Common Crawl: *Database Systems Ranking*

Anastasis Bazinis
Vrije Universiteit
Amsterdam
a.bazinis@student.vu.nl

Salmane Dazine
Vrije Universiteit
Amsterdam
s.dazine@student.vu.nl

Effrosyni Stergiopoulou
Vrije Universiteit
Amsterdam
e.stergiopoulou2@student.vu.nl

ABSTRACT

This project solves a real-world large-scale data engineering project. It consists of analyzing the Common Crawl corpus, inspecting database system mentions in the plain texts and the URLs of the webpages provided, and measuring the evolution of the popularity of database systems accordingly. To solve this challenge on a large set of data, big data computing services like Amazon Web Services, Spark in Python, and the Databricks platform were employed.

1. MOTIVATION

The Common Crawl corpus provides a large pool of information and hidden insights. Change over time represents an essential factor that contributes to creating useful visualizations. We strive to make this project valuable by leveraging the Common Crawl data in order to reveal and visualize hidden insights. Tracking and documenting the evolution of database system mentions over the last decade would facilitate correlating this development to potential factors and repercussions. It could also open the field for readers to make sense of the future popularity evolution of database systems. Not only could our sought-after conclusions provide readers with the rate of growth or decline of the studied database systems, but could also guide their choice of technology based on the past and current sentiment of the online public regarding it.

2. DESCRIPTION

This project consists of analyzing samples of websites extracted from the Common Crawl corpus by examining mentions of database systems. Using these mentions we created a timeline of the rankings of some of the most popular database systems. Due to the size of web crawls, much consideration must be given to how to sample and which pages to obtain. Petabytes of information gathered over 12 years of web crawling make up the Common Crawl[1] corpus. The corpus includes text extracts, metadata extracts, and raw data from web pages. The Common Crawl archives are provided by publicly Amazon Web Services inside the S3 Bucket.

The information utilized was provided by Common Crawl in Web ARChive (WARC) format. The multi-billion web page archives from Common Crawl, which can be hundreds of terabytes in size, are provided in WARC format. The WARC format holds metadata about the crawl process, the HTTP response from the websites it contacts, as well as information about how that information was requested (WARC-Type: response, request, metadata).

The Common Crawl dataset also offers WET files that only carry the extracted plaintext. This textual information is saved in the WET format in a very straightforward manner making it easier to process for tasks that require solely the textual corpus. The length of the plaintext data and other information are contained in the header, which is followed by the plaintext data. In this study, we showed two ways to investigate the popularity of database systems. For the first approach, we chose the WET files because they contain the textual data from each webpage from which we will get the occurrences of the various database systems throughout the years. WET files are significantly smaller in size than WARC. For example, when it comes to June/July crawl archive there is a 89.3% reduction of the size from WARC to WET format[2].

For the second approach, we utilized the Index to WARC Files and URLs in a columnar format[3]. The columnar format (Apache Parquet) allows us to query or process the index to WARC files more effectively, saving time and computational resources. The table format makes it possible to run fast SQL queries on columnar data using SparkSQL. These files contain only the index to the WARC files and not the plain text inside the website. For our solution, we used only part of the URL to identify relevant mentions of database systems.

The Amazon S3 distributed file system allows for immediate usage of the corpus without having to download it locally.

3. RESEARCH QUESTION

3.1 Scope

For the scope of our project we decided to focus on 8 database systems namely: Oracle, MySQL, Microsoft SQL Server, PostgreSQL, MongoDB, SparkSQL, ClickHouse and DuckDB. The first five database systems are the most popular for 2022[4]. We also wanted to apply our method on less popular database systems released after 2013 (SparkSQL, ClickHouse and DuckDB) in order to verify how close to

[1]https://commoncrawl.org/

[2]https://commoncrawl.org/2022/07/june-july-2022-crawl-archive-now-available/

[3]https://commoncrawl.org/2018/03/index-to-warc-files-and-urls-in-columnar-format/

[4]https://db-engines.com/en/ranking

their actual launch we can detect mentions of them in the web archives.

3.2 Goals

This study aims to answer the following research questions:

- How to efficiently extract database systems mentions given the large dataset provided by the Common Crawl platform?
- How to select representative samples containing comparable numbers of database system mentions per year?
- How did the popularity of database systems change from 2013 until 2022?

4. DATA INVESTIGATION

The initial data investigation aims to investigate the size, format, and distribution of this data to identify our strategy for cleaning and organizing it efficiently during further stages of the project.

4.1 Data Size & Format

Prior to investigating the data that lives on Amazon S3, we investigate the Common Crawl official website[5], through which we observe the nature of the data that has been evolving during the past years. The Common Crawl dataset contains incredibly large amounts of scraped web pages from 2008 until 2022. Starting by the period from 2008 until 2012, when the website only offered ARC-type data, and moving to the summer of 2013, when this traditional version has been extended to the WARC format, allowing to add of more metadata. The year 2013 also marked the first appearance of WET files that store the plain text version of the WARC data. Given that our analysis aims to recognize database mentions in the web pages, our focus will shift to analyzing the textual data more than the metadata. Therefore, the WET files published from 2013 until 2022 will be the main focus of our investigation.

After examining the data residing in Amazon S3, and accessible through the Databricks platform, we observe that the data is classified into a large number of folders based on the month and the year of extraction. The needed WET files can be found with the extension "wet.gz" under the "segments" folder under the folder of each month. Up until this date, we have read 4,940,667 WET files, in total, from all years. The distribution of WET files for each year is laid out in Table 2.

Each one of these files consists of a slightly different number of web pages, but it usually roams around 50,000. These web pages are sorted based on their unique URIs in ascending order from A to Z. Figure 1 demonstrates two data frames of the URIs extracted from the two first files from 2013. In the two extracted files, all URIs were unique. This might not be representative of all the other files, therefore we aim to check thoroughly for duplicates of the same date in further stages.

After printing the first WET file, we discover that each web page recorded follows the following structure: The file displays first the metadata, as can be seen in Figure 11, about WARC-Type, WARC-Target-URI, WARC-Date, WARC-Record-ID, WARC-Refers-To, WARC-Block-Digest, Content-Type, and Content-Length. After that, it immediately displays the plain text extracted from the web page.

However, it's been observed that not all web pages in the WET files contain textual data, some only output the aforementioned metadata, and don't extract any text afterward. That can be due to the fact that some web pages can contain other forms of information such as images, graphs, or videos. This requires further cleaning to get rid of redundant data that won't take part in our database mention analysis.

[5]https://commoncrawl.org/the-data/get-started/

2013	76,763
2014	401,585
2015	324,432
2016	302,520
2017	847,927
2018	800,000
2019	688,000
2020	571,600
2021	615,840
2022	312,000

Table 1: Distribution of WET files per year

4.2 Tools Used

The initial data investigation has been conducted using a Python-based notebook in the Databricks platform. Several libraries and tools have been attempted and compared to efficiently read and extract the Common Crawl data from the Amazon S3 storage.

After encountering difficulties installing and importing the warc library, we decided to replace it with the warcio library to extract the URIs and HTML pages from the WARC files. On top of that, we ran built-in Beautifulsoup functions to extract the plaintext from the HTML files. We have also employed the function os.walk from the os library to identify all the WARC files stored in the root and subdirectories of Common Crawl, by using file.startswith and endswith to detect the needed extensions. This method took 3.62 minutes to extract the plain texts from one WARC file. After successfully importing the warc library, we made our second attempt, where we directly processed the WET files without needing Beautifulsoup to extract the texts, and we used glob built-in functions to walk over the required directories. Our running timing has improved to 2.69 minutes for extracting the texts from one WET file. Finally, we optimized our method by extracting only URIs instead of texts, and we managed to read the web pages of the file in under 1.52 minutes. In order, to extract the needed database system mentions, it is not necessary to store the plain-texts or URI data, therefore we only count the database system mentions from the web pages of each file. Applying this reduces the running time to 7 seconds for each WET file.

4.3 Data Distributions

In order to understand more about the distributions of the data, we selected two small samples. One from the first release of 2013, in Figure 3 and the other from the first release of 2022, in Figure 4.

These two samples consist of the very first WET file stored in their corresponding releases. Although the selected sam-

	uri
0	http://%20jwashington@ap.org/Content/Press-Rel...
1	http://004eeb5.netsolhost.com/stephensilver.htm
2	http://02d2262.netsolhost.com/roguesgallery.html
3	http://0pointer.de/photos/?galerie=Reichstag%2...
4	http://0pointer.de/photos/?gallery=Avoriaz%202...
...	...
53801	http://zurb.com/article/1180/how-to-use-founda...
53802	http://zurich.ai.mit.edu/hypermail/thinkpad/20...
53803	http://zurich.ai.mit.edu/hypermail/thinkpad/20...
53804	http://zwischenwelt.org/~hagish/irisdocs/trunk...
53805	http://zztube.com/pornstars/336/ava_lauren/1.html

53806 rows × 1 columns

	uri
0	http://06-17.tumblr.com/page/3
1	http://08denovembrode2011.tumblr.com/
2	http://0hmy0x.tumblr.com/post/26634553706
3	http://0pointer.de/photos/?gallery=Istanbul%20...
4	http://1000bulbs.com/product/5655/DEC-100138.html
...	...
54390	http://zsazsabellagio.blogspot.com/2013/01/nee...
54391	http://zsido.com/konyv/49/823/Melysegbol_kialt...
54392	http://zub-kapusty.tumblr.com/tagged/food
54393	http://zuihitsu.org/journal-paper-of-the-day-4
54394	http://zztop.com/users/1wBJKdqw/blogs/

54395 rows × 1 columns

Figure 1: URIs from the two first WET files from 2013

ples might not be representative distribution-wise, they allowed us to sense the data and know what to expect in the further stages i.e. What data size do we expect to need from the Common Crawl dataset? How much does that represent percentage-wise from the whole corpus? What does the latency of the platform dictate for us to work with? How can we adjust our data access patterns in order to optimize the running time? What is the absolute minimum amount of data we can reduce our dataset to and still get meaningful insights?

As stated before, the URIs of the extracted web pages were unique for each file. This shifts our focus to the domains and suffixes of the URIs. Counting the repetitions of the unique values of the suffixes yields the following results in Figure 5:

It is not surprising that .com is the most popular suffix, followed by .org in both samples. We can also notice top-level domains that belong to popular countries such as Germany (de), France (fr), Italy (it), and Netherlands (nl).

```
WARC/1.0\r\nWARC-Type: conversion\r\nWARC-Target-URI:
http://%20jwashington@ap.org/Content/Press-Release/2012/How-AP-reported-in-all
-formats-from-tornado-stricken-regions\r\nWARC-Date:
2013-05-18T05:48:54Z\r\nWARC-Record-ID:
<urn:uuid:4fd7be4a-d884-4c99-aff4-8a3707f7f82c>\r\nWARC-Refers-To:
<urn:uuid:d66bc6fe-8477-4adf-b430-f6a558ccc8ff>\r\nWARC-Block-Digest:
sha1:ZVFBFW6W3XHYJJVZG66R35V2HHM5LADS\r\nContent-Type:
text/plain\r\nContent-Length: 5284\r\n\r\n
```

Figure 2: Structure of the first web page of the first WET file from 2013

	uri	domain	suffix
0	http://%20jwashington@ap.org/Content/Press-Rel...	ap	org
1	http://004eeb5.netsolhost.com/stephensilver.htm	netsolhost	com
2	http://02d2262.netsolhost.com/roguesgallery.html	netsolhost	com
3	http://0pointer.de/photos/?galerie=Reichstag%2...	0pointer	de
4	http://0pointer.de/photos/?gallery=Avoriaz%202...	0pointer	de
...	...	...	...
53801	http://zurb.com/article/1180/how-to-use-founda...	zurb	com
53802	http://zurich.ai.mit.edu/hypermail/thinkpad/20...	mit	edu
53803	http://zurich.ai.mit.edu/hypermail/thinkpad/20...	mit	edu
53804	http://zwischenwelt.org/~hagish/irisdocs/trunk...	zwischenwelt	org
53805	http://zztube.com/pornstars/336/ava_lauren/1.html	zztube	com

53806 rows × 3 columns

Figure 3: Sample from the first release in 2013

This speaks of the variety of data and potentially the different languages that we would need to deal with in our analysis.

When it comes to domains as shown in Figure 6, we can notice that there is not a significant discrepancy between the two samples, as Blogspot takes the lead as the most extracted website in both rankings, while popular websites like Wordpress, Wikipedia, Yahoo, and Google are also ranked highly. If we're aiming to significantly improve our running time, it would not be a successful strategy to reduce the dataset to focus on these websites, since they are extracted heavily and are representing a large proportion of the dataset.

Our best bet to reduce our dataset would be to find web pages related directly to the topic of our analysis. Inspecting the plain texts for any explicit mentions of the word "database" yields the following results in Figure 7.

In 2013, it's been found that 1,383 out of the total 53,806 entries had a "database" mention, representing 2.57%. While 613 out of the 42,127 entries in 2022 had a "database" mention, representing 1.46%. Although it may seem that the extracted mentions are more relevant to our investigation, mentions of databases can also be employed in non-technical contexts. A simple Wikipedia search[6] of the available senses of the named entity "database" is found to be employed in fields of academia, biology, chemistry, and neuroscience. In order to reduce our search results to the relevant entries to our project, we try searching for mentions of a specific technical database, e.g. MySQL. Figure 8 demonstrates the results of our query. 68 out of the 53,806 entries in 2013 had a "mysql" mention, representing 0.13%. While 47 out of the

[6]https://en.wikipedia.org/wiki/Lists_of_databases

	uri	domain	suffix
0	http://0575ls.cn/news-52300.htm	0575ls	cn
1	http://0575ls.cn/pan/list?j=4&qu=0&lpmj=0	0575ls	cn
2	http://08.od.ua/bytovaya_tehnika/stiralnye_mas...	08	od.ua
3	http://100lat.pck.pl/fakty/pawel-sapieha/	pck	pl
4	http://121.100.18.218/tampil_ipk_ikm/ipk_ikm.php	121.100.18.218	
...	...	...	...
42122	https://zwielicht-bremen.de/events/tags/barbar...	zwielicht-bremen	de
42123	https://zxmarket.ro/tag/ignifugare-lemn/	zxmarket	ro
42124	https://zychar.pl/sklep/czesci-do-maszyn-rolni...	zychar	pl
42125	https://zyukuzyo.com/archives/2990	zyukuzyo	com
42126	https://zzsh111.com/a/Hu4SGefM/	zzsh111	com

42127 rows × 3 columns

Figure 4: Sample from the first release in 2022

42,127 entries in 2022 had a "mysql" mention, representing 0.11%.

The achieved percentages already induce progress in reducing our dataset, therefore, our chosen methodology will consist of filtering the web pages by looking for the mention of 'database' first, then applying another filtering layer by looking for the mention of the specific database system.

5. METHODOLOGY

5.1 Technologies

Other studies that attempted to extract data from the Common Crawl corpus were using languages like Spark, Python, R, and Java. We decided to implement our solution in two different ways in two languages. We run our code in Databricks notebooks.

In the first approach, we gathered data from the WET files we had at our disposal. We implemented that in Python. The libraries that helped were the glob2 to get all the filenames in the directory of the WET files. Also, to extract the text we needed the warcio library which contains the payload.read() function to extract the text from web pages.

In the second approach, we only considered the WARC files. We were provided with the possibility to read parquet files which are rather faster to access compared to .warc.gz. Spark and SparkSQL proved to be excellent tools for this work. We were able to get data from every WARC file without sampling. We read the files given with the spark.read() function and then we queried with the aid of SparkSQL.

5.2 Data Engineering Pipeline

Figure 9 shows the steps that led to the final results. The code was written in a Databricks notebook. The data was stored in a folder inside the DBFS folders. It contained data from Amazon S3 which is where the common crawl data is stored. This data was loaded and processed with two different approaches and the final outcome consisted of two charts that demonstrated the popularity of database systems throughout the last decade.

5.3 Methods

The data that Common Crawl provides is massive and is therefore difficult to access. Thus, the decision was to take a

unique_suffixes	counts
com	37957
org	5011
net	1877
edu	1629
co.uk	1032
de	869
gov	714
ca	315
com.au	274
fr	253
it	234
nl	176
se	164
es	143
tv	136
ac.uk	134
ch	129
info	128
eu	123
dk	108

unique_suffixes	counts
com	18463
org	2515
ru	1833
de	1825
net	1486
co.uk	756
fr	751
it	711
nl	644
pl	635
edu	582
com.br	462
jp	462
cz	458
es	400
ca	350
com.au	335
info	284
eu	269
be	255

Figure 5: Top 20 popular suffixes from the 2013 sample (left) and the 2022 sample (right)

significantly smaller sample of 100 WET files per year compared to the 50,000 that exist. It is suited best to gather data from every year from the same time period to understand the evolution of the popularity better. During our investigation of the WET files, we noticed that we didn't have data from every week of every year so we decided to find the common weeks of each year and take data from there so that we would have consistency in the results. We ended up taking data from approximately the 25th week of each year (sometimes the 22nd or the 26th depending on the year). At first, we were looking for texts on websites that contained the word "database" and then some of the most popular database systems like "mysql", "oracle", "sparksql" etc. Furthermore, we counted how many webpages we were going through whether they contained a keyword or not. This method took about 15 minutes per year to finish running. The popularity of each database system per year is shown in Figure 10.

So we decided to double the sample and take the first 200 files, and along with the aid of the private cluster provided to us by our professor to run our code, we were able to get more accurate results in approximately the same run-time. Thus, we now had a sample of about 850 million web pages extracted from the same period of each year which proved to be a more representative sample.

We also discovered that the parquet files required significantly less time to be queried, so we opted to read all of them and obtain the data per year. We searched for the same mentions as in the previous methods. The only difference being that the search took place in the URL of the webpage rather than in its content. More specifically, we isolated the url_surtkey from each webpage and we queried each one for the database systems. We then calculated the popularity of each database system per year and semester plotted in Figures 12, 13.

6. RESULTS

unique_domains	counts
blogspot	935
tumblr	726
wordpress	295
patch	202
wikipedia	197
citysearch	196
stackexchange	194
tripadvisor	171
photobucket	151
yahoo	115
mlb	96
go	95
google	93
ebay	89
deviantart	73
oclc	66
rivals	66
state	63
wikia	61
typepad	58

unique_domains	counts
blogspot	239
wordpress	212
wikipedia	117
google	97
europa	41
yahoo	32
tapuz	31
fandom	30
nimo	29
herokuapp	28
hatenablog	28
amazon	27
pinterest	27
stackexchange	26
netlify	26
home	25
nih	24
airbnb	23
photoshelter	23
altervista	21

Figure 6: Top 20 popular domains from the 2013 sample (left) and the 2022 sample (right)

In this section, we are going to navigate to our main findings corresponding to the insight obtained on the popularity of database systems after applying the two data pipeline approaches. In order to tackle the inconsistencies of the number of web pages that we used per year, we calculated the percentage of the mentions over the total number of web pages for each year. This way, the ranking of the web pages will not be dependent on the number of web pages per year, but on a percentage of the overall content. Also, we used a logarithmic scale for displaying the data because there were large differences between the percentages of each database system and the charts were not interpretable. Furthermore, the use of a logarithmic scale allowed us to compare our results with the existing solution from the DB-Engines site.

6.1 Visualization

For the visualization, we created a static webpage using HTML, CSS for the styling, and JavaScript for the interactive components of the web page. Since we want a single-page web application, we decided to structure our visualization using tabs that would allow us to display different tables and charts. In the first tab, we displayed the ranking in the form of a table. In the second tab, we plotted our results in the form of a line graph from 2013 to 2022. For our plots, we used some of the JavaScript chart templates given by canvasjs.com[7]. In these charts, we manually inserted the values for each of the database systems in the form of a percentage and we set at axis Y, `logarithmic:true` in order to apply a logarithmic scale on the values.

6.2 WET files

[7]https://canvasjs.com/javascript-charts/

	uri	domain	suffix
0	http://2012.phillytechweek.com/	phillytechweek	com
1	http://9rank.com/hosting-location/Bangkok,Krun...	9rank	com
2	http://aangirfan.blogspot.ca/2008/12/gambia-an...	blogspot	ca
3	http://abcphp.com/login.php?return=%2Fupcoming...	abcphp	com
4	http://advsites.net/modecelebration.com	advsites	net
...	...	...	...
1378	http://www.zymic.com/forum/index.php?showuser=...	zymic	com
1379	http://www1.cse.wustl.edu/~jain/cse571-09/ftp/...	wustl	edu
1380	http://www2.gsu.edu/~wwwhiv/oct-nov.html	gsu	edu
1381	http://www3.ridsport.se/Distriktssajter/Mittsv...	ridsport	se
1382	http://xboxmovies.teamxbox.com/xbox-360/10370/...	teamxbox	com

1383 rows × 3 columns

	uri	domain	suffix
0	http://61-64-230-168-adsl-tpe.dynamic.so-net.n...	so-net	net.tw
1	http://b0mfnc.aguwe.com/	aguwe	com
2	http://blog.ganyongmeng.com/?cat=30	ganyongmeng	com
3	http://cikewudi.com/?tag=SEO%E6%96%87%E7%AB%A0	cikewudi	com
4	http://geromescuttle.awardspace.info/rss.php/a...	awardspace	info
...	...	...	...
608	https://xn--flyttstdningstockholm-c2b.se/st%C3...	xn--flyttstdningstockholm-c2b	se
609	https://xtremehostingservices.com/location/ind...	xtremehostingservices	com
610	https://ywamships.org/medical-ship/volunteer/	ywamships	org
611	https://zaraexpo.com/canadian-online-casino/sl...	zaraexpo	com
612	https://zklpawliszyn.pl/shotcrete/21_8/1400/	zklpawliszyn	pl

613 rows × 3 columns

Figure 7: Mentions of "database" from the 2013 sample (up) and the 2022 sample (bottom)

We decided to use the plain text data from the WET files provided by common crawl. By looking at the chart in Figure 11 we observe that the leading database system in terms of popularity is Oracle, with an almost identical evolution with MySQL followed by Spark. PostgreSQL appears to be the fourth in terms of popularity, followed by Microsoft SQL Server. Finally, ClickHouse debuted in 2017 with a highly fluctuating route, and DuckDB made its appearance in 2021 with an increasing trend.

Regarding the comparison of our method using the WET files and the DB-Engine's ranking(Figure 14) our results deviate significantly.

Although Oracle and MySQL are almost equally dominant in the field, Microsoft SQL Server ranks lower than expected and Spark SQL ranks higher than the DB-Engine ranking. ClickHouse also follows a highly fluctuating trajectory although its popularity should be steadily increasing.

6.3 WARC files

We acquired access to a larger load of data in a less computationally intensive manner by using the Index to the WARC Files and URLs that were provided by Common Crawl in Columnar Format. Since the findings of this strategy only affect the URLs of the websites, we gained insight by examining the title of each webpage in isolation from the text. As we can see from Figure 12, Oracle is the most popular database system closely followed by MySQL throughout the years 2013 to 2022. After that, Microsoft SQL Server and

	uri	domain	suffix
0	http://abcphp.com/login.php?return=%2Fupcoming...	abcphp	com
1	http://apolot.com/index.php?content=xot	apolot	com
2	http://archives.neohapsis.com/archives/mysql/2...	neohapsis	com
3	http://b0op.com/blog/index.php/category/News	b0op	com
4	http://buffalo.nas-central.org/w/index.php?tit...	nas-central	org
...	...	...	...
63	http://www.top4download.com/free-text-extraction/	top4download	com
64	http://www.unixmen.com/linux-vs-windows-five-f...	unixmen	com
65	http://www.webdeveloper.com/forum/showthread.p...	webdeveloper	com
66	http://www.wiredatom.com/blog/category/geek-st...	wiredatom	com
67	http://www.zarafa.com/wiki/index.php?title=Sto...	zarafa	com

68 rows × 3 columns

	uri	domain	suffix
0	http://61-64-230-168-adsl-tpe.dynamic.so-net.n...	so-net	net.tw
1	http://b0mfnc.aguwe.com/	aguwe	com
2	http://blog.ganyongmeng.com/?cat=30	ganyongmeng	com
3	http://cikewudi.com/?tag=SEO%E6%96%87%E7%AB%A0	cikewudi	com
4	http://geromescuttle.awardspace.info/rss.php/a...	awardspace	info
...	...	...	...
42	https://www.hometaurus.com/home/343727270T-679...	hometaurus	com
43	https://www.jobatus.it/lavoro-sviluppatore-typ...	jobatus	it
44	https://www.myhairparadise.com/	myhairparadise	com
45	https://www.pakainfo.com/c-linq-left-outer-joi...	pakainfo	com
46	https://www.vps.net/community/knowledgebase/la...	vps	net

47 rows × 3 columns

Figure 8: Mentions of "mysql" from the 2013 sample (up) and the 2022 sample (bottom)

PostgreSQL are following, with Microsoft SQL Server showing a slightly more increasing trend from the years 2017 to 2020. MongoDB ranks fifth followed by Spark SQL. ClickHouse made its appearance in 2016 following an increasing trend and DuckDB also launched in 2020.

The results of this approach tend to be closer to the ones of DB-Engine(Figure 14). The three leading database systems are Oracle, MySQL and Microsoft SQL Server for both our approach and DB-Engine ranking but in our approach, PostgreSQL appears to have similar popularity with Microsoft SQL Server while it should be lower. Also, although MongoDB ranks a bit lower than PostgreSQL, they should be similar for the years 2015 and 2016, while in our chart it is obvious that MongoDB is mentioned less. SparkSQL follows with a higher popularity than expected even years before its launch. This could be a result of a false specification of our query or because there might be a mention of Spark as a framework and not a database. Except for the SparkSQL, we have captured in greater detail than the first approach the launches of the new database systems ClickHouse and DuckDB. We also plotted our results for 6-month intervals (per semester) to obtain a higher level of detail in the trends of the database systems(Figure 13).

6.4 Methods Comparison

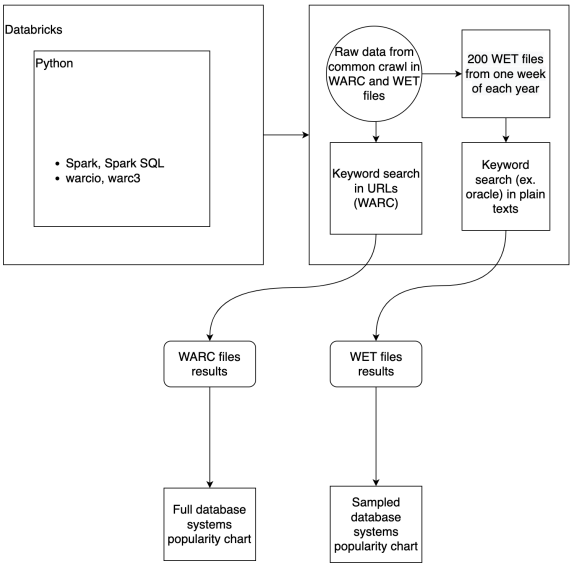

Figure 9: Pipeline

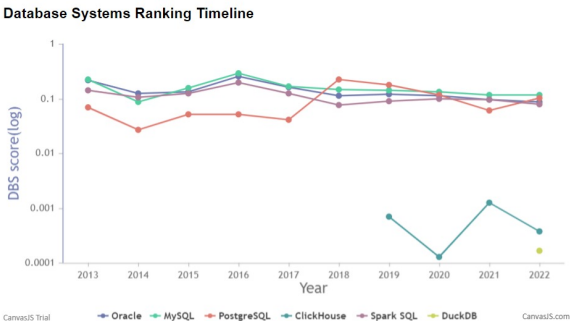

Figure 10: Database Systems Ranking Timeline for 100 WET files per Year

We realized that the method using the titles of the webpages provided by the WARC files had more similarities with the DB-Engines ranking than the approach using the text of the webpages. The most successful strategy utilizes only the titles of websites via URLs and outperforms the strategy that explores their textual content. As a result, despite the fact that a significant portion of relevant information was lost from the text, it has access to a much larger sample of data regarding the web pages that it explored. The second approach also is more sensitive to changes. For ClickHouse and DuckDB, it detected results closer to their actual debut than the first approach.

When we scan the text, we gain a more in-depth understanding of webpages that may not mention the name of the database in the title but may be part of the corpus. This, of course, reflects on the computational cost of each website so in order to come up with a viable solution we had to reduce the number of web pages we investigated.

According to the results, a more superficial approach with a larger amount of data produces better results than an in-depth search with less data.

6.5 Validity of Results

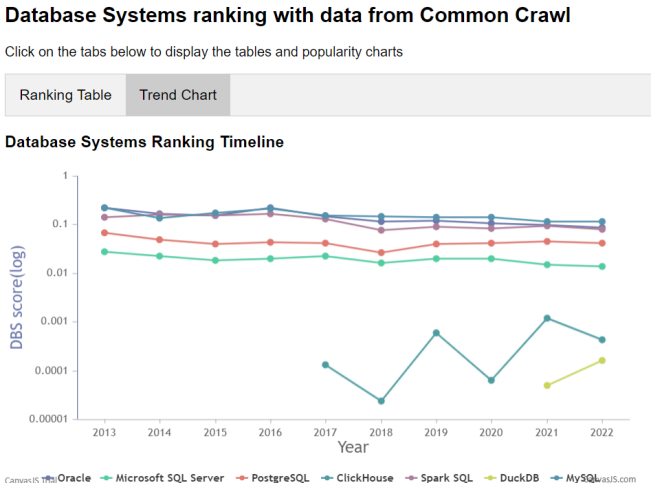

Figure 11: Database Systems Ranking Timeline for WET files

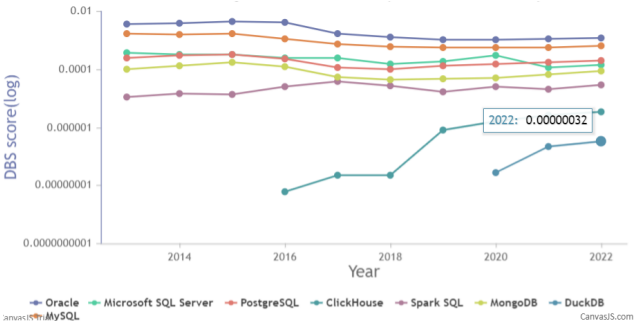

Figure 12: Database Systems Ranking Timeline for the titles of WARC files

As we have already discussed in the previous sections both our approaches contain some remarkable shortcuts that we had to apply in order to manage the load of the data which was not viable with the tools that we utilized. As a result, although we employed techniques to provide a representative portion of the given data, the quality of our results was compromised.

Furthermore, in cases where the database system has the same name as other tools or frameworks (like Oracle or Spark) there might be cases where mentions of this tool and the word database have occurred but these mentions are not related to each other. Given these circumstances, we may incorrectly count it as a valid mention. In order to solve this problem, we should have examined the context around our words of interest to specify whether it is indeed a valid occurrence.

7. DISCUSSION

In this project, we explored the web archives provided by Common Crawl looking for mentions of database systems to construct a timeline of their popularity. Our biggest barrier was the size of the data that we had to manage in order to calculate our results. In the Results section, we have discussed the technical barriers and their implications for

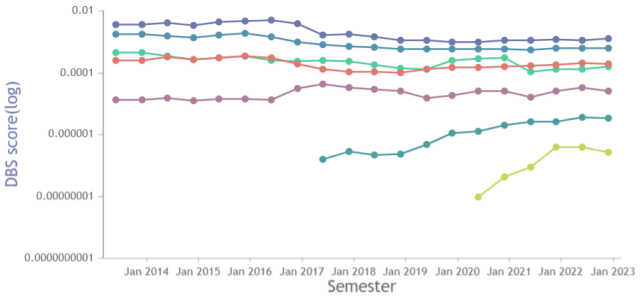

Figure 13: Database Systems Ranking Timeline per Semester for the titles of WARC files

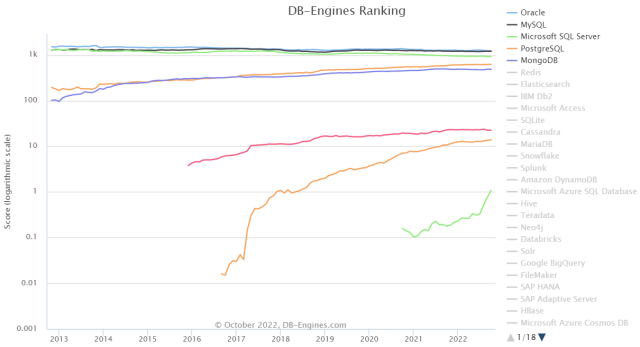

Figure 14: Ranking of Database Systems in DB-Engines

the validity of our results. Our main conclusion was that a breadth-first approach using more files provides better results than a depth-first approach with fewer files.

Our point of reference for the evaluation of our results is the DB-Engine website which provides a more sophisticated method for calculating the rank of the database systems by popularity. According to their website, they combine several methods to form their ranking. Besides that, they measure the number of results in search engine queries. They also measure the frequency of searches in Google Trends, the frequency of technical discussions about the system, and the number of interested users on well-known IT-related Q&A sites, job offers on job search engines, profiles in professional networks, in which the system is mentioned and the number of Twitter tweets.

Our approach is limited to the data provided by Common Crawl which contains a subset of data from the web and not the whole web. Although Common Crawl attempts a broad search, obtaining a large sample of the web rather than a deep sampling of a few websites, our data is limited to the data sampled by Common Crawl's criteria. Furthermore, there are studies that have investigated some bugs in the sampling of Common Crawl which may have affected the quality of the initial dataset[1]. More specifically, they have found out that there are cases of duplication of crawled pages within a single release, and this could affect our results. As a result, our sampling process was affected by the Common Crawl's selection criteria.

Furthermore, we took the decision to downsample the initial dataset to a limited amount of data that would allow us to work with the technologies used. In large-scale set-

Salmane Dazine	Effrosyni Stergiopoulou	Anastasios Bazinis
Data Investigation and Analysis WET files query from 2013 to 2017 Report (following the respective tasks)	Queries on the WARC files Website Development Report (following the respective tasks)	WET files query from 2018 to 2022 Presentation structure Report (following the respective tasks)

Table 2: Distribution of work

tings, fast performance is a key requirement, so Python is not considered an optimal solution. For example, Scala has faster performance than Python, and since Spark is native to Scala, it would be a better fit for our project. If we chose Scala, we would be able to use more data for our project, which would result in more reliable results. One more way to improve our solution would be a more targeted sampling process. For example, we could have added one more step in our data ingestion pipeline, employing Data Mining on the webpages by deciding whether we are going to keep them or not based on their content. This is a binary classification problem (relevant or not) that would allow us to keep the most relevant webpages in a less computationally intensive manner.

8. WORK DISTRIBUTION

Each team member is given a specific job in the project. Because of the team's size in relation to the project's complexity, team members frequently cooperate within their given duties. Nevertheless, the work distribution was carried on equally between team members in the following way.

9. CONCLUSION

To conclude, this study demonstrates the process of analyzing the popularity of database systems by extracting their mentions from samples taken from the Common Crawl corpus. First of all, we analyzed small samples from 2013 and 2022 to learn about the format, size, and distribution of data, and to invent sampling techniques to reduce the running time without affecting the quality of the results. Then we applied two different approaches consisting of a depth-first approach that samples the data by taking the plain texts from similar periods from each year and a breadth-first approach that samples through analyzing the URLs of all the 230 billion websites provided by the Common Crawl dataset. Although both approaches consumed a similar amount of resources and time, the breadth-first approach achieved similar results to our selected point of reference and proven to be more representative of the popularity of the database systems. Nevertheless, it is still questionable whether public opinion on this popularity is positive or negative. Therefore, the depth-first approach needs to be employed and improved by selecting the context of the mentions in order to analyze its overall sentiment.

10. REFERENCES

[1] H. S. Thompson and J. Tong. Can common crawl reliably track persistent identifier (pid) use over time. In *Companion Proceedings of the The Web Conference 2018*, pages 1749–1755, 2018.