Formación senior en Hive

PARTE DE LA FORMACIÓN BIG DATA ENGINEER DE BIG DATA ACADEMY PERÚ

FORMADOR: ALONSO MELGAREJO [alonsoraulmgs@gmail.com]

Concepto

Hive

Es una **infraestructura de almacenamiento batch** de datos **construida sobre Hadoop**, que permite **ejecutar procesos en diversos motores de procesamiento por medio de sintaxis SQL.**

Objetivo fundamental

Objetivo fundamental de Hive

1. Ofrecer una capa “fachada” (patrón facade) para **ejecutar procesos con sintaxis SQL**
2. Servir de **repositorio central de metadata**

Naturaleza de funcionamiento

Si programáramos directamente en MapReduce

```
27          //Función Map
28          //Notar como los tipos de datos de la clave y el valor coinciden con los definidos en la definición genérica de la clase Mapper.
29          //Adicionalmente se utiliza un tercer parámetro, el "Context", el cual almacena los resultados parciales de la función map.
30          public void map(Object clave, Text valor, Context contexto) throws IOException, InterruptedException{
31              //Tokenizamos las palabras
32              StringTokenizer tokenizador = new StringTokenizer(valor.toString());
33
34              //Recorremos las palabras tokenizadas
35              while(tokenizador.hasMoreTokens()){
36                  //Obtenemos la palabra
37                  palabra.set(tokenizador.nextToken());
38
39                  //Convertimos la entrada en una estructura clave/valor
40                  contexto.write(palabra, laUnidad);
41              }
42          }
43
44      }
45
46      //Implementación de la función reduce
47      //La clase Reducer define cuatro tipo de datos genéricos
48      //Los dos primeros son para definir los tipos de datos de la clave y cada uno de los elementos de la lista de valores de entrada
49      //Los dos últimos son para definir los tipos de datos de la clave y el valor de salida
50      public static class ClaseReduce extends Reducer<Text, IntWritable, Text, IntWritable>{
51
52          //Variable para el resultado de contar cada palabra
53          private IntWritable resultado = new IntWritable();
54
55          //Función Reduce
56          //Notar como los tipos de datos de la clave y la lista de valores coinciden con los definidos en la definición genérica de la clase Reducer.
57          //Adicionalmente se utiliza un tercer parámetro, el "Context", el cual almacena los resultados parciales de la función reduce.
58          public void reduce(Text clave, Iterable<IntWritable> listaDeValores, Context contexto) throws IOException, InterruptedException{
59              int suma = 0;
60
61              //Iteramos cada valor de la lista de valores
62              for(IntWritable valor : listaDeValores){
63                  suma = suma + valor.get();
64              }
65
66              //Colocamos el resultado
67              resultado.set(suma);
68
69              //Escribimos el resultado en el contexto
70              contexto.write(clave, resultado);
71          }
72      }
73
74      public static void main(String[] args) throws Exception{
```

Programación avanzada en Java con todo lo que ello implique:

- Perfiles senior Java
- Mantenibilidad compleja
- Compilación de programas
- Nullpointerexception…

Lo mismo en Hive

```
1  SELECT COUNT(*) FROM TABLA_PALABRAS;
```

¿Cómo se ve un programa en Hive?

```
SELECT
    p.name Product_Name,
    s.name Supplier_Name
FROM
    products_suppliers ps
        JOIN products p ON ps.productID = p.productID
            JOIN suppliers s ON ps.supplierID = s.supplierID
WHERE
    p.name = 'Pencil 3B'
UNION
SELECT
    products.name Product_Name,
    suppliers.name Supplier_Name
FROM
    products_suppliers
        JOIN products ON products_suppliers.productID = products.productID
            JOIN suppliers ON products_suppliers.supplierID = suppliers.supplierID
WHERE
    price < 0.6
UNION
SELECT
    p.name Product_Name,
    s.name Supplier_Name
FROM
    products p,
    products_suppliers ps,
    suppliers s
WHERE
    p.productID = ps.productID AND
    ps.supplierID = s.supplierID AND
    s.name = 'ABC Traders';
```

Imaginen la complejidad para construir este programa con procesos **MapReduce** sobre Hadoop

Características de Hive

1. Tiene una **sintaxis SQL** conocida como HiveQL que manipula los archivos de HDFS
2. Nos abstrae de la necesidad de implementar programar con MapReduce, las implementaciones se hacen con SQL lo cual **facilita el desarrollo y el mantenimiento de los programas**
3. Es una **tecnología batch**, ya que los queries están orientados a correr por **minutos u horas**.
4. Soporta diferentes **formatos de almacenamiento** como TEXTFILE, ORC, AVRO y PARQUET.
5. Permite la creación de **UDFs** (funciones definidas por el usuario)

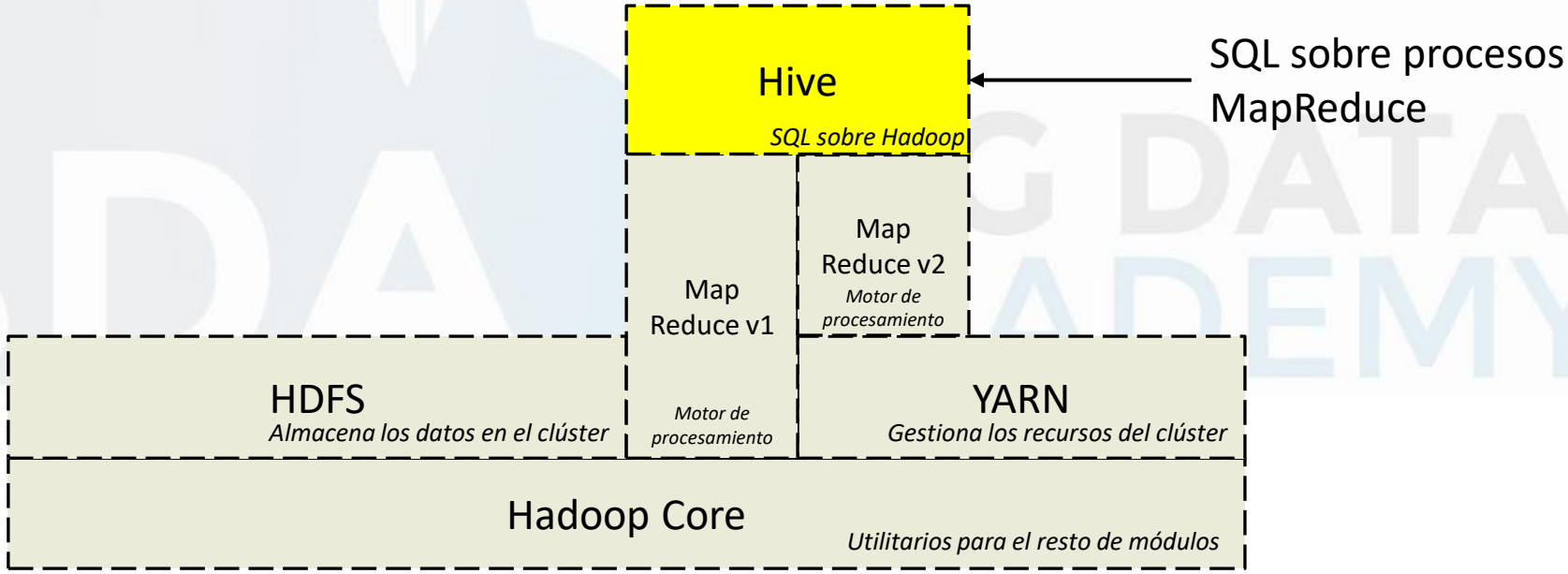
Las consola de Hive

Existen dos consolas en Hive:

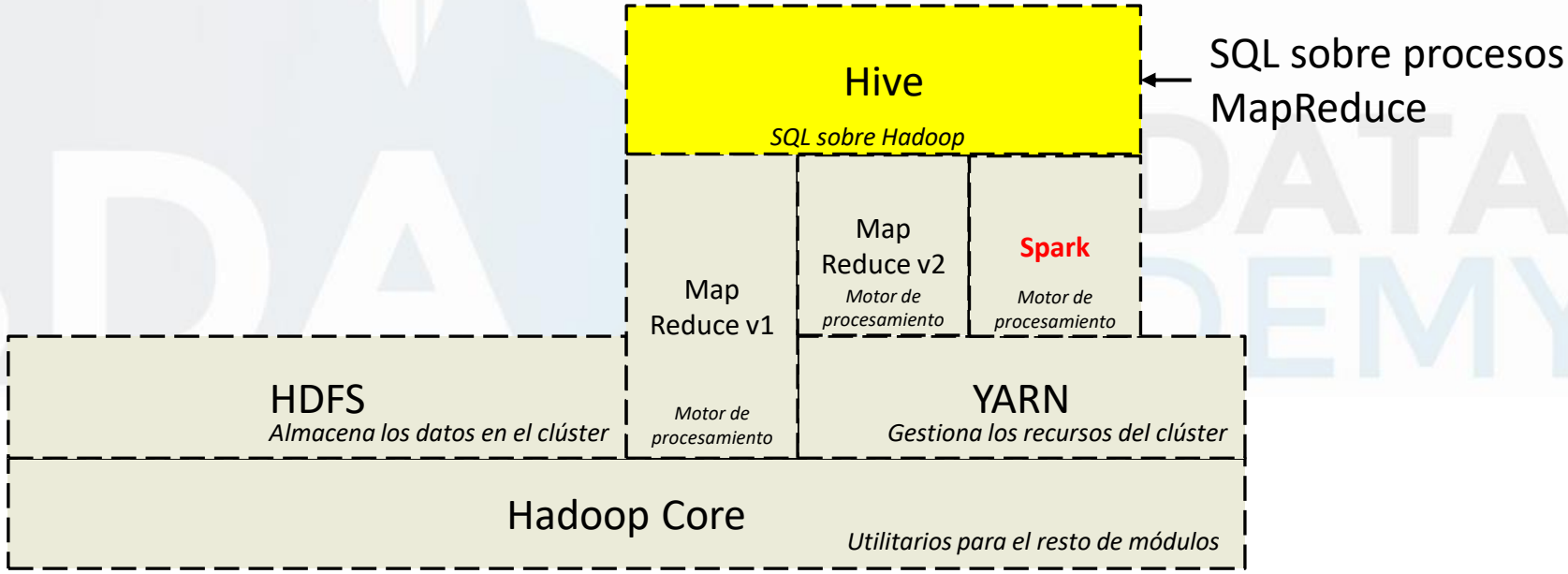
1. **Hive CLI**: Primera consola de Hive, **actualmente en desuso** ya que sólo permite una conexión a la vez.
2. **Beeline**: Consola de Hive que **funciona por medio de una conexión JDBC**. Es la que actualmente se usa

Arquitectura de componentes

Hive dentro del ecosistema Hadoop

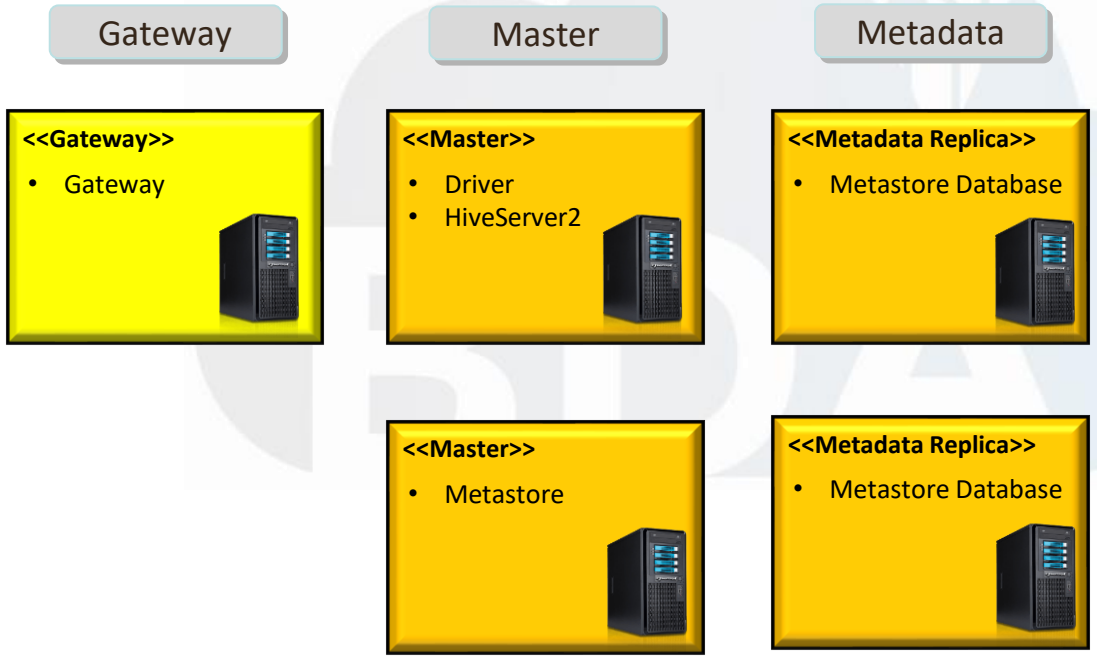


Hive dentro del ecosistema Hadoop



Arquitectura de servicios

Servicios de Hive

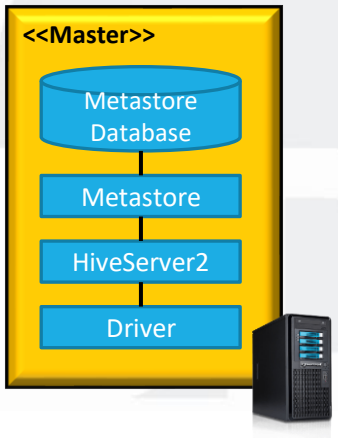


Hive no necesita esclavos, el "driver" se encarga de convertir los queries a procesos MapReduce y le **delega el trabajo a Hadoop**

DESCRIPCIÓN

- **Gateway**: Nodo con la configuración de Hive y acceso a las shells (Hive CLI, Beeline)
- **Driver**: Compila y ejecuta los queries SQL.
- **HiveServer2**: Servicio que permite que múltiples clientes ejecuten queries sobre Hive.
- **Metastore**: Servicio que gestiona los metadatos en Hive.
- **Metastore Database**: Base de datos relacional donde se almacenan los metadatos de Hive, por ejemplo la ruta HDFS asociada a una tabla Hive.

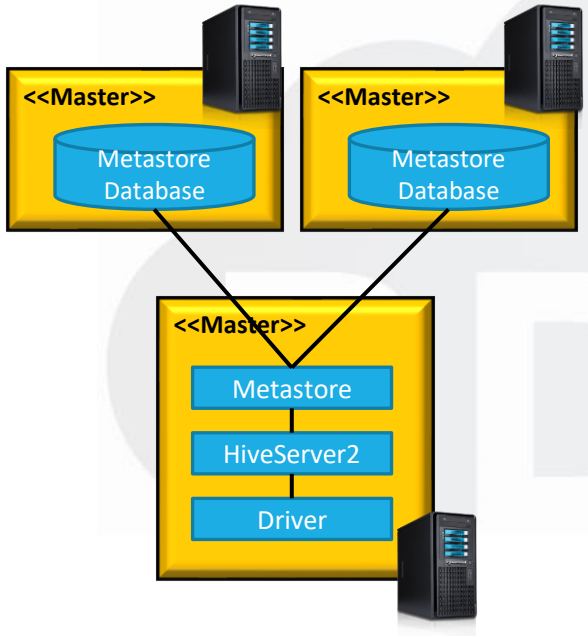
Distribución de los servicios de Hive: Modo Embebido



Todos los servicios están centralizados en un solo nodo. Sólo se recomienda para entornos donde se hacen pequeñas pruebas y nunca para un entorno productivo.

Si este nodo falla, perdemos toda la metadata de Hive y por lo tanto todas las tablas

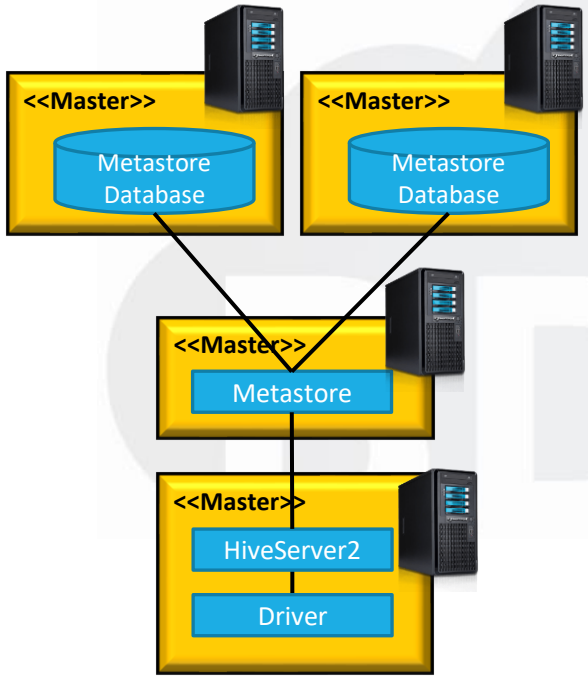
Distribución de los servicios de Hive: Modo Local



La base de datos del metastore está separada del resto de servicios. De esta manera no dependemos de un único punto de falla.

En el modo local las base de datos del Metastore puede configurarse para que trabaje con copias de seguridad.

Distribución de los servicios de Hive: Modo Remoto



Aquí cada servicio se ejecuta sobre su propio nodo.

Es el modo más recomendado de ejecución para un clúster productivo.

Ejercicios teóricos

Ejercicios teóricos

1. ¿Qué es Hive?
2. ¿Cuál es el objetivo fundamental de Hive?
3. ¿Cuáles son las características de Hive?
4. ¿Qué consolas existen en Hive?, ¿Cuál es la actualmente usada?
5. Dibuje la arquitectura de componentes de Hive + Hadoop + Spark
6. ¿Cuáles son los servicios de Hive?
7. ¿Por qué se recomienda separar los sevicios de Hive en nodos físicos diferentes?
8. ¿Cuál es el modo de funcionamiento recomendado y por qué? Dibuje su arquitectura
9. ¿Por qué Hive no tiene nodos esclavos?

Programación básica

Programación básica

Codifiquemos…

Programación avanzada

Programación avanzada

Codifiquemos…

Resumen

Resumen

Hablemos…