PRÁCTICA BIG DATA ARCHITECTURE

Jaime Gómez Recasens

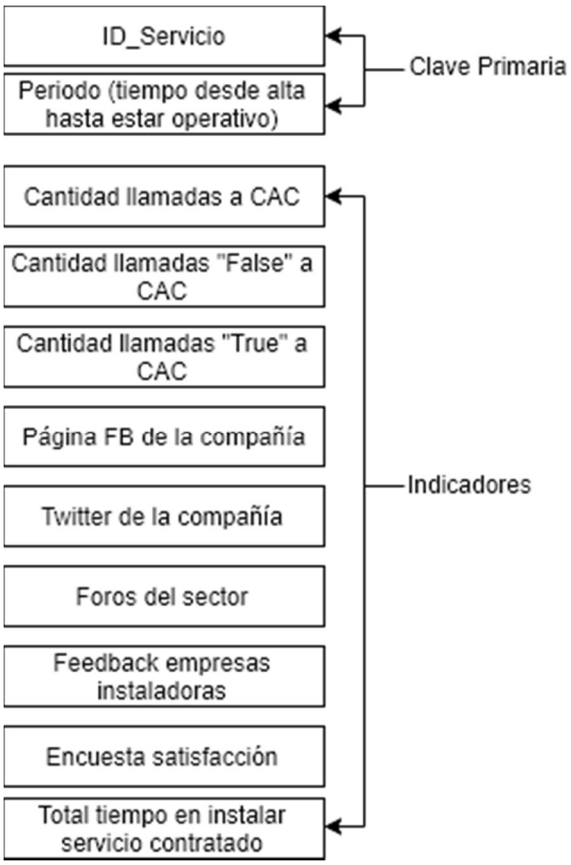
Índice

Primera parte: Diseño

Con esta práctica lo que se pretende es hacer un estudio del tiempo que se tarda en realizar una instalación de un servicio en el domicilio de un cliente para así conseguir optimizar el tiempo de instalación, dar una mejor estimación de cuándo estará el servicio operativo y adelantarse a posibles quejas y/o reclamaciones. También buscaríamos intentar adelantarnos a las posibles bajas.

Para ello vamos a realizar análisis a partir de datos de instalación de líneas de clientes obtenidos de los operacionales y datos de interacciones de esos clientes con nuestro call center y quejas o alabanzas obtenidos en la web y actividad de los clientes en las redes sociales.

De esta forma diseñamos el siguiente tablón con la información que vamos a necesitar:



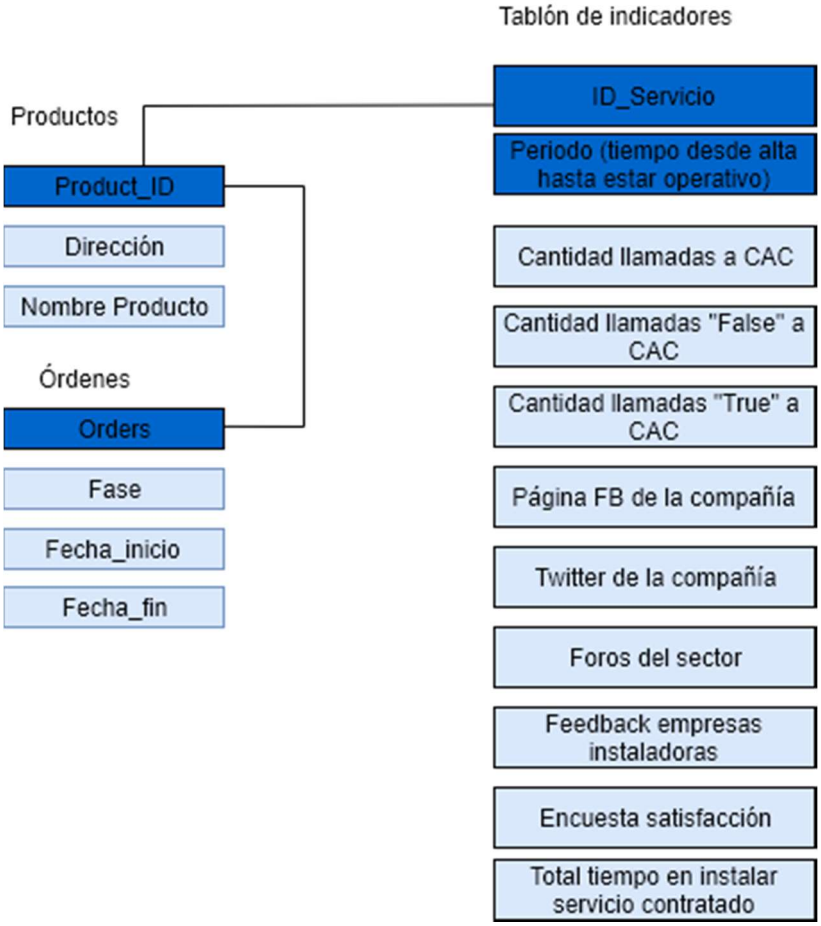
Como claves primarias tenemos el identificador de servicio contratado y el periodo de tiempo que va desde que el cliente lo contrata hasta. Recordemos que es esto lo que queremos optimizar y estudiar en detalle.

Como indicadores tenemos:

- Cantidad llamadas a CAC (Centro de Atención al Cliente).
- Cantidad llamadas “False” a CAC.
- Cantidad llamadas “True” a CAC.
- Página Facebook de la compañía: para monitorizar si hay alguna duda/opinión/sugerencia/crítica.

- Twitter de la compañía: mismo caso que el anterior.
- Foros del sector: mismo caso que los dos anteriores.
- *Feedback* de empresas instaladoras.
- Encuesta satisfacción.
- Total tiempo en instalar servicio contratado: Contabilizar cuánto se ha tardado en instalar el servicio.

En cuanto al modelo de datos a nivel lógico tenemos que sólo vamos a utilizar dos bases de datos: ORDERS y PRODUCTOS. De la primera obtenemos el identificador de servicio y de la segunda la dirección y el nombre del producto. Nuestro modelo de datos a nivel lógico quedaría así:



Una vez definido el nivel lógico ya podemos definir el modelo físico sobre Hive. Las decisiones que tenemos que tomar son:

- Tipo de almacenamiento.
- Compresión.
- Particionamiento.
- Bucketing.
- Tablas externas o gestionadas.

Tipo de almacenamiento

Para tomar una decisión sobre el tipo de almacenamiento que vamos a usar tenemos que tener en cuenta qué vamos a hacer. Básicamente, dos tipos de operaciones: escritura y lectura. En el primer caso tenemos que responder a estas preguntas para poder elegir:

- Tipo de dato.
- ¿Es el formato de dato compatible con nuestras herramientas de procesamiento y consulta?
- Tamaño de los ficheros.
- Existencia de esquemas que evolucionan con el tiempo.

Teniendo esto en mente, podemos realizar las siguientes afirmaciones:

- Usaremos AVRO cuando tengamos datos de eventos que pueden cambiar con el tiempo.
- Usaremos ficheros secuenciales cuando tengamos conjuntos de datos entre trabajos MapReduce.
- Utilizaremos ficheros de texto cuando queramos incluir grandes cantidades de datos en HDFS de una manera rápida.

En cuanto a la operación de lectura debemos considerar estos aspectos:

- Tipos de consultas: si vamos a realizar consultas por una columna específica o por un grupo de columnas usaremos Parquet u ORC puesto que usan formato columnar.
- Comprimir el fichero independientemente del formato aumenta el tiempo de velocidad de las consultas.
- El formato texto es muy lento de leer.
- Parquet y ORC optimizan el rendimiento de lectura a costa del rendimiento de escritura.

Basándonos en esto podemos considerar que:

- Elegiremos Avro cuando tengamos que hacer consultas a conjuntos de datos que han cambiado en el tiempo.
- Parquet será nuestra elección cuando solo tengamos que consultar algunas columnas en tablas grandes.

Resumiendo, nuestra elección estará entre Avro y Parquet. Para poder tomar una decisión lo más acertada posible, lo mejor sería realizar una batería de pruebas en la que pudiéramos ver el rendimiento y el tiempo de respuesta.

Compresión

En cuanto a la compresión, la elección parece clara: Snappy. Está soportado por Avro y Parquet y es la mejor opción.

Particionamiento y *bucketing*

En este caso nos decantaremos por el *bucketing*. Al tener un tablón con dos campos como claves primarias haremos el *bucketing* de uno de ellos. En nuestro caso, será el campo "ID_Servicio el que utilizaremos como columna de "*bucketing*" con lo que los registros que tengan el mismo ID_Servicio se guardarán siempre dentro del mismo *bucket*.

Tablas externas o gestionadas

Por último utilizaremos tablas externas ya que los archivos van a estar presentes en la máquina o remotamente y van a ser archivos que deban mantenerse aunque se borre la tabla.

Plan de pruebas

El plan de pruebas se centraría sobre todo en tomar una decisión sobre el tipo de almacenamiento a elegir. Habría que cargar una cantidad significativa de datos en la base de datos y hacer pruebas de lectura y escritura con compresión o sin compresión para poder tomar una decisión del tipo de almacenamiento a elegir: Avro o Parquet.

Creación tabla Productos

Con todo esto podemos plasmar la sentencia que tendremos que escribir en Hive para crear la base de datos PRODUCTOS:

```
CREATE EXTERNAL TABLE PRODUCTOS

(PRODUCT_ID INT,

DIRECCION STRING,

NOMBRE_PRODUCTO STRING)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ';'

STORED AS PARQUET LOCATION '/ruta/donde/guardar';
```

Fuentes de datos

En este punto identificamos las fuentes de cada uno de los datos. En la siguiente tabla se puede observar la procedencia:

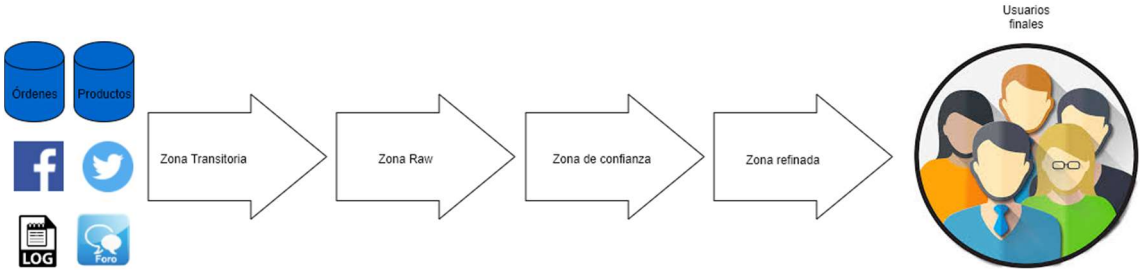
Tabla	Columna	Comentarios
Tabla Órdenes	Orden	Número de orden.
	Fase	Fase en la que se encuentra.
	Fecha_Inicio	Fecha inicio de la fase.
	Fecha_Fin	Fecha fin de la fase.
Tabla Productos	Product_ID	Identificador de producto.
	Dirección	Dirección en la que se va a realizar la instalación.
	Nombre_Producto	Nombre del producto.

En cuanto a las fuentes externas son:

Fuentes externas
Página Facebook de la empresa
Twitter oficial de la empresa
Foros
Feedback empresas instaladoras
Encuesta satisfacción

De estas fuentes externas obtenemos posibles quejas, reclamaciones, protestas o enfados que pudieran tener los clientes sobre la instalación de un servicio que han solicitado.

Una vez definido el origen de los datos, ya podemos diseñar una estrategia de carga de los mismos. Siguiendo la teoría definiremos 4 zonas: transitoria, zona de datos en crudo, zona de confianza y zona refinada. En el siguiente gráfico se refleja:



Como podemos ver, tenemos todos nuestros datos (2 bases de datos de nuestro ODS, los comentarios capturados de Facebook, foros y Twitter y log que podamos sacar de las máquinas) y el circuito que seguimos hasta llegar a los usuarios finales es el siguiente:

- Zona transitoria: zona de carga y donde realizamos las primeras comprobaciones básicas de calidad de los datos usando MapReduce y Spark.
- Zona *Raw*: el siguiente paso es cargar los datos en esta zona donde podemos enmascarar la información sensible.
- Zona de confianza: esta zona funciona como la única fuente de verdad para los datos después de haber sido limpiados y validados. Puede contener tanto datos maestros como datos referenciados. Datos maestros son conjuntos de datos básicos que necesitan estar actualizados. Los datos referenciados consisten en conjuntos de datos más complejos creados con información proveniente de múltiples fuentes.
- Zona refinada: aquí los datos se enriquecen, preparan y seleccionan para el análisis. En esta zona los datos pueden ser utilizados por aplicaciones, analistas de negocios o herramientas de inteligencia empresarial.

Nos queda ahora definir la periodicidad de la carga de los datos. La tabla Productos no requiere mucha renovación por lo que con hacer una carga cada 15 días o un mes sería suficiente. Respecto a la tabla Órdenes habría que cargarla y actualizarla cada día porque así entrarían en el sistema las nuevas altas que fuéramos teniendo.

En cuanto al resto de datos también tendrían que ser diarios: hay que tener constancia de toda noticia que se produzca en los distintos medios y que puedan afectar a las órdenes que tenemos.

Gobierno del dato

Respecto al gobierno del dato, lo desplegaremos por zonas. Así en cada una de ellas aplicaremos las que creamos más convenientes. Así

- Zona transitoria: las políticas que aplicaremos son:
    - Privacidad del dato: mediante tokenización y enmascaramiento.
    - Seguridad del dato: controlando el acceso de los usuarios.
    - Calidad del dato: realizando perfilado y verificaciones a nivel de entidad.

        - Gestión del ciclo de vida del dato: aquí los datos son de corta duración, temporales.

- Zona Raw: aplicamos las mismas que en el punto anterior y añadimos:
    - Transformación del dato: requerido para conseguir un Single View of truth.

- Zona de confianza
    - Seguridad del dato: Control de los usuarios que acceden.
    - Gestión del cicle de vida del dato: vida útil del caso de uso.

- Zona refinada: aquí sería igual que el anterior solo que dejaríamos el dato visible (obviamente si tuviera acceso a él) sin tokenización ni enmascaramiento.

Componentes a utilizar

Respecto de los componentes que utilizaremos distinguiremos entre las bases de datos de PRODUCTOS y ÓRDENES y la información que sacamos de Internet. Así, en el primer caso los pasos serán:

1. Squoop para cargar las tablas PRODUCTOS y ÓRDENES. Lo haremos en formato "Snapshot" a la zona Raw. Las cargaremos en tablas externos de Hive y las pasaremos a la zona Raw. Los datos en esta zona ya estarán en HDFS.
2. En la zona Raw definiremos una zona de Staging para hacer las últimas transformaciones.
3. Por último, pasaremos a la zona de confianza mediante comandos de Hive.

En cuanto a la información que obtenemos de Web (foros, Twitter, Facebook) el circuito es idéntico en su parte final. Difiere al comienzo en el que en vez de usar Sqoop lo haremos mediante APIs para cargar la información en la zona Raw. A partir de aquí y como dijimos el circuito es el mismo que en las bases de datos.