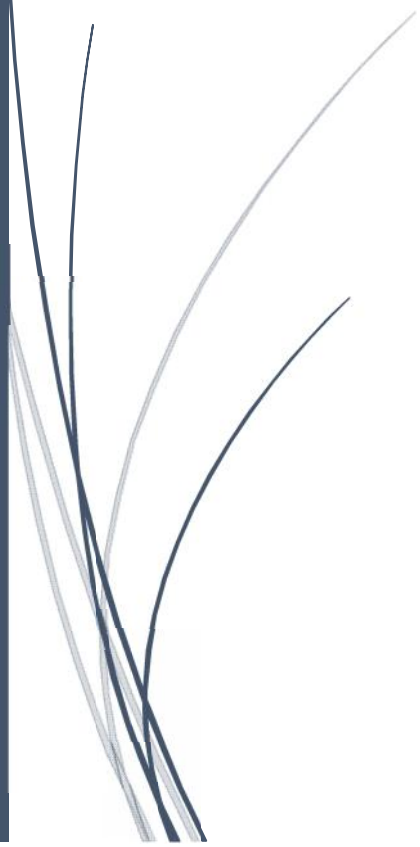
10-6-2017

TEC | Tecnológico de Costa Rica

Proyecto II

Hadoop



Estudiantes:

Valeria Garro Abarca

Florencio Solís Durán

Ariel Herrera Fernández

José Alvarado Chaves

Profesor:

Erick Hernández B.

Base de Datos II

I Semestre 2017

Tabla de Contenido

Tabla de Figuras

Introducción

Big data y el uso de bases de datos no SQL para el análisis de información es un enfoque importante en la actualidad, donde se trabaja con grandes niveles de información. Para esta tarea el objetivo planteado es tratar de hacer una implementación que conlleva mucho de esta parte, donde vamos a explorar el uso de hadoop y otras aplicaciones que nos ayuden a realizar trabajos de manejos importantes de información con un menor tiempo y en un paradigma distinto al habitual de bases de datos relacionales.

Explicación del Diseño

En este parte del documento explicaremos cómo está concedido el diseño de la solución, donde ampliaremos a detalle las principales partes que componen el proyecto.

Estructura General

La estructura general se puede ver como una serie de procesos seriales, donde iniciamos con un proceso twitter de lectura de datos, seguidos de una inserción en base de datos NoSQL (hbase). Luego se realiza mediante hadoop un análisis de la información con un único job. Seguido tenemos un web service que atiende peticiones para mostrar datos en una interfaz desarrollada con la biblioteca d3js trabajado en Angular.

Diseño del Aplicativo

El diseño del aplicativo muestra que se manejan conexiones entre los diferentes esquemas explicados anteriormente y donde se explicaran a detalle en las capas. Pero a grandes rasgos es un diseño integrado, que ejecuta algunas tareas secuenciales como las del análisis de los datos y la actualización en la base de datos relacional, donde utilizamos MySQL.

Arquitectura de la solución.

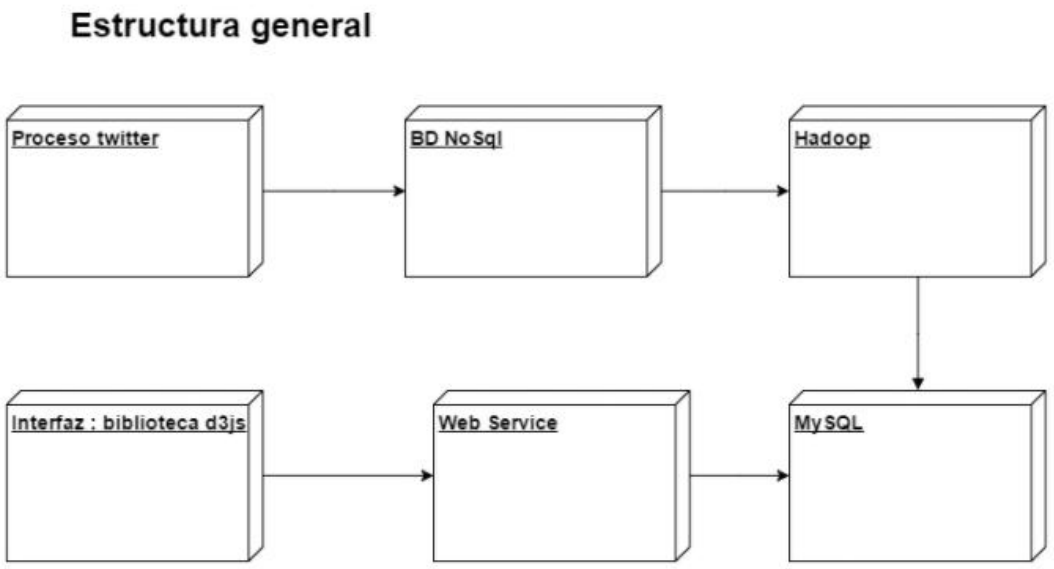


ILUSTRACIÓN 0-1

Diseño de la Arquitectura

El diseño de la arquitectura responde un conjunto de sistemas que se encargan de realizar tareas específicas que dan como salida la entrada para el siguiente sistema. En la Ilustración 2 se muestra este proceso serial.

ILUSTRACIÓN 0-2

Arquitectura

El segundo proyecto trata sobre analizar información generada por la red social twitter, analizando los datos con hadoop y presentándolos utilizando la herramienta visualizadora llamada d3js. Este proyecto lo podemos dividir en tres grandes partes.

1. Primeramente se tiene un proyecto en el cual se implementó una extracción utilizando una implementación del API de Twitter, en el cual bajamos la información necesaria buscando 8 diferentes hastags, por los cual obteníamos datos tales como los usuarios, los tweets, y las fechas de los mismos. Con esta información realizábamos una limpieza de los datos en el tweet donde dejábamos finalmente solo la información importante.
2. Un proyecto de hadoop donde se implementó map reduce para realizar análisis de los datos obtenidos, y guardando la información de la salida.
3. La tercer parte contempla el análisis y el front end de la información. Existe un paso intermedio entre el punto tres y cuatro donde se pasan los valores calculados por el job de hadoop de las tablas de salida de hbase. Esta información se mueve hacia la base de datos MySQL, donde tenemos las estructuras resumidas que mostraran la información.
4. El punto 4 es sobre un API de MySQL el cual será un servicio activo que se mantendrá en ejecución preguntando por las consultas del FrontEnd.
5. El FronEnd fue desarrollado en d3js, en donde se muestran gráficas y tablas de datos contenidos en una especie de dashboard de analysis.

Capa de Twiter

En esta capa hacemos la conexión al API de twitter, el cual tiene un manejo interesante, ya que maneja restricciones de cantidad conexiones en tiempos de 15 minutos, por lo que hay que definir un diseño que nos permita hacer lecturas sin que nos delimite un error conocido por el API ya que restringe dichas conexiones. Para esta clase se definió una clase conexión que controla la forma de conectarnos a hbase para poder guardar los datos leídos, y depositados finalmente en una tabla llamada TWITTER_API. Así mismo para la lectura de datos en el API se definieron varias clases, una Controler para manejar los threads de conexiones para las lecturas. La clase APIcon, en esta hacemos la lógica para lograr hacer la conexión a twitter y poder bajar los datos. También definimos una clase Parser para limpiar en los tweets la información basura dejando solo lo importante, en esta clase Parser se eliminan caracteres especiales, preposiciones, sujetos y artículos.

Capa de Hbase

Tenemos una base de datos hbase muy útil, que nos ayuda a encapsular la información obtenida del API de twitter. Esta base de datos cuenta la particularidad de que aporta mucha funcionalidad a la hora de utilizar hadoop. Cuenta con una clase zookeper que tiene funciones especiales que ayudan a la lectura y manejo de los datos. La conexión hacia esta base de datos está presente tanto en el aplicado del API, para lograr lo inserción de los datos, como durante los procesos de análisis en hadoop. También finalmente en el proyecto comReader76 hacemos lectura de los datos para moverlos a la base de datos relacional llamada MySQL.

Capa de Hadoop

En la capa de hadoop, tenemos varias clases importantes, dos que no pueden faltar son las clases Mapper y Reducer. En la Mapper definimos todos aquellos campos que debemos leer para poder hacer el análisis requerido con las salidas esperadas. En la clase Reducer, incluimos la lógica necesaria para lograr los resultados, en estas clases hacemos usos de tipos complejos creados por nosotros, para lo cual se crea una clase SummaryWrittable, esta clase nos permite escribir nuestros propios tipos. Nosotros utilizamos la clase en lugar de la conocida Result que usa hadoop, y gracias a esto podemos lograr que en un solo job de MR, podamos resolver los 6 análisis requeridos, ya que podemos jugar con el resultado de ciertas salidas que nos sirven como llave para otros cálculos.

Capa de MySQL

La capa MySQL se define como una capa de bajo nivel, ya que aquí se acceden mucho a procedimientos almacenados que nos permiten hacer inserciones y lecturas de datos resumidos. Tenemos una serie de tablas y otro tipo de estructuras que nos ayudan a mantener información organizada y lista para poder mostrar luego, cuando esta sea requerida por el frond end. Esta capa va muy ligada a un proceso Reader que alimenta el modelo en la base de datos SQL, y también a un API SQL que se mantiene en ejecución a la espera de peticiones de las cuales micro servicios se encargarán de proveer los datos al front end.

Capa de FrontEnd

Esta es la capa final, en donde se mostrarán los datos y se presentarán de manera elegante. Lo que se desarrollo fue una aplicación web que luce como un dashboard de negocio donde presentamos los datos esenciales y consolidados al inicio y luego podemos ir adentrándonos en el análisis viendo los datos agregados por dia y si se quiere un nivel de detalle más amplio, podemos ver los datos hasta por día y hora. Tenemos sumarizaciones de totales generales y top de algunas medidas que dan visibilidad general a los datos. Es importante mencionar que el frond end se desarrolló con Angular D3js. El d3js es un framework creado en java script para hacer gráficas y uso de modelos 3d. Esta combinación muy interesante, pero a la ves retadora permitió llevar a cabo un bonito diseño agradable a la vista y útil para el análisis. Así se puede observar en la ilustración 3.

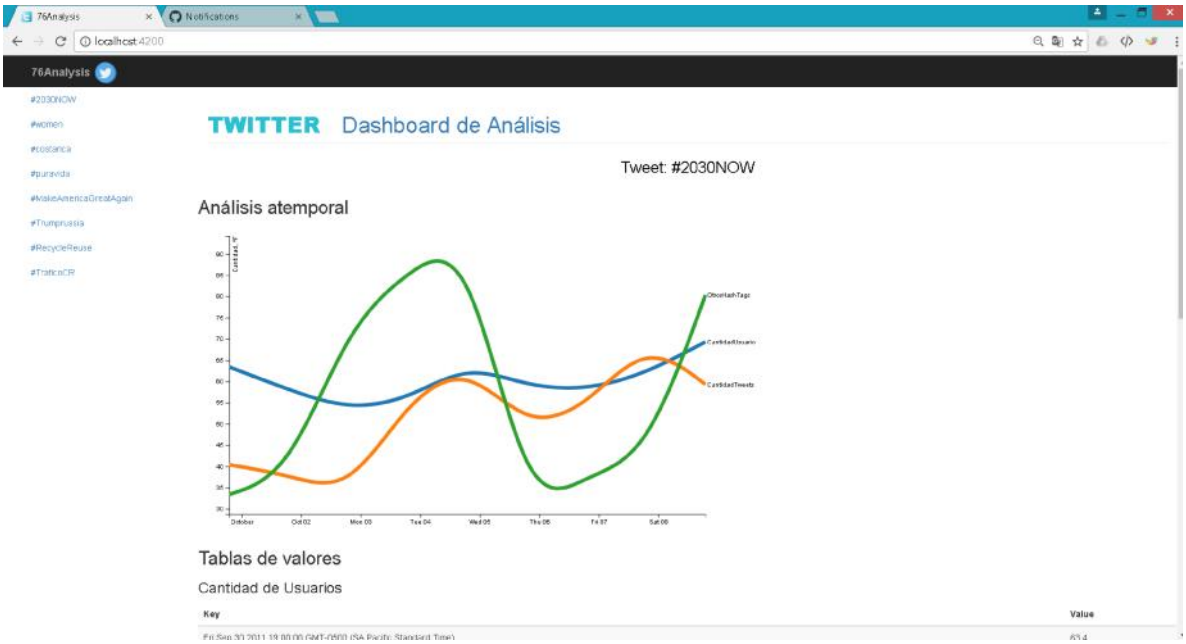


ILUSTRACIÓN 0-3

- En la ilustración 2, podemos ver las vistas que utilizamos para mostrar los datos agregados requeridos para el análisis.

ILUSTRACIÓN 0-4

Formato de datos y queries

Los datos en MySQL se guardan en estructuras temporales y luego pasan a tablas de resúmenes. Para poder hacer uso de los datos se tienen procedimientos almacenados que consultan las tablas resúmenes y vistas prediseñadas que nos ayudan al manejo de la información.

En el caso de Hbase, esta base de datos la utilizamos como primer origen para análisis y primera salida de datos, resultado del job de hadoop.

Capa de persistencia

Apoyamos un api simple para la persistencia y usamos la edición de BDB Java como el valor por defecto. Otros motores de almacenamiento soportados son MySQL, almacenamiento en memoria (usado para pruebas unitarias) y nuestro propio motor de almacenamiento personalizado.

- En la ilustración 3 vemos estructuras estas, algunas de salidas y tablas temporales de la base de datos de MySQL llamada 76analysis.

ILUSTRACIÓN 0-5

Front End Diseño

Como podemos ver en las Ilustraciones 4 y 5, el diseño presentado muestra un diseño limpio y ordenado que permite gran visibilidad de los datos para el análisis de la información. Se hace una programación en type script para modelar las gráficas y la interfaz en general.

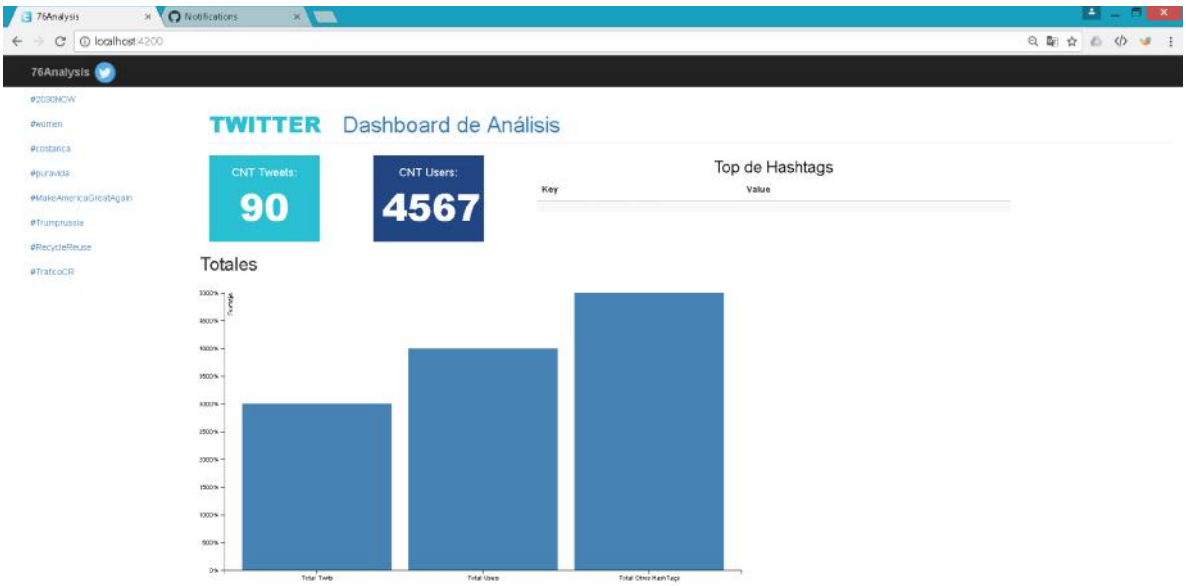


ILUSTRACIÓN 0-6

Conclusiones

Al inicio tuvimos quisimos implementar un modelo de datos final tipo estrella, finalmente nos decidimos por tomar información que resulto del análisis de hadoop para crear unas tablas con la información organizada que pudiésemos presentar utilizando la herramienta d3js, con la que tuvimos problemas por el desconocimiento que se tenía de la misma.

Es normal que cuando instalamos por primera vez algo se nos pueda presentar algún tipo de problema, nosotros no fuimos la excepción ya que al ser una instalación para un proyecto universitario utilizamos máquinas virtuales y hardware con un no tan alto rendimiento; algunos parámetros en los archivos de configuración hay que modificarlos. Dichos archivos ya vienen por defecto con configuraciones que van de la mano de acuerdo al rendimiento y recursos, por lo que se tuvo que jugar con estos parámetros para lograr un estado de equilibrio y que realmente funcionara los datos.

El manejo de la información a nivel de querys en en hbase y enMySQL fue relativamente fácil aunque

Notamos que el rendimiento de la base de datos aun cuando se almacenábamos bastante información seguía siendo muy bueno y rápido. Creemos firmemente que al ser hbase como una gran tabla hash el acceso a la información es sencillo.

Pueda ser que nos faltó experimentar un poco más con hbase y hadoop trabajándo de forma distribuida con varios nodos, esto nos habría ejemplificado más las bondades que hbase y hadoop ofrece en sistemas distribuidos.

Manual de Usuario

Se deben cumplir una serie de pasas lógicos que nos guiaran al resultado final donde el análisis será presentado en un aplicativo web que nos permita ver datos fácilmente en dashboard de información estilo gerencial.

1. Ejecución de proyecto com.76Analysis. Este proyecto utiliza un API de twitter, donde conectamos para descargar la información de 8 principales hashtags que fueron datos en la definición del proyecto. Este proceso corre durante días bajando información que posteriormente será analizada. Tenemos acá una conexión a una base de datos de hbase, donde tenemos una estructura llamada TWITTER_API, que contiene toda la información que se ha descargado. Se deben tener en ejecución los servicios de hbase.

2. Ejecución Hadoop. Seguidamente se ejecuta un proyecto llamado MR76Analysis, este proyecto es el que realiza el análisis utilizando los datos cargados en hbase y se implementan el job de análisis de los datos en hadoop. Para que este proceso corra correctamente es importante tener en ejecución tanto los servicios de hadoop, como los servicios de hbase.

3. Carga de datos a MySQL. Hay un proyecto intermedio que se debe poner en ejecución, para esto los servicios de MySQL deben estar corriendo. Este proyecto se llama comReader76 y se encarga de tomar los datos finales entregados por el job de hadoop y mover esta información a tablas temporales, y a partir de estas es que se cargan la información final presentada, utilizando vistas y consultas estructuradas que serán solicitadas por el frontEnd.

4. Ejecución del front End. Finalmente se leventa un API que tien una conexión a MySQL y maneja un conjunto de peticiones que serán solicitadas por el front end para presentar los datos utilizando d3js. El proyecto que se debe ejecutar se llama comAPI_MySQL.

Bibliografía

Varios. (2015). Obtenido de API twitter: https://dev.twitter.com/

Varios. (2017). *hbase.* Obtenido de https://hbase.apache.org/

Varios. (2017). *Nodejs.* Obtenido de https://nodejs.org/es/

Varios. (s.f.). *Github.* Obtenido de https://github.com/Ariel-dono/76Analysis.git