Big Data con Hadoop y Spark

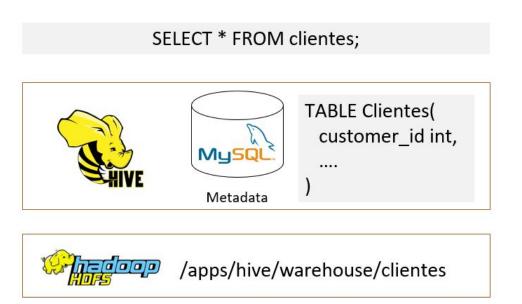
Módulo 02 - Hive



Hive

Permite crear infraestructuras de tipo de data warehouse sobre Hadoop para realizar análisis de grandes volúmenes de datos

Asigna una estructura tabular (metadata) a los datos en bruto almacenados en HDFS



HiveQL (Hive Query Language)

Hive utiliza un subconjunto de comandos SQL.

Data Definition Language https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL

Data Manipulation Language https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DML

IMPORTANTE: las operaciones de UPDATE y DELETE no están habilitadas por defecto.

Tipos de Tablas

MANAGED	EXTERNAL
Hacen referencia a un path dentro de HDFS que es administrado por Hive	Generan metadata para un path de HDFS que no es administrado por Hive
El valor por defecto se especifica en en el parámetro hive.metastore.warehouse.dir y tipicamente es /user/hive/warehouse/	Debemos agregar la palabra clave EXTERNAL y especificar el path de HDFS en la sección LOCATION
En caso de realizar una operación de tipo DROP TABLE, Hive eliminaría la metadata de la tabla y los datos	En caso de realizar una operación de tipo DROP TABLE, Hive eliminaría la metadata de la tabla pero no los datos

Tipos de Dato

Hive, además de los tipos de datos comunes a todos los motores de bases de datos relacionales, ofrece una nueva categoría de tipos de datos complejos

Complex Types

- ARRAY<data_type>
- MAP<primitive_type, data_type>
- STRUCT<col_name : data_type, ...>

•		♦ Name	♦ Type
0	<u>lılıl</u>	id	int
1	dil	lastname	string
2	dil	firstname	string
3	dil	dob	date
4	dil	newsletter	boolean
5	dil	contacts	map <string,string></string,string>
6	<u>ldtl</u>	orders	array <string></string>
7	dil	site	string

Formatos de Almacenamiento

Hive permite leer y escribir datos en diferentes formatos de archivos.





Habitualmente se utilizan 2 formatos:

- CSV para los datos en bruto
- Parquet para los datos procesados





Particiones

El particionamiento es una forma de dividir una tabla en partes relacionadas en función de los valores de columnas particulares (por ej. fecha, la ciudad y el departamento).

Cada tabla puede tener una o más claves de partición para identificar una partición particular.

Esta forma de almacenar los datos permite realizar consultas mas eficientes.

```
/user/hive/warehouse/logs
   dt=2001-01-01/
       country=GB/
        country=US/
        — file3
   dt=2001-01-02/
       country=GB/
        └─ file4
        country=US/
           file5
```

Ejemplo Hive

```
CREATE EXTERNAL TABLE page view stg(viewTime INT, userid BIGINT,
                page_url STRING, referrer_url STRING,
                ip STRING COMMENT 'IP Address of the User',
                country STRING COMMENT 'country of origination')
COMMENT 'This is the staging page view table'
ROW FORMAT DELIMITED FIELDS TERMINATED BY '44' LINES TERMINATED BY '12'
STORED AS TEXTFILE
LOCATION '/user/data/staging/page_view';
hadoop dfs -put /tmp/pv 2008-06-08.txt /user/data/staging/page view
FROM page view stg pvs
INSERT OVERWRITE TABLE page view PARTITION(dt='2008-06-08', country='US')
SELECT pvs.viewTime, pvs.userid, pvs.page url, pvs.referrer url, null, null, pvs.ip
WHERE pvs.country = 'US';
```

Links de referencia

Hive https://cwiki.apache.org/confluence/display/Hive/Home

Gracias

