University of Windsor

Amazon Keyword Visibility Tracker

Prof: Dr. Shafaq Khan

Team: Avengers

Tanvi Madaan (110063576)
Asad (110083660)
Varunkumar Patel (110077751)
Deepanshu (110086311)

Abstract

Amazon Keyword Visibility Tracker is an intelligent, highly customizable, and flexible tool that allows the clients using this tracker to keep track of their products listed on Amazon and their relative position to the products of their competitors. This tracker automates otherwise tedious manual tasks in an efficient manner by automating and mimicking the human approach to manual comparison of the products that the client owns versus the competition. This tracker could also second as a bot that can scrape data and maintain trends of the product listed in the same category as our client.

1. Introduction

1.1 Purpose

The ecommerce industry has been around and has become the norm for consumer shopping over the past two decades. The relationship between the platform and a business is a mutually beneficial one but in order for businesses to grow, whether large, medium or smaller businesses, they need to adapt to the market strategy to keep up with the competition. Ecommerce giants have made it easy for anyone to sign up and start selling on their platforms although taking a cut to list the product on the platform itself. This makes the competition to generate the maximum profit by keeping their products on the top shelf a major concern for companies and this is where the “Amazon Keyword Visibility Tracker” comes in. Businesses want their products to have more discoverability and buyability[1] amongst the customers. This tool will provide a detailed report of how their products are performing when searched on a specific keyword on amazon.

1.2 Problem Statement

There is a need for a software solution that provides the flexibility to the users(clients) to scrape information of as many products they want, including their own products as well as competitors. On top of the crawled data, the analysis on that is required.

1.3 Intended Audience & Use

The Tracker Engine is a multi faceted tool which can be used for small to medium scale businesses with the intention of tracking, automating and extracting live raw data from the real time product feed of Amazon, to store into the database. Finally the stored raw data can be leveraged by the client in any shape or form that they want. This raw data can also serve as a historical record on which the client can run prediction analysis. The scope of this project does not just end at data retrieval but stretches to data analytics and real time decision making.

This engine will be deployed and serve as a service for clients. Clients with products totaling 1 to any number amazon allows can use this engine as a service. They will update the input file which the engine will read, with the list of SKU (Stock keeping unit) which is an unique ID assigned to their products on amazon, and the keywords they want to analyze.

1.4 Scope

With the ecommerce dataspace increasing, a big volume of data is being generated every day. This engine will leverage this data to get useful insights for the clients. Clients can analyze this processed data to make required decisions for their products to perform better on Amazon i.e having maximum reach to the potential customers and increasing sales.

1.5 Methodology

The approach used in this solution is to take user input which includes the keywords they want to target for their products. Scrape top results data of those keywords from amazon, and save it into our database. After post-processing on the data, providing them with results which will give them ideas about their performance. A potential future use of this project might be to build a customer facing UI around this data source accumulated with the help of this tool over time. Most pre-processing and some post processing will be done in the database itself, thereby reducing the overhead cost of bringing the data to a middleware server where it is being processed and then being led to and from across multiple endpoints.

1.6 Results & Conclusions

The Retrospective analysis of using this tool will most likely be the findings that it is not the tool itself that is of long term value but instead the historical and real-time data that this tool collects in the form of a data store. This data can be used to predict brand analysis and the occurrences of a competitor's product in the sponsored sections of the website. The correlation between the occurings of the product on certain positions of the platform and an occasion/sale/festive season might help a client establish a pattern/rhythm on which a competitor spends their advertising money. This gives the company using this tool the means to target the competition in a smart manner. This tool levels the playing field for smaller businesses to compete with the bigger companies.

2. Background Study

2.1 Existing solutions

We did a thorough market research to find if there are any existing solutions that justifies the problem statement. We found multiple existing solutions that were somehow relevant, and narrowed down our results to the best three solutions that have high relevance to our problem statement.

2.2 Amazon keyword analytics

This is an inbuilt KPI dashboard[2] provided by amazon to the clients that sell on amazon.
It shows the performance of an individual sku by keeping its historical data.
Also it assigns a rank to the product in a particular category which manifests how that product is performing in that particular category on Amazon.

Limitations

You can't analyze your competitor's product's performance. The premium features which include historical performance of the product is a paid service.

2.3 Parsehub

Parsehub[3] is a free and powerful web scraping tool. You can scrape any site by just selecting the html tags you want.
This tool can be used to scrape the amazon keyword data, upto desired number of pages.

Limitations

There is no database, so you can't store the historical data. It needs to be run every time for different keywords, so it helps but there is still a lot of manual effort required.

2.4 Sonar Amazon Keyword Tool

Selics Sonar[4] is a very powerful tool, and it satisfies our problem statement to much extent. It takes input from the user including keywords and pages to crawl. It provides KPi indicators that help clients analyze their performance.

Limitations

If the product searched is not in the sonar's

database, then it gives vague results. Also it doesn't save historical data of the keywords scraped so you can have data older than 15 days.

3. Proposed Model

3.1 Product Perspective

As we move ahead with the development of this engine, we used multiple tools and technologies.
To understand the significance of every tool and technology used, we first have a look at the functional requirements and then the architecture.

3.2 Functional Requirements

- System takes input from the client from the input sheet, which must include the SKU(product Id), the keyword they want to track, and the number of pages they want to track (optional).
- System must scrape every keyword once. (If there are two different SKUs but mapped with the same keyword, then the system should only scrape that keyword once, and use the same data for those different SKUs).
- System must scrape data from Amazon without getting the IP blocked.
- System must store the scraped Information after post processing, to the main database in PostgreSQL.
- System must store the historical data (There must be a timestamp tagged with data scraped everytime.)

  Example- ABC SKU is scraped for the last 10 days for a keyword xyz. There must be scraped data results for those 10 days for that keyword- SKU map in the database.
- System must generate data insights from the historical data stored in the database, and generate resulting excel sheets.

3.3 Architecture

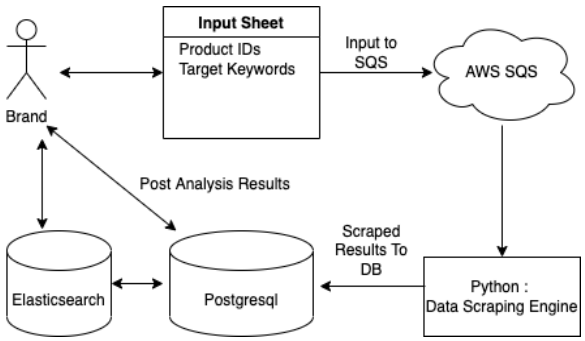

Image 1: Architecture Model

Breakdown of Components:

:- Input Sheet
:- AWS SQS
:- Data Scraping Engine
:- PostgreSQL

:- INPUT SHEET

- Clients will be given an instruction sheet about how to fill the input excel sheet.
- The input excel sheet will take input parameters which include the keyword to scrape, the skus to target including self and competitor.
- The excel file will have n number of sheets (n= number of target keywords)
- **SKU**: The stock keeping unit ID assigned to the product by Amazon
- **Keyword**: The keyword client wants to search for that product

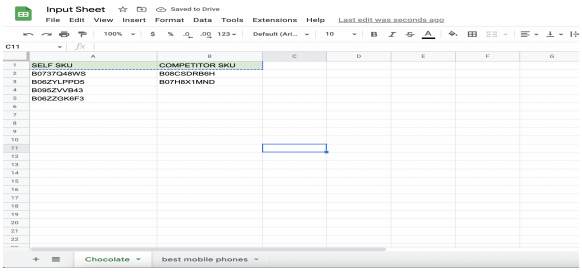

Image 2: Input Sheet

:- AWS SQS

- SQS- Simple Queue Service[5] is a volatile database that enables you to decouple and scale your solution.
- It enables you to insert, update, delete the messages which are read by multiple instances at a time.
- It is a perfect example of a distributed system, where the messages are stored in sites in different regions

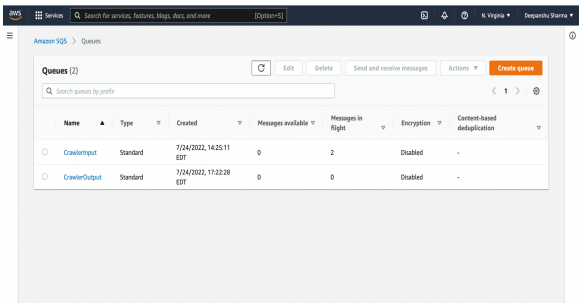

Image 3: AWS Sequence Queuing Service

:- SCRAPING ENGINE

- This is the core engine which scrapes the data from the amazon site.
- Python is used to scrape the products info which includes ranking from the amazon keyword results page.

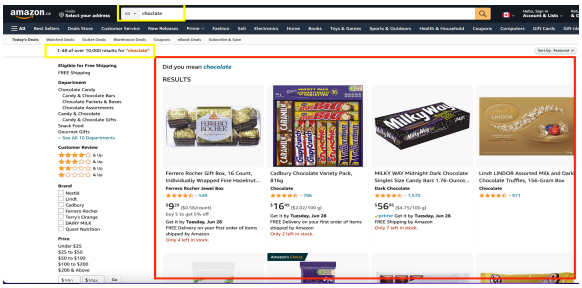

Image 4: Actual View Of Data Points Scraped

- The red highlighted area shows the results that are scraped for the keyword highlighted in the yellow box.
- The scraper will put the input keyword which is in yellow on the amazon site and scrape the results quickly in the red box.
- After scraping these results are pre-processed and are inserted into the database.

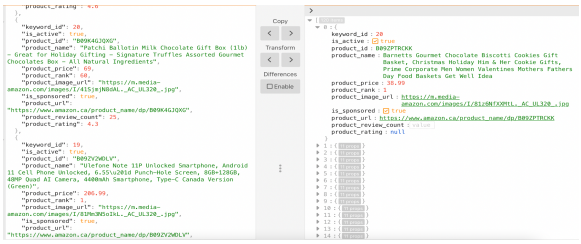

Image 5: Crawled Data in JSON

:- POSTGRESQL

The scraped results are inserted into the database.The schema We are using the PostgreSql database for data storage. It has a very efficient and reliable handling support of JSON type of data[6].

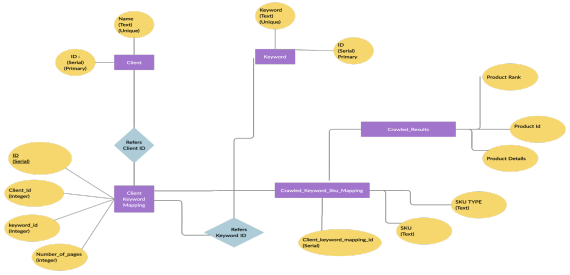

Image 6: Database Design View

```
-- Table: ADT_Master.Keyword

-- DROP TABLE IF EXISTS "ADT_Master"."Keyword";

CREATE TABLE IF NOT EXISTS "ADT_Master"."Keyword"
(
    id integer NOT NULL DEFAULT nextval('"ADT_Master"."Keyword_id_seq"'::regclass),
    keyword text COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Keyword_pkey" PRIMARY KEY (keyword)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "ADT_Master"."Keyword"
    OWNER to postgres;
```

Image 7: Master Keyword Table

```
-- Table: ADT_Master.Client_Keyword_Mapping

-- DROP TABLE IF EXISTS "ADT_Master"."Client_Keyword_Mapping";

CREATE TABLE IF NOT EXISTS "ADT_Master"."Client_Keyword_Mapping"
(
    id integer NOT NULL DEFAULT nextval('"ADT_Master"."Client_Keyword_Mapping_id_seq"'::regclass),
    client_id integer NOT NULL,
    keyword_id integer NOT NULL,
    CONSTRAINT "Unique row" UNIQUE (client_id, keyword_id)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "ADT_Master"."Client_Keyword_Mapping"
    OWNER to postgres;
```

Image 8: Master- Client Keyword Mapping Table

```
-- Table: ADT_Master.Client_Keyword_Sku_Mapping

-- DROP TABLE IF EXISTS "ADT_Master"."Client_Keyword_Sku_Mapping";

CREATE TABLE IF NOT EXISTS "ADT_Master"."Client_Keyword_Sku_Mapping"
(
    id integer NOT NULL DEFAULT nextval('"ADT_Master"."Client_Keyword_Sku_Mapping_id_seq"'::regclass),
    client_id integer NOT NULL,
    keyword_id integer NOT NULL,
    sku text COLLATE pg_catalog."default" NOT NULL,
    is_comp boolean NOT NULL DEFAULT false,
    CONSTRAINT "Unique Row" UNIQUE (client_id, keyword_id, sku)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "ADT_Master"."Client_Keyword_Sku_Mapping"
    OWNER to postgres;
```

Image 9: Master- Client Keyword Sku Mapping Table

```
-- Table: ADT_Master.Crawled_Keyword_Data

-- DROP TABLE IF EXISTS "ADT_Master"."Crawled_Keyword_Data";

CREATE TABLE IF NOT EXISTS "ADT_Master"."Crawled_Keyword_Data"
(
    id integer NOT NULL DEFAULT nextval('"ADT_Master"."Crawled_Keyword_Data_id_seq"'::regclass),
    keyword_id integer NOT NULL,
    crawl_date date,
    is_active boolean,
    product_id text COLLATE pg_catalog."default",
    product_name text COLLATE pg_catalog."default",
    product_rank integer,
    product_image_url text COLLATE pg_catalog."default",
    is_sponsored boolean,
    product_url text COLLATE pg_catalog."default",
    product_review_count integer,
    product_rating double precision,
    creation_timestamp timestamp without time zone,
    product_price double precision
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "ADT_Master"."Crawled_Keyword_Data"
    OWNER to postgres;
```

Image 11: Master- Crawled Keyword Data Table

3.4 TOOLS & TECHNOLOGIES:

Tools: PyCharm, Visual Code Studio, PgAdmin 4, Postman

Technologies: Python, AWS SQS, Pandas, PostgreSQL, JSON

4. Limitations & Challenges

1. Everyone in the team was new to the postgres technology, so implementing the database design and queries was a good learning challenge.
2. Scraping data from amazon was very difficult, as it runs a bot detection algorithm which will detect that you are running bots on their sites and will block your IP.
3. Integration of AWS component SQS with almost no prior knowledge was a very challenging task, and for that we need to go through its official documentation and online video tutorials.

5. Future Work

1. To integrate a high end front end dashboard at which the client can give the input and check the results at.
2. Adding machine learning algorithms to forecast the performance of products.
3. Fragmentation of databases to support big data volume.
4. Adding elastic search for fast selection results at the front end.

6. References

[1] “How Discoverability and Buyability can drive success on Amazon,” DigitalStackAI. [Online]. Available: https://1digitalstack.ai/how-discoverability-and-buyability-can-drive-success-on-amazon/. [Accessed: 05-Jun-2022].

[2] “Amazon Keyword Analytics”, Advertising.Amazon.com [Online].

Available: https://advertising.amazon.com/en-ca/solutions/products/sponsored-products?ref_=a20m _us_hnav_p_sp. [Accessed: 07-Jun-2022].
[3] “Parsehub”, Parsehub.com [Online]. Available: https://www.parsehub.com/. [Accessed: 11-Jun-2022]
[4]“SonarSelics”,sellics.com [Online]. Available: https://sellics.com/sonar-amazon-keyword-tool/. [Accessed: 11-Jun-2022]
[5] “AWS SQS Like never before”, AWS [Online]. Available: https://aws.amazon.com/message-queue/benefits/. [Accessed: 15-Jun-2022]
[6] “Unleash the power of json in Postgre”, Cloudbees.com [Online]. Available: https://www.cloudbees.com/blog/unleash-the-power-of-storing-json-in-postgres. [Accessed: 15-Jun-2022]
[7] “Amazon XML Parsing” [Online]. Available https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.133.1056&rep=rep1&type=pdf [Accessed: 22-July-2022]