Programa de formación **MLDS**

UNIVERSIDAD NACIONAL DE COLOMBIA

Módulo
BIG DATA
Apache Hive

Por
Ing. Jorge Camargo, PhD.

UNIVERSIDAD NACIONAL DE COLOMBIA

Agenda

- Apache Hive
- Arquitectura
- HiveQL – HQL
- Referencias

Apache Hive

UNIVERSIDAD NACIONAL DE COLOMBIA

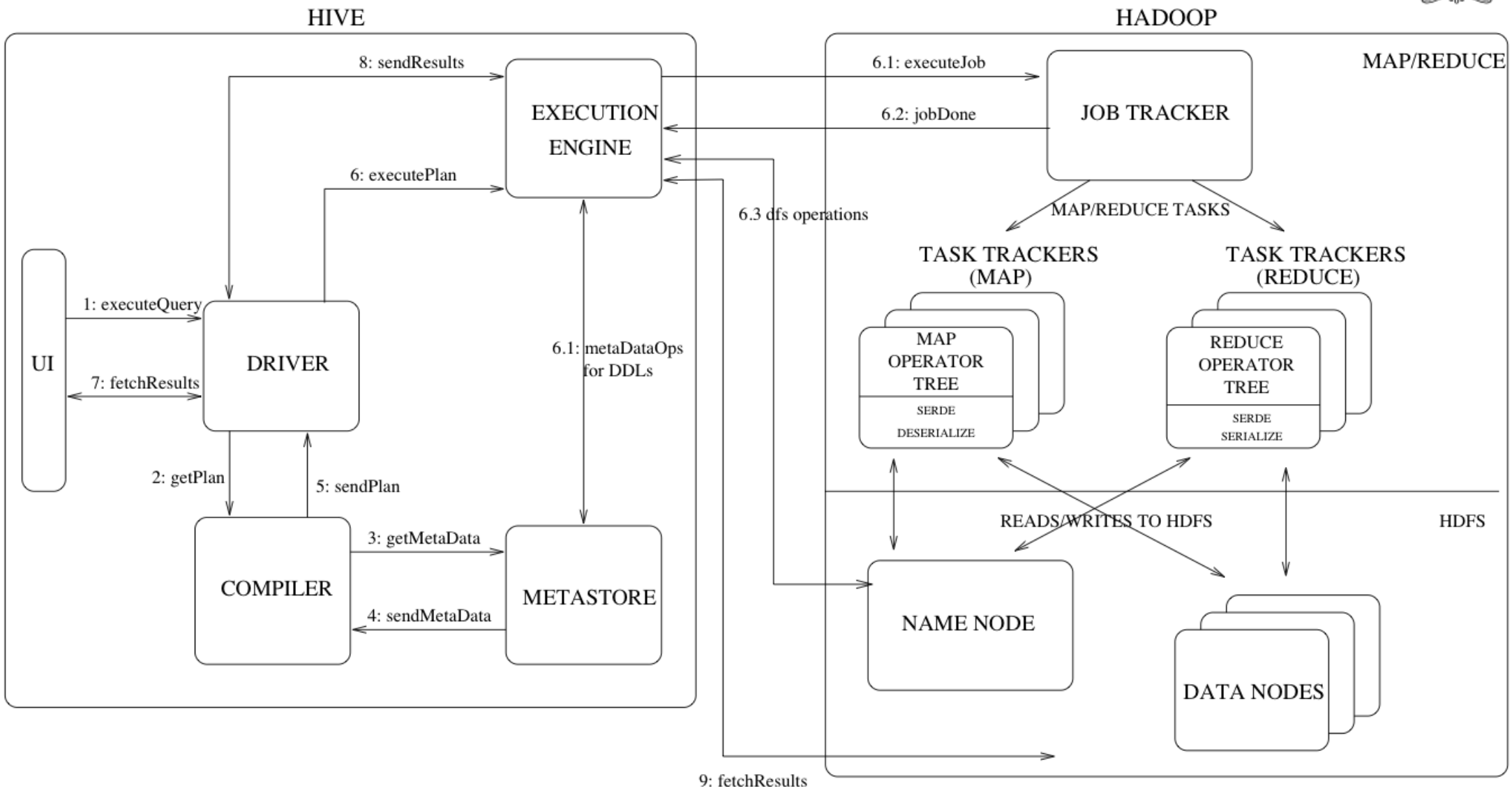
- Es un software para Bodegas de Datos, que permite leer, escribir y analizar grandes conjuntos de datos almacenados de manera distribuida.
- HiveQL (HQL) es un lenguaje similar a SQL que permite consultar y analizar datos estructurados.
- Está diseñado para OLAP (Online Analitical Processing)
- Trabaja sobre Hadoop. Los conjuntos de datos son almacenados en HDFS. Las consultas son convertidas internamente a tareas MapReduce, Apache Tez o tareas Apache Spark.

Apache Hive

- Almacena el esquema en una base de datos relacional y los datos procesados en HDFS.

- Es rápido, escalable y extensible.

Arquitectura



Arquitectura

- UI: Interfaz gráfica de usuario para enviar consultas y otras operaciones al sistema. También cuenta con una interfaz de línea de comandos (CLI).

- Driver: Componente que recibe las consultas (*queries*). Este componente implementa las APIs de ejecución y la extracción sobre interfaces JDBC/ODBC.

- Compiler: Verifica la sintaxis de la consulta y analiza semánticamente los diferentes bloques y expresiones de la consulta. Genera un plan de ejecución apoyado por los datos obtenidos del MetaStore.
- MetaStore: Almacena la estructura de la información de las tablas y particiones de la bodega de datos, incluyendo información sobre las columnas y los archivos HDFS en los cuales se almacenan los datos.

Arquitectura

- Execution Engine: Ejecuta el plan de ejecución creado por el compilador. El plan es un DAG (Directed Acyclic Graph) de etapas. Administra las dependencias entre las diferentes etapas.

HiveQL

UNIVERSIDAD NACIONAL DE COLOMBIA

- Tipos de Datos:
  - Tipos de Columnas
  - Literales
  - Valores Nulos
  - Tipos Complejos

- Tipos de Columnas: Son los tipos de datos para definir columnas en Hive.

  - Enteros:
    - **TINYINT** -> *5Y*
    - **SMALLINT** -> 15S
    - **INT** -> 146
    - **BIGINT** -> 12424L
    - **FLOAT**
    - **DOUBLE**

HiveQL

UNIVERSIDAD NACIONAL DE COLOMBIA

- Tipos de Columnas: Son los tipos de datos para definir columnas en Hive.

  - Cadenas de caracteres: Especificados mendiante ‘ ’ o “ ”.
    - **STRING**
    - **CHAR**. Longitud máxima 255
    - **VARCHAR**. Longitud máxima 65355

  - **TimeStamp**: Tiempo en milisegundos más Precisión de nanosegundos opcional en Formato UNIX.
    - También en formato YYYY-MM-DD HH:MM:SS.fffffffff

HiveQL

UNIVERSIDAD NACIONAL DE COLOMBIA

- Tipos de Columnas: Son los tipos de datos para definir columnas en Hive.
  - Misc:
    - **BOOLEAN**
    - **BINARY**
  - Fechas:
    - **DATE** -> YYYY-MM-DD
  - Decimales: Valores de punto flotante equivalentes a BigDecimal en Java.
    - **DECIMAL(precision, scale)** -> decimal(10, 0)
  - Uniones: Colección heterogénea de tipos de datos.
    - **UNIONTYPE** -> UNIONTYPE<int, double, array<string>, struct<a:int,b:string>>
    -> {0:1} {1:2.0} {2:["three","four"]} {3:{"a":5,"b":"five"}}
    -> {2:["six","seven"]} {3:{"a":8,"b":"eight"}} {0:9} {1:10.0}

HiveQL

UNIVERSIDAD NACIONAL DE COLOMBIA

- Literales:
  - Mapas:
    - **MAP<primitive_type, data_type>**
  - Arreglos:
    - **ARRAY<data_type>**
  - Estructuras:
    - **STRUCT<col_name:data_type, …>**

HIVE

HiveQL

- Valores Nulos:
    - Representan datos faltantes:
        - **NULL**

HiveQL - DDL

UNIVERSIDAD NACIONAL DE COLOMBIA

- **CREATE DATABASE:** Crea una nueva base de datos en Hive

```
CREATE DATABASE|SCHEMA [IF NOT EXISTS] <database name>
```

  - Ejemplo:

```
hive> CREATE SCHEMA userdb;
```

```
hive> SHOW DATABASES;
default
userdb
```


HiveQL - DDL

- **DROP DATABASE:** Elimina las bases de datos en Hive.

```
DROP (DATABASE|SCHEMA) [IF EXISTS] database_name [RESTRICT|CASCADE];
```

  - Ejemplo:

```
hive> DROP DATABASE IF EXISTS userdb;
```

CASCADE indica la eliminación previa de las tablas antes de eliminar la base de datos.

HiveQL - DDL

- **CREATE TABLE:** Crea una nueva tabla en una base de datos.

- Suponga que queremos crear una tabla **Employee** con las siguientes columnas:
  - eid int
  - name string
  - salary float
  - designation string

- Ejecutamos la siguiente consulta

HiveQL - DDL

```
hive> CREATE TABLE IF NOT EXISTS employee ( eid int, name String,
salary String, destination String)
COMMENT ‘Employee details’
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ‘\t’
LINES TERMINATED BY ‘\n’
STORED AS TEXTFILE;
```

HiveQL - DDL

- **LOAD:** Carga datos a las tablas en Hive desde archivos

- Suponga que tenemos el archivo **sample.txt** en el directorio **/home/user** con el siguiente contenido:

```
1201  Gopal       45000    Technical manager
1202  Manisha     45000    Proof reader
1203  Masthanvali 40000    Technical writer
1204  Kiran       40000    Hr Admin
1205  Kranthi     30000    Op Admin
```

- Para cargarlo a la tabla **employee** se ejecuta la siguiente consulta.

HiveQL - DDL

```
hive> LOAD DATA LOCAL INPATH '/home/user/sample.txt'
OVERWRITE INTO TABLE employee;
```

- OVERWRTIE es opcional e indica que se deben reemplazar todos los datos en la tabla.

HiveQL - DDL

- **ALTER TABLE:** Modifica la estructura de las tablas en Hive

Sintaxis:

```
ALTER TABLE name RENAME TO new_name
ALTER TABLE name ADD COLUMNS (col_spec[, col_spec ...])
ALTER TABLE name DROP [COLUMN] column_name
ALTER TABLE name CHANGE column_name new_name new_type
ALTER TABLE name REPLACE COLUMNS (col_spec[, col_spec ...])
```

HiveQL - DDL

UNIVERSIDAD NACIONAL DE COLOMBIA

- **ALTER TABLE:** Modifica la estructura de las tablas en Hive

Ejemplos:

```
hive> ALTER TABLE employee RENAME TO emp;
```

```
hive> ALTER TABLE employee CHANGE name ename String;
hive> ALTER TABLE employee CHANGE salary salary Double;
```

```
hive> ALTER TABLE employee ADD COLUMNS (
dept STRING COMMENT 'Department name');
```


HiveQL - DDL

UNIVERSIDAD NACIONAL DE COLOMBIA

- **DROP TABLE:** Elimina tablas en Hive

Sintaxis:

```
DROP TABLE [IF EXISTS] table_name;
```

Ejemplo:

```
hive> DROP TABLE IF EXISTS employee;
```

HiveQL - Operadores

UNIVERSIDAD NACIONAL DE COLOMBIA

- **Relacionales:**
  - `=, !=, <, <=, >, >=`
  - `IS NULL, IS NOT NULL`
  - `LIKE, RLIKE, REGEXP`

- **Aritméticos:**
  - `+, -, *, /, %`

- **Nivel de Bits:**
  - `&` (AND)
  - `|` (OR)
  - `^` (XOR)
  - `~` (NOT)

HiveQL - Operadores

UNIVERSIDAD NACIONAL DE COLOMBIA

- **Lógicos:**
  - `AND equivalente a &&`
  - `OR equivalente a ||`
  - `NOT equivalente a !`

- **Acceso a elementos en ARRAY, MAP, STRUCT:**
  - `A[n] – Retorna el` **`n-th`** `elemento en el array` **`A`**`.`
  - `M[k] – Retorna el elemento de clave k en el mapa` **`M`**`.`
  - `S.a – accede al campo de nombre a en la estructura` **`S`**`.`

HiveQL - Vistas

UNIVERSIDAD NACIONAL DE COLOMBIA

- **CREATE VIEW:** Crea vistas a partir de consultas SELECT

```
hive> CREATE VIEW emp_30000 AS
SELECT * FROM employee
WHERE salary>30000;
```

- **DROP VIEW**: Elimina una vista

```
hive> DROP VIEW emp_30000;
```


HiveQL - Índices

UNIVERSIDAD NACIONAL DE COLOMBIA

- **CREATE INDEX:** Crea índices en los campos de una tabla

```
hive> CREATE INDEX inedx_salary ON TABLE employee(salary)
AS 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler';
```

- **DROP INDEX**: Elimina un índice

```
hive> DROP INDEX index_salary ON employee;
```


HiveQL

UNIVERSIDAD NACIONAL DE COLOMBIA

- **SELECT:** Consulta sobre los registros de las tablas

```
hive> SELECT * FROM employee WHERE salary>30000;
```

```
hive> SELECT Id, Name, Dept FROM employee ORDER BY DEPT;
```

```
hive> SELECT Dept,count(*) FROM employee GROUP BY DEPT;
```

```
hive> SELECT * FROM employee WHERE Salary>40000 && Dept=TP;
```

HiveQL

UNIVERSIDAD NACIONAL DE COLOMBIA

- **JOIN:** Misma definición de OUTER JOIN en SQL.

```
hive> SELECT c.ID, c.NAME, c.AGE, o.AMOUNT
FROM CUSTOMERS c JOIN ORDERS o
ON (c.ID = o.CUSTOMER_ID);
```

- Hive cuenta además con las clausulas (definidas de igual manera que en SQL):
  - **LEFT OUTER JOIN**
  - **RIGHT OUTER JOIN**
  - **FULL OUTER JOIN**

HiveQL

UNIVERSIDAD NACIONAL DE COLOMBIA

- **INSERT:** Agregar datos a una tabla

```
CREATE TABLE students (name VARCHAR(64), age INT, gpa DECIMAL(3, 2))
  CLUSTERED BY (age) INTO 2 BUCKETS STORED AS ORC;

INSERT INTO TABLE students
  VALUES ('fred flintstone', 35, 1.28), ('barney rubble', 32, 2.32);
```


HiveQL

UNIVERSIDAD NACIONAL DE COLOMBIA

- **UPDATE:** Modificar registros de una tabla. Sintaxis similar a la SQL.

```
Standard Syntax:
UPDATE tablename SET column = value [, column = value ...] [WHERE expression]
```

- **DELETE:** Eliminar registros de una tabla. Sintaxis similar a la SQL.

```
Standard Syntax:
DELETE FROM tablename [WHERE expression]
```


Referencias

- Apache Hive, https://hive.apache.org/
- Wiki Apache Hive, https://cwiki.apache.org/confluence/display/Hive/Home
- HQL, https://cwiki.apache.org/confluence/display/Hive/LanguageManual