Lo primero a definir, es la analítica a realizar. Por ejemplo, construcción de un "tablón" de indicadores de Cliente

Queremos realizar un análisis de los clientes que tenemos para poder clasificarlos en función de su capacidad adquisitiva / nivel social y el grado de uso de las nuevas tecnologías. Para ello vamos a utilizar datos obtenidos de nuestro operacional, como la profesión del cliente y el lugar donde vive, que serán utilizados para estimar la capacidad adquisitiva del cliente a partir de datos estadísticos como el sueldo medio para una profesión o el precio medio de las casa en la ciudad en la que vive. Por otro lado, para medir el grado de uso de las nuevas tecnologías usaremos datos de actividad de este usuario en las redes sociales, el canal habitual por el que contrata nuestros productos, así como el perfil del usuario en función de su edad.

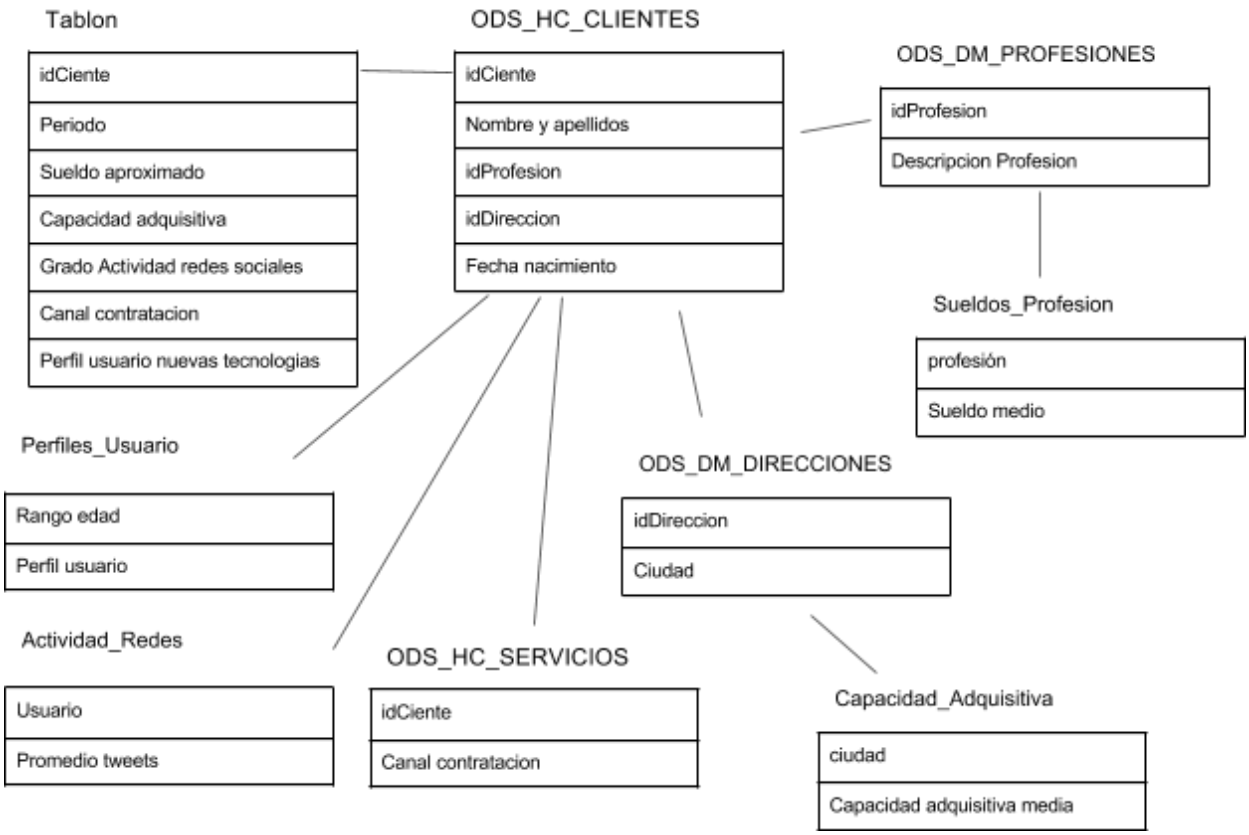
TABLON

- IdCliente (Operacional)
- Periodo
- Sueldo aproximado (obtenido del sueldo medio de su profesión)
- Capacidad adquisitiva (por la calidad del barrio donde vive)
- Actividad en redes sociales
- Canal habitual de contratación de productos
- Perfil de usuario de tecnologías en función de la edad

Habrá que definir un modelo lógico para este "datamart" y luego un modelo físico sobre Hive, indicando de forma inicial el tipo de almacenamiento, compresión, particionamiento, bucketing, cuando se trata de tablas externas o gestionadas, etc.

Modelo lógico

A continuación se muestra el modelo lógico de la trusted zone



- Las tablas cargadas desde el ODS serán tratadas como tablas externas. Las tablas generadas a partir de estas, ya procesadas en la trusted zone serán tablas internas.
- Todas las tablas del modelo físico en la trusted area final serán tablas internas gestionadas por hive, tanto el tablón (refined zone), como el resto de las tablas (trusted zone). La razón es porque todas serán tablas ya procesadas respecto a las originales.
- Respecto al formato de la tablas, en ninguno de estos casos elegiremos el formato texto para almacenar la información ya que ocupa más espacio y la eficiencia el las lecturas y escrituras es peor que en los formatos comprimidos. Y la ventaja principal, que es la legibilidad por parte de una persona, en este caso no creo que sea necesario.
    - Tablón: El formato empleado para esta tabla será **ORC**. La razón es porque este formato se comporta bien en tablas anchas, con muchas columnas. En nuestro caso no es grande, pero es muy probable que a futuro esta tabla se ampliara

significativamente en número de columnas. Este formato es eficiente en la recuperación de ciertas columnas en una tabla con muchas columnas, que es el comportamiento que queremos. Además es “splittable”, lo que permite que se puedan paralelizar las operaciones sobre el fichero.

- Tablas ODS_DM: Estas tablas van a ser muy pequeñas, por lo que no tiene mucha importancia el formato a elegir. Elijo **AVRO**, por elegir uno de los formatos comprimidos
- Tablas ODS_HC: Las características de esta tablas es que van a tener pocas columnas, y muchas filas, para este tipo de tablas es **AVRO** que se adapta bien a este tipo de formatos de tablas. AVRO es “splittable” al igual que ORC.
- Resto de tablas: El resto de tablas no provendrán de una base de datos, si no de fuentes de datos desestructuradas. Para este tipo de tablas se adapta bien **AVRO**, ya que es adecuado para evolucionar los esquemas de las tablas. Ya que los datos de estas tablas hay más probabilidad que varíen en el tiempo, ya que las fuentes originales

El formato de los ficheros en la RAW ZONE, será **AVRO**, ya que este formato es el más eficiente en escritura, lo que nos permitirá almacenar todo a gran velocidad en el formato origen.

Particionamiento y bucketing:

- las tablas ODS_DM_PROFESIONES y Sueldos_profesion no tiene sentido que tengan particiones, ya que el número de filas de estas tablas nunca va a ser muy elevado.
- Las tablas ODS_DM_DIRECCIONES y Capacidad_Adquisitiva tampoco van a crecer, el número de ciudades es limitado, por lo que no tiene sentido particionarlas
- La tabla de perfiles de usuario va a ser una tabla pequeña, tampoco vamos a introducir particiones
- La tabla de actividad en redes tiene sentido hacer un particionamiento estático por la fecha de inserción. Los datos de cada día se cargarán en una partición diferente para evitar que el tamaño del fichero crezca mucho.
- Vamos a suponer que la tabla ODS_HC_CLIENTES puede llegar a ser muy grande. Para acelerar los join que vamos a tener que realizar para crear el tablón (ver detalle de los join en los script de carga) vamos a introducir particionamiento y bucketing en las columnas id_profesión e id_dirección, que son las columnas por las que se van a realizar los join.

En este caso no vamos a implementar buckets porque la máquina sobre la que ejecutamos es muy pequeña. En caso de estar en una instalación grande, podríamos hacer buckets en la tabla ODS_HC_CLIENTES por la partición id_profesión, de esta manera podríamos ejecutar en un cluster diferente todos estos grupos de datos asociados

Una vez definido el modelo, se identificarán las fuentes de datos estructurados y "no estructurados" que formarán parte del proyecto

Orígenes de datos

- Tablas ODS del Datawarehouse
- Relación salarios por profesiones: Presupongo que obtenemos estos datos de un fichero que nos proporciona alguna organización
- Relación de la dirección con la capacidad adquisitiva: Presupongo que obtenemos estos datos de un fichero que nos proporciona alguna organización
- Perfil de nuevas tecnologías por rango de edad: Obtenemos los datos de una web por web scraping
- Datos de las redes sociales: Los obtenemos a traves de un API.

Excel adjunta ciclo vida del dato**

Periodicidad de carga y estrategias de carga:

- Raw area
    - Tablas ODS:
        - Periodicidad de carga: Una vez al día, ya que es la periodicidad con la que se carga el ODS en el datawarehouse
        - Estrategia de carga:Realizaremos una carga inicial (la que vamos a realizar en la práctica) con todos los datos de las tablas ODS que necesitamos. A partir de ahí realizaremos una carga incremental. En las tablas ODS tenemos unos campos que indican en qué fecha se dio de alta un registro o se modifico, vamos a utilizar estos campos para saber que es lo que tenemos que cargar.
    - Fichero salario_profesiones:
        - Periodicidad de carga: estos datos son el resultado de estudios estadísticos, presuponemos que vamos a tener un indicador en el origen que nos informe de cuando se ha realizado un nuevo estudio. Cada día comprobaremos si hay un nuevo estudio y de ser así realizaremos la carga
        - Estrategia de carga: En este caso el fichero no va a ser muy grande, por lo que vamos a hacer siempre cargas completas borrando el anterior, por lo que siempre será como una carga inicial
    - Fichero capacidad_adquisitiva
        - Periodicidad de carga: En este caso suponemos que no tenemos un indicador como en el caso anterior, por lo que como estos datos cambian poco y la tabla no es muy grande cargaremos una vez cada 15 dias
        - Estrategia de carga: Misma que en el fichero de salarios profesiones.
    - Datos redes sociales

        - Periodicidad de carga: A traves del API estaremos recolectando datos en intervalos de tiempo pequeños (10 minutos) ya que la actividad a traves de estos medios es muy alta
        - Estrategia de carga: Incremental, se ira pidiendo a traves del API la nueva información de los ultimos 10 minutos y se irá cargando toda.
    - Datos nuevas tecnologías por edad.
        - Periodicidad de carga: Presuponemos que la web no se va a actualizar mucho, por lo que se realizará la carga una vez al mes.
        - Estrategia de carga: La misma que para el fichero de salarios profesiones
- Trusted zone
    - Tablas ODS
        - Periodicidad de carga: Una vez al día, ya que es la periodicidad con la que se carga el ODS en el datawarehouse
        - Estrategia de carga:Realizaremos una carga inicial (la que vamos a realizar en la práctica) con todos los datos de las tablas ODS que necesitamos. A partir de ahí realizaremos una carga incremental. En las tablas ODS tenemos unos campos que indican en qué fecha se dio de alta un registro o se modifico, vamos a utilizar estos campos para saber que es lo que tenemos que cargar.
    - Sueldos_Profesion / Perfiles_Usuario / Capacidad_Adquisitiva
        - Periodicidad de carga: La misma que la del fichero de la que carga
        - Estrategia de carga: Las cargas en esta área son siempre totales, tendemos un campo de fecha de inserción, para poder cargar todos los datos de la tabla cada vez que hagamos la carga, pero cada vez con una fecha diferente de tal manera que no tengamos que borrar nada
    - Actividad_redes
        - Periodicidad de carga: Procesaremos toda la actividad en las redes sociales una vez al día
        - Estrategia de carga: Misma estrategia que en el caso anterior, tendremos una columna fecha, para poder dejar almacenados los procesamientos de todos los días
- Refined zone
    - Tablon
        - Periodicidad de carga: El tablon se generará una vez al día. No es necesario tenerlo más actualizado para la toma de decisiones
        - Estrategia de carga: Misma estrategia que en los casos anteriores, utilizaremos un campo de fecha

Gobierno del dato

Garantizar que procesamos las tablas solo una vez
Informar de errores / tratamiento de errores / reprocesamiento

- Raw area

    - Fichero salario_profesiones / Fichero capacidad_adquisitiva / Datos nuevas tecnologías por edad : En estos casos realizaremos la carga antes de borrar el fichero anterior, de producirse un error, se informará del error, se borrara lo que se haya realizado de la carga y se seguirá utilizando el fichero anterior. Estos datos no son criticos y así aseguramos que tenemos datos para poder procesar los pasos posteriores
    - Datos redes sociales: En este proceso no es crítico perder parte de la información, si se produce una pequeña perdida no se informará. Si los problemas persisten durante un cierto tiempo se informará.

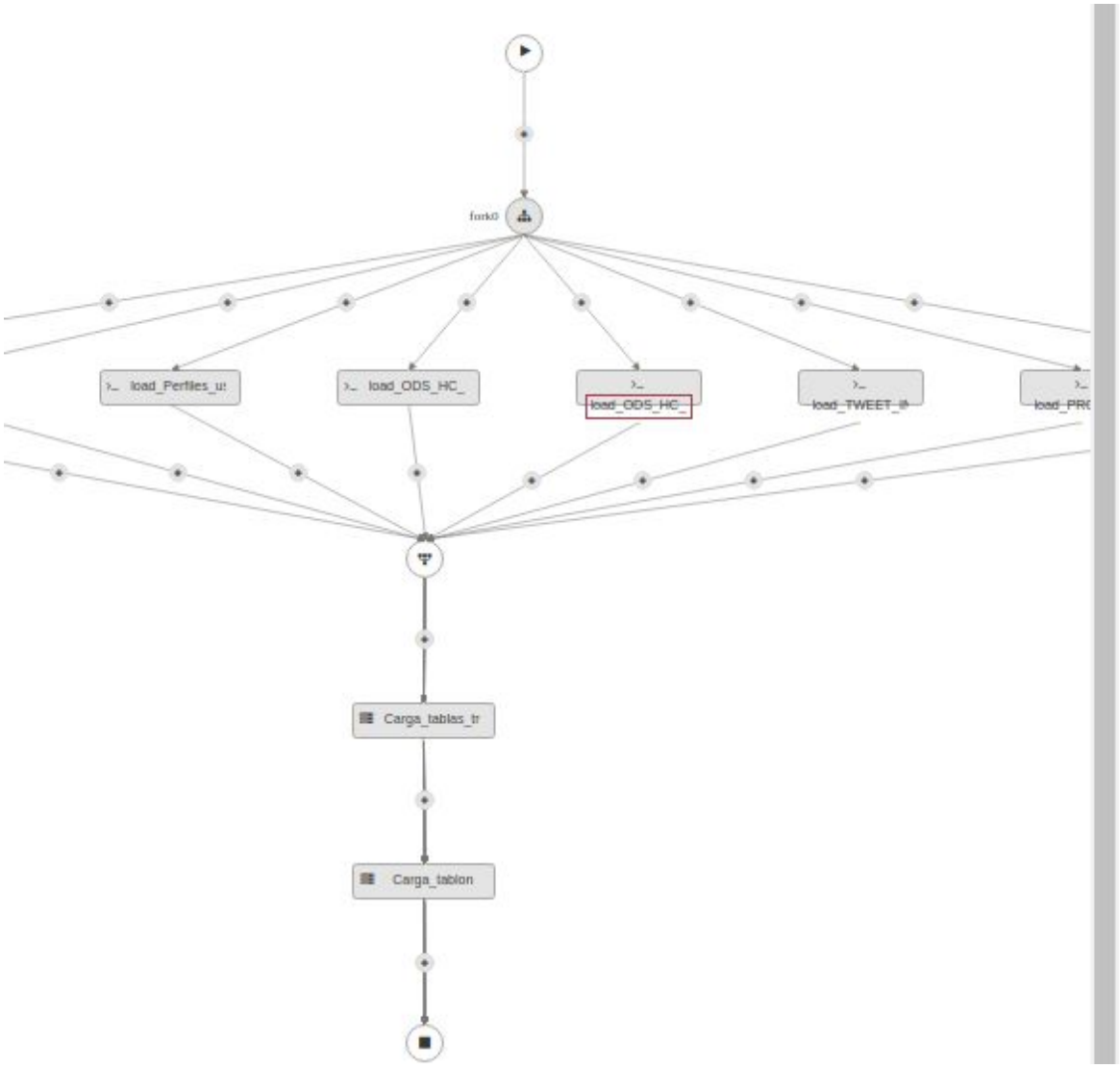
- Trusted zone
    - Tablas ODS: Al ser la carga incremental, el volumen de datos a cargar en cada incremento no es tan grande. Si el proceso de alguna de la tablas falla, se informará del fallo y se deshará lo que se ha cargado para volver a intentar el incremento de manera completa.
    - Sueldos_Profesion / Perfiles_Usuario / Capacidad_Adquisitiva: Misma estrategia que en el caso anterior. En caso de no arreglarse en el momento de ir a generar el tablón se generará con los datos anteriores a la carga.
    - Actividad_redes: en este caso no los consideramos datos críticos, si falla no se re-procesan, se informa y se espera al siguiente ciclo de carga.
- Refined zone
    - Tablon: El procesamiento de este tablón consideramos que puede ser costoso y necesitamos tener la información aunque no sea completa. En este caso en caso de fallo de la generación de un registro almacenaremos el error y continuaremos procesando hasta que se produzca un número máximo de errores que detendrá la carga, preferimos tener un tablón no completo que no tenerlo.

Herramientas de carga:

- Para la carga de las tablas desde mysql usaremos sqoop
- Para la carga de los ficheros lo realizaremos mediante un programa python
- La carga de los datos de las redes sociales lo realizaremos a traves de un API

ANEXO I

He realizado la creación de un flujo en Oozie para la ejecución secuencial de los scripts. En realidad no he conseguido hacerlo funcionar, los scripts de python solo los ejecuto desde mi local.


fork0
load_Perfiles_u
load_ODS_HC_
load_ODS_HC_
load_TWEET_#
Carga_tablas_tr
Carga_tablon