Práctica Hive

Vamos a importar un conjunto de datos en formato CSV a Apache Hive y a realizar consultas básicas utilizando HiveQL. Utilizaremos el archivo owid-co2-data.csv, que contiene información detallada sobre las emisiones de CO_2 y otros gases de efecto invernadero a nivel mundial.

Prerrequisitos

Antes de comenzar, asegúrate de contar con lo siguiente:

1. Acceso a un clúster Hadoop con HDFS y Hive instalados.
2. Permisos para acceder a la consola de Hive mediante HiveQL.
3. Conocimiento básico de comandos de terminal y SQL.

Configuración de Hive y Hadoop.

Antes de arrancar los servicios de Hive y Hadoop comprobar los siguientes ficheros

/home/hadoop/hadoop/etc/hadoop/core-site.xml

Escribe en la propiedad `fs.defaultFS` la IP de la máquina virtual, no vale el hostname, ha de ser la IP

```
<configuration>
      <property>
        <name>fs.defaultFS</name>
        <value>hdfs://192.168.253.128:9000</value>
      </property>
      <property>
  <name>hadoop.proxyuser.hadoop.hosts</name>
  <value>*</value>
  <description>Permitir al usuario 'hadoop' realizar impersonación desde
cualquier host</description>
</property>
```

/home/hadoop/apache-hive-4.0.0-bin/conf/hive-site.xml

Comprobar que las siguiente propiedades apuntan a myubuntu o la IP actual de vuestras máquina virtual.

```
<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>
```

Arrancar hadoop:

```
start-all.sh
```

Arrancar hive:

```
hiveserver2 &
```

Accede a la consola de Hive para comenzar a trabajar con HiveQL.

```
beeline -u jdbc:hive2://localhost:10000 -n hadoop
```

O levanta el docker de Hue:

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

Consultar la tabla “employees” que ya está creada:

```
select * from employees;
```

Insertar una nueva fila en “employees”:

```
INSERT INTO employees VALUES (11, 'Paco ', 73000.0);
```

Cada insert genera un fichero nuevo*:

```
hdfs dfs –ls /user/hive/warehouse/employees
```

Cada operación INSERT genera un archivo separado, y esto puede llevar a la creación de múltiples archivos pequeños, lo que es ineficiente para sistemas distribuidos como HDFS. Este problema se puede resolver de dos formas: optimizando las inserciones con un insert múltiple o realizando una operación de agrupación o consolidación de archivos.

1. Usar INSERT INTO con múltiples valores:

En lugar de realizar varias inserciones individuales, puedes insertar todos los registros en una sola operación. Esto evitará la creación de múltiples archivos. Esto generará un solo archivo en lugar de múltiples archivos para cada INSERT.

2. Usar INSERT OVERWRITE:

Puedes usar INSERT OVERWRITE para sobrescribir todos los datos en la tabla y consolidar los archivos en uno solo. Después de varias inserciones, puedes ejecutar una consulta como esta para reorganizar los datos en un solo archivo:

```
INSERT OVERWRITE TABLE employees
SELECT * FROM employees;
```

Esto sobrescribirá los archivos existentes y consolidará los datos en un solo archivo.

3. Ejecutar el comando ALTER TABLE ... CONCATENATE:

Hive ofrece una funcionalidad para fusionar archivos pequeños generados por múltiples INSERTS. Este comando agrupa los archivos pequeños en bloques más grandes. Esto solo funciona con ciertos formatos de archivo, como ORC y PARQUET.

```
ALTER TABLE employees CONCATENATE;
```

Descarga el archivo, por ejemplo, usando wget

```
wget https://github.com/owid/co2-data/raw/master/owid-co2-data.csv
```

Una vez descargado el archivo, súbelo al sistema de archivos distribuido HDFS para que Hive pueda acceder a él.

```
hdfs dfs -mkdir -p /curso/datos/co2
```

Sube el archivo CSV a HDFS

```
hdfs dfs -put owid-co2-data.csv /curso/datos/co2
```

Crear la Tabla externa en Hive a partir de datos ya existentes

A continuación, crea una tabla en Hive que refleje la estructura del archivo CSV. Dado que el archivo tiene múltiples columnas, definiremos cada una con su tipo de datos correspondiente.

```
CREATE DATABASE IF NOT EXISTS bigdata;
USE bigdata;

CREATE EXTERNAL TABLE IF NOT EXISTS owid_co2_data (
    country STRING,
    year INT,
    iso_code STRING,
    population DOUBLE,
    gdp DOUBLE,
    cement_co2 DOUBLE,
    cement_co2_per_capita DOUBLE,
    co2 DOUBLE,
    co2_growth_abs DOUBLE,
    co2_growth_prct DOUBLE,
    co2_including_luc DOUBLE,
    co2_including_luc_growth_abs DOUBLE,
    co2_including_luc_growth_prct DOUBLE,
    co2_including_luc_per_capita DOUBLE,
    co2_including_luc_per_gdp DOUBLE,
    co2_including_luc_per_unit_energy DOUBLE,
    co2_per_capita DOUBLE,
    co2_per_gdp DOUBLE,
    co2_per_unit_energy DOUBLE,
```

```
coal_co2 DOUBLE,
coal_co2_per_capita DOUBLE,
consumption_co2 DOUBLE,
consumption_co2_per_capita DOUBLE,
consumption_co2_per_gdp DOUBLE,
cumulative_cement_co2 DOUBLE,
cumulative_co2 DOUBLE,
cumulative_co2_including_luc DOUBLE,
cumulative_coal_co2 DOUBLE,
cumulative_flaring_co2 DOUBLE,
cumulative_gas_co2 DOUBLE,
cumulative_luc_co2 DOUBLE,
cumulative_oil_co2 DOUBLE,
cumulative_other_co2 DOUBLE,
energy_per_capita DOUBLE,
energy_per_gdp DOUBLE,
flaring_co2 DOUBLE,
flaring_co2_per_capita DOUBLE,
gas_co2 DOUBLE,
gas_co2_per_capita DOUBLE,
ghg_excluding_lucf_per_capita DOUBLE,
ghg_per_capita DOUBLE,
land_use_change_co2 DOUBLE,
land_use_change_co2_per_capita DOUBLE,
methane DOUBLE,
methane_per_capita DOUBLE,
nitrous_oxide DOUBLE,
nitrous_oxide_per_capita DOUBLE,
oil_co2 DOUBLE,
oil_co2_per_capita DOUBLE,
other_co2_per_capita DOUBLE,
other_industry_co2 DOUBLE,
primary_energy_consumption DOUBLE,
share_global_cement_co2 DOUBLE,
share_global_co2 DOUBLE,
share_global_co2_including_luc DOUBLE,
share_global_coal_co2 DOUBLE,
share_global_cumulative_cement_co2 DOUBLE,
share_global_cumulative_co2 DOUBLE,
share_global_cumulative_co2_including_luc DOUBLE,
share_global_cumulative_coal_co2 DOUBLE,
share_global_cumulative_flaring_co2 DOUBLE,
share_global_cumulative_gas_co2 DOUBLE,
share_global_cumulative_luc_co2 DOUBLE,
share_global_cumulative_oil_co2 DOUBLE,
share_global_cumulative_other_co2 DOUBLE,
share_global_flaring_co2 DOUBLE,
share_global_gas_co2 DOUBLE,
```

```
    share_global_luc_co2 DOUBLE,
    share_global_oil_co2 DOUBLE,
    share_global_other_co2 DOUBLE,
    share_of_temperature_change_from_ghg DOUBLE,
    temperature_change_from_ch4 DOUBLE,
    temperature_change_from_co2 DOUBLE,
    temperature_change_from_ghg DOUBLE,
    temperature_change_from_n2o DOUBLE,
    total_ghg DOUBLE,
    total_ghg_excluding_lucf DOUBLE,
    trade_co2 DOUBLE,
    trade_co2_share DOUBLE
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/curso/datos/co2';
```

Notas:

- **EXTERNAL TABLE:** Utilizamos una tabla externa para que Hive no administre físicamente los datos. Los datos permanecen en HDFS incluso si la tabla se elimina.
- **Tipos de Datos:** Se han asignado tipos de datos básicos (STRING, INT, DOUBLE). Puedes ajustar estos tipos según la naturaleza de los datos para optimizar el almacenamiento y las consultas.
- **FIELDS TERMINATED BY ',':** Especifica que el delimitador de campos en el CSV es una coma.

Verificar la Tabla y los Datos

Después de crear la tabla, verifica que Hive la haya reconocido correctamente y que los datos se hayan cargado.

-- Mostrar las primeras 10 filas de la tabla

```
SELECT * FROM owid_co2_data LIMIT 10;
```

-- Contar el número total de registros

```
SELECT COUNT(*) FROM owid_co2_data;
```

Realizar Consultas Básicas con HiveQL

A continuación, se presentan algunos ejemplos de consultas que puedes realizar sobre la tabla owid_co2_data.

a. Obtener las Emisiones Totales de CO_2 por País y Año

```
SELECT country, year, co2
FROM owid_co2_data
WHERE co2 IS NOT NULL
ORDER BY country, year;
```

b. Calcular el Promedio de Emisiones de CO_2 por País

```
SELECT country, AVG(co2) AS promedio_co2
FROM owid_co2_data
WHERE co2 IS NOT NULL
GROUP BY country
ORDER BY promedio_co2 DESC;
```

c. Encontrar los 10 Países con Mayor Emisión de CO_2 en un Año Específico

```
SELECT country, co2
FROM owid_co2_data
WHERE year = 2020
ORDER BY co2 DESC
LIMIT 10;
```

d. Analizar la Relación entre GDP y Emisiones de CO_2 per Cápita

```
SELECT gdp, co2_per_capita
FROM owid_co2_data
WHERE gdp IS NOT NULL AND co2_per_capita IS NOT NULL;
```

Crear una nueva tabla gestionada por Hive

Vamos a crear una tabla gestionada por Hive que almacene los datos en formato **Parquet** a partir de los datos que descargaremos de este CSV:

```
https://ourworldindata.org/economic-inequality#explore-data-on-economic-inequality
```

La tabla se almacenará en formato **Parquet** y luego podrás importar los datos desde el archivo CSV.

```
CREATE TEMPORARY TABLE staging_inequality_data(
    Country STRING,
```

```
    Year INT,
    gini_before_tax FLOAT,
    income_share_richest_10 FLOAT,
    income_share_richest_1 FLOAT,
    income_share_richest_0_1 FLOAT,
    income_share_poorest_50 FLOAT,
    palma_ratio_before_tax FLOAT,
    p0p100_gini_posttax_nat FLOAT,
    s80_s20_ratio_pretax FLOAT,
    p90_p10_ratio_pretax FLOAT,
    p90_p50_ratio_pretax FLOAT,
    p50_p10_ratio_pretax FLOAT,
    palma_ratio_posttax_nat FLOAT,
    s80_s20_ratio_posttax_nat FLOAT,
    p90_p10_ratio_posttax_nat FLOAT,
    p90_p50_ratio_posttax_nat FLOAT,
    p50_p10_ratio_posttax_nat FLOAT,
    -- Añade el resto de las columnas aquí
    median_pretax FLOAT,
    median_posttax_nat FLOAT,
    headcount_ratio_40_median_pretax FLOAT,
    headcount_ratio_50_median_pretax FLOAT,
    headcount_ratio_60_median_pretax FLOAT,
    headcount_ratio_40_median_posttax_nat FLOAT,
    headcount_ratio_50_median_posttax_nat FLOAT,
    headcount_ratio_60_median_posttax_nat FLOAT
)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS
TEXTFILE;
```

Copias el fichero:

```
hdfs dfs -put inequality.csv /curso/datos/
```

Y desde Hive escribes:

```
LOAD DATA INPATH '/curso/datos/inequality.csv' INTO TABLE
staging_inequality_data;
```

Para crear la tabla en formato parquet usa:

```
CREATE TABLE inequality_data STORED AS PARQUET AS SELECT * FROM
staging_inequality_data;
```

Comprobamos los ficheros que sean generado en /user/hive/warehouse/

Crear una nueva tabla particionada

El objetivo de este conjunto de instrucciones es **crear una tabla particionada en Hive**, lo que te permitirá dividir los datos en función de una columna específica, en este caso, el campo `Year`. Al particionar los datos, optimizas su almacenamiento y consulta, especialmente cuando se trabaja con grandes volúmenes de información. También se utiliza el formato **Parquet**, un formato columnar que mejora el rendimiento de las lecturas y reduce el uso de espacio en disco. A continuación, se explican los pasos para lograrlo y el propósito de cada instrucción.

Habilitar particiones dinámicas

```
SET hive.exec.dynamic.partition=true;
SET hive.exec.dynamic.partition.mode=nonstrict;
```

Estas dos líneas configuran Hive para **habilitar las particiones dinámicas**. Esto significa que, al insertar datos en una tabla particionada, no será necesario definir manualmente cada partición. En lugar de eso, Hive creará automáticamente las particiones necesarias en función de los datos que se insertan. Esto es especialmente útil cuando tienes un campo como `Year` que varía entre los registros y no deseas crear las particiones una por una.
La primera línea activa esta capacidad, mientras que la segunda ajusta el modo de particionamiento, permitiendo que las particiones se creen incluso si algunos campos no están completamente especificados. Esto ofrece flexibilidad en cómo se cargan los datos en la tabla.

Crear la tabla particionada

```
CREATE TABLE inequality_data_year (
    Country STRING,
    gini_before_tax FLOAT
)
PARTITIONED BY (Year INT)
STORED AS PARQUET;
```

Aquí estás creando una tabla llamada **`inequality_data_year`** que contiene dos columnas: `Country` y `gini_before_tax`. Además, la tabla está **particionada por `Year`**, lo que significa que los datos se organizarán en directorios separados en HDFS, uno por cada año. Esto mejora la eficiencia de las consultas que se centran en un año específico, ya que Hive solo tendrá que acceder a la partición relevante.
El formato de almacenamiento que utilizamos es **Parquet**, un formato columnar que es eficiente tanto en términos de almacenamiento como en velocidad de lectura. Este formato es ideal cuando trabajas con grandes volúmenes de datos, ya que optimiza las operaciones de análisis y reduce el uso de recursos.

Insertar datos en la tabla particionada

```
INSERT INTO TABLE inequality_data_year PARTITION (Year)
SELECT Country, gini_before_tax, Year
FROM inequality_data WHERE Year IS NOT NULL LIMIT 100;
```

Ahora que tienes la tabla creada, es el momento de **insertar los datos**. La consulta selecciona las columnas `Country`, `gini_before_tax` y `Year` de una tabla existente llamada `inequality_data`. La cláusula `PARTITION (Year)` indica que los datos se dividirán en particiones basadas en el valor del campo `Year`.
Para evitar problemas con datos nulos, hemos añadido un filtro `WHERE Year IS NOT NULL`, asegurándonos de que solo los registros con un valor de año válido se inserten en la tabla. Además, limitamos la cantidad de registros a **100** con la cláusula `LIMIT 100`, lo que permite que la inserción se ejecute más rápido, siendo útil para pruebas o entornos con recursos limitados.