

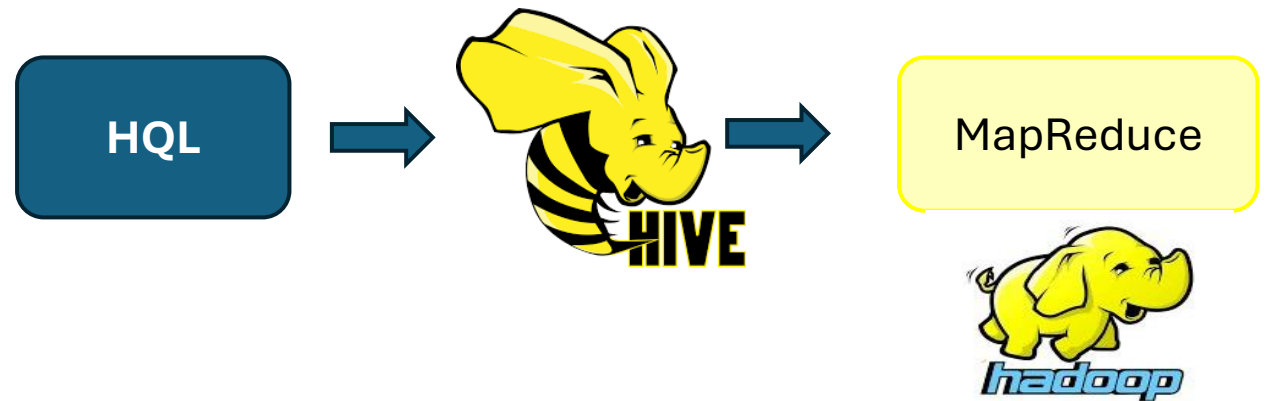
Hive

# ¿Qué es Hive?

Herramienta de data warehousing construida sobre Hadoop  
Facilita el análisis de grandes conjuntos de datos almacenados en Hadoop mediante consultas HiveQL (parecido a SQL)

Hive traduce las consultas HiveQL en trabajos de MapReduce que se ejecutan sobre Hadoop

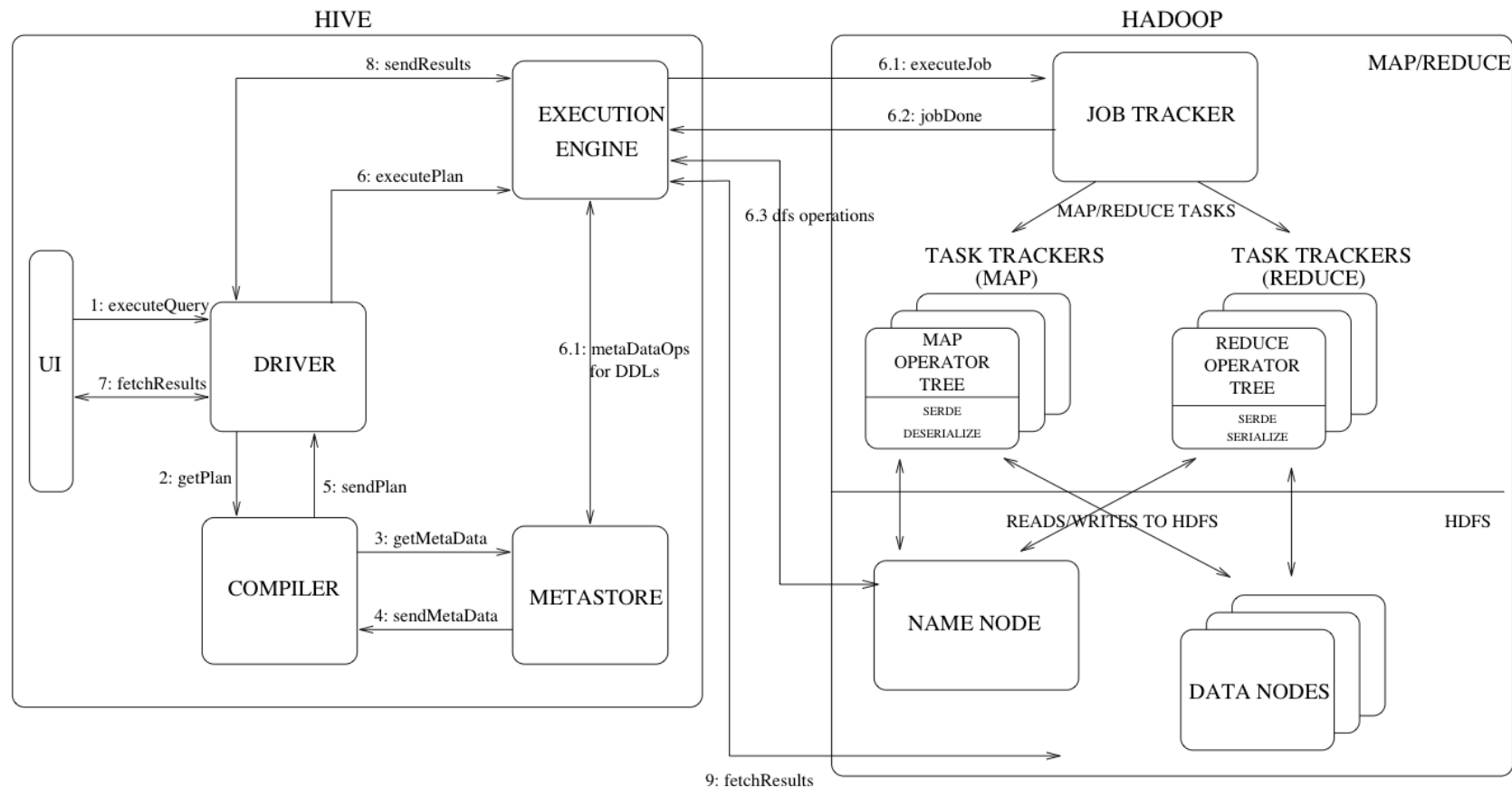
Procesamiento Batch



# Historia y Evolución de Hive



# Arquitectura



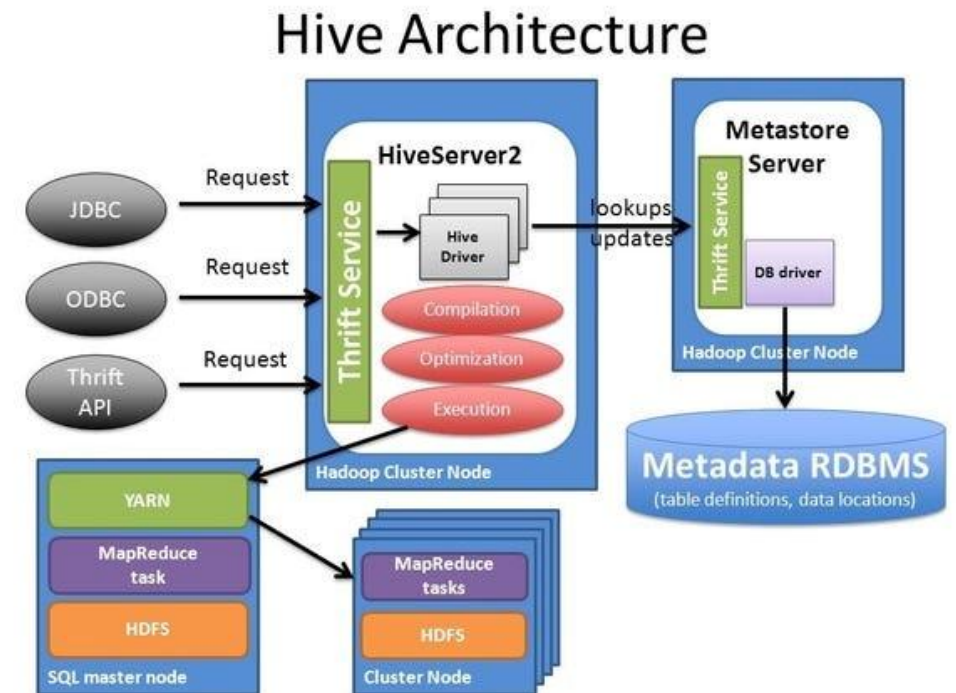
- **Hive Driver:** Gestiona la ejecución de consultas.
- **Compiler:** Convierte HiveQL a DAG de MapReduce o Tez.
- **Metastore:** Almacena metadatos de tablas y esquemas.
- **Integración con el cliente:**
  - Se utiliza el framework de RPC Thrift para la comunicación entre HiveServer2 y los clientes.
- **Interacción con Hadoop**
  - Integración con HDFS para almacenamiento.
  - Utilización de MapReduce/Tez/Spark para procesamiento.

# Metastore

Repositorio central para metadatos de Hive.

- **Funciones Principales**
  - Almacenamiento de esquemas de tablas y particiones
  - Gestión de ubicaciones de datos en HDFS.
- **Tipos de Metastore**
  - Embedded (por defecto con Derby).
  - Externo (MySQL, PostgreSQL, etc.).
- **Acceso y Seguridad**
  - Control de acceso a metadatos.
  - Integración con sistemas de autenticación.
- **Optimización y Rendimiento**
  - Caching de metadatos para consultas rápidas.

Los metadatos definen las tablas, su estructura y el tipo de datos.



# Lenguaje de Consulta HiveQL

- **Características de HiveQL**

- Similar a SQL, fácil de aprender para usuarios SQL.
- Soporta consultas ad-hoc y programadas.

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual>

- **DDL (Data Definition Language)**

- CREATE, ALTER, DROP.

- **DML (Data Manipulation Language)**

- SELECT, INSERT, UPDATE, DELETE.

- **Funciones Integradas**

- Funciones agregadas, de ventana, string, matemáticas, etc.

- **Extensibilidad**

- Soporte para UDFs (User Defined Functions). Pueden desarrollarse en lenguajes como Java y registrarse en Hive para su uso en consultas, ofreciendo mayor flexibilidad.

## Optimización de Consultas

- Uso de particiones y bucketing.

```
CREATE TABLE employees (id INT, name STRING, salary FLOAT);  
SHOW TABLES;  
INSERT INTO employees VALUES (1, 'Alice', 50000.0), (2, 'Bob', 60000.0);  
SELECT * FROM employees;
```

# Tipos de Tablas en Hive

- **Tablas Internas (Managed Tables)**
  - Hive gestiona completamente los datos.
  - Eliminación de tabla elimina los datos.
- **Tablas Externas (External Tables)**
  - Hive solo gestiona el esquema.
  - Los datos permanecen en HDFS al eliminar la tabla.
- **Tablas Particionadas**
  - División de datos en particiones basadas en una columna.
  - Mejora el rendimiento de las consultas.
- **Tablas Clusterizadas (Bucketing)**
  - Distribución de datos en buckets para optimizar consultas.

```
CREATE TABLE empleados (  
    id INT,  
    nombre STRING,  
    edad INT,  
    departamento STRING  
)  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
STORED AS TEXTFILE;
```

- **Tablas Temporales y Vistas**
  - Vistas virtuales para simplificar consultas complejas.

# Tablas internas

En Hive, las **tablas internas** (también conocidas como **tablas gestionadas**) son controladas completamente por Hive en términos de almacenamiento y gestión de datos.

- **Almacenamiento:** Hive gestiona completamente los datos almacenados en tablas gestionadas. Los datos se almacenan en el directorio de almacenamiento de Hive (por defecto, en el directorio /user/hive/warehouse en HDFS).
- **Creación:** Se crean usando el comando CREATE TABLE.
- **Eliminación:** Cuando se elimina una tabla gestionada con DROP TABLE, tanto la definición de la tabla como los datos subyacentes se eliminan del sistema de archivos.
- **Uso:** Son útiles cuando deseas que Hive gestione el ciclo de vida completo de los datos.



# Tablas internas

## Tipos de tablas gestionadas:

- **Tablas CRUD Transaccionales (ACID):**
  - Estas tablas permiten **todas las operaciones CRUD** y son totalmente transaccionales, es decir, cumplen con los principios **ACID** (Atomicidad, Consistencia, Aislamiento, Durabilidad).
  - Se almacenan por defecto en formato **ORC**, que está optimizado para lecturas y escrituras eficientes.
- **Tablas Insert-Only Transaccionales:**
  - Aunque son **transaccionales**, estas tablas **solo soportan inserciones** (INSERT) y no permiten actualizaciones ni eliminaciones (UPDATE y DELETE).
  - Pueden almacenar datos en cualquier formato de archivo (ORC, texto, CSV, etc.).
- **Tablas Temporales:**
  - Usadas para almacenar datos **temporales** o **intermedios** que no necesitan persistencia más allá de una sesión. No son transaccionales y se eliminan automáticamente cuando finaliza la sesión en la que se crearon.

# Tablas internas

Como crear tablas

[https://docs.cloudera.com/cdw-runtime/cloud/using-hiveql/topics/hive\\_create\\_a\\_crud\\_transactional\\_table.html](https://docs.cloudera.com/cdw-runtime/cloud/using-hiveql/topics/hive_create_a_crud_transactional_table.html)

- **Tablas CRUD Transaccionales (ACID):**

- CREATE TABLE T(a int, b int);

- **Tablas Insert-Only Transaccionales:**

- CREATE TABLE T2(a int, b int)
- STORED AS ORC
- TBLPROPERTIES ('transactional'='true',
- 'transactional\_properties'='insert\_only');

- **Tablas Temporales:**

- CREATE TEMPORARY TABLE tmp2 AS SELECT c2, c3, c4 FROM mytable;

# Tablas externas

Tablas externas en Hive permiten definir una estructura sobre datos existentes en HDFS u otros sistemas de almacenamiento sin que Hive sea responsable de gestionarlos.

Hive no elimina los datos al borrar una tabla externa, lo que las hace ideales para datos compartidos o reutilizados por diferentes aplicaciones.

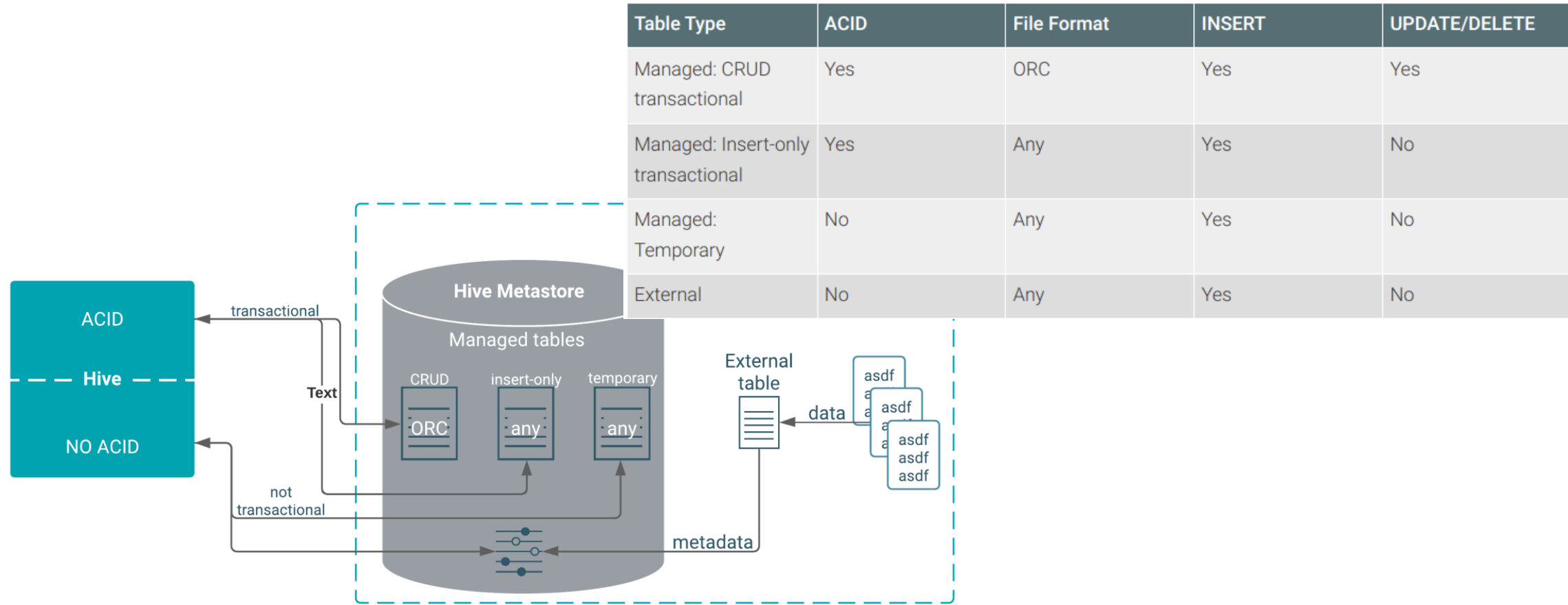
- **Gestión de datos externa:** Hive solo gestiona el **esquema** de la tabla, pero no los datos. Los datos permanecen intactos incluso si se elimina la tabla.

- **Ubicación definida manualmente:** Los datos de una tabla externa deben estar en una ubicación especificada explícitamente usando la cláusula `LOCATION`.

- **Uso típico:** Ideal para datos que se actualizan o gestionan fuera de Hive, o para integrar datos de diferentes fuentes de almacenamiento.

```
CREATE EXTERNAL TABLE sales_data
(
  id INT, product STRING, amount
  FLOAT
)
ROW FORMAT DELIMITED FIELDS
TERMINATED BY ',' LOCATION
'/user/data/sales/';
```

# Tabla externa vs interna



# Particionado

Cada tabla puede tener una o más claves de partición que determinan cómo se almacenan los datos

Las particiones permiten que el sistema elimine los datos que se van a inspeccionar en función de los predicados de la consulta

```
CREATE TABLE empleados (  
  id INT,  
  nombre STRING,  
  edad INT,  
  departamento STRING,  
  salario FLOAT  
)  
PARTITIONED BY (  
  cargo STRING,  
  periodo STRING  
)  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
STORED AS TEXTFILE;
```

Una consulta para un cargo y un periodo seleccionado nos daría la ruta HDS donde se encuentran los datos:

/user/hive/warehouse

## Ventajas

- Reducción del tiempo de consulta.
- Manejo eficiente de grandes datasets.

/empleados

/cargo=gerente

/cargo=asistente

/periodo=2010

/periodo=2020

/periodo=201

/user/hive/warehouse/cargo=gerente/periodo=2020/

# Bucketing

División de datos en un número fijo de buckets basado en una función hash de una columna.

```
CREATE TABLE empleados (  
  id INT,  
  nombre STRING,  
  edad INT,  
  departamento STRING  
)  
PARTITIONED BY (  
  cargo STRING,  
  gerente STRING  
)  
CLUSTERED BY (salario) INTO 2 BUCKETS  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ','  
STORED AS TEXTFILE;
```

## Ventajas

- Optimización de joins y agregaciones.
- Distribución uniforme de datos.

# Particionado y Bucketing

Particionado	Bucketing
Un directorio es creado en HDFS para cada partición	Un fichero es creado en HDFS para cada bucket
Puedes tener una o más columnas de partición	Solo puedes tener una única columna de bucketing
No puedes gestionar el número de particiones para crear	Puedes gestionar el número de buckets a crear especificandolo.
En HQL: PARTITIONED BY	En HQL: CLUSTERED BY


# Ecosistema Hadoop

- **Apache HBase:**
  - Base de datos NoSQL distribuida para almacenamiento en tiempo real.
  - Integración con Hive para consultas SQL sobre datos almacenados en HBase.
- **Apache Pig:**
  - Plataforma de alto nivel para crear scripts de procesamiento de datos.
  - Complementa a Hive proporcionando alternativas de scripting.
- **Apache Oozie:**
  - Sistema de flujo de trabajo para orquestar tareas en Hadoop.
  - Permite programar y gestionar pipelines de datos que incluyen consultas Hive.
- **Apache Sqoop:**
  - Herramienta para transferir datos entre Hadoop y bases de datos relacionales.
  - Facilita la importación y exportación de datos hacia y desde Hive.
- **Apache Flume:**
  - Servicio para recopilar y mover grandes cantidades de datos de registro.
  - Integración con Hive para almacenar y analizar datos en tiempo real.



# Instalación

1. Instalar Hadoop
2. Opcional - Instalar MySQL (para metastore)
3. Ir a <https://www.apache.org/dyn/closer.cgi/hive/> y descargar la última versión de Hive
4. Configurar variables de entorno
5. Inicializa el metastore
6. Configurar el fichero: hive-site.xml
7. Arranca el servicio hive: *hiveserver2* &
8. Prueba a conectarte a Hive con *beeline -u jdbc:hive2://localhost:10000*

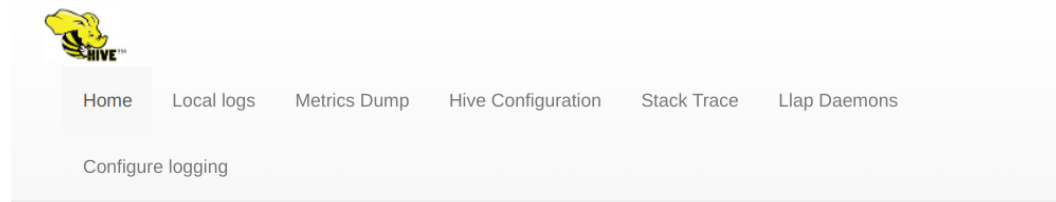


```
<property>
  <name>yarn.resourcemanager.address</name>
  <value>myubuntu:8032</value>
  <description>Dirección del ResourceManager de YARN</description>
</property>

<property>
  <name>yarn.resourcemanager.scheduler.address</name>
  <value>myubuntu:8030</value>
  <description>Dirección del programador de trabajos de YARN</description>
</property>
```

# HiveServer2 Web UI

http://myubuntu:10002/



## Active Sessions

User Name	IP Address	Operation Count	Active Time (s)	Idle Time (s)
-----------	------------	-----------------	-----------------	---------------

Total number of sessions: 0

## Open Queries

User Name	Query	Execution Engine	State	Opened Timestamp	Opened (s)	Latency (s)	Drilldown Link
-----------	-------	------------------	-------	------------------	------------	-------------	----------------

Total number of queries: 0

## Last Max 25 Closed Queries

User Name	Query	Execution Engine	State	Opened (s)	Closed Timestamp	Latency (s)	Drilldown Link
-----------	-------	------------------	-------	------------	------------------	-------------	----------------

Total number of queries: 0

HiveServer2 es el servicio principal de Hive que gestiona las conexiones de los clientes, las consultas y la autenticación. Ofrece una interfaz web que te permite ver el estado de las sesiones, consultas activas, y otras estadísticas útiles.

- Estado de las sesiones.
- Consultas activas o finalizadas.
- Información sobre la configuración de HiveServer2.

# HUE

<http://myubuntu:8888/>

The screenshot displays the Hue web interface for interacting with Hive. The top navigation bar includes the Hue logo, a search bar for saved documents, and buttons to add new documents or descriptions. The left sidebar contains various icons for navigation. The main content area shows a Hive query editor with the query `select * from employees`. Below the editor, the execution logs show the query being compiled and executed successfully. The results are displayed in a table with two columns: `employees.id` and `employees.name`. The table contains six rows of data.

	employees.id	employees.name
1	1	Alice
2	2	Bob
3	3	Charlie
4	4	Diana
5	5	Eve
6	6	Frank

Hue es una interfaz web muy utilizada para interactuar con Hive y otros componentes del ecosistema Hadoop. Proporciona una UI amigable para ejecutar consultas, ver resultados, administrar tablas, y más.

## arrancar Hue:

```
cd hue
docker-compose up
```

## parar Hue:

```
CTRL+C
docker-compose rm
```

User/passwd: hadoop/hadoop