



## ADVANCE PROJECT 1

BY  
Konrad Burchardt  
20-06-2019

# Search Engine Data Integration Dashboard

Professor: Dr. Aldabert Wilhelm  
Spring Term 2019  
MRD005-340001

# Executive Summary

## Task and Goal

The task for this project is to create an ETL pipeline using Google search organic traffic and load the data in a user friendly dashboard. ETL process performs data cleaning during extraction process and load significant data into data warehouse. Our main goal for this project is by having this pipeline we will be able to have the data in one place and make it easier for a businesses stakeholders to access this data, analyze it and discover different business insights.

## Data Background

We will be working with Monthly Google search organic traffic from the domain <https://www.tuves.cl>. This data will be acquire by using the Google search API. We will use 9 different data sets of search traffic and they are the following:

- **Mobile, Desktop and Tablet - Day to day Organic data:** These are multiples CSV files that are organized by date and includes Keywords, Clicks, Impressions, CTR, Ranking, Device and Date.
- **Mobile, Desktop and Tablet - Monthly Top 10 Keywords:** One data set that includes Keyword, Clicks, Impressions, Avg CTR and Avg Ranking for the specific device and time frame.
- **Mobile, Desktop and Tablet - Monthly Top 10 URLs:** One data set that includes URL, Clicks, Impressions, Avg CTR and Avg Ranking for the specific device and time frame.

## Approach and Methods used

This project is composed by 5 steps.

1. **Extraction:** Extract the Google search data using the Google search API. Our data will be saved in CSV files.
2. **Transform:** Transform the day to day data into 3 monthly data sets. A mobile data set, a desktop and a tablet. The Keywords and URLs are ready to go.
3. **Load** Load the data to the data warehouse. We will be using a MySQL database.
4. **Load** Build a REST API that will parse all of our MySQL tables into JSON.
5. **Analyze** Create a dashboard where we will load our data and analyze it using tables and visualizations.

## Results

Our objective was to create a easy-to-use interface that displays Organic Search data in a simple way. Our result can be seen live in the following URL: <https://www.metacrawler.es/dashboard> .

Our outcome was positive, we were able to create a user friendly dashboard that allowed the SEO Managers of the business, find important insights and use it as an easy monthly reporting tool across the company.

The next steps for this project would be to take this report to the next level. First thing will be to automate each of the process, then we will include new KPI's and include machine learning to find which pages need attention and optimization.

# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Background of Data</b>	<b>7</b>
2.1	What is SEO? . . . . .	7
2.2	How does SEO Work? . . . . .	7
2.3	Organic Traffic . . . . .	7
2.4	Our Data . . . . .	8
2.4.1	Day to day Traffic . . . . .	8
2.4.2	Top 10 Keywords (Mobile, Desktop and Tablet) . . . . .	9
2.4.3	Top 10 URLs (Mobile, Desktop and Tablet) . . . . .	9
<b>3</b>	<b>Data Preprocessing</b>	<b>11</b>
3.1	Extraction: Google Search Console API . . . . .	11
3.1.1	Day to day Data . . . . .	11
3.1.2	Top 10 Keywords . . . . .	12
3.1.3	Top 10 Urls . . . . .	13
3.2	Transform: Data Cleaning . . . . .	14
3.2.1	Day to Day Data Cleaning . . . . .	14
3.2.2	Recap . . . . .	15
3.3	Load: Data to data warehouse(MySQL) . . . . .	16
3.3.1	Creating Database . . . . .	16
3.3.2	Loading data sets . . . . .	16
3.4	Load: RestAPI(NodeJS) . . . . .	18
3.4.1	API Index . . . . .	20
3.5	Analysis: SEO Dashboard(ReactJS) . . . . .	20
3.5.1	Dashboard structure . . . . .	21
3.5.2	React App Structure: . . . . .	26
<b>4</b>	<b>Data Exploration</b>	<b>28</b>
4.1	Mobile . . . . .	28
4.1.1	Example 1 . . . . .	28
4.1.2	Example 2 . . . . .	29
<b>5</b>	<b>Results and Conclusions</b>	<b>31</b>



# Chapter 1

## Introduction

This report is focused on the creation of an ETL pipeline and a user friendly SEO dashboard. ETL pipelines refers to a set of processes extracting data from one system, transforming it, and loading into some database or data-warehouse. A SEO data dashboard is a information tool that includes visualization and display important KPI's Related to search engine traffic. These metrics and key data points are used to monitor the health of the SEO effort in a business and to make important business decisions. In this report we used Google search data(Organic traffic) from the domain <https://www.tuves.cl>. Tuves HD is an international provider of satellite television in high definition that transmits hundreds of digital signals , with headquarters in Santiago de Chile and customers throughout South and Central America[1]. Organic traffic refers to the visitors that come to our website as a result of unpaid search results[2].

In this project we used different methods, tools and steps to create our pipeline as well as our dashboard. The steps and technologies used are the following (*All of these steps will be explained in depth in the Data Prepossessing section* :

1. **Extraction:** Extract the Google search data using the Google search API.We will use a Python Script that will make an API call to Google servers and export the desired data into multiples CSV files.
2. **Transform:** Transform the day to day data into 3 monthly data sets. Each of the CSV files is equivalent to o day of data. Our focus is to create 3 data sets with 30 days of data from these CSV files. For each day we will create a line in our main CSV file. The outcome will be 3 data sets, Mobile, Desktop and Tablet, with the 30 of day Organic data. Here we use a python script.
3. **Load** Load the data to the data warehouse. We will create an empty database in MySQL and every time we load one of our CSV files it will generate a table with the CSV columns.

4. **Load** Build a REST API that will parse all of our MySQL tables into JSON. We will send our data to our app in JSON format using NodeJS and ExpressJS. The API will have 9 different urls that include all of the desired data that we need to make our dashboard app.
5. **Analyze** Create a dashboard using ReactJS and ChartJS. We will load our data from our RESTapi. After loading the data we will generate user friendly charts and tables. We will give a simple analysis of what can we find using this dashboard.

## Chapter 2

# Background of Data

In this chapter we will explain what SEO is and how does it work to give a good introduction on the type of data we are working on. We also will explain what is Organic traffic because we will use this term many times in our project and finally we will explain how our data is composed and the structure of it.

### 2.1 What is SEO?

SEO stands for Search Engine Optimization and is the process of getting free organic traffic from search engines. SEO is different than paid search, in SEO you can't pay to be in the number one position of the search result page. There are multiple ways you can optimize your website for a target keyword to rank in top positions. Keyword have different search volumes and the better you rank for a high volume keyword the more clicks.

### 2.2 How does SEO Work?

Major search engines use web crawlers to discover publicly available web pages. These crawlers look at web pages and follow links on these web pages. They go link to link gathering information and taking notes of key signals, from keywords to website freshness . All the information that the crawler find is added to the search index, where they contain billions of web pages and is over 100,000,000 gigabytes in size[3]. This index is fed to a search algorithm that checks all the information and signals of the web page and creates an entry that is assigned to a keyword.

### 2.3 Organic Traffic

The term organic traffic is used to refer to the traffic that go to a website as a result of unpaid search results. As mentioned before, Organic traffic is the



opposite of paid traffic. Traffic that is considered organic, are visitors that finds a website after using a search engine like Yahoo, Bing or Google.

## 2.4 Our Data

The Data we are using is all organic traffic from May 2019(30 days) from the website <https://www.tuves.cl>. Our data is divided into 9 different data sets:

- **Overall Day to day traffic**
- **Mobile, desktop and tablet top 10 keywords**
- **Mobile, desktop and tablet top 10 urls**

### 2.4.1 Day to day Traffic

Day to day traffic is all traffic that we had in the month of May in 2019. This is composed by multiple CSV files. Each CSV file is a day of the month of May and are named with the date for example **20190501.csv**

Each of the CSV files contain the following rows:

Keywords	Clicks	Impressions	CTR	Position	Device	Date
----------	--------	-------------	-----	----------	--------	------

Each line correspond to a Keyword that brought traffic to our site that day. For example the first 3 lines of **20190501.csv** would look like this:

Keywords	Clicks	Impressions	CTR	Position	Device	Date
tuves hd	80	113	70.8%	1	Mobile	1/05/19
tuves	52	114	45%	1.2	desktop	1/05/19
tu ves	43	94	45.7%	1.2	desktop	1/05/19

The columns have the following meanings:

- **Keywords:** Keyword that the website was ranking for.
- **Clicks:** How many times a user clicked through to the site. How this is counted depends on the search result type.
- **Impressions:**How many times a user saw a link to the site in search results. This is calculated differently for images and other search result types, depending on whether or not the result was scrolled into view.
- **CTR:** Is the percentage of impressions that resulted in a click.
- **Position:** Is the average position of the site in search results, based on its highest position whenever it appeared in a search.

- **Device:** Device the user searched for that keywords.
- **Date:** Date the search was made.

### 2.4.2 Top 10 Keywords (Mobile, Desktop and Tablet)

Top 10 Keywords are the top 10 keywords that brought traffic in the month of May in 2019. This is composed by 3 different CSV files. One for the top 10 mobile keywords, desktop top 10 keywords and tablet top 10 keywords.

Each of the CSV files contain the following rows:

Keywords	Clicks	Impressions	CTR	Position
----------	--------	-------------	-----	----------

Each line of this set will correspond to a top keywords that brought traffic in the month of may 2019. For example the top 3 desktop keywords for May 2019 would look like this:

Keywords	Clicks	Impressions	CTR	Position
'tuves'	640	1066	0.6	1.1
'tuves hd'	301.0	482.0	0.6	1.0
'tu ves'	210.0	515.0	0.4	1.5

The columns have the following meanings:

- **Keywords:** Top Keyword that the website was ranking for the month of May 2019.
- **Clicks:** How many times a user clicked through to the site in the month of may. How this is counted depends on the search result type.
- **Impressions:**How many times a user saw a link to the site in search results in the month of May. This is calculated differently for images and other search result types, depending on whether or not the result was scrolled into view.
- **CTR:** Is the percentage of impressions that resulted in a click in the month of May.
- **Position:** Is the average position of the site in search results, based on its highest position whenever it appeared in a search in the month of May.

### 2.4.3 Top 10 URLs (Mobile, Desktop and Tablet)

Top 10 URLs are the top URLs that brought traffic in the month of May in 2019. This is composed by 3 different CSV files. One for the top 10 mobile URLs, desktop top 10 URLs and tablet top 10 URLs.

Each of the CSV files contain the following rows:

URLs	Clicks	Impressions	CTR	Position
------	--------	-------------	-----	----------

Each line of this set will correspond to a top URL that brought traffic in the month of may 2019. For example the top 3 URLs for May 2019 would look like:

URLs	Clicks	Impressions	CTR	Position
<a href="https://www.tuves.cl/">https://www.tuves.cl/</a>	13,786	94,267	14.6%	8.4
<a href="https://www.tuves.cl/ayuda/te-llamamos/">https://www.tuves.cl/ayuda/te-llamamos/</a>	1,704	30,193	5.6%	2.7
<a href="https://www.tuves.cl/entretenimiento/guia-de-canales/">https://www.tuves.cl/entretenimiento/guia-de-canales/</a>	816	25,694	3.2%	5.1

The columns have the following meanings:

- **URLs:** Top URL for the month of May 2019.
- **Clicks:** How many times a user clicked through to the site in the month of May of 2019. How this is counted depends on the search result type.
- **Impressions:**How many times a user saw a link to the site in search results in the month of May of 2019. This is calculated differently for images and other search result types, depending on whether or not the result was scrolled into view.
- **CTR:** Is the percentage of impressions that resulted in a click in the month of May of 2019.
- **Position:** Is the average position of the site in search results, based on its highest position whenever it appeared in a search in the month of May of 2019.

## Chapter 3

# Data Preprocessing

In this chapter we will go through our data preprocessing steps. Starting with The extraction of the data, followed by data transformation, Loading the data to our data warehouse, creating a RestAPI and finally creating a SEO dashboard for analysis.

### 3.1 Extraction: Google Search Console API

To extract our data, we use the Search Console API. The Search Console is a free tool that helps webmasters monitor their site's performance in Google search, ensure that Google can crawl their sites or apps correctly, and to test the validity and performance of a given page. The Search Console provides programmatic access to the service through API[4]. The advantage of using this API is that we can run advanced queries for Google Search results data for our properties that we manage in our Search console.

To extract our data we created 2 python applications using the python client library **google-api-python-client**. For full documentation to set up this you can access <https://developers.google.com/webmaster-tools/search-console-api-original/v3/quickstart/quickstart-python>.

#### 3.1.1 Day to day Data

To request our day to day Data we created a Python app(GSC.py) that fetches 30 days of Organic data from the site <https://www.tuves.cl> and export each day as a CSV file.

We run our app in our terminal console and we need to pass 3 arguments, property URL, starting date and ending date. To get the data for the month of May 2019 we will run the following bash command:

#### Code:

---

```
$ python3 GSC.py https://www.tuves.cl 2019-05-01 2019-05-31
```

---

This command will output 31 separate CSV files that are named with the date and ordered by name.

### 3.1.2 Top 10 Keywords

To request our Top 10 Keywords we created a Python app that fetches the top 10 keywords for the month of May 2019 for the site <https://www.tuves.cl>. Since we want data for 3 different devices, we created 6 folders and added a filter in the code for each specific device.

For example, for fetching the desktop top Keywords, the filter would look like this:

#### Code:

```
# Get top 10 queries for the date range, sorted by click count, descending.
request = {
    'startDate': flags.start_date,
    'endDate': flags.end_date,
    'dimensions': ['query'],
    'dimensionFilterGroups': [{
        'filters': [{
            'dimension': 'device',
            'expression': 'desktop'
        }]
    }],
    'rowLimit': 10
}
response = execute_request(service, flags.property_uri, request)
print_table(response, 'Top Queries')
```

We run our app in our terminal console and we need to pass 3 arguments, property url, starting date and ending date. To get the desktop data for the month of May 2019 we will run the following shell command:

#### Code:

---

```
$ python3 kws-desktop.py https://www.tuves.cl 2019-05-01 2019-05-31
```

---

#### Output:

-Top Queries:

Keys	Clicks	Impressions	CTR	Position
'tuves'	640.0	1066.0	0.600375234521576	1.1894934333958724
'tuves hd'	301.0	482.0	0.6244813278008299	1.0269709543568464
'tu ves'	210.0	515.0	0.4077669902912621	1.5650485436893202
'tu ves hd'	169.0	231.0	0.7316017316017316	1.0303030303030303
'tuves programaci	154.0	156.0	0.9871794871794872	1.0
'canales online'	128.0	2022.0	0.06330365974282888	8.138476755687439
'tuves chile'	121.0	183.0	0.6612021857923497	1.0163934426229508
'tuveshd'	107.0	219.0	0.4885844748858447	1.004566210045662
'tuves.cl'	66.0	101.0	0.6534653465346535	1.0
'tv cable'	59.0	864.0	0.06828703703703703	3.8043981481481484

### 3.1.3 Top 10 Urls

To request our Top 10 URLs we created a Python app that fetches the top 10 URLs for the month of May 2019 for the site <https://www.tuves.cl>. Since we want data for 3 different devices, we created 6 folders and added a filter in the code for each specific device.

For example, for fetching the desktop top URLs, the filter would look like this:

**Code:**

```
# Get top 10 pages for the date range, sorted by click count, descending.
request = {
    'startDate': flags.start_date,
    'endDate': flags.end_date,
    'dimensions': ['page'],
    'dimensionFilterGroups': [{
        'filters': [{
            'dimension': 'device',
            'expression': 'mobile'
        }]
    }],
    'rowLimit': 10
}
response = execute_request(service, flags.property_uri, request)
print_table(response, 'Top Pages')
```

We run our app in our terminal console and we need to pass 3 arguments, property URL, starting date and ending date. To get the desktop data for the month of May 2019 we will run the following shell command:

**Code:**

```
$ python3 kws-desktop.py https://www.tuves.cl 2019-05-01 2019-05-31
```

## Output:

-Top Pages:

Url	Clicks	Impressions	CTR	Position
https://www.tuves.cl/	8707.0	44036.0	0.19772458897265874	5.363952220910164
'https://www.tuves.cl/.../'	1083.0	19234.0	0.0563065405011958	2.2743059166060102
'https://www.tuves.cl/.../'	373.0	14946.0	0.0249565101030376	2.546433828449083
'https://www.tuves.cl/.../'	296.0	19296.0	0.015339966832504145	2.204342868988391
'https://www.tuves.cl/.../'	270.0	9031.0	0.029897021370833794	2.6283910973314137
'https://www.tuves.cl/.../'	231.0	21045.0	0.01097647897362794	2.0939415538132575
'https://www.tuves.cl/.../'	195.0	9864.0	0.019768856447688565	3.527169505271695
'https://www.tuves.cl/.../'	141.0	17018.0	0.008285344928898814	3.2336937360441884
'https://www.tuves.cl/.../'	108.0	12813.0	0.00842893935846406	9.725747287910716
'https://www.tuves.cl/.../'	107.0	13324.0	0.008030621435004503	2.957370159111378

## 3.2 Transform: Data Cleaning

After we Extract all of our data, we need to transform the data so it can meet our requirements for the dashboard. In this step of our pipeline, we will only transform the Day to Day data set. The reason the top 10 Keywords and top 10 URLs data sets don't need to be transformed, is because when we do the extraction we created a filter that takes only what we need.

### 3.2.1 Day to Day Data Cleaning

As we mention in the Data Background section, each CSV file of the day to day data is a day of organic traffic for the month of May 2019. We have 31 CSV files(May has 31 days) that are ordered by date. They are composed by multiple keywords and 6 other variables(Clicks, Impressions, CTR, Position, Date and Device). Our goal is to transform each of these daily CSV files into a line of data that contains the following variables:

- **id**:Id of the row.
- **Clicks**:Sum of all clicks.
- **Impressions**: Sum of all impressions.
- **CTR**: Average CTR.
- **Position**: Average Position.
- **Date**: Date of the set.
- **CTR**: Device of the data.

After converting each of the data set into a line, we concatenate all the lines into 3 data sets. One for mobile, one for Tablet and one for Desktop.

For example, the mobile data set looks like this:

id	Clicks	Impressions	CTR	Position	Date	Device
0	379	1014	0.615711695	2.602450144	5/1/19	mobile
1	400	960	0.573132299	2.822567794	5/2/19	mobile
2	360	945	0.589793023	2.689822625	5/3/19	mobile
3	363	843	0.613342878	2.383801717	5/4/19	mobile
4	391	2918	0.580306715	2.909648219	5/5/19	mobile
5	396	1367	0.519853109	2.362280997	5/6/19	mobile
6	340	879	0.510336075	4.337077398	5/7/19	mobile
7	325	814	0.549115849	3.875868415	5/8/19	mobile
8	286	616	0.652413786	2.150957361	5/9/19	mobile
9	268	664	0.447668505	4.642320224	5/10/19	mobile
10	369	758	0.529216957	4.710535639	5/11/19	mobile
11	334	1669	0.513077464	2.941114507	5/12/19	mobile
12	346	915	0.576892733	3.225619362	5/13/19	mobile
13	309	735	0.544008395	4.322843589	5/14/19	mobile
14	362	850	0.533903694	3.964114416	5/15/19	mobile
15	397	867	0.634449593	2.012856143	5/16/19	mobile
16	427	970	0.536084115	2.765507274	5/17/19	mobile
17	498	1253	0.556790163	2.271800498	5/18/19	mobile
18	321	863	0.586871754	3.191501317	5/19/19	mobile
19	315	844	0.580621592	2.994429223	5/20/19	mobile
20	312	750	0.60706999	3.055721023	5/21/19	mobile
21	286	800	0.457393728	3.301287637	5/22/19	mobile
22	307	717	0.525906528	5.2718324	5/23/19	mobile
23	282	727	0.45576858	3.833712856	5/24/19	mobile
24	369	920	0.591262244	2.671607584	5/25/19	mobile
25	278	784	0.545042768	3.367077096	5/26/19	mobile
26	298	887	0.464505513	2.504708956	5/27/19	mobile
27	265	757	0.408925207	4.350114295	5/28/19	mobile
28	356	964	0.539083878	3.925230915	5/29/19	mobile
29	370	880	0.577955233	2.413126305	5/30/19	mobile
30	405	1008	0.573972137	2.778068554	5/31/19	mobile

Each of line represent one day worth of data. Each data sets, will be exported as a CSV file and added to the folder where we have our other data sets.

### 3.2.2 Recap

Now that we finished the extraction and load, we should have 9 data sets that contain all of our data that we want to include in our dashboard. These data



sets are the following:

- **mobile.csv.**
- **keywords-mobile.csv**
- **urls-mobile.csv**
- **desktop.csv**
- **keywords-desktop.csv**
- **url-desktop.csv**
- **tablet.csv**
- **keywords-tablet.csv**
- **url-tablet.csv**

You can find all of these in the folder folder **/3data-to-sql**

### 3.3 Load: Data to data warehouse(MySQL)

After the data extraction and transformation takes places, we now need to store our 9 data sets into a data warehouse. For this we will be using a MySQL server to store all of our data sets as new tables.

Since this is only an academic project we will use a tool called MAMP[5] that creates a local server environment. To download the tool you can go to: <https://www.mamp.info/en/downloads/>

#### 3.3.1 Creating Database

To store our data sets we need to create a database in our MySQL server. We will create an empty database that will be fed with our data set. Each data set will be a table and each of the rows of the data set will be the rows of the table. The database will be called **ecom**.

To store our data we created a python script that pushes each of our CSV files into the database ecom. When we push a file, the script creates a table with the file name and the rows of the data set.

In the python script we first create a connection to our database using the library sqlalchemy. then we proceed to export each set together.

#### 3.3.2 Loading data sets

To store our data sets we used the pandas library to read our csv, then we use os library to find the path of the file and finally we use pymysql library to send the dataset to the MySQL.

For each device we send the 3 data sets in order. The code below shows an example of how we push the mobile data.

**Code:**

```
#-----exporting mobile-----
csvFileName = 'mobile.csv'
mobile = pd.read_csv(os.path.join(csvFileName))
mobile.to_sql(name=csvFileName[:-4], con=mydb, if_exists = 'replace', index=False)
```

```
#exporting Keywords
csvFileName ='keywords-mobile.csv'
tablet =pd.read_csv(os.path.join(csvFileName))
tablet.to_sql(name=csvFileName[:-4], con=mydb, if_exists ='replace', index=False)

#exporting Urls
csvFileName ='urls-mobile.csv'
tablet =pd.read_csv(os.path.join(csvFileName))
tablet.to_sql(name=csvFileName[:-4], con=mydb, if_exists ='replace', index=False)
```

Once the data is sent to MySQL, it creates a new table with the same structure our CSV file had.

### Output:

id	Clicks	Impressions	CTR	Position	Date	Device
0	103	403	0.228740243528401	23.4732087320574	2019-05-01	desktop
1	118	442	0.269557736433362	17.333159658899	2019-05-02	desktop
2	124	408	0.313530567307244	18.7469327772201	2019-05-03	desktop
3	76	329	0.237820115620649	23.8930991881287	2019-05-04	desktop
4	97	571	0.294868698985078	19.4575104450794	2019-05-05	desktop
5	143	512	0.309775120979167	13.4997204737123	2019-05-06	desktop
6	144	486	0.336182322557356	16.8096413001483	2019-05-07	desktop
7	114	427	0.289007308801782	20.5837963192721	2019-05-08	desktop
8	85	296	0.259474567099567	23.1886764069264	2019-05-09	desktop
9	70	320	0.185561875746086	30.5637430882957	2019-05-10	desktop
10	63	374	0.153491557559768	32.2221138613503	2019-05-11	desktop
11	72	335	0.207208042258739	27.6258860263404	2019-05-12	desktop
12	111	429	0.195264944739116	23.247848314865	2019-05-13	desktop
13	95	355	0.204612277667984	27.0620506011199	2019-05-14	desktop
14	95	357	0.241827175215984	25.882503873252	2019-05-15	desktop
15	95	341	0.255837866176281	27.7442700583066	2019-05-16	desktop
16	110	375	0.222438497124818	29.4945966645165	2019-05-17	desktop
17	96	363	0.241766790714159	22.1652276967672	2019-05-18	desktop
18	74	562	0.14736322986242	28.6770538146868	2019-05-19	desktop
19	100	522	0.177308488440367	27.5989309568603	2019-05-20	desktop
20	65	315	0.18568299244615	27.8295139772508	2019-05-21	desktop
21	105	406	0.218920264420264	29.6130644378144	2019-05-22	desktop
22	89	314	0.278300662531404	27.3666166615849	2019-05-23	desktop
23	97	317	0.245178800984708	31.202211335802	2019-05-24	desktop
24	70	284	0.194839066105818	32.382163256607	2019-05-25	desktop
25	57	293	0.178964576344372	31.9423884423334	2019-05-26	desktop
26	98	404	0.185800674318641	25.6649521301847	2019-05-27	desktop
27	86	264	0.246714779372674	26.9595873205742	2019-05-28	desktop
28	93	376	0.252782613503067	28.2051154602192	2019-05-29	desktop
29	141	465	0.287644975078249	19.2252538136157	2019-05-30	desktop

After the script is done we should have 9 new tables in our database. Every time we run this script, the new table will overwrite the old table. This is in case we want to do different time frames.

The ecom database should look something like this:

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> desktop	Browse  Structure  Search  Insert  Empty  Drop	31	InnoDB	utf8_general_ci	16 K B	-
<input type="checkbox"/> keywords-desktop	Browse  Structure  Search  Insert  Empty  Drop	10	InnoDB	utf8_general_ci	16 K B	-
<input type="checkbox"/> keywords-mobile	Browse  Structure  Search  Insert  Empty  Drop	10	InnoDB	utf8_general_ci	16 K B	-
<input type="checkbox"/> keywords-tablet	Browse  Structure  Search  Insert  Empty  Drop	10	InnoDB	utf8_general_ci	16 K B	-
<input type="checkbox"/> mobile	Browse  Structure  Search  Insert  Empty  Drop	31	InnoDB	utf8_general_ci	16 K B	-
<input type="checkbox"/> tablet	Browse  Structure  Search  Insert  Empty  Drop	31	InnoDB	utf8_general_ci	16 K B	-
<input type="checkbox"/> urls-desktop	Browse  Structure  Search  Insert  Empty  Drop	10	InnoDB	utf8_general_ci	16 K B	-
<input type="checkbox"/> urls-mobile	Browse  Structure  Search  Insert  Empty  Drop	10	InnoDB	utf8_general_ci	16 K B	-
<input type="checkbox"/> urls-tablet	Browse  Structure  Search  Insert  Empty  Drop	10	InnoDB	utf8_general_ci	16 K B	-
9 tables	Sum	153	InnoDB	utf8_general_ci	144 K B	0 B

### 3.4 Load: RestAPI(NodeJS)

After the data is stored in a data warehouse, we need to create a way to send our data in a format that works with HTTP. To do this we will use a RESTful application that will use a GET HTTP request to get our data.[6] Since our Dashboard is built in React we will be using JSON as our transport data format as it is easy to work with in JavaScript. There are many benefits of using JSON APIs, but the most important one in my opinion is that the data is structured in a way that is easy for a human to understand.

To build our API we will be using NodeJS as the server environment. NodeJS is designed to build scalable network applications[7]. To build our API we will be using ExpressJS. ExpressJS is a fast and easy to use web frame for NodeJS, it helps us manage everything, from routes, to handling requests and views.

To get our specific data into different URL, we create 9 different routes. We build a get request and specify the route, for example for the mobile data the route will be /mobile. Then we specify the callback function where we create 2 variables. The first variable is the **connection** to our MySQL server this will look like this:

Code:

```
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  socketPath : '/Applications/MAMP/tmp/mysql/mysql.sock',
  password: 'root',
  database : 'ecom'
})
```

The second variable is our query execution to pull the data from our database. We use the first variable, the **connection** and use a the query method. Since we want everything in our data we will do SELECT \* from table name. Then we create a function with 3 parameters, err, rows and fields. Inside our function we create an if statement that if its error send a 500 https status else response json rows.

The code looks like this:

#### Code:

```
connection.query("SELECT * FROM desktop", (err,rows,fields) => {
  if (err) {
    console.log("Failed to query for" + err)
    res.sendStatu(500)
    return
    //throw err
  }
  console.log("I think we fetch users succesfully")
  res.json(rows)
})
```

If everything is working correctly we should be able to go to <http://localhost:3003/mobile> and get our whole mobile table into JSON format.

#### Output:

```
{
  id: 0,
  Clicks: 379,
  Impressions: 1014,
  CTR: 0.615711695306193,
  Position: 2.6024501443029,
  Date: "2019-05-01",
  Device: "mobile"
},
```

This is only one result of the whole JSON file. We should have 31 results with all the data from mobile.

### 3.4.1 API Index

To make things easier I created an API index, where we can see all the information and how to use the API.

## NodeJS REST API

REST API with ExpressJS in NodeJS. This API fetches organic data from a MySQL database. The data is from [Tuves](#). This page is a index with example and paths to help the use of the API.

#### Mobile

Get all the Mobile Clicks, Impressions, CTR and Ranks

**Endpoints:**

- `/mobile/` -- Get all data
- `/mobile/:id` -- Get specific resource

**Example request:**

```
http localhost/mobile/1
```

**Example response:**

```
{("id":1,"Clicks":63,"Impressions":138,"CTR":0.288225814,"Position":4.437770563,"Date":"1/1/19","Device":"mobile") }
```

[Go to path](#)

#### Desktop

Get all the Desktop Clicks, Impressions, CTR and Ranks

#### Top 10 Keywords

Get the top 10 Keywords Clicks, Impressions, CTR and Ranks

**Endpoints:**

- `/Keywords/` -- Get all data

**Example**

```
{(Keyword: "b'tuves", Clicks: 269, Impressions: 354, CTR: 0.759887085649718, Position: 1)} }
```

[Go to path](#)

#### Top 10 URLs

Top 10 URLs Clicks, Impressions, CTR and Ranks

**Endpoints:**

- `/URLs/` -- Get all data

**Example**

## 3.5 Analysis: SEO Dashboard(ReactJS)

After we successfully fetched our API, we want to pass this data to our dashboard so that it can be easily read and analysed. To create our Dashboard we used 4 libraries, ReactJs, ChartJs, Axios and Bootstrap.

**ReactJS** is a JavaScript library for building user interfaces. React applications are made using components that are modules that render some output.React instead of using regular JavaScript uses JSX which is JavaScript that allows HTML syntax to render components[8]. We use react to manage all of our components in our application.

**ChartJS** is a JavaScript library that allow us to create different types of charts using the HTML5 canvas element[9]. We are using this library to build our graphs.

**Axios** is a JavaScript library that is used to make HTTP request from

node.js or React.js.[10] We use axios to make the API call from our app to our RESTapi.

**Bootstrap** is a responsive and mobile-first CSS framework[11]. Is the easiest and most popular framework and we will use it to build our interface. We will be using a Stateless version that is called reactstrap[12].

### 3.5.1 Dashboard structure

Our dashboard will have 3 different views, Mobile, Desktop and Tablet. All of these will have the following structure:

1. **Top KPI:** The top of our dashboard we will have overall metrics for the period of time that include Total Clicks, Total Impressions, Average CTR and average Rank.

How do we calculate this numbers?

- **Total Clicks:** The sum of all clicks in the click rows in our JSON file.

**Code:**

```
//sum of clicks
const sumclick = clicks.reduce((a,b)=> a+b,0)
this.setState({kpi1: sumclick})
```

**Output:**

```
Total Clicks:
10714
```

- **Total Impressions:** The sum of all Impressions in the impression row in our JSON file.

**Code:**

```
//sum of impressions
const sumimpr = impressions.reduce((a,b)=> a+b,0)
this.setState({kpi2: sumimpr})
```

**Output:**

```
Total Impressions:
29938
```

- **Average CTR:** The average CTR in the CTR row in our JSON file.

Here we create a function that calculates the average. Since our numbers are not rounded we also make sure we round our numbers to have only 2 decimals.

**Code:**

```
//Creating average function
function getAvg(avg) {
  const total = avg.reduce((acc, c) => acc + c, 0);
  return total / avg.length;
}

//CTR avg
const avgctr = getAvg(ctr);
//rounding number
var roundedctr = Math.round(avgctr * 10) / 10;
this.setState({kpi3:roundedctr})
// console.log("average ctr is " + avgctr)
```

**Output:**

```
Average CTR:
0.5%
```

- **Average Rank:** The average Rank for the position row in our JSON file.

Here we create a function that calculates the average. Since our numbers are not rounded we also make sure we round our numbers to have only 2 decimals.

**Code:**

```
//Ranking AVG
const avgrank = getAvg(rank);
//rounding number
var roundedrank = Math.round(avgrank * 10) / 10;
this.setState({kpi4:roundedrank})
// console.log("average rank is " + avgrank)
```

**Output:**

```
Average Rank:
3.2
```

Our Top KPI's look like this:

### Mobile SEO Dashboard

Share Export This week ▾

Total Clicks:

10714

Total Impressions:

29938

Average CTR:

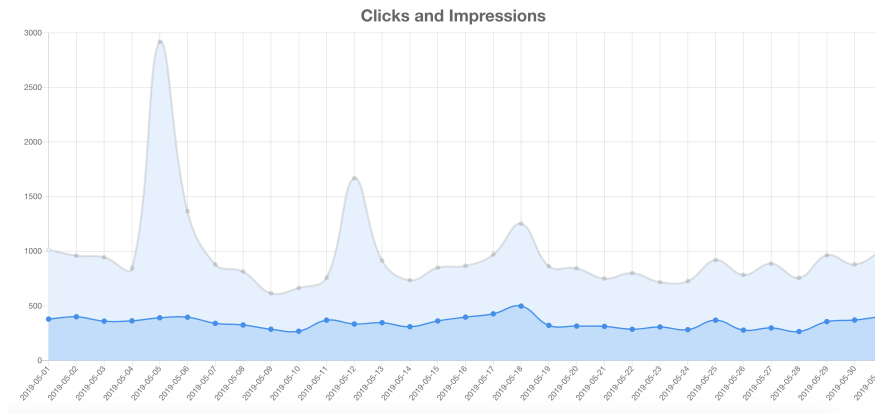
0.5%

Average Rank:

3.2

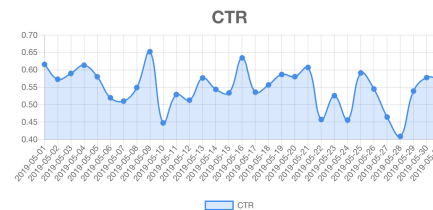
2. **Clicks and Impressions Line Chart:** After Our top KPI's we will have a line chart that will include all of the clicks and all of the impressions ordered by date. These charts are created using the ChartJS library.

Our Chart looks like this:



3. **CTR and Rank bar and Line Charts:** After our clicks and impressions we will have 2 separate charts. A bar chart for rankings and a line chart for CTR. These charts are created using the ChartJS library.

The charts look like this:





4. **Top 10 driving Keywords:** Here we will show a table with the top 10 Keywords for the time period. They will be ordered by clicks.

The charts look like this:

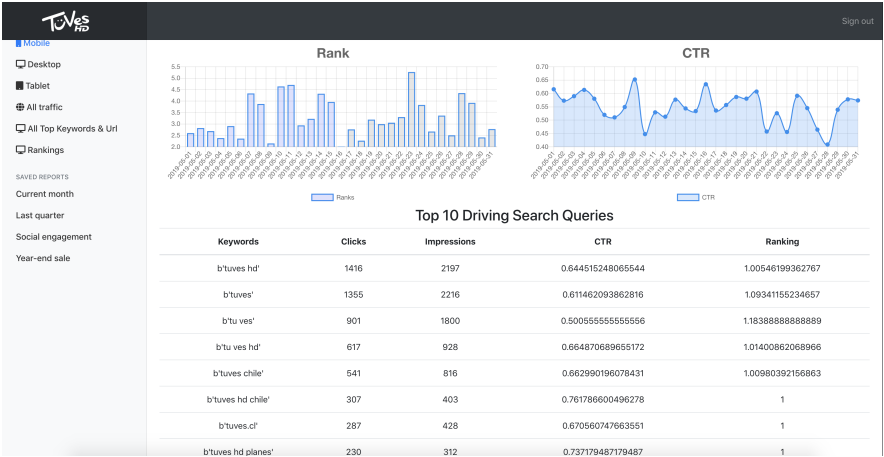
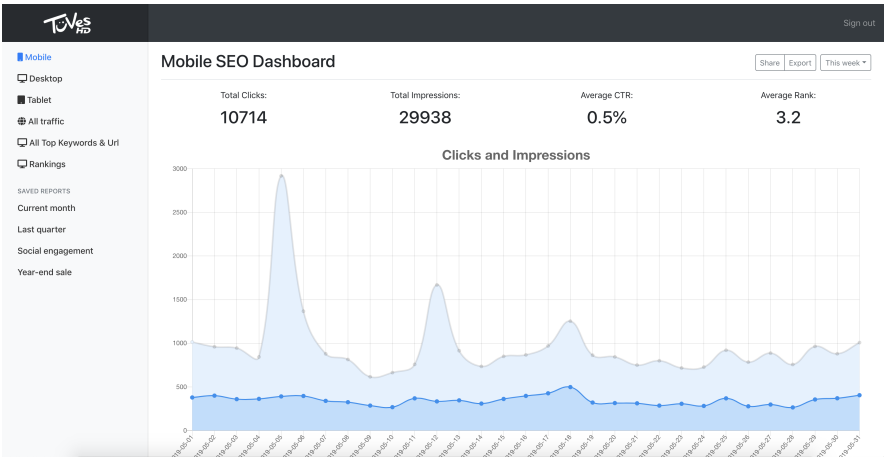
Top 10 Driving Search Queries				
Keywords	Clicks	Impressions	CTR	Ranking
b'tuves hd'	1416	2197	0.644515248065544	1.00546199362767
b'tuves'	1355	2216	0.611462093862816	1.09341155234657
b'tu ves'	901	1800	0.500555555555556	1.18388888888889
b'tu ves hd'	617	928	0.664870689655172	1.01400862068966
b'tuves chile'	541	816	0.662990196078431	1.00980392156863
b'tuves hd chile'	307	403	0.761786600496278	1
b'tuves.cl'	287	428	0.670560747663551	1
b'tuves hd planes'	230	312	0.737179487179487	1
b'tuveshd'	225	526	0.427756653992395	1.00570342205323
b'tv cable'	173	2000	0.0865	4.0065

5. **Top 10 driving URLs:** Here we will show a table with the top 10 URLs for the time period. They will be ordered by clicks.

The charts look like this:

Top 10 Driving URLs				
Keywords	Clicks	Impressions	CTR	Ranking
b'https://www.tuves.cl/'	8707	44036	0.197724588972659	5.36395222091016
b'https://www.tuves.cl/ayuda/te-llamamos/'	1083	19234	0.0563065405011958	2.27430591660601
b'https://www.tuves.cl/entretenimiento/guia-de-canales/'	373	14946	0.0249565101030376	2.54643382844908
b'https://www.tuves.cl/entretenimiento/plan-y-kit/arrenda-tu-kit/'	296	19296	0.0153399668325041	2.20434286898839
b'https://www.tuves.cl/ayuda/centros-de-pago/'	270	9031	0.0298970213708338	2.62839109733141
b'https://www.tuves.cl/entretenimiento/plan-y-kit/compra-tu-kit/'	231	21045	0.0109764789736279	2.09394155381326
b'https://www.tuves.cl/ayuda/preguntas-frecuentes/'	195	9864	0.0197688564476886	3.52716950527169
b'https://www.tuves.cl/entretenimiento/canales-play/'	141	17018	0.00828534492889881	3.23369373604419
b'https://www.tuves.cl/entretenimiento/guia-de-programacion/'	108	12813	0.00842893935846406	9.72574728791072
b'https://www.tuves.cl/ayuda/factibilidad-tecnica/'	107	13324	0.0080306214350045	2.95737015911138

The final result looks like this:



The screenshot shows the Tuves HD dashboard. On the left is a sidebar with navigation options: Desktop, Tablet, All traffic, All Top Keywords & Url, Rankings, and Saved Reports (Current month, Last quarter, Social engagement, Year-end sale). The main content area displays a table of traffic data for three keywords: 'b'tuves hd planes', 'b'tuveshd', and 'b'tv cable'. Below this is a section titled 'Top 10 Driving URLs' with a table listing the top 10 URLs, their clicks, impressions, CTR, and ranking.

Keywords	Clicks	Impressions	CTR	Ranking
b'tuves hd planes'	230	312	0.737179487179487	1
b'tuveshd'	225	526	0.427756653992395	1.00570342205323
b'tv cable'	173	2000	0.0865	4.0065

Keywords	Clicks	Impressions	CTR	Ranking
b'https://www.tuves.cl/'	6707	44036	0.197724588972659	5.36395222091016
b'https://www.tuves.cl/ayuda/te-llamamos/'	1083	19234	0.0563065405011958	2.27430691660601
b'https://www.tuves.cl/entretenimiento/guia-de-canales/'	373	14946	0.0249565101030376	2.54643382844908
b'https://www.tuves.cl/entretenimiento/plan-y-kit/arrenda-tu-kit/'	296	19296	0.0153399668325041	2.20434286898839
b'https://www.tuves.cl/ayuda/centros-de-pago/'	270	9031	0.0298970213706338	2.62839109733141
b'https://www.tuves.cl/entretenimiento/plan-y-kit/compra-tu-kit/'	231	21045	0.0109764789736279	2.09394155381326
b'https://www.tuves.cl/ayuda/preguntas-frecuentes/'	195	9864	0.0197688564476886	3.52716950527169
b'https://www.tuves.cl/entretenimiento/canales-play/'	141	17018	0.00828534492889881	3.23369373604419
b'https://www.tuves.cl/entretenimiento/guia-de-programacion/'	108	12813	0.00842893935846406	9.72574728791072
b'https://www.tuves.cl/ayuda/factibilidad-tecnica/'	107	13324	0.0080306214350045	2.95737015911138

To see the final result live you can go to <https://www.metacrawler.es/dashboard>

### 3.5.2 React App Structure:

Our React app will have the following file structure:

```

/src
├── charts
│   ├── chartjs.js
│   ├── ctr.js
│   └── rank.js
├── components
│   ├── Chart.js
│   └── Chartbar.js
├── dashboard
│   ├── Content.js
│   ├── dashboard.css
│   ├── dashboard.js
│   ├── nav.js
│   └── sidebar.js
├── tables
│   ├── table1.js
│   └── table2.js
├── topkpi
│   ├── top.css
│   └── top.js
├── App.css
└── App.js

```

- └─ App.test.js
- └─ index.css
- └─ index.js
- └─ logo.svg

## Chapter 4

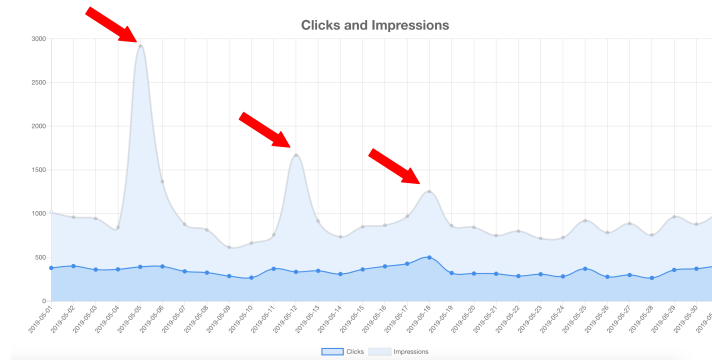
# Data Exploration

In this chapter we will explore our data using our dashboard. We will give some examples of what can we use this dashboard for and what insight we can detect. We will only do the data Exploration for Mobile.

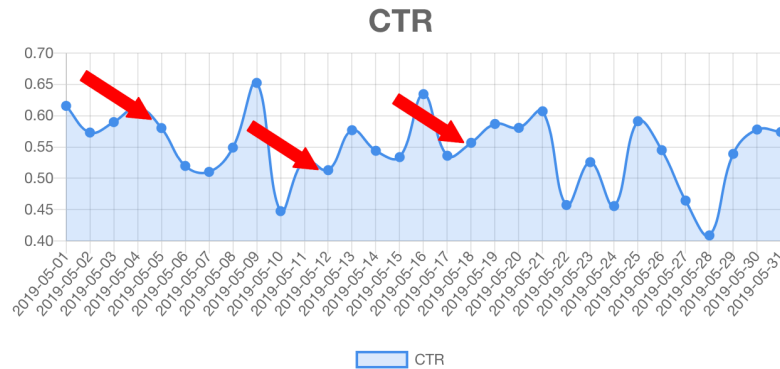
### 4.1 Mobile

#### 4.1.1 Example 1

If we look at the mobile impressions, we can see that there are high spikes on 3 different dates: 2019-05-05, 2019-05-12 and 2019-05-18. When we look at clicks, we don't see the same spikes in clicks. What can this tell us? This could mean that maybe Google started ranking us for a high volume keyword but the keyword they ranked us was not very related to our business, because we can see that the clicks are stable and didn't spike up like the impressions. We can also check if the Click through rate from those dates went up.



To double check that our theory is correct, we can check if the click through rate(CTR) went up on those dates.




We clearly see that there were no spikes on the given dates. This gives us a more clear understanding that these spikes didn't produce any traffic to our site. A good addition to the our dashboard would be the ability to check keywords by date, so that we could go into those specific dates and see which keywords had this spike in impressions and see if we could optimize our URL that target that keyword to get users to click into our result.

#### 4.1.2 Example 2

When we take a look at the table Top 10 Driving Search Queries we can see the top performers in the selected period of time. For example we can see that the top performer was the branded keyword '**tuves hd**'. A branded keyword is a keyword that includes the brand name[13]. We can see that this keyword had a very high click through rate and is the average rank is 1.0. If we go to Google and check this, we should be able to see our url <https://www.tuves.cl> in the number one position:

Google **tuves hd** Volume: 6,600/mo | CPC: \$0.20 | Competition: 0.15

SEOquake 

**Tuves HD - TV Satelital HD prepago**  
<https://www.tuves.cl/> [Translate this page](#)  
 Televisión Satelital HD con la mejor señal para todo Chile con Pausar y Grabar la Televisión en vivo.

**Pagos**  
 Consulta y pago de servicio prepago  
 TuVes.

**Te Llamamos**  
 Te Llamamos. Deje sus datos y  
 mensaje y nosotros lo ...

**Guía de Canales**  
 Mira los mejores canales de  
 Deportes, Cine, Infantiles ...

**Cómpralo**  
 Agregar Todo FOX \$11.990. Agregar  
 FOX Sports Premium ...

**Factibilidad Técnica**  
 Factibilidad Técnica de Tuves HD.  
 ORDEN DE DESPLIEGUE ...

**Centros De Pagos**  
 Dónde Puedes pagar Tuves HD?  
 Encuentra todo los centros de ...

[More results from tuves.cl »](#)

**TuVes HD - Wikipedia, la enciclopedia libre**  
[https://es.wikipedia.org/wiki/TuVes\\_HD](https://es.wikipedia.org/wiki/TuVes_HD) [Translate this page](#)  
 Tú Ves S.A, también conocido como TuVes HD es un proveedor internacional de televisión satelital en  
 alta definición que transmite cientos de señales digitales, ...

Tipo: **Sociedad Anónima** Industria: Telecomunicación; Televisión satelital  
 Personas clave: Eduardo Stigol, Presidente; K... Fundador(es): Konrad Burchardt

[Historia](#) · [Información técnica](#) · [Telepuerto](#) · [Servicios en otros países](#)

**Tuves HD | Servicio Satelital Mayorista y Transporte de Señales**  
<https://www.tuves.com/> [Translate this page](#)  
 TuVes S.A es una empresa de televisión satelital chilena que se dedica principalmente a la venta  
 mayorista de DTH marca blanca, entre otros servicios.

**TuVes HD SA: Private Company Information - Bloomberg**  
<https://www.bloomberg.com/research/stocks/private/snapshot.asp?privcapid...>  
 TuVes HD SA company research & investing information. Find executives and the latest company news.

This table help us monitor the top drivers of our month without us having to go to Google and type every keyword and check the rank. It also allow us to monitor the top drivers in terms of URL.

## Chapter 5

# Results and Conclusions

To see the results of our app, we can go to <https://www.metacrawler.es/dashboard>. This is only a vanity URL.

The whole code it can be found at <https://github.com/sundios/SEO-Dashboard>.

Our first version of our app is to be able to have the most basic and important metrics in a user friendly dashboard to help the monthly reporting for SEO managers. I achieve my goal and was able to create each step of the pipeline and display it into a dashboard. The SEO managers were able to use this tool to find important insights and to have a basic reporting dashboard to share with important stakeholders around the business.

This is my first attempt to create a ETL pipeline and would love to receive feedback in what things I could do different, what can I improve, which tools to use or ideas in what can I add or change. I would also like to get a honest review in if this is considered something useful in business or in data engineering. Overall it was a very interesting project to work on I learn a lot about many different technologies that I had not have the chance to work with and it helped me to understand visually how everything works together.

Our next steps for this tool is to include more important metrics. Metrics like Keywords base on specific dates or other important SEO metrics. We would also like to include some machine learning algorithms that will help SEO Manager find more important things. I was experimenting with K-Means clustering using the URL, CTR and Impressions. Basically I would like to cluster URLs depending on the amount of impressions and CTR this way we can detect which URLs need attention and should be optimized. URLs that had high CTR and high impression will be cluster into one cluster. URLs that had low CTR but very high impressions will be cluster in one cluster. And these URLs will be the ones I would need to focus on optimizing to increase the CTR.



I Hope you enjoy my report and my project.

## Chapter 6

# References

- [1] “TV Satelital HD Prepago.” Tuves HD, [www.tuves.cl/](http://www.tuves.cl/).
- [2] “Search Engine Users.” Pew Research Center: Internet, Science and Tech, Pew Research Center: Internet, Science and Tech, 8 Apr. 2014, [www.pewinternet.org/2005/01/23/search-engine-users/](http://www.pewinternet.org/2005/01/23/search-engine-users/).
- [3] ”How Search Works - Indexing, Google”, [www.google.com/search/howsearchworks/crawling-indexing/](http://www.google.com/search/howsearchworks/crawling-indexing/).
- [4] “Search Console API — Google Developers.” Google , Google, [developers.google.com/webmaster-tools/search-console-api-original/](https://developers.google.com/webmaster-tools/search-console-api-original/).
- [5] MAMP GmbH. “MAMP and MAMP PRO - Die Lokale Webserver Entwicklungsumgebung Für PHP - Und WordPress-Entwicklung.” MAMP and MAMP PRO - Ihre Lokale Entwicklungsumgebung, MAMP GmbH, [www.mamp.info/de/](http://www.mamp.info/de/).
- [6] Doerrfeld, Bill. “The Benefits of Using JSON API — Nordic APIs —.” Nordic APIs, 22 Nov. 2017, [nordicapis.com/the-benefits-of-using-json-api/](http://nordicapis.com/the-benefits-of-using-json-api/).
- [7] Foundation, Node.js. “About.” Node.js, Node.js Foundation, [nodejs.org/en/about/](http://nodejs.org/en/about/).
- [8] “React – A JavaScript Library for Building User Interfaces.” – A JavaScript Library for Building User Interfaces, [reactjs.org/](http://reactjs.org/).
- [9] Chart.js. “Chart.js.” Chart.js · Chart.js Documentation, ChartJS, [www.chartjs.org/docs/latest/](http://www.chartjs.org/docs/latest/).

[10] Axios. “Axios/Axios.” GitHub, 1 June 2019, [github.com/axios/axios](https://github.com/axios/axios).

[11] Otto, Mark, and Jacob Thornton. “Bootstrap.” · The Most Popular HTML, CSS, and JS Library in the World., [getbootstrap.com/](https://getbootstrap.com/).

[12]Reactstrap. “Reactstrap/Reactstrap.” GitHub, 13 June 2019, [github.com/reactstrap/reactstrap](https://github.com/reactstrap/reactstrap).

[13] “What Are Branded Keywords? - Define Branded Keywords.” Brick Marketing - SEO Marketing Solutions Company, [www.brickmarketing.com/define-branded-keywords.htm](http://www.brickmarketing.com/define-branded-keywords.htm).