자 취업하신 분 인터뷰하는...	[미국, 부트캠프, 콜드, 이메일, 레퍼런스, 개발자, 취업, 인터뷰, 콜드, 이메...
3	Thu Feb 10 06:03:54 +0000 2022	RT @choiaemarket: [할머니의 손뜨개 포토카드 홀더 이벤트]\n개발자의...	할머니의 손뜨개 포토카드 홀더 이벤트개발자의 할머니가 뜨개질로 만드신 포토카드 ...	[할머니, 포토, 카드, 홀더, 이벤트, 개발자, 할머니, 뜨개질, 포토, 카드, ...
4	Thu Feb 10 06:03:44 +0000 2022	RT @espressivo_h: 나도 옷차림+성별 때문에 “개발자 아닌 줄 알았다....	나도 옷차림성별 때문에 개발자 아닌 줄 알았다 라는 얘기를 정말 많이 들었는데 ...	[옷차림, 성별, 때문, 개발자, 얘기, 정말, 형태, 개발자, 일도, 개발자]

<Step3. 분석> : 연관 분석을 이용한 키워드 분석

[연관 키워드 추출하기]

[연관 규칙]

- https://ratsgo.github.io/machine%20learning/2017/04/08/apriori/ (https://ratsgo.github.io/machine%20learning/2017/04/08/apriori/)
- 연관규칙 (아프리오리, Apriori 알고리즘) https://blog.naver.com/zzz90zzz/221807210555 (https://blog.naver.com/zzz90zzz/221807210555)
- 연관규칙은 비지도학습으로 대규모 거래 데이터로 부터 함께 구매될 규칙을 도출하여 고객이 특정 상품 구매시 이와 연관성이 높은 상품을 추천하는 것
- Apriori 알고리즘 : 간단한 성능 측정치를 이용해 거대한 DB에서 데이터간의 연관성을 찾는 알고리즘

[연관규칙에서 사용하는 3가지 통계척도]

- 1. 지지도 (support) : 특정 아이템이 데이터에서 발생하는 빈도
- 2. 신뢰도 (confidence) : 두 아이템의 연관규칙이 유용한 규칙일 가능성의 척도
- 3. 향상도 (lift) : 두 아이템의 연관 규칙이 우연인지 아닌지를 나타내는 척도
- 아래 코드 실행을 위해, anaconda prompt 혹은 Terminal에서 아래와 같은 패키지들을 설치
  - (env_name) `pip install apriori apyori`
- 혹은 아래의 코드로 라이브러리를 설치

In [64]:

```
!pip install apriori apyori
```

```
Requirement already satisfied: apriori in c:\users\yj\anaconda3\lib\site-packages (1.0.0)
Requirement already satisfied: apyori in c:\users\yj\anaconda3\lib\site-packages (1.1.2)
```

연관 분석 연습

In [65]:

```
##### 연관 분석 연습 1
from apyori import apriori

# 장바구니 형태의 데이터(트랜잭션 데이터)를 생성
transactions = [
    ['손흥민', '시소코'],
    ['손흥민', '케인'],
    ['손흥민', '케인','포체티노']
]

# 연관 분석을 수행
results = list(apriori(transactions))
for result in results:
    print(result)
```

```
RelationRecord(items=frozenset({'손흥민'}), support=1.0, ordered
_statistics=[OrderedStatistic(items_base=frozenset(), items_add=
frozenset({'손흥민'}), confidence=1.0, lift=1.0)])
RelationRecord(items=frozenset({'시소코'}), support=0.333333333
3333333, ordered_statistics=[OrderedStatistic(items_base=frozen
set(), items_add=frozenset({'시소코'}), confidence=0.33333333333
33333, lift=1.0)])
RelationRecord(items=frozenset({'케인'}), support=0.66666666666
66666, ordered_statistics=[OrderedStatistic(items_base=frozense
t(), items_add=frozenset({'케인'}), confidence=0.66666666666666
66, lift=1.0)])
RelationRecord(items=frozenset({'포체티노'}), support=0.3333333
333333333, ordered_statistics=[OrderedStatistic(items_base=froz
enset(), items_add=frozenset({'포체티노'}), confidence=0.3333333
333333333, lift=1.0)])
RelationRecord(items=frozenset({'손흥민', '시소코'}), support=0.33
33333333333333, ordered_statistics=[OrderedStatistic(items_bas
e=frozenset(), items_add=frozenset({'손흥민', '시소코'}), confidenc
e=0.3333333333333333, lift=1.0), OrderedStatistic(items_base=fr
ozenset({'손흥민'}), items_add=frozenset({'시소코'}), confidence=0.
3333333333333333, lift=1.0), OrderedStatistic(items_base=frozen
set({'시소코'}), items_add=frozenset({'손흥민'}), confidence=1.0, lift
=1.0)])
RelationRecord(items=frozenset({'케인', '손흥민'}), support=0.6666
666666666666, ordered_statistics=[OrderedStatistic(items_base=
frozenset(), items_add=frozenset({'케인', '손흥민'}), confidence=0.6
666666666666666, lift=1.0), OrderedStatistic(items_base=frozens
et({'손흥민'}), items_add=frozenset({'케인'}), confidence=0.666666
6666666666, lift=1.0), OrderedStatistic(items_base=frozenset({'케
인'}), items_add=frozenset({'손흥민'}), confidence=1.0, lift=1.0)])
RelationRecord(items=frozenset({'손흥민', '포체티노'}), support=0.3
333333333333333, ordered_statistics=[OrderedStatistic(items_ba
se=frozenset(), items_add=frozenset({'손흥민', '포체티노'}), confide
nce=0.3333333333333333, lift=1.0), OrderedStatistic(items_base=
frozenset({'손흥민'}), items_add=frozenset({'포체티노'}), confidence
=0.3333333333333333, lift=1.0), OrderedStatistic(items_base=froz
enset({'포체티노'}), items_add=frozenset({'손흥민'}), confidence=1.
```

```
0, lift=1.0)])
RelationRecord(items=frozenset({'케인', '포체티노'}), support=0.33
33333333333333, ordered_statistics=[OrderedStatistic(items_bas
e=frozenset(), items_add=frozenset({'케인', '포체티노'}), confidenc
e=0.3333333333333333, lift=1.0), OrderedStatistic(items_base=fr
ozenset({'케인'}), items_add=frozenset({'포체티노'}), confidence=0.
5, lift=1.5), OrderedStatistic(items_base=frozenset({'포체티노'}), it
ems_add=frozenset({'케인'}), confidence=1.0, lift=1.5)])
RelationRecord(items=frozenset({'케인', '손흥민', '포체티노'}), supp
ort=0.3333333333333333, ordered_statistics=[OrderedStatistic(it
ems_base=frozenset(), items_add=frozenset({'케인', '손흥민', '포체
티노'}), confidence=0.3333333333333333, lift=1.0), OrderedStatisti
c(items_base=frozenset({'손흥민'}), items_add=frozenset({'케인',
'포체티노'}), confidence=0.3333333333333333, lift=1.0), OrderedSt
atistic(items_base=frozenset({'케인'}), items_add=frozenset({'손흥
민', '포체티노'}), confidence=0.5, lift=1.5), OrderedStatistic(items_b
ase=frozenset({'포체티노'}), items_add=frozenset({'케인', '손흥민'}),
confidence=1.0, lift=1.5), OrderedStatistic(items_base=frozenset
({'케인', '손흥민'}), items_add=frozenset({'포체티노'}), confidence=0.
5, lift=1.5), OrderedStatistic(items_base=frozenset({'손흥민', '포체
티노'}), items_add=frozenset({'케인'}), confidence=1.0, lift=1.5), Or
deredStatistic(items_base=frozenset({'케인', '포체티노'}), items_ad
d=frozenset({'손흥민'}), confidence=1.0, lift=1.0)])
```

In [66]:

```
##### 연관 분석 연습 2

# 지지도 0.5, 신뢰도 0.6, 향상도 1.0 이상이면서 (손흥민, 케인) 처럼 규칙의 크기가 2 이하인 규칙
list(apriori(transactions,
        min_support=0.5,
        min_confidence=0.6,
        min_lift=1.0,
        max_length=2))
```

Out[66]:

```
[RelationRecord(items=frozenset({'손흥민'}), support=1.0, ordered
_statistics=[OrderedStatistic(items_base=frozenset(), items_add=
frozenset({'손흥민'}), confidence=1.0, lift=1.0)]),
 RelationRecord(items=frozenset({'케인'}), support=0.6666666666
666666, ordered_statistics=[OrderedStatistic(items_base=frozens
et(), items_add=frozenset({'케인'}), confidence=0.66666666666666
66, lift=1.0)]),
 RelationRecord(items=frozenset({'케인', '손흥민'}), support=0.666
6666666666666, ordered_statistics=[OrderedStatistic(items_base
=frozenset(), items_add=frozenset({'케인', '손흥민'}), confidence=0.
6666666666666666, lift=1.0), OrderedStatistic(items_base=frozen
set({'손흥민'}), items_add=frozenset({'케인'}), confidence=0.666666
6666666666, lift=1.0), OrderedStatistic(items_base=frozenset({'케
인'}), items_add=frozenset({'손흥민'}), confidence=1.0, lift=1.0)])]
```

트위터 연관 키워드 분석

In [67]:

```
# 트랜잭션 데이터를 추출
transactions = df['nouns'].tolist()
#transactions = [transaction for transaction in transactions if transaction] # 공백 문자
print(transactions)
```

```
[['할머니', '포토', '카드', '홀더', '이벤트', '개발자', '할머니', '뜨개질', '포
토', '카드', '홀더', '참여', '방법', '멘션', '최애', '마켓', '개최', '애자', '대
회', '해시태그', '본인', '최애', '자랑'], ['시니어', '직원', '실무', '직접', '하
라', '강조', '시니어', '개발자', '인분', '시스템', '직접', '시니어', '기획',
'인분', '직접', '설계', '마케터', '인분', '채널', '운영', '매니'], ['미국', '부
트캠프', '콜드', '이메일', '레퍼런스', '개발자', '취업', '인터뷰', '콜드', '이
메일', '런가', '인터뷰', '말미', '학연', '지연', '경력', '링크드인', '인맥',
'소리'], ['할머니', '포토', '카드', '홀더', '이벤트', '개발자', '할머니', '뜨개
질', '포토', '카드', '홀더', '참여', '방법', '멘션', '최애', '마켓', '개최', '애
자', '대회', '해시태그', '본인', '최애', '자랑'], ['옷차림', '성별', '때문', '개
발자', '얘기', '정말', '형태', '개발자', '일도', '개발자'], ['양쪽', '개발자',
'유입', '진짜'], ['다크', '모드', '일전', '트위터', '언급', '지금', '개발자',
'서버', '유저', '접속', '유저', '업뎃', '예정', '듯함', '공지', '이제', '토큰',
'기능', '위치', '양임'], ['상관', '일만', '자꾸', '외모', '어쩌구', '거리', '개
발자', '힙스터', '인척', '제일', '힙스터'], ['다크', '모드', '일전', '트위터',
'언급', '지금', '개발자', '서버', '유저', '접속', '유저', '업뎃', '예정', '듯함',
'공지', '이제', '토큰', '기능', '위치', '양임'], ['할머니', '포토', '카드', '홀
더', '이벤트', '개발자', '할머니', '뜨개질', '포토', '카드', '홀더', '참여',
'방법', '멘션', '최애', '마켓', '개최', '애자', '대회', '해시태그', '본인', '최
애', '자랑'], ['할머니', '포토', '카드', '홀더', '이벤트', '개발자', '할머니',
'뜨개질', '포토', '카드', '홀더', '참여', '방법', '멘션', '최애', '마켓', '개최',
```

In [68]:

```
# 연관 분석을 수행
#적당한 superset 규칙을 설정한뒤 연관 규칙을 추출하여 results에 저장

results = list(apriori(transactions,
                       min_support=0.05,
                       min_confidence=0.1,
                       min_lift=5,
                       max_length=2))
print(results)
```

```
[RelationRecord(items=frozenset({'경력', '런가'}), support=0.088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'경력'}), items_add=frozenset({'런가'}), confidence=1.0, lift=11.363636363636365), OrderedStatistic(items_base=frozenset({'런가'}), items_add=frozenset({'경력'}), confidence=1.0, lift=11.363636363636365)]), RelationRecord(items=frozenset({'경력', '레퍼런스'}), support=0.088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'경력'}), items_add=frozenset({'레퍼런스'}), confidence=1.0, lift=11.363636363636365), OrderedStatistic(items_base=frozenset({'레퍼런스'}), items_add=frozenset({'경력'}), confidence=1.0, lift=11.363636363636365)]), RelationRecord(items=frozenset({'경력', '링크드인'}), support=0.088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'경력'}), items_add=frozenset({'링크드인'}), confidence=1.0, lift=11.363636363636365), OrderedStatistic(items_base=frozenset({'링크드인'}), items_add=frozenset({'경력'}), confidence=1.0, lift=11.363636363636365)]), RelationRecord(items=frozenset({'경력', '말미'}), support=0.088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'경력'}), items_add=frozenset({'말미'}), confidence=1.0, lift=11.363636363636365), OrderedStatistic(items_base=frozenset({'말미'}), items_add=frozenset({'경력'}), confidence=1.0, lift=11.363636363636365)]), RelationRecord(items=frozenset({'말그', '경력'}), sup
```

In [69]:

```
for result in results:
    print(result)
```

```
RelationRecord(items=frozenset({'경력', '런가'}), support=0.088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'경력'}), items_add=frozenset({'런가'}), confidence=1.0, lift=11.363636363636365), OrderedStatistic(items_base=frozenset({'런가'}), items_add=frozenset({'경력'}), confidence=1.0, lift=11.363636363636365)])
RelationRecord(items=frozenset({'경력', '레퍼런스'}), support=0.088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'경력'}), items_add=frozenset({'레퍼런스'}), confidence=1.0, lift=11.363636363636365), OrderedStatistic(items_base=frozenset({'레퍼런스'}), items_add=frozenset({'경력'}), confidence=1.0, lift=11.363636363636365)])
RelationRecord(items=frozenset({'경력', '링크드인'}), support=0.088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'경력'}), items_add=frozenset({'링크드인'}), confidence=1.0, lift=11.363636363636365), OrderedStatistic(items_base=frozenset({'링크드인'}), items_add=frozenset({'경력'}), confidence=1.0, lift=11.363636363636365)])
RelationRecord(items=frozenset({'경력', '말미'}), support=0.088, ordered_statistics=[OrderedStatistic(items_base=frozenset({'경력'}), items_add=frozenset({'말미'}), confidence=1.0, lift=11.3636
```

In [70]:

```
# 네트워크 그래프의 '관계' 역항르 하는 선 생성
# network_df의 연관 분석 데이터를 기반으로 네트워크 그래프의 '관계' 역할을 하는 선을 생성
# 데이터 프레임 형태로 정리
columns = ['source', 'target', 'support']
network_df = pd.DataFrame(columns=columns)

# 규칙의 조건절을 source, 결과절을 target, 지지도를 support 라는 데이터 프레임의 피처로 변
for result in results:
    if len(result.items) == 2:
        items = [x for x in result.items] #items의 2개 값을 items 리스트에 저장
        row = [items[0], items[1], result.support] #row 에 item[0], item[1], result의 suppo
        series = pd.Series(row, index=network_df.columns) #시리즈 형태로 만들기
        network_df = network_df.append(series, ignore_index=True)# 시리즈를 network_

network_df.head()
```

Out[70]:

	source	target	support
0	경력	런가	0.088
1	경력	레퍼런스	0.088
2	경력	링크드인	0.088
3	경력	말미	0.088
4	미국	경력	0.088

[단어 빈도 추출하기]

말뭉치 추출

In [71]:

```
# 말뭉치를 추출합
tweet_corpus = "".join(df['ko_text'].tolist()) #ko_text 값을 리스트 형태로 변환 ->join() 으
print(tweet_corpus)
```

```
가 있고  유저는 접속 가능한듯함 전유저 업뎃 예정인듯함 공지 이쪽 그리고
뭣보다 이제 토큰 잠금 기능 위치 잠금 이 생긴다는 모양임  롤 다크모드 일
전쯤에 트위터에서 언급하더니 지금은 개발자 서버에 들어가 있고  유저는
접속 가능한듯함 전유저 업뎃 예정인듯함 공지 이쪽 그리고 뭣보다 이제 토
큰 잠금 기능 위치 잠금 이 생긴다는 모양임  롤 다크모드 일전쯤에 트위터에
서 언급하더니 지금은 개발자 서버에 들어가 있고  유저는 접속 가능한듯함
전유저 업뎃 예정인듯함 공지 이쪽 그리고 뭣보다 이제 토큰 잠금 기능 위치
잠금 이 생긴다는 모양임  개발자트친   롤 다크모드 일전쯤에 트위터에서 언
급하더니 지금은 개발자 서버에 들어가 있고  유저는 접속 가능한듯함 전유
저 업뎃 예정인듯함 공지 이쪽 그리고 뭣보다 이제 토큰 잠금 기능 위치 잠금
이 생긴다는 모양임  롤 다크모드 일전쯤에 트위터에서 언급하더니 지금은
개발자 서버에 들어가 있고  유저는 접속 가능한듯함 전유저 업뎃 예정인듯
함 공지 이쪽 그리고 뭣보다 이제 토큰 잠금 기능 위치 잠금 이 생긴다는 모
양임  롤 다크모드 일전쯤에 트위터에서 언급하더니 지금은 개발자 서버에
들어가 있고  유저는 접속 가능한듯함 전유저 업뎃 예정인듯함 공지 이쪽 그
리고 뭣보다 이제 토큰 잠금 기능 위치 잠금 이 생긴다는 모양임  할머니의
손뜨개 포토카드 홀더 이벤트개발자의 할머니가 뜨개질로 만드신 포토카드
홀더예요참여 방법  해주세요 멘션으로 최애마켓손뜨개최애자랑대회 해시
태그를 달고 본인의 최애 자랑  받고  롤 다크모드 일전쯤에 트위터에서 언급
하더니 지금은 개발자 서버에 들어가 있고  유저는 접속 가능한듯함 전유저
업뎃 예정인듯함 공지 이쪽 그리고 뭣보다 이제 토큰 잠금 기능 위치 잠금 이
생긴다는 모양임  롤 다크모드 일전쯤에 트위터에서 언급하더니 지금은 개
```

In [72]:

```
from konlpy.tag import Okt
from collections import Counter

# 명사 키워드를 추출, 빈도수 계산
nouns_tagger = Okt()
nouns = nouns_tagger.nouns(tweet_corpus)
count = Counter(nouns)

# 한글자 키워드를 제거
remove_char_counter = Counter({x : count[x] for x in count if len(x) > 1})
print(remove_char_counter)
```

```
Counter({'개발자': 1127, '포토': 673, '홀더': 673, '할머니': 672, '카드': 672, '최애': 672, '본인': 339, '이벤트': 337, '방법': 337, '뜨개질': 336, '참여': 336, '멘션': 336, '마켓': 336, '개최': 336, '애자': 336, '대회': 336, '해시태그': 336, '자랑': 336, '트친': 243, '유저': 236, '인터뷰': 185, '콜드': 176, '이메일': 176, '지금': 123, '미국': 120, '모드': 119, '접속': 118, '예정': 118, '이제': 118, '토큰': 118, '다크': 117, '일전': 117, '트위터': 117, '언급': 117, '서버': 117, '업뎃': 117, '듯함': 117, '공지': 117, '기능': 117, '위치': 117, '양임': 116, '취업': 101, '소리': 89, '부트캠프': 88, '레퍼런스': 88, '런가': 88, '말미': 88, '학연': 88, '지연': 88, '경력': 88, '링크드인': 88, '인맥': 88, '때문': 57, '정말': 56, '우리': 56, '얘기': 55, '옷차림': 53, '성별': 53, '형태': 53, '일도': 53, '얼마나': 53, '영상': 53, '개발': 43, '시스템': 39, '시니어': 36, '직접': 36, '인분': 36, '능력': 28, '북디자이너': 26, '커뮤': 26, '구현': 26, '날북': 26, '자컾': 26, '외치': 26, '다른사람': 26, '의뢰': 26, '게임': 21, '생각': 19, '기획': 18, '회사': 18, '다른': 18, '트친소': 16, '싱글': 15, '엔지니어': 15, '권장': 15, '대한': 14, '여성': 14, '브라우저': 14, '주소창': 14, '치면': 14, '직원': 13, '운영': 13, '기술': 13, '대해': 13, '실무': 12, '하라': 12, '강조': 12, '설계': 12, '마케터': 12, '채널': 12, '매니': 12, '공부': 12, '프로그램': 10, '협업': 10, '노르웨이': 10, '부분': 9, '시작': 8, '진행': 8, '질문': 8, '이야기': 8, '피해자': 8, '면접': 8, '다시': 8, '현재': 7, '디자이너': 7, '바로': 7, '잘못': 7, '프로젝트': 7, '기업': 7, '경험': 7, '단골': 7, '진짜': 6, '코인': 6, '추천': 6, '시간': 6, '환영': 6, '꿍햄': 6, '어디가': 6, '러시': 6, '가요': 6, '미래': 6, '기존': 6, '가지': 5, '오류': 5, '사실': 5, '위주': 5, '소통': 5, '준비': 5, '팔로우': 5, '주시': 5, '맞팔': 5, '자트': 5, '알티': 5, '타네': 5, '해외': 5, '하나': 5, '투자': 5, '업계': 5, '사용': 5, '후기': 5, '아이디어': 5, '제공': 5, '뉴비': 4, '백엔드': 4, '대학교': 4, '휴학': 4, '한국': 4, '블로그': 4, '사람': 4, '디자인': 4, '요즘': 4, '모두': 4, '정도': 4, '그냥': 4, '잡담': 4, '닷컴버블': 4, '아예': 4, '안해': 4, '실수': 4, '사기': 4, '의심': 4, '대감': 4, '폰지사기': 4, '팟캐스트': 4, '개인': 4, '결제': 4, '기분': 4, '친구': 4, '서비스': 4, '과정': 4, '어제': 4, '배그': 4, '지니': 4, '타임': 4, '구합': 4, '만들기': 3, '여러분': 3, '발행': 3, '프로그래밍': 3, '부터': 3, '상황': 3, '연봉': 3, '주니어': 3, '테크': 3, '영어': 3, '도움': 3, '중심': 3, '주제': 3, '스프링': 3, '보고': 3, '내용': 3, '채용': 3, '프로그래머': 3, '로그': 3, '지식': 3, '선물': 3, '목표': 3, '제대로': 3, '스타일': 3, '방통대': 3, '편입': 3, '컨셉': 3, '퍼즐': 3, '설명': 3, '재활용': 3, '케이스': 3, '동료': 3, '주기도': 3, '클라우드': 3, '귀요미': 3, '사랑': 3, '어디': 3, '취직': 3, '일단': 3, '소개': 3, '인생': 3, '피터': 3, '온보딩': 3, '정리': 3, '최적화': 3, '기사': 3, '힙스터': 2, '인척': 2, '얼티밋': 2, '스쿨': 2, '한지': 2, '분위기': 2, '요식업': 2, '오타': 2, '지원': 2, '보드': 2, '배포': 2, '물량': 2, '이유': 2, '역시': 2, '이름': 2, '한번': 2, '탐라': 2, '최소한': 2, '게이머': 2, '유지': 2, '보수': 2, '의미': 2, '안정': 2, '상주': 2, '보지': 2, '솔루션': 2, '체력': 2, '본캐': 2, '프로': 2, '오픈소스': 2, '처음': 2, '선결':
```

2, '조건': 2, '동시': 2, '하니': 2, '편이': 2, '코드': 2, '로써': 2, '몇개': 2, '듣기': 2, '전반': 2, '프론트엔드': 2, '보통': 2, '가끔': 2, '프론트': 2, '강의': 2, '산업': 2, '모든': 2, '해커': 2, '공유': 2, '암호': 2, '화폐': 2, '엔딩': 2, '고민': 2, '추첨': 2, '사이': 2, '마이크로소프트': 2, '주석': 2, '스타트업': 2, '이해': 2, '구두': 2, '제품': 2, '다음': 2, '시안': 2, '원칙': 2, '발표': 2, '책임': 2, '일인': 2, '누구': 2, '출시': 2, '스팀': 2, '미리': 2, '기기': 2, '수령': 2, '국내': 2, '사도': 2, '포함': 2, '설정': 2, '비트코인': 2, '응원': 2, '완전': 2, '귀여미': 2, '뉴스': 2, '유료': 2, '백신': 2, '야근': 2, '빌런': 2, '룸살롱': 2, '선언': 2, '선배': 2, '대선': 2, '캠프': 2, '지급': 2, '약속': 2, '인건비': 2, '선거': 2, '국고': 2, '보조금': 2, '그때': 2, '절반': 2, '만화': 2, '일러스트': 2, '완성': 2, '양쪽': 1, '유입': 1, '상관': 1, '일만': 1, '자꾸': 1, '외모': 1, '어쩌구': 1, '거리': 1, '제일': 1, '편집자': 1, '촬영': 1, '깅러': 1, '소비러': 1, '장비': 1, '친소': 1, '회사원': 1, '겹트': 1, '도저히': 1, '어지간': 1, '트윗': 1, '기준': 1, '반도체': 1, '전공자': 1, '로볼': 1, '과학': 1, '재미': 1, '부도': 1, '포탈': 1, '지프': 1, '보도자료': 1, '소프트웨어': 1, '진흥': 1, '협회': 1, '양성': 1, '국비': 1, '무료': 1, '교육': 1, '출처': 1, '상공': 1, '머리': 1, '화이팅': 1, '블루투스': 1, '뇌파': 1, '기전': 1, '가격': 1, '투하': 1, '걱정': 1, '레이븐': 1, '사전': 1, '채굴': 1, '소신발': 1, '유닉스': 1, '사과': 1, '집단': 1, '최면': 1, '커밋': 1, '금발': 1, '전문직': 1, '반항': 1, '가기': 1, '프레임': 1, '직업': 1, '덕질': 1, '일산': 1, '치과': 1, '창원시': 1, '렌섬': 1, '웨어': 1, '웹사이트': 1, '홈홈': 1, '다운중': 1, '셀프': 1, '헤어': 1, '코팅': 1, '송탄': 1, '아리': 1, '스타': 1, '외풍': 1, '방지': 1, '동읍': 1, '군복': 1, '멕시코': 1, '항공': 1, '찾기': 1, '적혈구': 1, '수치': 1, '소량': 1, '뱃지': 1, '제작': 1, '남해': 1, '일주일': 1, '살기': 1, '근무': 1, '대충': 1, '출근': 1, '분전': 1, '마음': 1, '노트북': 1, '스티커': 1, '자마자': 1, '잔뜩': 1, '감성': 1, '어째서': 1, '문제': 1, '왜냐면': 1, '사파리': 1, '자판': 1, '법도': 1, '예전': 1, '인정': 1, '도태': 1, '운영자': 1, '참고': 1, '일지': 1, '요큐': 1, '룡톱': 1, '중복': 1, '예술': 1, '심리학': 1, '연관': 1, '서로': 1, '딩코님': 1, '도구로': 1, '로스쿨': 1, '패스': 1, '문과': 1, '여기': 1, '자칭': 1, '공대생': 1, '월말': 1, '확대': 1, '인간': 1, '확신': 1, '달러': 1, '안내': 1, '평균': 1, '대기업': 1, '신입': 1, '서민': 1, '미여닌데': 1, '가슴': 1, '뻐렁친다': 1, '스택': 1, '단상': 1, '배우': 1, '습관': 1, '여지': 1, '개념': 1, '예시': 1, '부트': 1, '김영한': 1, '만원': 1, '이번': 1, '내내': 1, '오늘': 1, '대부분': 1, '유튜브': 1, '말씀': 1, '회고': 1, '위유': 1, '성공': 1, '여러가지': 1, '스타폭스': 1, '예외': 1, '일리': 1, '알고리즘': 1, '발명': 1, '세계': 1, '최초': 1, '에이': 1, '러브레이스': 1, '전문가': 1, '인재': 1, '확보': 1, '평가': 1, '얼른': 1, '기적': 1, '소요': 1, '이점': 1, '문서': 1, '화가': 1, '주먹구구': 1, '마르스': 1, '지혁': 1, '대표': 1, '이하': 1, '건승': 1, '기원': 1, '잡부': 1, '번의': 1, '아이템': 1, '합의': 1, '여러': 1, '당장': 1, '메인': 1, '관심': 1, '역량': 1, '혼자': 1, '어찌': 1, '또한': 1, '당시': 1, '구글': 1, '아마존': 1, '이동': 1, '포스트': 1, '금요일': 1, '목요일': 1, '업뎃해': 1, '주말': 1, '평일': 1, '해결': 1, '월요일': 1, '이유지': 1, '잔업': 1, '자아': 1, '현질': 1, '노조': 1, '각오': 1, '노무사': 1, '경고': 1, '치킨': 1, '커피집': 1, '작업실': 1, '어차피': 1, '마지막': 1, '그게': 1, '조금': 1, '힐링': 1, '도트': 1, '퍼즐게임': 1, '희망': 1, '각기': 1, '훈련': 1, '법등': 1, '조언': 1, '후원': 1, '다섯': 1, '제이': 1, '서적': 1, '여캐': 1, '기아': 1, '말임': 1, '마케터들': 1, '직무': 1, '시각': 1, '충돌': 1, '발생': 1, '급글': 1, '마찬가지': 1, '인격': 1, '마법사': 1, '바드': 1, '걸음걸이': 1, '매번': 1, '지누': 1, '여자': 1, '본적': 1, '외신': 1, '비즈니스': 1, '클래스': 1, '리움': 1, '플랫폼': 1, '하드': 1, '비영리': 1, '재단': 1, '전환': 1, '스토어': 1, '영역': 1, '품질': 1, '안전': 1, '보안': 1, '정보': 1, '보호': 1, '공정': 1, '성과': 1, '투명': 1, '선택': 1, '먼저': 1, '따라서': 1, '제작사': 1, '규제': 1, '선제': 1, '대응': 1, '위해': 1, '오픈': 1, '앱스토어': 1, '자사': 1, '강제': 1, '배드': 1, '우롱': 1, '사마의': 1, '알람': 1, '야호': 1, '누군가': 1, '요구': 1, '프레': 1, '미스': 1, '젠킨스': 1, '민준': 1, '비중': 1, '포기': 1, '한국인': 1, '언니': 1, '오빠': 1, '세상': 1, '생명체': 1, '존재': 1, '최강': 1, '우주': 1, '뿌셔': 1, '만수무강': 1, '범규': 1, '곰

돌이': 1, '자체': 1, '아기': 1, '아주': 1, '찰떡': 1, '쫀쫀': 1, '완죤': 1, '커뮤니티': 1, '전이': 1, '부합': 1, '무엇': 1, '계속': 1, '초대': 1, '손님': 1, '라면': 1, '최신': 1, '트랜드': 1, '느낌': 1, '항상': 1, '수준': 1, '에피소드': 1, '사촌동생': 1, '즉흥': 1, '가족': 1, '친지': 1, '애플': 1, '계정': 1, '등록': 1, '계정은': 1, '아이디': 1, '리딤': 1, '정색': 1, '초과근무': 1, '노예': 1, '일상': 1, '똥글': 1, '장려': 1, '자주': 1, '야간': 1, '년전': 1, '폐쇄병동': 1, '일이': 1, '순간': 1, '마법': 1, '재질': 1, '토니스타크': 1, '경쟁': 1, '본부장': 1, '파커': 1, '스타크': 1, '무기': 1, '제조': 1, '주로': 1, '화학무기': 1, '쪽임': 1, '재개발': 1, '스파이더맨': 1, '활동': 1, '아이언맨': 1, '보스': 1, '프리랜서': 1, '코파': 1, '운더': 1, '여친': 1, '시위': 1, '유니콘': 1, '창업가': 1, '우버': 1, '창업': 1, '자도': 1, '가라': 1, '강요': 1, '사업': 1, '뭔가': 1, '주변': 1, '중인': 1, '래핑': 1, '거래': 1, '로열티': 1, '수수료': 1, '영국인': 1, '철자': 1, '따름': 1, '달코': 1, '메이코': 1, '제발': 1, '새벽': 1, '세이브': 1, '로딩': 1, '버그': 1, '수정': 1, '어서': 1, '입문': 1, '그래픽': 1, '아마추어': 1, '대학생': 1, '구조': 1, '프로세싱': 1, '차세대': 1, '객체': 1, '관계': 1, '데이터베이스': 1, '관련': 1, '퍼블리시': 1, '전과': 1, '확인': 1, '칼라': 1, '오디': 1, '건강': 1, '한국어': 1, '로서': 1, '해킹': 1, '도라도': 1, '스크립트': 1, '키디': 1, '파파': 1, '이즈리얼': 1, '마오카이': 1, '충격': 1, '딥스': 1, '로버트': 1, '말론': 1, '박사': 1, '총총': 1})

단어 빈도 점수 추가

In [73]:

```
# 키워드와 키워드 빈도 점수를 'node', 'nodesize' 라는 데이터 프레임의 피처로 생성
node_df = pd.DataFrame(remove_char_counter.items(), columns=['node', 'nodesize'])
node_df = node_df[node_df['nodesize'] >= 50] # 시각화의 편의를 위해 'nodesize' 50 이하
node_df.head()
```

Out[73]:

	node	nodesize
0	할머니	672
1	포토	673
2	카드	672
3	홀더	673
4	이벤트	337

<Step4. 시각화> : 연관 키워드 네트워크 시각화

[연관 키워드 네트워크 시각화]

- 아래 코드 실행을 위해, anaconda prompt 혹은 Terminal에서 아래와 같은 패키지를 설치해 줍니다.
  - (env_name) `pip install networkx`
- 혹은 아래의 코드로 라이브러리를 설치합니다.
- networkx 버전 2.3 이하를 설치하는 경우, 아래 코드를 다음과 같이 바꿔주어야 합니다.
  - `sizes = [G.nodes[node]['nodesize']*25 for node in G]` -> `sizes = [G.node[node]['nodesize']*25 for node in G]`

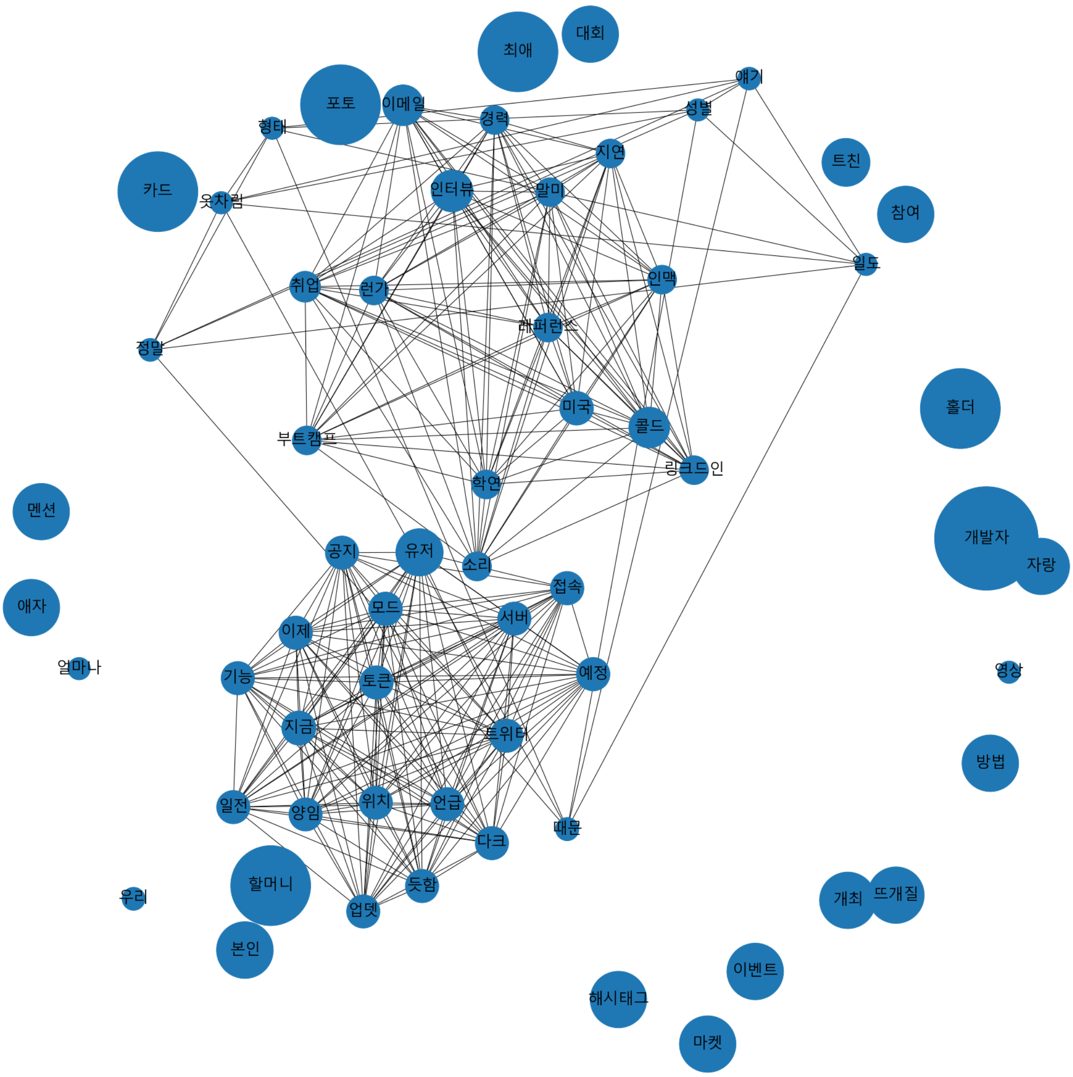
In [74]:

```
#네트워크 시각화를 위한 라이브러리 설치
!pip install networkx
```

```
Requirement already satisfied: networkx in c:\users\yj\anaconda
3\lib\site-packages (2.6.3)
```

In [75]:

```
import networkx as nx
plt.figure(figsize=(25,25))

# networkx 그래프 객체를 생성
G = nx.Graph()

# node_df의 키워드 빈도수를 데이터로 하여, 네트워크 그래프의 '노드' 역할을 하는 원을 생성 ac
for index, row in node_df.iterrows():#node_df 데이터 프레임의 행을 순회하면서 각 행의 값
    G.add_node(row['node'], nodesize=row['nodesize'])

# network_df의 연관 분석 데이터를 기반으로, 네트워크 그래프의 '관계' 역할을 하는 선을 생성
for index, row in network_df.iterrows(): #.add_weighted_edges_from
    G.add_weighted_edges_from([(row['source'], row['target'], row['support'])])

# 그래프 디자인과 관련된 파라미터를 설정
pos = nx.spring_layout(G, k=0.6, iterations=50)
# nx.draw(G, pos=pos)
sizes = [G.nodes[node]['nodesize']*25 for node in G]
nx.draw(G, pos=pos, node_size=sizes)


# Windows 사용자는 AppleGothic 대신,'Malgun Gothic'. 그 외 OS는 OS에서 한글을 지원하
# nx.draw_networkx_labels(G, pos=pos, font_family='NanumGothic', font_size=25)
nx.draw_networkx_labels(G, pos=pos, font_family='Malgun Gothic', font_size=25)

# 그래프를 출력
ax = plt.gca()
plt.show()
```


최애
대회
얘기
포토
이메일
형태
경력
성별
지연
트친
카드
옷차림
인터뷰
말미
참여
인맥
정말
홀더
미국
콜드
부트캠프
링크드인
학연
멘션
공지
유저
소리
개발자
자랑
접속
모드
서버
애자
이제
얼마나
기능
토큰
예정
영상
지금
트위터
방법
일전
위치
언급
다크
때문
듯함
할머니
우리
업뎃
개최
뜨개질
본인
이벤트
해시태그
마켓

In [ ]:

In [ ]: