**Script Overview**

The Hive script provides information’s on the total number of requests or essentially the number of users who has accessed the company’s website and viewed the services over a specified timeframe. The data used for analysis is basically present in the form of web server logs. Each data within the file provides an extended information of a single user request. The log details are too huge to be imported into the SQL database as they are not in a relational format. These large-scale data can be analysed with Hive which provides accessible and effective architecture.

It is very much essential for the data to be converted for the Hive to interact with it such that it can be translated into a more representable form as a database table. This transition is made possible with the aid of **SerDe**(serializer/deserializer) which comes in a variety of data formats. The script makes use of HiveQL which is almost alike SQL scripting language for analysis and data warehousing. A table with the name “**cloudfront\_logs**” is created upon validating and confirming that the same name doesn’t exist in the database. The parameters present within the HiveQL represents the information which are present in the log file converted to a much more identifiable format assigned to the respective data types. These parameters which depicts the attributes of the table helps in making valuable inferences when the query is executed. The hive gets the instructions to serialize/deserialize the data from the file using the class “**ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'**”. The row value read from the file is split up into different attributes for the hive table “**cloudfront\_logs**” using the property **“input.regex”** within the class RegexSerde. The log files which are used by the hive for the analysis is present in AWS S3.

**HiveQL Code Analysis**

The main objective here is to make necessary modifications in the available script to excavate on any potential business insights that might benefit the organisation to a great extent. The analysis done by invoking some rudimentary changes in the code brought a spectacular impact which can assist in obtaining relevant information’s from the log files particular to enhance the business strategy.

**Code Analysis1:** This analysis is done to figure out the number of users who has successfully viewed the company website from different operating systems and thereby concluding as to which OS performed the best and least. The necessary changes made in the code in provided below and business impact is explained with the help of a pivot chart for the ease of understanding.

**HiveQL:**

**CREATE** **EXTERNAL** **TABLE** **IF** **NOT** **EXISTS** cloudfront\_logs **(**

DateLog DATE,

Time STRING**,**

Location STRING**,**

Bytes INT,

RequestIP STRING**,**

Method STRING**,**

Host STRING,

Uri STRING,

Status INT,

Referrer STRING**,**

OS STRING**,**

Browser STRING**,**

BrowserVersion STRING

**)**

**ROW** **FORMAT** SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'

**WITH** SERDEPROPERTIES **(**

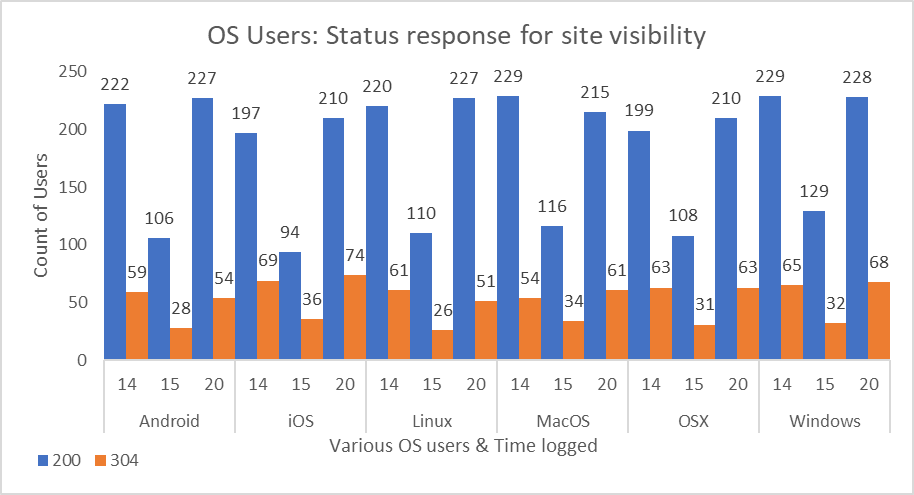
"input.regex" **=** "^(?!#)([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+[^\(]+[\(]([^\;]+).\*\%20([^\/]+)[\/](.\*)$"

**)** LOCATION '${INPUT}/cloudfront/data'**;**

-- Status of the requests from a particular OS at a given time frame.

**INSERT** OVERWRITE DIRECTORY '${OUTPUT}/os\_requests/' **SELECT** Time, Status**,**OS**,**COUNT(\*) **FROM** cloudfront\_logs **WHERE** DateLog **BETWEEN** '2014-07-05' **AND** '2014-08-05' **GROUP** **BY** Time, Status,OS**;**

**Inference:** The above code returns the selected values from the table **cloudfront\_logs** i.e, **Time, Status** and the **OS** which the users used to view the company website, grouped by the respective columns to get an ordered result. An extended analysis done with respect to the pivot chart created on extracting the results obtained from the query is provided below.

`

The graph created is to identify the number of users who were able to view the site successfully depending on the Status retrieved on each view from the server. As the legend in the graph shows, **“200”** depicts a success where the users from different OS could view the site and **“304”** depicts an idle/not modified state which can indicate that the user was inactive upon entering the site. Hence it can be observed from the plot that almost all the user who tried viewing the company website were successful and Android users stands ahead when compared to any other OS users and only a few inactive state were observed. It is also interesting to note that most of the users tried accessing the website at 14.00(time) from almost all the OS and were successful in viewing. Hence, if the company is planning to put up some offers or services, this would possibly be the right time.

**Code Analysis2:** The analysis done focuses mainly on understanding the volume of users from different locations accessing the company website at specific point of time. This would indeed help in acquiring knowledge about the areas whether the marketing strategy of the company was good enough or not and would provide enough insights on improving the strategies accordingly wherever needed.

**HiveQL:**

**CREATE** **EXTERNAL** **TABLE** **IF** **NOT** **EXISTS** cloudfront\_logs **(**

DateLog DATE,

Time STRING,

Location STRING,

Bytes INT,

RequestIP STRING,

Method STRING,

Host STRING,

Uri STRING,

Status INT,

Referrer STRING,

OS STRING,

Browser STRING**,**

BrowserVersion STRING

**)**

**ROW** **FORMAT** SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'

**WITH** SERDEPROPERTIES **(**

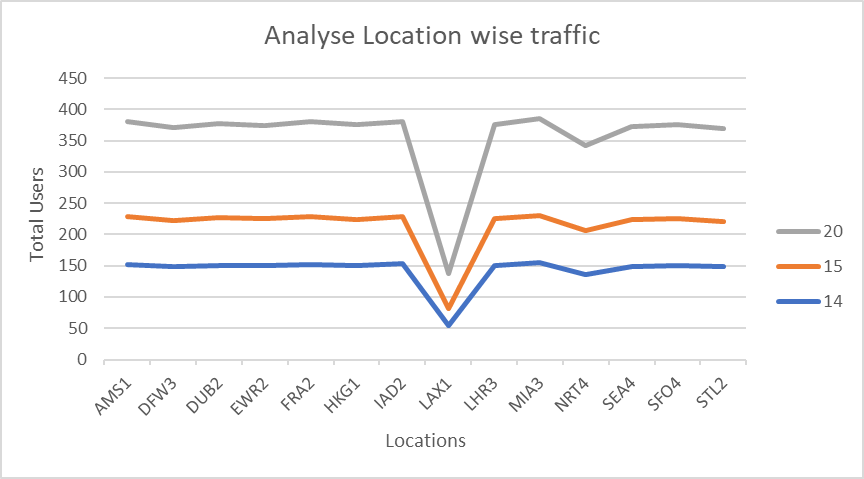
"input.regex" **=** "^(?!#)([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+[^\(]+[\(]([^\;]+).\*\%20([^\/]+)[\/](.\*)$"

**)** LOCATION '${INPUT}/cloudfront/data'**;**

-- Returns the location, time and total users accessed the site for a given time frame.

**INSERT** OVERWRITE DIRECTORY '${OUTPUT}/os\_requests/' **SELECT** Location,Time,COUNT(\*) **FROM** cloudfront\_logs **WHERE** DateLog **BETWEEN** '2014-07-05' **AND** '2014-08-05' **GROUP** **BY** Location**,**Time;

**Inference:** The above code returns the selected values from the table **cloudfront\_logs** i.e, **Location**, **Time** of the users who viewed the company website, grouped by the respective columns to get an ordered result. An extended analysis done with respect to the pivot chart created on extracting the results obtained from the query is provided below.



On closely observing the line graph, the average number of users in almost all the locations at different times (14:00,15:00,20:00) remains nearly constant with trivial deviations at certain areas. But an exception lies within the data which is evident on viewing the graph that a drastic decline is observed corresponding to the location LAX1 where in all the time frames considered the number of users has dropped down significantly. This could possibly be an issue which the users are facing in that region which has resulted in this dramatic variation. This might have a potential impact on the business of the company as the number of users could possibly be high in that region and the products and offers delivered through the site is evidently not reaching them.

**Log Management: Summary Report**

A cumulative process used to enable the transmission, analysis, storage and archiving of huge amount of log data formulated within an information system. Log data is automatically generated and time-stamped pertinent to a system. Furthermore, all the applications and systems produce log files. Log management as such comprises of various stages which covers the areas detailed below:-

1. Centralized log aggregation
2. Log collection
3. Log rotation
4. Log analysis
5. Log search and reporting.

In a nutshell logs are a series of unceasing digital records generated by a software stack which comprises of servers, networks, application, cloud infrastructure and so on.

**Importance of Log Management**

Very often people tend to get misguided understanding the rudimentary difference between having security logs and dynamically utilising them to monitor security related concerns within an organisation. For instance, the benefits of having system logged messages and security logs, following a breach would aid in the forensic investigations undoubtfully. But the relevance of actively reviewing and analysing the logs is of great importance as devoid of such a systematic process would be of little or no use to any management of information security. Hence, to take effective counter measure on incidents which questions the security systems it is of utmost importance for the security logs to be monitored and analysed.

One of the largest uses cases for big data solutions is none other than the Log management and it has become an inevitable part across organisations to tap into the log data and figure out critically relevant information’s. The concept of log management is far more than just collecting log files but rather includes centralising logs, storing the records for future validations and finally analysing the results to gain any relevant business insights. Eventually, it helps in improving operations within an organisation if monitored and tracked down efficiently. There are several domains in the organizations where this comes handy, a few of which are detailed below.

* **Solving Application Issues**

To a technical person within an organisation, log data are very beneficial as they help to drill down deep into the application related troubles and helps resolve them by tracking down the areas of low performance, spotting out the root cause of run-time errors and assess application troubleshooting.

* **Managing Resources**

IT systems makes use of log data to monitor across the systems to detect patterns and log events. One of the major ordeals which is faced in many applications or systems would be to identify the configuration or performance issues which can easily be tracked down with these data. This simplifies the process of performing root cause analysis when a failure occurs to further drill down into the data and gain proper insights.

* **Business Analytics**

Log data is nothing but a treasure trove of business insights. It is quite essential to gain specific business goals within a stipulated time frame which gets even more easier if proper analysis is done with respect to the log data available (e.g.: transactional data which provides information’s on number of transactions completed per second)

* **Centralisation**

It is practically impossible to manually run down through the log files of several hundreds of thousands of services which are running parallelly on multiple servers. Hence, a centralized log management with achieve this by aggregating them and store the logs which eventually allows the users to access, search and finally manage the logs from a single location.

Web Server Logs

App Server Logs

Buffer System

Log Server

Application Logs

Messaging Logs

* **Identifying and alerting issues**

Another useful characteristic of log management would be that it can easily be converted to a proactive aid, where the data can be monitored and scanned for irregularities or infrequent trends. In effect, this process can help resolve the issues way before they can even grow.

**Big-Data Analytics: Log Management**

As the word reciprocates, big data deals with data holding some unique or specific characteristic which separates it from other sources due to the variety, velocity, volume and veracity. The logs produced by the machine are predominantly enriched with significant information’s which can be excavated for various purposes that can significantly enhance the efficiency of any organisation. It should also be considered that the logs produced can have both useful and unserviceable information’s and without a proper log management system the data would become immense and unable to comprehend.

The apparent question which arises is none other than figuring out a potential solution to which is capable enough of collecting and analysing the terabytes of data easily. This brings about the importance of Big Data solutions to the picture.

**Architecture**

Business Intelligence

Apps

Apps

Apps

Web logs Data

Social Data

The Big Data solutions focuses mainly on uplifting the procedure for accumulation or aggregation of logs into a centralised source point. Beyond which the log files are converted to a single format which is a crucial step involved so that the data embedded within can be later analysed. The information’s are basically stored in NoSQL databases which is a distributed file system. This immediately brings out an impact as the loss of information contained in the log files gets diminished and thereby aids in retention of data. The Big data technologies such as Hadoop, Apache Spark constitutes to provide efficient methods to process the logs. This in turn helps in sorting out the relevant information’s and aids in efficiently analysing the data.

**Inference**

Well analysed log files can be used to monitor the applications better, gain better business insights and enhance customer intelligence, prevent fraudulence and tighten the security features of the entire system. However, it is quite essential to understand that the true power of log management cannot be explored without the right tools as this is basically a time-consuming process. The Big Data solutions provide a gateway for the organisations to monitor the whole hardware and software system through various analytic dashboard, visualization mediums and enables us to see the bigger picture of understanding the overall performance of all the systems. The benefits of custom reporting, obtaining the status of automated monitoring and eventually identifying real-time alerts can never be excluded.