Apache Hive
Big Data Aplicado

Curso de Especialización en Inteligencia Artificial y Big Data

Francisco Gallego Perona

¿Qué es Hive?

Apache Hive

Es un Data warehouse open-source diseñado para la infraestructura de Hadoop.

- Se usa para procesar grandes datasets de **datos estructurados**
- Provee de una forma de ejecutar **queries (consultas)** de HiveQL

Ventajas de Apache Hive

Las ventajas del uso de Apache Hive son las siguientes:

- Soporta trabajar con conjuntos de datos de gran tamaño
- Se ejecuta en la arquitectura de Hadoop, utilizando equipos con componentes de bajo presupuesto.
- Soporta la sintaxis SQL
- Se puede conectar a Hive desde numerosos lenguajes como Java, Scala, Python, etc.

Hive y HDFS

Hive guarda los datos de sus tablas en HDFS. Por defecto utiliza la ruta /user/hive/warehouse dentro de HDFS si no se especifica nada a la hora de crear la tabla.

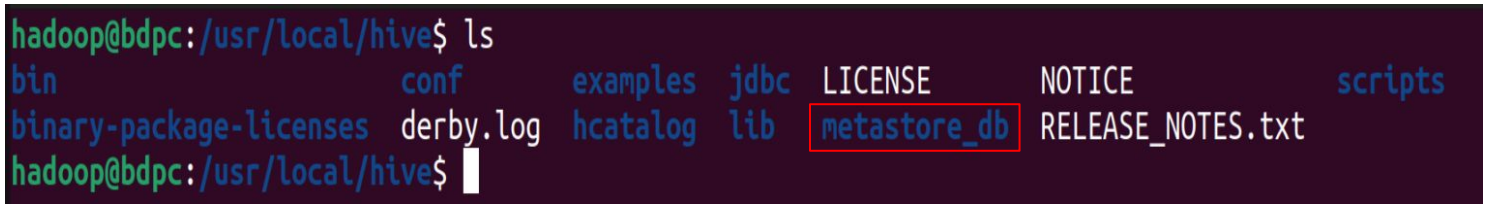
Para especificar la ruta donde se encuentran los datos de la tabla usamos la cláusula LOCATION.

Trabajando con Hive necesitamos saber acerca de dos puntos de almacenamiento:

- Hive Metastore
- Hive Data warehouse Location → Donde se almacenan los datos de las tablas.

Hive y HDFS: Metastore

El metastore de Hive se utiliza para almacenar metadatos de las bases de datos y tablas creadas.



A la hora de ejecutar Hive (usando el comando hive al añadir la carpeta de hive al PATH), debemos ejecutar el comando estando en la carpeta donde tenemos toda la configuración de Hive.

Hive y HDFS: Almacenamiento en HDFS

Por defecto los datos se almacenan en /user/hive/warehouse, a menos que se especifique la dirección de la tabla que se está creando.

En esta localización tendremos un directorio por base de datos y subdirectorios por cada una de las tablas creadas.

Database

```
LOCATION
  'hdfs://localhost:8020/user/hive/warehouse/dbm.db/migration'
```

Table

Hive y HDFS: Almacenamiento en HDFS

En las siguientes imágenes podemos ver cómo haciendo un ls del directorio warehouse dentro de HDFS, tenemos el directorio correspondiente a la base de datos dbm.db. Dentro de este tendremos los datos de todas las tablas.

```
hadoop@bdpc:/usr/local/hive$ hdfs dfs -ls /user/hive/warehouse
Found 1 items
drwxr-xr-x   - hadoop supergroup          0 2022-10-27 21:53 /user/hive/warehouse/dbm.db
```

```
hadoop@bdpc:/usr/local/hive$ hdfs dfs -ls /user/hive/warehouse/dbm.db
Found 2 items
drwxr-xr-x   - hadoop supergroup          0 2022-10-28 11:25 /user/hive/warehouse/dbm.db/migration
drwxr-xr-x   - hadoop supergroup          0 2022-10-27 21:34 /user/hive/warehouse/dbm.db/migration_ext
```

Hive y HDFS: Almacenamiento en HDFS

Una vez dentro de cada una de las tablas podemos ver que, los datos que contienen son ficheros .csv.

```
hadoop@bdpc:/usr/local/hive$ hdfs dfs -ls /user/hive/warehouse/dbm.db/migration
Found 2 items
-rw-r--r--   1 hadoop supergroup         93 2022-10-28 11:24 /user/hive/warehouse/dbm.db/migration/000000_0
-rw-r--r--   1 hadoop supergroup   24099622 2022-10-27 21:49 /user/hive/warehouse/dbm.db/migration/imj.csv
```

Mostrando las bases de datos y tablas

Para mostrar las bases de datos existentes en Hive podemos usar el comando:

show databases;

Un ejemplo de la salida del comando es la siguiente:

```
hive> show databases;
OK
dbm
default
Time taken: 0.466 seconds, Fetched: 2 row(s)
```

Mostrando las bases de datos y tablas

Para usar una de las bases de datos disponibles en Hive, de cara a acceder a las tablas que esta contiene usamos el comando:

use <database>;

De esta forma podremos consultar las tablas que contiene dicha base de datos de la siguiente forma:

```
hive> use dbm;
OK
Time taken: 0.063 seconds
hive> show tables;
OK
migration
Time taken: 0.101 seconds, Fetched: 1 row(s)
```

Creando Bases de datos en Hive

En primer lugar, debemos crear una base de datos para gestionar tablas. Para ellos usamos el comando CREATE DATABASE de la siguiente manera:

CREATE DATABASE [IF NOT EXISTS] <database_name>;

El [IF NOT EXISTS] es opcional, para que intente crear la tabla solo si esta no existe previamente.

Actividad: Creando Bases de datos en Hive

Crea una base de datos de prueba llamada test en Hive. Accede a la dirección en HDFS donde se crean los datos de dicha base de datos.

Comprueba que se ha creado en la dirección de HDFS concreta. Cuando crees una tabla, puedes comprobar que existe un subdirectorio dentro de dicha carpeta.

Creación de tablas en Hive

En Hive tenemos dos tipos de tablas:

- **Externas** (external) → Hive no gestiona los datos de la tabla.
- **Internas** (managed o internal) → Hive se encarga de gestionar, tanto los datos de la tabla como los metadatos (estructura de la tabla).

Cuando haces un drop de la tabla (eliminas la tabla), si esta tabla es externa, los datos se quedan almacenados en HDFS, mientras que si es interna (managed), tanto la tabla como los datos se eliminan.

Hive: Managed vs External

INTERNAL OR MANAGED TABLE	EXTERNAL TABLE
By default, Hive creates an Internal or Managed Table.	Use EXTERNAL option/clause to create an external table
Hive owns the metadata, table data by managing the lifecycle of the table	Hive manages the table metadata but not the underlying file.
Dropping an Internal table drops metadata from Hive Metastore and files from HDFS	Dropping an external table drops just metadata from Metastore with out touching actual file on HDFS.
Hive supports ARCHIVE, UNARCHIVE, TRUNCATE, MERGE, CONCATENATE operations	Not supported
Supports ACID/Transactional	Not supported
Supports result caching	Not supported

Hive: Managed vs External

Cuándo usar tablas Externas o Internas (managed):

- Usa tablas managed cuando generes tablas temporales
- Usa tablas externas cuando los datos estén en localizaciones remotas → Por ejemplo, localizaciones de HDFS en otro sistema.
- Otra opción para usar tablas externas es cuando es necesario mantener los datos al eliminar la tabla.

Creación de tablas en Hive

CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.] table_name

[(col_name data_type [column_constraint] [COMMENT col_comment], ...)]

[PARTITIONED BY (col_name data_type [COMMENT 'col_comment'], ...)]

[CLUSTERED BY (col_name, col_name,.....]

[COMMENT table_comment]

[ROW FORMAT row_format]

[FIELDS TERMINATED BY char]

[LINES TERMINATED BY char]

[LOCATION 'hdfs_path']

[STORED AS file_format]

Creación de tablas en Hive

Ejemplo de sintaxis de creación simple de tabla en Hive:

```
CREATE TABLE [IF NOT EXISTS] [db_name].table_name (
 field field_type,
 field field_type,
 field field_type,
 field field_type )
 COMMENT 'This is a comment'
 ROW FORMAT DELIMITED
 FIELDS TERMINATED BY ',';
```

Tipos de datos en Hive

NUMERIC TYPES	DESCRIPTION
TINYINT	1-byte signed integer, from `-128` to `127`
SMALLINT	2-byte signed integer, from `-32,768` to `32,767`
INT/INTEGER	4-byte signed integer, from -2,147,483,648 to 2,147,483,647
BIGINT	8-byte signed integer, from `-9,223,372,036,854,775,808` to `9,223,372,036,854,775,807`
FLOAT	4-byte single precision floating point number
DOUBLE	8-byte double precision floating point number
DOUBLE PRECISION	Alias for DOUBLE, only available starting with Hive 2.2.0
DECIMAL	It accepts a precision of 38 digits.
NUMERIC	Same as DECIMAL type.

Tipos de datos en Hive

DATE/TIME TYPES	DESCRIPTION
TIMESTAMP	Accepts Both Date and Time
DATE	Accepts just Date
INTERVAL	Interval

Tipos de datos en Hive

STRING TYPES	DESCRIPTION
STRING	The string is an unbounded type. Not required to specify the lenght. It can accept max up to 32,767 bytes.
VARCHAR	Variable length of characters. It is bounded meaning you still need to specify the length like VARCHAR(10).
CHAR	Fixed length of Characters. if you define char(10) and assigning 5 chars, the remaining 5 characters space will be wasted.

Actividad: Creación de tablas en Hive

En esta actividad se crearán:

- Una base de datos de prueba
- Una tabla llamada iris dentro de esa base de datos
- La tabla contendrá los campos del csv: https://gist.github.com/netj/8836201

Elige los tipos de datos necesarios para la correcta creación de la tabla, teniendo en cuenta los valores existentes dentro del csv.

Tablas temporales vs regulares

TEMPORARY TABLE	**REGULAR TABLE** (INTERNAL/EXTERNAL)
Creates a table within a session	Creates globally
Can be accessed only from a session it created	Table could be accessed from different session right after created
Stores at users scratch directory */tmp/hive/<user>/**	Stores at Hive warehouse directory `/user/hive/warehouse`
Automatically removed when session terminated	Persist until explicitly dropped
Doesn’t support partitions	Supports partitions
Indexes cannot be created	You can create Indexes

Particiones en Hive

El particionado de datos sirve para incrementar el rendimiento:

- Los valores de una columna particionada **dividen una tabla en segmentos**
- Al realizar consultas podemos **ignorar particiones enteras**

Las particiones deben ser creadas y manejadas por los usuarios:

- Debe ser especificada la partición al insertar datos
- Al realizar consultas Hive filtra las particiones de forma automática

Particiones en Hive

```
CREATE TABLE tweets (
        user STRING,
        post STRING,
        time BIGINT)
PARTITIONED BY (country STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ‘,’
STORED AS TEXTFILE;
```

Partición de la tabla por el campo ‘country’.

No hay diferencia entre una partición y las columnas de datos normales en el esquema.

Particiones en Hive

Cargar datos en una tabla particionada:

LOAD DATA LOCAL INPATH 'path_hasta_los_datos_us.csv'

OVERWRITE INTO TABLE tweets PARTITION (country='US');

LOAD DATA LOCAL INPATH 'path_hasta_los_datos_es.csv'

OVERWRITE INTO TABLE tweets PARTITION (country='ES');

Particiones en Hive

Para mostrar los datos de las particiones:

SHOW PARTITIONS tweets;

Las particiones se crean en subcarpetas dentro de la ruta de la tabla en HDFS.

Si hacemos un listado de los ficheros:

hdfs dfs -ls -R /user/hive/warehouse/tweets

/user/hive/warehouse/tweets/country=US

→ Todos los ficheros dentro de la partición US

/user/hive/warehouse/tweets/country=ES

→ Todos los ficheros dentro de la partición ES

Particiones en Hive

Cómo realizar **consultas** sobre tablas particionadas:

- No hay diferencia en la sintaxis
- Cuando la columna particionada es especificada en el WHERE, se realiza la discriminación por directorios/particiones.

SELECT * FROM tweets WHERE country='US' LIMIT 10;

Particiones en Hive

Más información sobre particiones en:

https://sparkbyexamples.com/apache-hive/hive-partitions-explained-with-examples/

https://www.analyticsvidhya.com/blog/2020/12/15-basic-and-highly-used-hive-queries-that-all-data-engineers-must-know/