TUGAS III

PENGANTAR DATA MINING

Nama Anggota:

- **Maftuh Mashuri (11160940000076)**
- **Zahrotul Aulia (11160940000024)**
- **Alif Tito SA (11160940000052)**
- **Khairul Umam (11160940000073)**
- **Fathi Syuhada (11150940000001)**

Import modul

In [43]:
```python
import json
import pandas as pd
import nltk
from nltk.tokenize import TweetTokenizer as tw_tokenizer
from unidecode import unidecode
import time
```

Pengoperasian file

Membuka file dan memasukkan data tweet json dan menyimpannya kedalam variabel list *twetts_data*

In [44]:
```python
tweets_data = [] # Membuat list kosong untuk menyimpan data json perbaris
tweets_file = open('data/dataset.txt', "r") # membuka file
for line in tweets_file:
    try:
        tweet = json.loads(line)  # Membaca data dalam format json dari file perbaris
        tweets_data.append(tweet) # Menambahkan data dari file ke dalam list
    except:
        continue
tweets_file.close()
```

1. Mapping data json kedalam bentuk dataframe

In [45]:
```python
tweets = pd.DataFrame()
# Informasi tweet
tweets['created_at'] = list(map(lambda tweet: tweet['created_at'], tweets_data))
tweets['id_str'] = list(map(lambda tweet: tweet['id_str'], tweets_data))
tweets['text'] = list(map(lambda tweet: tweet['text'], tweets_data))
tweets['source'] = list(map(lambda tweet: tweet['source'], tweets_data))
tweets['lang'] = list(map(lambda tweet: tweet['lang'], tweets_data))

# Informasi user
tweets['user_id'] = list(map(lambda tweet: tweet['user']['id_str'], tweets_data))
tweets['user_name'] = list(map(lambda tweet: tweet['user']['name'], tweets_data))
tweets['user_screen_name'] = list(map(lambda tweet: unidecode(tweet['user']['screen_name']), tweets_data))
tweets['user_location'] = list(map(lambda tweet: tweet['user']['location'], tweets_data))
tweets['user_url'] = list(map(lambda tweet: tweet['user']['url'], tweets_data))
tweets['user_description'] = list(map(lambda tweet: tweet['user']['description'], tweets_data))
tweets['user_followers_count'] = list(map(lambda tweet: tweet['user']['followers_count'], tweets_data))
tweets['user_friends_count'] = list(map(lambda tweet: tweet['user']['friends_count'], tweets_data))
tweets['user_favourites_count'] = list(map(lambda tweet: tweet['user']['favourites_count'], tweets_data))
tweets['user_statuses_count'] = list(map(lambda tweet: tweet['user']['statuses_count'], tweets_data))
tweets['user_created_at'] = list(map(lambda tweet: tweet['user']['created_at'], tweets_data))

# Informasi Tempat
tweets['address'] = list(map(lambda tweet: tweet['place']['full_name'] if tweet['place'] != None else None, tweets_data))
tweets['country'] = list(map(lambda tweet: tweet['place']['country'] if tweet['place'] != None else None, tweets_data))

# Entities
hashtags = list(map(lambda tweet: tweet['entities']['hashtags'] if tweet['entities'] != None else None, tweets_data))
tweets['hashtags'] = list(map(lambda tweet : ', '.join(list(map(lambda tw : tw['text'], tweet))), hashtags))

urls = list(map(lambda tweet: tweet['entities']['urls'] if tweet['entities'] != None else None, tweets_data))
tweets['url'] = list(map(lambda tweet : ', '.join(list(map(lambda tw : tw['url'], tweet))), urls))

mentions = list(map(lambda tweet: tweet['entities']['user_mentions'] if tweet['entities'] != None else None, tweets_data))
tweets['mentions'] = list(map(lambda tweet : ', '.join(list(map(lambda tw : tw['screen_name'], tweet))), mentions))

# Informasi quoted status
tweets['quoted_status_id'] = list(map(lambda tweet : tweet['quoted_status']['id_str'] if 'quoted_status' in tweet.keys() else None, tweets_data))
tweets['quoted_status_name'] = list(map(lambda tweet : tweet['quoted_status']['user']['name'] if 'quoted_status' in tweet.keys() else None, tweets_data))
tweets['quoted_status_screen_name'] = list(map(lambda tweet : tweet['quoted_status']['user']['screen_name'] if 'quoted_status' in tweet.keys() else None, tweets_data))
tweets['quoted_status_text'] = list(map(lambda tweet : tweet['quoted_status']['text'] if 'quoted_status' in tweet.keys() else None, tweets_data))
tweets['quoted_status_quote_count'] = list(map(lambda tweet : tweet['quoted_status']['quote_count'] if 'quoted_status' in tweet.keys() else None, tweets_data))
tweets['quoted_status_reply_count'] = list(map(lambda tweet : tweet['quoted_status']['reply_count'] if 'quoted_status' in tweet.keys() else None, tweets_data))
tweets['quoted_status_retweet_count'] = list(map(lambda tweet : tweet['quoted_status']['retweet_count'] if 'quoted_status' in tweet.keys() else None, tweets_data))
tweets['quoted_status_favorite_count'] = list(map(lambda tweet : tweet['quoted_status']['favorite_count'] if 'quoted_status' in tweet.keys() else None, tweets_data))

tweetsIn = tweets[tweets.lang == 'in']
tweetsIn.to_csv("data/" + str(time.time()) + "_export_dataframe.csv") # Eksport dataframe ke csv
# tweetsIn.to_excel("data/" + str(time.time()) + '_export_dataframe.xlsx') # Eksport dataframe ke excel
```

Pengoperasian stopword

- Membuka file stopword(indonesia, inggris, noise)
- kemudian menggabungkan semua kata yang ada di ketiga file sehingga menjadi satu teks panjang dan menyimpannya kedalam variabel *stopword_file_all*
- kemudian teks panjang di tokenize(dipisah perkata) dan menyimpannya ke dalam variabel list *stopwords*

In [46]:
```python
stopword_file1 = open('stopword/stopword_id.txt', "r").read() # Membuka file stopword bahasa indonesia dan menjadikan isi file tersebut sebagai string
stopword_file2 = open('stopword_en/stopwords_en.txt', "r").read()  # Membuka file stopword bahasa inggris dan menjadikan isi file tersebut sebagai string
stopword_file3 = open('stopword_noise/stopword_noise.txt', "r").read()  # Membuka file stopword noise dan menjadikan isi file tersebut sebagai string
stopword_file_all = stopword_file1 + stopword_file2 + stopword_file3  # Menggabungkan ketiga string stopword sebelumnya kedalam satu string
stopwords = stopword_file_all.split('\n') # Memisahkan kata dalam string yang sudah digambungkan berdasarkan baris
# print(stopwords)
```

Pengoperasian slangwords

- Membuka file slangword(*colloquial-indonesian-lexicon.csv* dan *20190327_slangword.txt*)
- File yang pertama membuka dengan modul pandas dan mengkonfersinya kedalam dataframe, kemudian menyimpan masing-masing kata kedalam variabel dictionary *slangwords*
- File yang kedua sama halnya pengoperasian pada file stopwords kemudian menyimpan masing-masing kata kedalam variabel dictionary *slangwords*

In [47]:
```python
slangwords = dict() # Membuat dictionary kosong untuk menyimpan kata slang dan formal sebagai key dan value
slangwords_dataframe = pd.read_csv('slangword/colloquial-indonesian-lexicon.csv') # Membuka file csv yang berisi kata slang dan formal dan mengkonversi kedalam dataframe
for slang, formal in zip(slangwords_dataframe['slang'], slangwords_dataframe['formal']):
    slangwords[slang] = formal # Mapping kata slang dan formal dan memasukkan ke dalam dictionary secara berulang

slangword_file = open('slangword/slangword.txt', "r").read() # Membuka file yang berisi kata slang dan kata formal dan mengkonversi kedalam string
slangwords_text = slangword_file.split('\n') # Memisahkan kata berdasarkan baris namun kata slang dan kata formal masih belum terpisah. output : (['slang:formal', ...])
#print(slangwords_text)
for slang in slangwords_text:
    split_slang = slang.split(":") # Memisahkan semua kata slang dan kata formal berdasarkan "titik dua (:)"
    slangwords[split_slang[0]] = split_slang[1] # Mapping semua kata slang dan kata formal ke dalam dictionary. Output : {'slang' : 'formal', ...}
#print(slangwords)
```

2-3. Tokenisasi dan Filtering

- Mapping semua tweet dan menyimpannya kedalam variabel list *array_text*
- Menggabungkan semua teks kedalam satu teks panjang *long_text*
- Memisahkan teks perkata
- Menghapus simbol, ASCII, link (https: atau www.), dan kata lainnya

In [48]:
```python
array_text = list(map(lambda tweet: unidecode(tweet).lower(), tweets['text'])) # Mapping semua text twitter dan memasukan ke dalam list. Output : ['teks panjang ...', 'teks panjang', ...]
long_text = ' '.join(array_text) # Menggabungkan semua text yang berada dalam list kedalam satu text
tokenized = tw_tokenizer().tokenize(long_text) # Memisahkan kata dalam text berasarkan "spasi"
filtered_alfanumeric = [w for w in tokenized if w.isalnum()] # Filtering kata yang hanya berisi karakter a-z dan 0-9 (Menghapus url, hashtag, mention)
print(filtered_alfanumeric[:100])
```

```
['whahaha', 'iya', 'sih', 'bener', 'ledakan', 'mercon', 'besar', 'menghancurkan', 'rumah', 'nanang', 'menewaskan', 'asih', 'pembantunya', 'se', 'kalideres', 'arah', 'batu', 'ceper', 'tangerang', 'tol', 'cawang', 'tmii', 'cibubur', 'bogor', 'ciawi', 'well', 'makin', 'kesini', 'ku', 'tahu', 'yang', 'mana', 'yang', 'minta', 'tolong', 'dan', 'yang', 'manfaatin', 'doang', 'honestly', 'setelah', 'balik', 'ke', 'indonesia', 'hal', 'yg', 'paling', 'berasa', 'beda', 'di', 'gw', 'itu', 'adalah', 'cuaca', 'astagah', 'tiap', 'jam', 'keringeta', 'kang', 'boong', 'di', 'bohongin', 'gmn', 'rasanya', 'dikecewain', 'tapi', 'tetep', 'cinta', 'sama', 'hehe', 'dasar', 'akuu', 'hee', 'aku', 'jadi', 'inget', 'minggu', 'kemaren', 'ga', 'sengaja', 'abis', 'nginjek', 'kocheng', 'ya', 'allah', 'kochengnya', 'biasa', 'a', 'yuhh', 'winterfell', 'sunyi', 'lah', 'donnaruma', 'ngapa', 'itu', 'udh', 'diganti', 'aja', 'kalo', 'ga']
```

4. Stopword removal

Menghapus kata-kata yang sering muncul dan tidak memiliki makna (yang, di, kan)

In [49]:
```python
removed_stopwords = [w for w in filtered_alfanumeric if w not in stopwords]
# Filtering data dengan menghapus kata yang tidak bermakna (Stopword yang diperoleh dari file)
print(removed_stopwords[:100])
```

```
['whahaha', 'bener', 'ledakan', 'mercon', 'menghancurkan', 'rumah', 'nanang', 'menewaskan', 'asih', 'pembantunya', 'kalideres', 'arah', 'batu', 'ceper', 'tangerang', 'tol', 'cawang', 'tmii', 'cibubur', 'bogor', 'ciawi', 'kesini', 'ku', 'tolong', 'manfaatin', 'honestly', 'indonesia', 'yg', 'berasa', 'beda', 'gw', 'cuaca', 'astagah', 'jam', 'keringeta', 'kang', 'boong', 'bohongin', 'gmn', 'dikecewain', 'tetep', 'cinta', 'dasar', 'akuu', 'hee', 'inget', 'minggu', 'kemaren', 'ga', 'sengaja', 'abis', 'nginjek', 'kocheng', 'allah', 'kochengnya', 'yuhh', 'winterfell', 'sunyi', 'donnaruma', 'ngapa', 'udh', 'diganti', 'ga', 'bales', 'tdr', 'ga', 'tdr', 'siang', 'tidur', 'siang', 'dibilang', 'tdr', 'mulu', 'ga', 'tdr', 'siang', 'dikasianin', 'bucin', 'banget', 'gak', 'opo', 'opo', 'iso', 'kecuali', 'siji', 'tangi', 'esuk', '1', 'nggak', 'dikasih', 'nggak', 'ngember', 'nggak', 'cerita', 'alesannya', 'simpl', 'nggak', 'baca', 'sepotong', 'berita']
```

5. Handling SlangWord

Mengubah kata-kata yang sudah melalui tahap *Stopword removal* dari kata slang menjadi kata formal

In [50]:
```python
handled_slangword = list(map(lambda w : slangwords[w] if w in slangwords.keys() else w, removed_stopwords))
# Mengubah kata slang menjadi kata formal (kata slang dan kata formal yang diperoleh dari dictionary)
print(handled_slangword[:100])
```

```
['whahaha', 'benar', 'ledakan', 'mercon', 'menghancurkan', 'rumah', 'nanang', 'menewaskan', 'asih', 'pembantunya', 'kalideres', 'arah', 'batu', 'pendek', 'tangerang', 'tol', 'cawang', 'tmii', 'cibubur', 'bogor', 'ciawi', 'kesini', 'ku', 'tolong', 'manfaatin', 'honestly', 'indonesia', 'yang', 'berasa', 'beda', 'saya', 'cuaca', 'astagah', 'jam', 'keringeta', 'kang', 'bohong', 'bohongin', 'bagaimana', 'dikecewain', 'tetap', 'cinta', 'dasar', 'aku', 'hee', 'ingat', 'minggu', 'kemarin', 'tidak', 'sengaja', 'habis', 'menginjak', 'kocheng', 'allah', 'kochengnya', 'yuhh', 'winterfell', 'sunyi', 'donnaruma', 'mengapa', 'sudah', 'diganti', 'tidak', 'balas', 'tidur', 'tidak', 'tidur', 'siang', 'tidur', 'siang', 'dibilang', 'tidur', 'mulu', 'tidak', 'tidur', 'siang', 'dikasianin', 'bucin', 'banget', 'tidak', 'opo', 'opo', 'iso', 'kecuali', 'siji', 'tangi', 'esuk', '1', 'tidak', 'dikasih', 'tidak', 'ngember', 'tidak', 'cerita', 'alesannya', 'simpl', 'tidak', 'baca', 'sepotong', 'berita']
```

6. Lemmalization

mengubah kata-kata dalam dataset tweet menjadi kata dasar dengan menggunakan modul *spacy* dan *sastrawi*

a.) Sastrawi

Untuk data yang sangat besar, tidak disarankan menggunakan sastrawi karena membutuhkan running time yang sangat besar. Sampai laporan ini dibuat, belum bisa lemmatize kata dari dataset dengan menggunakan sastrawi `stemmed_by_sastrawi = [stemmer.stem(w) for w in handled_slangword]`

In [51]:
```python
start_time_sastrawi = time.time()
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory # Import modul stemmer dari sastrawi

# create stemmer
factory = StemmerFactory() # Membuat factori untuk stemmer dengan memanggi objek StemmerFactory()
stemmer = factory.create_stemmer() # Membuat stemmer
# stemmed_by_sastrawi = [stemmer.stem(w) for w in handled_slangword] # Mengubah kata-kata menjadi kata dasar dengan menggunakan modul sastrawi
print(stemmer.stem("Melihat"))
end_time_sastrawi = time.time()

total_time_sastrawi = end_time_sastrawi - start_time_sastrawi

print("Hasil stem menggunakan modul Sastrawi adalah", total_time_sastrawi, "detik")
```

```
lihat
Hasil stem menggunakan modul Sastrawi adalah 0.4459097385406494 detik
```

b.) Spacy

In [52]:
```python
start_time_spacy = time.time()
from spacy.lang.id import Indonesian # Import modul spacy bahasa indonesia

nlp = Indonesian() # memanggi objek Indonesian() pada modul spacy
def stem_spacy(text): # Fungsi untuk mengubah kata-kata menjadi kata dasar
    for txt in nlp(text):
        t = txt.lemma_
    return t
stemmed_by_spacy = [stem_spacy(w) for w in handled_slangword]  # Mengubah kata-kata menjadi kata dasar dengan menggunakan modul spacy
end_time_spacy = time.time()

total_time_spacy = end_time_spacy - start_time_spacy

print("Hasil stem menggunakan modul Spacy adalah", total_time_spacy, "detik\n") # Hasil running terakhir adalah 67.62633967399597 detik
print(stemmed_by_spacy[:100])
```

```
Hasil stem menggunakan modul Spacy adalah 43.73394703865051 detik

['whahaha', 'benar', 'ledak', 'mercon', 'hancur', 'rumah', 'nanang', 'tewas', 'asih', 'bantu', 'kalideres', 'arah', 'batu', 'pendek', 'tangerang', 'tol', 'cawang', 'tmii', 'cibubur', 'bogor', 'ciawi', 'kesini', 'ku', 'tolong', 'manfaatin', 'honestly', 'indonesia', 'yang', 'rasa', 'beda', 'saya', 'cuaca', 'astagah', 'jam', 'keringeta', 'kang', 'bohong', 'bohongin', 'bagaimana', 'dikecewain', 'tetap', 'cinta', 'dasar', 'aku', 'hee', 'ingat', 'minggu', 'kemarin', 'tidak', 'sengaja', 'habis', 'injak', 'kocheng', 'allah', 'kochengnya', 'yuhh', 'winterfell', 'sunyi', 'donnaruma', 'apa', 'sudah', 'diganti', 'tidak', 'balas', 'tidur', 'tidak', 'tidur', 'siang', 'tidur', 'siang', 'dibilang', 'tidur', 'mulu', 'tidak', 'tidur', 'siang', 'dikasianin', 'bucin', 'banget', 'tidak', 'opo', 'opo', 'iso', 'kecuali', 'siji', 'tangi', 'esuk', '1', 'tidak', 'dikasih', 'tidak', 'ngember', 'tidak', 'cerita', 'alesannya', 'simpl', 'tidak', 'baca', 'sepotong', 'berita']
```

Menghitung frekuensi semua kata yang muncul

Kata yang **belum** melalui tahap 2-6 disimpan ke dalam variabel ***count_words_before***

Kata yang **sudah** melalui tahap 2-6 disimpan ke dalam variabel ***count_words_after***

In [53]:
```python
from collections import Counter

Counter1  = Counter(tokenized) # Menghitung frekuensi muncul semua kata
count_words_before = Counter1.items() # Menghitung kata
#print(count_words_before)

Counter2  = Counter(stemmed_by_spacy) # Menghitung frekuensi muncul semua kata
count_words_after = Counter2.items() # Menghitung kata (Tidak termasuk kata yang tidak bermakna)
#print(count_words_after)
```

Fungsi untuk membuat wordcloud

Fungsi tersebut untuk membuat wordcloud dan file yang berisi kata beserta frekuensinya

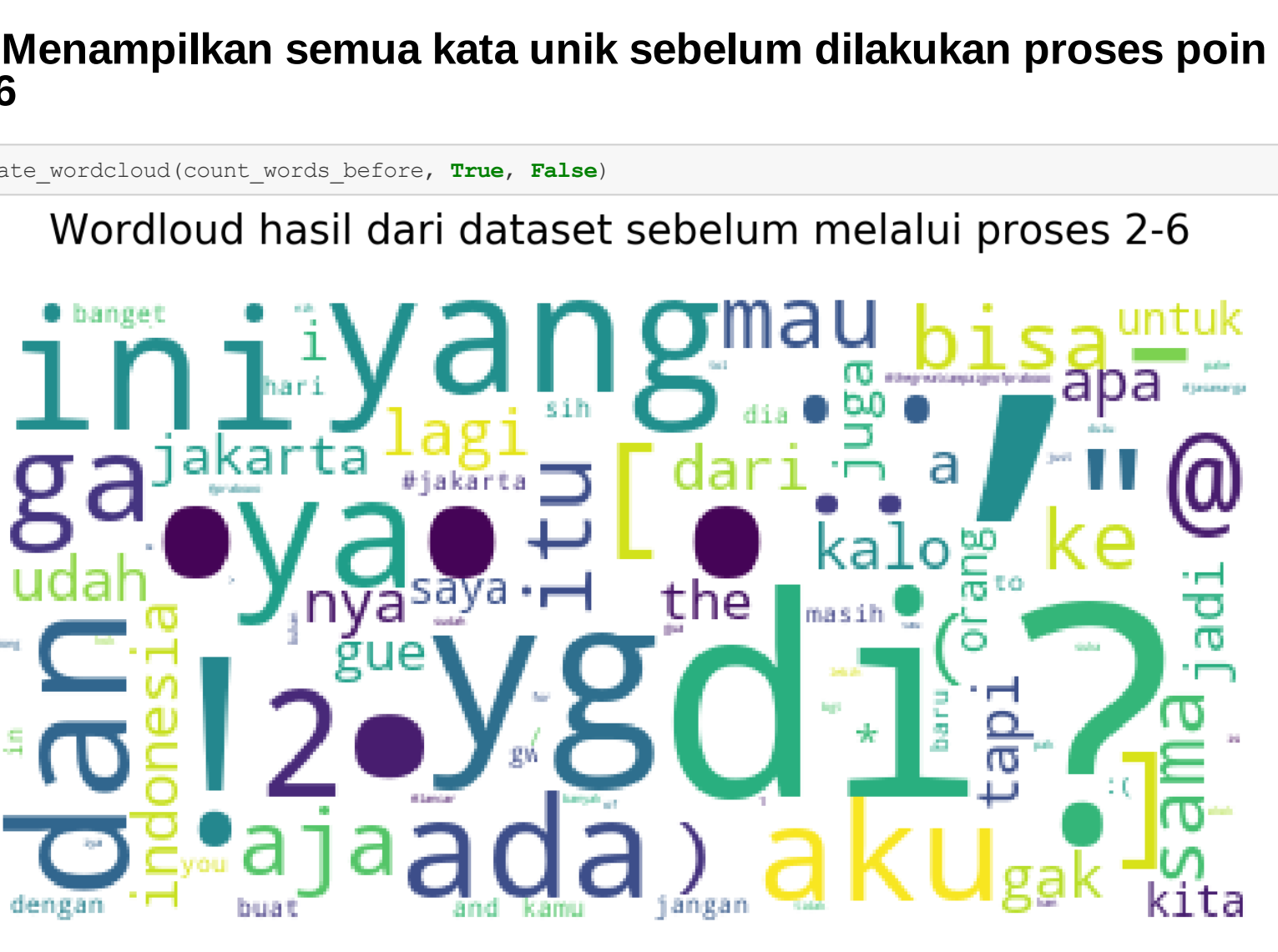
Dengan parameter:

- *count_words*, yaitu untuk menginput dictionary ***count_words_before*** dan ***count_words_after***
- *create_file*, yaitu boolean untuk membuat file atau tidak
- *after*, yaitu boolean untuk menentukan dia sudah melalui proses 2-6 atau belum

In [60]:
```python
import matplotlib.pyplot as plt
from wordcloud import WordCloud

def create_wordcloud(count_words, create_file = False, after = True):
    text = ""
    data_tweet = {}
    for element in count_words:
        text += str(element[0]) + " " + str(element[1]) + "\n"
        data_tweet[element[0]] = element[1]
    if after:
        nama_file = "sesudah"
    else:
        nama_file = "sebelum"
    if create_file:

        file = open("data/" + str(time.time()) + "_wordcloud_" + nama_file + "_filtering.txt", "w").write(text) # membuat file wordcloud

    wordcloud = WordCloud(background_color = 'white', max_words=100, contour_width=2)

    wordcloud.generate_from_frequencies(frequencies=data_tweet)
    plt.figure(figsize=(20,10))
    plt.title("Wordloud hasil dari dataset " + nama_file + " melalui proses 2-6\n", fontsize=40)
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.tight_layout(pad=0)
    plt.show()
```

7. Menampilkan semua kata unik sebelum dilakukan proses poin 2-6

In [61]:
```python
create_wordcloud(count_words_before, True, False)
```



8. Menampilkan semua kata unik sesudah dilakukan proses poin 2-6

In [62]:
```python
create_wordcloud(count_words_after, True)
```

In [63]:
```python
data_user = pd.DataFrame()

data_user['username'] = tweets['user_screen_name']
data_user['nama'] = tweets['user_name']
data_user['total_like'] = tweets['user_favourites_count']
data_user['quoted_total_retweet'] = tweets['quoted_status_retweet_count']
```

9. Menampilkan Top 10 user yang paling banyak di retweet

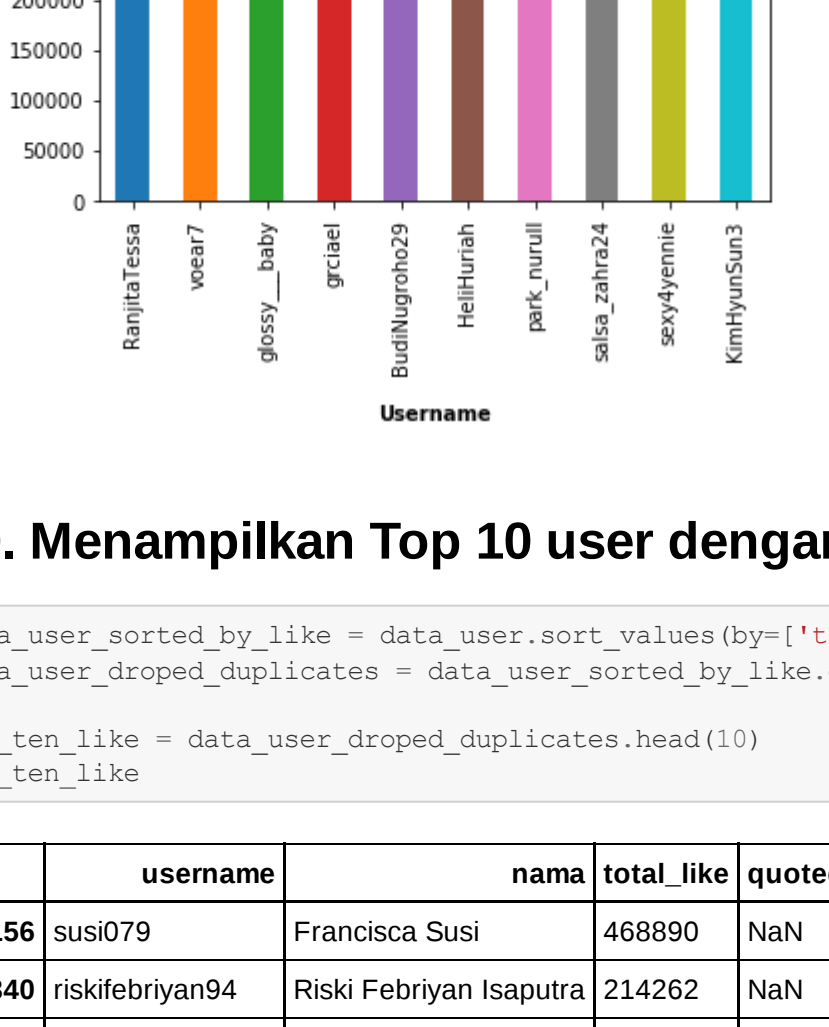
In [64]:
```python
data_user_sorted_by_retweet = data_user.sort_values(by=['quoted_total_retweet'], ascending=False)
data_user_droped_duplicates_retweet = data_user_sorted_by_retweet.drop_duplicates(subset='username', keep='first')
top_ten_retweet = data_user_droped_duplicates_retweet.head(10)
top_ten_retweet
```

Out[64]:

	username	nama	total_like	quoted_total_retweet
24264	RanjitaTessa	supergirl :3	891	411554.0
14511	voear7	Voeller Ari7onang	8	374673.0
20025	glossy___baby	زحر	16745	362080.0
20001	grciael	GRACE	2472	361667.0
5292	BudiNugroho29	Mendoan Addict.	217	279108.0
16113	HeliHuriah	thkim_id	158	278130.0
16049	park_nurull	Pipit N Fhitriyah	365	275919.0
68924	salsa_zahra24	'call me by your name'	1359	236910.0
28622	sexy4yennie	Yennie	15057	232927.0
26862	KimHyunSun3	김훈순□□	9424	202560.0

Membuat grafik 10 user retweet terbanyak

In [65]:
```python
plot = top_ten_retweet.plot(kind='bar', x='username', y='quoted_total_retweet')
plot.set_title('Top 10 user dengan retweet terbanyak', fontsize=15)
plot.set_xlabel('Username', weight='bold', labelpad=15)
plot.set_ylabel('Jumlah retweet', weight='bold', labelpad=15)

plot.tick_params(axis='x', pad=5)
```



10. Menampilkan Top 10 user dengan likes paling banyak

In [66]:
```python
data_user_sorted_by_like = data_user.sort_values(by=['total_like'], ascending=False)
data_user_droped_duplicates = data_user_sorted_by_like.drop_duplicates(subset='username', keep='first')
top_ten_like = data_user_droped_duplicates.head(10)
top_ten_like
```

Out[66]:

	username	nama	total_like	quoted_total_retweet
55156	susi079	Francisca Susi	468890	NaN
24340	riskifebriyan94	Riski Febriyan Isaputra	214262	NaN
58606	weatherarena	weatherarena	188499	NaN
33678	trackoftear	amirah	124330	NaN
69579	nabilanabilan2	nabilanabilan	118008	NaN
53630	yuliarti22	Yuliarti22Hardjono	114416	NaN
22767	jessiej99142542	Irenlim□	113514	NaN
70342	ewin8923	ewin winarti	104290	NaN
34477	MuhammadDarry	Mohammad Darry	103947	NaN
52658	okta_nindya_22	fadillah oktanindya	98978	NaN

Membuat grafik 10 user like terbanyak

In [67]:
```python
plot = top_ten_like.plot(kind='bar', x='username', y='total_like')
plot.set_title('Top 10 user dengan likes terbanyak', fontsize=15)
plot.set_xlabel('Username', weight='bold', labelpad=15)
plot.set_ylabel('Jumlah like', weight='bold', labelpad=15)
plot.tick_params(axis='x', pad=5)
```