SEARCH ENGINE DATABASES

[1]Archana Verma, [2]Prof(Dr.) Ajay Rana

[1]Head, Information Technology Department, IEC Group of Institutions, Greater Noida, U.P.(India)
[2]Program Director, Amity School of Engineering and Technology, Amity University, Noida, U.P. (India)

ABSTRACT:

Search engines are an integral part of the internet. They are used by millions of users across the world to search information. We take a brief look at how the databases of search engines are organized. And also look at the databases of top 3 search engines.

Keywords: *Google, Yahoo, AOL, BigTable, Vertica*

I. INTRODUCTION

A search engine handles more than 3 billion documents using close to 10 TB of data and serving more than 150 million queries per day where peak queries become of the order of thousands of queries per second. The search engine should be taking care of the following issues:

1) Tracking millions of words in billions of documents according to their position and relative importance of every word.
2) No of words in a query are less which results in larger no of hits and ranking of these hits is a challenge.
3) They should be highly available.
4) They should always be fresh and obsolete data should not exist.

A DBMS cannot be used in a search engine for the reason that it is very slow. A DBMS generally relies upon the ACID properties and is supposed to be consistent and durable.

A search engine has a higher priority of availability over consistency. Also Isolation is not really required in search engines as the updates are few and most transactions are read only. This implies that we need a static database to serve all of the read only queries and a large degree of offline work to build and rebuild the static databases. The offline work normally involves indexing.

2 HOW DOCUMENTS ARE SEARCHED

The first step is to Crawl the documents which means going to the web pages and accumulating pages from thousands of servers at the same time.

The next step is of the indexer which interprets collections of documents to produce a static database called a chunk. The online servers are free to process query whereas the offline indexer does the scoring and generating normalized score for every word in every document.

The third step is of the server to simply execute queries against a collection of chunks. The steps involved are query parsing, query optimization and query execution.

The only update is to replace the chunk; there is no need for concurrency control technique and isolation of transactions.

3 HOW QUERIES ARE HANDLED

A query to retrieve a document contains two parameters on which a match is made. One is property and the other is word. Property needs an exact match whereas the word needs a score on which to fetch the document. The property has a Boolean value either true or false. Example of properties are:

lang: English means document is an English language

cont: picture means contains a picture

A query term is a property or a word. Complex queries include query terms connected by AND, OR, NOT expressions

Query Q = { w1, w2, ……wk}

The score of a document d for query d is the sum of an overall score for the document and a score for each term in the query

$$Score(Q, d) = Quality(d) + \sum score(wi, d)$$

The score for each word is determined at index time and depends on frequency and location (such as in the title or headings, or bold).

The database lends it abstract principle of logical query plan to the search engines in order to execute their queries.

The general schema for the databases is:

Document Table

Doc ID	URL	Data	Size	Abstract

Word Table

Word Id	Doc ID	Score	Position Info

Property Table

Word ID	Doc ID

Term Table

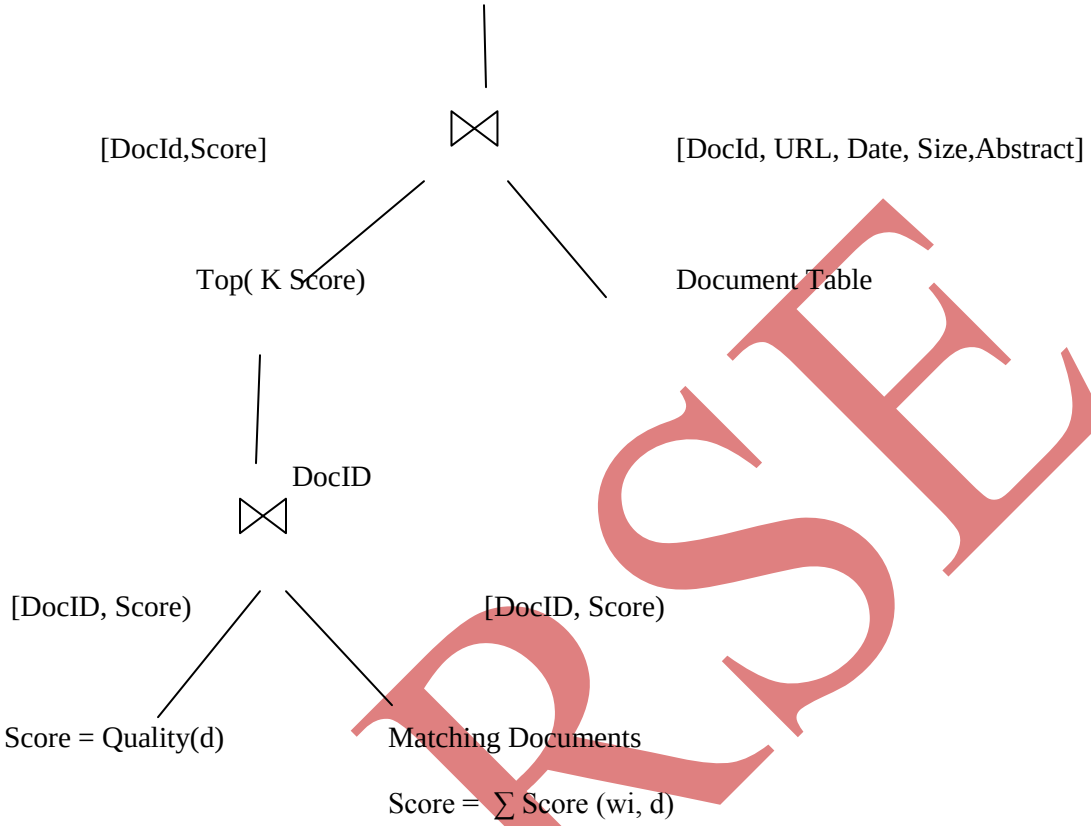
String	Word ID	Stats

The WordID is an integer key for each word, the term table keeps statistics about the word (or property). The stats are used for scoring and to compute the selectivity for query optimization. One way of determining the stat is to count the number of terms in the rows of tables in the corpus which will give an estimate of how common the term is. High counts means high selectivity and lower scores imply that the word is common.

The document table stores all the details of the document. The word table stores the details of the word or property in the document. The property table stores which word is present in which document.

The quality is first found which determines parameters like popularity, incoming links , quality of the containing site and external reviews. Then the score is found on the matching documents which mainly capture frequency and location of the word. These are equijoined on the DocID. And top k items from the input set are taken to equijoin with the DocID of the document table to give the result.

RESULT = { DocID, Score, URL, Date, Size, Abstract}



I

4 GOOGLE'S DATABASE

Google Uses BigTable which is a distributed storage system for managing structured data that is designed to scale to a very large size PetaByte of data across thousands of commodity servers.

BigTable is not a relational database. It does not support joins nor does it support rich SQL like queries, each table is a multi dimensional sparse map. Tables consist of rows and columns and each cell has a timestamp. There can be multiple versions of a cell with different time stamps. The time stamp allows for operation such as "Select n version of this web page" or "delete cells that are older than a specific date/time".

BigTable splits tables at row boundaries and saves them as tablets. A table is around 200MB and each machine saves about 100 tablets. The tablets from a single table are spread among many servers. If one table is receiving many queries, it can shed load to other machines which are not so busy.

BigTable is built on Google File System (GFS) which is used as backing store or log and data files.

There are 3 primary types of servers in the BigTable System:

1) Master Server: Responsible for assigning tablets to tablet servers and maintain a list of where each tablet is located and also for load redistribution among the tablets.

2) Tablet Servers: All read write queries are handled by these servers. They also split themselves when their size limit is exceeding. If a tablet server fails then they have a recovery method to pick up a new tablet server.
3) Lock Servers: These servers are required to maintain locks at the time when tablets are opened for writing

5 YAHOO'S DATABASE

The data yahoo gathers is structured data, as opposed to unstructured data like e-mails and other documents, it is about how people use our web site both from the advertising perspective and from consumer experience perspective.
Yahoo's database is built out of commodity intel boxes, strung together in large clusters. The classic industry approach has been to go for big SMP (symmetric multiprocessing) boxes.

Yahoo started with the Postgre SQL engine and replaced the query processing layer with code design for its commodity hardware cluster.
SQL database are organized as tables, which consist of rows and columns, they are traditionally arranged as rows of data, but yahoo chose to store its data as distributed columns.
The choice to organize it as column helps in deep analytics queries and that you go into the data that interests you, which makes it very effective in terms of reducing the amount of data you have to move through for a particular query.

6 AOL'S DATABASE

AOL uses Vertica which was selected to replace MySQL as the database of choice for delivering online data sets and reporting applications. As these data sets have grown, the company was running into performance problems including data sharding and difficulty replicating information across multiple servers.

Vertica is data warehousing tool. It helps to support robust reporting and data consumption whether its through front end web services, APIs or direct SQL queries against Vertica Boxes.

Vertica's column database supports massively parallel processing (MPP) commodity hardware. The column store approach – an architecture shared by competitors maximizes data compression and speeds analytics queries that typically interrogate only selected dimensions across rows of data. MPP further enhances performance by spreading workloads across tens, hundreds or even thousands of nodes.

7 CONCLUSION

Full text search engines came after a long time since the invent of traditional database engines. Search engines mostly use unstructured data which is unable to fit into traditional table oriented style of databases. Many search engines still rely upon the table structure format of databases as their underlying base to build records on top of that.

Whereas Google's Big Table database also used commodity hardware clusters, but yahoo's approach differs in that it is designed for SQL interface. Typically with Big Table you would be writing programs with C++ or Java programs. Whereas Yahoo would be able to do the same work using SQL.

Both of them use Petabyte of data. A Petabyte equals one million gigabyte. AOL has the capability to serve upto 10- 15 Terabyte of data.

REFERENCES

[1] Eric, A. Brewer, "*Combining Systems and Databases: A Search Engine Retrospective*", Database Systems, Fourth Edition, 2004.

[2] Mark Bennett, "*Contrasting Relational Databases and Full Text Search Engines*", New Idea Engineering, Inc. Issue 9, June 2004.

[3] Sources from Internet.