**WCD SEO Project Readme**

This document explains in detail regarding the Weclouddata SEO project.

Outlined are the necessary files, the jupyter notebook code structure, how to use it, and how to potentially improve this strategy in the future.

For any additional questions not covered in this document, please feel free to contact Luiz at <luizg.lima92@gmail.com>,James at <jamesdguo@gmail.com>.

1. **WCD SEO SERP**
2. **1\_SEO\_Query\_Serp notebook**

**1.The objective of this script is to collect:**

1 From bootcamp query list(7 queries, can be added more or changed keywords "bootcamp" to "courses"), get google serp results

2 Based on google serp results and ranking pages(urls), do web scraping and crawl data from these pages

3 Select some/part of important and useful columns from crawl data preparing for future analysis

#### 2. This script will perform the following set of steps:

1 Install advertools

2 Read bootcamp query(keywords) lists,7 queries

3 Use google custom search API(account,key, id) and advertools function adv.ser\_goog() to serp top 20 ranking pages for each query.

4 Get all urls(ranking pages) from serp result, use advertools function adv.crawl() to do web scraping(crawl web page content), output file is a Jason file(.jl)

5 Read Jason file to dataframe and save data to crawl data csv

6 Get some part/columns from crawl data for future seo analysis

#### 3. The input/output of this script will be csv files containing:

1. input csv: query\_bootcamp.csv:

columns:[query\_no,query,keyword]

2. output csv:

1. Googleapi\_query\_serp.csv

columns: ['gl','searchTerms', 'rank', 'title', 'snippet', 'displayLink', 'link', 'queryTime', 'totalResults']

2. jason file: query\_bootcamp\_crawl.jl

3. query\_bootcamp\_crawl\_20230224.csv

columns: ['url', 'title', 'meta\_desc', 'viewport', 'charset', 'h1', 'h2', 'h3', ... 'resp\_headers\_x-imarc-service', 'resp\_headers\_x-robots-tag'

4. query\_bootcamp\_crawl\_part\_20230224.csv

columns: ['url', 'title', 'meta\_desc', 'h1', 'h2', 'h3', 'h4', 'h5', 'h6', 'canonical', 'alt\_href', 'body\_text', 'img\_alt', 'img\_width', 'img\_sizes', 'img\_height', 'img\_srcset', 'img\_src', 'size', 'depth', 'status', 'links\_url', 'links\_text', 'links\_nofollow', 'nav\_links\_url', 'nav\_links\_text', 'nav\_links\_nofollow', 'header\_links\_url', 'header\_links\_text', 'header\_links\_nofollow']

Reference:

1. advertools:

<https://advertools.readthedocs.io/en/master/readme.html>

1. advertools serp:

<https://advertools.readthedocs.io/en/master/advertools.serp.html>

1. **2\_SEO\_Query\_Keywords\_Count notebook**

#### 1.The objective of this script

is to collect bootcamp ranking pages keywords count numbers in

url,title,snippet,meta,img\_atl,h1,h2,h3.and body\_text/content

#### 2. This script will perform the following set of steps:

1 Read google serp results csv: Googleapi\_query\_serp\_20230224.csv

2. Read bootcamp ranking pages part craw data: query\_bootcamp\_crawl\_part\_20230224.csv

3. Merge data with serp and crawl data on same urls, create a new dataset including title,meta\_desc,htags,body\_text(content): query\_rankingpage\_content\_20230224.csv

4. Count keywords number in url,title,snippet,meta,img\_atl,h1,h2,h3.and body\_text/content, and then add all these numbers to a total number: key\_count

5. Save data to query\_rankingpage\_keywords\_count\_20230224.csv

6. Save weclouddata ranking pages keywords count data: query\_rankingpage\_keywords\_count\_wcd\_20230224.csv

#### 3. The input/output of this script will be csv fileS containing:

1. input : Googleapi\_query\_serp\_20230224.csv

query\_bootcamp\_crawl\_part\_20230224.csv

1. output:

\*query\_rankingpage\_content\_20230224.csv :

columns: ['query\_no', 'query', 'keyword', 'rank', 'url', 'website', 'title', 'snippet', 'queryTime', 'totalResults', 'meta\_desc', 'h1', 'h2', 'h3', 'h4', 'h5', 'h6', 'alt\_href', 'img\_alt', 'img\_srcset', 'img\_src', 'depth']

\*query\_rankingpage\_nocontent\_20230224.csv (not including body\_text easy for open with excel format, ,as body\_text is a very big size column)

\*query\_rankingpage\_keywords\_count\_20230224.csv

columns: ['query\_no','query', 'keyword', 'rank', 'website', 'url', 'title', 'snippet', 'meta\_desc','img\_alt', 'h1', 'h2', 'h3', 'body\_text', 'key\_count', 'key\_url', 'key\_title', 'key\_snippet','key\_meta','key\_img\_alt', 'key\_h1', 'key\_h2','key\_h3', 'key\_content','queryTime','totalResults']

\*query\_rankingpage\_keywords\_count\_wcd\_20230224.csv

columns: ['query\_no','query', 'keyword', 'rank', 'website', 'url', 'title', 'snippet', 'meta\_desc','img\_alt', 'h1', 'h2', 'h3', 'body\_text', 'key\_count', 'key\_url', 'key\_title', 'key\_snippet','key\_meta','key\_img\_alt', 'key\_h1', 'key\_h2','key\_h3', 'key\_content','queryTime','totalResults']

1. **3\_SEO\_Query\_Keywords\_Count\_Speed notebook**

#### 1. The objective of this script is to collect

ranking pages' keywords count data,desktop speed and mobile speed data for streamlit dashboard

#### 2.This script will perform the following set of steps:

1.Read bootcamp ranking pages keywords count csv: query\_rankingpage\_keywords\_count\_20221224.csv

2.Add desktop speed and mobile speed columns, and save data to csv: SEO\_Keywords\_query\_count\_speed.csv

3.Input desktop and mobile speed data by open csv with excel format and use Google tools to get speed data:

Analyze web pages speed by Google PageSpeed Insights <https://pagespeed.web.dev/>

#### 3. The input/output of this script will be a csv file containing:

1. input csv:

query\_rankingpage\_keywords\_count\_20221224.csv

columns: ['query\_no', 'query', 'keyword', 'rank', 'website', 'url', 'title', 'snippet', 'meta\_desc', 'img\_alt', 'h1', 'h2', 'h3', 'key\_count', 'key\_url', 'key\_title', 'key\_snippet', 'key\_meta', 'key\_img\_alt', 'key\_h1', 'key\_h2', 'key\_h3', 'key\_content', 'queryTime', 'totalResults']

1. output csv:

SEO\_Keywords\_query\_count\_speed.csv

columns: ['query\_no', 'query', 'keyword', 'rank', 'website', 'url', 'desktop speed', 'mobile speed', 'title', 'snippet', 'meta\_desc', 'img\_alt', 'h1', 'h2', 'h3', 'key\_count', 'key\_url', 'key\_title', 'key\_snippet', 'key\_meta', 'key\_img\_alt', 'key\_h1', 'key\_h2', 'key\_h3', 'key\_content', 'queryTime', 'totalResults']

**2.WCD SEO Sitemap Crawl**

**1. 1\_Sitemap\_Crawl\_wcd notebook**

#### 1. The objective of this script is to collect

Weclouddata all pages’ content data from it's website sitemap xml file

#### 2.This script will perform the following set of steps:[¶](http://localhost:8889/notebooks/WeCloudData/Clientproject/WCD%2520SEO_Document/WCD_SEO_Sitemap_Crawl_2/Sitemap_Crawl_wcd_1.ipynb#2.This-script-will-perform-the-following-set-of-steps:)

1.Use advertools function adv.sitemap\_to\_df() to read website’s sitemap xml file(<https://weclouddata.com/sitemap.xml>) and convert to a dataframe

2.Save data to csv: weclouddata\_url\_sitemap.csv

3.Get all urls from sitemap csv and use advertools function adv.crawl() to crawl pages’ data from urls. Output is a Jason file: wcd\_sitemap.jl

4.Read the Jason file to save data to csv: weclouddata\_crawl.csv

#### 3. The input/output of this script will be a csv file containing:

1.Input: <https://weclouddata.com/sitemap.xml>

2.Output

* weclouddata\_url\_sitemap.csv

Columns: ['url', 'sitemap']

* wcd\_sitemap.jl
* weclouddata\_crawl.csv

Columns: ['url', 'title', 'viewport', 'charset', 'h1', 'h5', 'canonical', 'alt\_href', … 'redirect\_times', 'redirect\_ttl', 'redirect\_urls', 'redirect\_reasons']

1. **2\_Sitemap\_Crawl\_competitors notebook**

#### 1. The objective of this script is to collect[¶](http://localhost:8889/notebooks/WeCloudData/Clientproject/WCD%2520SEO_Document/WCD_SEO_Sitemap_Crawl_2/Sitemap_Crawl_competitors_2.ipynb#1.----The-objective-of-this-script-is-to-collect)

Weclouddata's competitors' all pages’ content data from their sitemap xml file

#### 2.This script will perform the following set of steps:

1.Use advertools function adv.sitemap\_to\_df() to read website’s sitemap xml file(ex. <https://www.lighthouselabs.ca/sitemap.xml>) and convert to a dataframe

2.Save data to csv: websitename\_url\_sitemap.csv

3.Get all urls from sitemap csv and use advertools function adv.crawl() to crawl pages’ data from urls. Output is a Jason file: websitename\_sitemap.jl

4.Read the Jason file to save data to csv: websitename\_crawl.csv

#### 3. The input/output of this script will be a csv file containing:

1.Input: <https://websitename.com/sitemap.xml>

2.Output

\*websitename\_url\_sitemap.csv

Columns: ['url', 'sitemap']

\*websitename\_sitemap.jl

\*websitename\_crawl.csv

Columns: ['url', 'title', 'viewport', 'charset', 'h1', 'h5', 'canonical', 'alt\_href', … 'redirect\_times', 'redirect\_ttl', 'redirect\_urls', 'redirect\_reasons']

**3.** **WCD\_SEO\_Pyseoanalyzer**

**1.The objective of this script is to collect:**

1.1 Based on the 'SEO\_Keywords\_query\_count\_speed.csv' use Pyseoanalyzer for each url

1.2 Changing the format and colleting important information fount in the crawl from pyseoanalyzer

1.3 Select some/part of important and useful columns from crawl data preparing for future analysis

**2. This script will perform the following set of steps:**

2.1 Install pyseoanalyzer

2.2 Read 'SEO\_Keywords\_query\_count\_speed.csv'

2.3 Use Pyseoanalyzer to scrap url's previously collected and output a Jason file (.jl)

2.4 Preping the pyseoanalyzer results and transforming them into a pandas dataframe

2.5 EDA of dataframe and creation of useful columns

2.6 Prepare dataframes to be used in future with streamlit application

**3. The input/output of this script will be csv files containing:**

3.1 input csv: 'SEO\_Keywords\_query\_count\_speed.csv':

columns:['query', 'keyword', 'rank', 'website', 'url', 'desktop speed',

'mobile speed', 'title', 'snippet', 'meta\_desc', 'img\_alt', 'h1', 'h2',

'h3', 'key\_count', 'key\_url', 'key\_title', 'key\_snippet', 'key\_meta',

'key\_img\_alt', 'key\_h1', 'key\_h2', 'key\_h3', 'key\_content', 'queryTime',

'totalResults']

3.2 output csv:

1. 'complete\_SEO\_dataframe.csv'

columns: ['query', 'keyword', 'rank', 'website', 'url', 'desktop speed',

'mobile speed', 'title\_x', 'snippet', 'meta\_desc', 'img\_alt', 'h1',

'h2', 'h3', 'key\_count', 'key\_url', 'key\_title', 'key\_snippet',

'key\_meta', 'key\_img\_alt', 'key\_h1', 'key\_h2', 'key\_h3', 'key\_content',

'queryTime', 'totalResults', 'title\_y', 'description', 'word\_count',

'keywords', 'bigrams', 'trigrams', 'warnings', 'content\_hash',

'headings', 'additional\_info', 'count\_warning', 'all\_warnings',

'top5keywords', 'top5bigrams', 'top5trigrams']

2, warnings\_per\_page.csv

columns: ['query','rank','website','url','all\_warnings']

3, 'top5keywords\_per\_page.csv'

columns: ['query','rank','website','url','top5keywords']

4, 'top5bigrams\_per\_page.csv'

columns: ['query','rank','website','url','top5bigrams']

**4.WCD SEO Streamlit**

**1. 1\_Streamlit\_update\_top5\_keywords notebook**

#### 1. The objective of this script is to collect[¶](http://localhost:8889/notebooks/WeCloudData/Clientproject/WCD%2520SEO_Document/WCD_SEO_Streamlit_3/Streamlit_update_top5_keywords_1.ipynb#1.----The-objective-of-this-script-is-to-collect)

column top5\_keywords from complete\_top5\_keywords.csv and update it’s format to a new column top5\_count to show on Streamlit dashboard

#### 2. This script will perform the following set of steps:

1.Read csv: complete\_top5\_keywords.csv (from pySEO results)

2.Update column top5\_keywords’ format:

[[[92.'data'], [47, 'bootcamp'], [36, 'career'], [28, 'program'], [27, 'learn']]]

get a new column: top5\_count, it’s new format is: [ 'data' 92],[ 'bootcamp' 47],[ 'career' 36],[ 'program' 28],[ 'learn' 27]

3.Sort values and save data to csv: complete\_top5\_keywords\_count.csv

#### 3.The output of this script will be a csv file containing:

1.Input: complete\_top5\_keywords.csv

Columns:['query\_no', 'query', 'keyword', 'rank', 'url', 'top5\_keywords','website']

2.Output: complete\_top5\_keywords\_count.csv

Columns:['query\_no', 'query', 'keyword', 'rank', 'url', 'top5\_keywords', 'top5\_count', 'website']

**2. 2\_Streamlit\_update\_top5\_bigrams notebook**

#### 1. The objective of this script is to collect

column top5\_bigrams from complete\_top5\_bigrams.csv and update it’s format to a new column top5\_count to show on Streamlit dashboard

#### 2. This script will perform the following set of steps:

1.Read csv: complete\_top5\_bigrams.csv (from pySEO results)

2.Update column top5\_bigrams’ format:

[[('data science', 24), ('data analytics', 14), ('data scientist', 12), ('flex program', 11), ('science bootcamp', 10)]]

get a new column: top5\_count, it’s new format is: ['data science' 24],['data analytics' 14],['data scientist' 12],['flex program' 11],['science bootcamp' 10]

3.Sort values and save data to csv: complete\_top5\_bigrams\_count.csv

#### 3.The input/output of this script will be a csv file containing:

1.Input: complete\_top5\_bigrams.csv

Columns: ['query\_no', 'query', 'keyword', 'rank', 'url', 'top5\_bigrams', 'top5\_count', 'website']

2.Output: complete\_top5\_bigrams\_count.csv

Columns:['query\_no', 'query', 'keyword', 'rank', 'url', 'top5\_bigrams', 'top5\_count', 'website']

1. **3\_Streamlit\_update\_top5\_trigrams notebook**

#### 1. The objective of this script is to collect

Column top5\_trigrams from complete\_top5\_trigrams.csv

and update it’s format to a new column top5\_count to show on Streamlit dashboard

#### 2. This script will perform the following set of steps:

1.Read csv: complete\_top5\_trigrams.csv (from pySEO results)

2.Update column top5\_trigrams’ format:

[[('data science bootcamp', 10), ('learn how to', 5), ('data analytics bootcamp', 4), ('web development bootcamp', 4), ('weeks full time', 4)]]

get a new column: top5\_count, it’s new format is:

['data science bootcamp' 10],['learn how to' 5],['data analytics bootcamp' 4],['web development bootcamp' 4],['weeks full time' 4]

3.Sort values and save data to csv: complete\_top5\_trigrams\_count.csv

#### 3.The input/output of this script will be a csv file containing:

1.Input: complete\_top5\_trigrams.csv

Columns: ['query\_no', 'query', 'keyword', 'rank', 'url', 'top5\_trigrams', 'top5\_count', 'website']

2.Output: complete\_top5\_trigrams\_count.csv

Columns:['query\_no', 'query', 'keyword', 'rank', 'url', 'top5\_trigrams', 'top5\_count', 'website']

**4. 4\_Streamlit\_wcd\_web\_structure notebook**

#### 1.The objective of this script is to collect

home,main, and sub category from website urls,

sitemap\_sub category from website sitemap xml category

#### 2.This script will perform the following set of steps:

1.Read sitemap csv: weclouddata\_url\_sitemap.csv (from Sitemap\_Crawl\_wcd\_1.ipynb)

2.Split the column url to new columns: home,main, and sub category, split column sitemap to new column sitemap\_sub category

3.Save data to csv: weclouddata\_url\_sitemap\_sub.csv

4.Count urls group by main\_cat , and update some pages main\_cat=’ Non\_category’ which value\_counts==1

5.Sort values to csv: weclouddata\_url\_sitemap\_sub\_sort.csv

6.Recount urls group by main\_cat save data to csv: weclouddata\_url\_sitemap\_cat\_count.csv

#### 3.The input/output of this script will be a csv file containing:

1.input

weclouddata\_url\_sitemap.csv

columns: ['url', 'sitemap']

2.Output:

weclouddata\_url\_sitemap\_sub.csv:

columns: ['url', 'home\_cat', 'main\_cat', 'sub\_cat', 'sub\_cat2', 'sitemap', 'sitemap\_sub', 'maincat']

weclouddata\_url\_sitemap\_sub\_sort.csv:

columns: ['url', 'home\_cat', 'main\_cat', 'sub\_cat', 'sub\_cat2', 'sitemap', 'sitemap\_sub', 'maincat']

weclouddata\_url\_sitemap\_cat\_count.csv

columns: ['maincat', 'url\_count']

**5. 5\_Streamlit\_update\_warnings notebook**

#### 1. The objective of this script is to collect

column warnings from complete\_warnings.csv and update it’s format to new column warnings\_count(ratio/percentage) to show on Streamlit dashboard

#### 2.This script will perform the following set of steps:

1.Read csv: complete\_warnings.csv (from pySEO results)

2.Update column warnings format: [('Anchor missing title tag', 48), ('Image missing alt tag', 4)]

get a new column: warnings\_count, it’s new format is:

['Anchor missing title tag' 48 , 92.0%] , ['Image missing alt tag' 4 , 8.0%]

3.Save data to csv: complete\_warnings\_total.csv

#### 3.The input/output of this script will be a csv file containing:[¶](http://localhost:8889/notebooks/WeCloudData/Clientproject/WCD%2520SEO_Document/WCD_SEO_Streamlit_3/Streamlit_update_warnings_5.ipynb#3.The-input/output-of-this-script-will-be-a-csv-file-containing:)

1.Input: complete\_warnings.csv

Columns: ['query\_no', 'query', 'keyword', 'rank', 'url', 'warnings', 'website']

2.Output: complete\_warnings\_total.csv

Columns: ['query\_no', 'query', 'keyword', 'rank', 'url', 'warnings', 'warnings\_count', 'warnings\_total', 'website']

1. **6\_Streamlit\_broken\_links notebook**

#### 1.The objective of this script is to collect

and count broken links from top 10 data science bootcamps.

#### 2. This script will perform the following set of steps:

1.Read website crawl data, for example: weclouddata\_crawl.csv, lighthouselab\_crawl.csv…. coursera\_crawl.csv

2.Find urls/links which status>=400

3.Count broken links to csv: broken\_links\_count.csv

4.Save wcd broken links to csv: broken\_links\_wcd.csv

#### 3.The input/output of this script will be csv file containing:

1.Input: website crawl data, like “weclouddata\_crawl.csv”,” lighthouslab\_crawl.csv”

2.Output:

broken\_links\_wcd.csv:

Columns: website,url,title,status

broken\_links\_count.csv:

Columns: rank,website,broken\_link,link\_total,percent

**7. 7\_Streamlit\_wcd\_seo.py**

#### 1.The objective of this script is to create a dashboard for WCD SEO project results.

#### 2. This script will perform the following set of steps:

1.Display data table and charts of weclouddata and competitors desktop speed and mobile speed for queries and bootcamps

2.Show top 5 keyword(keywords , bigrams, trigrams) count by rank of weclouddata and competitors’ bootcamp ranking pages

3.Show weclouddata website sitemap and urls structure(top 10 main category, value counts), Non category pages, Blogs Non\_category sample pages

4.Show weclouddata and competitors’ warnings of bootcamp pages

5.Show broken links and their counts of weclouddata and competitors’ website

**# Notes: Run streamlit:**

1.Download streamlit:

https://docs.streamlit.io/library/get-started/installation#install-streamlit-on-windows

#pip install streamlit

2.Run streamlit:

1. Open a terminal in your environment: Anaconda Prompt ,

2. In the terminal that appears, use Streamlit as usual: streamlit run myfile.py

**5.WCD SEO Neo4j**

**1. 1\_Neo4j\_WCD\_bootcamp\_url\_bodytext notebook**

#### 1.The objective of this script is to collect

Weclouddata home page and 12 bootcamp(full time&part time) urls’ body\_text and ranking information.

#### 2.This script will perform the following set of steps:

1.Read ranking pages’ data csv: SEO\_Keywords\_query\_count\_speed.csv

WCD web pages data: weclouddata\_crawl\_20230130.csv (from Sitemap\_Crawl\_wcd\_1.ipynb)

Selected 13 urls including home page and bootcamp urls(full time, part time) :weclouddata\_graph\_url\_13.csv

Merge urls list,crawl data(page body\_text) , and wcd ranking page(rank,query): data from weclouddata\_graph\_url\_13.csv+weclouddata\_crawl\_20230130.csv+ SEO\_Keywords\_query\_count\_speed.csv

2.Save data to csv: weclouddata\_key\_text.csv

#### 3.The input/output of this script will be csv file containing:

1.Input: weclouddata\_crawl\_20230130.csv+ weclouddata\_graph\_url\_13.csv+ SEO\_Keywords\_query\_count\_speed.csv

2.Output: weclouddata\_key\_text.csv

columns: ['url', 'body\_text', 'links\_url', 'redirect\_urls', 'query','rank']

**2. 2\_Neo4j\_WCD\_bootcamp\_url\_content notebook**

#### 1.The objective of this script is to collect

Weclouddata home page and 12 bootcamp(full time and part time) urls’ paragraph text

which is same as body\_text, but it’s format is different .

#### 2.This script will perform the following set of steps:

1.Read wcd 13 pages’ body\_text data csv:

weclouddata\_key\_text.csv (from Notebook: Neo4j\_WCD\_bootcamp\_url\_bodytext)

2.Scrape paragraph content from pages by BeautifulSoup

3.Merge body\_text and paragraph content and save data to a new dataset: weclouddata\_key\_text\_content.csv

#### 3.The output of this script will be a csv file containing:

1.Input: weclouddata\_key\_text.csv

columns: ['url', 'body\_text', 'links\_url', 'redirect\_urls', 'query','rank']

2.Output: weclouddata\_key\_text\_content.csv

columns: ['url', 'body\_text',’content’, 'links\_url', 'redirect\_urls', 'query','rank']

**3. 3\_Neo4j\_WCD\_Graph\_url\_keywords notebook**

### **1.The objective of this script is to collect**

top 10 keywords from Weclouddata home page and 12 bootcamp(full time and part time) urls’ paragraph text.

#### 2. This script will perform the following set of steps:

1.Define an extract keywords function that can extract keywords and construct them back from tokens for a text. Use model: "yanekyuk/bert-uncased-keyword-extractor"

2.Define a function get\_keywords that can extract top 10 keywords from a text.

3.Read wcd bootcamp pages' text data: weclouddata\_key\_text\_content.csv (from Notebook Neo4j\_WCD\_bootcamp\_url\_content(paragraph)\_3)

4.Run function get\_keywords() to get top 10 keywords for each url’s page content(paragraph text).

5.Save data to csv: weclouddata\_data\_knowledgegraph.csv

6.Create a dataset for neo4j graph: weclouddata\_graph\_url\_key.csv

7.Use Neo4j desktop and cyper to draw a knowledge graph of wcd bootcamp urls and their top 10 keywords in neo4j

#### 3.The input/output of this script will be csv file and weclouddata knowledge graph containing:

1.Input: weclouddata\_key\_text\_content.csv

columns: ['url', 'body\_text',’content’, 'links\_url', 'redirect\_urls', 'query','rank']

2.Output: weclouddata\_graph\_url\_key.csv

columns: ['link', 'url', 'keywords', 'K’]

3. weclouddata knowledge graph :

wcd\_knowledge\_graph\_link\_keywords\_3.jpg,

wcd\_knowledge\_graph\_link\_keywords\_13.jpg

**# Notes: Run Neo4j:**

1.Install Neo4j desktop:

<https://neo4j.com/download-neo4j-now/?gclid=Cj0KCQiAi8KfBhCuARIsADp-A56vosSYSoxu4RCTjaQUPA20SWT-GErdfkltOlCZtfnXiPo7PmFK8JYaAmmEEALw_wcB>

2. Run Neo4j desktop:

create dbms: like "wcd";

set user='neo4j', password(default='neo4j');

create database;

use database;

open neo4j browser;

run the sql in cyper

Reference:

1.Neo4j Knowledge graph:

<https://towardsdatascience.com/analyze-your-website-with-nlp-and-knowledge-graphs-88e291f6cbf4#a18c-73acbf822d9>

**Analyze Your Website with NLP and Knowledge Graphs**: Combine various NLP techniques to construct a knowledge graph representing your website

All the code of this blog post is [available on GitHub](https://github.com/tomasonjo/blogs/tree/master/neo4jdocs).